Parallel Computation of Large-Scale Dynamic Optimal Power Flow Problems

Nico Meyer-Huebner, Student Member, IEEE, Abolfazl Mosaddegh, Student Member, IEEE, Michael Suriyah, Member, IEEE, Thomas Leibfried, Member, IEEE, Claudio A. Cañizares, Fellow, IEEE, and Kankar Bhattacharya, Fellow, IEEE

Abstract—In this paper, a novel approach is presented to compute large Dynamic Optimal Power Flow (DOPF) problems in a distributed computing architecture, referred to as a Smart Grid Communication Middleware (SGCM) system. The time horizon is split into shorter time intervals using the SGCM system to solve one iteration of the Primal Dual Interior Point Method (PDIPM) for each sub-problem in parallel. The sub-problems are therefore not solved for optimality, but exchange boundary variables after each iteration. The algorithm was improved by introducing three different stages where the variable exchange is handled differently. The sub-problems eventually converge to a solution near to the solution of the entire problem without solving the global Karush-Kuhn-Tucker (KKT)-conditions. The methodology was tested on the German transmission grid, where the computational effort was reduced significantly. With an entire horizon of 96 time steps and a decomposition into 8 sub-problems, the run-time was decreased from over 2 hours to below 10 minutes with an overall system cost increase of only 1%, which means the DOPF model is suitable for real-time applications.

Keywords—Distributed computing, dynamic optimal power flow, energy storage, model predictive control, optimization, smart grids.

NOMENCLATURE

Indices

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>Intersection between two sub-problems.</td>
</tr>
<tr>
<td>p</td>
<td>Sub-problem.</td>
</tr>
<tr>
<td>t</td>
<td>Time step.</td>
</tr>
</tbody>
</table>

Functions

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Objective function.</td>
</tr>
<tr>
<td>g</td>
<td>Equality constraints.</td>
</tr>
<tr>
<td>h</td>
<td>Inequality constraints.</td>
</tr>
<tr>
<td>(g_{\text{lin}}, g_{\text{nlin}})</td>
<td>Linear and nonlinear equality constraints.</td>
</tr>
<tr>
<td>(\overline{g}<em>{\text{lin}}, \overline{g}</em>{\text{nlin}}, \overline{h}<em>{\text{lin}}, \overline{h}</em>{\text{nlin}})</td>
<td>Time-independent constraints.</td>
</tr>
<tr>
<td>(\underline{g}<em>{\text{lin}}, \underline{g}</em>{\text{nlin}}, \underline{h}<em>{\text{lin}}, \underline{h}</em>{\text{nlin}})</td>
<td>Linear and nonlinear inequality constraints.</td>
</tr>
<tr>
<td>(\mathcal{L})</td>
<td>Lagrangian function.</td>
</tr>
</tbody>
</table>

Parameters

- \(E_{\text{min}}, E_{\text{max}}\): Minimum and maximum storage energy.
- \(n_i\): Number of inequalities.
- \(P\): Number of sub-problems.
- \(P_{\text{min}}, P_{\text{max}}\): Minimum and maximum power of generators.
- \(P_{\text{sc max}}, P_{\text{sd max}}\): Maximum charging and discharging power of storage.
- \(P_{\text{Up}}, P_{\text{Down}}\): Ramping limits up and down.
- \(T\): Total horizon length.
- \(T_p\): Horizon length of one sub-problem.
- \(\Delta t\): Length of time step.
- \(\eta_c, \eta_d\): Energy storage efficiency for charging and discharging.

Variables

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(E_s)</td>
<td>Storage energy.</td>
</tr>
<tr>
<td>(P_g)</td>
<td>Active power output of generators.</td>
</tr>
<tr>
<td>(P_{\text{g min}}, P_{\text{g max}})</td>
<td>Lower and upper limit of generator power at inter-sections.</td>
</tr>
<tr>
<td>(P_{\text{g fix}}, E_{\text{fix}})</td>
<td>Fixed generator power and storage energy at inter-sections.</td>
</tr>
<tr>
<td>(P_{\text{sc}}, P_{\text{sd}})</td>
<td>Storage power output for charging and discharging.</td>
</tr>
<tr>
<td>(x)</td>
<td>Optimization variables.</td>
</tr>
<tr>
<td>(Z)</td>
<td>Slack variables.</td>
</tr>
<tr>
<td>(\alpha, \alpha_0)</td>
<td>Absolute feasibility condition and initial feasibility condition.</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>Relative feasibility condition.</td>
</tr>
<tr>
<td>(\gamma)</td>
<td>IPM perturbation parameter.</td>
</tr>
<tr>
<td>(\sigma)</td>
<td>IPM update coefficient.</td>
</tr>
<tr>
<td>(\Delta E_s)</td>
<td>Additional energy in the storage when charging or discharging.</td>
</tr>
<tr>
<td>(\delta_{E_{\text{Up}}}, \delta_{E_{\text{Down}}})</td>
<td>Absolute gap of storage energy and generator ramping power up and down at inter-sections.</td>
</tr>
<tr>
<td>(\delta_{E_{\text{Up}}}, \delta_{E_{\text{Down}}})</td>
<td>Relative maximal gap of storage energy and generator ramping at inter-sections.</td>
</tr>
<tr>
<td>(\lambda, \mu)</td>
<td>Lagrangian multipliers.</td>
</tr>
</tbody>
</table>

I. INTRODUCTION

OPTIMAL Power Flow (OPF) was introduced by Carpentier in the sixties [1] and has since been used widely in many power system applications [2], [3]. It is possible to find the optimal generation while minimizing the operating costs of the system and respecting network constraints such as voltage or line capacity limits. Nevertheless, secure
network operation becomes more and more challenging with the increasing number of installed Renewable Energy Sources (RESs). The fluctuating in-feed requires more flexibility in generator ramping capacity as well as storing the energy. In order to take into account these restrictions, it is necessary to include inter-temporal constraints and perform a Dynamic Optimal Power Flow (DOPF) based on a forecast of RESs and load profiles. In [4], efficient computation methods for the DOPF problem were presented using Interior Point Methods (IPM), which turned out to be the best suitable solving method for OPF problems. In [5]–[7], the main focus of the DOPF lies in the optimization of demand side management; In [5], the energy demand over a horizon is fixed but the power demand of each time step is optimized by taking into account the price elasticity. In this paper, a DC model for the electrical network is used, and in [6] it is extended to an AC model. Further improvement was achieved when a hydro-electric system model and time-dependent constraints such as water volume limits were included in [7]. Lately, it became popular for active network management in distribution grids [8], [9]. In [8], a framework is laid out to optimize energy storage as well as flexible demand in a distribution grid, and in [9], the DOPF is embedded in an architecture to cope with forecast uncertainty. In these works, the network sizes are usually not too large. Contrarily, when considering inter-connected transmission grids, the problem size grows significantly with increasing time horizon or grid size. A single OPF in a large-scale network is already quite challenging, being a nonlinear and non-convex problem. Thus, there is a need to reduce the computational effort of large-scale DOPF problems. Novel modeling approaches for e.g. energy contracts can reduce the number of non-zero entries in the Karush-Kuhn-Tucker [10] (KKT) system and thus reduce the computational effort [11].

A multi-area decomposition methodology was presented and applied to conventional OPF problems in [12] and the same methodology was later adopted to the DOPF problem in [13]. A different approach to the multi-area DOPF was shown in [14], where the areas are solved to optimality and cooperated strategically. A direct solution method for large-scale DOPF problems is shown in [15], where a constant diagonal perturbation is introduced in the KKT system to utilize the Cholesky factorization and reordering algorithms are integrated in the solver to improve efficiency. The block structure of the KKT system is exploited through decomposition in [16], [17]. Here, for every iteration the global KKT-system is partitioned into sub-blocks, which are solved independently with a direct solver [16] or an iterative solver [17], respectively.

In [12], a multi-area decomposition approach is applied to a single time step OPF. The sub-problems exchange their boundary variables after each Newton step. Similar to [12], this paper proposes a novel decomposition approach applied to the time horizon. The time-dependent variables between the sub-problems are exchanged after each iteration of the Primal-Dual Interior Point Method (PDIPM), which was developed in [18]. The variable exchange process is divided into three stages where boundary conditions are treated from loose to tightened, and the PDIPM acts with different degrees of speed when approaching the original problem. To take advantage of the independence between the sub-problems, the Smart Grid Communication Middleware (SGCM) system is chosen as the preferred platform. It proved to be an efficient distributed computing architecture in [19], [20] and is used to distribute the sub-problems among computing worker-nodes, which calculate each sub-problem in parallel. Hence, the main contributions of the present work are as follows:

- Develop novel decomposition methodology for a DOPF based on the decoupling between inter-temporal constraints.
- Implement distributed computation of the decomposed problem by using the SGCM system.
- Speed up the computation time to make it suitable for real-time applications.

The rest of the paper is structured as follows: in Section II, the general DOPF problem and how it is solved is described. In Section III, the distributed computing architecture, which will be used later for solving the decomposed sub-problems, is presented. In Section IV, the decomposition methodology is described in detail. The results for a realistic case study is shown in Section V, and a conclusion is given in Section VI.

II. Model

A general OPF problem can be described as

\[ \min_{x} F(x) \] (1)

s.t.

\[ g(x) = 0 \] (2)

\[ h(x) \leq 0, \] (3)

where a cost function \( F(x) \) is minimized in (1) while respecting the equality constraints \( g(x) \) of (2) and the inequality constraints \( h(x) \) of (3). In conventional OPF, the equality constraints are the AC load flow equations, while the inequality constraints are the limits on generation active and reactive power output and bus voltages.

A. Inter-temporal Constraints

In addition to the conventional OPF constraints, there are time-dependent constraints, if multiple time steps are optimized. This includes ramp rates of generators and energy storage devices, which are modeled as follows:

1) Generator Ramp Rates: The power output of generators depends on the generated power from one time step earlier. The ramp rates \( P_{\text{Up}}^{\text{Pg}} \) and \( P_{\text{Down}}^{\text{Pg}} \) denote the active power ramp up and down limits and the constraints are given as follows:

\[ P_{\text{Up}}^{\text{Pg}} - P_{\text{Down}}^{\text{Pg}} \]
\[ P_{\text{Up}}^{\text{Pg}} - P_{\text{Down}}^{\text{Pg}} \]

\[ P_{\text{Up}}^{\text{Pg}} - P_{\text{Down}}^{\text{Pg}} \] (4)

\[ P_{\text{Up}}^{\text{Pg}} - P_{\text{Down}}^{\text{Pg}} \] (5)
2) Energy Storage Devices: An energy storage device is considered technology-independent. The efficiency $\eta_c$ for charging and $\eta_d$ for discharging are assumed to be constant. The following energy is extracted from or inserted to the storage device:

$$\Delta E_{s^t} = (\eta_c P_{sc} - \eta_d^{-1} P_{sd}) \Delta t$$

where

$$0 \leq P_{sc} \leq P_{s max}$$

$$0 \leq P_{sd} \leq P_{sd max}$$

Energy capacity of the storage device is crucial for time constrained optimization. The stored energy at time $t$ is calculated as

$$E_{s^t} = E_{s^{t-1}} + \Delta E_{s^t}$$

while

$$E_{s min} \leq E_{s^t} \leq E_{s max}$$

B. Dynamic Optimal Power Flow

In the OPF formulation with inter-temporal constraints, the state vector becomes

$$x = \begin{bmatrix} x^1 \\ \vdots \\ x^t \\ x^T \end{bmatrix}$$

The constraints $g(x)$ in (2) and $h(x)$ in (3) can be split into two types of constraints: nonlinear constraints $g_{lin}(x), h_{lin}(x)$, and linear constraints $g_{lin}(x)$ and $h_{lin}(x)$:

$$g(x) = \begin{bmatrix} g_{lin}(x) \\ h_{lin}(x) \end{bmatrix}$$

Further, the constraints can be divided into groups of time-independent constraints $\bar{g}(x)$ and $\bar{h}(x)$, as well as constraints depending on the previous time step $\tilde{g}(x)$ and $\tilde{h}(x)$. The nonlinear constraints are entirely time-independent and are therefore expressed as:

$$g_{lin}(x) = \begin{bmatrix} \bar{g}_{lin}(x^1) \\ \vdots \\ \bar{g}_{lin}(x^t) \\ \tilde{g}_{lin}(x^T) \end{bmatrix}$$

$$h_{lin}(x) = \begin{bmatrix} \bar{h}_{lin}(x^1) \\ \vdots \\ \bar{h}_{lin}(x^t) \\ \tilde{h}_{lin}(x^T) \end{bmatrix}$$

The linear constraints may contain time-dependent equations:

$$g_{lin}(x) = \begin{bmatrix} \bar{g}_{lin}(x^1) \\ \vdots \\ \bar{g}_{lin}(x^T) \end{bmatrix}$$

$$h_{lin}(x) = \begin{bmatrix} \bar{h}_{lin}(x^1) \\ \vdots \\ \bar{h}_{lin}(x^T) \end{bmatrix}$$

The state vector $x^0$ describes the initial state for inter-temporal constraints like the State of Charge (SOC) of storage or the power output of generators with ramping limits.

The costs of each time step only depend on the generated power and the total costs sum up to

$$F(x) = \sum_{t=1}^{T} F^t(x^t)$$

The cost function can be a piecewise linear or polynomial function.

C. Solving the Problem

The problem is solved with a PDIPM, assigning a Lagrangian multiplier to each constraint. The equality constraints are represented by $\lambda$, and the inequality constraints by $\mu$. To include the inequalities in a function which shall be minimized, slack variables $Z > 0$ are introduced. With $h(x) + Z = 0$, the inequalities are transformed to equality functions and the Lagrangian is formed to

$$L(x, \lambda, \mu, Z) = F(x) + \mu^T(h(x) + Z) + \lambda^T g(x)$$

In the PDIPM, the slack variables are weighted with the barrier coefficient $\gamma$ to keep the inequality constraints away from zero in the first iterations. Eventually, $\gamma$ is reduced to zero and the problem approaches the original one. The Lagrangian is built up to be minimized, while maximizing the multipliers. This leads to a minimization of the cost function $F$, while $h(x)$ is kept below and $g(x)$ is close to zero. Common Newton steps are applied to solve the KKT-conditions [10]. The main effort lies in solving the linear system

$$\begin{bmatrix} M & g^T_x \\ g_x & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = -N$$

with

$$M = \nabla^2_x L(x, \lambda, \mu, Z) + h^T_x[Z]^{-1}[\mu]h_x$$

$$N = \nabla_x L(x, \lambda, \mu, Z) + h^T_x[Z]^{-1}(\gamma e + [\mu]h(x))$$

The perturbation parameter $\gamma$ must converge to zero in order to satisfy the first order optimality conditions of the original problem. It is updated with the rule

$$\gamma \leftarrow \sigma \frac{Z^T \mu}{n_1}$$
where \( n_i \) is the number of inequalities and \( \sigma \) is a scalar constant which is set between 0.1 and 0.3 in this paper. It indicates the speed of decreasing \( \gamma \) and therefore, the speed of approaching the original problem. Typically, \( \sigma = 0.1 \) is a good trade-off between convergence speed and robustness [18]. Note that \([A]\) creates a diagonal matrix with the vector \( A \) on its diagonal; \( e \) in (19) is a vector of ones; the Jacobian and the Hessian of a function \( f \) with respect to \( x \) is given by \( f_x \) and \( f_{xx} \), respectively. It is therefore necessary to provide first and second order derivatives of the cost function and all constraints with respect to the optimization variables \( x \). A detailed description of the PDIPM can be found in [18].

III. DISTRIBUTED COMPUTING ARCHITECTURE

Similar to [19], [20], the distributed computing architecture based on the SGCM system is presented in this section. In order to reduce the run-time of the DOPF model, the SGCM system [19], [20], a framework for a “MapReduce” model, is employed as a platform for a parallel computing approach. The SGCM uses proprietary protocols on top of Transmission Control Protocol (TCP)/ Internet Protocol (IP) for inter-communication of computing components. The core components of the SGCM system are the console and the nodes. The console is a client program that is responsible for application deployment, installation of master and worker applications, execution, and management of the model. The computing nodes are the worker-nodes and the master-node. These nodes are able to make TCP connections to other nodes, and also they wait for TCP connections to provide services. The nodes are responsible to execute groups of instructions, in order to deploy, call, bundle, transfer, execute, and manage files.

The main steps of the MapReduce model are “Map” and “Reduce”. The master-node divides the data into smaller sub-problems based on the work-flow logic of the algorithm, and maps those sub-problems amongst related worker-nodes. When the worker-nodes finish the runs, the master-node collects the results from entire worker-nodes, analyzes the data based on the algorithm, and reduces/combines the data into an output format. Since each worker-node operates independently, the problem is simultaneously run as different sub-problems, and the computational cost is reduced considerably. If some modifications are needed after Map operation, a “Shuffle” mid-stage is introduced in the MapReduce model to sort or exchange data between worker-nodes, and finally, prepare the data for Reduce operation. If the system has a tree structure, some worker-nodes act as new master-nodes for their sub-problems.

The proposed distributed computing approach is executed on a server based on Windows 64-bit operating system with two E5-2670 v3 Intel Xeon 2.30 GHz processors and 128 GB RAM with 48 cores. The computing nodes with different port addresses and the console are installed in this server.

IV. DECOMPOSITION METHODOLOGY

In Fig. 1, the flowchart of the proposed methodology is shown. First, the problem is split into the sub-problems (Section IV-A) and each sub-problem gets initialized for the PDIPM. Then, the sub-problems are distributed among the worker-nodes of the SGCM system [19], [20], which was described in Section III. Each worker-node performs one iteration of the PDIPM including an update for the new search direction. The update is saved in the worker-node and the boundary variables, i.e. storage energy levels and power output of ramping constrained generators at the beginning and at the end of the sub-problem, are returned to the master-node. The master-node checks the progress of the optimization and the boundary variables are exchanged or modified depending on the current stage (Section IV-B). The sub-problems with the new boundary conditions are distributed again amongst worker-nodes, and the next iteration starts. Once all sub-problems have converged, the master-node collects and prints the final results.

A. Decomposition

If the original problem has a number of \( T \) time steps, \( T \) must be divisible without remainder by the number of sub-problems \( P \). Therefore, \( T = P \times T_p \), where the horizon of one sub-problem, i.e. the number of time steps, is \( T_p \). In Fig. 2, the exemplary decomposition of the solution vector \( x \) is shown.
The linear equality constraints from (14) are decomposed into

\[
g_{\text{lin},1}(x_1) = \begin{bmatrix} g_{\text{lin},1}(x_1^1) \\ \vdots \\ g_{\text{lin},P}(x_TP) \\ g_{\text{lin},1}(x_1^P) \\ \vdots \\ g_{\text{lin},P}(x_{P+1}^P) \end{bmatrix} = \begin{bmatrix} g_{\text{lin},1}(x_1^1) \\ \vdots \\ g_{\text{lin},1}(x_1^P) \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix} \tag{21} \]

and the linear inequality constraints become

\[
h_{\text{lin},1}(x_1) = \begin{bmatrix} h_{\text{lin},1}(x_1^1) \\ \vdots \\ h_{\text{lin},p}(x_{P+1}^P) \\ h_{\text{lin},1}(x_1^P) \\ \vdots \\ h_{\text{lin},p}(x_{P+1}^P) \end{bmatrix} = \begin{bmatrix} h_{\text{lin},1}(x_1^1) \\ \vdots \\ h_{\text{lin},1}(x_1^P) \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix} \tag{22} \]

The lower case denotes the number of the sub-problem and the upper case denotes the time step. All variables of the first time step in sub-problem \(p\) are therefore collected in \(x_1^1\). It includes the same variables as \(x^{(P+1):P+1}\) of the original vector.

The nonlinear equality constraints from (13) are decomposed into \(P\) new sets of constraints, one for each sub-problem:

\[
g_{\text{lin},1}(x_1) = \begin{bmatrix} g_{\text{lin},1}(x_1^1) \\ \vdots \\ g_{\text{lin},1}(x_1^P) \\ \vdots \\ g_{\text{lin},p}(x_{P+1}^p) \\ \vdots \\ g_{\text{lin},p}(x_{P+1}^p) \end{bmatrix} = \begin{bmatrix} g_{\text{lin},1}(x_1^1) \\ \vdots \\ g_{\text{lin},1}(x_1^P) \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix} \tag{24} \]

and the linear inequality constraints become

\[
h_{\text{lin},1}(x_1) = \begin{bmatrix} h_{\text{lin},1}(x_1^1) \\ \vdots \\ h_{\text{lin},p}(x_{P+1}^p) \\ \vdots \\ h_{\text{lin},p}(x_{P+1}^p) \end{bmatrix} = \begin{bmatrix} h_{\text{lin},1}(x_1^1) \\ \vdots \\ h_{\text{lin},1}(x_1^P) \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix} \tag{25} \]

\[\text{B. Different stages}\]

In the following, the three stages of different interaction among sub-problems are explained:

1) Stage I: The boundaries are set loose and the sub-problems are therefore decoupled from each other. This enables them to evolve into a direction which is favorable for each specific sub-problem and decreases the dependency on the starting point. The IPM-parameter Sigma is set to \(\sigma = 0.3\), in order not to converge too quickly to a solution. The resulting feasibility condition \(\alpha_p\) is determined for each sub-problem \(p\).

\[
\hat{\alpha}_p = \frac{\max (\max |h_p(x_p)|, |g_p(x_p)|)}{1 + \max (||\lambda_p||_\infty, ||\mu_p||_\infty)} \tag{26} \]

\[
\alpha_p = \frac{\hat{\alpha}_p}{\alpha_{0,p}} \tag{27} \]

The variable \(\hat{\alpha}_p\) is calculated in every iteration, while \(\alpha_{0,p}\) is only calculated once in iteration 0, it is therefore depending on the starting point.
2) **Stage II:** Once the feasibility condition has evolved to $\alpha_p < 0.1$ for all sub-problems, Stage II is reached. Now the exchange of boundary condition starts and Sigma is further reduced to $\sigma = 0.2$. The master-node considers the boundaries always intersection-wise. For an intersection between sub-problems $p$ and $p+1$, the sub-problem $p$ receives the following constraints for the next iteration:

\[
E_s^{1,p+1} - \Delta E_s^{1,p+1} \leq E_s^{T,p} \leq E_s^{1,p+1} - \Delta E_s^{1,p+1} \tag{30}
\]

\[
P_g^{1,p} \leq P_g^{T,p} \leq \overline{P}_g^{p} \tag{31}
\]

with:

\[
\overline{P}_g^{p} = \max \left( \min \left( P_g^{1,p+1} + R_{pG}^{Up}, P_g^{\text{max}} \right), P_g^{\text{min}} \right) \tag{32}
\]

\[
\overline{P}_g^{T} = \min \left( \max \left( P_g^{1,p+1} - R_{pG}^{\text{Down}}, P_g^{\text{min}} \right), P_g^{\text{max}} \right) \tag{33}
\]

The sub-problem $p + 1$ receives the following constraints:

\[
E_s^{T,p} \leq E_s^{1,p+1} - \Delta E_s^{1,p+1} \leq E_s^{T,p} \tag{34}
\]

\[
P_g^{1,p+1} \leq P_g^{1,p+1} \leq \overline{P}_g^{p+1} \tag{35}
\]

with:

\[
\overline{P}_g^{p+1} = \max \left( \min \left( P_g^{1,p+1} + R_{pG}^{Up}, P_g^{\text{max}} \right), P_g^{\text{min}} \right) \tag{36}
\]

\[
\overline{P}_g^{T} = \min \left( \max \left( P_g^{1,p+1} - R_{pG}^{\text{Down}}, P_g^{\text{min}} \right), P_g^{\text{max}} \right) \tag{37}
\]

Equations (30), (31), and (35) are already implemented in the conventional model and only need a change of limits. Contrarily, (34) needs to be added to the model explicitly. For easier computation, the constraint is set from the beginning, only in Stage I the limits are the standard maximum and minimum energy values for the respective storages. The gap between the time-dependent variables on both sides of the intersection $\text{int}$ is calculated as

\[
\hat{\delta}_E,\text{int} = |E_s^{T,p} - (E_s^{1,p+1} - \Delta E_s^{1,p+1})| \tag{38}
\]

\[
\hat{\delta}_p,\text{int} = P_{g,p+1}^{1} - P_g^{T,p} - R_{pG}^{Up} \tag{39}
\]

\[
\hat{\delta}_{\text{Down}},\text{int} = P_g^{T,p} - P_{g,p+1}^{1} - R_{pG}^{\text{Down}} \tag{40}
\]

The convergence criteria at the intersections are chosen to

\[
\hat{\delta}_p,\text{int} = \max \left( \hat{\delta}_p,\text{int}, \hat{\delta}_{\text{Down}},\text{int} \right) \tag{41}
\]

\[
\hat{\delta}_E,\text{int} = \max \left( \hat{\delta}_E,\text{int}, \hat{\delta}_{\text{Down}},\text{int} \right) \tag{42}
\]

If $\delta_p,\text{int}$ and $\delta_E,\text{int}$ are below 0.03, the boundaries can be fixed. This means that a deviation of 3% between two sub-problems is considered as a satisfying boundary condition. It is a trade-off between algorithm robustness (larger value) and optimality (smaller value). The mean value between the two sub-problems is calculated. The generators are limited to half the ramping limit in order to keep the maximal possible flexibility while satisfying the boundary condition.

\[
E_{s\text{fix, int}} = \max \left( \frac{1}{2} (E_{s\text{p}}^T + E_{s\text{p+1}}^1 - \Delta E_{s\text{p+1}}^1), 0 \right) \tag{43}
\]

\[
\overline{P}_{g\text{fix, int}} = P_{g\text{fix, int}} + \frac{1}{4} (R_{pG}^{Up} + R_{pG}^{\text{Down}}) \tag{44}
\]

\[
\overline{P}_{g\text{fix, int}} = P_{g\text{fix, int}} - \frac{1}{4} (R_{pG}^{Up} + R_{pG}^{\text{Down}}) \tag{45}
\]

with:

\[
P_{g\text{fix, int}} = \max \left( \min \left( \frac{1}{2} (P_g^T + P_{g,p+1}^1), P_g^{\text{max}} \right), P_g^{\text{min}} \right) \tag{46}
\]

The constraints are set to:

\[
E_{s\text{fix, int}} \leq E_s^T \leq E_{s\text{fix, int}} \tag{47}
\]

\[
P_{g\text{fix, int}} \leq P_g^T \leq \overline{P}_{g\text{fix, int}} \tag{48}
\]

\[
E_{s\text{fix, int}} \leq E_{s,p+1}^1 - \Delta E_{s,p+1}^1 \leq E_{s\text{fix, int}} \tag{49}
\]

\[
P_{g\text{fix, int}} \leq P_{g,p+1}^1 \leq \overline{P}_{g\text{fix, int}} \tag{50}
\]

3) **Stage III:** Once all intersection have fixed boundaries, the final stage starts. Now, Sigma is set to the standard OPF value of $\sigma = 0.1$ and each worker can compute the final solution of the respective sub-problem independently. After all sub-problems have converged and the master-node collects the results, there is no more communication needed between the worker-nodes and the master-node.

V. RESULTS

The methodology is tested on a realistic model of the German transmission grid with over 1200 buses and 96 time steps with intervals of 1 hour. A more detailed description of the network model can be found in [21]. First, the results with the presented methodology is compared to the solution of the original problem. Then, the computational effort and specific parameters concerning the decomposition methodology are shown.

A. Comparison of solutions

To compare the solutions of the decomposed problems with the original problem, the crucial time coupling of the energy storages are considered. In Fig. 3, the accumulated energy of all storages in the grid is shown for the original problem and for a decomposition into 4 sub-problems. The inter-sections between the sub-problems are at hours 24, 48 and 72. It can be seen, that the deviation is almost negligible and the decomposed problem calculates a very similar storage operation. The total costs are therefore only slightly affected and increase by 0.4%, which can be seen in Fig. 4 and Table I. If the number of sub-problems is increased to 16, the deviation in storage operation becomes significant (Fig. 5). Each sub-problem only has an optimization horizon of 6 hours and the exchange of boundary variables is not sufficient to cope with the loss of information. Yet, secure operation is achieved and the cost increase is still relatively small with under 3%.
### B. Computational results

The Newton Step for a horizon of 96 time steps involves a KKT-system with the dimension of almost $1e6 \times 1e6$ and takes in average over 2 minutes, while it only takes 2 seconds for 6 time steps in the case of 16 sub-problems (Table I). This leads to an outstanding improvement in computation time when decomposing the problem. The number of iterations increases slightly for a smaller number of sub-problems. Note that the slowest sub-problem defines the overall time. The latter can be reduced from over 2 hours to a few minutes in the case of 8 sub-problems (Fig. 6), while allowing a cost increase of less than 1%. When the Newton Step is calculated within 2-3 seconds for more than 8 sub-problems, the overall time cannot further be reduced significantly because the communication overhead now plays an important role. Hence, based on the obtained results shown in Table I, 8 would be the optimal number of sub-problems, which can considerably reduce the run-time of the model and is closely enough to the cost of centralized approach with 1 sub-problem.

In Fig. 7, the storage energy gap from (42) is shown for all 3 inter-sections when decomposing into 4 sub-problems. It shows the development through the three stages described in Section IV-B. After the first stage, a gap of over 50% of the installed capacity can be seen, but due to the exchange of boundary variables between the sub-problems, it is reduced to below 3% after 15 iterations. After the boundaries are fixed, it converges to zero.

In Fig. 8, the relative feasibility condition in (29) for all 4 sub-problems is shown. It is quickly reduced to below 0.1, when the second stage starts. It is then notable, that the feasibility is not continuously decreasing. This is due to the fact that the boundary constraints are changing in every iteration and the search direction is a priori not exact. Only

<table>
<thead>
<tr>
<th># Sub-problems</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>12</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newton Step [s]</td>
<td>124</td>
<td>39</td>
<td>14</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td># Iterations</td>
<td>56</td>
<td>37-49</td>
<td>40-44</td>
<td>34-43</td>
<td>32-41</td>
<td>30-38</td>
</tr>
<tr>
<td>Wall clock [min]</td>
<td>120</td>
<td>39</td>
<td>16</td>
<td>9</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>Costs [M€]</td>
<td>70.61</td>
<td>70.69</td>
<td>70.91</td>
<td>71.24</td>
<td>72.31</td>
<td>72.58</td>
</tr>
<tr>
<td>Relative costs [%]</td>
<td>100.0</td>
<td>100.1</td>
<td>100.4</td>
<td>100.9</td>
<td>102.4</td>
<td>102.8</td>
</tr>
</tbody>
</table>
when reaching the last stage where the boundaries are fixed and the constraints are not changing anymore, the feasibility is continuously decreasing.

VI. CONCLUSION

A novel methodology was presented to decouple large DOPF problems into sub-problems to compute them efficiently in parallel using an SGCM system. The three stage variable exchange process was described in detail and simple to implement. It was tested on a realistic model of the German transmission grid for 96 time steps. The results showed that by implementing the DOPF model in a distributed computing platform, the run-time was considerably decreased, which is needed for dynamic studies of large-scale networks and real-time applications. A trade-off between decreased computation time and increased system costs must be taken into account. For instance in our case study, a computation time reduction by factor 13 was bought with a cost increase of 0.9% by choosing the number of 8 sub-problems.

ACKNOWLEDGMENT

The authors would like to thank Dr. Hongbing Fan from Wilfrid Laurier University, Waterloo, ON, Canada, for help and support with the implementation of the model on the SGCM system.

REFERENCES