Power System Dynamic Simulations using a Parallel Two-level Schur-complement Decomposition

Petros Aristidou, *Member, IEEE,* Simon Lebeau, and Thierry Van Cutsem, *Fellow Member, IEEE*

***Abstract*—As the need for faster power system dynamic simulations increases, it is essential to develop new algorithms that exploit parallel computing to accelerate those simulations. This paper proposes a parallel algorithm based on a two-level, Schur-complement-based, domain decomposition method. The two-level partitioning provides high parallelization potential (coarse- and fine-grained). In addition, due to the Schur-complement approach used to update the sub-domain interface variables, the algorithm exhibits high global convergence rate. Finally, it provides significant numerical and computational acceleration. The algorithm is implemented using the shared-memory parallel programming model, targeting inexpensive multi-core machines. Its performance is reported on a real system as well as on a large test system combining transmission and distribution networks.**

***Index Terms*—domain decomposition methods, Schur-complement, power system dynamic simulation, OpenMP, shared-memory**

I. Introduction

OVER the last decades, power system phasor simulations have become indispensable in the planning, design, operation, and security assessment of power systems.

Often, simulation speed requirements dictate the type of simulation used. Power flow simulations are the fastest, but rely on simplifying assumptions. They focus on a long-term equilibrium point, they don't give useful indications in infeasible cases and cannot take into consideration controls depending on the system time evolution. As system time evolution becomes critical to ensure system security, these simulations are not sufficient. Quasi Steady State (QSS) simulations, which consist of replacing the short-term dynamics with adequate equilibrium relations [1], have been also used to analyze the long-term voltage and frequency dynamics of power systems. QSS simulations are very fast but also have obvious limitations: i) they assume that the neglected short-term dynamics are stable, and ii) the sequence of discrete controls may not always be correctly identified from the simplified QSS models.

Detailed dynamic simulations do not suffer from these limitations but have been restricted to off-line calculations because of their computational burden. However, today the technology is mature and proper algorithms exist permitting the use of detailed dynamic simulations in real-time or close to real-time applications [2]. Thus, the target of this work is to deal with a unique, detailed, dynamic model, relevant to angle, frequency, and voltage stability studies, both in short- and long-term, while keeping the performance to the maximum so as to increase productivity.

Under the phasor approximation, complex electric components (like generators, motors, loads, wind turbines, etc.) used in dynamic simulations are represented by systems of stiff, non-linear Differential and Algebraic Equations (DAEs) [3]. At the same time, the network is represented by a system of linear algebraic equations. Together, they form an initial value DAE model describing the physical dynamic characteristics, interactions, and controls of the system. A large power system may involve tens of thousands of such equations whose dynamics span over very different time scales and undergo many discrete transitions imposed by limiters, switching devices, etc. Consequently, dynamic simulations are challenging to perform and computationally intensive [2].

A. Domain Decomposition Methods

The most prominent parallel algorithms used in power system dynamic simulations are inspired from the field of Domain Decomposition Methods (DDMs). DDM refers to a collection of techniques which revolve around the principle of "divide-and-conquer". They were primarily developed for solving large boundary value problems of Partial Differential Equations by splitting them into smaller problems over sub-domains and iterating between them to coordinate the solution [4]. Because of their high efficiency and performance, DDMs became popular in DAE problems, such as those appearing in power system dynamic simulations.

The first to envisage an application to power systems was probably Kron with the *diakoptics* method [5]. This technique belongs to the family of *parallel-in-space* methods: the formulation of the problem at a single time instant is decomposed and solved. Another family of methods is the *parallel-in-time* [6]–[8]. Despite the sequential character of the initial value problem, which stems from the discretization of differential equations, these techniques propose the solution of multiple consecutive time instants in parallel. Another well-known family is the *Waveform Relaxation* (WR) [9], [10]. These techniques decompose the system in space and solve each sub-domain for several time instances (time window interval) before exchanging the interface values. Hence, these values consist of waveforms from neighboring sub-systems (i.e. a sequence of interface values over a number of consecutive

Simon Lebeau is with the TransEnergie Division of Hydro-Québec, Canada, e-mail: Lebeau.Simon@hydro.qc.ca.

Petros Aristidou is with the Power System Laboratory, ETH Zürich, Switzerland. This work was performed while the author was at the Dept. of Elec. Eng. and Comp. Science, University of Liège, Belgium, e-mail: p.aristidou@ieee.org.

Thierry Van Cutsem is with the Fund for Scientific Research (FNRS) at the Dept. of Elec. Eng. and Comp. Science, University of Liège, Belgium, e-mail: t.vancutsem@ulg.ac.be.

time instants). After each solution, waveforms are exchanged between neighboring sub-systems, and this process is repeated until global convergence. If the time interval of the WR is restricted to only one time step, then the Instantaneous Relaxation method [11] is formulated. These techniques provide significant acceleration, however, their convergence strongly depends on the partitioning and on the time window interval [12], [13]. Some techniques to accelerate them have been proposed in [12], [14].

Several parallel DDMs have been proposed in the past for phasor simulation, but have not been widely used. Only recently have commercial software appeared that can achieve high performance, real-time simulations combining state-of-the-art hardware and algorithms, such as ePHASORsim [15] for instance. While these commercial, real-time simulators offer increased performance, advanced input/output/management features, hardware-in-the-loop facilities, and support, they are bound to specific computing platforms on which the software is optimized for highest performance. The objective of this work is to propose a general-purpose high-performance parallel algorithm that can be executed on cheap, multicore computers.

B. Contributions of paper

The work presented in this paper is a continuation of [16]. In the latter, a parallel, DDM-based algorithm was proposed to accelerate power system dynamic simulations. However, the aforementioned algorithm involves a sequentially treated part whose processing increases linearly with the network size. Thus, when the latter increases compared to the overall, so does the sequential portion, hence impeding the parallel performance of the algorithm. This becomes very important when dealing with large, combined transmission, sub-transmission, and distribution systems.

This paper proposes a two-level Schur-complement-based DDM to overcome this sequentiality problem and to accelerate dynamic simulations. First, a non-overlapping, topologically-based, decomposition scheme is applied to the electric network revealing a star-shaped sub-domain layout. This first decomposition is reflected to a separation of the DAEs, which are projected onto the sub-domains. Next, a second decomposition scheme is applied within each sub-domain, splitting the sub-domain network from the components connected to it. This second decomposition further partitions the system DAEs. Finally, all sub-systems are solved hierarchically, using a Very DisHonest Newton (VDHN) method, and their interface variables are updated with a Schur-complement approach [17], at each decomposition level.

The two-level partitioning provides high parallelization potential. In addition, due to the Schur-complement approach used to update the interface variables, the algorithm exhibits high global convergence rate that is not affected by the partition selection. Moreover, the simulation performance is augmented in two ways. First, the locality of the decomposed sub-systems is exploited to avoid many unnecessary computations and provide numerical acceleration. Second, the independent calculations of the sub-systems are parallelized. The proposed method belongs to the family of parallel-in-space methods with a Schur-complement approach to treat the interface variables. A more complete classification of DDMs for power system dynamic simulations can be found in [18].

In general, two-level, or multilevel, DDMs can be exploited when the parallelization potential is high enough and the structure of the underlying model is suitable. Such examples can be found in other power system applications [19], [20] and in other engineering fields [21]–[23].

The paper is organized as follows. In Section II the power system model formulation and the two-level decomposition are presented. The proposed algorithm is detailed in Section III. In Section IV, numerical acceleration techniques are presented, followed in Section V by parallel programming and implementation aspects. Next, simulation results obtained on a detailed model of the Hydro-Québec (HQ) system are reported in Section VI and on a combined Transmission and Distribution system in Section VII. A summary of the main features is offered in the closing section.

II. Power system decomposition

An electric power system, under the quasi-sinusoidal (or phasor) approximation, can be described in compact form by the following DAE initial value problem [24], [25]:

$$\mathbf{0} = \boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{V}) \tag{1a}$$

$$\boldsymbol{\Gamma}\dot{\boldsymbol{x}} = \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{V}) \tag{1b}$$

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0, \boldsymbol{V}(t_0) = \boldsymbol{V}_0 \tag{1c}$$

where $\boldsymbol{V}$ is the vector of rectangular components of network voltages (i.e. $\boldsymbol{V} = [v_{x1}, v_{y1}, v_{x2}, v_{y2}, \ldots]^T$), $\boldsymbol{x}$ is the expanded state vector containing the differential and algebraic variables (except the voltages) of the system and $\boldsymbol{\Gamma}$ is a diagonal matrix with:

$$(\boldsymbol{\Gamma})_{\ell\ell} = \begin{cases} 0 & \text{if the } \ell\text{-th equation is algebraic} \\ 1 & \text{if the } \ell\text{-th equation is differential} \end{cases} \tag{2}$$

Equation (1a) corresponds to the purely algebraic network equations:

$$\mathbf{0} = \boldsymbol{D}\boldsymbol{V} - \boldsymbol{I} \triangleq \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{V}) \tag{3}$$

where $\boldsymbol{D}$ includes the real and imaginary parts of the bus admittance matrix and $\boldsymbol{I}$ the rectangular components of the bus currents. The latter are included in the differential-algebraic states, i.e. $\boldsymbol{x} = [i_y, i_x, \ldots]^T$.

Equation (1b) describes the remaining DAEs of the power system components connected to the network and their controls. For reasons of simplicity, all the components connected to the network that either produce or consume power (such as power plants, distributed generators, induction motors, loads, etc.) are referred to as *injectors* [16].

Equations (1) change during the simulation due to the discrete dynamics, such as response to discrete controls or changes in continuous-time equations. For reasons of simplicity, the handling of these discrete events is not presented in this paper. More details concerning their handling can be found in [26] and its quoted references.

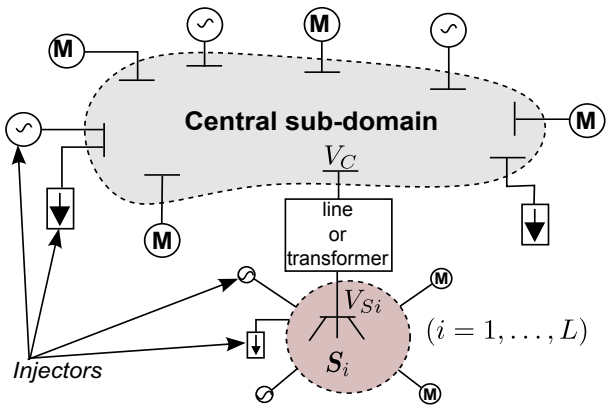

Figure 1. Proposed two-level power system partition

A. First level of decomposition: Network

An important step in developing a DDM is the identification of the partition scheme to be used. In this work, a star-shaped, topologically-based scheme has been chosen. The power system network is decomposed into a Central (C) and several Satellite sub-domains (S_i) each connected to the Central at one bus as shown in Fig. 1.

Good candidate systems with this topology are Transmission Networks (TN) with detailed representation of their sub-transmission or distribution systems. The only requirement is that each sub-system needs to be attached to the TN at a single bus. One such example is the HQ system used in this work. Its structure with radially connected sub-transmission systems (referred to as distribution in some countries) is a good candidate for this decomposition.

Combined transmission and distribution systems is another example of systems with this structure. The most noticeable developments in power systems involve Distribution Networks (DNs), which are called upon to actively support the TN with an increasing number of Distributed Generators (DGs) and flexible loads participating in ancillary services. Modeling the latter in detail, along with the bulk transmission grid, poses a great computational challenge. Thus, the proposed two-level DDM algorithm can be applied to accelerate their dynamic simulation procedure.

The decomposition can rely on the voltage levels (e.g. based on the sub-transmission/distribution transformers), or it can be obtained from an analysis of the network graph. One such graph-based method has been developed in [18]: the Python NetworkX [27] package is used to recursively merge the nodes with only one connection (leaves of the graph) with their neighbors, until no more leaf is left. Then, the satellite sub-graphs, connected to the remaining graph through a single branch, are identified using an exhaustive examination of the connecting lines and isolation of the satellite sub-domains.

The described automatic partition algorithm has a high complexity and can prove very costly for large networks (although it is executed only once for each network topology), thus the voltage level based method is preferred when possible. The latter approach also allows to retain a physical meaning of the sub-domains (sub-transmission, distribution, etc.).

Let the system be decomposed as described above into the Central and L Satellite sub-domains along with their injectors, as sketched in Fig. 1. This decomposition is reflected on Eq. (1) as follows.

The DAE system describing the Central sub-domain with its injectors becomes:

$$\begin{aligned} \mathbf{0} &= \boldsymbol{g}_C(\boldsymbol{x}_C, \boldsymbol{V}_C, \boldsymbol{V}_{S1}, \ldots, \boldsymbol{V}_{SL}) \\ \boldsymbol{\Gamma}_C \dot{\boldsymbol{x}}_C &= \boldsymbol{\Phi}_C(\boldsymbol{x}_C, \boldsymbol{V}_C) \end{aligned} \tag{4}$$

and, for the i-th Satellite sub-domain ($i = 1, \ldots, L$):

$$\begin{aligned} \mathbf{0} &= \boldsymbol{g}_{Si}(\boldsymbol{x}_{Si}, \boldsymbol{V}_{Si}, \boldsymbol{V}_C) \\ \boldsymbol{\Gamma}_{Si} \dot{\boldsymbol{x}}_{Si} &= \boldsymbol{\Phi}_{Si}(\boldsymbol{x}_{Si}, \boldsymbol{V}_{Si}) \end{aligned} \tag{5}$$

where $\mathbf{x}_C$, $\boldsymbol{x}_{Si}$, $\boldsymbol{V}_C$, $\boldsymbol{V}_{Si}$, $\boldsymbol{\Gamma}_C$, and $\boldsymbol{\Gamma}_{Si}$ are the projections of $\mathbf{x}$, $\boldsymbol{V}$, and $\boldsymbol{\Gamma}$, defined in (1), on the Central and Satellite sub-domains respectively.

The DAE systems (4) and (5) are coupled only through the voltage variables involved in the equations of the network elements (lines, transformers, etc.) connecting the sub-domains, and are completely equivalent to the original system (1).

B. Second level of decomposition: Injectors

A second level of decomposition is applied that separates the injectors from the sub-domain network as described in [16]. For example, the j-th injector connected to the Central sub-domain (see Fig. 1) is described by:

$$\boldsymbol{\Gamma}_{Cj} \dot{\boldsymbol{x}}_{Cj} = \boldsymbol{\Phi}_{Cj}(\boldsymbol{x}_{Cj}, \boldsymbol{V}_C) \tag{6}$$

where $\boldsymbol{x}_{Cj}$ and $\boldsymbol{\Gamma}_{Cj}$ are the projections of $\boldsymbol{x}_C$ and $\boldsymbol{\Gamma}_C$, defined in Eq. (4), on the j-th injector. That is, $\boldsymbol{x}_C = [\boldsymbol{x}_{C1} \ \ldots \ \boldsymbol{x}_{CN_C}]^T$ and $\boldsymbol{\Gamma}_C = diag[\boldsymbol{\Gamma}_{C1}, \ldots, \boldsymbol{\Gamma}_{CN_C}]$, where N_C is the number of injectors attached to the Central sub-domain network.

Therefore, system (4) becomes:

$$\begin{aligned} \mathbf{0} &= \boldsymbol{g}_C(\boldsymbol{x}_C, \boldsymbol{V}_C, \boldsymbol{V}_S) \\ \boldsymbol{\Gamma}_{Cj} \dot{\boldsymbol{x}}_{Cj} &= \boldsymbol{\Phi}_{Cj}(\boldsymbol{x}_{Cj}, \boldsymbol{V}_C),\ j = 1, ..., N_C \end{aligned} \tag{7}$$

where $\boldsymbol{V}_S = [\boldsymbol{V}_{S1}, \ldots, \boldsymbol{V}_{SL}]^T$ are the Satellite sub-domain voltage states. Of course, from this vector, only L elements are involved in the equations, i.e. the one bus voltage from each Satellite sub-domain involved in the network component connecting the Central to the Satellite sub-domain (see Fig. 1).

Similarly, system (5) becomes ($i = 1, \ldots, L$):

$$\begin{aligned} \mathbf{0} &= \boldsymbol{g}_{Si}(\boldsymbol{x}_{Si}, \boldsymbol{V}_{Si}, \boldsymbol{V}_{Ci}) \\ \boldsymbol{\Gamma}_{Sij} \dot{\boldsymbol{x}}_{Sij} &= \boldsymbol{\Phi}_{Sij}(\boldsymbol{x}_{Sij}, \boldsymbol{V}_{Si}),\ j = 1, ..., N_{Si} \end{aligned} \tag{8}$$

where N_{Si} is the number of injectors attached to the i-th Satellite network.

The rectangular components of the injected current are included in the differential-algebraic state vector $\boldsymbol{x}$ and can be rewritten for the Central and i-th Satellite sub-domain:

$$\boldsymbol{I}_C = \sum_{j=1}^{N_C} \boldsymbol{C}_{Cj} \boldsymbol{x}_{Cj} \quad \text{and} \quad \boldsymbol{I}_{Si} = \sum_{j=1}^{N_{Si}} \boldsymbol{C}_{Sij} \boldsymbol{x}_{Sij} \tag{9}$$

where $\boldsymbol{C}_{Cj}$ (resp. $\boldsymbol{C}_{Sij}$) is a trivial matrix with zeros and ones whose purpose is to extract the injector current components from $\boldsymbol{x}_{Cj}$ (resp. $\boldsymbol{x}_{Sij}$).

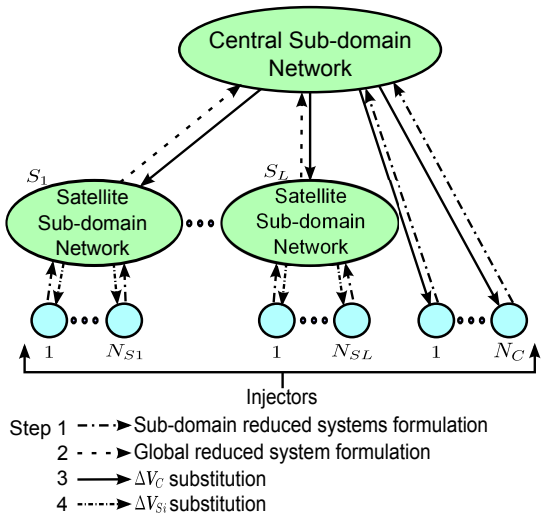

Figure 2. Hierarchical solution of the proposed two-level DDM

The sub-domain network and injector systems are coupled through the currents injected into the network and the voltages of the buses which the injectors are attached to. Systems (7) and (8) are completely equivalent to (4) and (5), and therefore to the original system (1).

III. Schur-complement-based algorithm

For the purpose of numerical simulation, the injector DAE systems are algebraized using a differentiation formula to get the corresponding nonlinear, algebraized, systems. In this work, second-order Backward Differentiation Formulae (BDF-2) is used. After algebraization, Eq. (6) becomes:

$$\mathbf{0} = \boldsymbol{f}_{Cj}(\boldsymbol{x}_{Cj}, \boldsymbol{V}_C), \;\; j = 1, \ldots, N_C \tag{10}$$

where the dependency on the integration time-step size has been omitted for clarity. Then, at each discrete time instant t_n the nonlinear algebraized injector equations are solved together with the network equations to compute the vectors $\boldsymbol{x}(t_n)$ and $\boldsymbol{V}(t_n)$, as outlined in Fig. 2 and summarized below.

First, the injector states ($\boldsymbol{x}_{Cj}$ or $\boldsymbol{x}_{Sij}$) are eliminated from the sub-domain network equations to formulate the sub-domain reduced systems. This leads to reduced systems that involve *only* the sub-domain voltage states ($\boldsymbol{V}_C$ and $\boldsymbol{V}_{Si}$). The elimination is performed in parallel, as the injectors are independent.

Second, the global reduced system is obtained by eliminating the Satellite sub-domain voltage states ($\boldsymbol{V}_{Si}$) from the Central reduced system equations. This leads to a global reduced system that involves *only* the voltage states of the Central sub-domain ($\boldsymbol{V}_C$). This elimination procedure is also performed in parallel, as the Satellite sub-domains are independent.

Third, the global reduced system is solved to compute the Central sub-domain voltages ($\boldsymbol{V}_C$), which are then back-substituted to the Satellite sub-domain reduced systems. This decouples their solution, which now involves only their sub-domain voltage states ($\boldsymbol{V}_{Si}$). Thus, they are solved independently and in parallel.

Finally, the sub-domain voltage states ($\boldsymbol{V}_C$ and $\boldsymbol{V}_{Si}$) are back-substituted in the injector equations, thus decoupling their solution as they now involve only their local states ($\boldsymbol{x}_{Cj}$ or $\boldsymbol{x}_{Sij}$). Hence, their solution is also performed independently and in parallel. The mathematical formulation of this procedure is detailed hereafter.

A. Sub-domain reduced systems formulation

All sub-systems are solved using a separate VDHN method. Thus, at the k-th iteration, the systems (11-14) are formulated. Equations (11) and (12) stem from the linearization of Eq. (4) relative to the Central sub-domain network and injectors connected to it, where $\boldsymbol{A}^k_{Cj}$ is the Jacobian matrix of the j-th injector towards its own states and $\boldsymbol{B}^k_{Cj}$ towards the components of the voltage at its connection bus. $\boldsymbol{E}^k_{Si}$ is the Jacobian of the Central sub-domain towards the voltages of the i-th Satellite sub-domain (S_i).

Equations (13) and (14) stem from the linearization of Eq. (8) relative to the i-th Satellite sub-domain and the injectors connected to it. $\boldsymbol{A}^k_{Sij}$ is similar to $\boldsymbol{A}^k_{Cj}$ and $\boldsymbol{B}^k_{Sij}$ to $\boldsymbol{B}^k_{Cj}$. Finally, $\boldsymbol{F}^k_{Si}$ is the Jacobian of S_i towards the voltages of the Central sub-domain. The decomposed system results in $L + 1 + N_C + \sum_{i=1}^{L} N_{Si}$ linear systems.

The sub-domain *reduced* systems are formulated by eliminating the injector states ($\boldsymbol{x}_{Cj}$ or $\boldsymbol{x}_{Sij}$) from the sub-domain network systems (11) and (13). This leads to the $L+1$ reduced systems (15-16) that involve *only* the sub-domain voltage states ($\boldsymbol{V}_C$ and $\boldsymbol{V}_{Si}$). It is worth mentioning that the reduced system matrices $\widetilde{\boldsymbol{D}}^k_{Si}$ (resp. $\widetilde{\boldsymbol{D}}^k_C$) maintain the sparsity pattern of $\boldsymbol{D}^k_{Si}$ (resp. $\boldsymbol{D}^k_C$) as the injector elimination procedure only modifies 2×2 block-diagonal sub-matrices, which are non-zero by construction.

B. Global reduced system formulation

The global reduced system is obtained by eliminating the Satellite sub-domain voltage states ($\boldsymbol{V}_{Si}$) from the Central reduced system (15). This leads to the global reduced system (17) that involves *only* the voltage states of the Central sub-domain ($\boldsymbol{V}_C$). Similarly to before, the global reduced system matrix $\widetilde{\boldsymbol{D}}^k_C$ maintains the sparsity pattern of $\boldsymbol{D}^k_C$ as the elimination procedure only modifies 2×2 block-diagonal sub-matrices, which are non-zero by construction. Moreover, the computations of $\boldsymbol{E}^k_{Si}\left(\widetilde{\boldsymbol{D}}^k_{Si}\right)^{-1}\boldsymbol{F}^k_{Si}$ and $\boldsymbol{E}^k_{Si}\left(\widetilde{\boldsymbol{D}}^k_{Si}\right)^{-1}\widetilde{\boldsymbol{g}}^k_{Si}$ are efficient as the matrices $\boldsymbol{E}^k_{Si}$ and $\boldsymbol{F}^k_{Si}$ are extremely sparse, given that each Satellite sub-domain is attached only to one bus of the Central sub-domain.

C. Back-substitution and solution

In this step, the global reduced system (17) is solved and the computed Central sub-domain voltage corrections ($\Delta\boldsymbol{V}^k_C$) are back-substituted into the sub-domain reduced systems (16). This decouples the solution of these systems which now involve only their sub-domain voltage states. Thus, they can be solved independently and in parallel to obtain $\Delta\boldsymbol{V}^k_{Si}$.

Linearized Systems

$$\boldsymbol{D}_C^k \Delta \boldsymbol{V}_C^k - \sum_{j=1}^{N_C} \boldsymbol{C}_{Cj} \Delta \boldsymbol{x}_{Cj}^k - \sum_{i=1}^{L} \boldsymbol{E}_{Si}^k \Delta \boldsymbol{V}_{Si}^k = - \underbrace{\boldsymbol{g}_C(\boldsymbol{x}_C^{k-1}, \boldsymbol{V}_C^{k-1}, \boldsymbol{V}_S^{k-1})}_{\boldsymbol{g}_C^k} \tag{11}$$

$$\boldsymbol{A}_{Cj}^k \Delta \boldsymbol{x}_{Cj}^k + \boldsymbol{B}_{Cj}^k \Delta \boldsymbol{V}_C^k = - \underbrace{\boldsymbol{f}_{Cj}(\boldsymbol{x}_{Cj}^{k-1}, \boldsymbol{V}_C^{k-1})}_{\boldsymbol{f}_{Cj}^k}, \; j = 1, ..., N_C \tag{12}$$

$$\boldsymbol{D}_{Si}^k \Delta \boldsymbol{V}_{Si}^k - \sum_{j=1}^{N_{Si}} \boldsymbol{C}_{Sij} \Delta \boldsymbol{x}_{Sij}^k + \boldsymbol{F}_{Si}^k \Delta \boldsymbol{V}_C^k = - \underbrace{\boldsymbol{g}_{Si}(\boldsymbol{x}_{Si}^{k-1}, \boldsymbol{V}_{Si}^{k-1}, \boldsymbol{V}_C^{k-1})}_{\boldsymbol{g}_{Si}^k}, \; i = 1, \ldots, L \tag{13}$$

$$\boldsymbol{A}_{Sij}^k \Delta \boldsymbol{x}_{Sij}^k + \boldsymbol{B}_{Sij}^k \Delta \boldsymbol{V}_{Si}^k = - \underbrace{\boldsymbol{f}_{Sij}(\boldsymbol{x}_{Sij}^{k-1}, \boldsymbol{V}_{Si}^{k-1})}_{\boldsymbol{f}_{Sij}^k}, \; j = 1, ..., N_{Si}, \; i = 1, \ldots, L \tag{14}$$

Reduced Systems

$$\underbrace{\left(\boldsymbol{D}_C^k + \sum_{j=1}^{N_C} \boldsymbol{C}_{Cj} \left(\boldsymbol{A}_{Cj}^k \right)^{-1} \boldsymbol{B}_{Cj}^k \right)}_{\widetilde{\boldsymbol{D}}_C^k} \Delta \boldsymbol{V}_C^k - \sum_{i=1}^{L} \boldsymbol{E}_{Si}^k \Delta \boldsymbol{V}_{Si}^k = \underbrace{-\boldsymbol{g}_C^k - \sum_{j=1}^{N_C} \boldsymbol{C}_{Cj} \left(\boldsymbol{A}_{Cj}^k \right)^{-1} \boldsymbol{f}_{Cj}^k}_{-\widetilde{\boldsymbol{g}}_C^k} \tag{15}$$

$$\underbrace{\left(\boldsymbol{D}_{Si}^k + \sum_{j=1}^{N_{Si}} \boldsymbol{C}_{Sij} \left(\boldsymbol{A}_{Sij}^k \right)^{-1} \boldsymbol{B}_{Sij}^k \right)}_{\widetilde{\boldsymbol{D}}_{Si}^k} \Delta \boldsymbol{V}_{Si}^k + \boldsymbol{F}_{Si}^k \Delta \boldsymbol{V}_C^k = \underbrace{-\boldsymbol{g}_{Si}^k - \sum_{j=1}^{N_{Si}} \boldsymbol{C}_{Sij} \left(\boldsymbol{A}_{Sij}^k \right)^{-1} \boldsymbol{f}_{Sij}^k}_{-\widetilde{\boldsymbol{g}}_{Si}^k}, \; i = 1, \ldots, L \tag{16}$$

$$\underbrace{\left(\widetilde{\boldsymbol{D}}_C^k + \sum_{i=1}^{L} \boldsymbol{E}_{Si}^k \left(\widetilde{\boldsymbol{D}}_{Si}^k \right)^{-1} \boldsymbol{F}_{Si}^k \right)}_{\bar{\boldsymbol{D}}_C^k} \Delta \boldsymbol{V}_C^k = \underbrace{-\widetilde{\boldsymbol{g}}_C^k - \sum_{i=1}^{L} \boldsymbol{E}_{Si}^k \left(\widetilde{\boldsymbol{D}}_{Si}^k \right)^{-1} \widetilde{\boldsymbol{g}}_{Si}^k}_{-\bar{\boldsymbol{g}}_C^k} \tag{17}$$

In turn, the just computed sub-domain voltage corrections ($\Delta \boldsymbol{V}_{Si}^k$ and $\Delta \boldsymbol{V}_C^k$) are back-substituted in the injector equations (12) and (14), thus decoupling their solution as they now involve only their local state corrections ($\Delta \boldsymbol{x}_{Cj}^k$ or $\Delta \boldsymbol{x}_{Sij}^k$). Finally, the injectors are solved independently and in parallel.

After updating the state vectors, i.e. $\boldsymbol{V}_C^k = \boldsymbol{V}_C^{k-1} + \Delta \boldsymbol{V}_C^k$, $\boldsymbol{x}_C^k = \boldsymbol{x}_C^{k-1} + \Delta \boldsymbol{x}_C^k$, $\boldsymbol{V}_{Si}^k = \boldsymbol{V}_{Si}^{k-1} + \Delta \boldsymbol{V}_{Si}^k$, and $\boldsymbol{x}_{Sij}^k = \boldsymbol{x}_{Sij}^{k-1} + \Delta \boldsymbol{x}_{Sij}^k$, the convergence of all sub-systems is checked independently and in parallel. If global convergence has been achieved, then the simulation proceeds to the next time instant, otherwise a new iteration ($k+1$) is performed with the updated variables.

IV. Numerical acceleration techniques

The proposed system decomposition allows accelerating the simulation by avoiding a significant number operations.

First, taking advantage of the fact that each sub-domain is solved by a separate VDHN method, the sub-system Jacobian updates are decoupled and the local matrices (such as $\boldsymbol{D}_C$, $\boldsymbol{D}_{Si}$, $\boldsymbol{A}_{Cj}$, $\boldsymbol{A}_{Sij}$, $\boldsymbol{B}_{Cj}$, $\boldsymbol{B}_{Sij}$, etc.), as well as their Schur-complement terms, are updated asynchronously. In this way, sub-domains which converge fast keep the same local matrices for many iterations and even time-steps, while sub-domains which converge slower update their matrices more frequently.

Second, after a decomposed solution, the convergence of each sub-system is checked individually. Thus, if the convergence criterion is satisfied for a sub-system, the latter is flagged as converged and for the remaining iterations of the current time instant, it is not solved. However, its mismatch, is monitored to guarantee that it remains converged until global convergence is achieved. This technique decreases the computational effort within one discretized time instant without affecting the accuracy of the solution.

V. Parallel processing acceleration

DDM-based algorithms offer parallelization opportunities as independent computations can be performed by different threads (a sequence of programmed instructions that can be managed independently by a scheduler). Most common computers today have a single CPU with multiple cores (independent actual processing units) or hyper-threading technology (ability to execute multiple threads per core). In this paper, hyper-threading was deactivated to keep the analogy of *one thread per core*. This helps to better analyze the performance of the proposed algorithm, as well as to bind each thread to one core for better usage of cache [28].

The proposed algorithm is outlined in Fig. 3, where the parallelized computations are shown in the shaded blocks.

First, the parallel update of the $L + 1 + N_C + \sum_{i=1}^{L} N_{Si}$ systems (11-14) is shown in the upper shaded block. Next, the $L + 1$ sub-domain reduced systems are computed in parallel in the second shaded block, where each parallel task updates one of the systems (15-16).

Then, the global reduced system is solved to compute the $\Delta \boldsymbol{V}_C$ corrections. Schur-complement-based algorithms suffer from the sequentiality introduced by the global reduced system solution [29]. However, due to the high sparsity (retained even after the elimination procedure), the linear nature of the network equations, and the infrequent Jacobian update, this bottleneck is bounded to 1-2% of the overall computational effort. Thus, even though system (17) could be solved with a parallel sparse linear solver, the overhead due to the new synchronization points would counteract the benefits.

Afterwards, the L Satellite sub-domain reduced systems are solved in parallel to compute the $\Delta \boldsymbol{V}_{Si}$ variables. This is shown in the third shaded block.

Finally, after the computed corrections $\Delta \boldsymbol{V}_C$ and $\Delta \boldsymbol{V}_{Si}$ are back substituted in Eqs. (12) and (14), the solution of the $N_C + \sum_{i=1}^{L} N_{Si}$ injector systems are computed in parallel, as shown in the lower shaded block, where each parallel task deals with one injector system.

A. Implementation specifics

Parallel computers can be roughly classified into shared- and distributed-memory. In the first category belong most of the common parallel computers today, such as laptops, desktops, mobile phones, etc. While these parallel computers are inexpensive and widely available, they are limited to their scalability [28] and can only reach up to a few hundred computing nodes. On the other hand, distributed memory computers can easily reach thousands of computing nodes, but are much more expensive usually available in bigger research centers or enterprises. The parallel algorithm presented in Sections II and III does not make any assumption on the type of parallel computer. However, for the implementation of the algorithm, the shared-memory parallel computing model has been used to allow the execution on cheap, multicore, parallel machines (e.g. laptop and desktop computers).

In this work, the OpenMP API was selected as it is supported by most hardware and software vendors and it allows for portable, user-friendly programming [30]. OpenMP allows the execution of a parallel application, without changes, on many different computers. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior. A set of predefined directives are inserted in Fortran, C, or C++ programs to describe how the work is to be shared among threads that will execute on different processors or cores and to order accesses to shared data [28], [30].

It is very important to make sure that parallel threads receive equal amounts of work. Imbalanced load sharing leads to delays, as some threads are still working while others have finished and remain idle. OpenMP includes three easy to employ mechanisms (namely *static*, *dynamic* and *guided*) for achieving good load balance [30]. With the static strategy, the

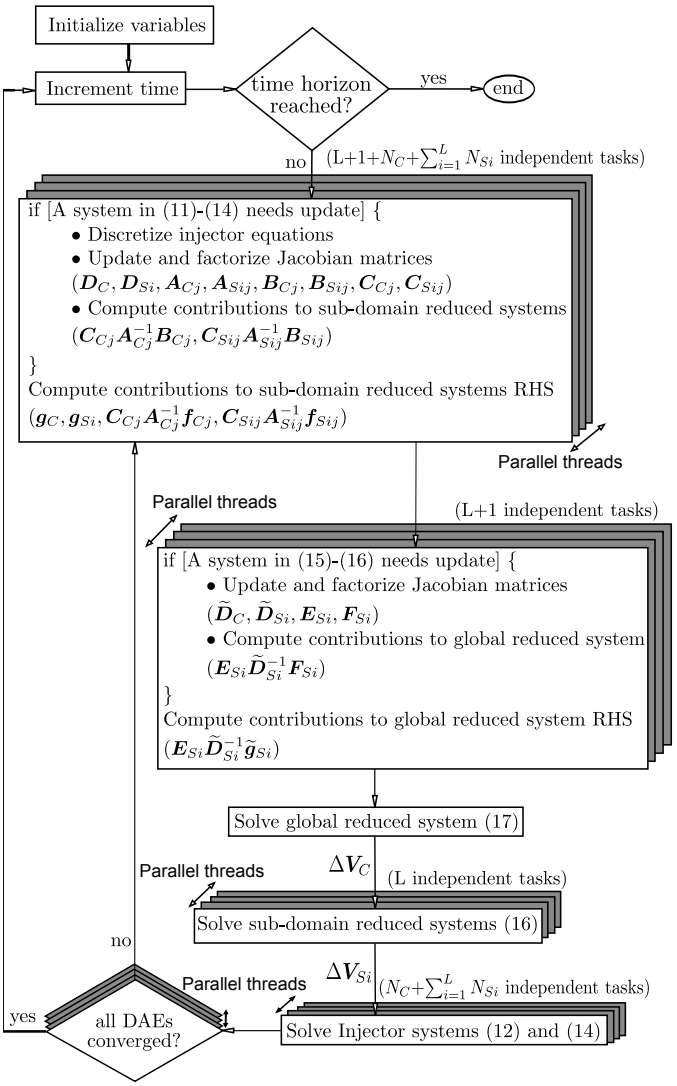

Figure 3. Proposed parallel two-level DDM

scheduling is defined at the beginning of the simulation and the parallel tasks are evenly assigned to the threads. This strategy has the lowest scheduling overhead but can introduce load imbalance if the work inside each task is not equal.

In the proposed algorithm, imbalance between parallel tasks can arise from different sizes of the sub-systems. For example, if the Satellite networks have different numbers of buses, hence different system sizes, the threads computing them will have different work loads. This is also true for the injector sub-systems as different types of components have different model size. Furthermore, the computational burden of the same parallel task over several iterations is also modified due to the techniques in Section IV. Due to these challenging load-balancing conditions, the dynamic strategy has been selected. That is, the scheduling is updated during the execution making sure that the threads are optimally balanced. This strategy comes with some extra OverHead Cost (OHC) for managing the threads. However, the benefit from proper load-balancing was found to more than compensate for this OHC.

In the proposed algorithm, a minimum number of successive

sub-systems assigned to each thread (*chunks*) is defined. Thus, by positioning the sub-system data consecutively in memory, spatial locality can be exploited. That is, the likelihood of accessing consecutive blocks of memory is increased and the amount of cache misses decreased [28]. Although the user can select and experiment with the chunk number to get the best performance, a default value of (number of parallel tasks)/(4× number of threads) has been selected in this work.

Finally, the guided strategy is a compromise between the other two. The scheduling in this strategy is dynamic but the number of tasks assigned to each thread is progressively reduced in size. This way, scheduling overheads are reduced at the beginning of the loop and good load balancing is achieved at the end [16], [28], [30]. Tests using this strategy in this work didn't give any better results than the dynamic.

B. Performance indices

Many different indices exist for assessing the performance of a parallel algorithm. The two indices used in this study, *speedup* and *scalability*, are defined as:

$$Speedup_M = \frac{T_1^*}{T_M} \qquad Scalability_M = \frac{T_1}{T_M} \tag{18}$$

where T_1^* is the runtime of the program with one worker using the fastest sequential algorithm, T_1 is the runtime using the proposed algorithm with one worker, and T_M is the runtime of the same program, using the same algorithm, with M parallel workers.

The first index shows how the algorithm compares to a fast sequential algorithm. For power system dynamic simulations, the popular VDHN scheme has been suggested as benchmark [31]. This consists of solving at each Newton iteration the integrated linear system stemming from the model (1), with an infrequently updated Jacobian. Although there is no proof that this is the fastest sequential algorithm, it is employed in many industrial and academic software and its capabilities and performance are well known. Here on, this method will be referred to as *Integrated*.

For the proposed algorithm, the speedup index shows the performance gained from using both the numerical acceleration technique of Section IV, as well as the benefit from parallelizing its execution. Thus, it shows the benefit from using a decomposed algorithm.

The second index shows how the parallel implementation of the proposed algorithm scales when the number of available processors increases. That is, the tested parallel algorithm is benchmarked against a sequential execution of the *same* algorithm. The scalability index is directly related to Amdahl's law [30] and can be rewritten as:

$$Scalability_M = \frac{T_1}{T_{1S} + \frac{T_{1P}}{M} + OHC(M)} \tag{19}$$

where T_{1S} is the time spent in the sequentially computed portion and T_{1P} in the parallel portions. These values can be estimated with a profiler monitoring the sequential execution ($M = 1$) of the algorithm. Of course, $T_{1S} + T_{1P}$ gives the total execution time T_1. Finally, OHC refers to the cost of making the code run in parallel (creating and managing threads, communication, memory latency, etc.).

For the proposed algorithm, the scalability index focuses on the performance gained from parallelizing its execution. The gain from the methods of Section IV is constant and does not change when more computational threads are used. Thus, by using the sequential execution of the algorithm as benchmark, the numerical acceleration gain is factored out of the calculation.

Both the proposed and the benchmark algorithms were implemented in the academic simulation software RAMSES, developed at the University of Liège. This software is currently being integrated into the operation planning tools of HQ (security limit calculations). The same models, algebraization method (namely BDF-2), and way of handling the discrete events [26] were used. For the solution of the sparse systems (the integrated Jacobian or the reduced systems of Eqs. (16) and (17)), the sparse linear solver HSL MA41 [32] was used. For the solution of the much smaller, dense injector linear systems (12) and (14), Intel MKL LAPACK library was used. In the sequential benchmark, the Jacobian matrix is updated every five iterations until convergence, while for the proposed algorithm, the matrices of *each* sub-domain are updated (independently of the other sub-domains) every five iterations unless convergence has already taken place. The same convergence criteria and tolerances are used for both. Keeping all the aforementioned parameters and solvers identical for both algorithms permits a rigorous evaluation of the proposed algorithm performance.

The results of the two-level Schur-complement-based algorithm are presented in the following sections. The simulations were performed on a 48-core desktop computer[1] (Machine 1), a dual-core[2] (Machine 2), and a quad-core[3] (Machine 3) laptop. The environment variable *OMP_NUM_THREADS* was used to vary the number of computational threads available to the simulation software at each execution.

VI. Results on HQ model

Due to geographic constraints, the HQ transmission system is characterized by long distances (more than 1,000 km) between the hydro-electric power generation and the main consumption centers. A large amount of its power is transferred over 735-kV, series-compensated, transmission lines. Owing to this structure (more than 11,000 km of 735-kV lines), the system is constrained by both transient (angle) and long-term voltage stability. Moreover, as it is connected to its neighbors through DC lines or radial generators, its frequency dynamics after a severe power imbalance are carefully checked. Extensive dynamic simulations are performed to update the secure power transfer limits taking into account the N-1 security

[1]AMD Opteron Interlagos CPU 6238 @ 2.60GHz, 16KB private L1, 2048KB shared per two cores L2 and 6144KB shared per six cores L3 cache, 64GB RAM, Debian Linux 8

[2]Intel Core i7 CPU 4510U @ 3.10GHz, 64KB private L1, 512KB private L2 and 4096KB shared L3 cache, 7.7GB RAM, Microsoft Windows 8.1

[3]Intel Core i7 CPU 2630QM @ 2.90GHz, 64KB private L1, 256KB private L2 and 6144KB shared L3 cache, 7.7GB RAM, Microsoft Windows 7

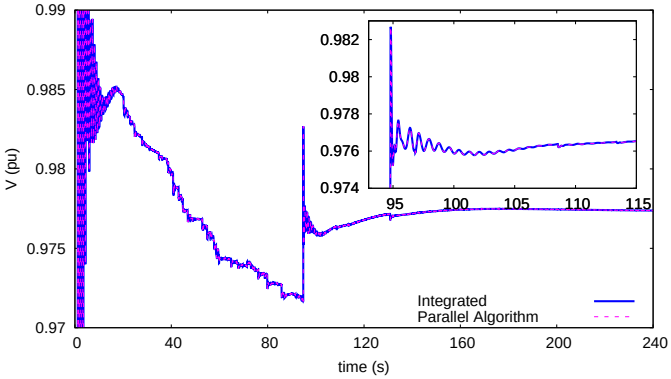

Figure 4. HQ: Evolution at high voltage bus in the Central sub-domain

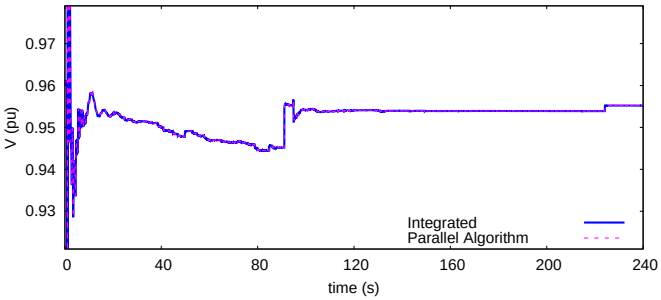

Figure 5. HQ: Evolution of voltage at a medium-voltage bus in a Satellite sub-domain

criteria, as well as to check the response of various stabilizing controls after severe disturbances.

The HQ network model includes 2565 buses, 3225 branches, and 290 power plants with a detailed representation of the synchronous machine, its excitation system, automatic voltage regulator, power system stabilizer, turbine, and speed governor. Moreover, 4311 dynamically modeled loads (different types of induction motors, voltages sensitive loads, etc.), are included to better capture the dynamic response of the real system. In the long-term the system evolves under the effect of 1111 Load Tap Changers (LTCs), 51 Automatic Shunt Reactor Tripping (ASRT) devices [33], as well as OvereXcitation Limiters (OXL). The resulting, integrated, model has 35559 differential-algebraic states.

The first level of partitioning, yields $L = 80$ Satellite sub-domains with a total of 1602 buses in them (thus leaving 963 buses in the Central). For the second level of decomposition, $290 + 4311 = 4601$ injectors are considered in the 81 sub-domains (Central and 80 Satellite). The decomposition was performed using the graph representation of the system and automatically identifying network sub-domains that are connected to the bulk transmission system at only one bus.

A. Test-case dynamics

The disturbance consists of a short circuit at an important transmission bus, lasting six cycles (at 60 Hz), and cleared by opening a 735-kV line. The system is simulated over an interval of 240 s with a time-step of one cycle.

Figure 4 shows the voltage evolution after the disturbance, at the faulted transmission bus, located in the Central sub-

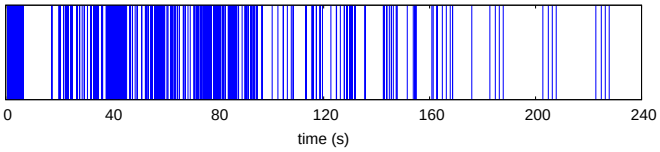

Figure 6. HQ: Discrete events during the simulation

Table I
HQ: PROFILING OF SEQUENTIAL EXECUTION ($M = 1$)

Tasks	% time	Parallel
Injector discretization, sub-domain network and injector Jacobian computation and factorization	9.69	YES
Sub-domain reduced system formulation	3.08	YES
Global reduced system solution	1.10	NO
Sub-domain reduced system solution	3.36	YES
Injector solution	62.39	YES
Convergence check	4.63	YES
Discrete events and controllers computation	5.64	YES
Time step initialization	10.11	NO
In parallel (T_{1P})	**88.79**	-

domain. The action of the ASRT device at $t \approx 95$ s has a significant effect on the dynamics of the system and secures its long-term stability. Figure 5 shows the voltage evolution of an LTC-controlled medium-voltage bus, located in a Satellite sub-domain. Finally, Fig. 6 shows the discrete events taking place during the simulation. These refer to discrete controller actions (e.g. LTCs, ASRTs, etc.) or internal to the injectors discrete transitions (i.e. OXLs, integration limits, etc.). It can be seen that until almost the end of the simulation there are still discrete events taking place.

As seen in Figs. 4 and 5, both the benchmark Integrated and the proposed parallel algorithms give exactly the same results as they solve the same DAE system with the same accuracy.

B. Test-case profiling

The sequential execution ($M = 1$) of the proposed algorithm has been profiled to identify how the computing time is shared among the various tasks. As seen in Table I, 88.79% of the time is spent in the parallel sections of the code. Thus, referring to Eq. (19), $T_{1P} = 0.8879$ and $T_{1S} = 0.1121$.

If an ideal parallel machine was considered, the theoretical scalability would be given by Eq. (19) with $OHC(M) = 0$. For example, with $M = 30$, the value is 7.1.

C. Sequential and parallel performance

The speedup and scalability performance indices are presented in Table II for Machine 1. First, it can be observed that

Table II
HQ: PERFORMANCE INDICES ON MACHINE 1

	Integr.	Proposed parallel algorithm						
Cores Used	1	1	2	6	12	24	30	36
Elapsed Time	417	282	157	89	60	47	44	46
Speedup	-	1.5	2.7	4.7	7.0	8.9	9.5	9.1
Scalability	-	1.0	1.8	3.2	4.7	6.0	6.4	6.1

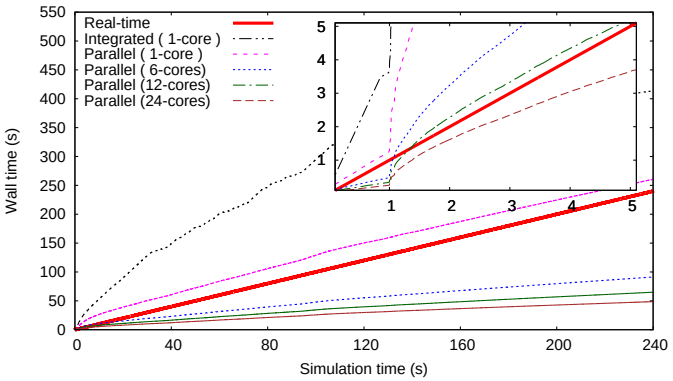

Figure 7. HQ: Real-time performance of algorithm on Machine 1

Table III
HQ: PERFORMANCE INDICES ON MACHINES 2 AND 3

		Integrated	Proposed parallel algorithm		
	Cores Used	1	1	2	4
Machine 2	Elapsed Time	245	139	106	-
	Speedup	-	1.8	2.3	-
	Scalability	-	1.0	1.3	-
Machine 3	Elapsed Time	200	121	83	67
	Speedup	-	1.7	2.4	3.0
	Scalability	-	1.0	1.4	1.8

the proposed algorithm is 1.5 times faster than the benchmark integrated scheme even in sequential execution. This speedup is a result of the numerical accelerations mentioned in Section IV, as explained in Section V-B. Therefore, the proposed algorithm can provide significant acceleration even when executed on single-core machines.

When proceeding to the parallel execution, the simulation is performed in 44 seconds, that is 9.5 times faster than the sequential Integrated scheme, with the use of 30 cores. At the same time, the scalability of the algorithm reaches 6.4 to be compared with the theoretical maximum of 7.1. The decreased performance is due to the parallelization OHC.

A decrease of performance is observed when the number of threads exceeds 30 (see Table II). That is, when using more than 30 threads, the performance degrades compared to the maximum achieved. The degradation occurs as the execution time gained when increasing from 30 to 36 threads ($\frac{P}{30} - \frac{P}{36}$), is smaller than the overhead cost of creating and managing the six extra threads ($OHC(36) - OHC(30)$). An extensive discussion of this phenomenon can be found in [16], [28].

Figure 7 shows the real-time performance of the algorithm. When the wall time curve is above the real-time line, then the simulation is lagging; otherwise, the simulation is faster than real-time and can be used for more demanding applications, like look-ahead simulations [34], training simulators or hardware/software in the loop. On this power system, the proposed algorithm performs faster than real-time when executed on 24 or more cores.

Finally, Table III summarizes the speedup and scalability of the algorithm of Machines 2 and 3, which are normal laptop multi-core computers. It can be seen that these computers can simulate the scenario in 106 s (resp. 67 s), providing a speedup of 2.3 (resp. 3) and a scalability of 1.3 (resp. 1.8) times.

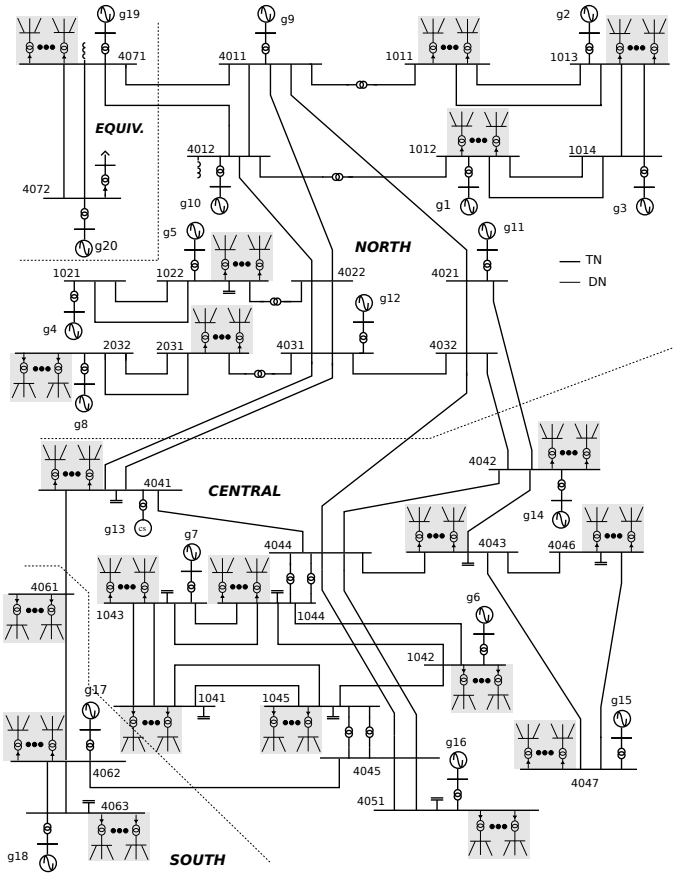

Figure 8. Expanded Nordic test-system

VII. RESULTS ON EXPANDED NORDIC TN-DN SYSTEM

A. Test system and its two-level decomposition

This section reports on results obtained with a large combined transmission and distribution network model. The TN part is based on the Nordic test system, documented in [35] and shown in Fig. 8. This model has been expanded with 146 DNs which replace the loads of the Nordic system. The model and data of each DN were taken from [36] and scaled to match the original aggregate loads seen by the TN. Multiple DNs were used to match the original load powers, taking into account the nominal power of the TN-DN transformers.

Each one of the 146 DNs is connected to the TN through two parallel transformers equipped with LTCs. Each DN includes 100 buses, one distribution voltage regulator equipped with LTC, three PhotoVoltaic (PV) units [37], three type-2, two type-3 Wind Turbines (WTs) [38], and 133 loads represented by small induction motors and exponential loads. All the DGs comply with the Low Voltage and Fault Ride Through (LVFRT) requirements, taken from [39].

The whole, combined transmission and distribution system includes 14653 buses, 15994 branches, 23 large synchronous machines, 438 PVs, 730 WTs, and 19419 dynamically modeled loads. The resulting model has 143462 differential-algebraic states. The first decomposition is performed on the boundary between TN and DNs, thus creating $L = 146$ Satellite DN sub-domains attached to the Central DN sub-domain.

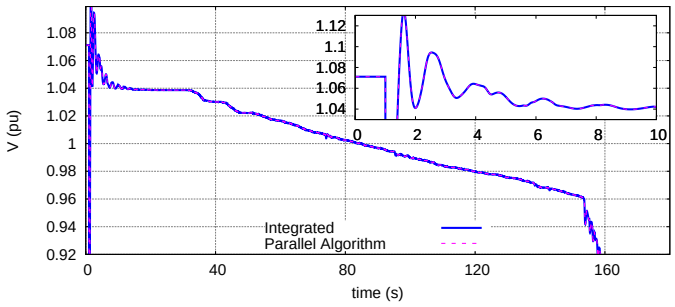

Figure 9. Expanded Nordic: Voltage evolution at TN bus 4044

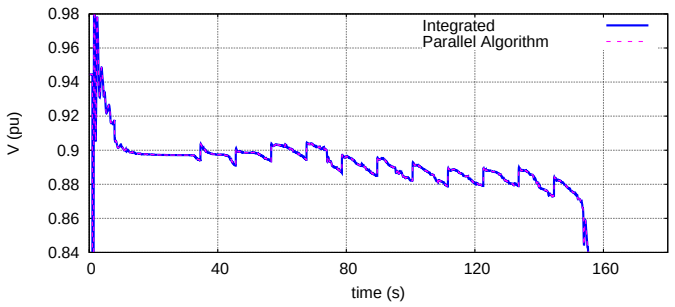

Figure 10. Expanded Nordic: Voltage evolution at a DN bus

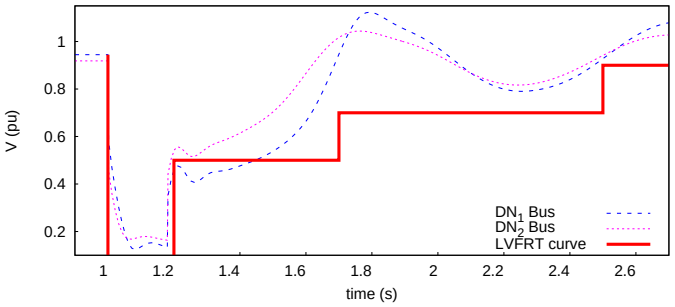

Figure 11. Expanded Nordic: DG terminal voltages in two DNs

Next, each sub-domain is decomposed into its network and injectors. This leads to $N_{S1} = N_{S2} = ... = 141$ and $N_C = 24$.

B. Test-case dynamics

The disturbance considered is a 5-cycle, three-phase, solid fault near the TN bus 4032, cleared by the opening the faulted line 4032-4042, which remains open. The system is simulated with a time-step size of one cycle (20 ms).

The system is stable in the short term but experiences long-term voltage instability under the effect of LTCs unsuccessfully attempting to restore the DN voltages, and OXLs acting on the TN-connected generators. Both contribute to depressing the TN voltages. This is illustrated in Fig. 9 which shows a zoom of the voltage evolution at TN bus 4044 until the final collapse ($t \approx 155$ s). The corresponding voltage evolution at a DN bus is shown in Fig. 10.

Furthermore, Fig. 11 shows the voltage at buses in two different DNs connected to TN buses 1041 (DN_1) and 4043 (DN_2), respectively. DGs are connected to these DN buses and the voltage evolution is compared to the LVFRT curve to decide whether they remain connected or not. It can be seen that the DG in DN_1 disconnects at $t \approx 1.2$ s, while that in DN_2 remains connected. The DG tripping leads to losing approximately 140 MW, which increases the power drawn by the DNs from the TN. This example illustrates the need for detailed combined dynamic simulations when the DG penetration level becomes significant.

Table IV
EXPANDED NORDIC: PERFORMANCE INDICES ON MACHINE 1

	Integr.	Proposed parallel algorithm				
Cores Used	1	1	6	12	24	36	44
Elapsed Time	1132	857	160	92	70	63	62
Speedup	-	1.3	7.1	12.3	16.2	17.9	18.3
Scalability	-	1.0	5.4	9.3	12.2	13.6	13.8

Table V
EXPANDED NORDIC: PERFORMANCE INDICES ON MACHINES 2 AND 3

		Integr.	Proposed parallel algorithm		
	Cores Used	1	1	2	4
Machine 2	Elapsed Time	532	449	264	-
	Speedup	-	1.2	2.0	-
	Scalability	-	1.0	1.7	-
Machine 3	Elapsed Time	464	371	166	128
	Speedup	-	1.3	2.8	3.6
	Scalability	-	1.0	2.2	2.9

C. Sequential and parallel performance

The speedup and scalability indices are presented in Table IV for Machine 1. The proposed algorithm is 1.3 times faster than the benchmark integrated scheme in sequential execution. As explained in Section V-B, this performance gain in sequential execution is mainly due to the numerical acceleration techniques shown in Section IV.

When proceeding to the parallel execution, the simulation is performed in 62 seconds, that is 18.3 times faster than the sequential Integrated scheme, with the use of 44 cores. The proposed algorithm, can simulate the test-case in real-time when using 44 cores on Machine 1.

Table V reports on the speedup and scalability of the algorithm when using Machines 2 and 3. It can be seen that even on these small multicore machines, this large-scale power system model can be simulated efficiently. A speedup of 1.2 (resp. 1.3) is observed in sequential execution on Machine 2 (resp. 3) due to the acceleration techniques of Section IV. When proceeding to parallel computing, this speedup reaches 2.0 (resp. 3.6) times, considering both the numerical acceleration and the parallelization gains.

From the scalability index, it can be seen that the execution is 1.7 (resp. 2.9) times faster in parallel compared to the sequential execution of the same algorithm. This performance gain can be attributed solely to the parallelization.

Furthermore, it is worth noting that in Machine 3, a slightly super-linear scalability is observed when using two cores (2.2 times on two cores). This type of behavior, although rarely observed, is usually due to the increase of the cache size and the optimization of the cache use when in parallel [28].

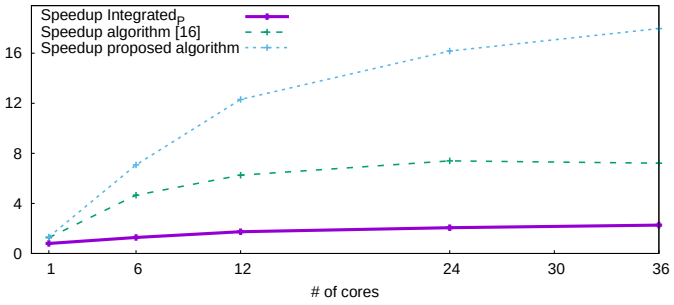

Figure 12. Expanded Nordic: Speedup comparison of parallel methods

D. Comparison with other parallel algorithms

In this section, the proposed algorithm is compared against two other parallel algorithms applied to the same test case. First, the sequential sparse solver used in the Integrated benchmark (namely MA41 from Harwell library [32]) is replaced by the parallel solver MKL PARDISO. Second, the proposed two-level algorithm is compared against the single-level, Schur-complement-based, decomposition algorithm in [16].

The speedup of each algorithm, compared to the Integrated with sequential sparse solver, is shown in Fig. 12.

The Integrated algorithm using the PARDISO solver (denoted Integrated$_P$) is 20% slower in sequential execution than the same algorithm using the MA41 solver. However, when more computing threads are used, Integrated$_P$ becomes up to 2.2 times faster. While the factorizations and solutions of the integrated sparse linear system are performed in parallel, these only sum for 65% of the simulation time. Thus, using (19) and ignoring the OHC, it can be seen that the scalability of Integrated$_P$ is limited to 2.9 times. The computation of the Jacobian matrix, the convergence check, the discrete events, etc. are all performed in the sequential part of Integrated$_P$. In addition, the numerical techniques presented in Section IV cannot be applied, since the entire system is treated as one domain.

Next, the single-level, Schur-complement, decomposition algorithm of [16] reaches a speedup of 6.1 times. While the parallelization potential of this algorithm is higher than Integrated$_P$, still the transmission and distribution network is treated as one sub-domain and this sequentiality impedes the scalability.

Finally, the proposed two-level algorithm reaches a speedup of 18.3 times (see also Table IV) due to the higher parallelization potential provided by the decomposition of the TN and the DNs, and their individual treatment allowing to exploit the techniques of Section IV.

VIII. Conclusion

In this paper, a parallel, two-level, Schur-complement-based algorithm for the dynamic simulation of electric power systems has been presented. It accelerates the simulation in two ways. On the one hand, the procedure is accelerated numerically, by exploiting the locality of the sub-systems and avoiding many unnecessary computations (factorizations, evaluations, solutions). On the other hand, the procedure is accelerated computationally, by exploiting the parallelization opportunities inherent to DDMs at both decomposition levels.

The previous acceleration techniques do not affect the accuracy of the simulated response. Furthermore, due to the Schur-complement approach used to treat the interface variables, the algorithm convergence is not sensitive to the selected partitioning or simulated contingency. However, another technique, first introduced in [40], can be used to calculate linear, sensitivity-based, equivalent models of the Satellite sub-domains during the dynamic simulation to further accelerate the simulation while sacrificing some accuracy.

Along with the proposed algorithm, an implementation based on the shared-memory parallel programming model has been presented. The implementation is portable, as it can be executed on any platforms supporting the OpenMP API. It exhibits good sequential and parallel performance on a wide range of inexpensive, shared-memory, multi-core computers.

Finally, it was shown that the proposed algorithm can provide significantly higher speedup compared to modern, "off-the-shelf", parallel sparse linear solvers. The higher performance is made possible by the two-level decomposition which allows to extract higher parallelization.

References

[1] T. Van Cutsem, M. E. Grenier, and D. Lefebvre, "Combined detailed and quasi steady-state time simulations for large-disturbance analysis," *International Journal of Electrical Power and Energy Systems*, vol. 28, no. August, pp. 634–642, 2006.

[2] R. C. Green, L. Wang, and M. Alam, "High performance computing for electric power systems: Applications and trends," in *Proceedings of 2011 IEEE PES General Meeting*, Geneva, 2011.

[3] J. Machowski, J. Bialek, and J. Bumby, *Power system dynamics: stability and control*. JohnWiley & Sons, 2008.

[4] A. Toselli and O. Widlund, *Domain Decomposition Methods - Algorithms and Theory*, ser. Springer Series in Computational Mathematics. Berlin-Heidelberg: Springer-Verlag, 2005, vol. 34.

[5] G. Kron, *Diakoptics: the piecewise solution of large-scale systems*. London: MacDonald, 1963.

[6] F. Alvarado, "Parallel Solution of Transient Problems by Trapezoidal Integration," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-98, no. 3, pp. 1080–1090, May 1979.

[7] M. La Scala, G. Sblendorio, and R. Sbrizzai, "Parallel-in-time implementation of transient stability simulations on a transputer network," *IEEE Transactions on Power Systems*, vol. 9, no. 2, pp. 1117–1125, May 1994.

[8] F. Iavernaro, M. La Scala, and F. Mazzia, "Boundary values methods for time-domain simulation of power system dynamic behavior," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 45, no. 1, pp. 50–63, 1998.

[9] M. Ilic'-Spong, M. L. Crow, and M. A. Pai, "Transient Stability Simulation by Waveform Relaxation Methods," *IEEE Transactions on Power Systems*, vol. 2, no. 4, pp. 943–949, 1987.

[10] M. Crow and M. Ilic, "The parallel implementation of the waveform relaxation method for transient stability simulations," *IEEE Transactions on Power Systems*, vol. 5, no. 3, pp. 922–932, Aug. 1990.

[11] V. Jalili-Marandi and V. Dinavahi, "Instantaneous Relaxation-Based Real-Time Transient Stability Simulation," *IEEE Transactions on Power Systems*, vol. 24, no. 3, pp. 1327–1336, Aug. 2009.

[12] F. Pruvost, P. Laurent-Gengoux, F. Magoules, and B. Haut, "Accelerated Waveform Relaxation methods for power systems," in *Proceedings of 2011 International Conference on Electrical and Control Engineering*, Wuhan, 2011, pp. 2877–2880.

[13] CRSA, RTE, TE, and TU/e, "D4.1: Algorithmic requirements for simulation of large network extreme scenarios," Tech. Rep., 2011. [Online]. Available: http://www.fp7-pegase.eu/

[14] Y. Liu and Q. Jiang, "Two-Stage Parallel Waveform Relaxation Method for Large-Scale Power System Transient Stability Simulation," *IEEE Transactions on Power Systems*, pp. 1–10, 2015.

[15] V. Jalili-Marandi, F. J. Ayres, E. Ghahremani, J. Belanger, and V. Lapointe, "A real-time dynamic simulation tool for transmission and distribution power systems," in *2013 IEEE Power & Energy Society General Meeting*. IEEE, 2013, pp. 1–5.

[16] P. Aristidou, D. Fabozzi, and T. Van Cutsem, "Dynamic Simulation of Large-Scale Power Systems Using a Parallel Schur-Complement-Based Decomposition Method," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2561–2570, Oct. 2013.

[17] D. Guibert and D. Tromeur-Dervout, "A Schur Complement Method for DAE/ODE Systems in Multi-Domain Mechanical Design," in *Domain Decomposition Methods in Science and Engineering XVII*. Springer, 2008, pp. 535–541.

[18] P. Aristidou, "Time-domain simulation of large electric power systems using domain-decomposition and parallel processing methods," PhD thesis, University of Liège, 2015. [Online]. Available: http://orbi.ulg.ac.be/handle/2268/183353

[19] D. Fang and Y. Xiaodong, "A New Method for Fast Dynamic Simulation of Power Systems," *IEEE Transactions on Power Systems*, vol. 21, no. 2, pp. 619–628, May 2006.

[20] K. Strunz and E. Carlson, "Nested fast and simultaneous solution for time-domain simulation of integrative power-electric and electronic systems," *IEEE Transactions on Power Delivery*, vol. 22, no. 1, pp. 277–287, Jan 2007.

[21] D. Kulkarni and D. Tortorelli, "A domain decomposition based two-level Newton scheme for nonlinear problems," *Domain Decomposition Methods in Science and Engineering*, pp. 615–622, 2005.

[22] L. Luo, Y. Zhao, and X.-C. Cai, "A hybrid implementation of two-level domain decomposition algorithm for solving elliptic equation on cpu/gpus," in *13th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, Dec 2012, pp. 474–477.

[23] M. Terracol, P. Sagaut, and C. Basdevant, "A multilevel algorithm for large-eddy simulation of turbulent compressible flows," *Journal of Computational Physics*, vol. 167, no. 2, pp. 439 – 474, 2001.

[24] P. Kundur, *Power system stability and control*. McGraw-hill New York, 1994.

[25] F. Milano, *Power System Modelling and Scripting*, ser. Power Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[26] D. Fabozzi, A. S. Chieh, P. Panciatici, and T. Van Cutsem, "On simplified handling of state events in time-domain simulation," in *Proc. of 17th Power System Computational Conference (PSCC)*, Stockholm, 2011.

[27] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of 7th Python in Science Conference (SciPy2008)*, vol. 836, Pasadena, 2008, pp. 11–15.

[28] D. Gove, *Multicore Application Programming: For Windows, Linux, and Oracle Solaris*. Addison-Wesley Professional, 2010.

[29] Y. Saad, *Iterative methods for sparse linear systems*, 2nd ed. Society for Industrial and Applied Mathematics, 2003.

[30] B. Chapman, G. Jost, and R. Van Der Pas, *Using OpenMP: Portable Shared Memory Parallel Programming*. MIT Press, 2007.

[31] J. Chai and A. Bose, "Bottlenecks in parallel algorithms for power system stability analysis," *IEEE Transactions on Power Systems*, vol. 8, no. 1, pp. 9–15, Feb. 1993.

[32] HSL, "A collection of Fortran codes for large scale scientific computation." 2014. [Online]. Available: http://www.hsl.rl.ac.uk/

[33] S. Bernard, G. Trudel, and G. Scott, "A 735 kV shunt reactors automatic switching system for Hydro-Quebec network," *IEEE Transactions on Power Systems*, vol. 11, no. 4, pp. 2024–2030, 1996.

[34] Z. Huang, S. Jin, and R. Diao, "Predictive Dynamic Simulation for Large-Scale Power Systems through High-Performance Computing," in *2012 SC Companion: High Performance Computing, Networking Storage and Analysis*. IEEE, Nov. 2012, pp. 347–354.

[35] T. Van Cutsem and L. Papangelis, "Description, Modeling and Simulation Results of a Test System for Voltage Stability Analysis," University of Liege, Tech. Rep. November, 2013. [Online]. Available: http://hdl.handle.net/2268/141234

[36] A. Ishchenko, "Dynamics and stability of distribution networks with dispersed generation," Ph.D. dissertation, Eindhoven University of Technology, 2008.

[37] "Wind Power Plant Dynamic Modeling Guide," Western Electricity Coordinating Council (WECC), Tech. Rep., 2014.

[38] A. Ellis, Y. Kazachkov, E. Muljadi, P. Pourbeik, and J. J. Sanchez-Gasca, "Description and technical specifications for generic WTG models - A status report," in *Proceedings of 2011 IEEE PES Power Systems Conference and Exposition (PSCE 2011)*, Phoenix, Mar. 2011.

[39] J. Schlabbach, "Low voltage fault ride through criteria for grid connection of wind turbine generators," in *Proceedings of 5th International Conference on the European Electricity Market*, Lisboa, May 2008.

[40] P. Aristidou and T. Van Cutsem, "Dynamic simulations of combined transmission and distribution systems using decomposition and localization," in *Proceedings of IEEE PES 2013 PowerTech conference*, Grenoble, 2013.

Petros Aristidou (M'10) obtained his Diploma in Electrical and Computer Engineering from the National Technical University of Athens, Greece, and his Ph.D. in Engineering Sciences from the University of Liège, Belgium, in 2010 and 2015, respectively. He is currently a postdoctoral researcher at the Swiss Federal Institute of Technology in Zürich (ETH Zürich). His research interests include power system dynamics, control, and simulation.

Simon Lebeau received his B.Eng. degree in Electrical Engineering from École Polytechnique, Montréal, in 2001. He is with the Hydro-Québec TransEnergie Division where is involved in operations planning for the Main Network using dynamic simulations.

Thierry Van Cutsem (F'05) graduated in Electrical-Mechanical Engineering from the University of Liège, Belgium, where he obtained the Ph.D. degree and he is now adjunct professor. Since 1980, he has been with the Fund for Scientific Research (FNRS), of which he is now a Research Director. His research interests are in power system dynamics, stability, security, monitoring, control and simulation. He has been working on voltage stability in collaboration with transmission system operators from France, Canada, Greece, Belgium and Germany. He is a fellow of the IEEE and Past Chair of the IEEE PES Power System Dynamic Performance Committee.