A Schur Complement Method for DAE Systems in Power System Simulation

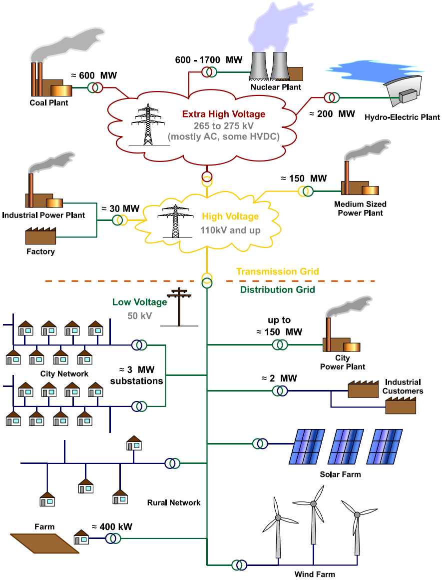
Petros Aristidou Davide Fabozzi Thierry Van Cutsem

Department of Electrical and Computer Engineering
University of Liège

21st International Conference on Domain Decomposition Methods,
Rennes, 28th June 2012

Outline

Large-Scale Electric Networks



Dynamic Simulation

Used by power system operation companies for:

- dynamic security assessment of the network by analyzing large sets of scenarios in real-time
- operator training
- scheduling day to day operation with optimal power flow studies using dynamic system response

Dynamic Simulation

Used by power system operation companies for:

- dynamic security assessment of the network by analyzing large sets of scenarios in real-time
- operator training
- scheduling day to day operation with optimal power flow studies using dynamic system response

Used by people involved in designing and planning for:

- interactive evaluation of changes
  - adding new transmission lines
  - increasing percentage renewable energy sources
  - implementing new control schemes
  - decommissioning old power plants
- hardware-in-the-loop simulations

Dynamic Simulation

Used by power system operation companies for:

- dynamic security assessment of the network by analyzing large sets of scenarios in real-time(Speed: Critical)
- operator training (Speed: Critical)
- scheduling day to day operation with optimal power flow studies using dynamic system response (in research stage)

Used by people involved in designing and planning for:

- interactive evaluation of changes (Speed: Desired)
    - adding new transmission lines
    - increasing percentage renewable energy sources
    - implementing new control schemes
    - decommissioning old power plants
- hardware-in-the-loop simulations (Speed: Critical, Real-time)

Dynamic Simulation

Used by power system operation companies for:

- dynamic security assessment of the network by analyzing large sets of scenarios in real-time(Speed: Critical)
- operator training (Speed: Critical)
- scheduling day to day operation with optimal power flow studies using dynamic system response (in research stage)

Used by people involved in designing and planning for:

- interactive evaluation of changes (Speed: Desired)
  - adding new transmission lines
  - increasing percentage renewable energy sources
  - implementing new control schemes
  - decommissioning old power plants
- hardware-in-the-loop simulations (Speed: Critical, Real-time)

Goal

Get accurate results, *faster*

Mathematical Formulation

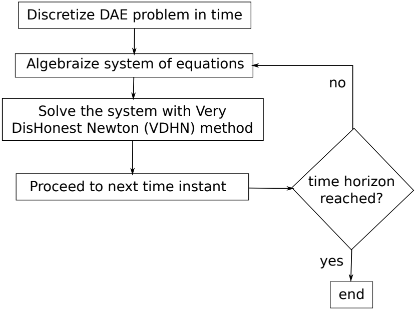
Initial value, hybrid, stiff, non-linear DAE problem:

$$\mathbf{\Gamma}\dot{\mathbf{x}} = \Xi(\mathbf{x}, \mathbf{V})$$
$$x(t_0) = x_0, V(t_0) = V_0$$

where:

- $\mathbf{V}$ is the vector of voltages through the network
- $\mathbf{x}$ is the expanded state vector containing the differential and algebraic variables (except the voltages) of the system
- $\mathbf{\Gamma}$ is a diagonal matrix with $(\mathbf{\Gamma})_{\ell\ell} = \begin{cases} 0 & \text{iff } \ell^{th} \text{ equation is algebraic} \\ 1 & \text{iff } \ell^{th} \text{ equation is differential} \end{cases}$

Standard Integrated Algorithm



Procedure Acceleration

- Sparse linear solvers
- Model Simplification/Reduction
- Better hardware (larger memory, faster CPUs, etc)

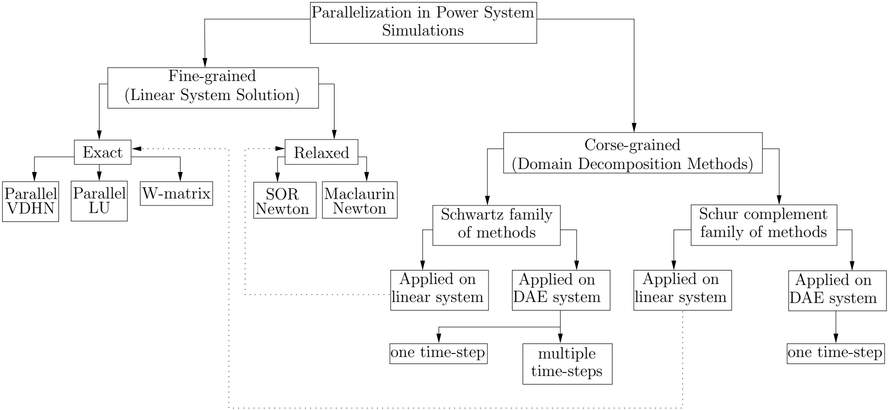
Increased Demands

- Bigger models
- Higher accuracy
- More complex, hybrid, components

Result

Dynamic simulations of large-scale systems *remain* time consuming

DDMs and Parallel Processing techniques

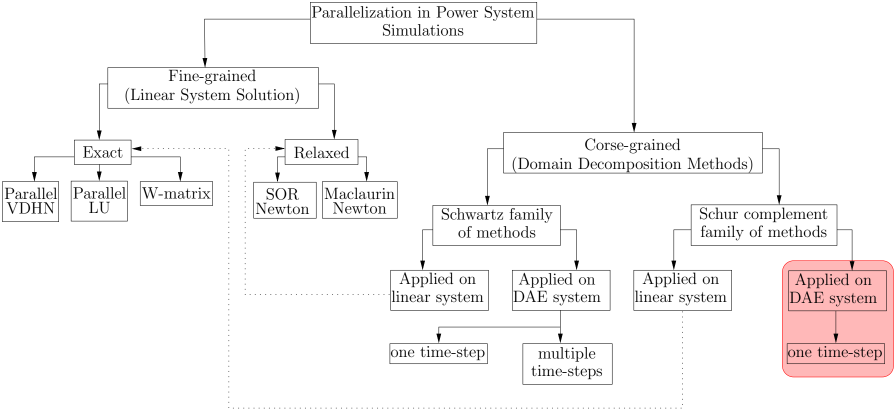


[Kron (1963)]

[Ilic'-Spong, Crow and Pai (1987)]

[La Scala, Sbrizzai and Torelli (1991)]

DDMs and Parallel Processing techniques

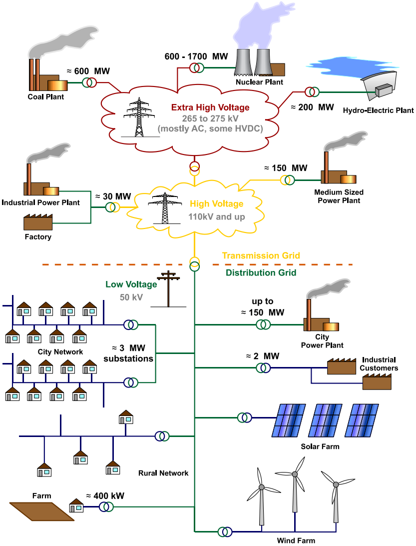


[Kron (1963)]

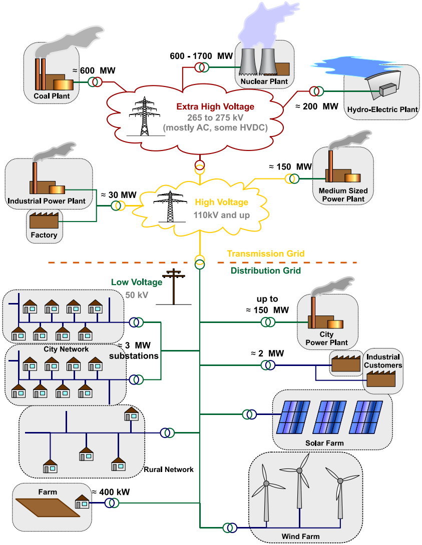
[Ilic'-Spong, Crow and Pai (1987)]

[La Scala, Sbrizzai and Torelli (1991)]

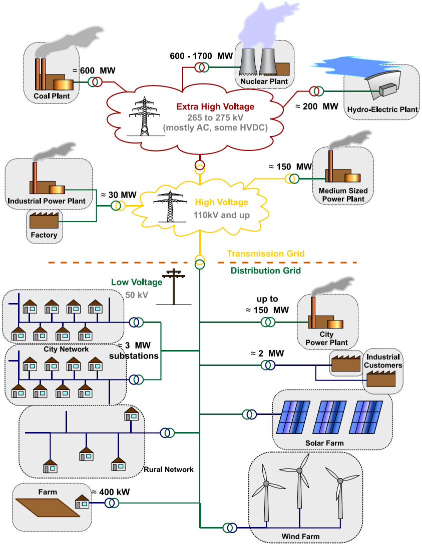
Decomposition Scheme

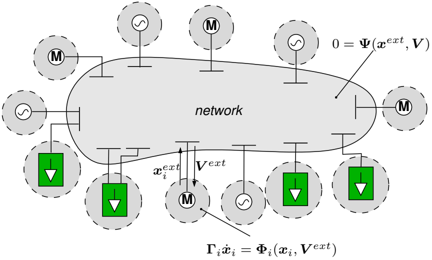


Decomposition Scheme

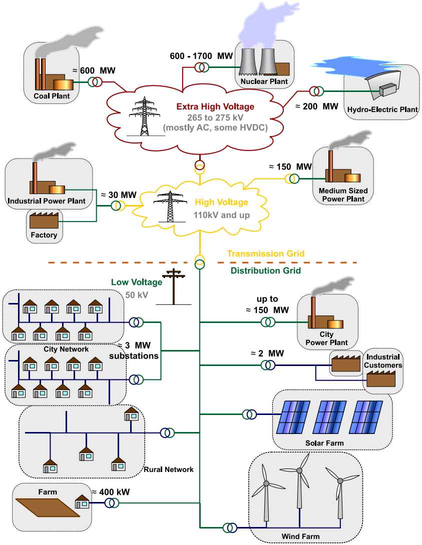


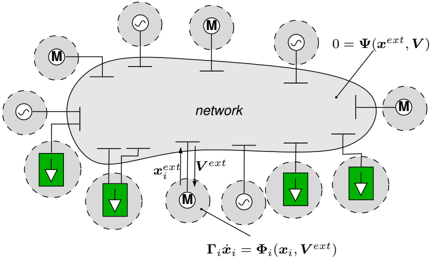
Decomposition Scheme





Decomposition Scheme





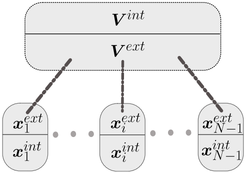
Network sub-domain:

$$0 = \mathbf{\Psi}(\mathbf{x}^{ext}, \mathbf{V})$$

Component i (*injector*) sub-domain:

$$\mathbf{\Gamma}_i \dot{\mathbf{x}}_i \quad = \mathbf{\Phi}_i(\mathbf{x}_i, \mathbf{V}^{ext})$$

Decomposition Scheme



where:

- $\mathbf{x}_i = \begin{pmatrix} \mathbf{x}_i^{int} \\ \mathbf{x}_i^{ext} \end{pmatrix}$, injector i sub-domain variables
  - $\mathbf{x}_i^{int}$: interior variables, coupled only with local equations
  - $\mathbf{x}_i^{ext}$: local Interface variables, coupled with both local and non-local (external to the sub-domain) equations
- $\mathbf{V} = \begin{pmatrix} \mathbf{V}^{int} \\ \mathbf{V}^{ext} \end{pmatrix}$, network sub-domain variables
  - $\mathbf{V}^{int}$: interior variables, coupled only with local equations
  - $\mathbf{V}^{ext}$: local Interface variables, coupled with both local and non-local (external to the sub-domain) equations

Solution Steps

Based on the Schur complement DDM:

1. Build sub-domain local systems (apply integration formula, compute Newton method matrices, etc)
2. Eliminate the interior variables from the sub-domains and build a global reduced system involving only the interface variables
3. Solve the reduced system to obtain the interface variables
4. Backward substitute the interface variables into the sub-domain local systems and solve to obtain interior variables
5. Repeat until all sub-domain local systems have converged

[Saad (2003)]

Injector sub-domain local system

$$\underbrace{\begin{pmatrix} \mathbf{A}_{1i} & \mathbf{A}_{2i} \\ \mathbf{A}_{3i} & \mathbf{A}_{4i} \end{pmatrix}}_{\mathbf{A}_i} \underbrace{\begin{pmatrix} \triangle \mathbf{x}_i^{int} \\ \triangle \mathbf{x}_i^{ext} \end{pmatrix}}_{\triangle \mathbf{x}_i} + \begin{pmatrix} 0 \\ \mathbf{B}_i \triangle \mathbf{V}^{ext} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathbf{f}_i^{int}(\mathbf{x}_i^{int}, \mathbf{x}_i^{ext}) \\ \mathbf{f}_i^{ext}(\mathbf{x}_i^{int}, \mathbf{x}_i^{ext}, \mathbf{V}^{ext}) \end{pmatrix}}_{\mathbf{f}_i}$$

- $\mathbf{A}_{1i}$: coupling between interior variables, $\mathbf{A}_{4i}$: coupling between local interface variables,
- $\mathbf{A}_{2i}$ and $\mathbf{A}_{3i}$: coupling between the local interface and the interior variables,
- $\mathbf{B}_i$: coupling between the local interface variables and external interface variables of network sub-domain

Matrix $\mathbf{A}_i$

- General
- Dense
- Size: $2 \div 40$

Injector sub-domain local system

$$\underbrace{\begin{pmatrix} \mathbf{A}_{1i} & \mathbf{A}_{2i} \\ \mathbf{A}_{3i} & \mathbf{A}_{4i} \end{pmatrix}}_{\mathbf{A}_i} \underbrace{\begin{pmatrix} \triangle \mathbf{x}_i^{int} \\ \triangle \mathbf{x}_i^{ext} \end{pmatrix}}_{\triangle \mathbf{x}_i} + \begin{pmatrix} 0 \\ \mathbf{B}_i \triangle \mathbf{V}^{ext} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathbf{f}_i^{int}(\mathbf{x}_i^{int}, \mathbf{x}_i^{ext}) \\ \mathbf{f}_i^{ext}(\mathbf{x}_i^{int}, \mathbf{x}_i^{ext}, \mathbf{V}^{ext}) \end{pmatrix}}_{\mathbf{f}_i}$$

Elimination of *injector* sub-domain interior variables

$$\mathbf{S}_i \triangle \mathbf{x}_i^{ext} + \mathbf{B}_i \triangle \mathbf{V}^{ext} = \tilde{\mathbf{f}}_i$$

- $\mathbf{S}_i = \mathbf{A}_{4i} - \mathbf{A}_{3i}\mathbf{A}_{1i}^{-1}\mathbf{A}_{2i}$: the "local" Schur complement matrix
- $\tilde{\mathbf{f}}_i = \mathbf{f}_i^{ext} - \mathbf{A}_{3i}\mathbf{A}_{1i}^{-1}\mathbf{f}_i^{int}$

Network sub-domain local system

$$\underbrace{\begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix}}_{\mathbf{D}} \underbrace{\begin{pmatrix} \triangle\mathbf{V}^{int} \\ \triangle\mathbf{V}^{ext} \end{pmatrix}}_{\triangle\mathbf{V}} + \begin{pmatrix} 0 \\ \sum_{j=1}^{N-1} \mathbf{C}_j \triangle\mathbf{x}_j^{ext} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathbf{g}^{int}(\mathbf{V}^{int}, \mathbf{V}^{ext}) \\ \mathbf{g}^{ext}(\mathbf{V}^{int}, \mathbf{V}^{ext}, \mathbf{x}^{ext}) \end{pmatrix}}_{\mathbf{g}}$$

- $\mathbf{D}_1$: coupling between interior variables, $\mathbf{D}_4$: coupling between local interface variables,
- $\mathbf{D}_2$ and $\mathbf{D}_3$: coupling between the local interface and the interior variables,
- $\mathbf{C}_j$: coupling between the local interface variables and external interface variables of sub-domain j

Matrix $\mathbf{D}$

- Structurally symmetric
- Sparse
- Size: up to $20\,000 \div 30\,000$

Network sub-domain local system

$$\underbrace{\begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix}}_{\mathbf{D}} \underbrace{\begin{pmatrix} \triangle\mathbf{V}^{int} \\ \triangle\mathbf{V}^{ext} \end{pmatrix}}_{\triangle\mathbf{V}} + \begin{pmatrix} 0 \\ \sum_{j=1}^{N-1} \mathbf{C}_j \triangle\mathbf{x}_j^{ext} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathbf{g}^{int}(\mathbf{V}^{int}, \mathbf{V}^{ext}) \\ \mathbf{g}^{ext}(\mathbf{V}^{int}, \mathbf{V}^{ext}, \mathbf{x}^{ext}) \end{pmatrix}}_{\mathbf{g}}$$

Elimination of *network* sub-domain interior variables

1. Eliminating the interior variables of the network sub-domain to reduce the size of the global reduced system, requires building the local Schur complement matrix $\mathbf{S}_N = \mathbf{D}_4 - \mathbf{D}_3\mathbf{D}_1^{-1}\mathbf{D}_2$ which is in general a **dense** matrix. Thus, destroying the matrix $\mathbf{D}$ sparsity.
2. Not eliminating the interior variables of the network sub-domain but including them in the global reduced system, retains the matrix $\mathbf{D}$ sparsity but increases the size of the reduced system.

Network sub-domain local system

$$\underbrace{\begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix}}_{\mathbf{D}} \underbrace{\begin{pmatrix} \triangle\mathbf{V}^{int} \\ \triangle\mathbf{V}^{ext} \end{pmatrix}}_{\triangle\mathbf{V}} + \begin{pmatrix} 0 \\ \sum_{j=1}^{N-1}\mathbf{C}_j\triangle\mathbf{x}_j^{ext} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathbf{g}^{int}(\mathbf{V}^{int},\mathbf{V}^{ext}) \\ \mathbf{g}^{ext}(\mathbf{V}^{int},\mathbf{V}^{ext},\mathbf{x}^{ext}) \end{pmatrix}}_{\mathbf{g}}$$

Elimination of *network* sub-domain interior variables

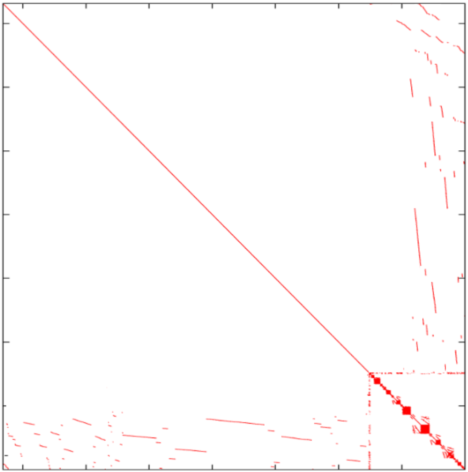
1. Eliminating the interior variables of the network sub-domain to reduce the size of the global reduced system, requires building the local Schur complement matrix $\mathbf{S}_N = \mathbf{D}_4 - \mathbf{D}_3\mathbf{D}_1^{-1}\mathbf{D}_2$ which is in general a **dense** matrix. Thus, destroying the matrix **D** sparsity.
2. Not eliminating the interior variables of the network sub-domain but including them in the global reduced system, retains the matrix **D** sparsity but increases the size of the reduced system.

Global Reduced System

$$\begin{pmatrix} \mathbf{S}_1 & 0 & 0 & \cdots & 0 & \mathbf{B}_1 \\ 0 & \mathbf{S}_2 & 0 & \cdots & 0 & \mathbf{B}_2 \\ 0 & 0 & \mathbf{S}_3 & \cdots & 0 & \mathbf{B}_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{C}_1 & \mathbf{C}_2 & \mathbf{C}_3 & \cdots & \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix} \begin{pmatrix} \triangle \mathbf{x}_1^{ext} \\ \triangle \mathbf{x}_2^{ext} \\ \triangle \mathbf{x}_3^{ext} \\ \vdots \\ \triangle \mathbf{V}^{int} \\ \triangle \mathbf{V}^{ext} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{f}}_1 \\ \tilde{\mathbf{f}}_2 \\ \tilde{\mathbf{f}}_3 \\ \vdots \\ \mathbf{g}^{int} \\ \mathbf{g}^{ext} \end{pmatrix}$$

Global Reduced System

$$\begin{pmatrix} \mathbf{S}_1 & 0 & 0 & \cdots & 0 & \mathbf{B}_1 \\ 0 & \mathbf{S}_2 & 0 & \cdots & 0 & \mathbf{B}_2 \\ 0 & 0 & \mathbf{S}_3 & \cdots & 0 & \mathbf{B}_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{C}_1 & \mathbf{C}_2 & \mathbf{C}_3 & \cdots & \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix}$$



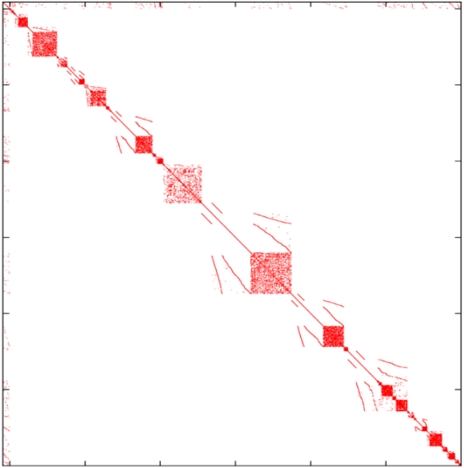
Global Reduced System

$$\begin{pmatrix} \mathbf{S}_1 & 0 & 0 & \cdots & 0 & \mathbf{B}_1 \\ 0 & \mathbf{S}_2 & 0 & \cdots & 0 & \mathbf{B}_2 \\ 0 & 0 & \mathbf{S}_3 & \cdots & 0 & \mathbf{B}_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{C}_1 & \mathbf{C}_2 & \mathbf{C}_3 & \cdots & \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix} \begin{pmatrix} \triangle\mathbf{x}_1^{ext} \\ \triangle\mathbf{x}_2^{ext} \\ \triangle\mathbf{x}_3^{ext} \\ \vdots \\ \triangle\mathbf{V}^{int} \\ \triangle\mathbf{V}^{ext} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{f}}_1 \\ \tilde{\mathbf{f}}_2 \\ \tilde{\mathbf{f}}_3 \\ \vdots \\ \mathbf{g}^{int} \\ \mathbf{g}^{ext} \end{pmatrix}$$

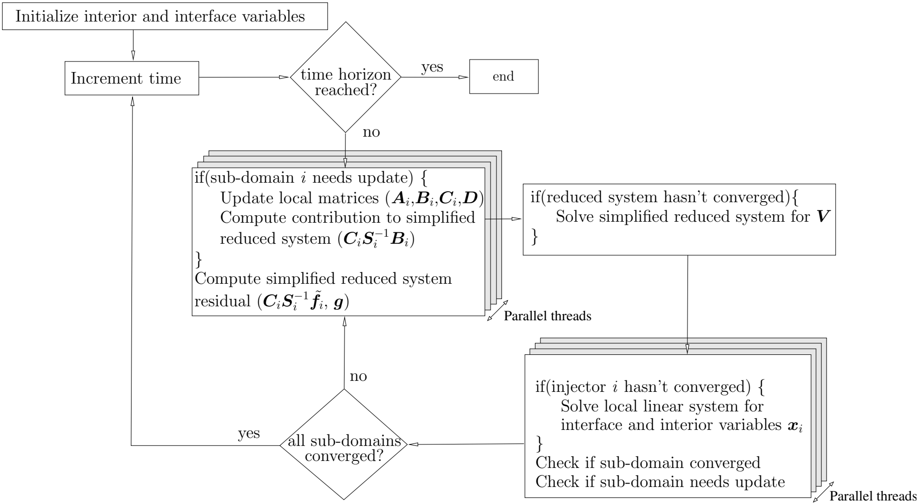
$$\underbrace{\begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 - \sum_{i=1}^{N-1} \mathbf{C}_i \mathbf{S}_i^{-1} \mathbf{B}_i \end{pmatrix}}_{\widetilde{\mathbf{D}}} \underbrace{\begin{pmatrix} \Delta\mathbf{V}^{int} \\ \Delta\mathbf{V}^{ext} \end{pmatrix}}_{\Delta\mathbf{V}} = \underbrace{\begin{pmatrix} \mathbf{g}^{int} \\ \mathbf{g}^{ext} - \sum_{i=1}^{N-1} \mathbf{C}_i \mathbf{S}_i^{-1} \tilde{\mathbf{f}}_i \end{pmatrix}}_{\widetilde{\mathbf{g}}}$$

Global Reduced System

$$\underbrace{\begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 - \sum_{i=1}^{N-1} \mathbf{C}_i \mathbf{S}_i^{-1} \mathbf{B}_i \end{pmatrix}}_{\widetilde{\mathbf{D}}}$$



Solution Algorithm



Implementation on RAMSES[1] platform

- General
  - Standard Fortran 95/2003
  - OpenMP shared-memory parallel programing API
  - HSL41 Sparse Linear Solver
  - BLAS-LAPACK (Intel MKL)
- Tested platforms
  - Intel/AMD, UMA/NUMA, 32/64 bit
  - Windows/Linux
  - Intel/GNU Fortran

[1] "Relaxable Accuracy Multithreaded Simulator of Electric power Systems"

Test case

Power System Features	
# of buses	15226
# of branches	21765
# of injectors	10694
DAE Systems features	
# of Differential states	72293
# of Algebraic states	73946
Total	146239

Disturbance

- Short circuit near a bus lasting for 5 cycles (100ms)
- Cleared by opening a double-circuit line
- System is simulated over a period of 240s

Test case

Power System Features	
# of buses	15226
# of branches	21765
# of injectors	10694
DAE Systems features	
# of Differential states	72293
# of Algebraic states	73946
Total	146239

Disturbance

- Short circuit near a bus lasting for 5 cycles (100ms)
- Cleared by opening a double-circuit line
- System is simulated over a period of 240s

Results: Overview

Machine No	1.	2.	3.	4.
No of Cores	2	4	8	24
Speedup	2.9	3.3	4.1	4.5
Scalability	1.4	1.7	1.8	2.3

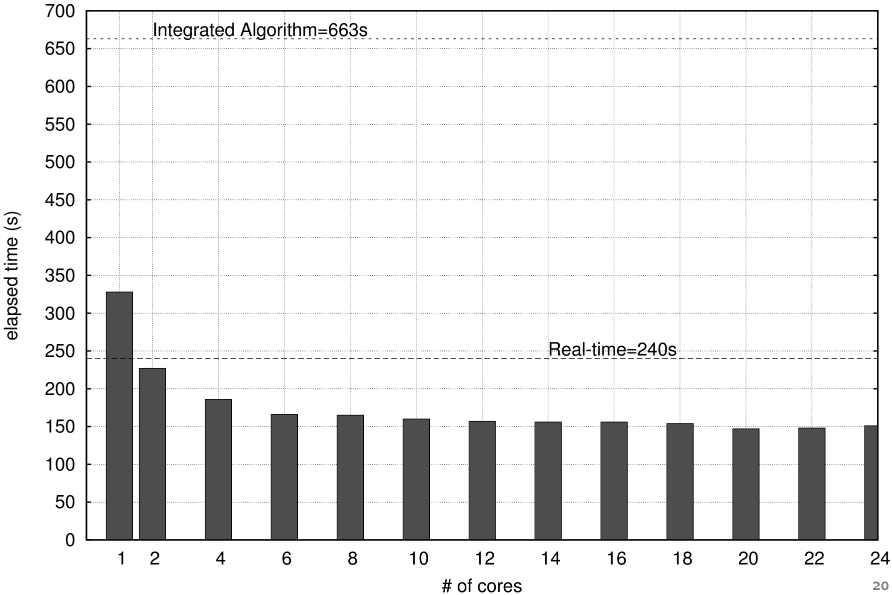
Platforms

1. Intel Core2 Duo CPU T9400 @ 2.53GHz, 3.9GB, Microsoft Windows 7
2. Intel Core i7 CPU 2630QM @ 2.90GHz, 7.7GB, Microsoft Windows 7
3. Intel Xeon CPU L5420 @ 2.50GHz, 16GB, Scientific Linux 5
4. AMD Opteron Interlagos CPU 6238 @ 2.60GHz, 64GB, Debian Linux 6

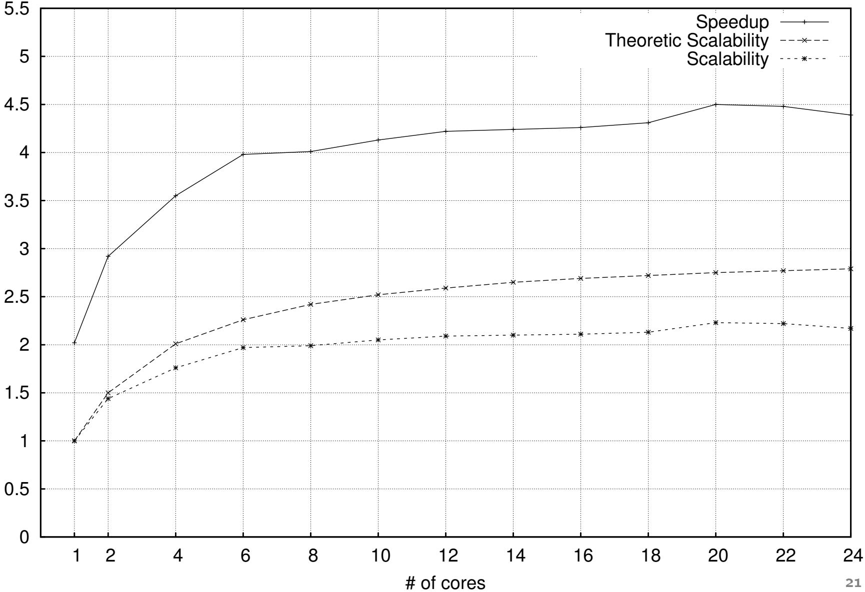
$$Speedup(M) = \frac{time\ elapsed\ integrated\ algorithm\,(1\ core)}{time\ elapsed\ parallel\ decomposed\ algorithm\,(M\ cores)}$$

$$Scalability(M) = \frac{time\ elapsed\ parallel\ decomposed\ algorithm\,(1\ core)}{time\ elapsed\ parallel\ decomposed\ algorithm\,(M\ cores)}$$

Results: Elapsed Time



Results: Speedup and Scalability



Results: Operations

Integrated

Time steps	4833
Integrated Jacobian update	1373
Integrated Newton solutions	10975

Decomposed

Time steps	4833
Global reduced system updates	32
Global reduced system solutions	8565
Local system updates *	40
Local system solutions *	5765

(*) Average per injector sub-domain (N=10694 injector sub-domains)

Conclusions

- Using a DDM for dynamic simulation of electric power systems allows the acceleration of the procedure
  - Numerically, by exploiting the locality of the sub-domain systems and avoiding the unnecessary computations (factorizations, evaluations, solutions)
  - Computationally, by exploiting the parallelization opportunities inherent to these methods
- The proposed domain decomposition algorithm
  - is accurate, DAE system is solved exactly, until global convergence, with no relaxation
  - is robust, applies to general power systems, for a great variety of disturbances without dependency on the exact decomposition
  - exhibits high convergence rate, each sub-problem is solved using a VDHN method with updated and accurate interface values during the whole procedure, convergence does not depend on number of sub-domains

Conclusions

- The implementation
  - is portable, as it can be executed on any platforms supporting the OpenMP API and a standard Fortran compiler
  - can handle general power systems, as no hand-crafted, system specific, optimizations were used
  - exhibits good sequential and parallel performance on a wide range of shared-memory, multi-core computers

The end

Thank you!