Parareal in Time for Dynamic Simulations of Power Systems

Gurunath Gurrala*, Aleksandar Dimitrovski*, Pannala Sreekanth†, Srdjan Simunovic†, Michael Starke*

***Abstract*—In recent years, there have been significant developments in parallel algorithms and high performance parallel computing platforms. Parareal in time algorithm has become popular for long transient simulations (e.g., molecular dynamics, fusion, reacting flows). Parareal is a parallel algorithm which divides the time interval into sub-intervals and solves them concurrently. This paper investigates the applicability of the parareal algorithm to power system dynamic simulations. Preliminary results on the application of parareal for multi-machine power systems are reported in this paper. Two widely used test systems, WECC 3-generator 9-bus system, New England 10-generator 39-bus system, is used to explore the effectiveness of the parareal. Severe 3 phase bus faults are simulated using both the classical and detailed models of multi-machine power systems. Actual Speedup of 5-7 times is observed assuming ideal parallelization. It has been observed that the speedup factors of the order of 20 can be achieved by using fast coarse approximations of power system models. Dependency of parareal convergence on fault duration and location has been observed.**

***Index Terms*—Parareal in time, parallel algorithms, power system dynamics, time parallel**

NOMENCLATURE

δ	Rotor angle (in "electrical radians").
ω_B	Rotor (electrical) synchronous speed.
S_m	Slip speed $= \frac{\omega - \omega_B}{\omega_B}$
T_{mech}, T_{elec}	Mechanical and Electrical torques
D	Damping coefficient
E'_q	Transient induced voltages due to field flux-linkages
i_d, i_q	d, q-axis components of stator current
T'_{do}, T'_{qo}	d, q-axis open circuit time constants
X_d, X'_d, X_q, X'_q	d, q-axis reactances
E_{fd}	Field voltage
$V_t \angle \theta$	Voltage measured at the generator terminal
V_{ref}	Reference voltage
V_S	PSS input
R_a	Armature resistance
v_d, v_q	d, q-axis components of terminal voltage
H	Inertia Constant of machine
I_{BUS}	Vector of bus current injections
Y_{BUS}	Bus admittance matrix including the generator and load impedances.
V_{BUS}	Bus voltages vector

This work was supported under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for the United States Government.

*Electrical and Electronics Systems Research Division, †Computational Sciences and Mathematics Division, Oak Ridge National Laboratory, USA email: gurralag@ornl.gov,dimitrovskia@ornl.gov

Paper submitted to the International Conference on Power Systems Transients (IPST2015) in Cavtat, Croatia June 15-18, 2015

I. INTRODUCTION

Computational complexity of power system dynamic simulations has been increasing steadily due to the growing interconnections, high penetration of renewables, network expansion to meet the growing demand, and increased use of power electronic based transmission controls (HVDC, FACTS, etc.). Maintaining power system stability is essential for secure and reliable power supply to the customers. Dynamic simulations analyze the impact of potential contingencies in a time frame of 10-20 seconds after a disturbance. Solving thousands of non-linear differential-algebraic equations (DAE) in each time-step due to the detailed mathematical modeling of grid components in power system dynamic simulations is a very challenging task. Parallel computing on high performance computing (HPC) architectures and parallelization techniques have become an increasingly attractive choice for researchers [1].

The parallelization techniques can be broadly classified into three categories:

1) Parallelism across the system [2], [3], [4], [5], which partitions the system equations in to various components and distribute the computation over different processors;
2) Parallelism across the method [6], [7], [8], [9], which exploits the parallelism in numerical scheme that is used to solve the equations;
3) Parallelism across the time domain[10], [11], [12], [13], which consists in dividing the integration interval into sub-intervals and solve concurrently over each sub-interval.

It has been realized that the above methods do not give speedups suitable for faster than real-time simulations when they are actually implemented on parallel computing hardware [1]. There are spatial, temporal, hardware, and numerical dependencies that need to be optimized in order to achieve higher speedups in parallelization. In [1] it has been observed that the parallel in time algorithms actually end up solving several smaller time steps in parallel due to inflexibility in the numerical method and difficulty in approximating the initial seeds in each sub-interval which limits the speedup.

"Parareal in Time" algorithm first proposed in [14] provides flexibility in terms of both the numerical method and the initial seed. The parareal method decomposes the time evolution problem into a series of independent evolution problems

on smaller time intervals. A coarse approximation of the trajectory is used to supply the initial seeds and an iterative algorithm based on a predictor corrector approach, that generally converges quite fast [15], leads to real time solution procedures when large number of processors can be availed. This method does extra work for the same simulation but can drastically reduce the wall-clock time of the simulations and that is what is needed for faster than real-time simulations. The method has been tested on a wide range of problems, molecular dynamics, reservoir simulations, partial differential equations, fluid structure computations, etc. [16]. In [17] a coupled parareal and waveform relaxation method has been reported for power systems on simplified power system models but the computational effectiveness of the parareal has not been analyzed.

This paper investigates the applicability of the parareal algorithm to power system dynamic simulations. Preliminary results on the application of parareal for multi-machine power systems is reported in this paper. Two widely used test systems, WECC 3-generator 9-bus system, New England 10-generator 39-bus system, are used to explore the effectiveness of the parareal. Severe 3 phase bus faults are simulated using both the classical and detailed models of multi-machine power systems. Coarse approximations are obtained using large time step and simple numerical techniques.

II. Parareal in Time Algorithm

Parareal algorithm needs decomposition of the time period of integration into sub-intervals. Define the decomposition into N intervals as $T_0 < T_1 < ... < T_n$ and $T_n - T_{n-1} = \Delta T$ as shown in Fig.1. The system to be solved is defined as

$$\dot{u} = f(u, t) \tag{1}$$

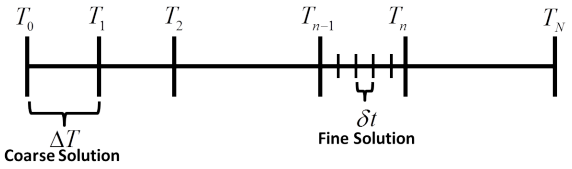

Fig. 1. Decomposition of time into smaller domains

Define a numerical operator $F_{\delta t}$ called as fine operator that operates on some initial state $u(T_{n-1}) = U_{n-1}$ and approximates the solution to (1) at time $T_{n-1} + \Delta T$, using some small time step $\delta t << \Delta T$. A coarse propagator is defined as $G_{\Delta T}$ which also operates on some initial state $u(T_{n-1}) = U_{n-1}$ and approximates the solution to (1) at time $T_{n-1} + \Delta T$, but now using a time-step ΔT. The fine operator ($F_{\delta t}$) operates in parallel on each of the sub-intervals. The coarse operator ($G_{\Delta T}$) produces initial seeds sequentially for all the sub-intervals. Following the concurrent solution of $n = 1, 2, ...N$ initial value problems using fine operator, the parareal method corrects the sequential coarse predictions using the following correction step

$$U_n^k = G_{\Delta T}(U_{n-1}^k) + F_{\delta t}(U_{n-1}^{k-1}) - G_{\Delta T}(U_{n-1}^{k-1}) \tag{2}$$

1: **procedure** PARAREAL( )
2: $U_0^0 \leftarrow \tilde{U}_0^0 \leftarrow u_0$ ▷ Iteration 0
3: **for** $n = 1$ to N **do**
4: $\tilde{U}_n^0 \leftarrow G_{\Delta T}\left(\tilde{U}_{n-1}^0\right)$ ▷ Initial prediction
5: $U_n^0 \leftarrow \tilde{U}_n^0$
6: **end for**
7: $U_0^1 \leftarrow u_0$;
8: **for** $k = 1$ to K_{max} **do**
9: **for** $n = 1$ to N **do**
10: $\hat{U}_n^{k-1} \leftarrow F_{\delta t}\left(\tilde{U}_{n-1}^{k-1}\right)$ ▷ Parallel Fine Step
11: **end for**
12: **for** $n = 1$ to N **do** ▷ Sequential Coarse Step
13: $\tilde{U}_n^k \leftarrow G_{\Delta T}\left(U_{n-1}^k\right)$ ▷ Prediction
14: $U_n^k \leftarrow \tilde{U}_n^k + \hat{U}_n^{k-1} - \tilde{U}_n^{k-1}$ ▷ Correction
15: **end for**
16: **if** $\left|U_n^k - U_n^{k-1}\right| < \epsilon \forall n$ **then** ▷ Exit on Convergence
17: **BREAK**
18: **end if**
19: **end for**
20: **end procedure**

Fig. 2. Pseudo code for Parareal implementation [16]

The pseudo code for implementation of the parareal algorithm is shown in Fig.2. In the pseudo code the notation $\tilde{U}$ and $\hat{U}$ are used for states obtained from coarse solution and fine solution respectively. The notation U is used for corrected coarse solution.

In a sub-interval T_{n-1}, T_n, define the left end values as the state values at time T_{n-1} and the right end values as the state values at time T_n. The algorithm in Fig.2 initially (lines 3-6) calculates the left end values (seeds) of all the sub-intervals with time step ΔT. Then in first parareal iteration (setps 9-11) the fine solver computes the right end values of all the sub-intervals using these seeds with a time step δt. This step will be done in parallel for all the sub-intervals. Now the coarse operator predicts the right end value of the first sub-interval using the corrected left end value (Note: there is no correction for the first interval left end value because they are the initial values). Then the predicted right end value from coarse solver, the right end value from the fine solver and the right end value obtained from coarse solver in the previous iteration, will be used to obtain the corrected right end value for the first sub-interval as shown in (2). The process is repeated sequentially for all sub-intervals. The parareal iterations will be repeated until the difference between corrected values of two successive iterations in all the sub-intervals is minimized to a desired level. Note that the first interval always gets corrected to the actual value which is the output of the fine solver in first iteration and this is true for subsequent intervals through the iterations.

Assuming that the fine solver takes ζ time for solving 1 sub-interval, the whole problem can be solved by using the fine solver sequentially for N intervals in $N \times \zeta$. If the iterations converge in $K < N$ iterations i.e. K-times the fine solver is used, then the time taken for the solution with parareal is $K \times \zeta$. This assumes that the coarse solver's time is negligible. The ideal speedup that can be obtained using parareal is defined as $(N \times \zeta)/(K \times \zeta) = N/K$. So the speedup depends on the number of iterations, number of coarse time intervals and the time taken for the coarse solution [16]. Other factors related to the parallelization are assumed negligible. The choice of the coarse operator in parareal algorithm is very important in

order to obtain fast convergence. If the coarse operator is too inaccurate, a lot of iterations are needed for convergence and can also cause divergence. If it is expensive then the parallel efficiency (time-to-solution) gets affected. Coarse solvers can be constructed in several different ways and few of them are listed below:

1) Using a bigger step-size than for the fine operator.
2) Using another time-stepping method that has a lower order and which is therefore faster.
3) Simplifying the underlying physics there by using a simpler model.
4) Using an iterative solver with limited number of iterations.

This paper investigates the first two approaches. In the following subsections the parareal algorithm operation is introduced using two examples. These examples confirm the implementation of parareal on dynamical systems similar to simple models of power systems, before going to more complex power system models.

A. Example: One dimentional system

Consider the one dimensional example [16] below

$$\dot{y} = sin(t)y(t) + t \tag{3}$$

$$y(0) = 1, t \in [0, 14]$$

Runge-Kutta 4th order (RK-4) numerical method is used for both the coarse and fine solvers. The total time 14s is divided into 6 intervals. So the coarse time step is $\Delta T = 2.33s$. Each interval is sub-divided into 30 fine steps. So the fine time step is $\delta t = 0.0777s$. Fig.3 shows the simulations results.

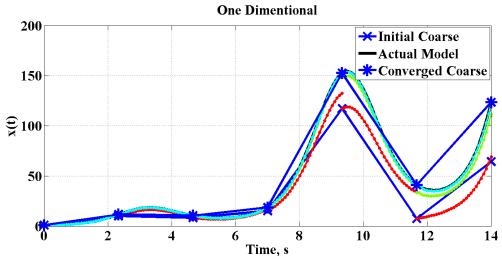

Fig. 3. One dimensional example using parareal

The solid lines with circles show the fine solution. The initial coarse solution is marked with '$\times$' and the converged coarse solution is marked with '$*$'. It can be observed that the fine and coarse solutions move towards the actual solution (solid line). Eventually, after 3 fine iterations the coarse solution converges to actual solution. The results conform to the results in [16] which uses forward Euler method with different time division. The ideal speedup achieved in this case is 2.

B. Example: Two dimensional system

Consider the following two dimensional system called as Van Der Pol's equations:

$$\dot{y_1} = y_2; \tag{4}$$

$$\dot{y_2} = (1 - y_1^2)y_2 - y_1; \tag{5}$$

It is a very stiff system. Here also Runge-Kutta 4th order numerical method is used for both the coarse and fine solvers. The total time 20s is divided into 30 intervals. So the coarse time step is $\Delta T = 0.6667s$. Each interval is sub-divided into 100 fine steps. So the fine time step is $\delta t = 0.0067s$. Fig.4 shows the simulation results. The initial coarse and final

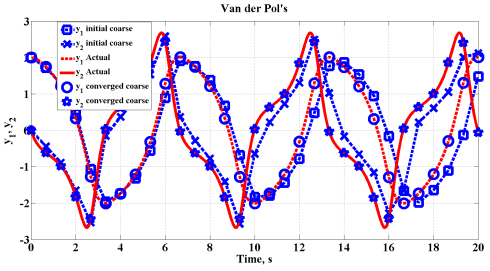

Fig. 4. Van der Pol's example using parareal

coarse solutions are distinctly marked. The parareal algorithm converged in 8 iterations. The speedup achieved in this case is 3.75. It can be observed that the coarse solution converges to the actual solution.

III. Application to Power Systems

In this section the parareal algorithm implementation on multi-machine power systems is discussed. The parareal is applied to classical models of power systems first and later extended to detailed models. The classical model contains only two differential equations with algebraic network constraints. So the parareal implementation on the two-dimensional system is extended to classical models.

A. Classical Modeling of power systems

The synchronous machine in classical power system model is represented as a voltage source behind transient reactance with constant flux linkages and constant mechanical torque. The dynamic equations governing the classical model are as follows.

$$\dot{\delta} = \omega_B S_m \tag{6}$$

$$\dot{S}_m = \frac{1}{2H}\left[-DS_m + T_{mech} - T_{elec}\right] \tag{7}$$

$$i_q = \frac{1}{R_a^2 + X_d'^2}\left[R_a\left(E_q' - v_q\right) - X_d' v_d\right] \tag{8}$$

$$T_{elec} = E_q' i_q \tag{9}$$

The network algebraic equations are given by

$$I_{GEN} = \frac{E'_q}{R_a + jX'_d} e^{(j\delta)} \tag{10}$$

$$I_{BUS} = Y_{BUS} V_{BUS} \tag{11}$$

The Y_{BUS} matrix includes the generator reactance X_d' and the equivalent impedance of the loads. Excitation and governor controls are neglected [18].

1) Results: WECC 3-generator 9-bus system: RK-4 is used for simulating the classical power system dynamic equations. Western Electricity Coordinating Council (WECC) 3-generator 9-bus equivalent system is a widely used test system. Three phase faults are the most severe conditions on power networks for which the system displays nonlinear behavior. So three phase faults are simulated.

In this case, t the total simulation time of 10s is divided into 60 intervals. The coarse time step is $\Delta T = 0.1667s$. Each sub-interval is divided into 100 fine intervals. So the fine time step is $\delta t = 0.00167s$. It has been observed that for coarse intervals fewer than 60 the parareal algorithm started diverging after few iterations. This is the maximum time step that could be used for the coarse RK4 solver for this example. Fig.5 shows the rotor angle response of generator 2 with respect to the generator 1, δ_{21}, for a 3-phase fault of 6 cycles duration at bus 5. The fault is assumed to self clear without tripping any lines. Disturbance is simulated at $t = 0s$. The figure contains the actual model response,

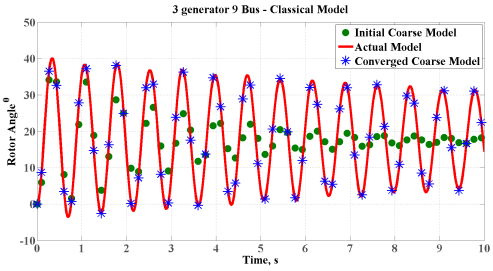

Fig. 5. WECC δ_{21}, 3-phase fault at bus 5, 6 cycles, Classical model

initial and converged coarse responses. It can be observed that the initial coarse response is significantly different from the actual response. The parareal converged in 6 iterations. The converged coarse response approximates the actual response very well. The ideal speedup in this case is 10. Actual CPU time taken for MATLAB® calculations on a laptop with core i7 2.9GHZ, 16GB RAM Quad-Core processor is shown in Table.I for various steps. If ideal parallelization is assumed for fine evaluations then the speedup achieved is 5.27 in this case.

TABLE I
CPU TIME, WECC, CLASSICAL MODEL

Coarse Initial,s	iteration k	Fine Evaluation,s	if Parallel,s	Coarse Step,s
	1.00	1.34	0.023	0.0147
	2.00	1.32	0.023	0.0147
	3.00	1.30	0.024	0.0135
	4.00	1.27	0.023	0.0134
	5.00	1.25	0.023	0.0129
	6.00	1.22	0.023	0.0131
0.04			**Total 0.134s**	**Total 0.082s**
Total Time Parareal	**0.2512s**			
Fine N intervals	**1.323s**			
Speedup	**5.27**			

2) Results: New England 10-generator 39-bus: New England 10-generator 39-bus system is another widely used system for validating the stability controls. It has 6181MW total generation and 6124MW load.

The data for the system is obtained from [18]. 3-phase fault of 6 cycles duration at bus 5 is simulated. The fault is cleared without tripping the associated line. Fig.6 shows the rotor angle response of generator 5 w.r.t. the generator 1,δ_{51}. Disturbance is simulated at $t = 0s$. The total simulation time of 10s is divided into 60 intervals. It has been observed that for time intervals fewer than 60, more number of iterations are required for convergence. The coarse time step is $\Delta T = 0.1667s$. Each sub-interval is divided into 100 fine intervals. So, the fine time step is $\delta T = 0.00167s$. The parareal converged in 5 iterations. In this case, the ideal speedup achieved is 12. The actual CPU time calculations are shown in Table.II. Actual speedup achieved is 4.74.

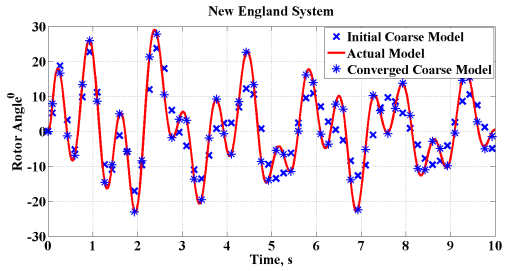

Fig. 6. New England δ_{51}, 3-phase fault at bus 5, 6 cycles

TABLE II
CPU TIME, NEW ENGLAND, CLASSICAL MODEL

Coarse Initial,s	iteration k	Fine Evaluation,s	if Parallel,s	Coarse Step,s
	1.00	0.827	0.014	0.018
	2.00	0.972	0.016	0.017
	3.00	0.801	0.014	0.016
	4.00	0.781	0.014	0.016
	5.00	0.773	0.014	0.016
0.04			**Total 0.072**	Total **0.082**
Total Time Parareal	**0.197s**			
Fine N Intervals	**0.935s**			
Speeed Up	**4.74**			

B. Detailed Modeling of power systems

This section investigates the application of parareal to detailed models of power systems. In this study IEEE model 1.1 as shown in Fig.7 is considered for synchronous machine with a IEEE Type1 excitation system and 1^{st} order turbine-governor models [18].

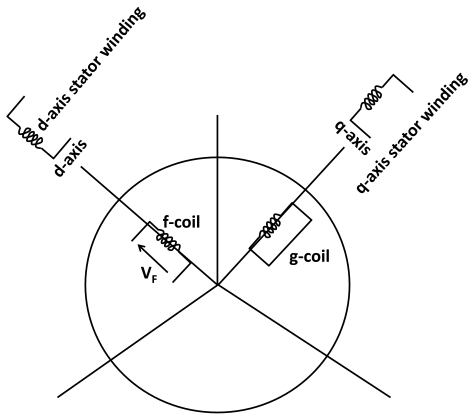

Fig. 7. Synchronous Machine: IEEE Model 1.1

Generator Mechanical Equations :

$$\dot{\delta} = \omega_B S_m \tag{12}$$

$$\dot{S}_m = \frac{1}{2H}\{T_{mech} - T_{elec} - DS_m\} \quad (13)$$

q,d-axis flux linkage equations :

$$\dot{E}'_q = \frac{1}{T'_{do}}\{-E'_q + (X_d - X'_d)i_d + E_{fd}\} \quad (14)$$

$$\dot{E}'_d = \frac{1}{T'_{qo}}\{-E'_d - (X_q - X'_q)i_q\} \quad (15)$$

Generator Electrical Torque Equation :

$$T_{elec} = E'_q i_q + (X'_d - X_q)i_d i_q \quad (16)$$

The stator algebraic equations are given by

$$\begin{aligned} E'_q + X'_d i_d - R_a i_q &= V_q \\ E'_d - X_q i_q - R_a i_d &= V_d \end{aligned} \quad (17)$$

The excitation system and turbine-governor models are shown in Fig.8 and Fig.9 respectively.

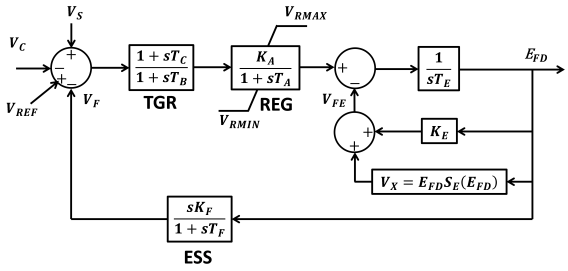

Fig. 8. IEEE Type1 Excitation System

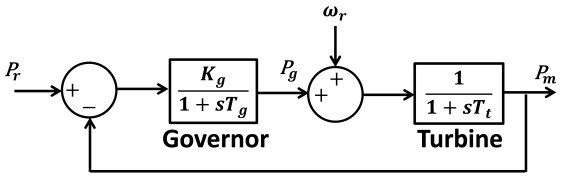

Fig. 9. 1^{st} order Turbine-Governor System

The detailed system model results in 11 state variables for each generator. Parareal is implemented using these models and the results are presented in the following subsections.

1) Results: WECC 3-generator 9-bus : For the detailed models, implicit trapezoidal method (TRAPZ) with a midpoint predictor [18] is used as a coarse solution with a large time step. For fine solution RK4 method is used. The total simulation time of 10s is divided into 400 intervals. The coarse time step is $\Delta T = 0.025s$. Each sub-interval is divided into 50 fine intervals. So, the fine time step is $\delta t = 0.0005s$. It has been observed that for coarse intervals fewer than 400 the parareal algorithm started diverging after few iterations. This is the maximum time step that could be used for the coarse trapezoidal solver for this example. Fig.10 shows the rotor angle response δ_{21} for a 3-phase fault of 6 cycles duration at bus 5. The fault is assumed to self clear without tripping any lines. Disturbance is simulated at $t = 0s$. It can be

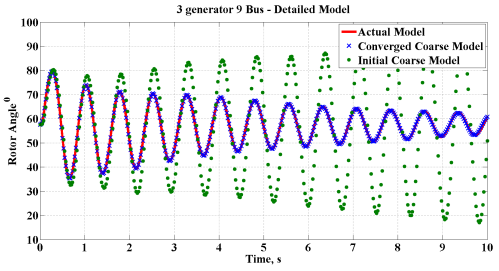

Fig. 10. δ_{21} response, 3-phase fault, 6 cycle, detailed model

observed from the Fig.10 that the coarse response has unstable oscillations. The parareal converged in 8 iterations. Theoretical speedup for this case is 50. The actual CPU time calculations are shown in Table.III. The actual speedup achieved is 6.28 in this case. It can be observed that significant time is spent in coarse evaluation. If the coarse solution is made (assume) at least 10 times faster then from the table one can calculate the speedup as 19.81 which could be significant improvement. This shows that fast coarse models can improve the speedup of parareal.

TABLE III
CPU TIME, WECC, DETAILED MODEL

Coarse Initial,s	iteration k	Fine Evaluation,s	if Parallel,s	Coarse Step,s
	1.00	9.78	0.02	0.15
	2.00	9.89	0.02	0.15
	3.00	9.81	0.02	0.14
	4.00	9.83	0.02	0.15
	5.00	9.76	0.02	0.14
	6.00	9.82	0.02	0.14
	7.00	9.69	0.02	0.14
	8.00	9.83	0.03	0.15
0.17			**Total 0.20**	**Total 1.16**
Total Time Parareal	**1.53s**			
Fine for N intervals	**9.61s**			
Speeed Up	**6.28**			

When the fine solver is changed to trapezoidal with other settings remaining the same, the speedup is dropped to 4.7. This is because RK4 is expensive than trapezoidal.

2) Results: 10-generator 39-bus: For this system again implicit trapezoidal method (TRAPZ) with a midpoint predictor is used as a coarse solution and RK4 method is used for fine solution. The total simulation time 10s is divided into 400 intervals. The coarse time step is $\Delta T = 0.025s$. Each sub-interval is divided into 50 fine intervals. So, the fine time step is $\delta T = 0.0005s$. For this system again for coarse intervals fewer than 400 the parareal algorithm started diverging after a few iterations. Fig.11 shows the rotor angle response δ_{21} for

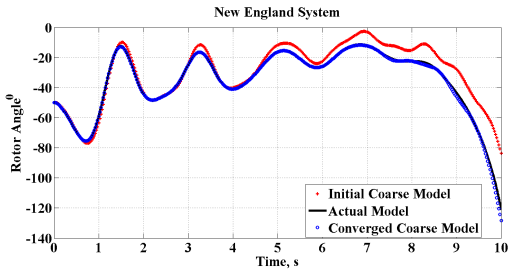

Fig. 11. δ_{21} response, 3-phase at bus 5, 6 cycles, detailed model

a 3-phase fault of 6 cycles duration at bus 5. Disturbance is

simulated at $t = 0s$. The parareal converged in 7 iterations. Ideal speedup for this case is 57.14. The actual CPU time is shown in Table.IV. The actual speedup achieved is 7.08. Here also significant time is spent in coarse solution. When the fine solver is changed to trapezoidal with other settings remaining the same the speedup is dropped to 5.27 as trapezoidal is cheaper than RK4. For this example also if the coarse model is made (assume) 10 times faster then 20.78 speedup could be achieved. This shows that the parareal can be promising if simpler coarse models are used for power systems. One can hope that parareal coupled with spatial and numerical method parallelism could achieve faster than real time simualtins. This is the focus of ongoing research.

TABLE IV
CPU TIME, NEW ENGLAND SYSTEM, DETAILED MODEL

Coarse Initial,s	**iteration k**	**Fine Evaluation,s**	**if Parallel**	**Coarse Step,s**
	1.00	10.21	0.03	0.15
	2.00	10.33	0.03	0.15
	3.00	10.32	0.03	0.15
	4.00	10.31	0.03	0.15
	5.00	10.22	0.03	0.15
	6.00	10.22	0.03	0.15
	7.00	10.20	0.03	0.15
0.20			**Total 0.18**	**Total 1.04**
Total Time Parareal	**1.42s**			
Fine for N intervals	**10.06s**			
Speeed Up	**7.08**			

In this paper only representative results are shown which confirm the parareal applicability to power system simulations. The speedup results are summarized in Table.V.

TABLE V
SUMMARY OF THE RESULTS

Model	Coarse-Fine	System	Achieved	Ideal Speedup (N/K)
	RK4-RK4	3-Gen 9-Bus	5.27	60/6=10
Classical	RK4-RK4	10-Gen 39-Bus	4.74	60/5=12
	TRAP-TRAP	3-Gen 9-Bus	4.7	400/8= 50
	TRAP-RK4	3-Gen 9-Bus	6.28	400/8=50
	TRAP-TRAP	10-Gen 39-Bus	5.27	400/7=57
Detailed	TRAP-RK4	10-Gen 39-Bus	7.08	400/7=57

Several simulations have been carried out varying the fault duration and fault location. It has been observed that the actual speedup in terms of CPU time varies with the fault location and duration. For long duration faults the speedup is less because the waveforms involve significant dynamics. Especially, the generators close to the fault location needs more iterations to converge than the far away generators. It is also observed that the parareal converges quickly for short duration simulations. This is consistent with the parareal behavior on other systems [16]. The parareal convergence is heavily dependent on the coarse model accuracy. It is also evident from the time calculations that the parareal speed up is directly coupled to the coarse model speed.

The MATLAB® codes used in the experiments are not optimized for performance. Future research focuses on fast coarse models development, implementation on $C/C++$ languages for actual parallel implementation and code optimization.

IV. CONCLUSION

This paper proposes parareal in time algorithm for the dynamic simulations of large power systems. Parareal algorithm seems to be a promising approach which can achieve higher speedup ratios. The algorithm is simple to implement and has flexibility to experiment with various numerical methods. Following are the contributions of the paper:

- Parareal algorithm is implemented for power system dynamic simulations.
- Parareal is verified on classical and detailed models of multi-machine power systems.
- Actual Speedup of 5-7 times is observed assuming ideal parallelization. It has been observed that the speedup factors of the order of 20 can be achieved by using fast coarse approximations of power system models
- Dependency of parareal convergence on fault duration and location has been observed.
- Need for fast coarse model development is identified to achieve higher speedups.

REFERENCES

[1] D. Koester, S. Ranka, and G. Foz, "Power systems transient stability - a grand computing challenge," School of Computer and Information Science and The Northeast Parallel Architectures Center (NPAC), NPAC Technical Report SCCS-549, August 1992.

[2] G. Kron, "Diakoptics-piecewise solutions of large systems," *General Electric and Also Published by McDonald*, vol. 158/162, 1963.

[3] S. Taoka, S. Abe, and Takeda, "Fast transient stability solution using an array processor," *IEEE Trans. Power Apparatus and Systems*, vol. PAS-102, no. 12, December 1983.

[4] S. Esmaeili and S. Kouhsari, "A distributed simulation based approach for detailed and decentralized power system transient stability analysis," *Electric Power Systems Research*, vol. 77, p. 673684, 2007.

[5] M. Tomim, J. Marti, and L. Wang, "Parallel solution of large power system networks using the multi-area thevenin equivalents (mate) algorithm," *Electrical Power and Energy Systems*, vol. 31, p. 497503, 2009.

[6] M. L. Scala, M. Brucoli, F. Torelli, and M. Trovato, "A gauss-jacobi-block-newton metiiod for parallel transient stability analysis," *IEEE Trans. Power Systems*, vol. 5 , No. 4, November 1990, no. 4, pp. 1168–1177, November 1990.

[7] A. Bose, "Parallel processing in dynamic simulation of power systems," *Sadhana, Indian Academy of Sciences*, vol. 18, no. 5, pp. 815–827, September 1993.

[8] L. Hou and A. Bose, "Implementation of the waveform relaxation algorithm on a shared memory computer for the transient stability problem," *IEEE Trans. Power Systems*, vol. 12, no. 3, p. 10531060, August 1997.

[9] V. Jalili-Marandi and V. Dinavahi, "Instantaneous relaxation based real-time transient stability simulation," *IEEE Trans.Power Systems*, vol. 24, no. 3, pp. 1327–1336, August 2009.

[10] F. Alvarado, "Parallel solution of transient problems by trapezoidal integration," *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-98, no. 3, May/June 1979.

[11] M. L. Scala, R. Sbrizzai, and F. Torelli, "A pipelined-in-time parallel algoiiitiim for transient stability analysis," *IEEE Trans. on Power Systems*, vol. 6, no. 2, pp. 715–722, May 1991.

[12] M. La Scala, R. Sbrizzai, F. Torelli, and P. Scarpellini, "A tracking time domain simulator for real-time transient stability analysis," *IEEE Tran. Power Systems*, vol. 13, no. 3, pp. 992–998, August 1998.

[13] F. Wang, "Parallel-in-time relaxed newton method for transient stability analysis," *IEE Proc.-Gener. Transm. Distribution*, vol. 145, No. 2, March, no. 2, pp. 155–159, March 1998.

[14] J. L. Lions, Y. Maday, and G. Turinici, "A parareal in time discretization of pdes," *C.R. Acad Sci. Paris, Serie I*, vol. 332, p. 661668, 2001.

[15] Y. Maday, E. nquist, and G. Staff, "The parareal-in-time algorithm: Basics, stability analysis and more," 2006, not formally published.

[16] A. S. Nielsen, "Feasibility study of the parareal algorithm," Masters Thesis, Technical University of Denmark, Informatics and Mathematical Modelling Building 321, DK-2800 Kongens Lyngby, Denmark, 2012.

[17] T. Cadeau and F. Magoules, "Coupling parareal and waveform relaxation methods for power systems," in *Electrical and Control Engineering (ICECE), 2011 International Conference on*, 2011, pp. 2947–2950.

[18] K.R.Padiyar, *Power system dynamics Stability and Control*. B.S.Publications, 2002.