Computational and Economic Limitations of Dispatch Operations in the Next-Generation Power Grid

Victor M. Zavala and Audun Botterud
Mathematics and Computer Science Division and Decision and Information Sciences Division, Argonne National Laboratory
9700 S Cass Avenue, Argonne, IL 60439, vzavala@mcs.anl.gov, abotterud@anl.gov

***Abstract*—We study the interactions between computational and economic performance of dispatch operations under highly dynamic environments. In particular, we discuss the need of extending the forecast horizon of the dispatch formulation in order to anticipate steep variations of renewable power and highly elastic loads. We present computational strategies to solve the increasingly larger optimization problems in real-time. To illustrate the developments, we use a detailed dispatch model of the entire Illinois interconnect with out-of-state wind generation.**

I. Introduction

The next-generation power grid will be operated under highly dynamic regimes including distributed storage and co-generation, large-scale renewable generation, and highly elastic loads. These resources act as fast disturbances that need to be balanced out in the grid in real-time. Wind power ramping events are already demanding more proactive and fast operational systems [12]. This is illustrated in Fig. 1, where we present typical profiles for the total load and wind power at different adoption levels. As can be seen wind power supply can fluctuate by an order of magnitude in a few minutes. Similar trends are expected for the loads under smart grid environments.

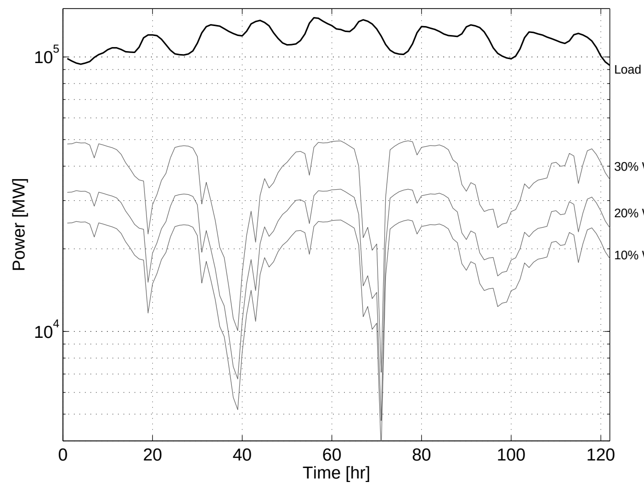

Fig. 1. Snapshot of total load and increasing levels of wind power.

Economic dispatch (ED) is one of the most important operational tasks in the power grid. The ED system updates the output levels of the committed generators to match the load demands in a cost-optimal manner. The solution has to satisfy both transmission and generation ramping constraints [17]. This task is of great importance since it clears the real-time market and sets the locational marginal prices (LMPs) [14]. The ED system is currently designed using forecast horizons on the order of a couple of hours with a time resolution (i.e.; time steps) of a few minutes. Under stable operations, this horizon might be sufficient to capture load trends. However, in the presence of steep trends such as those observed during wind ramping events, the performance of the ED system might deteriorate if it does not have enough foresight and resolution. This can lead, for instance, to load and wind curtailment.

Increasing the foresight and resolution of the ED problem comes at the expense of additional computational complexity. The problem is usually cast as a large-scale linear or quadratic optimization problem [18]. The main source of complexity is the inherent transmission network that has to be accounted for at each time step in the horizon. In addition, coupling due to ramping constraints can render the problem intractable for even a few time steps. While state-of-the-art optimization solvers are currently able to solve large ED problems, scalability bottlenecks are still a concern.

In this work, we analyze the interactions between economic and computational performance of ED. We first analyze the effect of increasing the horizon of the problem and discuss the associated computational implications. In order to do this, we present a detailed model for the Illinois interconnect. We compare the performance of barrier and simplex optimization solver. In particular, we identify scalability bottlenecks associated with the core linear algebra kernels of the solvers. We exploit the advantages of both barrier and simplex strategies, and notions of model predictive control to derive effective warm-starting strategies. The proposed developments can enable the implementation of detailed ED formulations under tight solution time constraints.

II. Economic Dispatch Problem

We consider the traditional, social welfare formulation where the objective is to minimize generation costs subject to direct-current (DC) transmission and generation ramping constraints. The ED problem has the form:

$$\min \sum_{k=\ell}^{\ell+N} \sum_{j\in\mathcal{G}} c_j \cdot G_{k,j} \tag{1a}$$

$$\text{s.t. } G_{k+1,j} = G_{k,j} + \Delta G_{k,j},\ k=\ell,...,\ell+N-1, j\in\mathcal{G} \tag{1b}$$

$$\sum_{(i,j)\in\mathcal{L}_j} P_{k,i,j} + \sum_{i\in\mathcal{G}_j} G_{k,i} + \sum_{i\in\mathcal{W}_j} (W_{k,i} - \Delta W_{k,i}) = \sum_{i\in\mathcal{D}_j} D_{k,i},\ k=\ell,...,\ell+N, j\in\mathcal{B} \tag{1c}$$

$$P_{k,i,j} = b_{i,j}(\theta_{k,i} - \theta_{k,j}), k=\ell,...,\ell+N, (i,j)\in\mathcal{L} \tag{1d}$$

$$G_j^{min} \le G_{k,j} \le G_j^{max},\ k=\ell,...,\ell+N, j\in\mathcal{G} \tag{1e}$$

$$\Delta G_j^{min} \le \Delta G_{k,j} \le \Delta G_j^{max},\ k=\ell,...,\ell+N-1, j\in\mathcal{G} \tag{1f}$$

$$P_{i,j}^{min} \le P_{k,i,j} \le P_{i,j}^{max},\ k=\ell,...,\ell+N, (i,j)\in\mathcal{L} \tag{1g}$$

$$\theta_j^{min} \le \theta_{k,j} \le \theta_j^{max},\ k=\ell,...,\ell+N, j\in\mathcal{B} \tag{1h}$$

$$0 \le \Delta W_{k,j} \le W_{k,j},\ k=\ell,...,\ell+N, j\in\mathcal{W} \tag{1i}$$

Here, ℓ is the real time index, k is the horizon time index, and N is the number of time steps in the horizon. The sets $\mathcal{B},\mathcal{L},\mathcal{G},\mathcal{W}$, and $\mathcal{D}$ are the buses, lines, thermal generators, wind generators, and load demands, respectively. Subindexed sets in j represent subsets at bus j. The problem *variables* are the thermal generation levels $G_{k,j}$, the ramp increments $\Delta G_{k,j}$, the power flows $P_{k,i,j}$, the bus angles $\theta_{k,j}$, and the wind curtailment flows $\Delta W_{k,j}$. The problem *data* are the load demands $D_{k,j}$ and wind power flows $W_{k,j}$. The multipliers of the network constraint (1c) are the LMPs $\lambda_{k,j}$ at time k and bus j.

III. Forecast Horizon and Cost Performance

At each time instant ℓ, the ED problem is solved using the observed and forecasted data for the load demands and wind flows. The observed flows are $W_{\ell,j}$, $D_{\ell,j}$ while $W_{\ell+i,j}$, $D_{\ell+i,j}, i=1,...,N-1$ are forecasted. The solution of this problem sets the generator levels for the current time step $G_{\ell,j}$ with associated cost $\varphi_\ell^{MH} = \sum_{j\in\mathcal{G}} G_{\ell,j}$, and the LMPs $\lambda_{j,\ell}, j\in\mathcal{B}$. At the next time step $\ell+1$, the *true* loads and wind power flows are observed and a new ED problem is solved to obtain $G_{\ell+1,j}$ and $\varphi_{\ell+1}^{MH}$. To manage the forecast horizon, it is possible to use either a moving or a shrinking horizon approach. These approaches have advantages and disadvantages from computational and implementation perspectives.

A. Moving Horizon

In the moving horizon approach, the horizon at time ℓ is $k=\ell,...,\ell+N$. At the next step, the horizon is shifted forward in time $k=\ell+1,...,\ell+N+1$. This approach has the computational advantage that the problem size remains *fixed*. The horizon length is usually constrained by the solution time which must match the time resolution (e.g.; two to five minutes). From an implementation perspective, the horizon needs to be shrunk towards the end of the *bidding cycle* where the unit commitment decisions are made. The cycle is usually 24 hours. Extending the horizon over the bidding cycle can introduce a significant amount of uncertainty since the minimum power outputs are determined after bidding. We thus have that if the bidding cycle contains T time steps and $\ell=0$ at the beginning of the cycle, the horizon satisfies:

$$N = \begin{cases} N & \text{if } \ell+N<T \\ T-\ell & \text{if } \ell+N\ge T \end{cases} \quad \ell=0,...,T-1. \tag{2}$$

Warm-starting moving horizon problems is complicated because of the horizon *shifting*. Reusing the solution at ℓ to initialize problem at $\ell+1$ is somewhat beneficial but inconsistent. This limits the achievable solution times.

B. Shrinking Horizon

In the shrinking horizon approach, the problem is solved for the entire bidding cycle $N=T$ and this is updated at each step by dropping only the first element of the horizon such that $N=T-\ell,\ \ell=0,...,T-1$. The advantages of this approach is that it is consistent with the bidding cycle and that it satisfies Bellman's principle of optimality [1]. Bellman's principle states that, *under perfect foresight*, the solution profile obtained with horizon $k=\ell,...,T$ is optimal for the problem with shrunk horizon $k=\ell+1,...,T$. A disadvantage of this approach is that the bidding cycle can be extremely large compared to the time resolution of the problem. For instance, if we use a resolution of 5 minutes, a bidding cycle will contain $T=288$ steps. However, as will be shown in Section IV, Bellman's principle can be exploited to derive effective warm-starting strategies that can enable the implementation of ED problems with high time resolutions and long horizons.

The perfect foresight problem with $N=T$ gives the *best possible* cost trajectory $\varphi_\ell,\ \ell=0,...,T$ over the bidding cycle. For the moving horizon approach, we have that as $N\to T$, the moving horizon cost approaches the optimal cost. The convergence rate is problem dependent and thus difficult to establish *a priori*. However, this property can be exploited to derive *hybrid* moving-shrinking horizon strategies that do not need to set $N=T$ and can still exploit Bellman's principle to generate warm-starts. This approach has been proposed in the model predictive control literature [6], [19].

IV. Computational Issues

We can write the ED problem (1) in the general form:

$$\min c^T x \tag{3a}$$

$$\text{s.t. } A\,x = b \tag{3b}$$

$$x \ge 0. \tag{3c}$$

Where $x\in\Re^n$ is the variable vector, $A\in\Re^{m\times n}$ is the Jacobian matrix, $c\in\Re^n$ is the cost vector and $b\in\Re^m$ is the data. There exist highly efficient solvers that can be used to solve large-scale problems of this form. To name a few, we have the commercial solvers Cplex from IBM, Gurobi, Mosek, and Knitro, and the non-commercial solvers Clp and Ipopt from the COIN-OR repository http://www.coin-or.org. Cplex, Gurobi, Mosek, and Clp are solvers targeted toward linear and quadratic optimization problems. These solvers use implementations of the primal and dual simplex method and of Mehrotra's predictor-corrector barrier method. Ipopt[16] and Knitro[5] are barrier solvers for general nonlinear optimization problems. The reason for considering these nonlinear solvers in this study is that their linear algebra kernels are highly efficient, making them ideal for large-scale applications. These advantages will be explained in the following section.

A. Simplex Methods

The simplex method starts by partitioning the variable space into *basic* and *non-basic* variables $x^T = [x_B^T \, x_N^T]$ with $x_B \in \Re^m, x_N \in \Re^{(n-m)}$. With this, the Jacobian matrix can be partitioned as $A = [A_B \, A_N]$ where $A_B \in \Re^{m \times m}$ is a square matrix and $A_N \in \Re^{m \times (n-m)}$. Similarly, the cost vector can be partitioned as $c^T = [c_B^T \, c_N^T]$. The optimality conditions of (3) are:

$$c - A^T\lambda - \nu = 0 \tag{4a}$$

$$Ax - b = 0 \tag{4b}$$

$$x^T\nu = 0,\, x \geq 0,\, c - A^T\lambda \geq 0, \tag{4c}$$

where $\lambda \in \Re^m$ and $\nu \in \Re^n$ are the constraint and bound multipliers, respectively. In pseudo-code, the basic steps of the simplex method are [2]:

- **At** iteration $k = 0$ *Start* with a non-singular basis A_B^0, $x_N^0 = 0$, and $x_B^0 \geq 0$. If basis not available, set $A_B^0 \leftarrow \mathbb{I}_{m\times m}$
- **For** iteration $k \geq 0$ do,
  1) *Factorize* basis A_B^k using LU decomposition to obtain L_B^k, U_B^k or *update* existing factors L_B^{k-1}, U_B^{k-1}.
  2) *Compute basic variables* by solving $A_B^k x_B^k = b - A_N^k x_N^k$ and *multipliers* by solving ${A_B^k}^T \lambda^k = c_B$ with available factors.
  3) *Check* $\nu_N^k = c_N - {A_N^k}^T \lambda^k \geq 0$. If it holds, *solution is optimal*, otherwise, choose any variable $x_N^{k,e}$ in x_N^k for which $\nu_N^{k,e} < 0$ as an *entering variable* for the basis.
  4) *Compute basis step* by solving $A_B^k \Delta x_B^k = A_N^k(:,e)$ and ratios $\Theta^k = x_B^k / \Delta x_B^k$. Here, $A_N^k(:,e)$ is the e-th column of A_N^k.
  5) *Apply ratio test* to find *leaving variable* $x_B^{k,l}$ with $\Theta_l^k \geq 0$ such that $x_B^{k+1} = x_B^k + \Theta_l^k \Delta x_B^k \geq 0$.
  6) *Update Basis* by setting $A_B^k(:,l) \leftarrow A_N^k(:,e)$, set $x_B^{k,l} \leftarrow 0$ and go to next step $k \leftarrow k+1$.

In the above algorithm, the factorization step (1) is the most computationally intensive step [11], [15]. Efficient LU factorization routines (such as MA48 from Harwell) are used to factorize the basis matrix which is sparse, unsymmetric, and indefinite. The factorization time of this matrix will increase with the horizon length and network complexity. Note that, if a basis is not originally supplied, the algorithm can take a very large number of iterations (on the order of m) to obtain a feasible basis. Consequently, a large number of factorizations and long computational times can be expected. Once a good basis matrix has been identified, strategies such as the Forrest-Tomlin and Golub-Bartels can be used to update the basis LU factors inexpensively [8], [15]. In real-time applications, it is thus critical to provide the algorithm with a good starting basis.

B. Barrier Methods

Another approach to solve the problem consists on relaxing the complementarity conditions (4c) as $x^T(c - A^T\lambda) = \mu\,\mathbf{e}$, $\mu^k \geq 0$ and apply Newton's method directly to the nonlinear optimality conditions. Here, $\mathbf{e} \in \Re^n$ is a vector of ones. The search step for the variables and multipliers is computed simultaneously by solving the optimality conditions for decreasing values of $\mu^k \to 0$. For fixed μ^k, the search step at iteration j is computed from the solution of the linear system:

$$\begin{bmatrix} \Sigma^j & A^T \\ A & \end{bmatrix} \begin{bmatrix} \Delta x^j \\ \Delta\lambda^j \end{bmatrix} = - \begin{bmatrix} c - A^T\lambda^j - X^j\mu^k\mathbf{e} \\ Ax^j - b \end{bmatrix}, \tag{5}$$

where $X^j = \text{diag}(x^j)$,$V^j = \text{diag}(\nu^j)$, and $\Sigma^j = {X^j}^{-1}V^j$. The bound multipliers are recovered from $\Delta\nu^j = -{X^j}^{-1}(\mu\mathbf{e} + V^j\Delta x^j) - \nu^j$. In the most basic setting, the Newton iterations $j > 0$ try to converge to the solution $x^*(\mu^k)$ and then μ_k is decreased. Some more advanced μ-updates can be used.

The factorization of the matrix on the left-hand side (Karush-Kuhn-Tucker matrix) is the most computationally intensive step in the algorithm. Note that this matrix is symmetric and indefinite and is much larger than the basis matrix factorized in the simplex method (i.e.; $(n+m)\times(n+m)$ against $m \times m$). To solve the linear system, two approaches are normally used. The first one consists on eliminating the step for the multipliers to form the normal equations:

$$\left(A{\Sigma^j}^{-1}A^T\right)\Delta\lambda^j = -\left(r_\lambda^j - A^j{\Sigma^j}^{-1}r_x^j\right), \tag{6}$$

where $r_x^j = -(c - A^T\lambda^j - X^j\mu^k\mathbf{e})$ and $r_\lambda^j = -(Ax^j - b)$. The matrix on the left-hand side is known as the normal matrix. The step for the primal variables is recovered from $\Delta x^j = {\Sigma^j}^{-1}\left(r_x^j - A^T\Delta\lambda^j\right)$. If the Jacobian matrix A is full-rank, then the normal matrix is positive definite. This enables the application of a Cholesky factorization to obtain factors of the form L^j and ${L^j}^T$. Even though the normal matrix is significantly smaller ($m \times m$) than the original KKT matrix, forming the normal system might destroy the sparsity of the original KKT matrix, making the Cholesky factorization inefficient. This is the strategy used in most barrier solvers specialized for linear optimization problems such as Cplex [4], [3], Clp, and Mosek.

A more efficient approach that can be used to factorize the KKT matrix consists on applying directly a saddle-point solver such as MA57 [7] and Pardiso [13]. Saddle-point solvers have advanced significantly in the last years and are capable of solving very large systems efficiently. The key of this approach

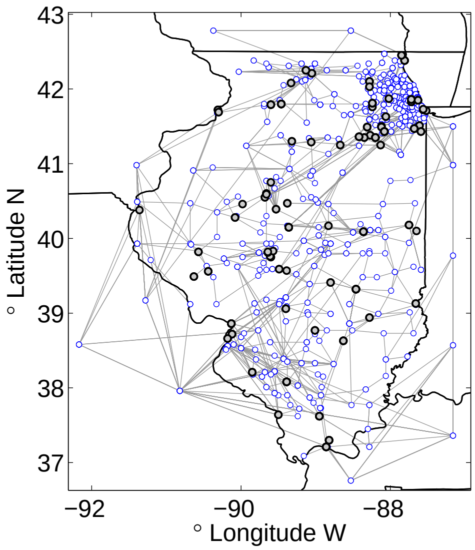

Fig. 2. Illinois interconnect. Gray dots are generation buses.

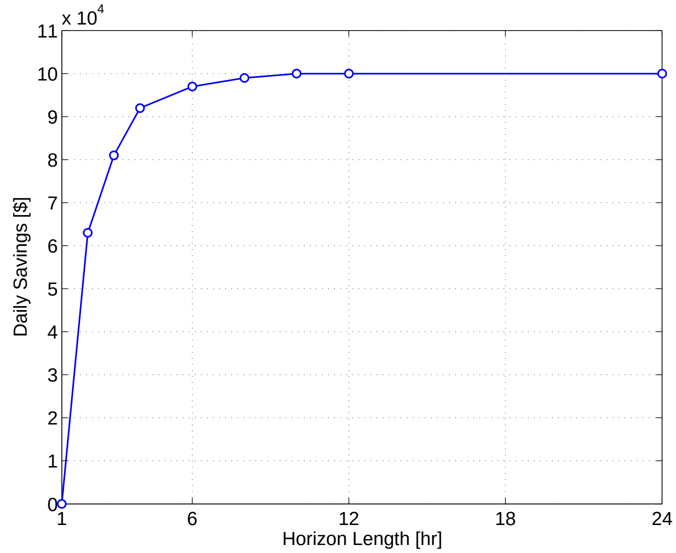

Fig. 3. Effect of horizon length on cost savings.

is the ability to preserve and exploit the high degree of sparsity of the KKT matrix.

A fundamental problem of interior-point solvers is that they cannot exploit warm-start information efficiently [9]. This is because barrier methods proceed progressively from the interior towards the boundary of the feasible region. On the other hand, this also makes the number of iterations insensitive to the problem size and number of variable bounds. Consequently, these solvers are much more efficient than simplex method when no warm-start information is provided. In the following section, we will evaluate the performance of barrier and simplex methods on large-scale ED problems. In addition, we will propose strategies to exploit the advantages of these competing approaches to accelerate the solutions.

V. Illinois Interconnect Set-Up

We have built an ED model using real data for the Illinois interconnect. The system comprises 1900 buses, 2538 transmission lines, 870 load nodes, and 261 generators. Out data consists of detailed data for the network topology, ramp and generation limits, fuel costs, and transmission lines specifications. The Illinois interconnect is sketched in Fig. 2. We have added artificial wind power data in out-of-state buses to simulate a nominal wind power adoption of 10%.

A. Economic Issues

We first analyze the effect of increasing the forecast horizon in the ED formulation. We run the system using a moving horizon approach for a single bidding cycle. In Fig. 3, we plot cost savings as a function of horizon length using a one-hour horizon as the reference. We use a time resolution of one hour. As can be seen, significant savings can be realized by extending the horizon over 8 hours. In addition, the optimal cost can be reached with an horizon of around 10 hours. The savings over the bidding cycle are around $\$100K$. We have observed that the magnitude of the savings depend on the ramp constraints, the initial conditions for the generators, and the variability of the wind power and loads. Consequently, while the overall trends are realistic, the actual savings should be interpreted with care.

B. Computations Issues

The previous study suggests that increasing the horizon of the ED formulation can bring increased performance. In Table I and Fig. 4 we present the problem dimensions as the horizon increases. In addition, we present solution times with no warm-start for the Ipopt (version 3.8) and Cplex (version 12.2) solvers. The Harwell subroutine MA57 was used for factorization of the KKT matrix in IPOPT. The best reordering strategy was nested dissection, implemented in the Metis package [10]. The dual simplex method was used in Cplex (Cplex-Dual). All calculations were obtained using a quad-core Intel processor running Linux at 2.4 GHz.

The size of the problem increases linearly with the horizon length. A problem with 24 time steps already contains more than 100,000 variables. Most of the complexity comes from the network constraints. However, it is interesting to observe that, despite the network complexity, the problems are very sparse and the sparsity increases with the problem size. Ipopt has been found to be significantly more efficient than Cplex-Dual in the case where no warm-start is supplied. In particular, for a problem with 24 time steps, the solution time of Ipopt is less than 3 minutes while that of Cplex is more than 11 minutes. The largest problem solved with Ipopt contains 28 time steps, 205,000 variables and can be solved in less than 10 minutes. We point out that the barrier method implemented in Cplex was not competitive in solution time and robustness. For instance, the solution of an ED problem with thee time steps using Cplex-Barrier takes around two minutes. This can be mainly attributed to the linear algebra kernel and

TABLE I

COMPUTATIONAL PERFORMANCE OF OPTIMIZATION SOLVERS (NO WARM-START).

N	n	m	Nonzeros Jacobian [%]	Ipopt CPUs - Iter	Cplex-Dual CPUs - Iter
1	4272	4009	0.068	**0.7** - 22	**0.3** - 1154
3	12816	12027	0.024	**4.2** - 36	**4.5** - 5100
5	21360	20045	0.014	**9.5** - 41	**17.6** - 10312
10	42720	40090	0.007	**36.0** - 46	**120.1** - 25427
12	51264	48108	0.006	**42.0** - 43	**181.7** - 31191
16	68352	64144	0.005	**94.1** - 51	**344.7** - 46099
20	85440	80180	0.004	**110.4** - 47	**600.5** - 63737
24	102528	96216	0.003	**163.8** - 50	**679.3** - 68540

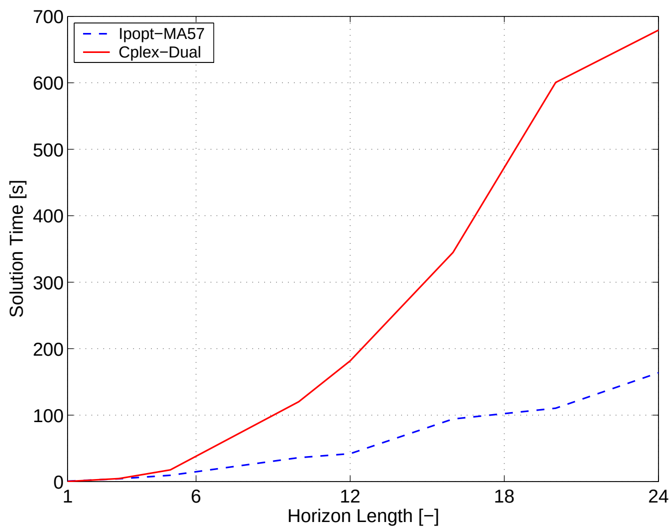

Fig. 4. Effect of horizon length on computational performance of optimization solvers (no warm-start).

ill-conditioning of the KKT matrix. Unfortunately, since the Cplex output display is limited, it is difficult to pinpoint performance bottlenecks.

In Table II and Fig. 5 we present the performance of the solvers when warm-start is provided. In this set-up, the problems are solved with nominal wind power values to obtain the initial solution. The wind power outputs are then perturbed by 10% of their nominal value. When warm-started, Cplex significantly outperforms Ipopt. In particular, note that for a 24 time step problem, Cplex takes 28 seconds while Ipopt takes more than 2 minutes. It is particularly interesting to observe that Cplex only requires 9 refactorizations of the basis matrix. Another interesting observation is that the factorization times of the KKT matrix are lower than those of the basis matrix, despite the fact that the KKT matrix is twice as large. This clearly illustrates the efficiency of MA57 and the Metis reordering.

In Fig. 6 we analyze the robustness of the warm-starts provided to the solvers. We solve a problem with six time steps and perturb the wind power profiles by $10, 20, 30, 40, 50\%$

TABLE II

COMPUTATIONAL PERFORMANCE OF OPTIMIZATION SOLVERS (WARM-START). TOTAL WIND POWER PERTURBATION OF 10%.

N	Ipopt CPUs - No. Refactorizations	Cplex CPUs - No. Refactorizations
1	**0.9** - 14	**0.2** - 1
6	**17.67** - 38	**4.85** - 8
12	**47.82** - 43	**14.95** - 9
18	**118.33** - 46	**16.06** - 8
24	**135.45** - 47	**28.28** - 9

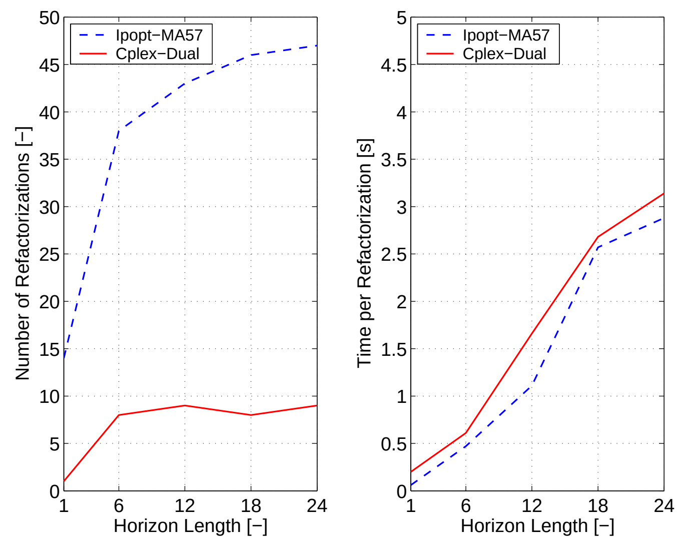

Fig. 5. Effect of horizon length on computational performance of optimization solvers (with warm-start).

of the nominal values. We have found that the basis matrix can keep the solution times of Cplex relatively stable despite the strong perturbations. Similar behavior has been observed for perturbations in the loads. For a problem with 12 time steps and a perturbation of 10% *in all the bus loads*, the solution time goes down from 442 seconds with no warm-start to 37 seconds with warm-start. In the warm-started case, only 883 dual simplex iterations and four refactorizations are needed. For a problem with 16 time steps the solution time does down from 350 seconds to 50 seconds with only 4 refactorizations needed. Note that a perturbation of 10% in the loads is of the same order as the forecast errors observed in real operations. This result is important because it suggests that *we can construct basis matrices in advance (e.g.; one day ahead using the forecasted load) and reuse them in real-time to accelerate the solutions*. Moreover, the warm-start basis matrix can be constructed with barrier solvers such as Ipopt or Knitro and this can be fed to the simplex solver to perform fast, real-time LU updates.

VI. CONCLUSIONS AND FUTURE WORK

We have presented a preliminary evaluation of the effects of increasing the resolution of dispatch formulations. In par-

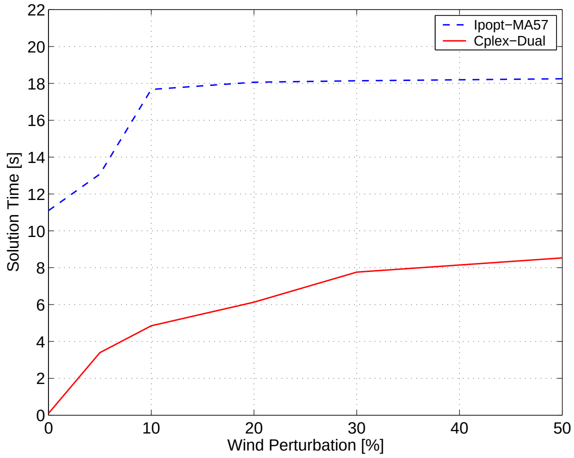

Fig. 6. Effect of data perturbations on solution times. Case with six time steps in horizon.

ticular, it is clear that longer horizons are needed in more dynamic operations as those expected in the next-generation grid. We have tested the performance of two state-of-the-art solvers implementing barrier and simplex methods in a large-scale interconnect system. We have found that the basis matrix in the simplex method is robust to data perturbations. In addition, we have found that barrier solvers that directly factorize the Karush-Kuhh-Tucker matrix scale well in large-scale problems. These complementing advantages can be used to derive warm-starting strategies to avoid computational bottlenecks. For instance, we suggest that warm-start basis matrices should be constructed one bidding cycle in advance using forecast information and re-used in real-time. The presented computational analysis also sets a reference for the expected performance of state-of-the-art solvers. This is important in moving forward to more complex dispatch formulations including real-time unit commitment, storage, and transmission switching decisions.

Acknowledgment

This work was supported by the U.S. Department of Energy through contract DE-AC02-06CH11357.

References

[1] R. Bellman. *Dynamic Programming*. Dover, New York, 1972.

[2] R. E. Bixby. Implementing the simplex method: The initial basis. Technical Report ADA453079, Rice University, http://handle.dtic.mil/100.2/ADA453079, 1991.

[3] R. E. Bixby. Solving real-world linear programs: A decade and more of progress. *Operations Research*, 50:3–15, 2002.

[4] R. E. Bixby, J. W. Gregory, I. J. Lustig, R. E. Marsten, and D. F. Shanno. Very large-scale linear programming: A case study in combining interior point and simplex methods. *Operations Research*, 40:885–897, 1992.

[5] R. H. Byrd, J. Gilbert, and J. Nocedal. A trust-region method based on interior-point techniques for nonlinear programming. *Mathematical Programming*, 89:149–185, 2000.

[6] M. Diehl, H. J. Ferreau, and N. Haverbeke. Efficient numerical methods for nonlinear MPC and moving horizon estimation. *In Nonlinear Model Predictive Control*, pages 391–417, 2009.

[7] I. S. Duff. Ma57 - a code for the solution of sparse symmetric definite and indefinite systems. *ACM Transactions on Mathematical Software*, 30:118–144, 2004.

[8] J. J. H. Forrest and J. A. Tomlin. Updated triangular factors of the basis to maintain sparsity in the product form simplex method. *Mathematical Programming*, 2:263–278, 1972.

[9] J. Gondzio and A. Grothey. Reoptimization with the primal-dual interior point method. *SIAM J. Opt.*, 13:842–864, 2003.

[10] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20:359–392, 1999.

[11] R. Luce, J. Duintjer, M. Groetschel, T. Koch, J. Liesen, R. Nabben, and O. Schenk. On the linear algebra kernel of simplex-based LP solvers. Technical report, Technical University of Berlin, Preprint, http://www.math.tu-berlin.de/∼nabben/Publikation/lasimp08.pdf, 2008.

[12] C. Monteiro, R. Bessa, V. Miranda, A. Boterrud, J. Wang, and G. Conzelmann. Wind power forecasting: state-of-the-art 2009. Technical report, INESC Porto and Argonne National Laboratory, 2009.

[13] O. Schenk, A. Wächter, and M. Hagemann. Matching-based preprocessing algorithms to the solution of saddle-point problems in large-scale nonconvex interior-point optimization. *J. Comp. Opt. and App.*, 36:321–341, 2007.

[14] M. Shahidehpour, H. Yamin, and Z. Li. *Market Operations in Electric Power Systems: Forecasting, Scheduling, and Risk Management*. Wiley, New York, NY, 2002.

[15] L. M. Suhl and U. H. Suhl. A fast LU update for linear programming. *Annals of Operations Research*, 43:33–47, 1993.

[16] A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57, 2006.

[17] C. Wang and S. M. Shahidehpour. Effects of ramp-rate limits on unit commitment and economic dispatch. *IEEE Transactions on Power Systems*, 8:1341–1350, 1993.

[18] A. J. Wood and B. F. Wollenberg. *Power Generation, Operation and Control*. Wiley, New York, NY, 1994.

[19] V. M. Zavala, C. D. Laird, and L. T. Biegler. Fast implementations and rigorous models: Can both be accomodated in NMPC? *Int. J. Robust Nonlinear Control*, 18:800–815, 2008.