EXPERIMENTAL TESTS OF THE EFFICIENCY OF COMPETITIVE MARKETS FOR ELECTRIC POWER USING POWERWEB

William D. Schulze, Simon Ede, Ray Zimmerman,
John Bernard, Timothy Mount, Robert Thomas, Richard Schuler

Cornell University

Introduction

The development of Web-based economics experiments promises a number of innovations and opportunities. In this paper, we describe how we have used this new technology to test alternative market designs for a restructuring the electric power industry under a grant from the National Science Foundation. In what follows, we describe our experiments and then draw conclusions based on our experiences.

The US electric power industry, in particular California and the Northeastern United States, has taken major steps to restructure its institutional arrangements to support competition among energy suppliers. The US is not the first in the world to embark on this path, and to refer to the undertaking as deregulation would be a mistake. In early 1990s the United Kingdom restructured it's industry to form separate generation, transmission and distribution companies (Newbery and Green 1996). Today, this arrangement represents one of the most complex regulatory environments in the world due to efforts to ensure that the independent companies provide reliable electric power at "fair" prices. Indeed the England and Wales market will shortly undergo a transformation by abandoning the mandatory pool for a system of bilateral transactions and a voluntary power exchange, a system similar the Californian market. Despite the experience in the UK, the historical experience with deregulation of other industries has been an unqualified success from the point of view of economic efficiency. For example, price decreases in the airline, natural gas, and long distance telephone industries have been well documented (Winston 1993; Crandall and Ellig 1997). However, the electric utility industry presents unprecedented complications for restructuring.

Since electric power networks offer multiple simultaneous commodities and there are a variety of externalities in transmission, a pure market solution is unlikely to be efficient. For this reason, Vernon Smith and his colleagues (McCabe, Rassenti et al. 1991) proposed the notion of a "smart market." Smart markets use a computer optimization algorithm that interacts with buyers and sellers (using appropriate trading or activity rules) to provide feedback on physical constraints, such as line congestion, which would not be attainable by the market alone. In the United States, auctions for power have begun to replace centralized dispatch algorithms as a means to determine unit commitment (when to turn on or turn off generators with non-zero start up costs) and derive the local price of electricity including transmission charges that reflect line constraints.

This paper reports on two sets of experiments that address the market's ability to produce a cost efficient outcome in power generation. The first experiment examines the ability of generators to exact market power in the presence of line constraints. Under regulation, returns on generating assets could be considered guaranteed. Today, however, with those guarantees removed, power producers will be driven by the profit motive. There exists ample evidence from other industries that owners will seek to sustain higher than competitive prices when possible. The second experiment examines the efficiency of self-commitment in comparison to centralized unit commitment. The unit commitment problem is a complex mixed integer programming problem. Is it realistic to assume that it can be solved in a decentralized manner?

In both experiments, we implement a smart market to account for the operational constraints imposed by the physical transmission network. In this context, the sellers and the buyer's demands are connected by a transmission network which must be operated at all times in a manner consistent with the laws of physics governing the flow of electricity. The operation of the network is also constrained by the physical limitations of the equipment used to generate and transmit the power. This results in two phenomena which may affect the auction: (1) transmission losses and (2) congestion.

A small percentage of the energy produced by the generators is dissipated by the transmission lines. The amount of power lost depends on the flow in the line and the length of the line, among other things. Transmission loss implies that the total amount of power the buyer must

purchase is slightly greater than the total demand and the exact amount is dependent on where the power is produced.

There are limits on the amount of electric power that can be transmitted from any given location to any other location. Some of the limits are simple line capacity limits and others are more subtle system constraints arising from voltage or stability limits. Congestion occurs when one or more of these network limits is reached. Congestion implies that some inexpensive generation may be unusable due to its location, making it necessary to utilize a more expensive unit in different location.

In our experiment platform, PowerWeb, the effects of losses and transmission system constraints are handled by adjusting all offers and prices by a location specific transmission charge which represents the shadow price of transporting the electricity. There is a two part transmission charge associated with each line which is divided up between the various generators based on their individual contributions to the flow in the line. The per-line transmission charges can be explained as follows. The value of the power dissipated by a transmission line is the loss component of the transmission charge for that line. The congestion component of the transmission charge is precisely the charge necessary to discourage overuse of the line. If there is no congestion, this component is zero. It is important to note that the transmission charges are dependent on the flow in each transmission line as well as each generator's contribution to that flow and therefore cannot be computed before performing the auction. In this context, each generator receives a price which is specific to its location.

Generator units are chosen so as to satisfy fixed location specific demand in the least expensive manner while satisfying the operational constraints of the transmission system. This is done by an optimal power flow program which computes the appropriate transmission charges for each generating station. The units selected by the optimization program are roughly those given by the following procedure. The appropriate transmission charge is added to the price of each offer, and the offers are ordered from lowest to highest adjusted offer price. Units are included for sale, starting from the low priced units and moving toward the higher priced units, until the supply

reaches the total buyer's demand plus transmission losses. The remaining, higher priced, units are excluded from sale.

The reigning price is set to the adjusted offer price of the last (most expensive) unit chosen. The price paid for each unit produced by a given generator is the reigning price minus the corresponding transmission charge. In prior research, we have shown that this last accepted offer mechanism (LAO) performs as well, or better, than the Vickrey Multiple Unit Auction or alternative uniform price auctions that set the price equal to the first rejected offer when sellers have multiple units (Bernard et al., 1998).

Market Power

Market power increases as sellers own a larger fraction of the capacity available for serving demand (load). In an electric power grid, the supply and demand are dispersed throughout the system. Each generator and each load lie at a specific network location. Due to the constraints imposed by the transmission grid, it may not always be possible to transfer power from an arbitrary generating station to any given load. This implies that the capacity available to serve a specific load may be a subset of the total generation capacity in the system and that market power may be present if a small number of sellers own a large fraction of this subset of generation. The market is partitioned into smaller market islands by the limitations on transmission imposed by the network. If areas A and B of a transmission grid are isolated by transmission constraint, then generator A in area A cannot compete with generator B in area B to serve the load in area B. Likewise, generator B cannot compete with generator A to serve load in area A. The owner of a generation facility may have market power if they own a significant percentage of capacity in an isolated area even if they own only a small fraction of the total generation in the system.

These transmission limits may be simple and relatively constant thermal limits on the lines or they may arise indirectly from voltage or stability limits. In the latter case, the constraints may be very sensitive to VAr (reactive power) injections necessary to maintain voltage and other operating conditions. Therefore, market power could also arise from ones ability to manipulate the operating condition of the network in order to partition the markets to one's own advantage.

In summary, there are at least two ways in which the transmission network can create market power opportunities in load pockets. First, transmission constraints, arising from line limits, voltage limits, or stability limits, may partition the market into islands which may create the type of market power described above. Second, one may exploit one's position in the network to strategically partition the market to one's own advantage. Simple auctions that do not take into account transmission system constraints would often lead to infeasible operating conditions if employed in a constrained network (see for example, Hogan, 1992). The answer to this problem, of course, is use of a smart market which employs an auction where offers are adjusted for nodal pricing through transmission charges determined by an optimal power flow (McCabe, Rassenti et al. 1991).

The Experiment

We conducted three experiments with student subjects and one with electricity traders using our web-based experimental platform, PowerWeb, which implements the smart market described above using an OPF that models a full non-linear lossy AC transmission network. These experiments utilized the six generator, 30-node network model, shown as a simplified block diagram in Figure 1. The PowerWeb platform is described in detail in the Appendix.

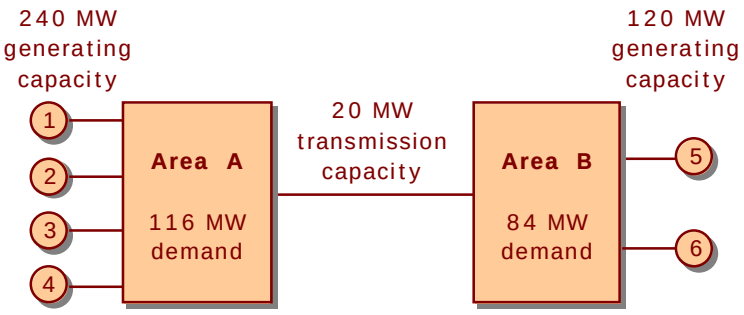

Figure 1: Transmission Network Block Diagram

Each of the six subjects in each experiment was one of six sellers in a market with a single buyer with a fixed demand. All generators had a capacity of 60 MW (megawatts) which was divided into 3 blocks, 12, 24, and 24 MW at marginal costs of $20, $40, and $50/MW-hr, respectively. All generators had identical capacity and cost structures. Each generator could generate between 12 and 60 MW of power, or could be shut down completely, in which case it incurred no costs. Given the inelastic demand, a limit price of $80/MW-hr was imposed.

The network was structured so as to create a load pocket in Area B, where generators 5 and 6 are located. The limitation on transmission capacity between areas A and B, can effectively separate the market into groups of four and two competitors, respectively. The demand levels and network constraints are such that neither generator 5 nor generator 6 can be shut down.

To see examples of the offer submission and auction result pages used by PowerWeb, please see Figures A1 and A2 in the Appendix.

Each of the three student sessions was run for 75 rounds, and each produced different results. Figure 2 shows the price results for a session that can be used to characterize all three sessions. In one session, the results for the prices received by the six generators remained similar to the price pattern shown in the figure prior to period 50. In other words, prices remained near the competitive level (shown by the heavy horizontal line in the figure) throughout the session. In a second session, prices were similar to those shown after trading period 50 in the figure, for the entire session. In other words, generators 5 and 6 were able to exploit their market power consistently from the initial trading periods through period 75. In the session shown in the figure, generators 5 and 6 were not able to coordinate their price offers to exploit the market power opportunities offered by the network until period 50. It appears that generator 5 (dashed/dotted line, 2nd from top) was not responsive to generator 6 (solid line, top) who attempted to raise prices earlier.

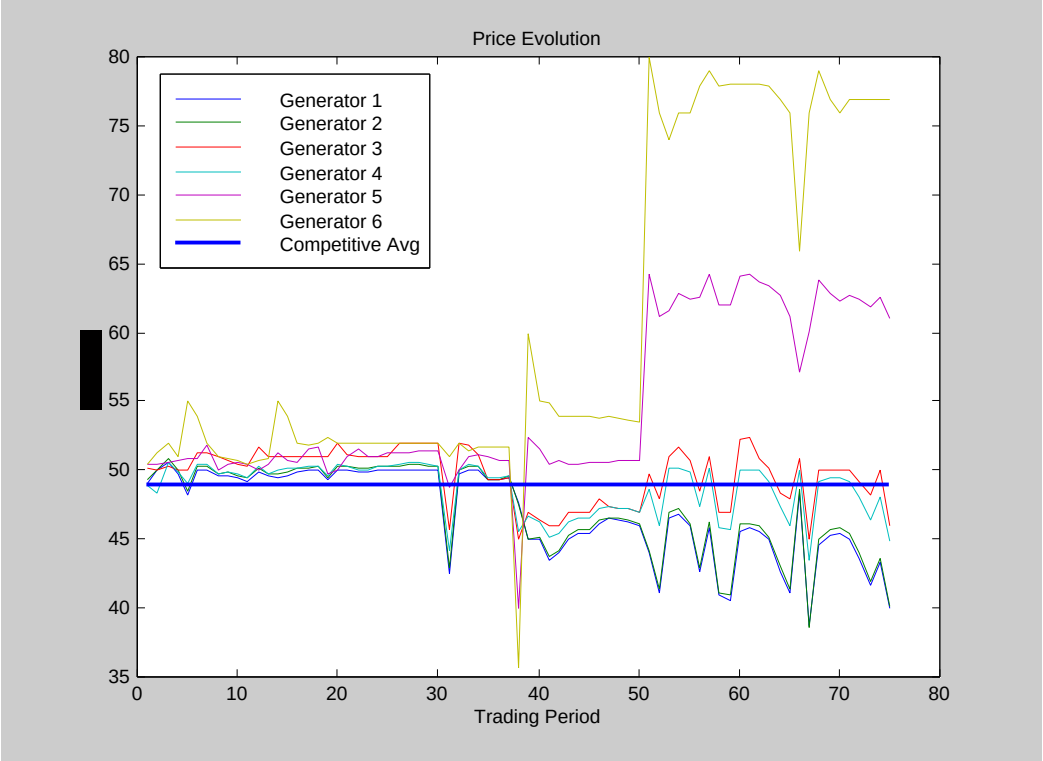

Figure 2: Nodal Prices, Undergraduate Session

We draw two conclusions from these results. First, in two of the three sessions generators 5 and 6 were able to exploit the opportunity to use market power. It should be noted that the 75 trading periods used provides far less experience than actual generators will accumulate over a summer season during peak load periods when networks are likely to be constrained. Thus, it is reasonable to conclude that market power will be exercised. Second, if generators exploit market power, prices will not only be higher in load pockets, but also price volatility will increase. This implies the possibility that network stability and reliability may be jeopardized since relays have been set on the basis of stable generation patterns throughout the networks.

An identical experiment, except for the use of larger incentives and 65 trading periods, was later performed using electricity traders as subjects at a utility headquarters. Figure 3, below, shows the results for the session with the traders. As can be seen, the market power opportunities

were quickly recognized and exploited. Prices well above competitive levels were observed at generators 5 and 6 as early as the second trading period, and remained consistently high after about 25 periods. This result supports the conjecture that the behavior of expert subjects does not differ significantly from that of the more accessible student subjects.

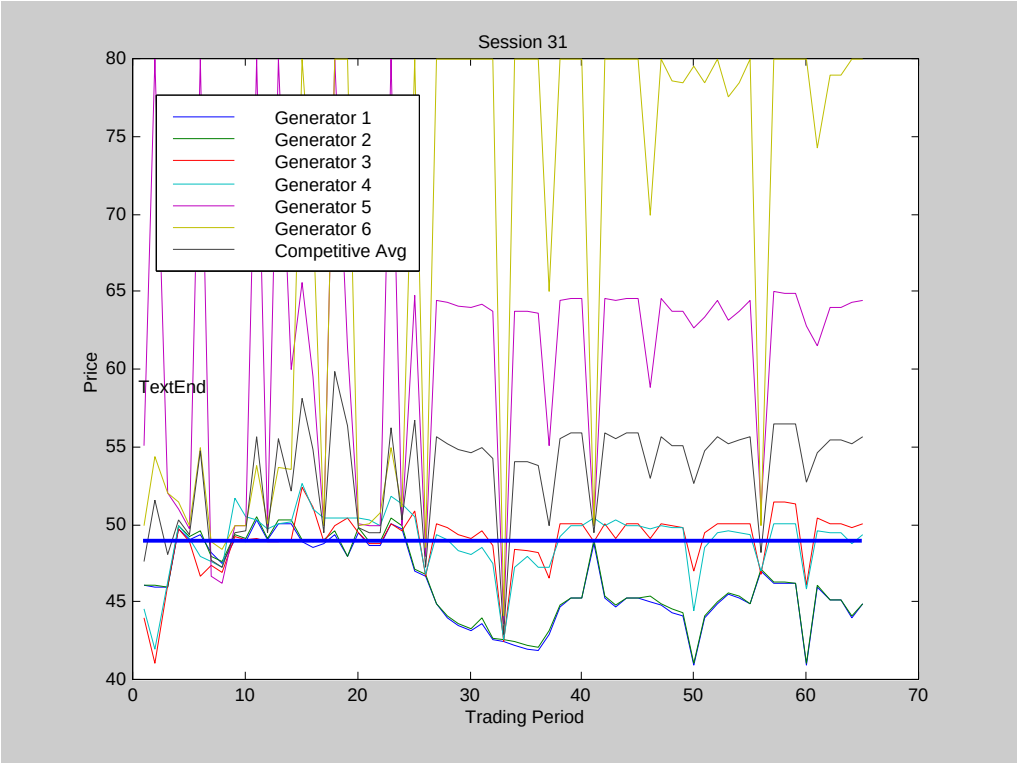

Figure 3: Nodal Prices, Electricity Traders

Unit Commitment

Given the load profile of most electricity markets and the capacities of the generators supplying power to markets, it is likely that only a subset of the total number of generators will be required to satisfy the load during periods of low demand. One of the most important roles for a system operator, whether it is a utility with a portfolio of generating assets, a state controlled government agency, or an independent system operator, is to determine which generators should be

running and for how long. This is frequently called the "unit commitment problem". While the task has to be solved, the method can dramatically vary in different markets. For example, the system operator in the United Kingdom solves both the unit commitment and dispatch problems in a day-ahead market. In contrast, the markets in Australia and California are based on self-commitment by generators and the system operator determines dispatch only. The emerging markets in the eastern United States are closer to the United Kingdom model than to the simpler markets with self-commitment. The basic question posed by these different markets is which approach is the best?

In the case of self-commitment, the inter-temporal dependencies caused by start-up costs provide an incentive to accept losses or reduced profits in some periods in order to increase profits overall. This may cause generators to offer blocks of capacity at below marginal cost in order to avoid a greater start-up cost in a future period. Every generator must determine whether this increases profitability or whether its cost structure is such that it should cycle on and off with the variations in demand. In an intensely competitive market, the optimal strategy should be one that leaves the generator at worst indifferent between the cycling and continual operation. In such a case, the losses incurred would exactly equal the start-up costs avoided. For that reason, this strategy would appear consistent with aims of profit maximization and lead to an efficient solution. It is this hypothesis that has been tested in our research.

The Experiments

We conducted eight experiments to test this hypothesis with our web-based PowerWeb platform, which implements the smart market, described previously, using an OPF that models a full non-linear lossy AC transmission network. These experiments used a six generator, 30 node network model. Each of the six subjects in each of the experiments was one of six sellers in a market with a single buyer with demand that alternated between 100 MW and 200 MW. All generators had a capacity of 60MW that was divided into three blocks, the size of which varied between generators. The costs for each block of capacity varied between generators too. Subjects knew their own capacities and costs but not those of their competitors. Table 1. below shows the capacity and cost structure of each of the competitors:

Generator	Variable Costs					
	Block 1		Block 2		Block 3	
	MW	Cost ($)	MW	Cost ($)	MW	Cost ($)
1	10	23	25	30	25	35
2	10	23	25	30	25	35
3	20	18	30	18	10	40
4	20	20	20	30	20	40
5	20	20	20	30	20	40
6	20	15	30	15	10	40

Table 1: Generator Capacities and Variable Costs

Each generator was required to sell at least its first block of capacity in its entirety. If this did not happen, the generator was shut down for that period. In the event of being shut down, a start-up cost was incurred when the generator again was selected to operate. Table 2. shows the start-up costs for each of the generators:

Generator	Type	Start-Up Cost ($)
1	Peaking	50
2	Peaking	50
3	Base-load	500
4	Mid-Level	150
5	Mid-Level	150
6	Base-load	500

Table 2: Start-Up Costs

The network was structured to eliminate any network constraints. Losses in the system still occurred but were too insignificant to affect the optimal offer strategy of each generator.

Six sessions were run with undergraduate business and economics students at Cornell University. The majority of students were sophomores and juniors taking an intermediate microeconomics class and/or a class in price analysis. One experiment was run with Graduate students in economics and a final experiment was run using larger payoffs with power industry professionals. The six undergraduate sessions and one professional session were run for 60 rounds

alternating between a total demand of 100MW and 200MW. The graduate experiment ran for 40 rounds, being also evenly split between high and low demand periods.

A uniform price auction was held in advance of each of the trading periods. Subjects were informed of the demand for that period and asked to submit offers for each of their blocks of capacity. Units were chosen based of their offers into the auction so as to satisfy demand in the least cost manner while satisfying the constraints of the transmission system (in this experiment to include losses only). Upon submission of offers and completion of the OPF, students were presented the results and profits (based on the reported clearing price and the quantity of electricity sold in the auction) from the previous trading period before submitting offers for the next period. Subjects were paid based on their performance in experimental dollars. An exchange rate was applied to this and students were shown their earnings in actual dollars at each stage. Each subject received an initial "show-up" fee, which was used as an incentive to encourage people to attend the experiment. It was then considered as a starting balance in the experiment. It was possible for subjects to lose money as well as make profits. Losses were capped at $0 (after application of the show-up fee). There was no cap on the profits that could be made.

Our hypothesis has been that some generators would find it profitable to offer sufficient capacity so as to be dispatched at below marginal cost in order to avoid start-up costs in the next period as required for efficiency. Invariably, given the demand and supply structure in these experiments, everyone sold something in high demand periods. The low demand periods are, therefore, of most interest. Table 3 below shows the appropriate offer strategy for each generator. The offer strategy is calculated using the following formula, applicable to two period games[1]:

On capacity < minimum capacity,
offer = average cost of block[2] - start-up cost/ size of first block
On capacity > minimum capacity,
offer = marginal cost

[1] If all generators followed this strategy, optimal dispatch of generators would occur.
[2] Because each MW in a block is the same price, average cost equals marginal cost. It is appropriate, however, to think in average cost terms because in the US power auctions often restrict the number of segments in a price/offer schedule. This forces generators to offer blocks of capacity at the same price.

	Block 1		*Block 2*		*Block 3*	
	Cost ($)	**Offer ($)**	**Cost ($)**	**Offer ($)**	**Cost ($)**	**Offer ($)**
1	23	18	30	30	35	35
2	23	18	30	30	35	35
3	18	-7	18	18	40	40
4	20	12.5	30	30	40	40
5	20	12.5	30	30	40	40
6	15	-10	15	15	40	40

Table 3: Optimal Offers

The Results

The experiments validated the hypothesis that last accepted offer auctions can produce cost efficient dispatch. The graphs in Figure 4 show the offer strategy of each of the six generators averaged over all of the undergraduate sessions in low demand periods. The upper boundary straight line is the offer expected if the generator submitted only marginal cost offers. The lower boundary represents the offer predicted which would leave the generator indifferent between being on in both periods or being on only in high demand periods. In reality the cost structures of the generators in the experiments meant that different generators faced different degrees of competition. The baseload generators faced the least competition while competition was fiercest between generators 1,2 (ordinarily cycling) and generators 4,5 (ordinarily dispatched). We believe that an offer pattern between marginal cost and lowest possible offer can be considered (close to) optimal.

As to be expected, generators 1,2 and 4,5 all converge on the predicted offer. The base-load generators were under less competitive pressure. Nonetheless, their offers also sank below cost in low demand periods, though to a lesser extent. This merely reflects the fact that it is only rational to lower the offer until dispatch is secured. For the base-load generators in this experiment, that was significantly higher than the minimum offer suggested in this paper. These results were also replicated in the graduate and professional experiments.

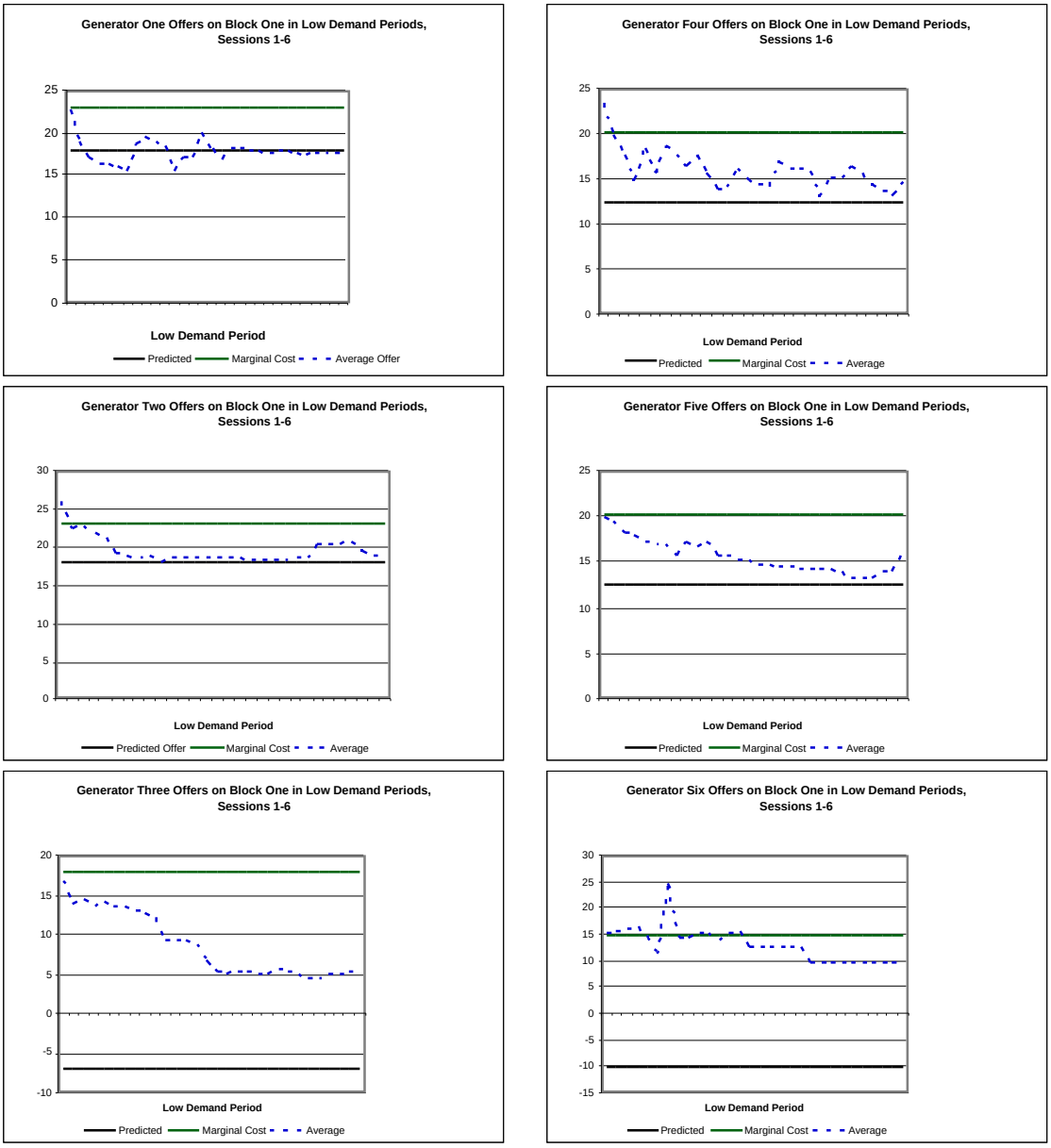

Figure 4: Low Demand Period Offers in Undergraduate Sessions

Figure 5. shows the cost efficiencies of the experiments over cycles of one high and low periods. It's a messy picture but one which conveys the convergence of each of the experiments to close to 100%. Efficiency in these experiments is defined as optimal system cost divided by realized system cost. By means of comparison, had generators submitted marginal cost offers, the efficiency would have been just over 96%. The results show that self-commitment using a uniform price auction converged to a higher efficiency than this.

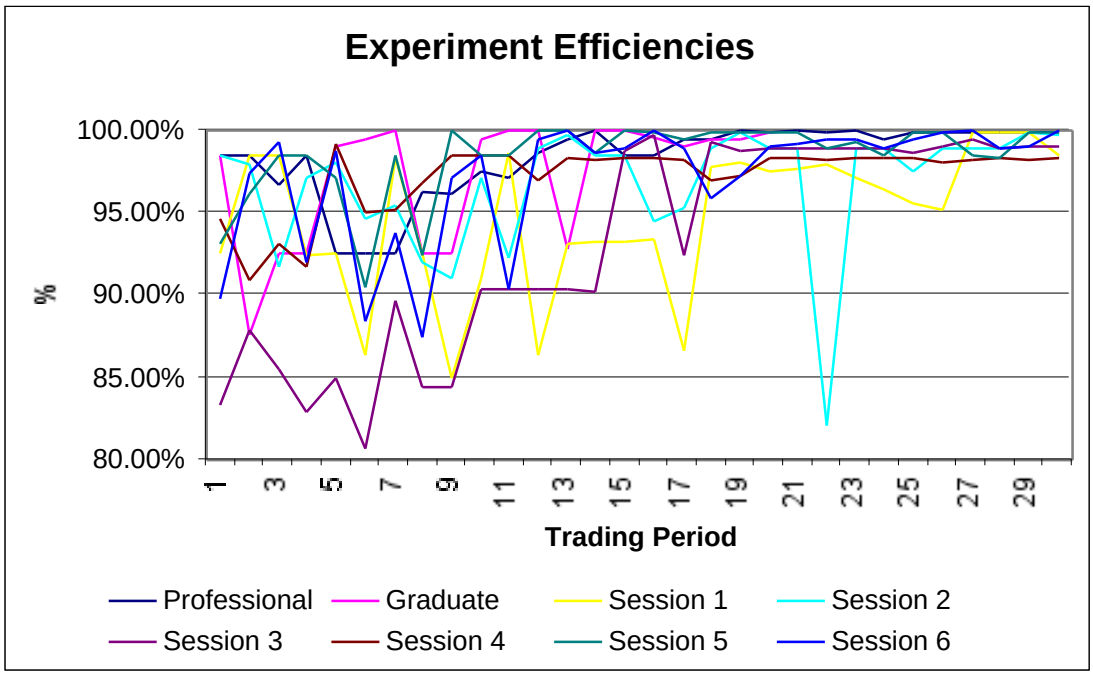

Figure 5: Experiment Efficiencies

Figure 6 shows the average efficiency of the undergraduate experiments compared to the efficiency of the graduate and the professional experiments. The only difference that can be seen between the three groups is the speed with which optimal dispatch was achieved. This again supports the conjecture that behavior of expert subjects does not differ greatly from more accessible student subjects.

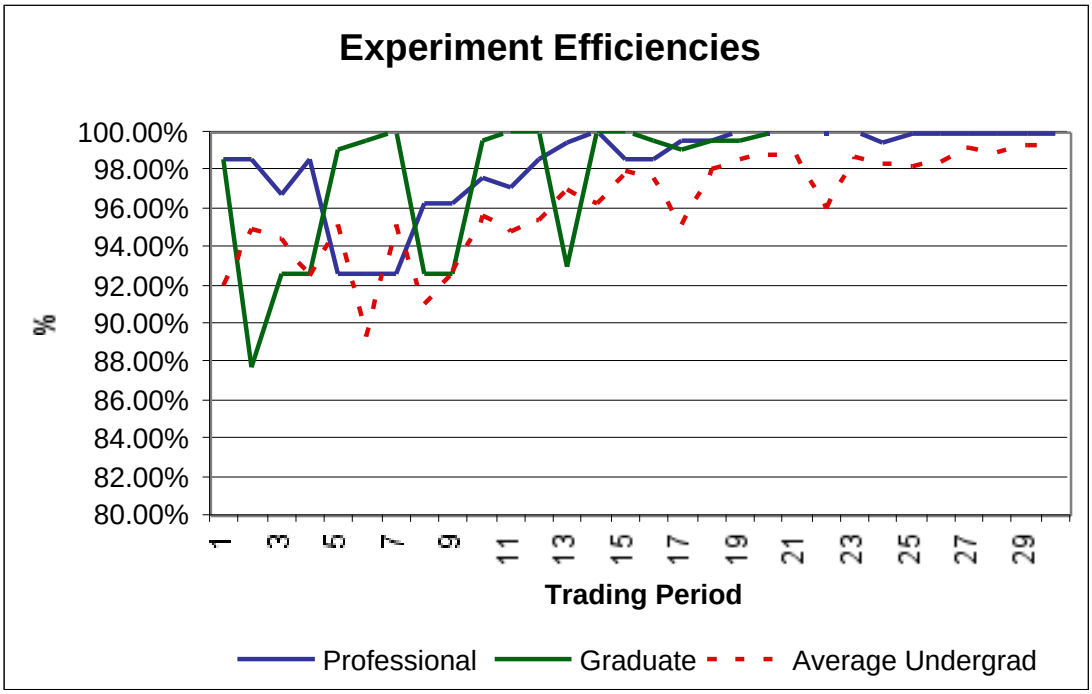

Figure 6: Comparison of Efficiencies

Our experiments show, in a simplified situation, self-commitment can produce a cost efficient dispatch of thermal units. Further complexity needs to be added to the model in the form of ramping constraint and minimum up and down times before it is possible to conclusively say that self-commitment is feasible. Nonetheless, the success of the uniform price auction in this instance is encouraging, given its position as auction-of-choice in electricity markets. Had it failed this simple test, severe doubt would be cast upon its ability to handle more complicated scenarios.

Conclusions

Most of the experimental sessions described above were conducted in the Laboratory for Experimental Economics and Decision Research at Cornell. Thus, one might conclude that moving from an in-house, Novell network environment to Web based experiments changed little. However, a number of advantages are apparent.

First, subjects were very familiar with the browser interface used in the experiments, as opposed to the custom interface utilized in traditional computerized experiments.

Second, as development proceeded, it was easy for investigators to evaluate alternatives by accessing new versions of the experimental interface on their desktop or home computers.

Third, the ability to run experiments at remote locations such as the headquarters of an electric utility literally allowed traders to step off the trading floor and participate in an experiment. In addition, this capability has allowed PowerWeb to be used for training and familiarization by a number of utilities and public utility commissions. Finally, PowerWeb was used as part of a course in experimental economics taught in Slovakia, where it would have been otherwise impossible to demonstrate a computerized, smart market experiment.

Looking to the future, we plan on two further specific uses for the platform. First, we intend to run an experiment with substantial incentives where we will solicit participants from academia, consulting firms, and the electric power industry, to be conducted for a multi-week period over many rounds. Second, we have developed automaton players that behave much like real subjects. We will use these automatons to allow any number of real players to sign on and participate in sessions that can be used either with, or without incentives for teaching purposes. Our long run goal at the Laboratory for Experimental Economics and Decision Research at Cornell is to have a variety of Web- based experiments available for public access that can be used either for demonstration purposes or teaching.

Bibliography

Allen, E. and M. Illic (1999). Price Based Commitment Decisions in the Electricity Market. Glasgow, Springer-Verlag.

Ausubel, L. M. and P. Cramton (1998). Demand Reduction and Inefficiency in Multi-Unit Auctions, University of Maryland: 42.

Bernard, J., T. Mount, et al. (1998). Alternative Auction Institutions for Purchasing Electric Power. Bulk Power Systems Dynamics and Control IV - Restructuring, Greece.

Bertsekas, D., G. S. Lauer, et al. (1983). "Optimal Short Term Scheduling of Large Scale Power Systems." IEEE Transactions on Automatic Control AC-28(1): 1-11.

Bohn, R., M. Caramanis, et al. (1984). "Optimal Pricing in Electrical Networks Over Space." Rand Journal of Economics 15(3): 370-376.

Chao, H.-p., H. G. Huntington, et al. (1998). Designing competitive electricity markets. Boston, Kluwer Academic.

Crandall, R. and J. Ellig (1997). Economic Deregulation and Customer Choice: Lessons for the Electricity Industry. Farifax, VA, Center for Market Processes, George Mason University.

Davis, D. D. and C. A. Holt (1993). Experimental Economics. Princeton, NJ, Princeton University Press.

Dixit, A. K. and B. J. Nalebuff (1993). Thinking Strategically: The Competitive Edge in Business, Politics and Everyday Life. New York, Norton.

Elmaghraby, W. and S. Oren (1999). "The Efficiency of Multi-Unit Electricity Auctions." Energy Journal 20(4): 89-116.

Erwin, S. R., J. S. Griffith, et al. (1991). "Using an Optimization Software to Lower Overall Electric Production Costs for Southern Company." Interfaces 21(1): 27-41.

Ethier, R., R. Zimmerman, et al. (1999). Energy Auctions and Market Power. Hawaii International Conference on System Sciences, Hawaii.

Ethier, R., R. Zimmerman, et al. (1999). "A Uniform Price Auction with Locational Price Adjustments for Competitive Electricity Markets." Electrical Power and Energy Systems 21: 103-110.

Exchange, C. P. (1998). California's New Electricity Market. The Basics: How the PX works, California Power Exchange. 1999.

Fehr, N. v. d. and D. Harbord (1993). "Spot Market Competition in the UK Electricity Industry." The Economic Journal 103: 531-546.

Ilic, M. D., F. D. Galiana, et al. (1998). Power systems restructuring : engineering and economics. Boston, Kluwer Academic Publishers.

Illic, M. and F. Galiana (1998). Power Systems Operation: Old Versus New. Power Systems Restructuring, Engineering and Economics. F. G. Marija Illic, Lester Fink. Boston, Kluwer.

J. Bernard, R. E., T. Mount, et al. (1998). Markets for Electric Power: Experimental Results For Alternative Auction Mechanisms. Hawaii International Conference on System Sciences, Hawaii.

Johnson, R. B., S. S. Oren, et al. (1996). "Equity and Efficiency of Unit Commitment in Competitive Electricity Markets." POWER Working Paper PWP-039.

McCabe, K. A., S. J. Rassenti, et al. (1990). "Auction Institutional Design: Theory and Behavior of Simultaneous Multiple Unit Generalizations of Dutch and English Auctions." American Economic Review 80(5): 1276-1283.

McCabe, K. A., S. J. Rassenti, et al. (1991). "Smart Computer Assisted Markets." Science 254: 254-538.

Milgrom, P. R. and R. J. Weber "A Theory of Auctions and Competitive Bidding." Econometrica 50(5): 1089-1122.

Newbery, D. (1995). "Power Markets and Market Power." Energy Jounral 16(3): 39-66.

Newbery, D. and R. Green (1996). Regulation, Public Ownership and Privatization in the English Electricity Indusry. International Comparisons of Electricity Regulation. R. Gilbert and E. Kahn. New York, Cambridge University Press.

Patrick, R. H. and F. A. Wolak (1997). "The Impact of Market Rules and Market Structure on the Price Determination Process in the England and Wales Electricity Market." POWER Working Paper PWP-047.

Smith, V. (1996). Market Power and Mechanism Design for Deregulated Mechanism Design. Economic Science Meetings, University of Arizona.

United States. Office of Electric Power Regulation. (1981). Power pooling in the United States. Washington, D.C., The Office.

United States. Office of Energy Markets and End Use. (1997). Electricity reform abroad and U.S. investment. Washington, DC, Energy Information Administration Office of Energy Markets and End Use U.S. Dept. of Energy.

VanDoren, P. M. and Cato Institute. (1998). The deregulation of the electricity industry : a primer. Washington, DC, Cato Institute.

Vickrey, W. (1961). "Counterspeculation, Auctions and Competitive Sealed Tenders." Journal of Finance 16: 8-37.

Vickrey, W. (1976). Auctions, Markets and Optimal Allocation. Bidding and Auctioning for Procurement and Allocation. Ahimud. New York, New York University Press.

Wilson, R. (1998). Efficiency Considerations in Designing Electricity Markets, Competition Bureau of Industry Canada: 24.

Winston, C. (1993). "Economic Deregulation = Days of Reckoning for Microeconomists." Journal of Economic Literature 31: 1263-1289.

Wolak, F. A. (1997). Market Design and Price Behavior in Restructured Electricity Markets: An International Comparison. 1998.

Zaccour, G. and Ecole des hautes études commerciales (Montréal Québec) (1998). Deregulation of electric utilities. Boston, Mass., Kluwer Academic.

Zimmerman, R., R. Thomas, et al. (1997). An Internet-Based Platform for Testing Generation Scheduling Auctions. Hawaii International Conference on System Sciences, Hawaii.

APPENDIX

The PowerWeb Platform

Ray Zimmerman

1 The POWERWEB Platform

POWERWEB is designed to be a flexible platform experimentally examining the behavior of various proposed electricity markets using realistic modeling of the physical network and real human decision-makers. As an Internet-based, network-centered computing environment, POWERWEB makes use of a wide variety of technologies in its implementation. The user interface is web-based, all data are handled by a relational database and market computations are performed by a Matlab-based optimal power flow (OPF) program.

1.1 Overview

Because of operational constraints on a power system, it is necessary to have a central agent acting as an independent system operator (ISO). In the current implementation of POWERWEB, the ISO receives offers to sell power from independently owned generation facilities. Based on a forecasted demand profile for the next day and the information gathered from the generator's offers, the ISO computes the optimal generator set points along with a corresponding price schedule which will allow the system to meet changing demand while satisfying all operational constraints.

As a web-based tool, POWERWEB may be used in several capacities. It can be utilized in a tightly controlled setting where a well-defined group of subjects are used for a very specific set of market experiments. It can also be used in a more open environment in which anyone on the web can log in and "play" as a generator competing against other generators, controlled by other humans or computer algorithms (agents), to generate power profitably. In either case, since POWERWEB is web-based it is accessible at all times to anyone with proper authorization, as long as the servers are up and running.

1.2 A Typical Session

To eliminate the need to coordinate accesses (via phone, e-mail, etc.) and to prevent one user's actions from interfering with another's, all accesses occur in the context of a given "session". The session specifies which power system is being simulated, who "owns" which system resources (generators, etc.), and what market mechanism is in use. Multiple sessions can be active at any given time and activity in each is completely independent of the others. Typically, a user in a session will "own" one or more generating plants.

After logging in as a generator in a simple auction session, for instance, the user is taken to the *Offer Submission* page shown in

Figure **A1**, which displays the cost and capacity information for their generator. Here they can enter offers to sell power to the ISO.

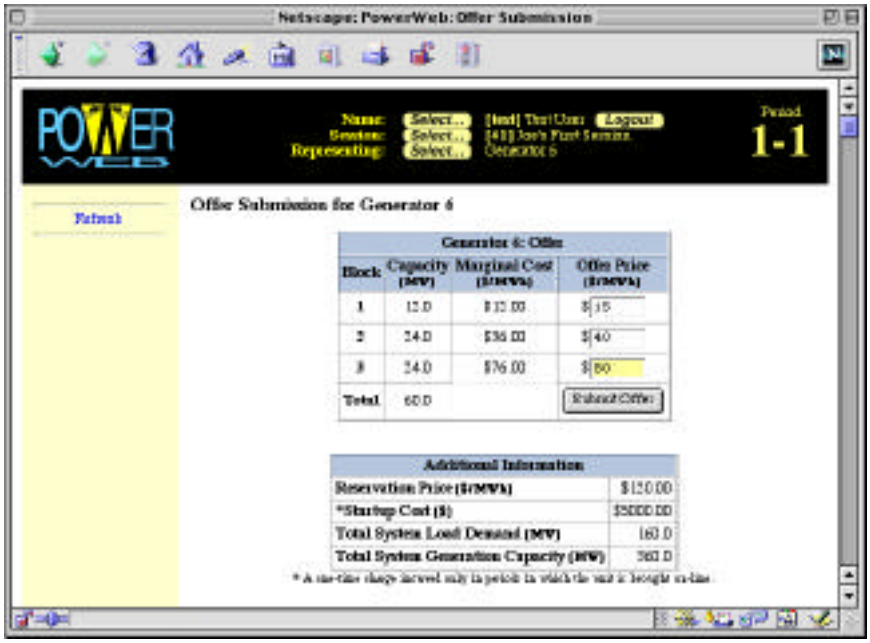

Figure A1 *Offer Submission* Page

When all participants have submitted their offers, POWERWEB's computational engine runs the auction according to the rules specified and reports back the results to the user. The *Auction Results* page is shown in Figure A2.

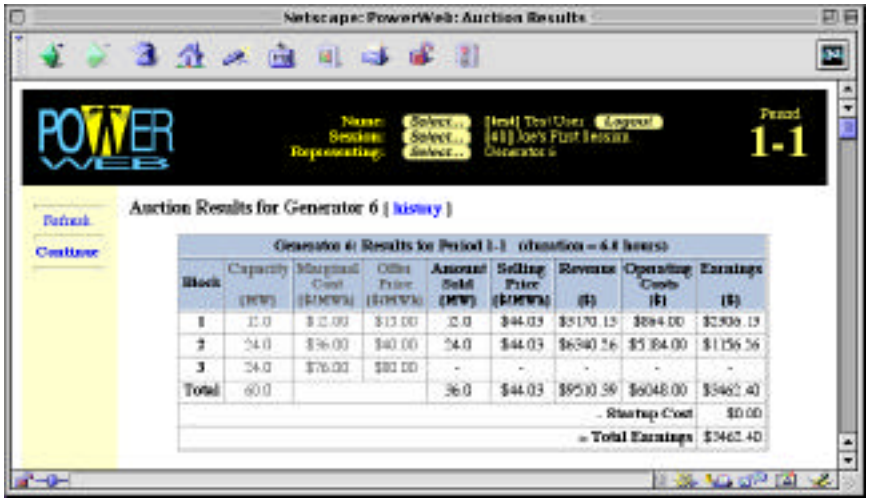

Figure A2 *Auction Results* Page

POWERWEB also has the capability to provide differing levels of information to the players, as specified by the experimenters. In a full information setting, each user would have access to the system information area, which gives tabular summaries of the system operation conditions as well as a "live" one-line diagram of the power system.

Figure A3 shows the one-line diagram of a 6 generator, 30 bus system in POWERWEB's database. This diagram is generated dynamically by a Java applet from information retrieved from a relational database server. The diagram can be panned and zoomed and it is interactive in that clicking on an object such as a line, bus, generator, or load will query the database for information about the object. For example, selecting a bus will display the current information about real and reactive flows into and out of the bus as well as information about the current voltage level of the bus.

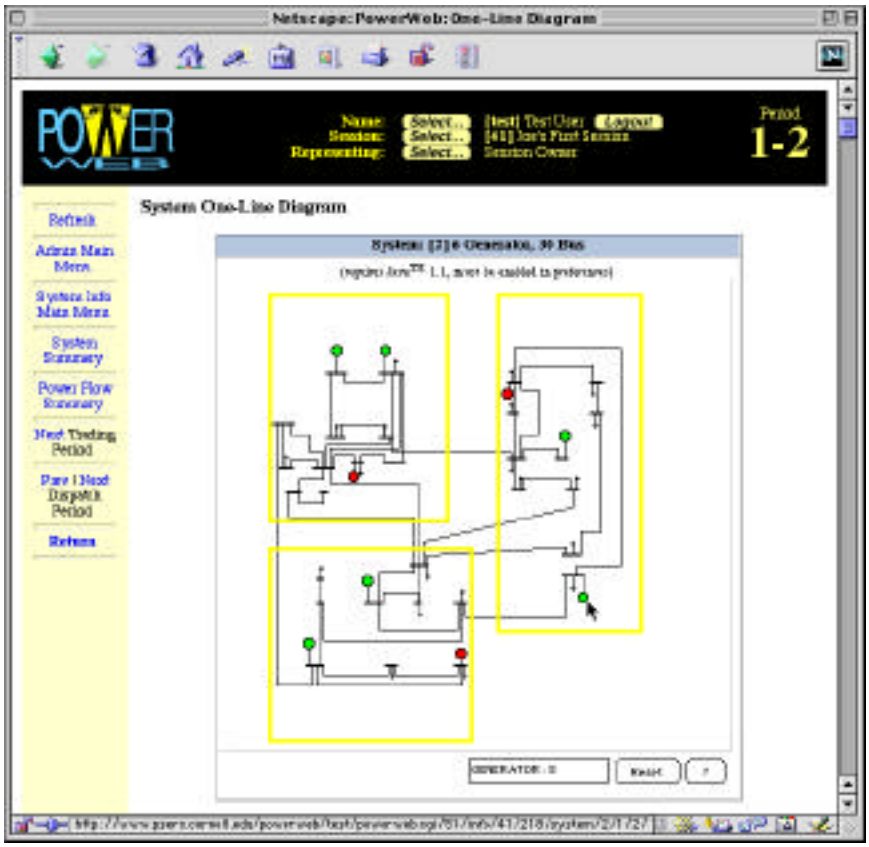

Figure A3: POWERWEB one-line diagram display, showing 30-bus system

The POWERWEB User's Manual, available from the POWERWEB home page at <http://www.pserc.cornell.edu/powerweb/> has more details regarding POWERWEB's functionality.

2 Technologies Employed

In order to understand some of the design choices that were made for POWERWEB, it is important to understand the capabilities and limitations of the currently available Internet technologies. These technologies include a rich collection of cross-platform, open standards that enable developers to quickly create and deploy network-centered applications.

This section explores some of the primary technologies utilized in POWERWEB. It should be noted that, for many of these technologies, the Internet is not the only, or necessarily even primary, context for their use. POWERWEB is a distributed application defined by various *programs* running simultaneously on different computers and the *protocols* by which these programs interact. POWERWEB uses a client-server architecture, where the programs involved take on the role of client or server for a specific of interaction. The technologies discussed below are divided into the *languages* used to implement POWERWEB's various programs and the *protocols* by which they communicate.

2.1 Languages

HTML [1]

HyperText Markup Language (HTML) is a very well known and widely used international standard maintained by the Internet Engineering Task Force (IETF) for defining a document with possible links to other network resources. An HTML document, as interpreted and rendered by a typical web browser, may include structured and formatted text, tables, fill-out forms, images, hypertext links, Java applets and references to other types of data which can be handled via helper applications or browser plug-ins.

HTML is ideal for displaying information to a user on the web. An HTML renderer is built into every web browser so it is very cross-platform in nature. It is limited in that it is static, so interactivity with an HTML document is generally in the form of a link to another (possibly dynamically generated) HTML document. The vast majority of the user interface in POWERWEB consists of dynamically generated HTML pages.

Perl [2]

Perl is a language originally designed as a UNIX administration tool. It has become tremendously popular with web developers as a language for writing programs which generate HTML pages as output. One of Perl's many strengths is in the area of text handling, which is exactly what is needed for producing and manipulating HTML. Perl's operating system, file system, network and database interface capabilities along with its object-oriented language features, uniquely coordinated

developer community, and extensive archive of high quality freely available reusable modules, make it an ideal choice for many of POWERWEB's tasks.

Java [3]

Java is a complete programming language that allows true platform-independent application development. It was developed by Sun Microsystems and has been submitted to the open standards process. It is an object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high-performance, multi-threaded, and dynamic language. Of particular significance to developers of network applications is the ability of a Java application class, called an "applet", to be securely downloaded from anywhere on the Internet. The application can then be loaded dynamically and executed immediately. It is simple to place references to Java applets into an HTML document. Users may then invoke an applet by simply accessing the relevant page.

POWERWEB currently uses Java applets to display the one-line diagram of the power system and to display cost and offer information graphically. In spite of the fact that there is some inconsistency across implementations, it promises to be a dominant player in the development of network-centered applications like POWERWEB.

JavaScript [4]

JavaScript is an interpreted scripting language developed by Netscape Communications which has been standardized as ECMA-262. Contrary to the implication of the name, it is not based on Java. JavaScript code can be embedded within an HTML document where it is executed by the web browser in response to specified events. For example, a button can be linked to some JavaScript code that executes when the button is pressed.

One application of JavaScript in POWERWEB is to create a button used to fill in an offer submission form with the values from a previously submitted offer.

SQL

The Structured Query Language (SQL) is a standard language for defining, querying and manipulating the data in a relational database. POWERWEB uses SQL extensively to access and modify data in its database.

Matlab

Matlab, the language, is an interpreted, procedural language developed by The MathWorks and designed for numerical mathematics, especially applications involving matrix and vector computations. It includes highly optimized dense and sparse matrix factoring routines among many others. Until the most recent version, Matlab was quite limited in the data structures available, but its strength in matrix and vector computations still make it a tool of choice for the types of computations required for power system simulations.

POWERWEB uses Matlab as the language for implementing most of the optimal power flow programs as well as the market pricing code which form the core of POWERWEB's computational server. Since Matlab is not explicitly designed as a network language, it was necessary to develop our own protocol for interacting with the Matlab programs. Some of the optimization routines are actually Fortran routines called from within a Matlab program.

2.2 Protocols

HTTP [5]

The HyperText Transfer Protocol (HTTP) is the standard protocol for communicating between clients and servers on the web. HTTP is a stateless protocol, which specifies how a client and server establish a connection, how the client requests a specific service from the server, how the server issues a response, and how the connection is terminated. The terms "client" and "server" are defined primarily in terms of their roles in an HTTP interaction. HTTP connections over the Internet are implemented using the TCP/IP protocol.

In POWERWEB all interaction between the web browser and the web server are based on HTTP, as are all communications with the computational server.

CGI [6]

Common Gateway Interface (CGI) is a very popular standard protocol for communication between a web server and an external program, typically referred to as a CGI program. The primary role of a CGI program is to dynamically create data on demand, such as a web page or image, for the web server to return to a client. Since the CGI protocol clearly defines the interface to the web server, any language that can implement this interface can be used to write a CGI program. One of

the limitations of CGI is the performance penalty arising from the overhead involved in spawning a new process for each request. FastCGI [7] is a lesser-used alternative that allows the external program to continue to run between requests to avoid this overhead. Most web servers also have application programming interfaces (APIs) which allow developers to directly extend the web server functionality to be able to generate dynamic pages.

POWERWEB uses CGI programs implemented in Perl for nearly all of the dynamically generated HTML pages which make up POWERWEB's user interface. Some of the other alternatives mentioned are being considered for future versions.

Cookies [8]

An HTTP cookie is an object containing state information, a simple name and value pair, that a web server informs a web browser to send along with any subsequent requests to a specified range of URLs. POWERWEB utilizes cookies to store login information to avoid requiring a user to type in their password for each page they want to access.

URL [9]

A Uniform Resource Locator (URL) is the standard means of identifying and locating any network resource. URLs are used to address specific web pages including those that may be generated dynamically by a program. POWERWEB uses URLs to identify the HTML documents that make up its user interface.

MIME [10]

The Multipurpose Internet Mail Extensions (MIME) standard specifies the type of data and encoding associated with a document. POWERWEB uses MIME to specify the type of data transmitted over the HTTP connections between clients and the web server and computational server.

DBI [11]

DBI is the database interface module and API for accessing SQL databases from Perl. It is designed to be independent of the database server being used. It is an ideal interface for POWERWEB to use to access its database server from the Perl CGI programs.

JDBC [12]

The Java Database Connectivity (JDBC) is the standard protocol for accessing a database from the Java programming language. POWERWEB uses JDBC for all of its Java-based database access.

3 Communications Architecture

POWERWEB employs a distributed architecture on several different levels. First, it is a client server architecture, in that all user interaction with POWERWEB is via a web client (a browser, or applet running within a browser) communicating with the POWERWEB server. Second, the POWERWEB server also has a distributed architecture consisting of several independent processes, such as the web server, the database server, and the computational server, each of which can be running on different computers. Even the computational server has several parts that need not reside on a single machine.

Figure A4 illustrates the various components of the POWERWEB communications architecture. Currently, the entire POWERWEB server is running on a Sun Ultra 2200. The web server is an Apache server [13] and the database server is mySQL [14].

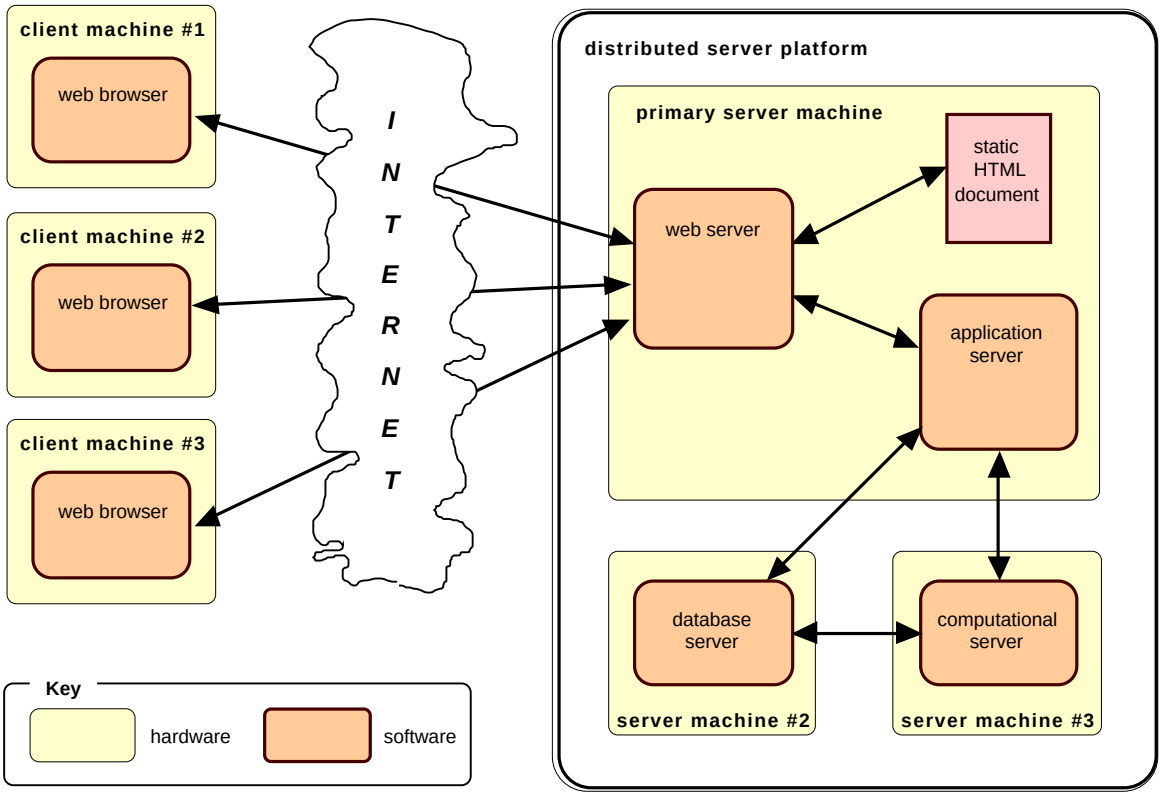

Figure A4 POWERWEB communications architecture

When a request for a specific URL is transmitted from one of the web clients to the web server, the server determines whether the URL refers to a static HTML document or to one that must be generated on the fly. In the first case, the web server retrieves the file from the disk and returns it to the client. In the second case, the web server passes the request on to what is referred to here generically as an "application server". This could be a separate program invoked via a CGI or FastCGI protocol, or it could simply be a server module that runs to produce the document to be returned.

The vast majority of accesses to POWERWEB are processed by the application server that, in turn, makes requests to the database server. In the current implementation, the application server is a CGI program written in Perl. All accesses subsequent to login are accompanied by a cookie containing authentication data that is compared with information provided by the database. The

protocol for communication with the database server is a specialized protocol defined by the developers of mySQL, which uses UNIX sockets for local communication and TCP/IP sockets for remote communication. The computational server also receives requests from the application server via the HTTP protocol. Based on the parameters sent with the request, it retrieves the necessary data from the database, runs the requested computation, and returns the output to the application server. The computational engine is implemented as a web server with a CGI program that makes database queries and invokes Matlab to perform the computations.

One of the difficulties presented by standard web protocols is the inability to contact a client when it needs to be made aware of new information. The standard web protocols facilitate only the following sequence: a client connects to the server with a request, the server responds and closes the connection. The HTTP protocol is a state-less protocol that does not provide the ability for the server to initiate a communication with a client. One way to overcome this limitation is for the server to keep a "live" connection to a Java applet running at each active client. This "live" connection must be handled via another server process. A typical use of this "update server" would be if the user acting as ISO triggers a recomputation of the dispatch and price schedules, when the computation is completed, the server would pass a message to the update server to notify the clients to retrieve the new information. The current POWERWEB implementation utilizes "server-push" technology to implement some of the functionality of such an update server. Server-push allows the web-server to keep the connection open after returning a page to the browser. When updated information is available, the server can send an updated page which replaces the one currently displayed by the browser.

4 Database structure

In the POWERWEB environment there is a tremendous amount of data that needs to be handled and used in varying contexts. A relational database server satisfies the needs for logical data organization with its relational model, synchronized updating of the data to maintain data integrity, and flexibility in access to the data via the SQL language.

In the interactive Internet-based environment, performance of the database server is also of utmost importance. The mySQL server [14] used in POWERWEB meets these requirements nicely.

The data handled by POWERWEB can be classified into three main categories:

- user administration data

- power system data
- session data

4.1 User administration data

The user administration data is used primarily to control who has access to what information in POWERWEB. When a request is made for user *X* to see cost information for a generator "owned" by group *Y*, for instance, these tables would be accessed to determine whether the request is coming from someone who is authenticated as user *X*, and to ensure that user *X* really is a member of group *Y*. These data are stored primarily in three tables:

Users user id, password, registration info

Groups group id and name

UserGroup mapping of users to groups

4.2 Power system data

The power system data refers to the coordinate data needed to display a one-line diagram of the system, all of the power flow data needed to run an optimal power flow (OPF), the results of the OPF, and the cost information required to compute profits given the resulting dispatch and price schedules. These data are held in the tables described below.

The main table which contains the top-level data for each base case, one row per case, is the *Systems* table. Each row of each of the *Areas*, *Buses*, *Branches*, *BranchSegments*, *Caps*, *Gens*, and *Loads* tables has a field which links it to a system in the *Systems* table and another which is the index of that particular area, bus, branch, etc. within that system. In addition, each row of the tables contain the following data:

Systems system id, system name, MVA base, number of buses, lines, gens, etc.

Areas area name, price reference bus

Buses area number, zone, voltage class, bus type, upper and lower voltage limits, initial voltage magnitude and angle, one-line coordinates

Branches "from" and "to" bus numbers, circuit number, line status, impedance and charging parameters, thermal capacity ratings, tap ratio and phase angle shift

BranchSegments one-line coordinates of branch segments

Caps bus number, status, admittance, one-line coordinates

Gens bus number, gen name, baseMVA, status, initial active and reactive power generation, initial voltage magnitude set-point, upper and lower limits on active and reactive output, ramp rate, min up and down times, key to entry in *GenCosts* table

GenCosts cost class, cost model, startup and shutdown costs

GenCostData key to entry in *GenCosts* table, parameters which define polynomial or piece-wise linear production cost curve

Loads bus number, real and reactive demand, percentages of constant impedance, constant current, and constant power

The data for the (optimal) power flow solutions is stored in separate tables, to avoid having to store all of the constant power flow data for each case that is run. The *Solns* table holds the top-level data, one row per solved case, to which the other tables are referenced. Each of the *BusSolns*, *BranchSolns*, and *GenSolns* tables has a column which links each record to an entry in the *Solns* table and another which is the bus, branch, or generator index. In addition, the following information is stored in theses tables:

Solns solution id, name, system id

BusSoln bus type, voltage magnitude and angle, Lagrange multipliers for real and reactive power balance requirements, Kuhn-Tucker multipliers for upper and lower voltage constraints

BranchSoln real and reactive power flow at each end of the branch, Kuhn-Tucker multipliers for flow constraints

GenSoln status, active and reactive power output, voltage set-point, Kuhn-Tucker multipliers for upper and lower real and reactive output constraints

The *Changes* and *ChangeData* tables provide a convenient way to specify modifications to an existing base case. Each row in *Changes* corresponds to an independent set of modifications that can be applied to an existing base case. *ChangeData* has a column that associates that particular change with a corresponding set in *Changes*.

Changes change set id, valid system id (0 for any system), name of change set

ChangeData table, column and index of data to be modified (all indices if index is 0), type of change (scale or replace), scale or replacement value

4.3 Session data

All interaction with POWERWEB is in the context of a "session" that the user is logged in to. The tables listed in this section handle the data for managing these sessions. The main top-level data is stored in the *Sessions* table, with one row per session. Many of the other tables here have references to a particular entry in this table which link their data to a specific session.

Sessions session id, name, user id of session owner, system id, market id, number of trading periods, number of dispatch periods, number of iterations per trading period, length of dispatch period, time given for each iteration, persistence level, auto vs. manual computation mode, logging detail level, creation time, start time, simulation clock time, session state, current trading period, current dispatch period, current iteration number, textual session description

The *Markets* table has an entry for each type of market implemented in POWERWEB. The table stores a few parameters common to the various markets, but the rules of each market are programmed separately in POWERWEB code.

Markets market id, name, type of offer (blocks, functions), auction id, offer dimensions

The *Resources* and *ResOwners* tables specify which system resources (e.g. generators) are "owned" by which user or group.

Resources resource id, session id, resource type (gen, load), index, name

ResOwners resource id, user id or group id

Each POWERWEB session is organized into a sequence of trading periods, during which offers are made to sell power to meet a forecasted demand schedule. The schedule for a given trading period may be divided into several dispatch periods, each of which has its own demand forecast, its own actual demand, and possibly its own set of other arbitrary changes to network parameters. A "system profile" is used to specify how the system parameters, including demand, vary through out the various trading periods and dispatch periods. The *SystemProfiles* table defines the system parameters used for each period.

SystemProfiles session id, trading period, dispatch period, type of profile (forecasted or actual), sequence number, change set id

In addition, the submission of offers and computation of dispatch and price schedules may be iterated several times for each period in the system profile. Each of these iterations is treated as a separate "case" to be run. The *Cases* table associates an id with each iteration of each period, which the other tables, can use as a reference. *CaseIOData* associates each case with it's set of offers and dispatch results, and *CaseSolnData* matches each case with its solution in the *Solns* table.

Cases case id, session id, trading period, dispatch period, iteration, state

CaseIOData case id, resource id, type of IO data (real power, reactive power, etc.), offer id, dispatch id

Offers offer id, sequence (block) number, quantity, price

Dispatches dispatch id, quantity, price, fixed cost, variable cost, startup cost, penalty, profit

CaseSolnData case id, solution id

5 Underlying Optimal Power Flow

At the heart of the POWERWEB computational engine is an optimal power flow (OPF) program that is executed by the ISO in response to offers submitted in an auction. The market activity rules determine what offers are valid, but it is the ISO's role to ensure the safe and reliable operation of the network. By using an OPF, the ISO can legitimately allocate generation in an "optimal" way while respecting line flow constraints, voltage magnitude constraints, VAr constraints and any other constraints that are necessary to ensure safety and reliability. As a by-product, the OPF also produces the shadow prices associated with locationally based marginal pricing (LBMP) of power. These prices can be used as determined by the market mechanism being employed.

In the context of a market in POWERWEB, the OPF may be subjected to widely varying costs and therefore dispatches which are far from typical base case operation. It is important in such an environment that the OPF be extremely robust. The latest release version of the Matlab OPF solvers used in POWERWEB and more detailed documentation of the algorithms employed are available at no cost at <http://www.pserc.cornell.edu/matpower/> as part of the MATPOWER package [15].

6 References

1. “IETF - HyperText Markup Language (HTML) Working Group”, <http://www.ics.uci.edu/pub/ietf/html/ >.
2. “The www.perl.com Home Page”, <http://www.perl.com/>.
3. “Java Documentation”, <http://www.javasoft.com/docs/>.
4. “JavaScript Guide”, <http://developer.netscape.com/docs/manuals/index.html?content=javascript.html>.
5. “IETF - Hypertext Transfer Protocol (HTTP) Working Group”, <http://www.ics.uci.edu/pub/ietf/http/>.
6. “The Common Gateway Interface”, <http://hoohoo.ncsa.uiuc.edu/cgi/overview.html>.
7. “FastCGI”, <http://www.fastcgi.com/>.
8. “Persistent Client State HTTP Cookies” <http://www.netscape.com/newsref/std/cookie_spec.html>.
9. “IETF - Uniform Resource Identifiers (URI) Working Group”, <http://www.ics.uci.edu/pub/ietf/uri/>.
10. “MIME (Multipurpose Internet Mail Extensions”, <http://www.oac.uci.edu/indiv/ehood/MIME/MIME.html>.
11. “DBI - A Database Interface Module for perl5”, <http://www.symbolstone.org/technology/perl/DBI/index.html>.
12. “The JDBC™ Database Access API”, <http://www.javasoft.com/products/jdbc/>.
13. “Apache HTTP Server Project”, <http://www.apache.org/httpd.html>.
14. “mySQL Home Page”, <http://www.mysql.com/>.
15. R. Zimmerman and D. Gan, “MATPOWER: A Matlab Power System Simulation Package”, <http://www.pserc.cornell.edu/matpower/>.