2026 ASEE ANNUAL CONFERENCE June 21–24, 2026 | Charlotte Convention Center, NC
Host Sponsor UNC Charlotte
Paper ID #50058

Gamification of the US Electricity Grid: Price Optimization in an Environmental Landscape

Lila Baxter, Ohio Northern University
Ian Meyer Kropp, Ohio Northern University
Dr. Lauren H. Logan, Ohio Northern University

Dr. Lauren H. Logan is an associate professor of civil and environmental engineering at Ohio Northern University. Her research explores the energy-water nexus, with a focus on energy economics and the intersections of biology and policy within engineering.

WIP: Gamification of the US Electricity Grid: Price Optimization in an Environmental Landscape

Lila Baxter[a], Ian Kropp[b], Lauren H. Logan[c]
a) undergraduate student in computer science (graduation May 2026), Ohio Northern University
b) assistant professor of computer science, Ohio Northern University
c) associate professor of civil & environmental engineering, Ohio Northern University

Motivation for Grid Gamification

The United States electricity grid is a highly complex system. Many factors play a role in determining grid reliability, stability, and electricity prices. For consumer prices, factors such as fuel cost, operations and maintenance, overall power demand, and weather patterns affect costs at the individual power plant scale, which in turn impacts market prices [1]. These fluctuations in market price impact the real-time (e.g., marginal) cost of electricity, but the average residential consumer responds to average market prices (e.g., the ostensibly consistent "flat" monthly rate as shown on a typical energy bill) [2]. As such, we might infer that the intricacies of our electricity market are not well-understood by the majority of residential electricity consumers, and average price billing does not incentivize consumers to reduce or change their electricity usage habits [3].

To model the interrelationships between factors affecting market price, and thus better educate students (who are also residential electricity consumers), we created an optimization tool for use in undergraduate engineering courses teaching energy grid concepts, with expansion potential into high school STEM courses to improve overall electricity-consumer literacy. Often overlooked at individual and regional-scale market models are environmental factors such as plant water usage (e.g., withdrawal and consumption in L/kWh or gal/kWh), and carbon footprint (e.g., CO_{2eq}/kWh). These environmental factors are termed externalities, or external factors, and unfortunately are rarely, if at all, considered in decision-making processes associated with electricity cost [4]. Some studies explore plant-type environmental impacts within an efficiency and/or cost landscape [5] [6] or environmental impact within a fuel-mix context [7]. However, a holistic approach to electricity cost, environmental impact, and consumer understanding is needed for integration into courses covering the fundamentals of our US electricity grid.

Methods and Current Status

Via optimization, we simulate environmental impact, operations and maintenance, and market volatility in relation to real-time electricity price. Since these objectives are often at odds with each other (e.g., conflicting interests), we simulate grid operations as a multiobjective optimization problem. Multiobjective optimization is a large and vibrant field of research on how to solve multiobjective problems effectively [8]. We hypothesize that teaching students grid optimization from a multiobjective viewpoint will help them understand the operation of the electricity grid from a systems perspective. Moreover, our work demonstrates how varied interests can impact electricity prices, while also allowing users to experience market functionality via gamification of complex concepts. Gamification of such complex topics will enhance student learning and engagement, as has been shown to be beneficial for learning in programming and other contexts [9].

The algorithm used for this particular optimization is the Non-dominated Sorting Genetic Algorithm (NSGA-II), which is a genetic algorithm for solving multiobjective optimization. Genetic algorithms (GA) are a type of evolutionary algorithm that simulates the evolution of a population based on the principle of survival of the fittest to find near-optimal solutions [10]. The NSGA-II implements a standard genetic algorithm with slight modifications to the mating and survival selection procedures [11] in order to produce a Pareto front of optimal solutions. At the end of the procedure, weights are used to select a single "best" solution from the aforementioned Pareto front according to the preferences of the decision maker. In the following paragraphs, we outline our methodology in three stages: 1) data processing, 2) problem formulation, and 3) problem configuration.

Data Preprocessing

In order for the optimization to be easily integrated into a game, it must access and format power plants' data, along with the total demand on the grid, provided in an input GDScript dictionary file (i.e., a JSON-like file). By default, our model includes sixteen fictitious[1] power plants of varying type and cost, but the game can be configured with a user-defined set of power plants. The data in each row is transformed into the attributes of a greater PowerPlant Python object with ID, name, type (e.g., coal, nuclear, solar, wind, natural gas, hydro), production cost (static, wholesale $/kWh), minimum power output (minimum MW operational parameter), maximum power output (nameplate capacity, MW), current demand (total power demand, MW), and wholesale price fields. This process is then repeated for every plant on the sheet.

This optimization also implements weather features: *sun* and *wind* values (percentages, meant to represent the current availability of sun and wind in a given region) used to calculate how much energy the solar and wind plants can produce under differing weather conditions. The game allows for optimization of water usage and carbon footprint, in addition to the cost of generation; and a weighting system in which each objective is assigned a weight between zero and one, with all weights together adding to 1.0, or 100%.

---

[1] All power plant data is informed by literature and Energy Information Administration online datasets, such as form 923 downloadable data [18].

Problem Definition

Since implementations of NSGA-II already exist in Python, our key task is to define the power plant optimization problem in a format that the GA library can optimize. This includes representing a given energy portfolio as an array of integers, writing objective functions that measure the impacts of a portfolio (i.e., price, carbon footprint, water footprint), defining a population initialization procedure, and defining constraints for the problem.

The first challenge in designing this problem is how to represent an energy portfolio in the problem. Using a GA, we represent each portfolio as an integer array (x), each element x_i having a lower and upper bound ($x^l \leq x_i \leq x^u$). However, there are actually two lower bounds to most power plants: the minimum output while operating in MW x_i^{min} , and the minimum output when shut off (i.e., 0 MW). For example, nuclear power plants have high capacity factors (power produced compared to maximum power potential, reported as a percentage) [12], in part due to the stable nature of nuclear reactions once generated. In other words, a nuclear power plant is not likely to operate below a capacity factor of 70%. There is also a maximum output value for each power plant, x^{max}. A naïve approach to addressing this design challenge is to set $x^l = 0$, and then make constraints to make any value between 0 and x_i^{min} infeasible. However, this poses a challenge to the optimization, as discontinuous search spaces are difficult for the algorithm to navigate, negatively affecting performance.

To overcome these challenges, we created a normalization routine that mapped the physical power plant values to integers between $\left\lfloor \frac{1}{4}\ (x^{min} - x^{max}) + \frac{1}{2} \right\rfloor$ and $\left\lfloor 4(x^{max} - x^{min}) + \frac{1}{2} \right\rfloor$ for x_i^l and x_i^u respectively. The $+\frac{1}{2}$ and floor notation represent rounding to the nearest integer. When converting x_i back to a physical power plant output level, negative numbers are normalized to zero (i.e., the power plant is shut off), and the positive numbers are added to the associated plants' minimum outputs. This provides a lower boundary a small distance into the negatives and a higher positive upper boundary, which provides the GA a continuous search space without discontinuities. If a plant then exceeds its maximum generation capability, the amount generated is simply set to its max. This process ensures that each plant is either generating power that rests within its bounds, or the plant is turned off. In other words, our example nuclear plant from before will not be caught operating at infeasible capacity factors like 15% or 110%.

For population sampling, each run begins by generating an initial population of megawatts generated per plant. Each portfolio of plants and their outputs are considered to be an individual in a population. The script will generate random power portfolios until a set number of individuals are created to represent the initial population.

The script then takes this initial population and, after checking whether or not each plant's power generation falls within its respective bounds, proceeds to the objective function. The objective function is what produces numbers (in this case, the overall cost to generate the plants' power, the total water usage, and the total carbon footprint) to be minimized (i.e., optimized). In order to calculate the overall generation cost, each individual plant's energy output (MW) is converted to kilowatts (kW), then multiplied by the plant's specific generation cost ($/kWh) for consumer pricing. These individual plant costs are then added together to find the total cost. This calculation can be represented with the equation $\sum(x_i * y_i)$, where *i* represents the current plant, x_i is the current plant's power generation in kW, and y_i is the current plant's cost per kilowatt-hour. For the purposes of in-game calculations of grid operational costs, the difference between energy (kWh) and power (kW) is neglected to remove the time element of pricing (the hours unit in kWh). Calculations of the total water usage and carbon footprint follow the same equation, with y_i representing the current plant's water use (L/kWh) and carbon footprint (CO_{2eq}/kWh), respectively.

Before a population can be considered as a potential solution, however, it must be verified to meet all necessary criteria. Here, there are two criteria that need to be met:

1. Every plant in the population must *either* generate an amount of energy that falls within its given bounds *or* be turned off (generating zero MW), and
2. The total amount of power being generated must meet or exceed the demand on the grid.

Criterion 1 is handled by the normalization routine, as no solution can go between 0 and the minimum output of each power plant. For criterion 2, a variable is set to the number of megawatts that the proposed population falls short of the demand (if demand is met or exceeded, it's simply set to 0). The verification of these criteria is then performed by a constraint function $\sum x_i + y$, where x_i represents a single constraint violation from criterion 1 and *y* represents a violation of criterion 2. If a population meets all constraints mentioned above, the function should equal zero; if it does not meet one or more constraints, the function will produce a number greater than zero. Thus, if the constraint function's result is zero, the population is considered a potential solution; if it is greater than zero, the population is thrown out.

Run Configuration

Each session of the GA required several trials with differential random seeds. This is common practice in the GA literature, as it ensures that no one single optimization run is developing abnormally good or bad solutions. In order to minimize execution time, the optimization script begins by splitting the provided number of trials of the program evenly across the number of threads allocated (for testing purposes and throughout this paper, 16 runs and 8 threads are used). In summary, each thread runs the program *runs / threads* times (e.g., 16 runs / 8 threads = 2 trials per thread), each with a new random seed. Once all threads finish executing all runs of the GA, we are left with a list of each run's best result. From here, the lowest overall cost that still meets all constraints is chosen and returned as the best, or optimal, answer.

For optimization, two strategies were run: *strategy 1,* which is a single-objective problem that minimizes cost, and *strategy 2,* which is a multiobjective problem that minimizes costs, carbon footprint, and water footprint. *Strategy 1* helps us validate whether or not the optimization is performing as well as a manual grid operator optimizing cost. *Strategy 2* demonstrates the full systems perspective of multiobjective optimization.

The NSGA-II algorithm is implemented via the *pymoo* (Python Multi-Objective Optimization) library [13]. The NSGA-II implementation was used with a few modifications to ensure that the functions are only working with integers, not floating-point values, as they add a lot of complexity and therefore increase run time. The population size was set to 250, which balanced time efficiency and accuracy, and 50 generations were used, as more generations did not improve accuracy.

Preliminary Results

The final step in the optimization is graphing the results to validate the performance. In the project's current state, we have four major results. For *strategy 1,* we analyzed the following results:

1. Production and price by plant
2. Cost per generation of both the best result and all results
3. A sensitivity analysis on the effects of differing numbers of generations on the results

And for *strategy 2,* we analyzed the Pareto front of multi-objective optimization where all constraints are weighted equally.

Strategy 1 Results

Figure 1 shows how much electricity (technically power, in megawatts) each plant produces (shown as the width of the bars) along with the price to do so ($/kWh, shown as the height of the bars). The wholesale price, or the market price paid by the consumer, is the price of the most expensive plant used and is indicated by a horizontal black line. Only the best result is displayed in this power plant generation graph in Figure 1. This price optimization is an oversimplification of complex market interactions, but demonstrates how the "market stack" is generated to determine electricity pricing. The approach to cost determination and external factors impacting cost follows similar work on grid pricing structures [14]. For a review of electricity markets and pricing, see work by Forero-Quintero et. al [15], and others.

Figure 2 visually demonstrates the run time per GA generation for both the overall best result found, and the best results of each time the algorithm ran. Each generation is indicated by a vertical bar. If a bar is red, the corresponding generation's solution did not meet the constraints necessary to be a valid solution. Green bars, then, represent generations that successfully met all criteria and could then be improved upon in future generations. As the graphs show, the total cost increased rapidly until the criteria had been met, after which the cost slowly decreases as the algorithm works to optimize it.

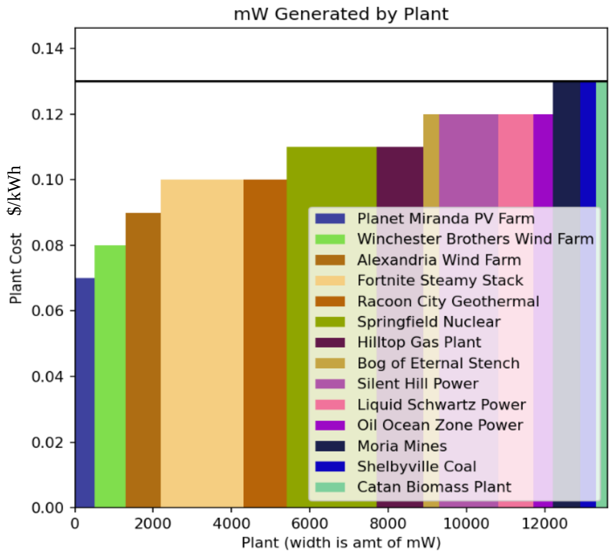

Figure 1: Individual power plant production levels (width, MW) and associated cost, stacked from least expensive to most expensive. Power plant names are from an in-class, on paper version of the game. [These pop-culture references were for amusement only, and will not be used in the finalized version of the game.]

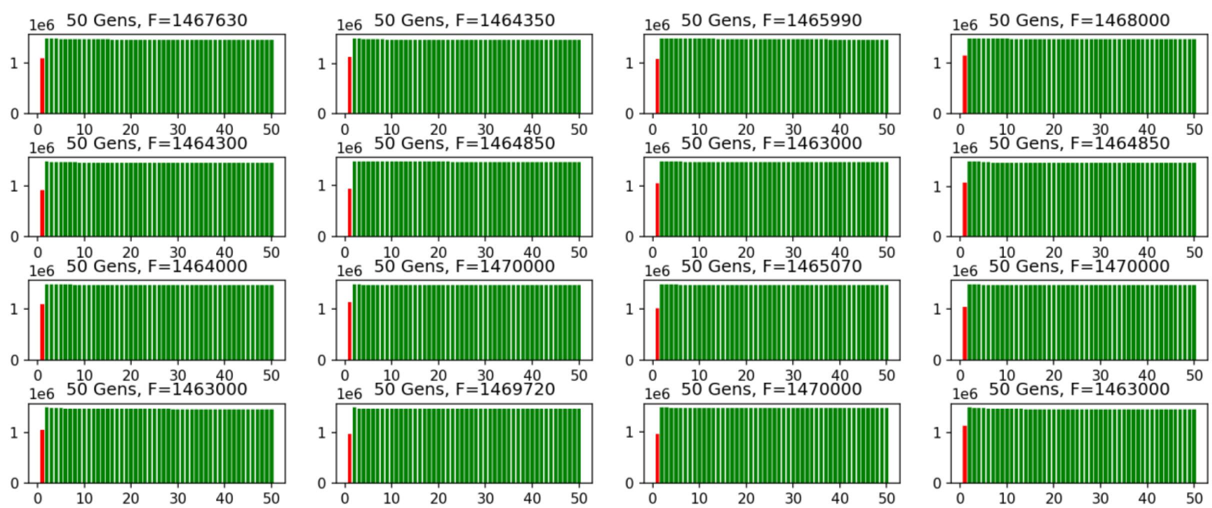

Figure 2: Comparison of generations and solution optimization. Sixteen runs are shown here, with different seeds. The array of runs demonstrates that in a typical run, the solution is optimized before 50 generations.

Our final analysis for *strategy 1* (Figure 3) illustrates the effect on the final result of changing the number of generations for which the algorithm ran. This analysis was designed to find the specific run of the algorithm after which adding further generations no longer improved the cost optimization, and thus instead only added unnecessary time to the analysis. In other words, finding the tradeoff between run time efficiency and result accuracy.

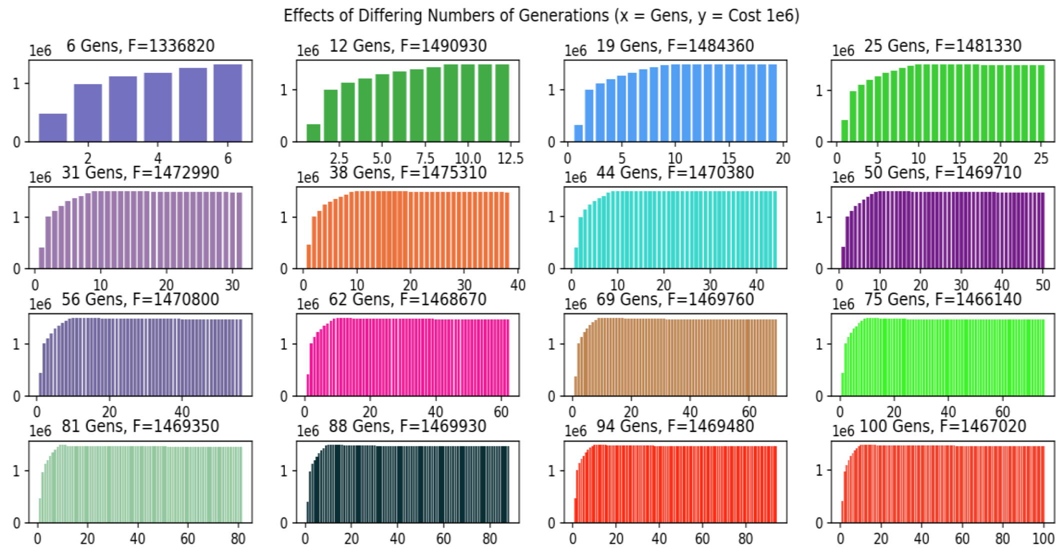

Figure 3: Sensitivity analysis on the number of generations compared to time needed to reach an optimal solution.

Strategy 2 Results

In addition to the single-variable optimization in relation to cost, a basic multiobjective optimization has been created in which cost, water footprint, and carbon footprint are weighted equally (default setting), or can be adjusted based on user preference (e.g., if water is a concern, the user could select water weighting as 50% of the total decision, carbon as 25%, and cost as 25%). The result of this multiobjective optimization is a Pareto front, in which the optimal electricity mix is denoted with a star, and all other generation mixes appear in the Pareto front. An example run of the multi-objective optimization is shown in Figure 4.

Pareto Front

In this game's current state, the optimization routine can balance the energy grid with respect to cost alone, or weight cost, carbon footprint, and water footprint. From a pedagogical view point, only considering the cost does not give students a full understanding of the electrical grid as a system. In other words, in today's workforce, it is no longer sufficient for engineering students to simply understand principles of engineering as isolated concepts, and ideally, they should be learning from a systems perspective [16]. Undergraduate students must enter the workforce understanding how to make design decisions that simultaneously balance systems with social, economic, and environmental objectives. Therefore, adding environmental objectives into our grid optimization framework is both logical and necessary. For example, each type of power plant contains environmental impacts (e.g., carbon and water footprints) which should be taken into account along with cost. To demonstrate the complexities of competing values, our multiobjective optimization provides users with the opportunity to compare and assess the impacts of environmental decision-making on consumer electricity prices.

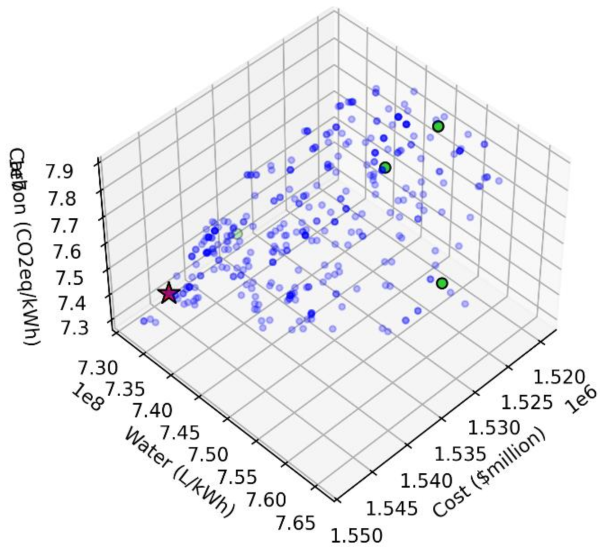

Figure 4: Pareto front for a multiobjective optimization run where cost, carbon footprint, and water footprint are given equal weight. The star indicates the algorithm's selected "best" option given the optimization parameters.

With complexity of real-world challenges comes complexity of optimization. Specifically, it is impossible to minimize carbon, water footprint, and cost objectives in a single solution for all sets of power plant conditions. For example, say an operator needs to pull 50 MW from either a nuclear power plant or a natural gas plant, in this scenario having the same marginal cost of electricity (e.g., $/kWh). Pulling all the power from a nuclear power plant may have a carbon footprint of zero (in the use-phase), but a nuclear plant uses more water than a non-zero carbon footprint natural gas plant (per electricity unit produced) [17]. Weighing each objective equally, each of these two choices are equally optimal, and human decision makers need to choose the best solution for the given scenario and stakeholder. In a water-stressed area, the natural gas plant might be the better solution, but at the environmental cost of increased carbon footprint. This is why accounting for a decision-makers influence on a decision, via weight of each constraint, is important to model real-world complexities in the game.

Future Work

Extending our current work into an educational multiobjective optimization software platform has followed three distinct stages: *Stage 1*) mathematically formulating our environmental objectives and constraints (i.e., water and carbon footprints), *Stage 2*) ensuring the optimization platform yields optimal solutions quickly enough to operate in a classroom setting on lab or personal computers, and *Stage 3*) designing a user interface (UI) for the students to run the optimization without required knowledge of Python coding. Stages 1 and 2 have been discussed in our preliminary results, and we are currently in progress on stage 3.

For *Stage 3* development of the game, a user-friendly interface for the optimization platform will be completed. In the project's current state, there exists a dummy-game framework created in Godot, complete with power plant characteristics and menu screens. The user interface was designed with input from college students during an optional focus-group session, where participants were informed of the game and research objectives, given a brief lesson on grid operations and power plant dynamics, then asked to sketch out game menus/features while providing verbal feedback to the designers. For the current optimization aspect of the game, a user must either know how to run Python scripts from the command line, a Python IDE, or from a Jupyter Notebook. Though undergraduate civil engineering students are technically savvy individuals, these coding skills are outside of their reasonable expected knowledge in a non-coding course environment. Therefore, it is essential that the Python optimization script be integrated into a simple UI that allows users to easily and quickly run the optimization. The UI for the grid game is currently in progress, and concept art has been generated for various menu screens (Figure 5) and power plant types (Figure 6).

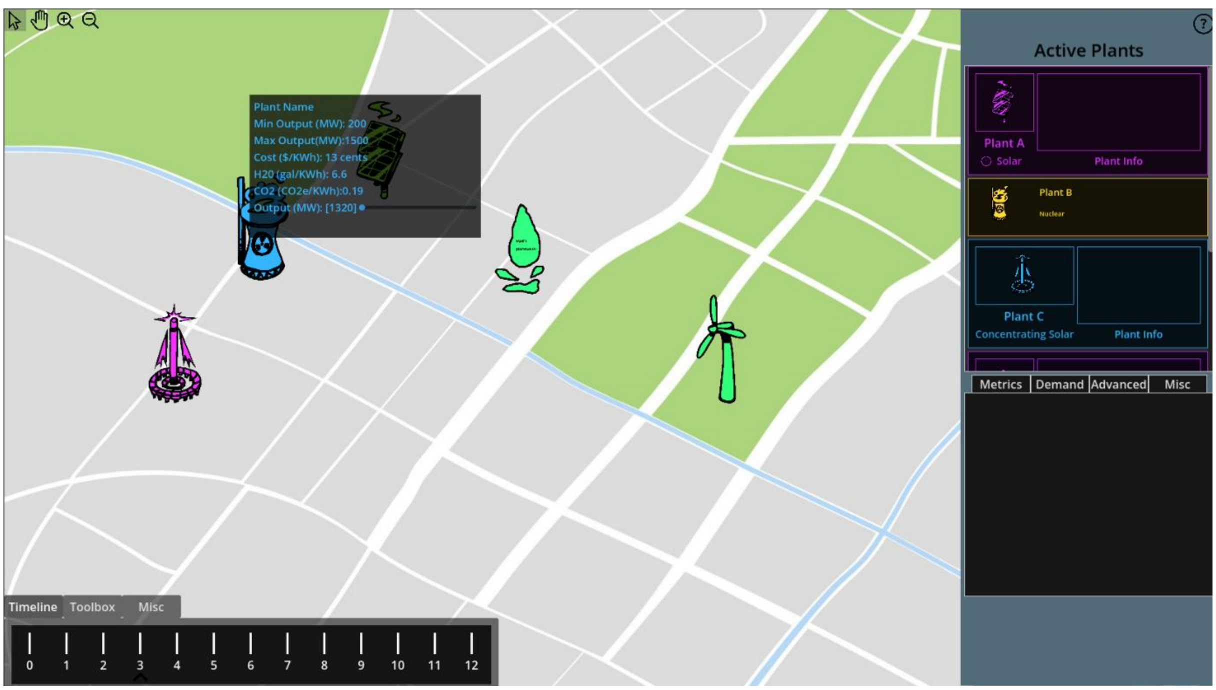

Figure 5: Screenshot of in-game play showing power plant placements, with each active plant comprised of operational data (min and max MW output), price data ($/kWh), water footprint (L/kWh), and carbon footprint (CO_{2eq}/kWh).

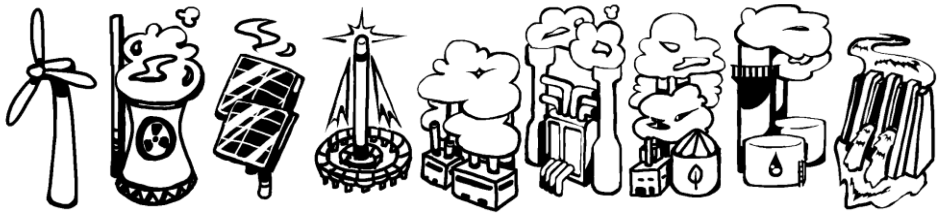

Figure 6: Early concept art for power plant types, which will be incorporated into gameplay as power plant icons used in plant placement.

Next steps include finishing the game in Godot, and developing instruction manuals complete with lesson plans for educators. Following user-manual creation, play-testing the game in a classroom environment will provide the opportunity to gather qualitative feedback from educators and students alike. Further focus group sessions and feedback surveys from play-testers will inform final versions of the game.

Acknowledgements
This work is funded by the James L. Kennedy Endowed Professorship for Energy Economics. The inclusion of student focus groups and surveys for game design falls under IRB approval via Protocol #: LL-EN-021826-1, Title: Gamification of power grid economics, Exempt category #2 – Tests, surveys, or observation of public behavior.

The authors wish to acknowledge the major contributions to game development by Jane MacDonald. Without her efforts, the game would not be what it is today.

References

[1] M. S. Merve, K. Gokhan, D. Levent and Y. Veli, "An Empirical Investigation of Explosive Price Behavior in U.S. State-Level Electricity Markets Across Consumer Segments," *Energies,* vol. 18, no. 21, 2025.

[2] K. Ito, "Do Consumers Respond to Marginal or Average Price? Evidence from Nonlinear Electricity Pricing," *The American Economic Review,* vol. 104, no. 2, 2014.

[3] F. A. Wolak and I. H. Hardman, "Current State of Retail Markets," in *The Future of Electricity Retailing and How We Get There,* Springer, 2021, pp. 37-54.

[4] J. L. Cohon, "Hidden costs of energy: unpriced consequences of energy production and use," The National Academies Press, Washington, D.C., 2010.

[5] T. Sueyoshi, M. Goto and T. Ueno, "Performance analysis of US coal-fired power plants by measuring three DEA efficiencies," *Energy Policy,* vol. 38, 2010.

[6] B. Hampf, "Cost and environmental efficiency of U.S. electricity generation: Accounting for heterogeneous inputs and transportation costs," *Energy,* vol. 163, 2018.

[7] I. W. Parry, D. Heine, E. Lis and S. Li, "Energy Systems, Environmental Problems, and Current Fiscal Policy: A Quick Look," in *Getting Energy Prices Right: From Principle to Practice,* 2014.

[8] K. Deb, Multi-objective Optimization Using Evolutionary Algorithms, John Wiley and Sons, 2009.

[9] B. MARÍN, J. FREZ, J. CRUZ-LEMUS and M. GENERO, "An Empirical Investigation on the Benefits of Gamificationin Programming Courses," *ACM Transactions on Computing Education,,* vol. 19, no. 1, 2018.

[10] D. Ganten and K. Ruckpaul, Encyclopedic Reference of Genomics and Proteomics in Molecular Medicine, Springer, 2006.

[11] [Online]. Available: https://pymoo.org/algorithms/moo/nsga2.html.

[12] Energy Information Administration, "Electric Power Monthly, Table 6.07.B. Capacity Factors for Utility Scale Generators Primarily Using Non-Fossil Fuels," 2026. [Online]. Available: https://www.eia.gov/electricity/monthly/epm_table_grapher.php?t=epmt_6_07_b. [Accessed April 2026].

[13] J. Blank and D. Kalyanmoy, "Pymoo: Multi-objective optimization in python," *IEEE Access,* vol. 8, pp. 89497-89509., 2020.

[14] J. D. Rhodes, C. King, G. Gulen, S. M. Olmstead, J. S. Dyer, R. E. Hebner, F. C. Beach, T. F. Edgarb and M. E. Webber, "A geographically resolved method to estimate levelized power plant costs with environmental externalities," *Energy Policy,* vol. 102, pp. 491-499, 2017.

[15] J.-F. Forero-Quintero, R. Villafáfila-Robles, S. Barja-Martinez, I. Munné-Collado, P. Olivella-Rosell and D. Montesinos-Miracle, "Profitability analysis on demand-side flexibility: A review," *Renewable and Sustainable Energy Reviews,* vol. 169, 2022.

[16] J. Monat, T. Gannon and M. Amissah, "The case for systems thinking in undergraduate engineering education," *International Journal of Engineering Pedagogy,* vol. 12, no. 3, 2022.

[17] M. M. Mekonnen, P. W. Gerbens-Leenes and A. Y. Hoekstra, "The consumptive water footprint of electricity and heat: a global assessment," *Environmental Science: Water Research & Technology,* 2015.

[18] Energy Information Administration, "Form EIA-923 detailed data with previous form data (EIA-906/920)," 2026. [Online]. Available: https://www.eia.gov/electricity/data/eia923/. [Accessed April 2026].