Market_4

An energy company is interested in investing in wind-power production. To assess the viability of this investment, the company wants to evaluate the expected revenue of a wind-power producer in a competitive spot market.

Suppose there are five energy producers participating in the day-ahead spot market. Three of them use conventional sources for energy production, such as coal, while the other two producers use solar and wind power, respectively. The market is cleared once per day for 24 hourly intervals of the following day. For each dispatch interval $t=1, \dots, 24$, every producer submits a bid (b_t, p_t) specifying the energy quantity b_t (MWh) it is willing to supply and the corresponding minimum acceptable price p_t (per MWh). Then, the market operator follows a merit-order process to determine the market-clearing price c_t and dispatch schedule, ensuring the total market demand D_t is met, where $D_t \sim \mathcal{N}(\mu_D, {\sigma_D}^2)$ with constant mean and variance. Merit-order ensures that any bid (b_t, p_t) at interval t, if $p_t \leq c_t$, the bid is accepted, and the producer is committed to delivering b_t at c_t. Otherwise, the producer earns zero revenue.

From the perspective of the wind-power producer, the bid (b_{it}, p_{it}) from a conventional producer i can be modeled as follows: b_{it} can be assumed to be stable, while $p_{it} \sim \mathcal{N}(\mu_{pi}, {\sigma_{pi}}^2)$ with deeply uncertain mean and variance since its bidding strategy is unknown. For the solar-power producer, its b_{st} is associated with the time t and assumed to be $b_{st} = max(0, a+b\cos(\frac{2\pi t}{24}))$, where a and b can be forecast based on historical data thereby assumed to be constants. Its p_{st} is also uncertain, where $p_{st} \sim \mathcal{N}(\mu_{ps}, {\sigma_{ps}}^2)$ with deeply uncertain mean and variance.

The actual energy production of the wind-power producer G_t is also uncertain, as it depends on weather conditions. The production G_t can be forecast from historical data and is modeled as $G_t \sim \mathcal{N}(\mu_G, {\sigma_G}^2)$ with constant mean and variance. In this case, the actual production may differ from the quantity committed to dispatch one day in advance. If the producer under-delivers relative to its dispatched quantity, the shortfall incurs a penalty of q_u per MWh. This cost is imposed by the market operator to maintain the grid's safe operational conditions and to ensure supply-demand balance in real time.

The energy company aims to build a simulation model to evaluate the expected revenue of the wind-power producer on any future day by selecting the hourly bid quantity and price (b_{wt}, p_{wt}), while accounting for uncertain generation and stochastic market clearing prices. This model can then be used to explore outcomes across various short-term and long-term scenarios, thereby informing the viability of the investment.

Please formalize this problem description as a simulation model and specify the key components, including state variables, decision variables, transition functions, objective functions, stochastic variables, constant parameters and deep uncertainties.

Great question—this is a classic **stochastic market-clearing + uncertain generation** problem, and it fits very naturally into a simulation (or stochastic control / Monte Carlo) framework. I'll formalize it cleanly and modularly so it's easy to implement and extend for scenario analysis.

---

1. Time Structure and Scope

- **Day-ahead market**, cleared once per day
- **Hourly intervals:**

$$t = 1, \ldots, 24$$

- Simulation horizon: one day (can be repeated independently for many simulated days)

---

2. Agents

- **Wind producer (decision-maker)**
- **Solar producer (strategic but exogenous)**
- **Three conventional producers (strategic but exogenous)**
- **Market operator (clears market via merit order)**

The wind producer is the only agent whose decisions we optimize.

---

3. State Variables

At each hour t, the system state consists of:

$$S_t = \big(D_t,\ \{(b_{it}, p_{it})\}_{i=1}^{3},\ (b_{st}, p_{st}),\ G_t\big)$$

Where:

- D_t: total system demand
- (b_{it}, p_{it}): bids from conventional producer i
- (b_{st}, p_{st}): solar producer bid
- G_t: realized wind generation

  In a day-ahead setting, these are **not observed** at decision time; they are realized after bids are submitted.

---

4. Decision Variables (Wind Producer)

For each hour t:

$$\boxed{(b_{wt}, p_{wt})}$$

Where:

- $b_{wt} \geq 0$: quantity bid (MWh)
- $p_{wt} \geq 0$: minimum acceptable price (€/MWh)

5. Stochastic Variables

Demand

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$$

Wind generation

$$G_t \sim \mathcal{N}(\mu_G, \sigma_G^2)$$

Conventional producers' bids

For $i = 1, 2, 3$:

- Quantity:

$$b_{it} = \bar{b}_i \quad \text{(constant)}$$

- Price:

$$p_{it} \sim \mathcal{N}(\mu_{pi}, \sigma_{pi}^2)$$

Solar producer bids

- Quantity:

$$b_{st} = \max\left(0, a + b\cos\left(\frac{2\pi t}{24}\right)\right)$$

- Price:

$$p_{st} \sim \mathcal{N}(\mu_{ps}, \sigma_{ps}^2)$$

6. Market-Clearing (Transition Function)

For each hour t:

Step 1: Collect bids

$$\mathcal{B}_t = \left\{(b_{it}, p_{it})_{i=1}^3,\ (b_{st}, p_{st}),\ (b_{wt}, p_{wt})\right\}$$

Step 2: Merit-order sorting

Sort bids in ascending order of price.

Step 3: Market clearing

- Accept bids until cumulative supply meets demand D_t
- Market-clearing price c_t is the price of the **marginal accepted bid**

Formally:

$$\sum_{j:p_{jt} \leq c_t} b_{jt} \geq D_t$$

Step 4: Wind dispatch

$$x_{wt} = \begin{cases} b_{wt}, & \text{if } p_{wt} \leq c_t \\ 0, & \text{otherwise} \end{cases}$$

7. Real-Time Imbalance and Penalties

Shortfall

$$u_t = \max(0, x_{wt} - G_t)$$

Penalty cost

$$\text{Penalty}_t = q_u \cdot u_t$$

8. Revenue Function (Per Hour)

$$R_t = \underbrace{c_t \cdot x_{wt}}_{\text{market revenue}} - \underbrace{q_u \cdot \max(0, x_{wt} - G_t)}_{\text{imbalance penalty}}$$

9. Objective Function

The wind producer maximizes **expected daily profit**:

$$\boxed{\max_{\{(b_{wt}, p_{wt})\}_{t=1}^{24}} \mathbb{E}\left[\sum_{t=1}^{24} R_t\right]}$$

Expectation is taken over:

- Demand D_t
- Market prices p_{it}, p_{st}

- Wind generation G_t

10. Constant Parameters

Parameter	Description
μ_D, σ_D	Mean and std. dev. of demand
μ_G, σ_G	Mean and std. dev. of wind generation
a, b	Solar production parameters
q_u	Penalty per MWh of under-delivery
$\bar{b}_i$	Conventional producers' capacities

11. Deep Uncertainties

These parameters are **not known with confidence** and should be treated as scenario inputs:

- μ_{pi}, σ_{pi}: conventional producers' bidding strategies
- μ_{ps}, σ_{ps}: solar producer's bidding strategy
- Possibly long-term shifts in:
  - μ_D (demand growth)
  - μ_G (climate effects on wind)

These can be explored via:

- Robust Monte Carlo simulation
- Scenario trees
- Stress testing and worst-case analysis

12. Simulation Workflow (High-Level)

1. Sample stochastic variables for all t
2. Clear the market hour by hour
3. Compute wind dispatch, penalties, and revenues
4. Aggregate daily profit
5. Repeat over many simulated days
6. Compare bidding strategies and scenarios

13. Output Metrics

- Expected daily revenue
- Revenue variance / downside risk
- Probability of under-delivery
- Sensitivity to deep uncertainties

If you want, next we can:

- Derive **closed-form acceptance probabilities** for wind bids
- Turn this into a **Monte Carlo pseudocode algorithm**
- Extend it to **risk-averse objectives** (e.g., CVaR)
- Or formulate it as a **stochastic optimization problem**

Just tell me where you want to go next 🌬️⚡

You do not need to consider deep uncertainties arising from long-term shifts in μ_D or μ_G. You can assume that they are constant over time. Please refine your specification.

Perfect — that simplifies the model nicely and keeps it firmly in the **short-term operational / market-design** regime. Below is a **refined and tighter simulation specification**, with **long-term structural uncertainty removed** and only **short-term stochasticity and strategic uncertainty** retained.

---

1. Time Structure

- Single **day-ahead market**
- Discrete time:

$$t = 1, \dots, 24$$

- Each day is simulated independently (i.i.d. across days)

---

2. Agents

- Wind producer (**decision-maker**)
- Solar producer (exogenous)
- Three conventional producers (exogenous)
- Market operator (merit-order clearing)

---

3. State Variables

For each hour t, define the (random) system state:

$$S_t = \left(D_t,\ \{(b_{it}, p_{it})\}_{i=1}^{3},\ (b_{st}, p_{st}),\ G_t\right)$$

These variables are **realized after bids are submitted** and are not observed at decision time.

---

4. Decision Variables (Wind Producer)

For each hour t:

$$\boxed{(b_{wt}, p_{wt})}$$

Subject to:

$$b_{wt} \geq 0, \quad p_{wt} \geq 0$$

5. Stochastic Variables (Short-Term Only)

Demand

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2), \quad \text{i.i.d. across } t$$

Wind generation

$$G_t \sim \mathcal{N}(\mu_G, \sigma_G^2), \quad \text{i.i.d. across } t$$

Conventional producers' bids

For each $i = 1, 2, 3$:

- Quantity (deterministic):

$$b_{it} = \bar{b}_i$$

- Price:

$$p_{it} \sim \mathcal{N}(\mu_{pi}, \sigma_{pi}^2)$$

Solar producer bids

- Quantity:

$$b_{st} = \max\left(0,\ a + b\cos\left(\frac{2\pi t}{24}\right)\right)$$

- Price:

$$p_{st} \sim \mathcal{N}(\mu_{ps}, \sigma_{ps}^2)$$

---

6. Market-Clearing Mechanism (Transition Function)

For each hour t:

Step 1: Bid aggregation

$$\mathcal{B}_t = \{(b_{wt}, p_{wt}), (b_{st}, p_{st}), (b_{1t}, p_{1t}), (b_{2t}, p_{2t}), (b_{3t}, p_{3t})\}$$

Step 2: Merit-order sorting

Sort bids in ascending order of price.

Step 3: Market clearing

Let cumulative supply satisfy:

$$\sum_{j:p_{jt} \le c_t} b_{jt} \ge D_t$$

- c_t: market-clearing price (price of marginal accepted bid)

Step 4: Wind dispatch

$$x_{wt} = \begin{cases} b_{wt}, & \text{if } p_{wt} \le c_t \\ 0, & \text{otherwise} \end{cases}$$

7. Real-Time Imbalance and Penalties

Wind production is realized after dispatch.

Shortfall

$$u_t = \max(0, x_{wt} - G_t)$$

Penalty

$$\text{Penalty}_t = q_u \cdot u_t$$

No compensation is assumed for overproduction.

8. Hourly Revenue Function

$$R_t = c_t \cdot x_{wt} \; - \; q_u \cdot \max(0, x_{wt} - G_t)$$

9. Objective Function

The wind producer chooses bids to maximize **expected daily revenue**:

$$\boxed{\max_{\{(b_{wt}, p_{wt})\}_{t=1}^{24}} \mathbb{E}\left[\sum_{t=1}^{24} (c_t x_{wt} - q_u \max(0, x_{wt} - G_t))\right]}$$

Expectation is taken over:

- D_t
- p_{it} for $i = 1, 2, 3$
- p_{st}
- G_t

10. Constant Parameters

Parameter	Description
μ_D, σ_D	Mean and variance of hourly demand
μ_G, σ_G	Mean and variance of wind generation
a, b	Solar output profile parameters
q_u	Under-delivery penalty
$\bar{b}_i$	Conventional producer capacities

All constants are **time-invariant**.

---

11. Remaining Deep Uncertainties (Strategic Only)

The only deep uncertainties retained are **strategic bidding behaviors** of other producers:

- (μ_{pi}, σ_{pi}), $i = 1, 2, 3$: conventional producers' price bids
- (μ_{ps}, σ_{ps}): solar producer's price bids

These parameters define **scenario families** rather than stochastic noise.

---

12. Simulation Algorithm (Refined)

For each simulated day:

1. For each hour t, sample:
   - D_t
   - p_{it}, $i = 1, 2, 3$
   - p_{st}
   - G_t
2. Clear market via merit order
3. Compute c_t and x_{wt}
4. Compute imbalance penalties and revenue
5. Aggregate total daily revenue
6. Repeat for many Monte Carlo runs

---

13. Key Outputs

- Expected daily revenue
- Revenue distribution (variance, downside risk)

- Acceptance probability of wind bids
- Expected imbalance costs

This refined model is now **fully consistent**, **tractable**, and **ready for Monte Carlo implementation or stochastic optimization**, without mixing short-term uncertainty and long-term structural change.

If you want, next we can:

- Reduce this to a **single-hour analytical model**
- Add **risk aversion (CVaR)**
- Or derive **closed-form expressions** for expected penalties and acceptance probabilities

Various stakeholders approach this problem from different perspectives. Please refine the model specification from the perspectives of (1) the wind-power producer; and (2) the system regulator. These two perspectives should share a common environment, with each stakeholder independently controlling its own decision variables. In the specification, specify the common environment, including the global parameters and variables. For each perspective, also specify the corresponding model and its key components, including the decision variables, transition functions, and objective functions. If there are multiple objectives, provide multiple objective functions instead of a weighted objective function.

Great framing — this is exactly how these problems are treated in **market design and policy analysis**. Below is a **two-stakeholder model specification** with a **shared stochastic environment** and **separate control and objectives**, written in a way that is simulation-ready and policy-relevant.

I. Common Environment (Shared by All Stakeholders)

This defines the **physical system, market rules, and uncertainty structure**. Neither stakeholder controls these elements directly.

---

1. Time Structure

- Day-ahead market
- Hourly intervals:

$$t = 1, \dots, 24$$

- Each simulated day is independent

---

2. Market Structure

- Uniform-price day-ahead market
- Merit-order dispatch
- Single-node (no network constraints)

---

3. Global Random Variables

For each hour t:

Demand

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$$

Wind generation

$$G_t \sim \mathcal{N}(\mu_G, \sigma_G^2)$$

Conventional producers ($i = 1, 2, 3$)

- Capacity:

$$b_{it} = \bar{b}_i$$

- Bid price:

$$p_{it} \sim \mathcal{N}(\mu_{pi}, \sigma_{pi}^2)$$

Solar producer

- Quantity:

$$b_{st} = \max\left(0,\ a + b\cos\left(\frac{2\pi t}{24}\right)\right)$$

- Bid price:

$$p_{st} \sim \mathcal{N}(\mu_{ps}, \sigma_{ps}^2)$$

4. Global Constants

Parameter	Meaning
μ_D, σ_D	Demand distribution
μ_G, σ_G	Wind generation distribution
a, b	Solar production parameters
$\bar{b}_i$	Conventional capacities
q_u	Imbalance penalty
Market rules	Uniform price, merit order

5. Market-Clearing Transition Function (Environment Dynamics)

For each hour t:

1. Collect bids:

$$\mathcal{B}_t = \{(b_{jt}, p_{jt})\}_{j \in \{w,s,1,2,3\}}$$

2. Sort by ascending price

3. Find clearing price c_t such that:

$$\sum_{j:p_{jt} \leq c_t} b_{jt} \geq D_t$$

4. Dispatch:

$$x_{jt} = \begin{cases} b_{jt}, & p_{jt} \leq c_t \\ 0, & \text{otherwise} \end{cases}$$

This transition function is **shared and fixed**.

II. Perspective 1: Wind-Power Producer

The wind producer is a **profit-maximizing agent** operating within the shared environment.

---

1. Decision Variables

For each hour t:

$$\boxed{(b_{wt}, p_{wt})}$$

where:

$$b_{wt} \geq 0, \quad p_{wt} \geq 0$$

---

2. State Variables (Observed at Decision Time)

- Distributional knowledge of:
  - D_t
  - p_{it}, p_{st}
  - G_t
- Market rules and penalty q_u

No real-time realizations are observed ex ante.

---

3. Wind-Specific Transition Components

After market clearing and realization of wind output:

Shortfall

$$u_t = \max(0, x_{wt} - G_t)$$

Hourly revenue

$$R_t^{(w)} = c_t x_{wt} - q_u u_t$$

---

4. Objective Functions (Multiple)

The wind producer may pursue **multiple objectives**:

(W1) Expected profit maximization

$$\max \quad \mathbb{E}\left[\sum_{t=1}^{24} R_t^{(w)}\right]$$

(W2) Downside risk minimization

$$\min \quad \mathrm{Var}\left(\sum_{t=1}^{24} R_t^{(w)}\right)$$

(W3) Reliability performance

$$\min \quad \mathbb{E}\left[\sum_{t=1}^{24} u_t\right]$$

These objectives can be evaluated **separately** via Pareto analysis.

III. Perspective 2: System Regulator

The regulator does **not bid in the market** but **designs or enforces market rules** affecting system-level outcomes.

---

1. Decision Variables (Policy Controls)

The regulator controls:

$$\boxed{q_u}$$

(Optionally extended to time-varying penalties $q_{u,t}$.)

---

2. State Variables

The regulator observes (ex post):

- Aggregate supply–demand balance

- Total system imbalance
- Market prices and dispatch
- Generator revenues

---

3. Regulator-Relevant Transition Components

System-wide imbalance

$$U_t = \sum_j \max(0, x_{jt} - G_{jt})$$

(Only wind contributes in this simplified model.)

Market price

$$c_t$$

---

4. Objective Functions (Multiple)

The regulator balances **efficiency, reliability, and market fairness**.

(R1) System reliability

$$\min \quad \mathbb{E}\left[\sum_{t=1}^{24} U_t\right]$$

(R2) Price stability

$$\min \quad \mathrm{Var}(c_t)$$

(R3) Renewable participation

$$\max \quad \mathbb{E}\left[\sum_{t=1}^{24} x_{wt}\right]$$

(R4) Total system cost

$$\min \quad \mathbb{E}\left[\sum_{t=1}^{24}\left(\sum_{j\in\mathrm{conv}} c_t x_{jt} + q_u U_t\right)\right]$$

IV. Interaction Structure

- The **environment** maps decisions → outcomes via market clearing
- The **wind producer** responds to q_u and price uncertainty
- The **regulator** anticipates wind bidding behavior when setting q_u

This naturally forms a **bi-level or game-theoretic simulation**:

- Upper level: regulator policy selection
- Lower level: producer bidding response
- Evaluation via Monte Carlo simulation

V. Summary Diagram (Conceptual)

```mathematica
Regulator (q_u)
       ↓
Market Rules + Environment
       ↓
Wind Producer (b_wt, p_wt)
       ↓
Market Clearing → Prices → Dispatch
       ↓
Real-Time Generation → Imbalances → Revenues
```

If you want, I can:

- Reformulate this as a **bi-level optimization problem**
- Show how to implement **policy stress testing**
- Add **carbon pricing** or **capacity payments**
- Extend the regulator's role to **market design variables**

Just say the word ⚡

OpenAI

According to the specification, q_u is a decision variable controlled by the system regulator and therefore should not be treated as a global constant. Please refine your specification.

Good catch — you're absolutely right. If q_u is controlled by the regulator, it **cannot** live in the global constants. Below is a **clean, internally consistent refinement** that strictly separates:

- **what is fixed by the environment**, and
- **what is chosen by each stakeholder**.

I'll keep the structure tight and simulation-ready.

I. Common Environment (Shared, Exogenous)

The environment defines **physics, market rules, and stochastic processes**, but **no stakeholder-controlled variables**.

---

1. Time Structure

- Day-ahead market
- Hours:

$$t = 1, \dots, 24$$

---

2. Market-Clearing Rules (Fixed)

- Uniform-price auction
- Merit-order dispatch
- Single-node system

---

3. Global Stochastic Variables

For each hour t:

Demand

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$$

Wind generation

$$G_t \sim \mathcal{N}(\mu_G, \sigma_G^2)$$

Conventional producers ($i = 1, 2, 3$)

- Quantity:

$$b_{it} = \bar{b}_i$$

- Price:

$$p_{it} \sim \mathcal{N}(\mu_{pi}, \sigma_{pi}^2)$$

Solar producer

- Quantity:

$$b_{st} = \max\left(0,\ a + b\cos\left(\frac{2\pi t}{24}\right)\right)$$

- Price:

$$p_{st} \sim \mathcal{N}(\mu_{ps}, \sigma_{ps}^2)$$

4. Global Parameters (Fixed)

Parameter	Description
μ_D, σ_D	Demand distribution
μ_G, σ_G	Wind generation distribution
a, b	Solar profile
$\bar{b}_i$	Conventional capacities

⚠️ **Note:** No penalty parameters appear here.

5. Environment Transition Function (Market Clearing)

For each hour t:

1. Aggregate bids
2. Sort by price
3. Determine clearing price c_t
4. Assign dispatch x_{jt}

Formally:

$$x_{jt} = \begin{cases} b_{jt}, & p_{jt} \le c_t \\ 0, & \text{otherwise} \end{cases}$$

II. Stakeholder 1: Wind-Power Producer

1. Decision Variables

For each hour t:

$$\boxed{(b_{wt}, p_{wt})}$$

2. Wind-Specific State Components

- Beliefs over:
  - D_t
  - p_{it}, p_{st}
  - G_t
- **Observed penalty parameter** q_u (set by regulator)

3. Wind-Specific Transition Functions

Real-time shortfall

$$u_t = \max(0, x_{wt} - G_t)$$

Hourly revenue

$$R_t^{(w)} = c_t x_{wt} - q_u u_t$$

4. Objective Functions

(W1) Expected profit

$$\max \quad \mathbb{E}\left[\sum_{t=1}^{24} R_t^{(w)}\right]$$

(W2) Imbalance exposure

$$\min \quad \mathbb{E}\left[\sum_{t=1}^{24} u_t\right]$$

(W3) Revenue volatility

$$\min \quad \mathrm{Var}\left(\sum_{t=1}^{24} R_t^{(w)}\right)$$

III. Stakeholder 2: System Regulator

The regulator **does not participate in bidding** but **sets reliability incentives**.

1. Decision Variables

$$\boxed{q_u \in \mathbb{R}_+}$$

(Optionally extended to $q_{u,t}$.)

2. Regulator State Variables

Observed ex post:

- Dispatch outcomes x_{jt}
- Market prices c_t
- Realized imbalances u_t

3. Regulator-Relevant Transition Components

System imbalance

$$U_t = u_t$$

(Only wind contributes here.)

4. Objective Functions

(R1) System reliability

$$\min \quad \mathbb{E}\left[\sum_{t=1}^{24} U_t\right]$$

(R2) Market efficiency

$$\min \quad \mathrm{Var}(c_t)$$

(R3) Renewable participation

$$\max \quad \mathbb{E}\left[\sum_{t=1}^{24} x_{wt}\right]$$

(R4) Penalty revenue neutrality (optional)

$$\min \quad \left|\mathbb{E}\left[\sum_{t=1}^{24} q_u U_t\right] - \bar{C}\right|$$

where $\bar{C}$ is a policy target (e.g. zero net penalty transfers).

IV. Interaction Structure (Corrected)

- The **regulator** sets q_u
- The **wind producer** observes q_u and optimizes bids
- The **environment** clears the market
- Outcomes feed back into regulator objectives

This defines a **leader–follower (Stackelberg) structure**, but is equally valid for **simulation-based policy evaluation**.

V. Summary Table (Corrected Roles)

Element	Role
μ_D, μ_G	Global constants
D_t, G_t	Stochastic environment
b_{wt}, p_{wt}	Wind producer decisions
q_u	Regulator decision
c_t, x_{jt}	Environment outcomes

This version is now **fully consistent** with control theory, market design, and simulation practice.

If you'd like, next we can:

- Formalize this as a **bi-level optimization**
- Add **bounds or convexity conditions on** q_u

- Introduce **alternative imbalance settlement schemes**

Just say where you want to go next.

To ensure internal consistency across these models, please compose them into a single, unified model that shares a common environment and states, while embedding perspective-specific decisions and objectives. The composed model should: 1. embed each perspective's decision variables and objectives; 2. enable modular evaluation of each perspective; 3. for each perspective, treat other perspectives' decisions as exogenous inputs; and 4. incorporate all other components from every perspective.

Below is a **single, unified composed model** that keeps **one common environment and state space**, while **embedding multiple stakeholder perspectives as modular "views"** with their own decision variables and objectives.
The key idea is: **one stochastic dynamical system**, with **role-specific control inputs and payoff functionals**. Each perspective can be evaluated independently by treating the others' decisions as **exogenous parameters**.

Unified Composed Model

1. Time and Indexing

- Day-ahead market, one day
- Hours:

$$t = 1, \dots, 24$$

---

2. Common Environment

The environment defines **what happens**, independent of *who evaluates it*.

2.1 Global State Variables

For each hour t, define the realized system state:

$$S_t = \Big(D_t,\ G_t,\ \{(b_{it}, p_{it})\}_{i=1}^{3},\ (b_{st}, p_{st}),\ (b_{wt}, p_{wt}),\ c_t,\ x_{wt}\Big)$$

Where:

- D_t: demand
- G_t: wind generation
- (b_{it}, p_{it}): conventional bids
- (b_{st}, p_{st}): solar bids
- (b_{wt}, p_{wt}): wind bid (decision input)

- c_t: clearing price
- x_{wt}: wind dispatch

2.2 Stochastic Variables (Exogenous Processes)

For each t:

$$\begin{aligned} D_t &\sim \mathcal{N}(\mu_D, \sigma_D^2), \\ G_t &\sim \mathcal{N}(\mu_G, \sigma_G^2), \\ p_{it} &\sim \mathcal{N}(\mu_{pi}, \sigma_{pi}^2), \quad i = 1, 2, 3, \\ p_{st} &\sim \mathcal{N}(\mu_{ps}, \sigma_{ps}^2) \end{aligned}$$

Deterministic quantities:

$$\begin{aligned} b_{it} &= \bar{b}_i, \\ b_{st} &= \max\left(0, a + b\cos\frac{2\pi t}{24}\right) \end{aligned}$$

2.3 Fixed Environment Parameters

$$\Theta = \{\mu_D, \sigma_D, \mu_G, \sigma_G, a, b, \bar{b}_i\}$$

These are **not controlled by any stakeholder**.

2.4 Environment Transition Function (Market Clearing)

Given:

- bids $\{(b_{jt}, p_{jt})\}$
- demand D_t

the environment deterministically computes:

$$(c_t, x_{jt}) = \mathcal{M}(D_t, \{(b_{jt}, p_{jt})\})$$

with:

$$x_{jt} = \begin{cases} b_{jt}, & p_{jt} \le c_t, \\ 0, & \text{otherwise.} \end{cases}$$

2.5 Imbalance Mapping

Wind under-delivery:

$$u_t = \max(0, x_{wt} - G_t)$$

---

3. Stakeholder Perspectives (Embedded Modules)

Each perspective is defined by:

- its **decision variables**
- its **objective functionals**
- its **evaluation mapping**
  while treating all other decisions as **exogenous inputs** to the environment.

4. Perspective A: Wind-Power Producer

4.1 Decision Variables

$$\boxed{U^{(w)} = \{(b_{wt}, p_{wt})\}_{t=1}^{24}}$$

4.2 Exogenous Inputs

- Regulator's penalty choice q_u
- Environment parameters Θ
- Stochastic processes $(D_t, G_t, p_{it}, p_{st})$

4.3 Wind Revenue Functional

Hourly payoff:

$$R_t^{(w)} = c_t x_{wt} - q_u u_t$$

4.4 Wind Objectives

(W1) **Expected profit**

$$J_1^{(w)}(U^{(w)}; q_u) = \mathbb{E}\left[\sum_{t=1}^{24} R_t^{(w)}\right]$$

(W2) **Expected imbalance**

$$J_2^{(w)}(U^{(w)}; q_u) = \mathbb{E}\left[\sum_{t=1}^{24} u_t\right]$$

(W3) **Revenue volatility**

$$J_3^{(w)}(U^{(w)}; q_u) = \mathrm{Var}\left(\sum_{t=1}^{24} R_t^{(w)}\right)$$

5. Perspective B: System Regulator

5.1 Decision Variable

$$\boxed{U^{(r)} = q_u \in \mathbb{R}_+}$$

5.2 Exogenous Inputs

- Wind bidding strategy $U^{(w)}$
- Environment parameters Θ
- Stochastic processes

5.3 System-Level Quantities

System imbalance:

$$U_t = u_t$$

Penalty revenue:

$$\Pi_t = q_u u_t$$

5.4 Regulator Objectives

(R1) **Reliability**

$$J_1^{(r)}(q_u; U^{(w)}) = \mathbb{E}\left[\sum_{t=1}^{24} U_t\right]$$

(R2) **Price stability**

$$J_2^{(r)}(q_u; U^{(w)}) = \mathrm{Var}(c_t)$$

(R3) **Renewable utilization**

$$J_3^{(r)}(q_u; U^{(w)}) = \mathbb{E}\left[\sum_{t=1}^{24} x_{wt}\right]$$

(R4) **Penalty magnitude**

$$J_4^{(r)}(q_u; U^{(w)}) = \mathbb{E}\left[\sum_{t=1}^{24} \Pi_t\right]$$

---

6. Composition Structure

The **composed model** can be written as:

$$\boxed{S_{1:24} = \mathcal{E}\big(\Theta,\ U^{(w)},\ U^{(r)},\ \omega\big)}$$

where:

- $\mathcal{E}$: shared environment simulator
- ω: realization of stochastic processes

Each perspective evaluates:

$$J^{(k)} = J^{(k)}(S_{1:24})$$

---

7. Modularity and Consistency

✔️ One environment
✔️ One state trajectory
✔️ Perspective-specific controls
✔️ Perspective-specific objectives
✔️ Other decisions treated as exogenous
✔️ Suitable for Monte Carlo, Stackelberg, or equilibrium analysis

---

8. Interpretation

- **Wind producer problem**: optimize $U^{(w)}$ given q_u
- **Regulator problem**: evaluate or optimize q_u anticipating $U^{(w)}$
- **Simulation problem**: sample ω, generate $S_{1:24}$, compute all $J^{(k)}$

If you'd like, the next natural steps are:

- writing this as a **bi-level program**
- adding **bounds or convexity assumptions**
- extending to **multiple wind producers**
- or embedding this into a **robust policy discovery loop**

Just tell me how far you want to push it.

Now, provide a modular Python implementation of the unified model, using classes to represent different perspectives. At the end of the program, define a function as the interface. This function should take as inputs a dictionary of decision variable values, a dictionary of uncertain parameter values, and a dictionary of constant parameter values; simulate the model dynamics; and return the value of each objective function. This function does not need to be executed.

Below is a **clean, modular Python implementation** of the unified composed model.
It uses **classes for the common environment and each perspective**, and finishes with a **single interface function** that wires everything together exactly as requested.

No execution is required; this is a **structural, simulation-ready specification**.

```python
import numpy as np
from typing import Dict, Any


# ============================================================
# Common Environment
# ============================================================

class MarketEnvironment:
    """
    Shared stochastic environment and market-clearing mechanism.
    """

    def __init__(self, constant_params: Dict[str, Any], uncertain_params: Dict[str, Any]):
        self.const = constant_params
        self.uncertain = uncertain_params

    def sample_exogenous_processes(self, t: int) -> Dict[str, Any]:
        """
        Sample stochastic variables for hour t.
        """
        D_t = np.random.normal(
            self.const["mu_D"], self.const["sigma_D"]
        )
        G_t = np.random.normal(
            self.const["mu_G"], self.const["sigma_G"]
        )

        p_conv = [
            np.random.normal(mu, sigma)
            for mu, sigma in self.uncertain["conv_price_params"]
        ]

        p_solar = np.random.normal(
            self.uncertain["solar_price_mu"],
            self.uncertain["solar_price_sigma"],
        )

        b_solar = max(
            0.0,
            self.const["solar_a"]
            + self.const["solar_b"] * np.cos(2 * np.pi * t / 24),
        )

        return {
            "D_t": D_t,
            "G_t": G_t,
```

```
            "p_conv": p_conv,
            "p_solar": p_solar,
            "b_solar": b_solar,
        }

    def clear_market(self, bids: Dict[str, Any], D_t: float):
        """
        Merit-order clearing.
        """
        # bids: list of (quantity, price, owner)
        bids_sorted = sorted(bids, key=lambda x: x[1])

        supplied = 0.0
        clearing_price = 0.0
        dispatch = {}

        for q, p, name in bids_sorted:
            if supplied < D_t:
                supplied += q
                clearing_price = p
                dispatch[name] = q
            else:
                dispatch[name] = 0.0

        return clearing_price, dispatch


# ============================================================
# Perspective: Wind Producer
# ============================================================

class WindProducer:
    def __init__(self, decision_vars: Dict[str, Any]):
        self.bids = decision_vars["wind_bids"]  # list of (b_wt, p_wt)

    def get_bid(self, t: int):
        return self.bids[t]


# ============================================================
# Perspective: Regulator
# ============================================================

class SystemRegulator:
    def __init__(self, decision_vars: Dict[str, Any]):
        self.q_u = decision_vars["q_u"]


# ============================================================
# Unified Simulator
# ============================================================

class UnifiedSimulator:
    def __init__(
        self,
        environment: MarketEnvironment,
        wind: WindProducer,
        regulator: SystemRegulator,
    ):
        self.env = environment
        self.wind = wind
        self.regulator = regulator

        # Storage for outcomes
        self.results = {
            "wind_profit": 0.0,
            "wind_imbalance": 0.0,
            "prices": [],
            "wind_dispatch": [],
            "penalty_revenue": 0.0,
        }

    def run(self):
        for t in range(24):
            exo = self.env.sample_exogenous_processes(t)
```

```
            # Build bids
            bids = []

            # Conventional producers
            for i, b_i in enumerate(self.env.const["conv_capacities"]):
                bids.append((b_i, exo["p_conv"][i], f"conv_{i}"))

            # Solar producer
            bids.append((exo["b_solar"], exo["p_solar"], "solar"))

            # Wind producer
            b_wt, p_wt = self.wind.get_bid(t)
            bids.append((b_wt, p_wt, "wind"))

            # Market clearing
            c_t, dispatch = self.env.clear_market(bids, exo["D_t"])
            x_wt = dispatch.get("wind", 0.0)

            # Real-time imbalance
            u_t = max(0.0, x_wt - exo["G_t"])
            penalty = self.regulator.q_u * u_t

            # Store results
            self.results["wind_profit"] += c_t * x_wt - penalty
            self.results["wind_imbalance"] += u_t
            self.results["penalty_revenue"] += penalty
            self.results["prices"].append(c_t)
            self.results["wind_dispatch"].append(x_wt)

        return self.results


# ============================================================
# Interface Function
# ============================================================

def simulate_unified_model(
    decision_vars: Dict[str, Any],
    uncertain_params: Dict[str, Any],
    constant_params: Dict[str, Any],
) -> Dict[str, float]:
    """
    Interface function required by the specification.

    Inputs
    ------
    decision_vars:
        {
            "wind_bids": [(b_w1, p_w1), ..., (b_w24, p_w24)],
            "q_u": penalty_per_MWh
        }

    uncertain_params:
        {
            "conv_price_params": [(mu_p1, sigma_p1), (mu_p2, sigma_p2), (mu_p3, sigma_p3)],
            "solar_price_mu": float,
            "solar_price_sigma": float
        }

    constant_params:
        {
            "mu_D", "sigma_D",
            "mu_G", "sigma_G",
            "solar_a", "solar_b",
            "conv_capacities": [b1, b2, b3]
        }

    Returns
    -------
    Dictionary mapping objective names to values.
    """

    env = MarketEnvironment(constant_params, uncertain_params)
    wind = WindProducer(decision_vars)
```

```
    regulator = SystemRegulator(decision_vars)

    simulator = UnifiedSimulator(env, wind, regulator)
    results = simulator.run()

    objectives = {
        # Wind producer objectives
        "wind_expected_profit": results["wind_profit"],
        "wind_total_imbalance": results["wind_imbalance"],
        "wind_profit_variance_proxy": np.var(results["prices"]),

        # Regulator objectives
        "system_price_variance": np.var(results["prices"]),
        "renewable_utilization": np.sum(results["wind_dispatch"]),
        "total_penalty_revenue": results["penalty_revenue"],
    }

    return objectives
```

Why this satisfies the specification

✔️ **Single unified environment**
✔️ **Perspective-specific classes**
✔️ **Each perspective controls only its own decisions**
✔️ **Other decisions treated as exogenous inputs**
✔️ **Multiple objectives returned explicitly (no weighting)**
✔️ **Clean interface function for scenario analysis**

If you want next steps, I can:

- convert this into a **bi-level optimization wrapper**
- add **multiple wind producers**
- integrate **CVaR or chance constraints**
- or refactor this for **parallel Monte Carlo evaluation**

Just say the word.

Please refine your implementation so that any bid with a price less than or equal to the clearing price is accepted, even if the total market demand has already been met.

Good point — that's an important **market-design detail**, and you're right to enforce it for internal consistency.

Below is a **refined implementation** where:

- **All bids with price ≤ clearing price are accepted**,
- Even if cumulative supply exceeds demand,
- The **clearing price is still determined by the marginal bid** that first meets demand.

Only the **market-clearing logic** needs to change; the rest of the model remains valid.

Key Conceptual Change

1. **Step 1:** Sort bids by price
2. **Step 2:** Identify the *marginal clearing price* c_t as the lowest price such that cumulative supply ≥ demand
3. **Step 3:** Accept **all bids** with price $\leq c_t$, regardless of oversupply

This corresponds to the specification:

> *Any bid* (b_t, p_t)*, if* $p_t \leq c_t$*, is accepted.*

Refined Python Implementation (Minimal, Targeted Changes)

Only the `clear_market` method is modified.

✅ Updated `clear_market` Method

```python
class MarketEnvironment:
    """
    Shared stochastic environment and market-clearing mechanism.
    """

    def __init__(self, constant_params, uncertain_params):
        self.const = constant_params
        self.uncertain = uncertain_params

    def sample_exogenous_processes(self, t: int):
        D_t = np.random.normal(
            self.const["mu_D"], self.const["sigma_D"]
        )
```

```
        G_t = np.random.normal(
            self.const["mu_G"], self.const["sigma_G"]
        )

        p_conv = [
            np.random.normal(mu, sigma)
            for mu, sigma in self.uncertain["conv_price_params"]
        ]

        p_solar = np.random.normal(
            self.uncertain["solar_price_mu"],
            self.uncertain["solar_price_sigma"],
        )

        b_solar = max(
            0.0,
            self.const["solar_a"]
            + self.const["solar_b"] * np.cos(2 * np.pi * t / 24),
        )

        return {
            "D_t": D_t,
            "G_t": G_t,
            "p_conv": p_conv,
            "p_solar": p_solar,
            "b_solar": b_solar,
        }

    def clear_market(self, bids, D_t):
        """
        Merit-order clearing with full acceptance of all bids
        priced at or below the clearing price.
        """

        # bids: list of (quantity, price, name)
        bids_sorted = sorted(bids, key=lambda x: x[1])

        cumulative_supply = 0.0
        clearing_price = None

        # Step 1: find clearing price
        for q, p, _ in bids_sorted:
            cumulative_supply += q
            if cumulative_supply >= D_t:
                clearing_price = p
                break

        # If demand is never met, highest price clears
        if clearing_price is None:
            clearing_price = bids_sorted[-1][1]

        # Step 2: accept all bids with price <= clearing price
        dispatch = {}
        for q, p, name in bids_sorted:
            if p <= clearing_price:
                dispatch[name] = q
            else:
                dispatch[name] = 0.0

        return clearing_price, dispatch
```

Why This Is Now Correct

✔️ Clearing price determined by *marginal* bid
✔️ No partial acceptance
✔️ No rejection of cheap bids due to oversupply
✔️ Fully consistent with the formal market rule you specified
✔️ Leaves imbalance handling to the **real-time penalty mechanism**, not day-ahead dispatch

Interpretation in the Unified Model

- Oversupply is allowed **by design**
- Wind (and other producers) face **real-time penalties** if they cannot deliver
- This properly shifts reliability incentives to:
  - bidding behavior (wind producer), and
  - penalty design (regulator)

If you want, next we can:

- add **partial acceptance** as an alternative market design
- include **downward regulation costs**
- or compare **efficient vs. inefficient clearing rules** side by side

Just let me know.