inDepth

ASSESSING THE CAPABILITY OF AGENTIC AI IN A STOCHASTIC HYDROTHERMAL SCHEDULING CASE STUDY

Julio Alberto Silva Dias and Mario Veiga Ferraz Pereira

Abstract

Agentic artificial intelligence is emerging as a paradigm in which reasoning-capable models interact with external tools, simulations, and data sources to perform multi-step analytical tasks. For energy-system applications, this raises the question of whether AI agents can operate effectively within structured analytical environments using domain-specific capabilities to explore complex decision problems, rather than merely as wrappers around existing solvers.

This paper investigates this question through a stochastic hydrothermal scheduling case study, which combines uncertainty, intertemporal coupling, physical constraints, and economically meaningful operational trade-offs while offering reliable optimization benchmarks such as stochastic dual dynamic programming (SDDP). We propose a capability-driven agentic architecture in which a reasoning agent interacts with the hydrothermal model only through controlled, typed domain capabilities, without access to pre-programmed dynamic programming methods, precomputed water values, dual variables, or internal solver information.

The architecture is evaluated on a simplified but structurally realistic representation of the Brazilian Interconnected Power System. Across ten independent sessions, the agent inferred water-value logic from simulation feedback, constructed approximate future-cost representations, and deployed storage-preserving policies that substantially improved upon myopic behavior. Relative to an independently computed SDDP benchmark, the best session achieved a 1.7% mean cost gap, while the ten-session average gap was approximately 6.0%.

These results indicate that structured capability orchestration can enable reasoning agents to function as analytical layers embedded in energy-system modeling environments, most valuably for problems where analytical formulations are incomplete, difficult to specify, or insufficient to capture the full solution-exploration process.

Keywords: agentic AI, hydrothermal scheduling, stochastic optimization, water value, SDDP

Introduction

Agentic artificial intelligence is emerging as a paradigm in which reasoning-capable language models interact with external tools, simulations, data sources, and computational environments to perform multi-step analytical tasks. Instead of producing only static text outputs, these systems can select structured actions, invoke tools, interpret returned observations, revise hypotheses, and coordinate multi-step analytical workflows toward a user-defined objective [1]. Recent agentic frameworks commonly separate a language-model inference core, responsible for reasoning, planning, and action selection, from an orchestration layer, or Harness, that manages tool calls, observations, memory, and execution state across steps [4, 2, 3]. This general structure is illustrated in Fig. 1, which shows the LLM inference core and the surrounding Harness.

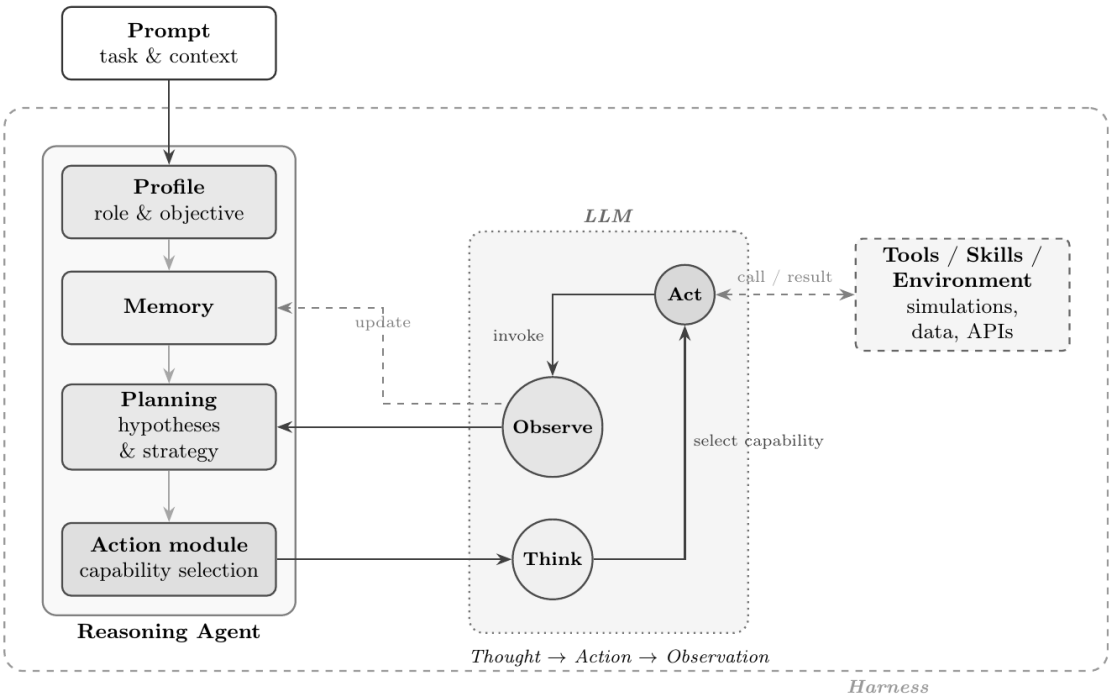

Figure 1: General structure of an LLM-based agentic reasoning loop.

Although these developments have been demonstrated primarily in generic enterprise, software, coding, and information-retrieval applications, they raise a relevant methodological question for energy-system analytics: can reasoning agents operate effectively inside structured analytical environments for complex engineering decision problems, or do they merely provide natural-language interfaces to existing computational tools? This question is particularly important in energy-system planning and operation, where decision-support processes rely on specialized models that represent uncertainty, physical constraints, intertemporal couplings, and economically meaningful trade-offs.

Hydrothermal scheduling provides a suitable setting for investigating this question. In hydrothermal systems, current reservoir-release decisions reduce present thermal generation costs but also affect future exposure to scarcity, expensive thermal dispatch, and energy deficits. Stored water therefore has an opportunity cost, commonly represented through water values or future-cost functions. The problem combines stochastic inflows, storage dynamics, transmission constraints, thermal substitution, and long-term operational trade-offs. At the same time, it has well-established stochastic optimization references, particularly Stochastic Dual Dynamic Programming (SDDP), which provides a rigorous benchmark for evaluating the quality of operating policies [5, 6].

Classical stochastic dynamic programming provides the conceptual basis for this sequential decision problem, but direct application is limited by the curse of dimensionality. In realistic hydrothermal systems, the state vector includes reservoir storage levels, hydrological conditions, and other intertemporal variables, causing the number of possible states to grow exponentially with system size and uncertainty representation. Decomposition methods such as SDDP avoid full state enumeration by approximating future-cost functions and remain the standard reference for large-scale hydrothermal operation planning. Machine learning methods, including reinforcement learning, have also been studied for sequential decision-making under uncertainty. In this paper, however, these methods serve primarily as conceptual references and benchmarks: the objective is not to propose a new hydrothermal optimization solver.

The central question addressed in this paper is instead architectural and methodological: can a reasoning agent, restricted to controlled domain capabilities, explore an energy-system analytical environment and

construct an operationally meaningful policy without access to a pre-programmed stochastic dynamic programming method, dual variables, precomputed water values, or solver internals? This question concerns how analytical functionality should be exposed to an AI agent so that reasoning, computation, control, and auditability remain separated. It is aligned with the emerging notion of domain-specific skills or capabilities: modular operations that expose selected functionality of an environment to an agent through structured interfaces.

To investigate this question, we propose a capability-driven agentic architecture for AI-assisted interaction with energy-system analytical environments. In this architecture, domain knowledge and model operations are exposed to the agent through structured and typed capabilities rather than through unrestricted code execution or direct access to solver internals. The agent can inspect the system, query scenario statistics, run simulations, evaluate candidate policies, and register policy-relevant artifacts, but all computations are executed by the controlled analytical environment. The agent therefore does not replace the domain model or the optimization solver; it acts as an analytical orchestration layer that formulates hypotheses, invokes capabilities, interprets outputs, and progressively constructs an approximate operating logic.

The proposed architecture is instantiated in a stochastic hydrothermal scheduling environment based on a simplified but structurally realistic representation of the Brazilian Interconnected Power System. The agent interacts with the model only through controlled capabilities and is not given access to an SDDP implementation, dynamic programming routines, dual information, precomputed water values, or internal solver state. Its resulting policy is then evaluated through stochastic simulation and compared with two references: a myopic policy that ignores future water value and an independently computed SDDP-based benchmark. This setting allows the agent's behavior to be assessed in a problem where the optimality structure is well understood and where reliable benchmarks are available.

The broader motivation is to evaluate whether reasoning agents can function as controlled analytical layers around energy-system models. Such agents are not expected to replace formal optimization methods. Rather, they may support exploration, interpretation, sensitivity analysis, model interrogation, and policy construction in settings where the relevant decision process extends beyond a single fully specified optimization formulation. Hydrothermal scheduling is used here as a benchmarked validation case before considering more open-ended energy-system problems in which analytical formulations may be incomplete, difficult to construct, or insufficient to capture the full solution-exploration process.

The paper addresses the following research questions:

RQ1. Can a reasoning agent use structured domain capabilities to construct an operationally meaningful policy in a stochastic energy-system scheduling problem?

RQ2. In the hydrothermal scheduling case, can the agent infer economically meaningful water-value logic from simulation feedback rather than from explicit dual information or precomputed optimization outputs?

RQ3. How reproducible is the policy-construction process across independent reasoning-agent sessions under the same task specification and capability interface?

RQ4. What architectural properties are required to support controlled, auditable, and modular evaluation of reasoning agents in energy-system analytical environments?

The main contributions of the paper are:

1. A capability-driven agentic architecture for AI-assisted interaction with energy-system analytical environments.
2. An instantiation of this architecture in a stochastic hydrothermal scheduling environment, connecting the agentic workflow to the classical multistage stochastic optimization formulation.

3. A structured capability interface that exposes controlled domain operations for model inspection, simulation-based exploration, policy registration, and stochastic evaluation while withholding solver internals, dual variables, and precomputed water values.
4. A benchmarked experimental assessment showing that a reasoning agent can construct storage-preserving hydrothermal operating policies through structured capability orchestration, with comparative evaluation against myopic operation and independently computed SDDP-based references.

The remainder of the paper is organized as follows. Section 2 reviews related work. Section 3 defines the multistage stochastic hydrothermal operation problem. Section 4 presents the capability-driven architecture, the analytical environment, and the experimental protocol. Section 5 presents the test case and results. Section 6 discusses implications and limitations. Section 7 concludes.

Related Work

Recent work on tool-using language models has shown that large language models can combine language-mediated reasoning with external computational actions, including tool calls, API access, and structured interaction with computational environments [1, 7, 8]. Related agentic architectures organize this process through modules for role specification, memory, planning, and action selection, and increasingly emphasize reusable skills or capabilities as the interface between the reasoning model and its environment [2, 3]. Industrial frameworks have followed a similar direction by providing reference patterns for combining foundation models, tool access, workflow orchestration, and guardrails [9, 10].

While many demonstrations of tool-using agents focus on web search, coding, question answering, or information retrieval, high-consequence engineering applications require stronger guarantees of control, auditability, and reproducibility. In such settings, unrestricted tool use is often inappropriate: the agent must interact with validated computational environments through well-defined interfaces. Interoperability protocols such as the Model Context Protocol (MCP) illustrate this direction by exposing external tools through typed interfaces [11]. This motivates the use of controlled capability interfaces for agentic interaction with energy-system models.

Machine learning has also been extensively applied in energy systems, including forecasting, optimal power flow approximation, and control [12, 13]. Reinforcement learning has been studied for reservoir operation and power-system control [14, 15]. These approaches usually learn policies or function approximations through data, simulation episodes, gradient updates, or explicit training procedures. The approach investigated here is different: the agent is not trained as a hydrothermal controller and does not learn a policy through repeated episodes. Instead, it uses reasoning and capability orchestration to inspect a structured energy-system environment, test hypotheses, and construct a policy-relevant approximation during an analytical session.

Hydrothermal operation planning has long been addressed through stochastic dynamic programming and decomposition. SDDP [5] represents the future-cost function through affine cuts and is widely used in large-scale hydrothermal systems. Subsequent work extended the method to convergence analysis, risk aversion, and intertemporal inflow models [6, 16, 17]. These methods provide the optimization benchmark for the present work and are used as external references, not as tools available to the agent.

Multistage Stochastic Hydrothermal Operation Problem

Consider a finite horizon $t = 1, \dots, T$. Let V_t be reservoir storage, H_t hydrological information, and $X_t = (V_t, H_t)$ the system state. Uncertainty is represented by ξ_t, including inflows and possibly demand or renewable generation. At each stage, decisions u_t include hydro generation, thermal generation, interconnection flows, spillage, deficit and next-stage storage. The stage problem is

$$\min_{u_t \in U_t(X_t, \xi_t)} c_t(u_t, \xi_t) + Q_{t+1}(X_{t+1}), \qquad (1)$$

where U_t contains water balance, generation, transmission and demand constraints.

For each hydro subsystem r, storage evolves as

$$V_{t+1,r} = V_{t,r} + a_{t,r}(\xi_t) - q_{t,r} - s_{t,r}, \qquad g^H_{t,r} = \eta_{t,r}\, q_{t,r}, \qquad (2)$$

with inflow $a_{t,r}$, turbined outflow $q_{t,r}$, spillage $s_{t,r}$ and production coefficient $\eta_{t,r}$. For each electrical area n,

$$\sum_{r \in R_n} g^H_{t,r} + \sum_{g \in G_n} g^T_{t,g} + R_{t,n}(\xi_t) + \sum_{\ell \in I_n} f_{t,\ell} - \sum_{\ell \in O_n} f_{t,\ell} + d_{t,n} = D_{t,n}(\xi_t). \qquad (3)$$

The feasible set also includes bounds on storage, hydro generation, thermal generation, transmission flows, spillage and deficits. The immediate cost is

$$c_t(u_t, \xi_t) = \sum_n \sum_{g \in G_n} C^T_{t,g}\; g^T_{t,g} + \sum_n C^D_{t,n}\; d_{t,n}. \qquad (4)$$

The multistage stochastic problem is

$$\min_{\pi} \mathbb{E}\left[\sum_{t=1}^{T} c_t\,(u_t, \xi_t) + \Phi(V_{T+1})\right], \qquad u_t = \pi_t(X_t, \xi_t), \qquad (5)$$

with nonanticipative policies. The Bellman recursion is

$$Q_t(X_t) = \mathbb{E}_{\xi_t | X_t}\left[\min_{u_t \in U_t(X_t, \xi_t)} \{c_t(u_t, \xi_t) + Q_{t+1}(X_{t+1})\}\right], \qquad (6)$$

with terminal condition $Q_{T+1}(X_{T+1}) = \Phi(V_{T+1})$. The marginal value of stored water is

$$\lambda_{t,r}(X_t) = \frac{\partial Q_t(X_t)}{\partial V_{t,r}}. \qquad (7)$$

Direct SDP discretization is intractable because R reservoirs discretized into K levels already produce K^R storage states. SDDP avoids full state enumeration by approximating future costs with affine cuts:

$$Q_t(V) \approx \max_{k \in K_t} \{\alpha^k_t + (\beta^k_t)^\top V\}. \qquad (8)$$

In this paper, SDDP is used as conceptual reference and external benchmark. The agent is not given access to an SDDP implementation.

Capability-Driven Architecture

Architectural principle

The proposed capability-driven architecture (CDA) separates the agent's reasoning process from the computational implementation of the underlying energy-system analytical model. The agent formulates hypotheses, selects analytical actions, interprets observations, and refines its strategy over a multi-step session, while the environment executes validated domain operations such as model inspection, simulation, policy evaluation, and artifact registration.

All interactions occur through typed capabilities with defined input schemas, output schemas, and execution semantics. The agent has no access to model files, solver internals, unrestricted code execution, dual variables, or precomputed optimization outputs. This separation allows the agent to operate as a reasoning and orchestration layer while preserving control, auditability, and a clear boundary between language-mediated inference and validated computation.

Formally, an analytical session under the CDA can be represented as an interaction trajectory

$$\tau = [(a_1, o_1), (a_2, o_2), \dots, (a_K, o_K)], \qquad (9)$$

where a_k denotes the capability invocation at step k and o_k the corresponding structured observation returned by the environment.

At each step, the agent selects the next capability invocation based on the full interaction history accumulated so far. Let $h_k = (\mathcal{T}, a_1, o_1, \dots, a_{k-1}, o_{k-1})$ denote that history, where $\mathcal{T}$ is the task specification provided at session initialization. The agent's reasoning loop can then be written as

$$a_k = \pi_A(h_k), \qquad (10)$$

where π_A is the agent's orchestration function: the mechanism by which the reasoning model interprets the accumulated history and selects the next capability invocation. This object should not be confused with an operational hydrothermal policy. It governs the sequencing of analytical actions during the agentic session, including inspection, simulation, hypothesis testing, and possible modification of policy-related artifacts such as future-cost approximations. Unlike a trained reinforcement learning controller, π_A is not learned through episodic interaction with the hydrothermal system; it is executed by the reasoning model through language-mediated inference over the session history.

The environment, in turn, maintains a state s_k comprising the loaded analytical model, the available capability definitions, and any stateful analytical artifacts created during the session. In the hydrothermal instantiation, the most important such artifact is the currently registered future-cost approximation.

Each capability invocation produces an observation and advances the environment state according to

$$(o_k, s_{k+1}) = \mathcal{E}(a_k, s_k), \qquad (11)$$

where $\mathcal{E}$ is a deterministic transition function. Read-only invocations (inspection, simulation, and scenario-statistics queries) leave s_k unchanged. Policy-encoding invocations update the registered cut set in s_{k+1}, making the environment stateful with respect to the policy construction process. The session terminates at step K when the agent declares the analytical task complete. The terminal environment state s_K contains the artifacts produced during the session. In the hydrothermal case, these artifacts include the registered future-cost approximation. This approximation subsequently induces an operational hydrothermal policy when embedded in the stage-dispatch problem and evaluated through stochastic simulation, as described in Section 5.

The trajectory τ is the primary object of post-hoc analysis: it records the full analytical path from system inspection to policy validation and enables reconstruction and auditability of the agent's reasoning process.

Figure 2 illustrates the overall component structure of the architecture, summarising the formal relationships described by equations (9)–(11).

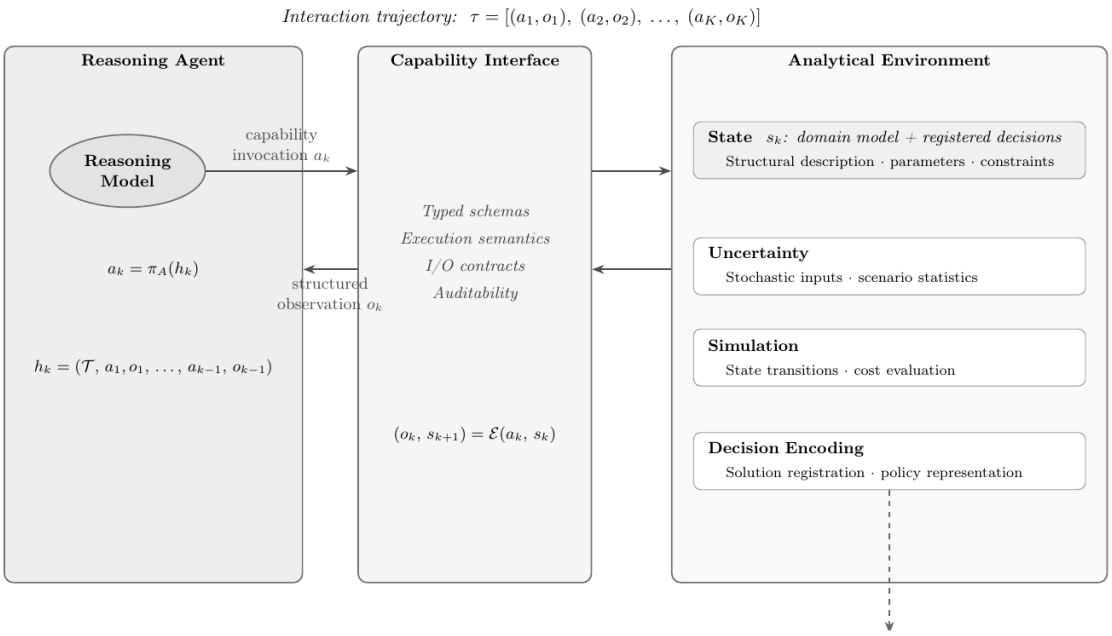

Figure 2: Capability-driven architecture.

Hydrothermal instantiation of the capability interface

The hydrothermal instantiation exposes domain operations to the agent through eight typed capabilities, organized into four analytical groups. The design follows a principle of deliberate information scoping: each capability provides the information required to support a well-defined analytical step while withholding solver internals, algorithmic templates, dual variables, and raw data streams that would allow the agent to bypass the reasoning task. This scoping is not merely a technical convenience; it is the mechanism by which the environment constrains the agent to perform domain-relevant reasoning rather than arbitrary computation or solver wrapping.

A further architectural property is session statefulness. The model is loaded once at session initialization, fixing the reservoir and establishing the storage-vector coordinate convention used in all subsequent simulation and policy-encoding calls. This explicit data contract between the environment and the agent ensures that every capability invocation operates on a shared, unambiguous representation of the physical system.

Table 1 summarizes the practical interface.

Table 1: Hydrothermal capability interface: capability groups, inputs, outputs, and analytical roles.

Group	Key inputs / outputs	Analytical role
System inspection	case data → reservoirs, stages, scenarios	Initialize session; fix storage-vector coordinate convention
	$\emptyset \rightarrow$ full topology	Expose subsystem graph, cost blocks, transmission, demand
	$\emptyset \rightarrow T,\ \lvert S \rvert$	Provide planning horizon and scenario count

Hydrological scenarios	$\emptyset \rightarrow$ mean, std per node per stage	Characterize distributional inflow structure; individual paths not accessible
Operation simulation	stage, scenario, $V \rightarrow$ costs, V'	Single-stage controlled oracle; primal outcomes only
	scenario $\rightarrow$ cost and storage trajectories	Full-horizon trajectory for one inflow path
	$\emptyset \rightarrow$ aggregate results	All-scenario batch evaluation
Policy encoding	stage, β, $\Lambda \rightarrow$ confirmation	Register affine future-cost approximations; replaces existing cuts at that stage

System-inspection capabilities expose the static structural description of the hydrothermal model. A data-loading call initializes the session and returns the ordered reservoir name, initial volume, maximum capacity, and physical unit for each storage component. This ordering is consequential: it defines the coordinate system for all storage vectors exchanged in subsequent calls, from single-stage simulation inputs to policy-cut coefficient matrices. A topology-inspection call provides the complete system description as a structured document, including subsystem nodes, directional interconnection arcs with capacity limits, thermal generation blocks with cost tiers, renewable generation profiles, and seasonal demand parameters. The resulting picture matches what a planning analyst would assemble by reviewing model documentation: system topology, resource characterization, and operating constraints.

Hydrological-scenario capabilities provide aggregate stochastic characterization of input uncertainty. For each input node, including reservoir inflows, renewable generation, and demand perturbations, the capability returns per-stage summary statistics across the scenario ensemble. Individual scenario realizations are intentionally not accessible through this capability. This choice serves two purposes. Analytically, it encourages the agent to reason from distributional structure rather than memorize individual sample paths. Operationally, it avoids injecting large raw scenario tables into the interaction history, which would expand the agent's context with low-level numerical detail and dilute the higher-level analytical signal needed for subsequent reasoning steps. The full scenario ensemble remains available indirectly through simulation and batch evaluation capabilities.

Operation-simulation capabilities constitute the computational core of the interface and offer three levels of analytical granularity. A single-stage simulation call evaluates one stage given an initial storage vector, a stage index, and a scenario index, returning a vector of per-node operating costs and the resulting end-of-stage storage vector. This oracle supports targeted perturbation experiments: by varying the initial storage vector across repeated calls while holding stage and scenario fixed, the agent can estimate the local sensitivity of operating cost to reservoir levels—the numerical procedure that yielded the marginal water values reported in Section 5. A sequence-simulation call runs a complete trajectory for a specified inflow scenario, returning stage-by-stage cost and storage profiles over the full planning horizon. A batch-evaluation call executes the model across all scenarios and is appropriate for final policy validation.

A defining property of this group is that operating costs and state transitions are observable, but dual variables and marginal prices are not. The stage solver executes internally for each simulation call and returns primal outcomes only; shadow prices are not surfaced through the interface. This constraint has direct consequences for the agent's analytical strategy. The opportunity cost of stored water must be inferred through active experimentation rather than read from solver output.

Policy-encoding capabilities allow the agent to register future-cost approximations that are incorporated into subsequent simulation calls. A policy-registration call accepts a stage index, a vector of

right-hand-side constants, and a matrix of storage coefficients, each row defining one affine future-cost approximations on the future cost function. This call replaces all existing cuts for the target stage: incremental accumulation across iteration steps is the agent's responsibility, not the environment's. The storage coefficients follow a sign convention aligned with the hydrothermal economics: a negative coefficient on reservoir i encodes the observation that additional water in that reservoir reduces expected future operating costs, i.e., that stored water has positive marginal value. Conversely, passing empty coefficient and right-hand-side arrays clears the cuts for a stage, enabling the agent to revise or reset its policy approximation during the analytical session.

This taxonomy is specific to the hydrothermal application, but it reflects a broader architectural principle: capabilities should correspond to meaningful analytical operations in the target domain, grouped at a level of abstraction that matches the reasoning granularity required for the task. In this case, the exposed capabilities mirror the workflow followed by hydrothermal planning analysts: inspect the system, characterize the uncertainty structure, simulate operational consequences under controlled conditions, and encode intertemporal policy logic. The deliberate withholding of dual information and individual scenario paths ensures that the agent must engage with the analytical problem rather than retrieve precomputed answers through the interface.

The policy-encoding capability accepts affine future-cost approximations of the form:

$$Q_t(V) \geq \beta_t - \sum_{i \in S} \lambda_i V_i, \qquad (12)$$

where $Q_t(V)$ is the approximation of future cost from stage $t+1$ onward, V_i is end-of-stage storage in subsystem i, λ_i is the marginal value of stored water, β_t is a stage-specific intercept, and S is the set of storage components represented in the model.

This representation is closely related to the affine cuts used in decomposition methods such as SDDP. However, the agent was not provided with an implementation of SDDP, any other pre-programmed stochastic dynamic programming method, or an algorithmic template for constructing these cuts. The capability only defined the admissible policy object that could be registered in the environment. The reasoning process used to estimate water values, choose coefficients, and assemble the approximation was left to the agent.

Results

Experimental task and evaluation

The agent was instructed to construct a cost-effective operating policy for the hydrothermal system. The session was allowed to proceed until the agent declared that it had completed policy construction and validation. The interaction log was then analyzed post hoc. Throughout the session, the agent had access to capability descriptions, input-output schemas, and the outputs of its own invocations. It did not receive an implementation of dynamic programming, such as SDDP, precomputed water values or dual variables, access to internal solver state, or human guidance after task initialization. All quantitative information used by the agent was therefore obtained exclusively through structured capability invocations, distinguishing this experimental setting from one in which an agent simply calls an existing optimization solver.

All sessions were conducted using Claude Sonnet 4.6 (Anthropic), a large language model with extended chain-of-thought reasoning capabilities. The model was deployed with extended thinking enabled, which allows the model to perform explicit intermediate reasoning steps before producing each response or capability invocation. Sessions used an automatic orchestration mode in which the primary reasoning model autonomously routes computationally lighter sub-tasks, such as structured-output parsing and

invocation formatting, to Claude Haiku 4.5, a smaller and lower-latency member of the same model family, while retaining full extended-thinking inference for analytical steps that require multi-step reasoning. This configuration reflects a practical deployment pattern for long-horizon agentic workflows, in which reasoning depth and computational efficiency are balanced automatically at the inference level without manual intervention.

The experiment was designed to evaluate whether the agent could:

1. Identify relevant hydrothermal system structure through inspection capabilities;
2. Infer intertemporal water-value logic through simulation;
3. Encode a future-cost approximation using the available policy representation;
4. Validate the resulting policy across stochastic inflow scenarios.

The resulting policy was evaluated using sample-mean operating cost, scenario cost variability, energy-deficit events, use of high-cost emergency thermal generation, reservoir storage trajectories, and scenario-level comparison with two references:

- **Myopic baseline:** a short-sighted policy that prioritizes immediate hydro generation without accounting for future water value, corresponding to solving each stage independently with a zero future-cost approximation.
- **SDDP benchmark:** a policy obtained independently using a purpose-built stochastic dual dynamic programming implementation.

Because reasoning agents are stochastic systems, in which their outputs depend on internal sampling processes that vary across sessions even under identical prompts and initial conditions, the experiment was conducted ten times independently, each run initialized from the same model and task specification. The ten runs produce different interaction trajectories $\tau^{(r)}$, $r = 1, \ldots, 10$, potentially leading to different policy objects $\pi_{\tau}^{(r)}$ and different operational costs. Analyzing the distribution of outcomes across runs prevents conclusions from being drawn based on a single, potentially atypical session.

Brazilian hydrothermal test case

The experiment used a simplified but structurally realistic representation of the Brazilian Interconnected Power System (SIN). The system is represented by four interconnected subsystems: Southeast/Centre-West (SECO), South (SUL), Northeast (NE), and North (N). The planning horizon covers 24 monthly stages, from January 2025 to December 2026, with ten stochastic inflow scenarios.

The test case includes aggregate hydro reservoirs, layered thermal generation blocks, renewable generation, demand profiles, and inter-regional transmission links. It is not intended to reproduce the full official Brazilian operation model. Rather, it provides a controlled benchmark with the key structural characteristics required for hydrothermal operation analysis: spatial coupling, storage dynamics, hydrological uncertainty, thermal substitution, and scarcity penalties.

Table 2 summarizes the main case parameters.

Parameter	**Value**
Planning horizon	January 2025–December 2026
Stages	24 monthly stages
Subsystems	SECO, SUL, NE, N
Hydrological scenarios	10 stochastic inflow scenarios

Parameter	Value
Policy representation	Affine future-cost cuts

Table 2: Main test-case parameters.

Table 3 summarizes the main subsystem characteristics used in the test case.

Subsystem	Hydro max. (MW)	Reservoir (GWh)	Initial storage	Demand range (MW)
SECO	35,000	146,000	75%	42,800–50,000
SUL	17,000	14,600	70%	12,900–15,800
NE	10,000	37,960	50%	12,400–14,000
N	22,000	10,950	85%	7,400–8,400

Table 3: Aggregate subsystem characteristics in the hydrothermal test case.

The test case also includes transmission links among the subsystems. The main directional limits are 15,600 MW from N to SECO, 2,000 MW from SECO to N, 6,200 MW from N to NE, 13,000 MW from NE to SECO, and 8,000 MW between SECO and SUL. Thermal generation is represented by layered blocks with increasing costs. The SECO subsystem plays a central role because its high-cost thermal block at 600 $/MWh frequently acts as a system-wide marginal resource in scarcity conditions.

Figure 3 shows the schematic of the test case, illustrating the four subsystems and the directional transmission links connecting them.

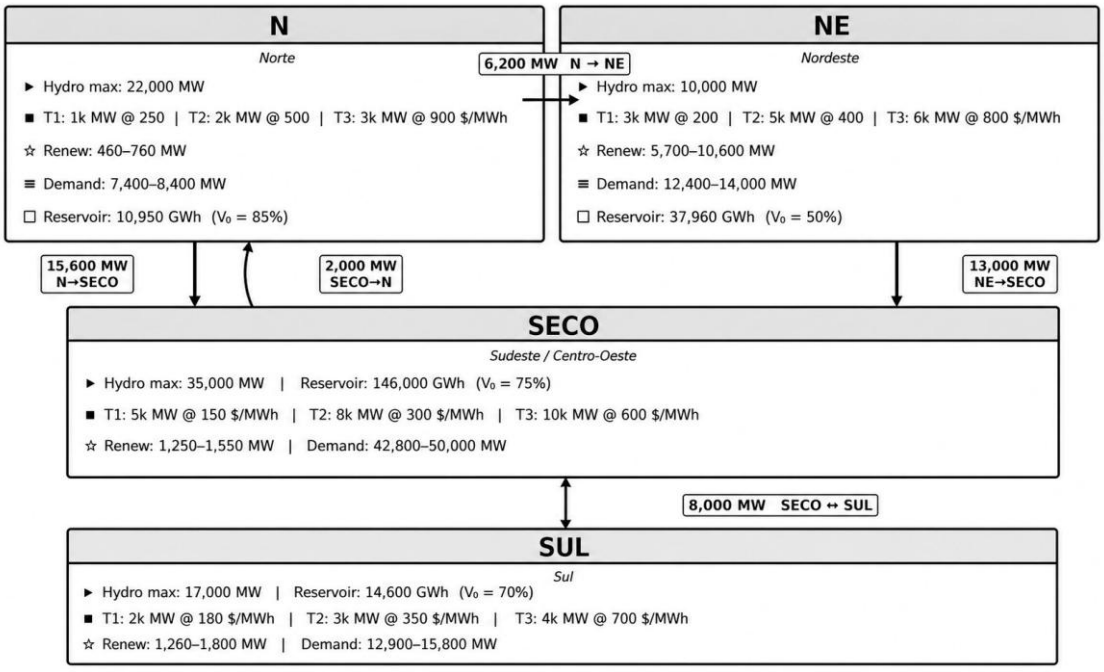

Figure 3: Schematic of the hydrothermal test case based on the Brazilian Interconnected System (SIN).

Hydrological scenario ensemble

The ten inflow scenarios were constructed to represent structurally distinct hydrological regimes rather than random perturbations around a central estimate. Table 4 reports each scenario's type and annual inflow deviation from the ensemble mean per subsystem and for the system as a whole.

Scenario	Type	SECO	SUL	NE	N	System
1	Normal	+10	−4	+4	0	+3
2	Normal	0	+11	+4	+2	+3
3	Normal	+14	+2	+10	+9	+9
4	Wet	+31	+34	+30	+31	+31
5	Wet (La Niña)	+45	0	+36	+39	+35
6	Wet	+25	+33	+21	+29	+28
7	Dry	−25	−20	−19	−25	−24
8	Dry (El Niño)	−38	+15	−28	−23	−23
9	Dry	−14	−28	−12	−15	−16
10	Critical	−47	−44	−45	−47	−46

Table 4: Hydrological scenario ensemble: type and annual inflow deviation from ensemble mean (%).

The ensemble spans three near-average realizations (scenarios 1–3), three wet years (4–6, +28% to +35% above mean), three dry years (7–9, −16% to −24% below mean), and one critical scenario (10) modelled after the 2001 Brazilian energy crisis, with all subsystems approximately 47% below their mean inflows. Total annual system inflows range from 233 300 GWh to 586 900 GWh, a factor of 2.5×, with individual subsystem ratios reaching 2.7× for SECO.

Beyond aggregate severity, several scenarios present pronounced cross-subsystem heterogeneity. Scenario 5 reproduces a La Niña-type pattern in which SECO, NE, and N are well above average (+45%, +36%, +39%) while SUL receives near-average inflows. Scenario 8 inverts this in an El Niño configuration: SECO, NE, and N severely below average (−38%, −28%, −23%) while SUL is above average (+15%). Together, the ten scenarios cover a broad spectrum of aggregate severity and spatial inflow structure, providing a demanding test of policy robustness despite the modest ensemble size.

Marginal water-value estimation

As established in the experimental design, each of the ten runs followed an independent reasoning trajectory. Despite this, inspection of the interaction logs revealed a consistent pattern across runs: the agent recurrently converged on a finite-difference approach to estimate the marginal value of stored water, independently of the specific sequence of tool invocations that preceded it. This convergence suggests that the approach is a natural consequence of the available capabilities and the problem structure, rather than an artefact of a particular session.

For each subsystem i, it compared the simulated operating cost at a reference storage vector with the cost obtained after perturbing that subsystem's storage by a small increment ΔV:

$$\hat{\lambda}_i = -\frac{C(V + \Delta V e_i) - C(V)}{\Delta V}. \qquad (13)$$

Backward construction of future-cost cuts

A second pattern that recurred consistently across runs was the strategy for constructing future-cost cuts. After estimating per-subsystem water values through finite differences, the agent systematically proceeded to encode them as affine future-cost approximations on the cost-to-go function by backward

induction — a procedure that emerged independently in each session without being prescribed in the task specification.

Using the per-subsystem water-value coefficients $\hat{\lambda}_i$ estimated in the previous step, the agent constructed a sequence of future-cost cuts by backward induction. Starting with a zero terminal value, for each stage $t = T, \ldots, 1$, it simulated operation from an initial state and computed:

$$\beta_t = c_t - \sum_i \hat{\lambda}_i V_{t,i}^{\text{fin}} + \beta_{t+1}, \qquad (14)$$

where c_t is the immediate operating cost returned by the simulator and $V_{t,i}^{\text{fin}}$ is the resulting end-of-stage storage in subsystem i.

The resulting policy is represented by an affine future-cost approximations on future cost:

$$Q_t(V) \geq \beta_t - \sum_i \hat{\lambda}_i V_i. \qquad (15)$$

Algorithm 1 summarizes the procedure reconstructed from the interaction log.

Algorithm 1 Agent-derived backward construction of affine future-cost cuts

Require: Per-subsystem water-value coefficients $\hat{\lambda}_i$, terminal intercept $\beta_{T+1} = 0$

1. **for** $t = T, T-1, \ldots, 1$ **do**
2. Simulate stage t from zero initial storage
3. Observe immediate cost c_t and final storage vector V_t^{fin}
4. Compute $\beta_t = c_t - \sum_i \hat{\lambda}_i V_{t,i}^{\text{fin}} + \beta_{t+1}$
5. Define cut $Q_t(V) \geq \beta_t - \sum_i \hat{\lambda}_i V_i$
6. **end for**
7. Register all non-terminal cuts with the optimization environment

This procedure resembles a simplified single backward pass of SDDP, but it was not supplied to the agent as an algorithmic template. The intercept profile decreases as the horizon approaches its terminal stage, reflecting the declining value of future operating periods.

Multi-run reproducibility analysis

Table 5 reports the mean operating cost obtained in each of the ten independent runs, listed in session order.

Run	Mean cost (M$)
R1	139.24
R2	148.33
R3	145.77
R4	153.26
R5	138.12
R6	153.94
R7	139.24

Run	Mean cost (M$)
R8	139.97
R9	139.21
R10	137.72
Grand mean	143.48
Std dev	6.02
Min	137.72
Max	153.94

Table 5: Mean total operating cost for each independent run (M$).

Six runs (R1, R5, R7–R10) produced policies with mean costs in the range 137.7–140.0 M$, a spread of only 2.3 M$ across six independent sessions. The remaining four runs (R2–R4, R6) yielded higher mean costs of 145.8–153.9 M$. The grand mean across all ten runs is 143.5 M$ with a standard deviation of 6.0 M$. A noteworthy observation is that runs R1 and R7 produced identical per-scenario costs to within numerical precision, indicating that the agent arrived at exactly the same policy through two independent interaction trajectories—a form of convergence that suggests the analytical path to the dominant water-value estimate is sufficiently constrained by the capability structure to be reproducible.

Reasoning trajectory of the best-performing run

Among the ten independent runs, the session that achieved the lowest mean operating cost (137.7 M$) left a sufficiently detailed interaction log to reconstruct its analytical trajectory in full. The agent began with system inspection (Phase 1), loading the model data, cost structure, and scenario statistics. In Phase 2, it simulated all ten scenarios without any registered cuts, establishing a baseline in which reservoirs drained between stages 6 and 8 in every scenario and the system operated entirely on thermal dispatch from stage 8 onward—the classic myopic collapse.

Phase 3 consisted of targeted single-stage perturbation experiments. By perturbing each reservoir in isolation at a stressed stage and scenario, the agent estimated the marginal value of stored water: approximately 600 $/MWh for SECO, SUL, and N, and 567 $/MWh for NE. A key inference was that all four subsystems were dominated by the same system-wide marginal resource—the third thermal block in SECO at 600 $/MWh—reflecting the high degree of interconnection in the system.

In Phase 4, the agent registered a first set of affine cuts with coefficients set slightly more negative than the estimated threshold ($\alpha = [-700, -700, -600, -700]$ $/MWh). Evaluating this policy revealed an inconsistency: SECO storage was accumulating while the expensive thermal block was simultaneously being dispatched, indicating that the coefficients were penalising water use too aggressively. The agent identified this over-conservation and revised the coefficients downward in Phase 5 ($\alpha_{\mathrm{SECO}} = -590$, $\alpha_{\mathrm{NE}} = -560$), aligning them with the marginal values estimated in Phase 3. The refined policy produced a mean operating cost of approximately 138 M$ and was accepted as the final output.

This trajectory is notable for its self-correcting structure. Without external guidance, the agent detected a policy inconsistency through simulation feedback and revised its representation of water value accordingly.

Comparison with myopic operation

Table 6 compares the agent-derived policy with the myopic baseline.

Indicator	Myopic	Agent
Mean cost (M$)	275.7	137.7
T3 emergency (stages)	10	0
Deficit events (stages)	4	0
Mean savings (M$)	–	138.0

Table 6: Comparison between myopic operation and the agent-derived policy.

The myopic policy rapidly depletes reservoirs during the first wet season, leaving insufficient hydraulic buffer for later dry periods. The agent-derived policy instead preserves storage, accepting higher short-term thermal costs to reduce future scarcity risk. This behavior is consistent with the economic interpretation of water values in stochastic hydrothermal operation: water should be used when its immediate benefit exceeds its expected future opportunity cost.

Figure 4 presents the stochastic evolution of reservoir storage across the ten inflow scenarios under the agent-derived policy and the myopic baseline. Each panel corresponds to one subsystem. Bold lines show the scenario mean; shaded bands report the P25–P75 and P10–P90 intervals across scenarios. Storage levels are expressed as a percentage of each subsystem's maximum capacity.

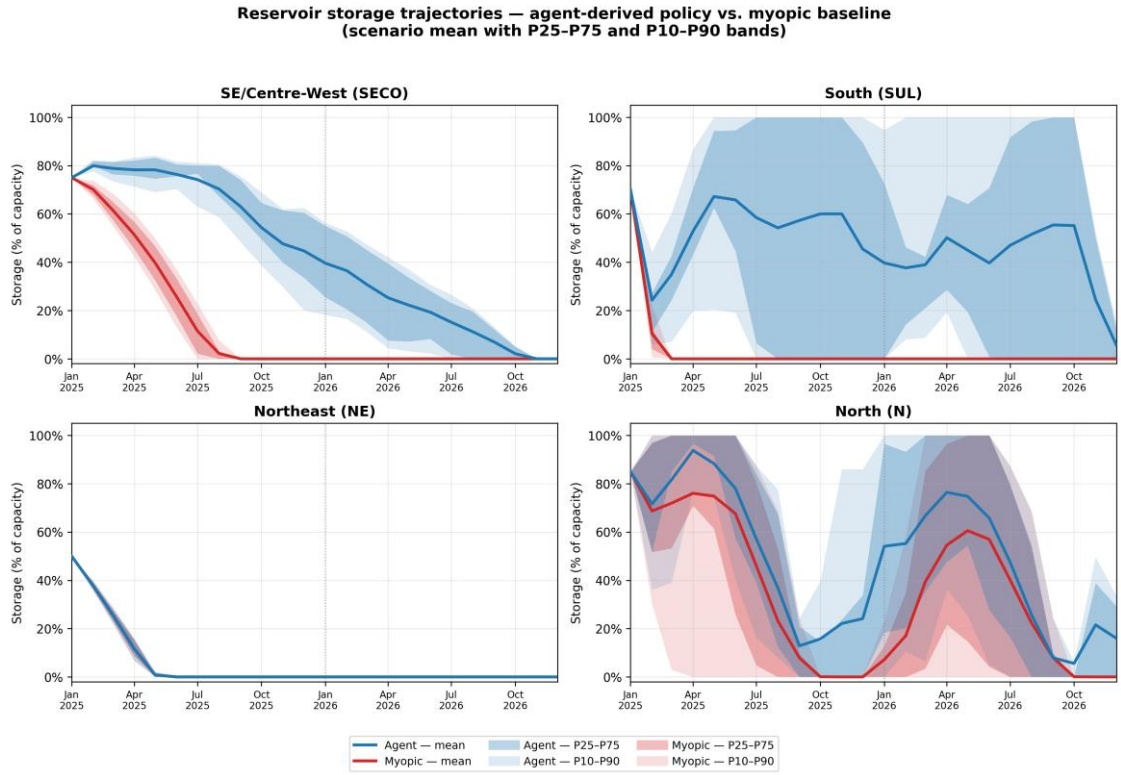

Figure 4: Reservoir storage trajectories under the agent-derived policy and the myopic baseline. Red curves denote the myopic baseline.

The contrast between the two policies is pronounced. Under the myopic baseline, all subsystems deplete to near-zero levels by stages 7–8 (July–August 2025) in most scenarios, and the system operates entirely on thermal dispatch for the remainder of both dry seasons. Under the agent-derived policy, the SECO reservoir, which concentrates approximately 70% of total system storage capacity, is maintained well above zero throughout the first dry season and enters 2026 with sufficient hydraulic reserve to buffer

the second dry period. The NE and N subsystems, which have smaller absolute capacities, exhibit similar qualitative differences, while SUL is partly recovered by its own inflow regime.

Comparison with SDDP

Table 7 compares the best-performing agent run (R10, mean cost 137.7 M$) with the SDDP benchmark using matched inflow realizations. This run is selected here because it produced the lowest mean cost among the ten independent sessions and thus represents the most favourable outcome of the capability-driven approach. Results for the remaining runs are reported in Section 5.5; across all ten runs, the mean cost gap relative to SDDP is approximately 6.0%, reflecting the variability introduced by the probabilistic nature of the reasoning agent. The SDDP reference was obtained using the PSR commercial solver [18], applied to the same ten inflow scenarios with five backward scenarios per iteration.

Scenario	Agent (M$)	SDDP (M$)	Diff. (M$)	Gap (%)
1	113.5	114.5	−1.0	−0.9
2	111.6	107.2	4.4	4.1
3	90.0	81.1	8.9	11.0
4	70.0	64.0	6.0	9.4
5	73.7	66.3	7.4	11.2
6	128.8	126.3	2.5	2.0
7	193.4	191.8	1.6	0.8
8	178.7	178.5	0.2	0.1
9	231.9	237.9	−6.0	−2.5
10	185.7	186.6	−0.9	−0.5
Mean	137.7	135.4	2.3	1.7
Std. dev.	53.2	58.5	–	–

Table 7: Scenario-level comparison between the best-performing agent run (R10) and the SDDP benchmark.

For the best run, the mean cost gap relative to SDDP is 1.7%. This result should be interpreted cautiously. The SDDP policy is obtained using a specialized stochastic optimization algorithm with repeated forward-backward passes, whereas the agent-derived policy is a simplified affine policy constructed through structured interaction. The relevant observation is therefore not that the agent competes with SDDP, but that the capability-driven environment enabled the agent to construct a policy with coherent intertemporal water-value logic.

In three scenarios, the agent-derived policy achieved lower ex post cost than the SDDP benchmark. This does not imply dominance over SDDP. Rather, it reflects scenario-level variation and the conservative nature of the agent-derived policy, which tends to preserve storage more aggressively. On average, the SDDP policy remains the better stochastic optimization benchmark.

Discussion

Structured capability orchestration

The experiment shows that a reasoning agent can use structured capabilities to carry out domain-relevant analytical operations in a stochastic energy-system environment. Its interaction trajectory resembled the workflow of a human analyst working with a simulation model: the agent did not receive an optimal policy directly, but progressively developed an approximate operating logic that guided subsequent decisions.

The important point is not that the agent discovered a new optimization method. Rather, the capability structure constrained the interaction in a productive way. By exposing capabilities at appropriate levels of abstraction, the environment enabled the agent to formulate and test operational hypotheses, infer intertemporal water-value logic, and construct an approximate representation of the cost-to-go structure underlying the stochastic dynamic programming problem.

Implications for AI-assisted energy analytics

The proposed architecture suggests a role for reasoning agents as analytical intermediaries embedded in energy-system modeling environments. Rather than replacing simulators, optimization solvers, or human experts, agents can support the exploratory layer around formal models by selecting computations, interpreting outputs, testing hypotheses, and coordinating iterative analysis.

This role is especially relevant for energy problems whose decision context is broader than a single well-specified optimization formulation. Planning and operational studies often combine uncertain assumptions, heterogeneous data, multiple models, regulatory constraints, market-design considerations, and trade-offs that are difficult to encode completely in closed analytical form. In such cases, an agentic system may be useful not because it directly computes an optimum, but because it helps structure the exploration of alternatives, sensitivities, scenarios, and policy-relevant insights.

Potential applications include:

- Exploration of large stochastic or stress-test scenario spaces;
- Comparative analysis across alternative assumptions and model configurations;
- Identification of sensitivities, bottlenecks, and operational drivers;
- Iterative refinement of candidate policies or planning alternatives;
- Orchestration of multiple simulation, forecasting, and optimization tools;
- Explanation and interrogation of model results for decision support.

In this view, capability design becomes central. The agent can only reason effectively about a domain model if relevant analytical operations are exposed in a controlled, semantically meaningful, and auditable way. Hydrothermal scheduling provides a benchmarked validation setting for this architecture because reliable optimization references are available. Future work should evaluate whether similar agentic frameworks can support solution exploration in problems where fully specified analytical formulations are incomplete, difficult to construct, or insufficient to represent the relevant decision context.

Limitations

Several limitations must be acknowledged. Some are specific to the hydrothermal scheduling case study used for validation, while others concern the broader use of reasoning agents in structured energy-system analytical environments.

Regarding the particular hydrothermal scheduling case study, the experiment was conducted on a simplified representation of the Brazilian SIN. Although structurally realistic, it does not capture the full complexity of real-world operation, including detailed unit commitment, security constraints, individual plant constraints, market rules, regulatory restrictions, and operational reserve requirements. The case should therefore be interpreted as a controlled benchmark for evaluating the architecture, not as a full representation of production-grade hydrothermal operation.

Second, the agent-induced procedure has no optimality guarantees. The quality of the resulting operating behavior depends on the hypotheses formulated by the agent, the sequence of capability invocations, and the capabilities made available by the environment. Classical stochastic optimization methods systematically explore the mathematical formulation and provide convergence guarantees under appropriate assumptions; the agent-derived procedure does not.

Finally, the scalability of sequential capability orchestration remains an open issue. Larger systems may require batch simulation capabilities, parallel scenario evaluation, stronger policy templates, context-management mechanisms, and tighter integration with production-grade solvers.

Conclusions

This paper evaluated whether a reasoning agent can operate as an analytical layer within a structured energy-system modeling environment. To this end, we proposed a capability-driven framework in which the agent interacts with the analytical environment only through structured, typed capabilities. This design separates reasoning and orchestration from validated domain computation, allowing the agent to inspect the model, run simulations, test hypotheses, and register policy-relevant artifacts without unrestricted code execution or access to solver internals.

The framework was assessed on a stochastic hydrothermal scheduling test case based on a four-subsystem representation of the Brazilian SIN. Without access to a pre-programmed stochastic dynamic programming method, dual variables, precomputed water values, or internal solver information, the agent inferred water-value logic from simulation feedback and constructed affine future-cost approximations. When embedded in the stage-dispatch model, these approximations induced storage-preserving operating policies that substantially improved upon myopic operation. Across ten independent sessions, the best policy achieved a 1.7% mean cost gap relative to an independently computed SDDP benchmark, while the average gap across sessions was approximately 6.0%.

These results do not suggest that reasoning agents should replace stochastic optimization methods. Rather, they show that, when constrained by an appropriate capability interface, agentic AI can support the exploratory layer around energy-system models: formulating hypotheses, selecting computations, interpreting outputs, and constructing policy-relevant artifacts while simulation and optimization tools remain the computational foundation.

Future work should evaluate similar capability-driven frameworks in energy-system problems where the relevant decision process extends beyond a single fully specified optimization formulation, including settings that require scenario exploration, coordination of multiple models, sensitivity analysis, and iterative policy assessment.

References

[1] S. Yao, J. Zhao, D. Yu, et al., ReAct: Synergizing reasoning and acting in language models, in: International Conference on Learning Representations, 2023.

[2] L. Wang, C. Ma, X. Feng, et al., A survey on large language model based autonomous agents, Frontiers of Computer Science 18 (2024) 186345.

[3] T. Masterman, S. Besen, M. Sawtell, A. Chandra, The landscape of emerging AI agent architectures for reasoning, planning, and tool calling: a survey, arXiv preprint arXiv:2404.11584 (2024).

[4] T.R. Sumers, S. Yao, K. Narasimhan, T.L. Griffiths, Cognitive architectures for language agents, Transactions on Machine Learning Research (2024).

[5] M.V.F. Pereira, L.M.V.G. Pinto, Multi-stage stochastic optimization applied to energy planning, Mathematical Programming 52 (1991) 359–375.

[6] A. Shapiro, W. Tekaya, J.P. da Costa, M.P. Soares, Risk neutral and risk averse stochastic dual dynamic programming method, European Journal of Operational Research 224 (2013) 375–391.

[7] T. Schick, J. Dwivedi-Yu, R. Dessì, et al., Toolformer: Language models can teach themselves to use tools, Advances in Neural Information Processing Systems 36 (2023).

[8] Y. Qin, S. Liang, Y. Ye, et al., ToolLLM: Facilitating large language models to master real-world APIs, in: International Conference on Learning Representations, 2024.

[9] NVIDIA, NVIDIA AI Blueprints: reference workflows for building generative AI and agentic applications, Technical Documentation, 2024. URL: https://www.nvidia.com/en-us/ai/blueprints/.

[10] NVIDIA, NeMo Guardrails: programmable guardrails for conversational AI applications, Technical Documentation, 2024. URL: https://docs.nvidia.com/nemo/guardrails/.

[11] Anthropic, Model Context Protocol specification, Technical Documentation, 2024. URL: https://modelcontextprotocol.io.

[12] S. Haben, S. Arora, G. Giasemidis, M. Voss, D. Vukadinović Greetham, Review of low voltage load forecasting: methods, applications, and recommendations, Applied Energy 304 (2021) 117798.

[13] A. Venzke, G. Qu, S. Low, S. Chatzivasileiadis, Learning optimal power flow: worst-case guarantees for neural networks, in: Proc. IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, IEEE, 2020.

[14] A. Castelletti, S. Galelli, M. Restelli, R. Soncini-Sessa, Tree-based reinforcement learning for optimal water reservoir operation, Water Resources Research 46 (2010) W09507.

[15] T. Yang, L. Zhao, W. Li, A.Y. Zomaya, Reinforcement learning in sustainable energy and electric systems: a survey, Annual Reviews in Control 49 (2020) 145–163.

[16] A.B. Philpott, Z. Guan, On the convergence of stochastic dual dynamic programming and related methods, Operations Research Letters 36 (2008) 450–455.

[17] P. Girardeau, V. Leclère, A.B. Philpott, On the convergence of decomposition methods for multistage stochastic convex programs, Mathematics of Operations Research 40 (2015) 130–145.

[18] PSR Energy Consulting and Analytics, SDDP – Stochastic Dual Dynamic Programming software, Software Documentation, 2024. URL: https://www.psr-inc.com/pt-br/software/sddp/.