Prospective Overthinking Mitigation: Hyperheuristics to Facilitate System 1.5

Steve Chan
Decision Engineering Analysis Laboratory, VTIRL, VT
Orlando, USA
schan@dengineering.org

***Abstract*—For certain AI-related combinatorial optimization problems, Hyperheuristics (HH) may have higher Energy Efficiency (EE) and results efficacy than that of a Metacognitive Module (MCM). For these cases, HH tend to excel in the strategy selection solution space (as contrasted to the parameter-tuning solution space). In particular, HH have shown some promise in the Large Reasoning Model (LRM) arena, where MCM have been beset by the paradigm of overthinking and analysis paralysis. This paradigm is not necessarily aligned with the need for heightened EE and effectiveness in the AI arena, and a performance paradox is illuminated for a number of cases (i.e., wherein more tokens actually segues to poorer performance). Given this counterintuitive System 2 dilemma conjoined with less robust contextual reasoning for System 1, there might be a useful intermediary state in the form of a System 1.5 (to serve as mitigation against overthinking) so as to advance the state of LRMs. Widely acknowledged NP-hard and NP-complete Combinatorial Optimization Problems (COPs) were selected, and HH as well as MCM were benchmarked. It turns out that the moniker of System 1.5 may be apropos, as the HH seem to provide some value-added propositions from both System 1 and 2.**

Keywords—Artificial Intelligence, Energy Efficiency, Large Reasoning Models, Combinatorial Optimization, Hyperheuristic, Metaheuristic, Metacognitive.

I. Introduction

A number of those engaged in the Artificial Intelligence (AI) ecosystem, such as Rangasayee, have asserted that "there must be a seismic shift in how the industry defines and measures AI advancement, pivoting our idea of technological evolution from raw computing power to Energy Efficiency" (EE) [1]. Indeed, EE has been noted as being one of the most critical challenges for the AI sector. Yet, as Lal and others have pointed out, the nexus has remained largely "unexplored in prior literature" and organizations, such as the International Energy Agency (IEA) have endeavored to "fill this gap" [2][3]. While this catch-up is occurring, advances in AI are accelerating. There are ongoing efforts to reduce AI hallucinations, improve coherence, and further the efficacy of AI construct validity efforts (e.g., ensuring that the utilized AI evaluation instrument actually gauges what it is intended to measure). Concurrently, the trend for Autonomous Systems (AS) has steered the AI sector towards certain envisioned metacognitive abilities, wherein AS are equipped with AI Systems (AIS) that can self-monitor, self-assess, and self-regulate; in this way, AS might be able to better handle intricate, unknown environments. The described Metacognitive Layer (MCL) holds promise, but some have challenged its efficacy, others have pointed out prospective cybersecurity vulnerabilities, and still others haved flagged the increased energy requirements (with dubious performance advantages). Some of these technical challenges have manifested themselves via the emergent MCL behaviors in Large Reasoning Models (LRMs)—a subset of Large Language Models (LLMs)—which are optimized for multi-step logic rather than simple next-token prediction. Ha has noted the issues of "overthinking," "redundant reasoning," "excessive computational costs," and "increased latency" characterizing many of the current LRMs, thereby "limiting the practical deployment" for Real-World Scenarios (RWS) [4].

On this point of overthinking, it is important to scrutinize Decision-Making (DM) against the dual-process theory of Epstein's Cognitive-Experiential Self-Theory (CEST) and Hammond's Cognitive Continuum Theory, which was popularized by Kahneman's framework "Maps of Bounded Rationality" and his publication, *Thinking Fast and Slow*; in the former, Kahneman presents his System 1 and System 2 as being centered upon intuition and reasoning, respectively [5]. Epstein and Hammond had referred to them as being intuitive/experiential and analytical/rational, respectively. Generally speaking, "System 1 is fast, automatic, and emotional" while "System 2 is slow, logical, and deliberate," and Tay's research is a somewhat cautionary tale of a predilection toward using System 1 [6]; Tay's research findings, wherein "up to half" of his test subjects (medical students) demonstrated full or partial reliance on System 1 (intuitive) thinking in response to" the analytical questions of Frederick's Cognitive Reflection Test (CRT), would seem to hint that using System 2 as a counterweight to System 1 seems prudent (given the described predilection); moreover, Tay notes that "complex cognitive operations eventually migrate from System 2 to System 1" anyway, "as proficiency and skill are acquired and pattern matching has replaced effortful serial processing" [6]. In addition, "when used alone, System 2 thinking can lead to poorer performance by slowing action processes down" and can "often be the fertile ground for the development of faulty thinking" [6]. This seems to be consistent with some of the findings within the AI ecosystem, wherein reasoning systems, such as LRMs, have been beset by the dilemma of *overthinking*. As IEEE Spectrum reports, "overthinking in reasoning models is linked to increased computational costs," and to aggravate matters, "it turns out that as their ability to reason improves, they increasingly fall victim to a relatable problem: analysis paralysis," wherein no action whatsoever is taken [7]; this paradigm runs counter to the issue of EE identified earlier.

Along this particular vein, Cuadron found that for some of the LRMs with high overthinking scores, their performance was not necessarily better than that of non-LRMs; in fact, Cuadron asserts that "higher overthinking scores correlate with decreased performance" [8]. To quantify this notion of overthinking, Cuadron developed an evaluation approach that focused on three key patterns: Analysis Paralysis (i.e., "excessive planning"), Rogue Actions (i.e., "multiple actions without waiting for feedback"), and Premature Disengagement ("concluding tasks without environmental validation"). Cuadron's challenge was entitled "Reasoning-Action Dilemma" (i.e., balancing "environmental engagement against internal reasoning"), and RWS tasks were evaluated against Jimenez's SWE-bench evaluation framework (which consists of "2,294 software engineering problems drawn from real GitHub issues") with Wang's Openhands platform (which contains an "agent hub with over 10 implemented agents") for the development of the utilized AI agents along with Wang's CodeAct ("a unified action space," wherein code actions can be executed and/or dynamically revised) as the "agent scaffolding" [9][10][11]. Based upon his experimental results, Cuadron asserts that "reasoning models consistently exhibit high overthinking scores – nearly three times higher than non-reasoning models" ("a phenomeon where models favor extended internal reasoning chains over environmental interaction") [8]. This translates to a higher energy cost with fewer problems being resolved; to this point, Caudron found that "generating two solutions with low reasoning effort ($800 total) and selecting the one with a lower overthinking score achieves 27.3% resolution rate" as contrasted to the high reasoning effort, which "achieves 29.1% issue resolution but costs $1,400" [8]. Elbey affirms this "performance paradox" and adds that "pushing AI to 'think longer'" may make it "less accurate," and the "resource cost" attributable to overthinking may expend "significantly more tokens," involve "generating excessive, speculative, or hallucinated details to fill the gap," and succumb to agentic "thought loops" (wherein an increased amount of the energy is spent "predicting every possible" outcome, but still leading to "analysis paralysis") [12].

Given the advantages and disadvantages of Systems 1 and 2, it seems that a System 1.5—to denote a state somewhere in-between Systems 1 and 2—might be fitting, particularly if System 1.5 is focused upon EE and well serves as a bulwark against overthinking. After all, Cuadron asserts that "overthinking *mitigation* can dramatically improve the efficiency of LRMs" in RWS applications [8]. Accordingly, this paper delves into the prospects of leveraging Hyperheuristics (HH) as a higher EE and efficacy System 1.5 mitigation approach vector (to offset some of the AIS MCL System 2 pitfalls).

The focus of this paper is on RWS AIS implementation. Section I sets the scene and provides the backdrop. The remainder of the paper is organized as follows. Section II provides pertinent background information regarding the relationship between algorithms and heuristics as well as distinguishing among Building Block Heuristics (BBH), Metaheuristics (MH), and HH. Section III describes some of the overlapping capabilities of the HH Layer (HHL) and MCL and presents some experimentation pertaining to these matters. Section IV provides results and some notable insights. Section V summarizes with concluding remarks, and proposed future work closes the paper.

II. Background

A. Algorithm versus Heuristic

An "algorithm is a set of instructions that is designed to accomplish a task" or solve a problem, and algorithms can be broadly divided into two categories: deterministic and non-deterministic [13][14]. Traditional/exact algorithms are often specific to a problem and typically "are guaranteed to locate and show an optimal solution" to that problem [15]; moreover, they are deterministic (i.e., the same input always yields the same output). While there is "no agreed mathematical definition," Rajwar notes that heuristic algorithms ("a method for producing acceptable solutions to optimization problems through trial and error") are problem agnostic, use randomness to explore varying solutions, and are generally referred to as Metaheuristic Algorithms (MAs or MHAs) [16][17]; moreover, they are non-deterministic (i.e., the same input may produce varying outputs). Algorithms can also be classified by type, such as sorting (e.g., places elements of a list in a particular order), searching (e.g., locates elements within a collection), brute force (e.g., proceeds through every candiate solution until the desired one is found), machine learning (engages in DM and/or puts forth suppositions predicated upon discerning patterns in data), etc.

Meanwhile, heuristics are defined as "approximate strategies or 'rules of thumb' for decision making and problem solving that do not guarantee a correct solution, but that typically yield a reasonable solution or brings one closer to hand" and "can be used to speed up the process of DM" [18][19]. Romanycia and Pelletier stressed the importance of clarifying the concept of a heuristic, and they put forth their own definition of 'heuristic,' which they "believe accurately summarizes what the majority of AI theorists mean by the term," for there is a "historical backdrop of conflicting definitions" [20]. They assert that a "heuristic could refer to any device used in problem solving, be it a program, a data structure, a proverb, a strategy, or a piece of knowledge," wherein: (1) "there is an element of 'rule of thumbishness' about the device," (2) there should be a "performance improvement" associated with it, (3) it should "be useful but need not guarantee success," and (4) it can guarantee supplying a solution, but if it is also provably the optimal device for arriving at a solution, then it is *not* a heuristic" [20]. Interestingly, they point out that while a myriad of "individual heuristics are discovered, tested, and modified in conjunction with a particular task or subtask,…the concept of a heuristic itself is rarely reflected upon" [20]. Along this vein, Professor Richard Karp (winner of the Association of Computing Machinery Turing Award as well as the Benjamin Franklin Medal in Computer and Cognitive Science) has been noted for asserting, "the surprising success of many heuristics remains a mystery" [21]. In any case, heuristics can also be classified by type, such as availability (e.g., DM predicated upon information that is readily at hand), anchoring & adjustment (e.g., DM

predicated upon a subjective anchoring point), affect (e.g., DM predicated upon non-logic facets, such as intuition, emotions, etc.), etc.

B. *Heuristics/Algorithms versus MCM*

Generally speaking, there are three broad categories of heuristics: BBH, MH, and HH. First, the BBH is the most basic unit and is considered to be a problem-specific heuristic that may address a facet of the problem at hand (i.e., narrow scope). Also, the BBH is often a specific algorithm or implemented as an algorithm (e.g., Greedy Algorithm, Nearest Neighbor Algorithm) to solve a specific problem. Second, the MH may guide various BBH to address the problem at hand (i.e., general-purpose scope), and the MH is typically a more generic algorithm that can be applied to a range of problems, such as Ant Colony Optimization (ACO), Genetic Algorithm (GA), and Simulated Annealing (SA). In fact, Carmacho-Villalon treats MH as a framework that defines problem-specific MHA that "provide 'good' solutions to various optimization problems with relatively few adaptations" [22]. Third, HH may select/sequence MH/BBH for the involved problem-solving or may generate MH/BBH (i.e., dynamic scope); the HH is an algorithm for managing heuristic selection/generation. In contrast, the MCM might adjust a MH, and the adjustment process (e.g., modification of the learning rates, tuning of the parameters, or switching heuristics) is executed using algorithms that are leveraged; however, the MCM itself is not a standalone algorithm. In terms of ability to take action, BBH>MH>HH>MCM; with regards to the MCM, while it is possible that the self-reflection and adapation may occur over time, there is no guarantee that this will indeed happen (i.e., there may be no action—analysis and/or decision paralysis).

Having said that, the MCM can be quite useful. Taking the case of the MH, as MH explorations endeavor to yield more promising candidate solutions, the MCM can analyze the MH maneuvers for the successful explorations, store them, and have them ready as part of the repertoire of known reliable solutions (i.e., apriori experience). In the case where the MH keeps exploring the solution space in a sub-optimal fashion, the MCM can intervene and reallocate resources towards something more productive (i.e., efficient resource allocation). Likewise, MCM can adjust the parameters of the involved MHA (i.e., adaptive parameter tuning). The MCM may also opt to switch MHAs. For example, it may select PSO (swarm-based) or GA (evolution-based) for broad exploration during the inital exploration phase, but for later phases, it may select SA (physics-based) for fine-tuned exploration. It should be noted that there is *no actual solution generation* via the MCM; the MCM's contribution is simply analysis (but not necessarily with any follow-on action). In a sense, the MH serves as the action engine and traverses the solution space graph to explore potentially better solutions. MHA (e.g., GA, SA, etc.) constitute the engine types. The HH may opt to switch the entire vehicle type (for a different approach vector entirely). Some researchers claim that "prompt evolution" segues to MCL being endowed with generative capabilities (e.g. Metacognitive LLM-Driven Architecture or MeLA) [23]. However, due to the uncertainties of output (as noted by many practitioners in the prompt engineering sector), this particular approach vector will be put aside for the purposes of this paper, which focuses upon RWS AIS implementations. Future work will delve into this matter further, such as exploring the pitfalls of prospective infinite, ungrounded loops (i.e., circularity). Ye argues that the MCL prompting "to perform automatic prompt engineering" has limited potential "due to insufficient guidance for complex reasoning in the meta-prompt" and points out that "the challenge of prompt engineering a prompt engineer remains ongoing" [24]. Accordingly, the more accepted functional roles of the HHL and MCL are used for this paper. Per Dokeroglu, HH are "search techniques for selecting, *generating*, and sequencing (meta)-heuristics" [25]. In contrast, MCM tend to adjust prompts and/or strategies based upon performance feedback. In other words, while HH are explicitly designed for selecting, sequencing, and *generating* heuristics in a more structured fashion, the MCM might or might not modify/evolve/improve heuristics over time (this will be contingent upon its self-awareness and learning process). Moreover, from a technical perspective, as the MCM is more System 2 oriented, the prospective generation of a new heuristic will take time. Even from a cognitive perspective, as noted by Thorstad, "the metacognitive control task is to decide whether to let heuristic deliberation go ahead or to override it in favor of explicit calculation" [26]. Thorstad further articulates the MCM's goal of rational metacognitive control—balance the risks of failing to block unreliable heuristic strategies (i.e., false positives) and intervening to block reliable heuristic strategies (i.e., false positives). In essence, the MCM is intended to be the arbiter of whether to utilize or reject a heuristic, rather than the generator of the heuristic itself.

III. Experimentation

Several Combinatorial Optimization Problems (COPs) (as well as a Multi-Objective Optimization Problem or MOOP, which overlaps as a Multi-Objective Combinatorial Optimization or MOCO problem) are selected, wherein both the HCL and MCL can be compared so as to articulate the impact of varied heuristic selection and parameter-tuning adjustments. The COPs were able to prominently feature either the HCL adaptability (e.g., switching among various heuristics) or MCL tuning capability (e.g., fine-tuning a single heuristic's parameters).

A. *Solution Strategies Selected for Experimentation*

The HHL and MCL differ in their objectives, temporal focus, and processes. While there is some overlap in their overarching capabilities, the involved processes are quite distinct. HH generally leverage lower-level heuristics (e.g., MH, BBH) "by selecting/generating" the most apropos heuristical amalgam "rather than traversing the search space itself, as this can be performed more efficiently by a lower-level heuristic" and/or a MHA [25]. The HH is an interesting construct in that it can select which MH and/or BBH to apply and can also generate new heuristics (MH and/or BBH) dynamically in a fashion more akin to System 1 (e.g., System 1.5). That is something that the MCM is not necessarily able to do (i.e., the MCM can switch the MHA utilized, modify the parameters utilized, and adjust strategies, but it is not able to generate BBH, MH/MHA, or HH in a System1/1.5 fashion). Rather, the focus of the MCM is on self-improvement and should it engage in heuristic generation, which may or may not

be the case, it would be the result of the MCM's evolving understanding of the problem space. However, the MCM may also be prone do analysis paralysis and decision paralysis given too much overthinking [12][27]. The promise of HH, as noted by Burke and others, is that it "raise[s] the level of generality" and is "broadly concerned with intelligently choosing the right heuristic or algorithm in a given situation" [28].

B. Optimization Problems Selected for Experimentation

Hamilton's and Kirkman's Traveling Salesman Problem (TSP) of the 1800s (which evolved with Menger and others), Johnson's Flow Shop Scheduling Problem (FSSP) of 1954, Koopman's and Beckmann's Quadratic Assignment Problem (QAP) of 1957, Dantzig's (and Ramser's) Vehicle Routing Problem (VRP) of 1959, Muth's and Thompson's Job Shop Scheduling Problem (JSSP) of 1963, and Johnson's Bin Packing Problem (BPP) of 1973 are all considered to be Non-deterministic Polynomial-time hard (NP-hard) [29][30][31][32][33][34][35]; likewise, MOOPs are typically NP-hard, which refers to the case, wherein there is no known polynomial-time algorithm to find the exact solution for all instances, and as the problem size increases, the time required to find the optimal solution grows exponentially (e.g., $O(2^n)$) (i.e., NP-hard problems include problems that might be outside of NP). Pertaining to the experimentation herein, Peres notes that COPs "are usually NP-hard problems, and there is not a deterministic method of solving them in polynomial-time" [36]. Turning to the NP-Complete case, Guthrie's Four-Color Problem/Conjecture of 1852, a specific case of the Graph Coloring Problem (GCP), and Dantzig's Knapsack Problem (KP) (a.k.a., 0/1 KP or KP01) are representative [37][38]. Delving into these exemplars, NP-Complete refers to the paradigm, wherein the solution is easily validated, but it is difficult to determine the solution; these are the most difficult problems within the NP class and are construed to be "complete," as the other problems in NP can be reduced to them in polynomial time (i.e., NP-complete problems are a subset of NP-hard problems that are also in NP). Restated in Perez's words, "if a problem is NP and other NP problems are polynomial-time reducible to it, the problem is NP-complete" [36]. Overall, NP-hard problems and NP-complete problems are deemed to be intractable, as no known algorithms can resolve them in polynomial time ($O(n^k)$) (e.g., as the input size grows), and they necessitate exponential time $O(2^n)$ to find exact solutions [39]; in those cases, wherein a brute-force approach is involved, the NP-hard problem might be resolved with factorial time complexity (e.g., $O(n!)$). Leyton-Brown put it quite well: "problems are intractable when they 'can be solved, but not fast enough for the solution to be usable'" [40]. For the experimentation herein, certain NP-hard and NP-Complete COPs were chosen due to their RWS AI applicability, as shown in Table I.

TABLE I. COP TYPE/RWS AI APPLICABILITY

COP	*Type/RWS AI Applicability*	*Typical objective*
TSP	Routing (pathfinding): AI circuit layout optimization, such as for wire length and signal delay.	Minimize total route distance.
FSSP	Scheduling (task ordering/sequencing): AI-related copper smelting and refining scheduling, as copper wiring for the Printed Circuit Board (PCB), interconnects, and networking infrastructure is vital.	Minimize makespan (total production time).
QAP	Minimizing a Quadratic Cost Function: AI Data Center (AIDC) optimal server placement so as to minimize communication delay and power consumption.	Minimize assignment cost (total cost/distance * flow).
VRP	Routing (pathfinding): AI workload scheduling to servers, such as for on-peak or off-peak or for when renewables load is high.	Minimize the total route distance or travel cost.
JSSP	Scheduling (task allocation/assignment): AI assembly line optimization for PCBs (e.g., soldering, inspection, testing).	Minimize the makespan or total requisite time.
BPP	Packing (resource allocation): AI workload assignment to servers so as to maximize utilization and minimize numbers of servers used.	Minimize the number of bins used.
MOOP	Identifying Pareto Optimality: For the AIDC, minimizing energy cost while maximizing reliability.	Minimize/maximize ≥ 2 conflicting objectives simultaneously (e.g., minimize cost & time simultaneously).
GCP	Constraint Satisfaction Problem (CSP): Variables within an AI program must be assigned to different registers.	Minimize the number of colors used.
KP	Packing (resource allocation): AI-related critical materials cargo loading, such as for palette positions on an aircraft.	Maximize the total value (or minimize cost/negative value in some variants).

The classical COPs of Table I are invaluable for testing LRMs. After all, LRMs are reasoning and problem-solving agents, and the COPs of Table I offer the challenges of, among others, multi-step reasoning/DM. When LRMs are tested on these COPs, they reveal their ability to decompose a complex problem into more tractable subproblems, evaluate alternatives in a systematic fashion, conduct planning, engage in generalization, and optimize for an objective under varied constraints. Since LRMs undertake planning and optimizing, by training on COPs, they can learn heuristical strategies that are transferable to RWS agentic tasks. For example, KP can test the LRM for resource allocation, and the LRM can obtain the agentic relevance of prioritizing objectives amidst constrained resources. As another example, TSP can test the LRM for path optimization, and the LRM can obtain the agentic relevance of planning multi-step actions in an efficient fashion. Furthermore, while KP, GCP, etc. are typically demonstrative of medium autonomy, TSP, VRP, JSSP, etc. are typically demonstrative of high autonomy.

C. A Python Framework for Experimentation

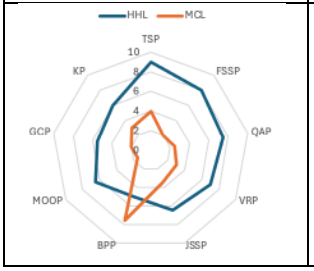
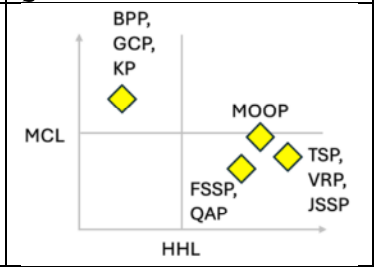
The requirements for the experimental framework included being able to: (1) address varied COPs (e.g., TSP, FSSP, QAP, etc.), (2) execute distinct and disparate solution strategies (i.e., HHL and MCL), and (3) compare and contrast the quantitative results and compute impact percentages for each. By way of background, each COP will have its own problem generator/instance loader and a solution evaluator/objective

function for the multiple instances involved (e.g., 100). Each COP is addressed via the coded HH and MCM Solvers. Specifically, the computed impact percentages are revalidated, wherein the revalidation is set to ([MCL solution quality – HHL solution quality]/[HHL solution quality])*100 (for which a positive value indicates that the MCL has greater impact while a negative value indicates that HHL has greater impact). A paired samples t-test (which "is a commonly used statistical procedure for comparing the means of two paired populations to determine their equality") solution quality and runtime was utilized [41]. The Fisher $P < 0.05$ (5% significance) cut-off point was used to determine whether the difference is statistically significant [42]. In this way, it can be ascertained for which COPs did MCL truly outperform HHL or vice versa.

IV. RESULTS AND DISCUSSION

As can be seen in Table II below, the HHL seems to have a relatively high impact on TSP, VRP, JSSP, FSSP, and QAP; MCL seems to have a high impact on BPP, GCP, and KP. MOOP seems be somewhat more highly impacted by HHL, but this needs to be further explored in future work. Also, it seems that HHL efficacy depends principally on heuristical diversity (i.e., the number of fundamentally distinct heuristics) as well as the instance diversity sensitivity (i.e., the varying efficacy differences across instances). Seemingly, the more distinct the heuristics (wherein the inner workings needs to be carefully understood), the higher the HHL efficacy. In addition, the inherent HHL adaptability seems to imbue it with variability across instances, thereby increasing HHL efficacy. In contrast, the MCL efficacy depends primarily on the heuristical parameter sensitivity, wherein the MCL efficacy dramatically increases only if the involved tuning has a significant impact. This also segues to the notion that MCL is intricately linked to the solution structure stability (wherein the involved heuristical efficacy increases, if it is well-tuned). Consequently, if the heuristic is sensitive to tuning, the MCL efficacy will be higher; along this vein, if the solution structure is indeed stable, then the MCL is more likely to deeply optimize. Overall, the MCL tends to have greater impact when the approach relies upon a single heuristic that requires fine-tuning; it should be noted that a MHA is typically comprised of an algorithm, and a BBH also contains algorithmic logic (which may be aggregated by the MH/HH). Those COPs, wherein the HHL seems to have greater impact, are more likely to be strategy-space dominant problems; those COPs, wherein the MCL seems to have higher impact, are more prone to be parameter-space dominant problems.

TABLE II. IMPACT PERCENTAGES FOR HHL&MCL (FOR VARIOUS COP; BY MCL:HHL)

Radar Chart	*Quadrant Chart*
	

On the earlier point of the inner workings, let us take BPP. Some of the more prototypical heuristics involved might be: First-Fit (FF), First-Fit Decreasing (FFD), Best-Fit (BF), Best-Fit Decreasing (BFD), Worst-Fit (WF), and Worst-Fit Decreasing (WFD); however, the aforementioned are considered to be closely related because they are all, in essence, greedy one-pass heuristics that are focused upon local DM (i.e., they are generally considered to be one heuristic family). Taking the case of VRP, there are several heuristic families that might be involved, such as those in Table III.

TABLE III. HEURISTIC FAMILIES, SUBCLASSES/VARIANTS [43]

Family Category	*Subclass/Variant*			*Ref*
Constructive (formulates solutions incrementally)	Greedy/Priority-Based			[44]
	Nearest Neighbor			[43]
	Savings			[45]
Improvement/ Local Search (refines existing solutions)	Intra-Route	Exchange		[43]
		Relocate		[43]
		2-Opt, 3-Opt		[46]
	Inter-Route	Insert		[47]
		Relax-and-Repair/Swap		[48]
Metaheuristic/ Adaptive (executes expeditionary/ exploratory endeavors for locating solutions	Single-Solution-based	Perturbation	Iterated Local Search (ILS)	[49]
		Change	Large Neighborhood Search (LNS)	[50]
		Adaptive	Simulated Annealing (SA)	[51]
	Population-based	Evolutionary	Genetic Algorithm (GA)	[51]
		Swarm Intelligence	Ant Colony Optimization (ACO)	[51]
			Particle Swarm Optimization (PSO)	[51]

Table III should make it quite clear that the family categories are distinct (e.g., constructive, improvement/local search, and metaheuristic/adaptive); in addition, there are a number of subclasses/variants delineated to illuminate the point that the heuristics are varied (i.e., high heuristic diversity).

V. CONCLUSION AND FUTURE WORK

As the focus and intent of this paper is on RWS AIS implementations, it was critical that a key tenet of security, resiliency, and efficacy was considered—that of prospective species diversification to avoid transitive closure (which facilitates ecosystem stability). Allesina and others buttress this tenet [52]. The combination of MH and MCM constitutes an adaptive, self-improving solution space search paradigm, and the amalgam of MH, MCM, and HH form an even more robust temporally- capable triumvirate. It was shown how HH, MCM, and MH can be complementary and potentially embody a more robust System 1, 1.5, and 2 paradigm. Also, the finding that HH excels in strategy-space dominant problems (e.g., TSP, VRP, etc.) while MCL performs better on parameter-space dominant problems (e.g., BPP, GCP, etc.) offers practical insight. Overall, the paper addresses a looming challenge within the AI ecosystem—overthinking in large LRMs—and proposes HH as

a "System 1.5" mitigation approach. More quantitative experimentation in this regard will be conducted in future work.

REFERENCES

[1] K. Rangasayee, "The AI energy challenge is coming to a head," *UtilityDive*, Feb. 2025. Accessed: Apr. 9, 2026. [Online]. https://www.utilitydive.com/news/ai-energy-challenge-efficiency-data-center/740277/.

[2] A. Lal and F. You, "Advances and challenges in energy and climate alignment of AI infrastructure expansion," *Advances in Applied Energy*, vol. 20, pp. 1-23, Dec. 2025.

[3] "Energy and AI," *International Energy Agency (IEA)*, Apr. 2025, Accessed: Apr. 9, 2026. [Online]. https://www.iea.org/reports/energy-and-ai.

[4] R. Ha, C. Li, R. Pu, and S. Su, "From 'Aha' Moments to Controllable Thinking: Toward Meta-Cognitive Reasoning in Large Reasoning Models via Decoupled Reasoning and Control," *Arxiv.org*, Aug. 2025. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2508.04460v1.

[5] D. Kahneman, "Maps of Bounded Rationality: Psychology for Behavioral Economics," *The American Economic Review*, vol. 93, pp. 1449-1475, Dec. 2003.

[6] S. Tay, P. Ryan, and C. Ryan, "Systems 1 and 2 thinking processes and cognitive reflection testing in medical students," *Canadian Medical Education Journal*., vol. 7, pp. e97-e103, Oct. 2016.

[7] M. Smith, "It's Not Just Us: AI Models Struggle With Overthinking: Overthinking in reasoning models is linked to increased computational costs," *IEEE Spectrum*, Mar. 2025. Accessed: Apr. 9, 2026. [Online]. https://spectrum.ieee.org/reasoning-in-ai.

[8] Cuadron et al., "The Danger of Overthinking; Examining the Reasoning-Action Dilemma in Agentic Tasks," *Arxiv.org*. Feb. 2025, Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2502.08235.

[9] C. Jimenez et al., "SWE-bench: Can Language Models Resolve Real-World Github Issues?" *Arxiv.org*, Nov. 2024. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2310.06770.

[10] X. Wang et al., "Openhands: An Open Platform for AI Software Developers as Generalist Agents," *Arxiv.org*, Apr. 2025. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2407.16741.

[11] X. Wang et al., "Executable Code Actions Elicit Better LLM Agents," *Arxiv.org*, Jun. 2024. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2402.01030.

[12] F. Elbey, "The overthinking problem in AI," *Amazon Science*, Nov. 2025. Accessed: Apr. 9, 2026. [Online]. https://www.amazon.science/blog/the-overthinking-problem-in-ai.

[13] "Algorithm," *National Library of Medicine*, Accessed: Apr. 9, 2026. [Online]. https://www.nnlm.gov/resources/data/data-glossary/algorithm.

[14] "Algorithms: their meaning in computer science," *International Institute in Geneva*, Accessed: Apr. 9, 2026. [Online]. https://www.iig.ch/en-en/blog/computer-science/algorithm-computer-science-definition-and-understanding.

[15] H. Tahami and H. Fakhravar, "A Literature Review on Combining Heuristics and Exact Algorithms in Combinatorial Optimization," *European Journal of Information Technologies and Computer Science*, vol. 2, pp. 6-12, 2022.

[16] K. Rajwar, K. Deep, S. Das, "An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges," *Artificial Intelligence Review*, pp. 1-71, Apr. 2023.

[17] X. Yang, "Nature-inspired optimization algorithms," *Elsevier*, 2020.

[18] "Heuristics," *ScienceDirect*, Accessed: Apr. 9, 2026. [Online]. https://www.sciencedirect.com/topics/biochemistry-genetics-and-molecular-biology/heuristics.

[19] M. Hjeij and A. Vilks, "A brief history of heuristics: how did research on heuristics evolve?" *Humanities of Social Sciences Communications*, vol. 10, pp. 1-15, Feb. 2023.

[20] M. Romanycia and R. Pelletier, "What is a heuristic?" *Computational Intelligence*, vol. 1, pp. 47-58, 1985.

[21] R. Karp, "Reducibility Among Combinatorial Problems," M. Junger et al. (eds.), *50 Years of Integer Programming 1958-2008*, Springer-Verlag, 2010, pp. 219-220.

[22] C. Camacho-Villalon, T. Stutzle, and M. Dorico, "Beyond metaphors: rethinking metaphors in metaheuristics algorithm design," *Artificial Intelligence*, vol. 2, pp. 1-4, Mar. 2026.

[23] Z. Qiu, X. Chen, L. Chen, and R. Bai, "MeLA: A Metacognitive LLM-Driven Architecture for Automatic Heuristic Design," *Arxiv.org*, Sep. 2025. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2507.20541.

[24] Q. Ye, M. Axmed, R. Pryzant, and F. Khani, "Prompt Engineering a Prompt Engineer," *Arxiv.org*, Jul. 2024. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/html/2311.05661v3.

[25] T. Dokeroglu, T. Kucukyilmaz, and E. Tabli, "Hyper-heuristics: A survey and taxonomy," Computers and Industrial Engineering, vol. 187, pp. 1-18, Jan. 2024.

[26] D. Thorstad, "Two paradoxes of bounded rationality," Philosophers' Imprint, vol. 22, pp. 1-16, Oct. 2022.

[27] Y. Liu, H. Li, X. Ma, J. Zhang, and S. Guo, "Think How to Think: Mitigating Overthinking with Autonomous Difficult Cognition in large Reasoning Models," *Arxiv.org*, Aug. 2025, Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/html/2507.02663v2.

[28] E. Burke et al., "Hyper-Heuristics: An Emerging Direction in Modern SearchTechnology" F. Glover and G. Kochenberger (eds.), *Handbook of Metaheuristics: International Series in Operations Research & Management Science*, vol. 57, pp. 457-474, 2003.

[29] E. Islam, M. Sultana, and F. Ahmed, "A Tale of Revolution: Discover and Development of TSP," *International Journal of Mathematical Trends and Technology (IJMTT)*, vol. 57, pp. 136-139, May 2018.

[30] J. Gupta and E. Stafford, "Flowshop scheduling research after five decades," *European Journal of Operational Research*, vol. 169, pp. 699-711, Mar. 2006.

[31] N. Daly, T. Krauss, J. Shapiro, "Quantum Approaches to the Quadratic Assignment Problem," *Arxiv.org*, Apr. 2025. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2505.00182.

[32] S. Elatar, K. Abouelmehdi, and M. Riffi, "The vehicle routing problem in the last decade: variants, taxonomy and metaheuristics," *Procedia Computer Science*, vol. 220, pp. 398-404, 2023.

[33] G. Col and E. Teppan, "Industrial-size job shop scheduling with constraint programming," *Operations Research Perspectives*, vol. 9, pp. 1-19, 2022.

[34] D. Johnson, "Near-optimal bin packing algorithms," PhD dissertation, MIT, Cambridge, MA, 1973.

[35] D. Johnson, "Fast Algorithms for Bin Packing," *Journal of Computer and System Sciences*, vol. 8 pp. 272-314, 1974.

[36] F. Peres and M. Castelli, "Combinatorial Optimization Problems and Metaheuristics: Review, Challenges, Design, and Development," *Applied Science*, vol. 11, pp. 1-39, Jul. 2021.

[37] V. Voloshin, "Graph Coloring: History, results and open problems," *Alabama Journal of Mathematics*, pp. 1-3, Feb. 2009.

[38] V. Cacchiani, M. Iori, A. Locatelli, and S. Martello, "Knapsack problems – An overview of recent advances. Part I: Single knapsack problems," *Computers & Operations Research*, vol. 143, Jul. 2022.

[39] M. Moldoveanu, "Intelligent Artificiality: Algorithmic Microfoundations for Strategic Problem Solving," *Harvard Business School Working Papers*. Accessed: Apr. 9, 2026. [Online]. https://www.hbs.edu/ris/Publication%20Files/19-072_1da6ef3d-ac69-47d7-aec3-3f7616c0e3f9.pdf.

[40] K. Leyton-Brown, H. Hoos, F. Hutter, and L. Xu, "Understanding the Empirical Hardness of NP-Complete Problems," *Communications of the ACM*, vol. 57, pp. 98-10, May 2014.

[41] A. Talikan et al., "On Paired Samples T-Test: Applications, Examples, and Limitations," *International Journal for Multidisciplinary Research*, vol. 2, pp. 1-9, Apr. 2024.

[42] T. Dahiru, "P-Value, A True Test of Statistical Significance? A Cautionary Note," *Annals of Ibadan Postgraduate Medicine*, vol. 6, pp. 21-26, Jun. 2008.

[43] F. Liu et al., "Heuristics for Vehicle Routing Problem: A Survey and Recent Advances," *Arxiv.org*, Mar. 2023, Apr. 2025. Accessed: Apr. 9, 2026. [Online]. https://arxiv.org/pdf/2303.04147.

[44] Y. Guo and M. Muramatsu, "A Greedy Algorithm for Priority-Based Vehicle Routing Problem," *19th IIAI International Congress on Advanced Applied Informatics*, Mar. 2026, pp. 25-30.

[45] S. Avdoshin and E. Beresneva, "Constructive heuristics for Capacitated Vehicle Routing Problem: a comparative study," *Proceedings of the Institute for System Programming of RAS*, vol. 31, pp. 145-156, 2019.

[46] E. Baker and J. Schaffer, "Solution Improvement Heuristics for the Vehicle Routing and Scheduling Problem with Time Window Constraints," *American Journal of Mathematical and Management Sciences*, vol. 6, pp. 261-300, Aug. 2013.

[47] A. Campbell and M. Savelsbergh, "Efficient Insertion Heuristics for Vehicle Routing and Scheduling Problems," *Transportation Science*, vol. 38, pp. 369-378, Aug. 2004.

[48] N. Absi, D. Cattaruzza, D. Feillet, and S. Housseman, "A relax-and-repair heuristic for the Swap-Body Vehicle Routing Problem," *Annals of Operations Research*, vol. 253, pp. 957-978, 2017.

[49] Y. Wu, H. Du, and H. Song, "An Iterated Local Search Heuristic for the Multi-Trip Vehicle Routing Problem with Multiple Time Windows," *Mathematics*, vol. 12, pp. 1-16, May 2024.

[50] D. Dumez, F. Lehuede, and O. Peton, "A large neighborhood search approach to the vehicle routing problem with delivery options," *Transportation Research Part B. Methodological*, vol. 144, pp. 103-132, Feb. 2021.

[51] M. Kumari, P. De, K. Chaudhuri, and P. Narang, "Utilizing a hybrid metaheuristic algorithm to solve capacitated vehicle routing problem," *Results in Control and Optimization*, vol. 13, pp. 1-15, Dec. 2023.

[52] S. Allesina and J. Levine, "A competitive network theory of species diversity," Proceedings of the National Academy of Sciences, vol. 108, pp. 5638-5642, Mar. 2011.