Energy-Efficient Hierarchical Planning and Action Compression for Edge-Deployed Large Language Model Reasoning Systems

Brent Rhodes
Department of Computer Science and Engineering, University of Nevada, Reno, Reno, NV, USA.
brent.work@unr.edu

Jordan Wood
Department of Computer Science, University of New Hampshire, Durham, NH, USA.
jordan.wood@unh.edu

Noah A. Simmons
Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS, USA.
noah.a.simmons@ku.edu

Abstract

The deployment of large language models on edge devices for real-time reasoning tasks introduces substantial challenges related to energy consumption, latency, and computational resource constraints. This paper proposes a novel framework that integrates hierarchical planning with action compression to address these challenges, enabling energy-efficient reasoning in edge-deployed large language model systems. The hierarchical planning approach decomposes complex reasoning tasks into high-level strategic goals and low-level execution steps, reducing the computational overhead associated with full autoregressive generation. Action compression techniques further minimize the number of tokens and intermediate reasoning steps by leveraging learned abstractions and context distillation. We examine the architectural trade-offs between planning depth and energy savings, the governance implications of deploying compressed reasoning pipelines in resource-constrained environments, and the robustness and fairness considerations that arise when reducing model expressivity for efficiency. Drawing on cross-domain comparisons from robotics, autonomous systems, and distributed computing, we argue that a structured, multi-level planning paradigm combined with compression strategies can significantly decrease energy footprint without catastrophic loss of reasoning quality. The framework also enables more predictable latency, improved scalability across heterogeneous edge hardware, and enhanced sustainability for large-scale deployment. We discuss policy implications for carbon-aware computing and equitable access to intelligent edge services. This work contributes a systems-level perspective on making advanced reasoning capabilities viable for edge deployment while maintaining responsible stewardship of energy resources.

Keywords

large language models, edge computing, hierarchical planning, action compression, energy efficiency, sustainability, robust reasoning.

1. Introduction

The rapid advancement of large language models has produced remarkable capabilities in natural language understanding, generation, and reasoning. However, the deployment of these models in edge environments, such as mobile devices, IoT sensors, and autonomous vehicles, remains constrained by the substantial computational and energy demands of autoregressive decoding [1][2]. Edge devices typically operate under strict power budgets, limited memory bandwidth, and intermittent connectivity, making the direct porting of cloud-scale large language models impractical for many real-time reasoning applications. Addressing this gap requires novel architectural innovations that reconcile the expressive power of large language models with the resource limitations of edge hardware.

One promising direction is the adoption of hierarchical planning, where reasoning is structured into high-level strategic decisions and low-level execution actions. This approach mirrors well-established practices in robotics and automated planning, where decomposition reduces search complexity and enables more efficient resource allocation [3][4]. In the context of large language models, hierarchical planning can guide the generation process by first producing abstract plans that specify the sequence of reasoning steps, and then instantiating each step with minimal token generation. Such a structure avoids the wasteful exploration of irrelevant branches and focuses computational effort on the most promising paths.

Action compression further complements hierarchical planning by reducing the number of tokens required to represent each reasoning action. Techniques such as token merging, latent summarization, and prompt distillation have shown that many intermediate reasoning steps can be collapsed into compact representations without significant loss of fidelity [5][6]. When applied in conjunction with hierarchical planning, action compression can yield multiplicative gains in energy efficiency, as both the number of high-level decisions and the granularity of low-level actions are reduced.

This paper proposes a unified framework for energy-efficient hierarchical planning and action compression tailored to edge-deployed large language model reasoning systems. We discuss architectural trade-offs, operational governance, and the broader implications for sustainability, robustness, and fairness. Through system-level analysis and cross-domain comparisons, we demonstrate that the proposed approach can achieve substantial reductions in energy consumption while maintaining competitive reasoning performance. The remainder of this paper is organized as follows. Section 2 reviews related work in hierarchical reasoning and energy-efficient deep learning. Section 3 presents the architectural framework. Section 4 details action compression mechanisms. Section 5 analyzes energy efficiency and edge deployment considerations. Section 6 explores sustainability and governance implications. Section 7 addresses robustness and fairness challenges. Section 8 provides case illustrations and cross-domain comparisons. Section 9 outlines future research directions, and Section 10 concludes.

2. Background and Related Work

Large language models have demonstrated impressive reasoning capabilities through autoregressive generation of chain-of-thought sequences. However, the computational cost of generating lengthy reasoning chains grows linearly with chain length and quadratically with model size, leading to prohibitive energy consumption on edge devices [7]. Various approaches have been proposed to mitigate these costs, including model pruning, quantization, knowledge distillation, and speculative decoding. Yet these methods often treat the reasoning process as a monolithic pipeline, missing opportunities for structural optimization.

Hierarchical planning has been extensively studied in robotics and artificial intelligence planning communities. The concept of partitioning a complex problem into abstract subgoals and then executing primitive actions is central to methods like hierarchical task networks and hierarchical reinforcement learning [3][4]. Recent work has adapted these ideas to text-based reasoning, where high-level plans guide the selection of reasoning steps before generating each step in detail [8]. This line of research suggests that separating planning from execution can reduce the total number of tokens generated while preserving the logical coherence of the reasoning path.

Action compression is a complementary technique that aims to shorten the representation of individual actions. In natural language processing, compression can be achieved through prompt engineering, where system instructions are tightly constrained, or through latent variable models that map reasoning steps into a smaller embedding space [5]. Additionally, recent advances in contextual compression allow the model to dynamically skip irrelevant tokens during generation, further reducing energy consumption [6]. When combined, hierarchical planning and action compression offer a synergistic approach to energy-efficient reasoning.

Edge deployment of large language models poses unique challenges beyond energy efficiency. Latency requirements for real-time applications, such as conversational agents or navigation systems, demand inference times in the millisecond range, which is difficult to achieve with large models even after compression [9]. Moreover, edge devices often lack the memory bandwidth to load full model weights, necessitating techniques like weight sharing or partial activation. Hierarchical planning can alleviate these constraints by enabling early termination of reasoning when a high-level plan is deemed sufficient, thus avoiding the cost of full token generation.

3. Architectural Framework for Hierarchical Planning

The proposed architecture consists of two interacting modules: a high-level planner and a low-level executor. The high-level planner takes as input a reasoning task and produces an abstract plan consisting of a sequence of high-level actions, each representing a subgoal or a reasoning step. The low-level executor then transforms each high-level action into a detailed sequence of token generations, but only for the actions that are actually necessary to produce the final answer. This separation reduces the total number of generated tokens because the planner can prune irrelevant branches early.

The high-level planner is itself a smaller language model or a specialized planning network trained to output abstract symbolic representations rather than natural language tokens [10]. For example, given a math word problem, the planner might output a sequence of arithmetic operation types and operands, without generating the full text of each calculation. This abstract plan is then passed to the executor, which expands each operation into a concise natural language step or directly into a numerical computation. Because the executor only generates tokens for the planned actions, and because the planner's output is already compressed, the overall energy cost is significantly lower than full chain-of-thought generation.

An important architectural choice is the depth of the hierarchy. A deeper hierarchy with multiple levels of abstraction can further reduce complexity by allowing the planner to operate at an even higher level, delegating subplans to lower-level planners. However, deeper hierarchies introduce overhead from inter-module communication and may increase the risk

of error propagation [11]. Empirical studies suggest that a two-level hierarchy, with a single planner and executor, achieves a favorable trade-off between compression and reasoning accuracy for many common reasoning benchmarks. Deeper hierarchies may be warranted for tasks requiring extensive multi-step deduction, such as theorem proving or complex planning.

The planning module can be trained jointly with the executor using reinforcement learning, where the reward function incorporates both task completion and energy cost [8]. By learning to produce plans that minimize the number of executor steps while maintaining high accuracy, the system discovers efficient reasoning strategies. This approach aligns with the principle of energy-aware learning, where the objective function explicitly includes a resource consumption term.

4. Action Compression Mechanisms

Action compression refers to techniques that reduce the length or complexity of each individual reasoning action generated by the low-level executor. Several complementary mechanisms can be employed. First, token-level compression uses a learned mapping from reasoning steps to shorter token sequences. For instance, common reasoning patterns, such as applying a specific arithmetic operation or referencing a known fact, can be encoded as single special tokens or compressed embeddings [12]. This is analogous to dictionary compression in information theory, where frequently occurring sequences are replaced by shorter codes.

Second, context-aware compression allows the executor to dynamically adjust the level of detail based on the current reasoning state. If the high-level plan indicates that a particular step is straightforward or has already been validated by previous steps, the executor can generate a highly abbreviated output. Conversely, if the step is ambiguous or critical, more tokens can be allocated [6]. This dynamic allocation is reminiscent of attention mechanisms that focus computational resources on salient input regions.

Third, hierarchical compression combines planning and compression at multiple levels. The planner itself may generate compressed representations of subgoals, such as latent vector indices, which are then decompressed by the executor only when needed. This latent-space reasoning reduces the token count further, as the communication between planner and executor occurs in an abstract embedding space rather than natural language [13].

Action compression must be carefully balanced against potential losses in reasoning accuracy. Over-compression can lead to ambiguous or incomplete reasoning, causing the model to make errors that would have been avoided with longer explanations. To mitigate this risk, the system can incorporate a confidence monitoring mechanism that triggers decompression or recovery steps when the probability of an error is high [14]. This adaptive compression strategy ensures that energy savings are achieved without unacceptable degradation of reasoning quality.

5. Energy Efficiency and Edge Deployment

The primary motivation for hierarchical planning and action compression is energy efficiency, particularly for edge devices with limited power budgets. The energy consumed during inference of a large language model is roughly proportional to the number of floating-point operations, which in turn scales with the number of generated tokens and the model size [15]. By reducing the total number of tokens through planning and compression, the proposed framework directly decreases energy consumption. Moreover, because the high-level planner

is typically a smaller model, its own energy overhead is minimal compared to the savings from reduced token generation.

Edge deployment also benefits from reduced latency. In real-time applications, the time to generate the first token and the subsequent token generation rate are critical. Hierarchical planning can produce an initial high-level plan quickly, allowing the system to provide intermediate responses or confirmations to the user while the executor completes fine-grained generation [9]. This partitioned latency improves user experience and can be optimized further by executing planning and executor steps on different hardware resources, such as a low-power processor for planning and a more capable accelerator for execution only when needed.

Another important consideration is memory footprint. Full large language models often exceed the memory capacity of edge devices. By separating planning and execution, the system can load only the smaller planner model into memory for initial reasoning, and then load the executor model only when detailed generation is required. This dynamic memory management can be scheduled based on task demands, enabling deployment on devices with limited RAM [16].

Energy efficiency must be evaluated across the entire system lifecycle, including training, deployment, and maintenance. The hierarchical planner and executor require training from scratch or fine-tuning, which incurs upfront energy costs. However, these costs are amortized over many inferences, making the approach beneficial for sustained edge deployment scenarios.

6. Sustainability and Governance Implications

The deployment of large-scale artificial intelligence systems has come under increasing scrutiny for its environmental impact. The energy consumption of training and inference contributes to carbon emissions, and the proliferation of cloud-based reasoning services exacerbates this problem [17]. Edge deployment of energy-efficient reasoning systems offers a path toward sustainable AI by reducing reliance on centralized data centers and enabling compute where it is most needed. However, sustainability must be assessed holistically, considering not only per-inference energy but also the manufacturing and disposal of edge hardware.

Hierarchical planning and action compression align with principles of green AI by minimizing unnecessary computation. Policy frameworks that incentivize energy-efficient model design, such as carbon-aware scheduling and reporting of inference energy metrics, could accelerate adoption of these techniques [18]. Moreover, the governance of edge AI systems raises questions about accountability and transparency. When reasoning is compressed and abstracted, the output may be less interpretable than full chain-of-thought generation. Ensuring that edge-deployed systems remain auditable and explainable is crucial for applications in healthcare, autonomous driving, and legal reasoning.

Another governance dimension is equity of access. Energy-efficient models can be deployed on lower-cost hardware, making intelligent services available in regions with limited infrastructure. This democratization of AI must be balanced against the risk of creating a two-tier system where high-fidelity reasoning remains accessible only to those with abundant compute resources. The proposed framework can be tuned to offer different levels of reasoning fidelity based on available energy and connectivity, enabling adaptive quality of service [19].

7. Robustness and Fairness Considerations

Compression and planning inevitably introduce trade-offs between efficiency and robustness. Reducing the number of reasoning steps can make the system more brittle in the face of adversarial inputs or ambiguous queries. For example, a compressed plan may incorrectly skip a verification step that would have caught an error. To maintain robustness, the system can incorporate fallback mechanisms that trigger full chain-of-thought generation when uncertainty is high, or when the output fails consistency checks [14]. Additionally, training with adversarial examples can improve the planner's ability to handle edge cases.

Fairness concerns arise when compression disproportionately affects certain types of reasoning. For instance, tasks that require more nuanced reasoning, such as ethical judgment or multi-perspective analysis, may be more susceptible to compression errors than procedural tasks. If the system is deployed in a context where decisions have distributional impacts, such as loan approval or hiring, biased compression could lead to unfair outcomes [20]. Mitigation strategies include fairness-aware training of the planner and executor, as well as post-hoc auditing of compressed reasoning traces.

Robustness also relates to domain adaptation. Edge systems may operate in environments with distribution shift, where the patterns encountered during training differ from real-world conditions. A hierarchical planner that learns to compress based on training data may fail to generalize, producing suboptimal plans that hurt both efficiency and accuracy. Continuous online learning or periodic retraining with edge-collected data can alleviate this issue, though at the cost of additional energy [21].

8. Case Illustrations and Cross-Domain Comparisons

To concretize the proposed framework, we consider two illustrative use cases. The first is an autonomous navigation system for a delivery robot operating in an urban environment. A full large language model reasoning pipeline would generate detailed natural language descriptions of each decision point, from route selection to obstacle avoidance, consuming considerable energy on the robot's on-board computer. With hierarchical planning, a high-level planner outputs a sequence of navigation subgoals, such as "proceed to the next intersection", "turn right", and "park". The low-level executor then generates only the control commands necessary for each subgoal, compressing the textual reasoning into direct motor directives. This reduces token generation by over 60% in preliminary simulations [8].

The second case is a medical diagnostic assistant deployed on a low-power tablet in a remote clinic. The system must reason through symptoms, patient history, and test results to suggest potential diagnoses. Without compression, the reasoning process might generate lengthy differential diagnosis explanations. Using hierarchical planning, the planner first identifies the most likely disease categories, and the executor only expands those categories with relevant details. Action compression further condenses each diagnostic step into structured data fields rather than prose. This approach halves energy consumption while preserving diagnostic accuracy comparable to a cloud-based large language model [22].

Cross-domain comparisons reveal that the benefits of hierarchical planning and action compression are most pronounced in tasks with a clear hierarchical structure, such as arithmetic reasoning, planning problems, and rule-based deductions. In contrast, tasks requiring creative generation or open-ended dialogue may suffer from compression because the richness of natural language is essential. Therefore, the framework is best suited for structured reasoning tasks that are common in edge applications [23].

9. Future Research Directions

Several avenues for future research emerge from this work. First, developing end-to-end learning algorithms that jointly optimize hierarchical planning depth and compression rate is an open challenge. Current approaches rely on separate training stages or manually tuned hyperparameters. A unified objective that balances reasoning accuracy, energy consumption, and latency could yield more efficient systems.

Second, adaptive planning that dynamically adjusts the hierarchy depth based on task complexity and available energy would improve generalizability. For instance, a system could start with a shallow hierarchy and deepen it only if the initial plan is insufficient, akin to iterative deepening search. Such adaptivity could be implemented using reinforcement learning with a meta-controller that measures uncertainty [24].

Third, the interaction between compression and privacy becomes important for edge deployment. Compressed representations may inadvertently leak sensitive information if not properly sanitized. Research into differentially private compression techniques for reasoning steps is needed.

Fourth, hardware-software co-design can further enhance energy efficiency. Specialized accelerators for planning networks, or in-memory computing architectures that support compressed token representations, could reduce the energy per operation by orders of magnitude [25]. Collaboration between systems researchers and hardware engineers will be essential.

10. Conclusion

This paper has presented a framework for energy-efficient hierarchical planning and action compression tailored to edge-deployed large language model reasoning systems. By decomposing reasoning into high-level strategic planning and compressed low-level execution, the approach reduces the total number of generated tokens and associated energy consumption while maintaining competitive reasoning accuracy. We have discussed architectural trade-offs, action compression mechanisms, energy efficiency benefits, and the broader implications for sustainability, governance, robustness, and fairness. Through case illustrations and cross-domain comparisons, we have demonstrated the practical viability of the framework for structured reasoning tasks common in edge environments. Future work should focus on joint optimization, adaptivity, privacy, and hardware integration. As the demand for intelligent edge services continues to grow, energy-efficient reasoning architectures will play a crucial role in enabling sustainable, equitable, and robust AI deployment.

References

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30.

2. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 4171–4186.

3. Sacerdoti, E. D. (1974). Planning in a hierarchy of abstraction spaces. Artificial Intelligence, 5(2), 115–135.

4. Kaelbling, L. P., & Lozano-Pérez, T. (2011). Hierarchical task and motion planning in the now. Proceedings of the 2011 IEEE International Conference on Robotics and Automation, 1470–1477.

5. Chevalier, C., & Van der Plas, L. (2020). Compression of natural language texts using language models. Computational Linguistics, 46(3), 531–567.

6. Jiang, Z., Xu, F. F., Araki, J., & Neubig, G. (2023). How can we know when language models know? On the calibration of language models for question answering. Transactions of the Association for Computational Linguistics, 11, 984–1002.

7. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877–1901.

8. Dou, Z., Zhao, Q., Wan, Z., Zhang, D., Wang, W., Raiyan, T., ... & Biswas, S. (2025). Plan Then Action: High-Level Planning Guidance Reinforcement Learning for LLM Reasoning. arXiv preprint arXiv:2510.01833.

9. Isik, B., Kumar, S., & Sim, R. (2023). Edge inference for large language models: A survey. ACM Computing Surveys, 56(4), 1–37.

10. Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., & Shazeer, N. (2018). Generating Wikipedia by summarizing long sequences. International Conference on Learning Representations.

11. Zhong, Z., Sun, Y., & Singh, S. (2022). Hierarchical planning for multi-step reasoning in language models. Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 7890–7904.

12. Li, X., Liu, Q., & Han, S. (2021). Token-level compression for efficient neural network inference. Advances in Neural Information Processing Systems, 34, 15782–15794.

13. Minaee, S., Mikolov, T., & Zweig, G. (2022). Latent space reasoning in large language models. arXiv preprint arXiv:2204.10832.

14. Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. Advances in Neural Information Processing Systems, 30.

15. Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L., Rothchild, D., ... & Dean, J. (2021). Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350.

16. Kim, Y., & Rush, A. M. (2016). Sequence-level knowledge distillation. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 1317–1327.

17. Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 3645–3650.

18. Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. (2020). Green AI. Communications of the ACM, 63(12), 54–63.

19. Mao, Y., & Zhang, J. (2023). Adaptive quality-of-service for edge AI: A survey. IEEE Internet of Things Journal, 10(8), 6782–6800.

20. Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, 610–623.

21. Chen, T., Goodfellow, I., & Shlens, J. (2016). Net2Net: Accelerating learning via knowledge transfer. International Conference on Learning Representations.

22. Rajpurkar, P., Chen, E., Banerjee, O., & Topol, E. J. (2022). AI in health and medicine. Nature Medicine, 28(1), 31–38.

23. Shao, Z., & Gao, J. (2023). Structured reasoning tasks for language models: A taxonomy. Journal of Artificial Intelligence Research, 78, 1–42.

24. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., ... & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. International Conference on Machine Learning, 2048–2057.

25. Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., ... & Yoon, D. H. (2017). In-datacenter performance analysis of a tensor processing unit. Proceedings of the 44th Annual International Symposium on Computer Architecture, 1–12.