

Zen AI Model Family

Zen4-Thinking

Fourth-Generation Extended Reasoning Model

Technical Whitepaper v2026.02

Zach Kelling*
research@hanzo.ai

Zoo Labs Foundation
foundation@zoolabs.org

February 2026

Abstract

We present **Zen4-Thinking**, a 72B parameter model capable of extended chain-of-thought reasoning with an explicit thinking token budget of up to 32,768 tokens before producing a final response. Zen4-Thinking is trained through a novel **Budgeted Reflection Optimization** (BRO) framework that teaches the model to allocate thinking tokens efficiently across problem difficulty, avoiding both under-thinking on hard problems and over-thinking on easy ones. The model achieves 93.7% on MATH, 87.5% on AIME 2024, 81.3% on GPQA Diamond, 72.4% on USAMO, and 88.2% on LiveCodeBench — establishing state-of-the-art results for the 72B dense parameter class on competition-level reasoning benchmarks. Zen4-Thinking demonstrates that extended thinking, implemented correctly, provides qualitative capability gains that cannot be matched by simply scaling parameters or data at standard inference budgets.

Contents

1	Introduction	3
1.1	Key Contributions	3
1.2	Thinking vs. Standard Mode	3
2	Architecture	4
2.1	Model Specifications	4
2.2	Thinking Token Architecture	4
2.3	Relation to Zen4-Pro	4
3	Training	5
3.1	Pretraining	5
3.2	Budgeted Reflection Optimization	5
3.2.1	Phase 1: Cold-Start SFT	5
3.2.2	Phase 2: Budget-Aware GRPO	5
3.2.3	Phase 3: Calibration Fine-Tuning	6
3.3	Thinking-Aware Quantization	6

*zach@lux.network

4	Evaluation	6
4.1	Competition Mathematics	6
4.2	Science and Formal Reasoning	6
4.3	Thinking Budget Utilization	6
4.4	Thinking Latency	7
4.5	Comparison Across Zen4 Family	7
5	Related Work	7
6	Deployment	8
6.1	Thinking Mode Configuration	8
6.2	Hardware Requirements	8
6.3	Recommended Use Cases	9
7	Safety and Alignment	9
7.1	Limitations	9
8	Conclusion	10
A	Model Card	11
B	BRO Ablation Study	11
C	Sample Thinking Token Output	11

1 Introduction

Mathematical competition problems and advanced scientific reasoning tasks share a structural property that makes them resistant to standard language model approaches: the correct answer depends not on memorized associations but on a sequence of inferential steps, each of which must be logically valid for the final answer to be correct. A model that produces the right answer via a flawed reasoning chain is brittle; it will fail when the surface features of the problem change while the underlying structure remains.

Extended thinking addresses this by giving the model an explicit budget of tokens to reason through the problem before committing to an answer. The thinking tokens are not shown to the user; they serve as a scratchpad where the model can:

- Explore multiple solution approaches before committing to one.
- Verify intermediate steps by re-deriving them from different angles.
- Identify and correct errors in prior reasoning.
- Enumerate cases systematically rather than guessing at coverage.

Prior work on chain-of-thought prompting [4] established that interleaving reasoning with answers improves performance on multi-step tasks. Extended thinking scales this insight: rather than a few hundred tokens of reasoning, Zen4-Thinking supports up to 32,768 thinking tokens. This qualitative increase in reasoning budget enables solution strategies that are simply not expressible in shorter reasoning chains.

1.1 Key Contributions

- **Budgeted Reflection Optimization (BRO):** A new post-training framework that teaches the model to allocate thinking budget proportionally to problem difficulty, measured by the outcome variance across rollouts at a given budget.
- **32K Thinking Token Support:** Native architectural support for thinking tokens up to 32,768 with no penalty to standard generation quality.
- **Thinking-Aware Quantization:** A quantization scheme that preserves higher precision in the layers most heavily used during thinking token generation.
- **State-of-the-Art Competition Math:** 87.5% on AIME 2024, 72.4% on USAMO averaged across 2024–2025 competition sets.

1.2 Thinking vs. Standard Mode

Zen4-Thinking operates in two modes:

Standard mode The model generates a direct response without thinking tokens, equivalent to Zen4-Pro behavior. This mode is used when thinking is disabled by the caller or when the model determines the query does not require extended reasoning.

Thinking mode The model generates a `<think>...</think>` block containing up to 32K tokens of internal reasoning before generating the final response. The thinking block is generated autoregressively and is subject to the same KV-cache and context window constraints as regular tokens.

The model selects between modes based on the input query when `thinking=auto`, or the caller can enforce `thinking=enabled` or `thinking=disabled`.

2 Architecture

2.1 Model Specifications

Component	Specification
Total Parameters	72.7B
Active Parameters	72.7B (dense)
Architecture	Hybrid Transformer
Attention Mechanism	Hierarchical Hybrid Attention (HHA)
Local Window Size	8,192 tokens
Global Anchor Tokens	1,024 per layer
Context Length (total)	131,072 tokens (128K)
Max Thinking Tokens	32,768
Max Response Tokens	98,304 (128K minus thinking)
Hidden Dimension	8,192
Intermediate Dimension	29,568
Attention Heads	64
Key-Value Heads	8 (GQA)
Layers	80
Vocabulary Size	151,936
Thinking Token IDs	Special tokens: <code><think></code> (ID 151668), <code></think></code> (ID 151669)
Positional Encoding	RoPE ($\theta = 10,000,000$)
Normalization	RMSNorm
Activation	SwiGLU

Table 1: Zen4-Thinking Architecture Specifications

2.2 Thinking Token Architecture

Thinking tokens are native to the Zen4-Thinking architecture from the beginning of training. Two special tokens, `<think>` and `</think>`, are added to the vocabulary. During generation with thinking enabled, the model generates `<think>` followed by a variable-length chain of reasoning, then `</think>`, and finally the user-visible response.

Critically, thinking tokens are treated as fully ordinary tokens by the attention mechanism. The HHA local and global attention applies identically to thinking tokens, reasoning tokens, and response tokens. This means:

- Reasoning steps in the thinking block can attend to all prior thinking steps.
- The response generation can attend to the entire thinking block, ensuring the reasoning is fully available during answer synthesis.
- The KV-cache for thinking tokens is held in memory and can be inspected for debugging; the cache can optionally be offloaded if response-only latency is more critical than thinking quality.

No architectural changes to the transformer layers were required to support extended thinking; the entire innovation is in the training methodology.

2.3 Relation to Zen4-Pro

Zen4-Thinking shares the base model architecture with Zen4-Pro (72B dense, HHA, same layer count and dimensions). The distinction is entirely in post-training: Zen4-Thinking’s post-training is optimized for reasoning quality through BRO, while Zen4-Pro’s is optimized for

agentic task success through trajectory-based GRPO. The two models represent different optimization targets for the same architectural foundation.

3 Training

3.1 Pretraining

Zen4-Thinking uses the same pretraining checkpoint as Zen4-Pro (20T tokens), with a higher proportion of mathematical and scientific content (see Zen4-Pro whitepaper). The extended thinking capability is entirely post-training; the pretraining checkpoint does not include thinking token training.

3.2 Budgeted Reflection Optimization

BRO is a three-phase post-training procedure:

3.2.1 Phase 1: Cold-Start SFT

A cold-start SFT phase teaches the model the format of thinking tokens and basic extended reasoning before RL training. The cold-start corpus consists of:

- 50K mathematics problems with manually written thinking-block solutions (average 3,200 tokens per thinking block).
- 20K competition problems with expert solutions reformatted into the `<think>...</think>` format.
- 15K code problems with step-by-step algorithm development in the thinking block followed by clean implementation in the response block.

After cold-start SFT, the model can produce syntactically correct thinking blocks but has not yet learned to allocate budget proportionally to difficulty.

3.2.2 Phase 2: Budget-Aware GRPO

Standard GRPO [9] generates G rollouts per problem and computes relative rewards within the group. BRO extends this with a budget-aware reward formulation.

For a problem p with difficulty level d_p (estimated by outcome variance at a fixed budget of 2K thinking tokens across 16 rollouts), the BRO reward is:

$$R_{\text{BRO}}(o, p) = R_{\text{outcome}}(o, p) - \lambda \cdot \max \left(0, \frac{T_{\text{think}}(o)}{B^*(p)} - 1 \right) \quad (1)$$

where $T_{\text{think}}(o)$ is the number of thinking tokens in rollout o , $B^*(p)$ is the problem-specific budget target (a function of d_p), and $\lambda = 0.1$ is a penalty coefficient. This reward encourages the model to:

- Use sufficient thinking budget to solve the problem (dominated by R_{outcome}).
- Not exceed the problem-appropriate budget (penalized by the second term).

The budget target function $B^*(p)$ is learned from the warm-up phase data:

$$B^*(p) = \alpha_0 + \alpha_1 \cdot d_p + \alpha_2 \cdot d_p^2 \quad (2)$$

with $\alpha_0 = 512, \alpha_1 = 1024, \alpha_2 = 128$ fitted from warm-up observations.

3.2.3 Phase 3: Calibration Fine-Tuning

After BRO, a final calibration SFT pass on 100K mixed-difficulty problems ensures that the model’s auto-selected thinking budget matches the BRO-learned budget targets. Without this calibration, the model sometimes over-thinks at inference time despite being penalized during training.

3.3 Thinking-Aware Quantization

During thinking token generation, certain model layers exhibit significantly higher activation variance than during standard generation. We identify these **thinking-critical layers** by comparing activation variance statistics on thinking versus standard generation tasks across 10K samples.

In INT8 quantization, thinking-critical layers are quantized using a finer granularity (per-channel rather than per-tensor), preserving numerical precision where it matters most. This approach retains 99.1% of thinking-mode benchmark performance versus BF16, compared to 96.3% with standard uniform INT8 quantization.

4 Evaluation

4.1 Competition Mathematics

Benchmark	Zen4-Pro (standard)	Zen4-Thinking	Δ
MATH Level 5 only	79.4%	93.7%	+14.3%
AIME 2024 (avg)	64.3%	87.5%	+23.2%
AIME 2025 (avg)	61.8%	84.2%	+22.4%
USAMO (avg ’24-’25)	48.3%	72.4%	+24.1%
AMC 10 (perfect score)	41.2%	68.7%	+27.5%
OlympiadBench	62.4%	78.9%	+16.5%

Table 2: Competition Mathematics: Standard vs. Extended Thinking

4.2 Science and Formal Reasoning

Benchmark	Zen4-Pro	Zen4-Thinking
GPQA Diamond (0-shot)	74.3%	81.3%
GPQA Diamond (w/ thinking)	—	84.7%
SciBench	80.1%	87.4%
ProofWriter (hard)	73.4%	88.2%
FOLIO (first-order logic)	80.2%	89.1%
LiveCodeBench (competition)	72.3%	88.2%

Table 3: Science and Formal Reasoning Benchmarks

4.3 Thinking Budget Utilization

A key goal of BRO is efficient budget allocation. The following table shows average thinking token usage by problem category:

Problem Category	Avg Thinking Tokens	Max Allowed
Simple arithmetic (GSM8K easy)	128	32,768
Math word problems (GSM8K hard)	512	32,768
MATH Level 3–4	1,840	32,768
MATH Level 5	4,210	32,768
AIME problems	8,730	32,768
USAMO problems	18,400	32,768
LiveCodeBench (hard)	6,340	32,768

Table 4: Average Thinking Token Usage by Problem Category

The model allocates thinking budget proportionally to problem difficulty, as intended by BRO. Simple arithmetic uses $<0.4\%$ of the maximum budget; USAMO problems use 56% of the maximum. This proportionality means Zen4-Thinking is cost-efficient on easy queries while still having headroom for the hardest problems.

4.4 Thinking Latency

Problem Type	Avg Latency (H100)	Avg Latency (A100)
Simple (128 think tokens)	0.8s	1.4s
Medium (1,840 think tokens)	5.2s	9.1s
Hard (8,730 think tokens)	24.3s	42.6s
Maximum (32,768 think tokens)	91.2s	159.8s

Table 5: End-to-End Latency by Thinking Budget (including response generation)

4.5 Comparison Across Zen4 Family

Benchmark	Zen4	Zen4-Pro	Zen4-Max	Zen4-Thinking
MATH (Level 5)	72.1%	79.4%	87.3%	93.7%
AIME 2024	48.7%	64.3%	81.4%	87.5%
GPQA Diamond	67.4%	74.3%	79.8%	84.7%
LiveCodeBench	64.3%	72.3%	78.4%	88.2%

Table 6: Competition-Level Benchmarks Across the Zen4 Family

Zen4-Thinking provides the highest scores on competition-level tasks within the 72B parameter class, exceeding even Zen4-Max (480B/48B active) on MATH Level 5 and LiveCodeBench competition tasks. This confirms that extended thinking provides qualitatively different capability gains that cannot be replicated by parameter scaling at standard inference budgets.

5 Related Work

Chain-of-thought prompting [4] established that intermediate reasoning steps improve accuracy on multi-step tasks. Self-consistency [5] further improved results by sampling multiple reasoning chains and taking the majority answer. Process reward models [6] provided step-level feedback, enabling training directly on reasoning quality rather than final answer correctness.

Extended thinking has been explored under different framings. Scratchpad methods [7] demonstrated early that giving models space to work through problems helps. More recent

work on inference-time compute scaling [8] has shown that allocating more compute at inference time (through sampling, search, or extended generation) can substitute for additional training compute, representing a new axis of model capability.

BRO’s budget-proportional reward extends prior work on length penalties in RLHF [11] and thinking-budget supervision from curriculum learning, applying them jointly in the GRPO setting for the first time.

6 Deployment

6.1 Thinking Mode Configuration

```

1 from hanzo import HanzoAI
2
3 client = HanzoAI()
4
5 # Auto-select thinking mode based on query complexity
6 response = client.chat.completions.create(
7     model="zen4-thinking",
8     messages=[
9         {"role": "user", "content": "Find all integer solutions to  $x^3 + y^3 = z^3$ ."}
10    ],
11    extra_body={
12        "thinking": "auto",          # "enabled", "disabled", or "auto"
13        "thinking_budget": 16384,    # max thinking tokens (default: 8192)
14        "show_thinking": False      # include <think> block in response
15    }
16 )
17
18 # Access the thinking content if show_thinking=True
19 if response.choices[0].message.thinking:
20     print("Reasoning:", response.choices[0].message.thinking[:200], "
21         ...)
22 print("Answer:", response.choices[0].message.content)

```

Listing 1: Zen4-Thinking Extended Reasoning

6.2 Hardware Requirements

Configuration	Hardware	Notes
Full BF16, 128K context	H100 $\times$ 2	Max context
FP8, 128K context	H100 $\times$ 1	Recommended
Q4_K_M, 32K context	A100 80GB	Budget inference

Table 7: Zen4-Thinking Hardware Requirements

Zen4-Thinking’s memory requirements during thinking generation are higher than Zen4-Pro’s at equivalent context lengths, because the KV-cache grows as thinking tokens are generated. At 32K thinking tokens in BF16, the KV-cache adds approximately 12 GB beyond the base model memory requirement.

6.3 Recommended Use Cases

Zen4-Thinking is specifically recommended for:

- Competition mathematics and mathematical proof verification.
- Scientific hypothesis evaluation requiring multi-step chain of evidence.
- Complex algorithm design and competitive programming.
- Logic puzzles and formal verification tasks.
- Any task where the asker can verify the answer independently and cares more about correctness than latency.

Zen4-Thinking is not recommended for:

- Low-latency conversational applications (use Zen4 or Zen4-Mini).
- High-throughput summarization or extraction tasks (the thinking overhead provides no benefit on tasks that do not require multi-step reasoning).
- Interactive agentic workflows requiring fast tool-call turnaround (use Zen4-Pro).

7 Safety and Alignment

Safety Metric	Standard Mode	Thinking Mode
Harmful content refusal	98.2%	97.8%
Over-refusal rate	3.7%	4.1%
Jailbreak resistance	97.4%	96.9%
TruthfulQA	83.1%	87.4%

Table 8: Safety Metrics: Standard vs. Thinking Mode

Thinking mode shows slightly lower jailbreak resistance (97.4% vs 96.9%) because extended reasoning chains provide more surface area for adversarial manipulation to redirect the model’s conclusions. Safety training includes adversarial thinking-mode examples where the thinking block contains manipulative intermediate steps; the model is trained to recognize and ignore these patterns in the final response generation.

Thinking mode shows improved TruthfulQA scores (83.1% to 87.4%) because extended reasoning allows the model to fact-check its own intermediate claims before committing.

7.1 Limitations

- USAMO performance (72.4%) indicates that competition-level proof problems remain partially out of reach; the model solves roughly 3 of 4 problems it encounters but with non-trivial error rate.
- Thinking token content is not guaranteed to be logically valid; the thinking block is a learned reasoning format, not a verified proof system. Users should verify mathematical claims independently.
- Long thinking chains (16K+) occasionally exhibit **reasoning drift**: the model begins exploring a correct approach but gradually transitions to a different, incorrect approach without recognizing the switch. This is flagged in the thinking block and is an active area of research.

8 Conclusion

Zen4-Thinking establishes that extended chain-of-thought reasoning, implemented through a principled training methodology, provides capability gains on competition-level reasoning tasks that significantly exceed what is achievable through parameter scaling at standard inference budgets. Budgeted Reflection Optimization addresses the key limitation of prior extended-thinking approaches — indiscriminate compute allocation — by teaching the model to match its reasoning effort to problem difficulty. With 87.5% on AIME 2024, 72.4% on USAMO, and 81.3% on GPQA Diamond, Zen4-Thinking is the recommended choice for any application where deep, verified reasoning quality is the primary objective.

Acknowledgments

We thank the mathematical olympiad community whose publicly available problem sets and solutions formed the foundation of the cold-start SFT corpus, the Zoo Labs Foundation formal reasoning research team, and the Hanzo AI post-training infrastructure team.

References

- [1] Vaswani, A. et al. (2017). Attention Is All You Need. NeurIPS 2017.
- [2] Brown, T. et al. (2020). Language Models are Few-Shot Learners. NeurIPS 2020.
- [3] Ouyang, L. et al. (2022). Training Language Models to Follow Instructions with Human Feedback. NeurIPS 2022.
- [4] Wei, J. et al. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. NeurIPS 2022.
- [5] Wang, X. et al. (2022). Self-Consistency Improves Chain of Thought Reasoning in Language Models. ICLR 2023.
- [6] Lightman, H. et al. (2023). Let’s Verify Step by Step. arXiv:2305.20050.
- [7] Nye, M. et al. (2021). Show Your Work: Scratchpads for Intermediate Computation with Language Models. arXiv:2112.00114.
- [8] Snell, C. et al. (2024). Scaling LLM Test-Time Compute Optimally Can Be More Effective Than Scaling Model Parameters. arXiv:2408.03314.
- [9] Shao, Z. et al. (2024). DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300.
- [10] Schulman, J. et al. (2017). Proximal Policy Optimization Algorithms. arXiv:1707.06347.
- [11] Singhal, K. et al. (2023). A Long Way to Go: Investigating Length Correlations in RLHF. arXiv:2310.03716.
- [12] Chowdhery, A. et al. (2023). PaLM: Scaling Language Modeling with Pathways. JMLR 2023.

A Model Card

Field	Value
Model Name	Zen4-Thinking
Version	v2026.02
Release Date	February 2026
Parameters	72.7B
Context Length	131,072 tokens
Max Thinking Tokens	32,768
License	Apache 2.0
Repository	huggingface.co/zenlm/zen4-thinking-72b
Documentation	papers.zenlm.org/zen4-thinking
Contact	research@hanzo.ai

Table 9: Zen4-Thinking Model Card

B BRO Ablation Study

Training Configuration	MATH L5	AIME 2024	Avg Budget
Standard SFT + unconstrained GRPO	84.7%	71.3%	14,200 tokens
+ Cold-start thinking SFT	88.4%	78.9%	11,800 tokens
+ Budget-aware reward (Phase 2)	92.1%	85.7%	5,340 tokens
+ Calibration fine-tuning (Phase 3)	93.7%	87.5%	4,890 tokens

Table 10: BRO Ablation: Each Phase Contribution

The budget-aware reward (Phase 2) provides the largest benchmark gain while simultaneously reducing average thinking token usage from 11,800 to 5,340 — a 55% reduction in thinking compute. This confirms the core hypothesis of BRO: proportional allocation is more efficient than unconstrained allocation, and the model can learn to discriminate hard from easy problems reliably.

C Sample Thinking Token Output

The following is an abbreviated example of Zen4-Thinking’s internal reasoning on AIME 2024 Problem 1 (with the thinking block truncated for space):

```
1 <think>
2 Let me read the problem carefully.
3 [Problem statement parsed...]
4
5 Let me try a substitution approach first.
6 Define x = ... [first approach developed ~1200 tokens]
7
8 That approach gives a system that doesn't close nicely. Let me try
9 a different angle using modular arithmetic.
10
11 Note that the equation must hold mod p for all primes p dividing ...
12 [modular approach developed ~800 tokens]
13
14 The constraint narrows candidates to {2, 5, 13}. Checking each:
```

```
15 x=2: [check] ... valid.
16 x=5: [check] ... does not satisfy the original equation.
17 x=13: [check] ... valid.
18
19 So the answer is  $2 + 13 = 15$ .
20 Let me verify by substituting back: [verification ~200 tokens]
21 Confirmed. Answer: 15.
22 </think>
23
24 The answer is  $\boxed{15}$ .
```

Listing 2: Abbreviated Thinking Block Example (AIME 2024 P1)