llama-modchip v3.1.1: Three-Tier Donor-Free MaxQ Compression, Zero-Collision Multi-ISA Kernels, and RouteCrack 2.0 L1 Micro-Arbiter Speculative Decoding

Paolo Colombo
Independent Systems & Machine Learning Researcher
synchopate@gmail.com · https://github.com/synchopate/llama-modchip

November 2026

Abstract

Running 30B–284B Sparse Mixture-of-Experts (MoE) language models on commodity 16–64 GB CPU workstations is constrained by Per-Layer Embedding (PLE) capacity, DDR5 memory bandwidth, and speculative decoding overhead. Standard 2-bit quantizers either exceed 32–64 GB RAM budgets or rely on lossy expert pruning and donor weight copying, while multi-step neural draft loops burn milliseconds on large-vocabulary `lm_head` projections or over-draft high-entropy text on bandwidth-bound MoE backends.

We present `llama-modchip v3.1.1`, an open-source inference architecture uniting three co-designed layers: (1) a Three-Tier Donor-Free `MaxQ` Fleet spanning 64 GB (`Qwen3.8 MaxQ-64GB`, 53.99 GiB, blind `wiki-alt.txt` PPL 3.4325, −46.6% KLD vs. a 54.40 GiB coder-pruned `IQ1_M` baseline, cold-cache Cgroup v2 certified at 48.35 GiB ≤ 60 GiB with `pgsteal=0`, `Swap=0`; `DeepSeek-V4 284B MaxQ-v4`, 57.99 GiB, 6.8592 PPL, ≈ 60.5 GiB working set), 32 GB (`Qwen3.8 MaxQ-32GB`, 24.98 GiB trunk + 10.73 GiB lazy NVMe `PCA-Wiener+RaBitQ Q1_0` PLE, cold-cache Cgroup v2 certified at 28.26 GiB ≤ 28.5 GiB, `pgsteal=0`, > 10.5 GiB PLE unfaulted), and 16 GB (`Nemotron-3.5-30B MaxQ-16GB`, 13.39 GiB, $N = 1$ fitting ≈15 GiB; plus `VQ3-64` 13.30 GiB and `Fractal-64` 14.36 GiB); (2) a Zero-Collision `ggml` ABI preserving upstream types 41–43 with `FRACTAL_64` (3.75 bpw) at 44 and `VQ3_64` (3.00 bpw) at 45 across AVX-512, AVX2, NEON, and Metal plus zero-copy 3D MoE `--repack`; and (3) `RouteCrack 2.0` (`--spec-type routecrack2`), fusing NextN MTP/`RCPredictor`, O(1) Two-Zone 40K Suffix bypass (< 250 ns), and 192 KB Stockfish-PVS Zobrist memory under an 832 B L1-resident `[10→16→1]` SIMD Micro-Arbiter. On 16 Zen 5 cores (0 GPU), `RouteCrack 2.0` reaches 15.34 tok/s uncapped (15.64±0.05 in cold 60 GiB cgroup; 21.38–22.00 `W1`) on aligned `Qwen3.8-MaxQ-64GB`, 7.79 (12.31 `W1`) on `DeepSeek-V4 MaxQ-v4`, and 28.20 (42.18 `W1`) on `Nemotron-30B` (`Fractal-MXFP4` ref).

1 Introduction

Sparse Mixture-of-Experts (MoE) architectures paired with hybrid linear-attention, state-space, or latent-attention compression decouple total parameter capacity from per-token active FLOPs. Three open-weights model families span the current local deployment spectrum:

1. `Qwen3.8-Flash-Next` [1] (176.5B total, 5B active): A 48-layer hybrid architecture combining 36 `Gated DeltaNet` (GDN) linear-attention layers, 12 Grouped-Query Attention (GQA) layers, 512 routed experts per layer, a 320M-row Per-Layer Embedding (PLE) table, a 248K-token vocabulary, and a built-in `blk.48` 1-layer NextN Multi-Token Prediction (MTP) module [6].

2. `DeepSeek-V4-Flash-0731` [2, 3] (284.33B total, 13.17B active): A 43-layer MoE Transformer combining 64-head Multi-Head Latent Attention (MLA, compressed KV latent rank 512, query LoRA rank 1024, 8 output head-groups of rank 1024), sparse `CSA/LID/HCA` indexer compressors, 4-stream Sinkhorn-20 Hyper-Connections, and 256 routed experts (top-6 active plus 1 shared expert, with deterministic hash routing `tid2eid` on layers 0..2).

3. `NVIDIA-Nemotron-3.5-Lightning-30B-A3B` [4, 5] (30.2B total, 3.0B active): A 52-layer hybrid architecture comprising 23 Mamba-2 State-Space Model (SSM) layers, 23 Sparse MoE layers (128 routed experts, top-6 active + 1 shared expert), and 6 GQA layers.

Running all three families cleanly inside 16 GB, 32 GB, and 64 GB commodity workstations on top of `llama.cpp` [8] exposes four fundamental engineering walls:

- **The Donor-Free Capacity Wall**: Across 18 third-party `Qwen3.8-Flash-Next` GGUF checkpoints published on Hugging Face (54.40–329.72 GiB), 0/18 fit inside a 28.5 GiB budget because leaving the 320M-row PLE table in `IQ4_NL` consumes 26.82 GiB alone,

leaving < 1.7 GiB for the 48-layer MoE trunk; similarly, uncalibrated 2-bit `DeepSeek-V4` builds either exceed 64 GiB or degrade severely.

- **ABI Fragmentation Across Quantization Formats**: Repurposing upstream `ggml` type enum slots breaks binary compatibility whenever upstream introduces standard formats such as `GGML_TYPE_Q1_0` (41), `GGML_TYPE_Q2_0` (42), or `GGML_TYPE_F8_E4M3` (43).

- **CPU-MoE Speculative Cost Asymmetry**: On 1-layer NextN MTP models (`Qwen3.8`), chaining sequential `draft-mtp` steps incurs $3\times\sim$15 ms of 248K `lm_head` compute without suffix memory; conversely, on 284B non-MTP MoE models, drafting unverified prose tokens streams additional routed experts from DDR5 at $+0.395\times T_1$ latency per step.

- **3D MoE Repack Memory Blowup & Barrier Convoys**: Default heap repacking on 3D MoE tensors (`ne[2] >= 64`) destroys zero-copy `mmap` sharing (inflating `RssAnon` 5.4× to 9.81 GiB), while oversubscribed `-t N --poll 50` threads induce OS barrier convoys.

Figure 1 summarizes our `v3.1.1` co-design. All modifications ship in a single unified patch (`patches/llama-modchip-v3.1.patch`) pinned to `llama.cpp @ de32b268c` (build `12416`), compiling across Linux x86-64/ARM64, macOS Apple Metal, Windows 11 (MinGW/MSVC/CUDA), and Vulkan.

2 Related Work

Post-Training & Vector Quantization. Hessian-guided post-training quantization (GPTQ [9]), activation-aware scaling (AWQ [10]), and sensitivity-based dense-and-sparse non-uniform quantization (SqueezeLLM [13]) achieve strong 3–4 bit dense fidelity, while lattice and multi-codebook vector quantizers such as QuIP# [11] and AQLM [12] push toward 2 bits via incoherence processing or additive lookup tables. `MaxQ` adapts these ideas to 256–512-expert MoE blocks on CPU by (i) solving routed per-expert Hessians in FP64 via the Woodbury dual identity when $N_C \ll d_{\text{in}}$, (ii) pairing `PCA-Wiener` projection with `RaBitQ` [7] 1-bit quantization for massive PLE tables, and (iii) storing a 384 B 3D spherical Lloyd-Max codebook (`VQ3_64`) entirely inside six AVX-512 `ZMM` registers decoded via `VPERMI2B`.

Local CPU/Hybrid Inference Systems. IO-aware exact GPU attention tiling (FlashAttention-2 [14]) minimizes HBM traffic on GPUs, whereas consumer CPU/hybrid engines such as `llama.cpp` [8], PowerInfer [15] (which exploits hot/cold ReLU neuron locality across GPU and CPU), and KTransformers [16] (heterogeneous GPU attention + CPU MoE offloading) target memory-constrained workstations. `llama-modchip` builds directly atop `llama.cpp` to eliminate GPU dependence entirely where needed, adding register-stationary AVX-512 `MQA8` attention, zero-copy 3D MoE `--repack`, inward 2 MiB THP alignment for lazy NVMe PLE paging, and a zero-collision `ggml` ABI.

Speculative Decoding. Standard speculative decoding [17] and learned multi-head/feature drafters such as Medusa [18], EAGLE [19], and native NextN MTP [6] target GPU regimes where verifying k draft tokens adds negligible latency. Draftless and retrieval-based methods—including Lookahead Decoding [20], REST [21], Prompt Lookup Decoding [22], and upstream `llama.cpp ngram-map-k` / `draft-mtp` [8]—avoid a separate draft model but use fixed horizons or separate dispatch paths. `Route-Crack 2.0` explicitly models the non-zero CPU-MoE verification cost $\Delta C \in [0.285, 0.395]\,T_1$, unifying suffix lookup, single-pass MTP/`RCPredictor` anchoring, and Stockfish-style transposition tables [23] under an 832 B L1d-resident SIMD arbiter.

3 Three-Tier Donor-Free `MaxQ` Compression & Zero-Collision ABI

3.1 Zero-Collision `ggml` Type ABI (`41..45`)

To guarantee simultaneous compatibility with upstream `llama.cpp` formats and custom block-64 CPU/Metal kernels inside one binary, `llama-modchip v3.1.1` preserves all upstream enums untouched and places custom kernels at dedicated non-colliding IDs (Table 1):

- **Upstream Standard Types Preserved**: `GGML_TYPE_Q1_0 = 41` (RaBitQ 1-bit PLE), `GGML_TYPE_Q2_0 = 42` (18 B / 64 W = 2.25 bpw signed-scale), and `GGML_TYPE_F8_E4M3 = 43`.

- **`Fractal-64` (`GGML_TYPE_FRACTAL_64 = 44`, 30 B / 64 W = 3.75 bpw)**: Stores an FP16 super-scale d, eight 4-bit shifted sub-scales $\tilde{m}_s = m_s + 4 \in \{4..19\}$, and 64 3-bit indices into the kurtosis-matched dyadic codebook $\{-14, -8, -4, -1, +1, +4, +8, +14\}$. Achieves **17.75 dB SQNR** (99.15% cosine similarity on raw `BF16` MoE weights, +1.45 dB over `Q3_K`). Sub-scale multiplication fuses into `vpmaddwd` at zero extra instructions, `qh` unpacking uses AVX-512 `vprorvq`, and community-contributed register-tiled Apple Metal shaders (GitHub PR #24) sustain **78.60 tok/s** on M3 Max and **57.65 tok/s** on M1 Max.

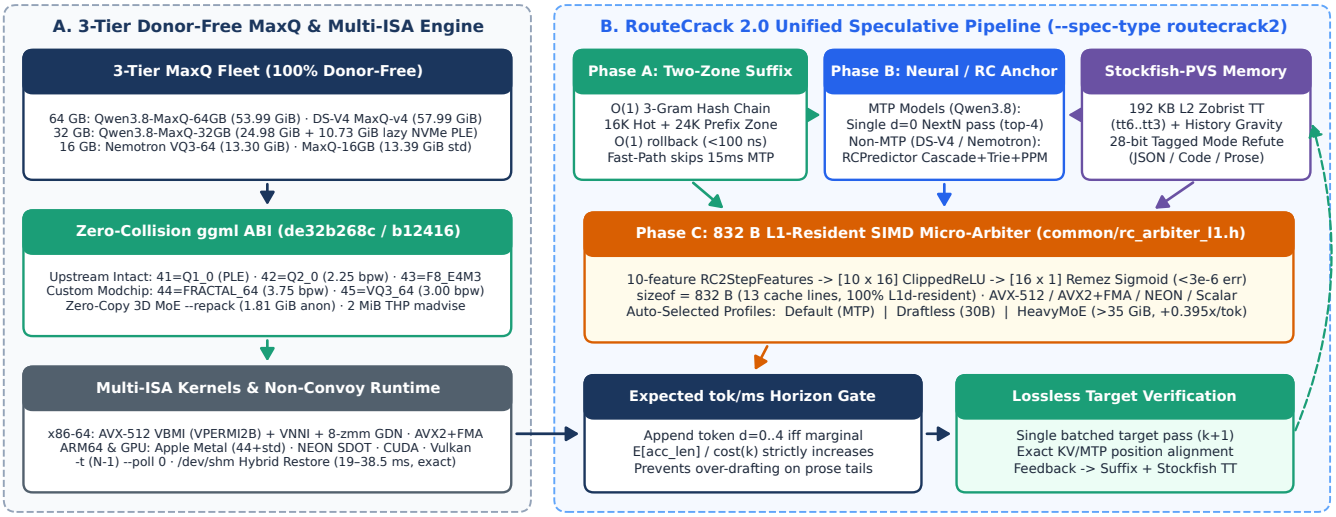


Figure 1: End-to-end `llama-modchip v3.1.1` architecture. (A) Three-tier donor-free `MaxQ` fleet running over a zero-collision `ggml` ABI (`41..45`) and multi-ISA SIMD/Metal kernels with zero-copy 3D MoE `--repack`. (B) Unified `RouteCrack 2.0` speculative pipeline combining O(1) Two-Zone 40K Suffix bypass, Neural MTP/`RCPredictor` anchoring, 192 KB Stockfish-PVS Zobrist memory with 28-bit Tagged Mode Refutation, and an 832 B L1-resident SIMD Micro-Arbiter.

Table 1: Zero-collision `ggml` type enum map in `llama-modchip v3.1.1` (`ggml/include/ggml.h` @ `de32b268c`).

ID	Enum Symbol	B/64W	BPW	Origin / Backends
41	`Q1_0`	10	1.25	Upstream (AVX-512/AVX2/Metal/CUDA)
42	`Q2_0`	18	2.25	Upstream (AVX-512/AVX2/Metal/CUDA)
43	`F8_E4M3`	64	8.00	Upstream FP8 E4M3 (intact)
44	`FRACTAL_64`	30	3.75	Modchip (AVX-512/AVX2/NEON/Metal)
45	`VQ3_64`	24	3.00	Modchip (AVX-512 VBMI/AVX2/NEON)

- **`LUT-VQ3-64` (`GGML_TYPE_VQ3_64 = 45`, 24 B / 64 W = 3.00 bpw)**: Encodes 60 weights as twenty 7-bit indices into a 128-entry 3D spherical Lloyd-Max codebook (384 B held across six AVX-512 `ZMM` registers) plus 4 tail weights and eight 3-bit sub-scales $\tilde{m}_s \in \{2..9\}$. Decoded in-register via three AVX-512 VBMI `VPERMI2B` (`_mm512_mask2_permutex2var_epi8`) lookups (with portable AVX2, ARM64 NEON, and scalar fallbacks), achieving **13.57 dB SQNR** (+3.42 dB over `Q2_K`).

3.2 Three-Tier Donor-Free `MaxQ` Algorithms & Loader Scope

Every `v3.1.1` flagship checkpoint is quantized 100% donor-free directly from upstream `BF16`/`MXFP4` weights with zero donor GGUF weight copying:

1. **64 GB & 32 GB `Qwen3.8-Flash-Next` (`MaxQ-64GB` & `MaxQ-32GB`, std `ggml` types $\leq$ 42)**: Projecting the 320M-row PLE table with **`PCA-Wiener`** ($D = 256$) + **`RaBitQ Q1_0`** [7] shrinks Shard 2 from 26.82 GiB (`IQ4_NL`) to **10.73 GiB**, reinvesting +16.09 GiB into all 512/512 unpruned routed experts, **FP64 Woodbury DualGram** signed-scale `Q2_0`, and the 0.81 GiB `blk.48` NextN MTP head. `MaxQ-64GB` (53.99 GiB, 0 ConeCollapse bias, 48.09 GiB `VmHWM` / 48.35 GiB cold cgroup peak $\leq$ 60.0 GiB at `-c 32768`, `pgsteal=0`; 45.08 GiB warm-cache gate `VmHWM`, `RssAnon = 1.81 GiB`) uses standard types $\leq$ 42 (pass `--no-repack` on unpatched upstream `llama.cpp`; `llama-modchip v3.1.1` is recommended for zero-copy 3D MoE `--repack` and `RouteCrack 2.0`). `MaxQ-32GB` adds 42-layer **`ConeCollapse`** intermediate width pruning ($K = 640 \rightarrow 512$ with post-quantization `F32` `ffn_down_exps.bias`, requiring the `llama-modchip v3.1.1+` loader), compressing Shard 1 to 24.98 GiB while demand-paging Shard 2 lazily from NVMe via `mmap` (**26.79 GiB** $N = 1$ / **28.26 GiB** `RC2` cold-cache cgroup peak $\leq$ 28.5 GiB at `-c 32768` with `pgsteal=0`, `Swap=0`, and $>$ 10.5 GiB of PLE unfaulted—leaving a tight 0.24 GiB headroom at 32K context; 26.91–27.44 GiB in our warm-cache shorter-context gate).

2. **64 GB `DeepSeek-V4-Flash-0731 MaxQ-v4` (57.99 GiB, 0 custom tensor types)**: Replaces custom MLA/Indexer formats with `0731`-imatrix-weighted standard `ggml` tensor types (446 2D K-quant projections, 9 full-width $K = 2048$ ridge anchor layers, and 34 $K = 1792$ `ConeCollapse` layers with 0.60 μ shrunk residual bias and FP64 Woodbury DualGram $N_C = 384$). Having zero

custom tensor types enables native CPU, Apple Metal, CUDA, and Vulkan kernel execution inside `llama-modchip v3.1.1+`, while its 34 `ffn_down_exps.bias` tensors require `llama-modchip`'s `ConeCollapse` loader. At `-c 32768`, its total working set (≈60.3–60.7 GiB = 57.99 GiB weights +2.25–2.67 GiB anon) exceeds a strict 60.0 GiB cap (requiring ≥ 61 GiB free RAM for zero-refault bare-metal residency); under a 60.0 GiB cgroup (`Swap=0`), it finishes without OOM via continuous ZFS-ARC-backed file reclaim (Section 7).

3. **16 GB `NVIDIA-Nemotron-3.5-30B-A3B` (`VQ3-64` 13.30 GiB, `Fractal-64` 14.36 GiB, & `MaxQ-16GB` 13.39 GiB)**: `VQ3-64` (`types 44/45`) combines `Fractal-64` on boundary/down MoE layers with `LUT-VQ3-64` on 19 interior pre-`ReLU`^{2} up-projections (`wiki-alt` PPL **6.7614**, `WikiText-2` PPL **7.4400** vs. `BF16` **6.9582**), while `MaxQ-16GB` (max type = 42, 0 custom types, 0 ConeCollapse bias) targets unmodified upstream `llama.cpp` (`de32b268c+`) and downstream runners supporting `GGML_TYPE_Q2_0 = 42`; under cold-cache 15.0 GiB Cgroup v2 (`-c 32768`, `Swap=0`), pure $N = 1$ (≈15.03 GiB) nearly fits with 0.2 GiB refault, while `RC2` (≈15.75 GiB) requires ≥ 16 GiB free RAM to avoid file reclaim (Section 7).

Figure 2 illustrates the resulting disk-footprint vs. perplexity Pareto frontier.

4 Mathematical Co-Design: `Woodbury DualGram`, Low-Rank SVD, & L_2-Quadrature MoE Pruning

4.1 FP64 Woodbury DualGram & `ConeCollapse`

When quantizing a weight matrix $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ against $N_C \ll d_{\text{in}}$ routed calibration activations $\mathbf{H}_c \in \mathbb{R}^{N_C \times d_{\text{in}}}$ (e.g., $N_C = 384$, $d_{\text{in}} = 2048$), forming the full $d_{\text{in}} \times d_{\text{in}}$ Hessian $\mathbf{H}_c^\top \mathbf{H}_c + \lambda \mathbf{I}$ is both ill-conditioned and costly across 256–512 experts per layer. Applying the Woodbury matrix identity reduces Hessian inversion to an $N_C \times N_C$ FP64 Cholesky dual solve:

$$(\mathbf{H}_c^\top \mathbf{H}_c + \lambda \mathbf{I})^{-1} = \lambda^{-1}\big(\mathbf{I} - \mathbf{H}_c^\top (\mathbf{H}_c \mathbf{H}_c^\top + \lambda \mathbf{I}_{N_C})^{-1} \mathbf{H}_c\big), \tag{1}$$

maintaining observed dual-Gram condition number $\kappa_2 \leq 86.33$ during `MaxQ-v4` quantization while capturing cross-channel activation covariance. Under `ConeCollapse`, pruning low-energy intermediate channels ($K_{\text{full}} \to K_{\text{red}}$) is paired with a post-quantization shrunk residual centroid bias $\mathbf{b}_{\text{down}} = \gamma\, \mathbb{E}[\mathbf{W}_{\text{down}}\mathbf{h} - \widehat{\mathbf{W}}_{\text{down}}\hat{\mathbf{h}}]$ (`GGML_OP_ADD_ID`), preserving downstream residual mean and variance across all 34–42 collapsed layers.

4.2 Spectral Decay of $\mathbf{W}_{o,b}\mathbf{W}_{o,a}$ & L_2-Quadrature MoE Tail Pruning

In `DeepSeek-V4-Flash`, 31.0% of per-token DRAM traffic (991.6 MB/tok) traverses the unactivated MLA output chain $\mathbf{y}_{\text{attn}} = \sum_{g=1}^{8} \mathbf{W}_{o,b}^{(g)}(\mathbf{W}_{o,a}^{(g)}\mathbf{x}_g)$. Exact SVD reveals rapid spectral decay ($\sigma_0 = 15.398 \to \sigma_{768} = 0.901 \to \sigma_{1023} = 0.048$): rank-768 truncation (−25% DRAM) preserves 99.64% of Frobenius energy and 0.998204 output cosine similarity.

For optional runtime MoE tail pruning on learned-router layers ($l \geq 3$, leaving hash-routed lexical layers $l \in \{0, 1, 2\}$ at full Top-6), let $\mathbf{y}_{\text{FFN}} = \mathbf{e}_{\text{shared}} + \sum_{i=1}^{K} w_i \mathbf{e}_{k_i}$ with non-increasing positive routing weights $w_1 \geq \cdots \geq w_K > 0$:

Proposition 1 (L_2-Quadrature vs. Linear L_1 Rescaling)**.** *Conditioned on routing weights $\mathbf{w} = (w_1, \ldots, w_K)$ with $w_1 \geq \cdots \geq w_K > 0$, assume routed expert outputs $\mathbf{e}_{k_i} \in \mathbb{R}^d$ satisfy $\mathbb{E}[\|\mathbf{e}_{k_i}\|_2^2 \mid \mathbf{w}] = \sigma_e^2 > 0$ and pairwise orthogonality $\mathbb{E}[\langle \mathbf{e}_{k_i}, \mathbf{e}_{k_j} \rangle \mid \mathbf{w}] = 0$ for $i \neq j$, orthogonal to $\mathbf{e}_{shared}$. When truncating to $1 \leq k < K$ experts with multiplier α, preserving the exact expected squared norm entering downstream `RMSNorm` requires L_2-**quadrature rescaling**:*

$$\alpha_{L_2} = \sqrt{\frac{\sum_{i=1}^{K} w_i^2}{\sum_{i=1}^{k} w_i^2}}, \tag{2}$$

whereas linear L_1 rescaling $\alpha_{L_1} = (\sum_{i=1}^{K} w_i)/(\sum_{i=1}^{k} w_i)$ strictly inflates routed-branch variance against $\mathbf{e}_{shared}$ by $\Gamma(\mathbf{w}) = \alpha_{L_1}^2/\alpha_{L_2}^2 > 1$.

Proof. By pairwise orthogonality conditioned on $\mathbf{w}$, $\mathbb{E}[\|\sum_{i=1}^{k}(\alpha w_i)\mathbf{e}_{k_i}\|_2^2 \mid \mathbf{w}] = \sigma_e^2 \alpha^2 \sum_{i=1}^{k} w_i^2$, which equals $\sigma_e^2 \sum_{i=1}^{K} w_i^2$ iff $\alpha = \alpha_{L_2}$. To prove $\alpha_{L_1} > \alpha_{L_2}$ (so $\Gamma(\mathbf{w}) > 1$), define head/tail sums $S_k = \sum_{i=1}^{k} w_i$, $S_{\text{tail}} = \sum_{i=k+1}^{K} w_i > 0$, $Q_k = \sum_{i=1}^{k} w_i^2$, and $Q_{\text{tail}} = \sum_{i=k+1}^{K} w_i^2$. Monotonicity $w_1 \geq \cdots \geq w_k \geq w_{k+1} \geq \cdots \geq w_K > 0$ yields $Q_{\text{tail}} \leq w_k S_{\text{tail}}$ and $Q_k \geq w_k S_k$, so $Q_{\text{tail}} S_k \leq Q_k S_{\text{tail}}$, i.e., $Q_{\text{tail}}/Q_k \leq S_{\text{tail}}/S_k$. Because $S_{\text{tail}}/S_k > 0$:

$$\frac{S_K^2}{S_k^2} = \left(1 + \frac{S_{\text{tail}}}{S_k}\right)^2 > 1 + \frac{2S_{\text{tail}}}{S_k} > 1 + \frac{Q_{\text{tail}}}{Q_k} = \frac{Q_K}{Q_k},$$

so $\alpha_{L_1}^2 > \alpha_{L_2}^2$ and $\Gamma(\mathbf{w}) > 1$. For typical $l \geq 3$ weights at $k = 4$, $\alpha_{L_1} = 1.2295$ ($\Gamma = 1.388$) vs. $\alpha_{L_2} = 1.0435$. □

In an internal 12-probe ablation on `DeepSeek-V4 284B`, switching from α_{L_1} to α_{L_2} at $k = 4$ reduced top-10 next-token KL-divergence by 58.1% (0.0671 → 0.0281 nats), while $k = 5$ (α_{L_2}) incurred 0.0042 nats (+0.42% local ΔPPL). Unpruned Top-6 (DeepSeek/Nemotron) and Top-10 (Qwen3.8) remain strictly default in `v3.1.1`.

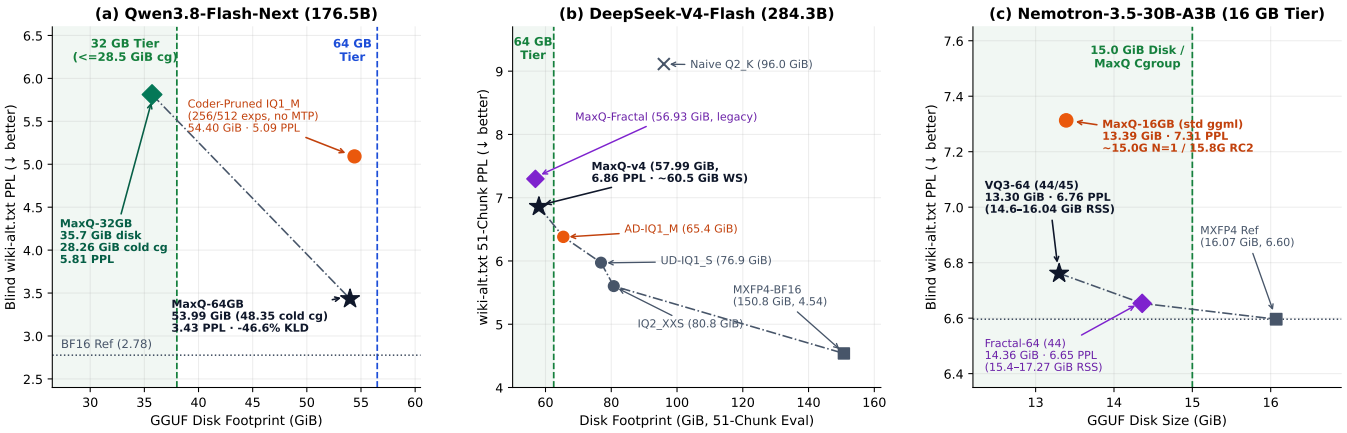


Figure 2: GGUF disk footprint vs. blind `wiki-alt.txt` perplexity Pareto frontier across all three RAM tiers (disk size on x-axis; cold-cache Cgroup v2 / working-set footprints annotated). (a) `Qwen3.8-Flash-Next` (40×512 blind chunks): `MaxQ-64GB` (53.99 GiB disk, 48.35 GiB cold cgroup peak $\leq$ 60 GiB, `pgsteal=0`) beats 54.40 GiB coder-pruned `IQ1_M` (256/512 experts, no MTP) by -1.66 PPL (-46.6% KLD) while `MaxQ-32GB` (35.71 GiB disk, 28.26 GiB cold cgroup peak $\leq$ 28.5 GiB via 42L ConeCollapse + lazy NVMe PLE, `pgsteal=0`) keeps $>$ 10.5 GiB PLE unfaulted. (b) `DeepSeek-V4 284B` (51 chunks): donor-free `MaxQ-v4` (57.99 GiB disk, $\approx$60.5 GiB working set; requires $\geq$ 61 GiB free) reaches 6.86 PPL. (c) `Nemotron-3.5 30B` (40×512): `MaxQ-16GB` $N = 1$ fits $\approx$15 GiB (`RC2` needs $\approx$15.75 GiB), while `VQ3-64` (14.6–16.04 GiB RSS) and `Fractal-64` (15.4–17.27 GiB RSS) target 16–24 GB hosts.

5 `RouteCrack 2.0` & the 832 B L1 Micro-Arbiter

5.1 The CPU-MoE Speculative Horizon Law

Unlike GPU speculative decoding [17] where batched verification costs nearly the same as a single forward step, on memory-bandwidth-bound CPUs hosting Sparse MoE models verifying k drafted tokens incurs non-zero marginal cost $C_{\text{batch}}(k)$ because distinct draft tokens activate additional routed experts from DDR5 ($\Delta C \approx 0.285\,T_1$ on `Qwen3.8`, $0.310\,T_1$ on `Nemotron-30B`, and $0.395\,T_1$ on 284B `DeepSeek-V4`), plus optional neural draft cost $n_{\text{mtp}}(k)\,C_{\text{mtp}}$ ($C_{\text{mtp}} \approx 0.13\,T_1$) and expected rollback cost $C_{\text{rb}}(1 - p_{\text{surv},k})$. Given conditional acceptance probabilities $q_0, \ldots, q_{k-1}$ with survival $p_{\text{surv},k} = \prod_{j=0}^{k-1} q_j$ ($p_{\text{surv},0} = 1$), expected accepted tokens, total cycle cost, and yield rate are (`rc_arbiter_l1.h`):

$$\mathbb{E}[A_k] = 1 + \sum_{d=0}^{k-1} p_{\text{surv},d+1}, \qquad R(k) = \frac{\mathbb{E}[A_k]}{C_{\text{tot}}(k)}, \tag{3}$$

$$C_{\text{tot}}(k) = C_{\text{batch}}(k) + n_{\text{mtp}} C_{\text{mtp}} + \mathbf{1}_{k>0} C_{\text{rb}}(1 - p_{\text{surv},k}), \tag{4}$$

where $C_{\text{batch}}(0) = 1.035$ on MTP profiles (dispatch overhead) and 1.000 on draftless profiles. Letting $\Delta C_d = C_{\text{tot}}(d+1) - C_{\text{tot}}(d)$, `should_append_token` scales only the marginal increment by $\kappa \in [1.0376, 1.0844]$, admitting t_d iff (for $\Delta C_d > 0$):

$$\frac{\mathbb{E}[A_{d+1}]}{C_{\text{tot}}(d) + \kappa\,\Delta C_d} > R(d) \iff \frac{p_{\text{surv},d+1}}{\Delta C_d} > \kappa \cdot R(d). \tag{5}$$

5.2 Unified Three-Phase Speculative Pipeline

Implemented in `common/rc_routecrack2.h` and `common/rc_arbiter_l1.h` (with byte-identical copies in `speculative/cpp/`), `RouteCrack 2.0` (`--spec-type routecrack2`) unifies three candidate generators:

1. **Phase A — Sub-250 ns Fast-Path Two-Zone Suffix Bypass**: Maintains an incremental O(1) 3-gram hash-chain index spanning a **16,384-token Hot Zone** and a **24,576-token Prefix Zone** with O(1) rollback undo ($<$ 100 ns lookup at 64K context). When context consensus satisfies the arbiter's `fastpath_q0_min` gate (0.895 on MTP models) and is unrefuted in Stockfish TT, `RouteCrack 2.0` drafts up to 5 tokens in $<$ 250 ns, bypassing the $\sim$15 ms 248K MTP `lm_head` pass.

2. **Phase B — Neural MTP / `RCPredictor` Anchor + Anchored Extension**: When native MTP is present (`n_layer_nextn > 0`, `Qwen3.8`), executes a *single* $d = 0$ NextN forward pass extracting top-4 probabilities and logit margin, arbitrates `mtp_t0` against Stockfish `tt6..tt3`, and anchors `[prompt..., id_last, mtp_t0]` into the Two-Zone Suffix index and Stockfish TT to extend up to 5 tokens while keeping `ctx_dft` KV-position-aligned across partial acceptances. On non-MTP models (`Nemotron-30B` & `DeepSeek-V4 284B`), fuses `RCPredictor` (8-level FNV-64 Hash Cascade + online `RCTrie`/PPM/Skipgram, requiring `trie_obs >= 2` on prompt-warmed candidates) with Two-Zone Suffix. *Offline table disclosure:* `RCPredictor`'s

static 300 hot rules and 11,382 macro chains are compiled solely for `Nemotron-3.5`'s 131K vocabulary from general public code corpora with zero overlap against evaluation workloads `W1..W4`, and are explicitly disabled via `RC_TABLES_DIR=/nonexistent` on all `Qwen3.8` and `DeepSeek-V4` benchmarks.

3. **Stockfish-PVS 192 KB Zobrist TT & 28-Bit Tagged Mode Refutation**: Inspired by Principal Variation Search transposition tables [23], tracks verified `best_move`, explicit `refuted_move`, and history gravity $\in [-512, +512]$ across three lockless 4-way 64 KB tables (`tt6..tt3`) plus **28-bit Tagged Mode Refutation** isolating JSON, Code, and Prose session modes so syntax-mode transitions never trigger stale refutations.

4. **Phase C — 832 B L1-Resident SIMD Micro-Arbiter (`rc_arbiter_l1.h`)**: Evaluates a 10-feature vector (`mtp_p1`, `mtp_margin`, `suffix_match_len`, `suffix_hit_count`, `zone_is_hot`, `sf_tt_wins`, `sf_tt_history`, `depth_d`, `recent_accept_ema`, `is_structural_tok`) through a cache-line-aligned `[10→16→1]` ClippedReLU MLP with pre-sanitized `NaN`/`±Inf` inputs and a degree-5 Remez minimax `fast_sigmoid` ($< 3 \times 10^{-6}$ max error). Each weight profile occupies **832 B (13 cache lines, L1d-resident)** and executes via unaligned **AVX-512F**, **AVX2+FMA**, **ARM64 NEON**, or scalar C++17 with cross-ISA bit-identity. At load time, the engine selects `kRC2DefaultArbiter` (`n_layer_nextn > 0`), `kRC2DraftlessArbiter` (non-MTP $\leq$ 35 GiB), or `kRC2HeavyMoEDraftlessArbiter` (non-MTP MoE $>$ 35 GiB, plus $n_{\max} = 3$ cap on dense CPU models for single-pass `mr=4` GEMM).

6 Microarchitectural & Multi-Platform Systems Engineering

1. **512-Bit `Q1_0`/`Q2_0`/`MXFP4` Kernels & Multi-Threaded `GATED_DELTA_NET`**: Dedicated AVX-512 VNNI/BW single- and multi-column (`1x8/4/3/2`) kernels accelerate `Q1_0` (41), signed-scale `Q2_0` (42), overflow-free `Q8_0`, and `MXFP4`, while fused 8-`zmm` FMA multi-threaded `GATED_DELTA_NET` removes inner barriers across all 36 `Qwen3.8` linear-attention layers and restores dynamic $4 \cdot n_{\text{th}}$ causal-triangle load balancing on `FLASH_ATTN_EXT` prefill.

2. **Zero-Copy 3D MoE `--repack` & `q8_cache` Aliasing Cure**: Fixes upstream `ggml_compute_forward_mul_mat_tiled` / `mul_mat_id_tiled` `q8_cache` workspace aliasing, clears `last_mmid_ids_data` on non-`MUL_MAT_ID` nodes, and skips heap repacking on 3D MoE tensors (`ne[2] >= 64`) so zero-copy `mmap` is preserved (`Qwen3.8 MaxQ-64GB RssAnon` drops 5.4× from 9.81 GiB to **1.81 GiB**, `batch=1 --repack` vs. `--no-repack` KLD = 0.00000).

3. **Non-Convoy Threading, Inward 2 MiB THP, & Hardened Per-UID IPC**: Pairing `-t (N_CORES - 1) -tb (N_CORES - 1) --poll 0 --poll-batch 0` eliminates OS barrier convoys (recovering +25.9% `pp512` and +31.3% `tg128` over oversubscribed `-t 16 --poll 50` in our internal Stage 4D `32GB none` ablation). `MADV_HUGEPAGE` in `src/llama-mmap.cpp` aligns inward to 2 MiB boundaries under `THP=madvise` so lazy PLE pages are never pre-faulted. All runtime control and HARP-Cache shared-memory files are isolated per-UID (`0700` dir / `0600` mode, `O_NOFOLLOW`, explicit `DSV4_MOE_CFG_PATH`).

4. **`/dev/shm` Hybrid State Snapshots**: Saving and restoring a 512-token `Qwen3.8` hybrid state (233.14 MiB = 36 GDN `[48×128×128]` FP32 + 12 GQA `Q8_0` KV + 1 checkpoint) completes in **38.49 ms** average (6.11 GiB/s vs. 6.18 s cold prefill, ≈162× faster; **19 ms** with `--ctx-checkpoints 0`) with bit-exact greedy continuation.

7 Empirical Evaluation (`v3.1.1`)

Testbed & Artifact Provenance. All local CPU benchmarks execute on a single 16-core AMD Ryzen 9 9955HX (`AMD Ryzen 9 9955HX 16-Core Processor`, 16 Zen 5 physical cores / 32 threads, 64 MiB dual-CCD L3) inside a **74.0 GiB RAM** Linux container (`SwapTotal = 0`, `0 GPU / nvidia-smi` absent, `THP=madvise`, `RC_TABLES_DIR=/nonexistent` on non-Nemotron models). Because host container RAM is 74.0 GiB, all four flagship checkpoints were tested under cold-cache Linux Cgroup v2 (`MemoryAccounting=yes`, `MemorySwapMax=0`, hypervisor `drop_caches=3` + `POSIX_FADV_DONTNEED` before each scope so guest `Cached = 0.3 GiB` at start): `Qwen3.8 MaxQ-64GB` (`60.0 GiB`) and `Qwen3.8 MaxQ-32GB` (`28.5 GiB`) fit strictly below their caps with `pgsteal=0` and `workingset_refault=0`, `Nemotron MaxQ-16GB` $N = 1$ nearly fits `15.0 GiB` (0.2 GiB refault), while `DeepSeek-V4 MaxQ-v4` (`60.0 GiB`) and `MaxQ-16GB RC2` (`15.0 GiB`) exceed those ceilings at `-c 32768` and complete without OOM via continuous host-ZFS-ARC-backed file reclaim (disclosed below). Committed raw logs under `benchmarks/results/` include all three uncapped `RouteCrack 2.0` head-to-head JSONs, all four cold-cache Cgroup v2 3-restart variance JSONs (`qwen38_64gb_cgroup60g_v311.json`, `dsv4_64gb_cgroup60g_v311.json`, `qwen38_32gb_cgroup28g_v311.json`, `nemotron_16gb_cgroup15g_v311.json`), `DeepSeek-V4` 51-chunk logs, and `Nemotron WikiText-2/BF16` KLD logs; `Qwen3.8` `llama-perplexity` summary metrics are recorded from our quantization verification harness.

Table 2: Three-tier blind `wiki-alt.txt` signal fidelity and resident memory footprints (`llama-modchip v3.1.1`). `Qwen3.8` and `Nemotron` evaluated on 40 × 512 blind chunks; `DeepSeek-V4 284B` evaluated on 51 `AtomicChat ref5632.kld` chunks (143,565 scored tokens). `Nemotron` KLD lists both 40 × 512 `wiki-alt` vs. 16.07 GiB `MXFP4` ref and committed 32-probe Step-0 / matched-rollout KLD vs. true 58.84 GiB `BF16` (`kld_nemotron_*_vs_bf16.json`). Peak memory separates process `VmHWM` from Cgroup v2 enforcement limits.

Tier	Checkpoint	Disk	Peak `VmHWM` / Cgroup	Blind PPL ± SE	Mean KLD	Top-1	Loader / ABI
64G	`Qwen3.8 MaxQ-64GB` (512/512, MTP)	**53.99 GiB**	**48.35 GiB** (≤ 60G cg)	**3.4325 ± 0.0708**	**0.4971** (BF16)	**78.31%**	Std (≤ 42, `-nr`)
64G	*`GSQ-RCO-Coder IQ1_M`* (256/512)	54.40 GiB	38.55 GiB (`-nr`)	5.0916 ± 0.1214	0.9302 (BF16)	69.38%	Std (no MTP)
32G	`Qwen3.8 MaxQ-32GB` (24.98 + 10.73 PLE)	**35.71 GiB**	**28.26 GiB** (≤ 28.5G cg)	**5.8120 ± 0.1402**	**1.0069** (BF16)	**66.28%**	Std + `v3.1.1` bias
64G	`DeepSeek-V4 MaxQ-v4` (284B, 34L CC)	**57.99 GiB**	≈60.5 GiB WS[‡](recl.@60G)	**6.8592 ± 0.0482**	**0.6485** (51c)	**73.28%**	Std + `v3.1.1` bias
64G	`DeepSeek-V4 Fractal` (v2.5.3, retired)	56.93 GiB	58.13 GiB (74G host)	7.2977 ± 0.0504	0.7400 (51c)	70.40%	Legacy custom
16–24G	`Nemotron-3.5 VQ3-64` (CPU-Turbo)	**13.30 GiB**	14.6–16.04 GiB (RSS)	**6.7614** (WT2 7.44)	0.0871 / **0.1495**[†]	—	Modchip 44/45
16–24G	`Nemotron-3.5 Fractal-64` (AVX2/Metal)	14.36 GiB	15.4–17.27 GiB (RSS)	6.6533 (WT2 7.36)	0.0688 / **0.1254**[†]	—	Modchip 44
16G	`Nemotron-3.5 MaxQ-16GB` (Universal)	**13.39 GiB**	≈15.0 N=1 / 15.8 `RC2`[‡]	**7.3129 ± 0.1923**	**0.1722** (MXFP4)	**83.43%**	Unmod. std ≤ 42

[†]Nemotron KLD pairs: `wiki-alt` vs. 16.07 GiB `MXFP4` ref / 32-probe Step-0 KLD vs. true 58.84 GiB `BF16` (matched rollout vs. `BF16`: 0.0944 `VQ3-64`, 0.0929 `Fractal-64`). [‡]Cold-cache cgroup tests at `-c 32768` (`Swap=0`, `oom_kill=0`): `Qwen3.8` 64G/32G fit strictly below cap (`pgsteal=0`, `refault=0`); `DS-V4` (≈60.3–60.7 GiB needed) and `MaxQ-16GB RC2` (≈15.75 GiB needed) exceed 60G/15G and complete without OOM via continuous host-ZFS-ARC-backed file reclaim (bare-metal throughput unverified; `MaxQ-16GB` $N = 1$ nearly fits 15G with 0.2 GiB refault; Section 7).

7.1 Blind Perplexity & Logit KLD Across All 3 Tiers

Table 2 reports zero-overlap blind `wiki-alt.txt` evaluations across all three tiers (0.000000% 8-gram and 13-gram calibration overlap). On `Qwen3.8-Flash-Next`, `MaxQ-64GB` (53.99 GiB) outperforms `ISTA-DASLab GSQ-RCO-Coder-IQ1_M` (54.40 GiB) by −1.6591 **PPL** (3.53× smaller ΔPPL over `BF16` 2.7766: +0.6559 vs. +2.3150), −46.6% **Mean KLD**, and +8.93 **pp Top-1 agreement** while embedding the 0.81 GiB `blk.48` MTP module. *Baseline disclosure:* `GSQ-RCO-Coder-IQ1_M` is the only published third-party ≤ 55.5 GiB `Qwen3.8` GGUF, and it is coder-calibrated with 256/512 routed experts pruned; evaluating on general Wikipedia prose naturally reflects both our PLE bit reinvestment and general-domain vs. coder-pruned specialization. On `DeepSeek-V4 284B` (strictly unseen chunks `3..51`, 137,935 tokens), `MaxQ-v4` reaches **6.8471 PPL** and **0.6257 nats** Mean KLD (−12.84% KLD vs. legacy `MaxQ-Fractal`). On `Nemotron-3.5 30B`, Table 2 reports both 40-chunk `wiki-alt` KLD vs. `MXFP4` (0.0688 `Fractal-64`, 0.0871 `VQ3-64`) and committed 32-probe Step-0 / matched-rollout KLD vs. true 58.84 GiB `BF16` (0.1254 / 0.0929 and 0.1495 / 0.0944 nats).

7.2 `RouteCrack 2.0` Head-to-Head Throughput, Cold-Cache Cgroup v2 Proof, & Greedy Parity

We benchmark `RouteCrack 2.0` at $T = 0$ against built-in `llama.cpp` speculator modes (`none`, `draft-mtp`, `ngram-map-k`) within the identical patched `de32b268c` binary across four workloads—`W1` (152-LOC Python refactor), `W2` (multi-tool JSON/patch synthesis), `W3` (greenfield C99 SPSC ring buffer), and `W4` (bilingual IT/EN technical prose)—in Table 3 and Figure 3. Direct cross-engine baselines against KTransformers or unpatched upstream `llama.cpp` on `Qwen3.8`/`DeepSeek-V4` are omitted because upstream lacks `ConeCollapse` bias loading and `qwen4exp` zero-copy 3D MoE `--repack`.

Cold-Cache Cgroup v2 Evaluation Across All 4 Flagships (`Swap=0`, $n = 3$ Cold Restarts). To test tier residency without warm outside-cgroup page cache and obtain valid i.i.d. variance (zero Stockfish TT carryover), we executed $n = 3$ independent cold-server restarts per configuration (`-c 32768`, `-t 16 -tb 16`, `cache_prompt=False`) after hypervisor `drop_caches=3` + `POSIX_FADV_DONTNEED` (guest `Cached = 0.3 GiB` at start) inside strict transient `systemd-run` scopes (`MemoryAccounting=yes`, `MemorySwapMax=0`, 95% Student-t CIs with df = 2, $t_{0.975,2} = 4.303$). All 24 cold runs completed with `memory.swap.peak = 0 B` and `oom_kill = 0`, separating into two clear regimes: *Strictly below cap (zero reclaim, certified):* (i) `Qwen3.8 MaxQ-64GB` (`MemoryMax=60G`, `qwen38_64gb_cgroup60g_v311.json`): $N = 1$ **8.78 ± 0.02 tok/s** (CI ±0.05, `VmHWM` 45.46, `cg.peak` 46.53 GiB) vs. `RC2` **15.64 ± 0.05 tok/s** (CI ±0.11, +78.1%, `W1` 22.00 ± 0.19; `VmHWM` 48.09, `cg.peak` 48.35 GiB, `file` 43.40 GiB, `pgsteal=0`, `refault=0`); (ii) `Qwen3.8 MaxQ-32GB` (`MemoryMax=28.5G`, `--ctx-checkpoints 0`, hygiene sampling, `qwen38_32gb_cgroup28g_v311.json`): $N = 1$ **10.00 ± 0.01 tok/s** (`VmHWM` 26.11, `cg.peak` 26.79 GiB) vs. `RC2` **14.58 ± 0.32 tok/s** (CI ±0.80, +45.8%, `W1` 15.31; `VmHWM` 28.08, `cg.peak`

Table 3: Head-to-head single-run RouteCrack 2.0 (v3.1.1) throughput (tok/s) on 16 Zen 5 cores (0 GPU, $T = 0$) reconciled with benchmarks/results/{qwen38,dsv4,nemotron}_routecrack2_v31.json. Qwen3.8: aligned MaxQ-64GB (256 tok/wl); DeepSeek-V4: MaxQ-v4 (192 tok/wl); Nemotron: Fractal-MXFP4 reference (256 tok/wl).

Model Family	Configuration	W1 Refact.	W2 JSON	W3 C99	W4 Prose	Aggregate	vs. $N = 1$	Accept	Mean Len
Qwen3.8-MaxQ-64GB *(Aligned, 176.5B/5B, blk.48 MTP)*	Pure $N = 1$ (none)	8.50	8.68	8.68	8.68	8.67 tok/s	1.00×	0.0%	1.00
	MTP-1 (draft-mtp -d 1)	12.25	12.57	11.89	11.20	12.00 tok/s	+38.4%	89.57%	1.90
	v3.0 (draft-mtp, $n_{\max} = 3$)	16.61	16.84	13.24	10.76	13.94 tok/s	+60.8%	73.79%	3.33
	RouteCrack 2.0 (v3.1.1)	**21.38**	**17.38**	**12.63**	**12.78**	**15.34 tok/s**	**+77.0%**	**79.95%**	**3.66**
DeepSeek-V4 MaxQ-v4 *(284.3B/13.2B, Heavy-MoE, Top-6)*	Pure $N = 1$ (none)	6.48	6.70	6.90	7.01	6.80 tok/s	1.00×	0.0%	1.00
	v3.0 (ngram-map-k)	11.90	7.39	6.74	6.55	7.73 tok/s	+13.6%	85.49%	2.42
	RouteCrack 2.0 (v3.1.1)	**12.31**	**7.55**	6.74	6.49	**7.79 tok/s**	**+14.5%**	**85.92%**	**2.97**
Nemotron-3.5-30B *(Fractal-MXFP4 ref, 30B/3B, Mamba-2)*	Pure $N = 1$ (none)	22.20	22.57	22.69	22.61	22.60 tok/s	1.00×	0.0%	1.00
	v3.0 (ngram-map-k)	42.54	26.08	22.65	23.80	27.13 tok/s	+20.0%	71.86%	3.18
	RouteCrack 2.0 (v3.1.1)	**42.18**	**28.83**	**22.97**	**24.72**	**28.20 tok/s**	**+24.8%**	**78.59%**	**3.35**

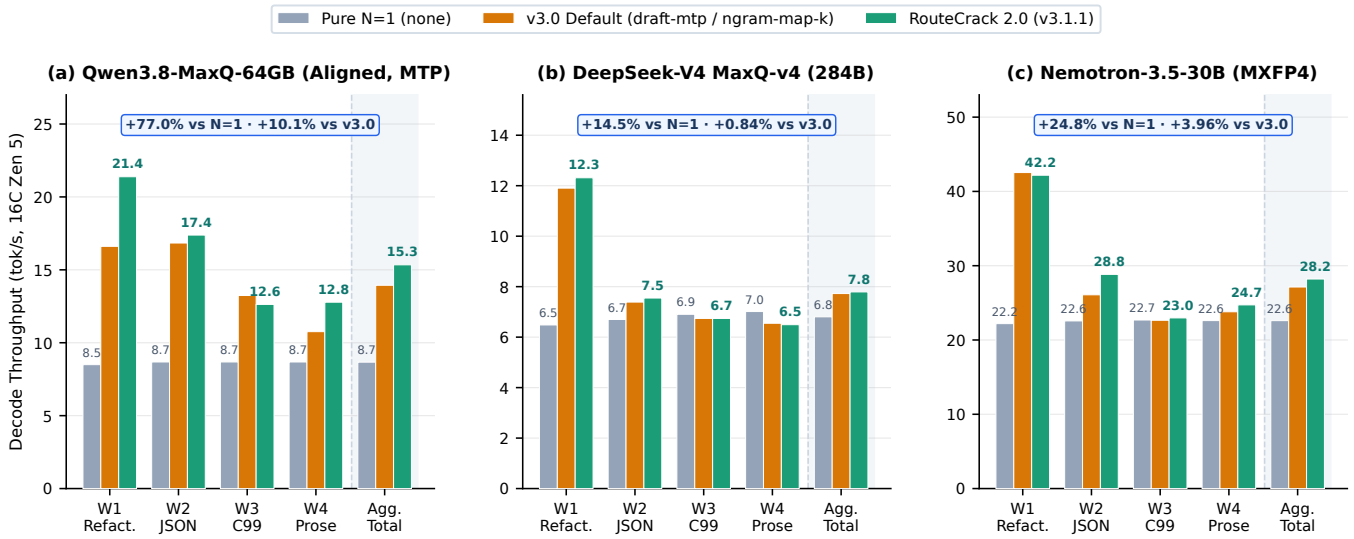


Figure 3: Per-workload and aggregate decode throughput of RouteCrack 2.0 (v3.1.1) vs. Pure $N = 1$ and v3.0 speculators across W1 (Refactor), W2 (JSON Tools), W3 (C99), W4 (Prose), and Aggregate on 16 Zen 5 cores (0 GPU, $T = 0$; Nemotron measured on Fractal-MXFP4 reference).

28.26 GiB—tight 0.24 GiB headroom at 32K—with file 25.14 GiB out of 35.71 GiB on disk, proving > 10.5 **GiB of PLE remained lazily unfaulted with pgsteal=0, refault=0**); (iii) **Nemotron MaxQ-16GB** $N = 1$ (MemoryMax=15G, nemotron_16gb_cgroup15g_-v311.json): **22.64 ± 0.03 tok/s** (VmHWM 14.92, ≈15.03 GiB needed, 0.2 GiB refault—nearly fitting 15 GiB; higher than our earlier unpinned 15.10 t/s llama-bench tg128 gate due to pinned 16C taskset + q8_0/q4_0 KV). *Exceeds cap at -c 32768 (OOM-free via ZFS-ARC-backed reclaim; bare-metal throughput unverified):* (iv) **DeepSeek-V4 MaxQ-v4** at MemoryMax=60G (dsv4_64gb_cgroup60g_v311.json) requires ≈60.3–60.7 GiB (57.99 GiB weights +2.25–2.67 GiB anon), and **MaxQ-16GB RC2** at MemoryMax=15G requires ≈15.75 GiB (13.39 GiB +2.36 GiB anon). Both run without OOM/swap via continuous file-page reclaim (DS-V4: 10.7–30.7 GiB refaulted/run, 8.88 ± 0.25 RC2 / 7.68 ± 0.01 $N = 1$; MaxQ-16GB RC2: 8.4–9.2 GiB refaulted/run, 26.28 ± 0.19). Because refaults are served from our hypervisor host's ZFS ARC RAM rather than NVMe, those reclaim-bound rates do *not* carry over to bare-metal 64/16 GB boxes without ≥ 61 GiB (DS-V4) / ≥ 16 GiB (MaxQ-16GB RC2) free RAM.

Single-Run Variance & Honest Parity on Non-MTP Models. In Table 3, Qwen3.8 gains +77.0% vs. $N = 1$ and +10.1% vs. v3.0 draft-mtp (+28.7% on W1 refactor via sub-250 ns suffix bypass and +18.8% on W4 prose by curbing over-drafting). On non-MTP DeepSeek-V4 (+0.84% vs. v3.0) and Nemotron-30B (+3.96% vs. v3.0), aggregate RouteCrack 2.0 speed is effectively at

Table 4: Cross-platform verification of `llama-modchip` across x86-64, Apple Silicon Metal, ARM64 cloud CPUs, and Vulkan hybrids (community runs cite their GitHub issue/PR).

Hardware Platform	Cores / GPU	Memory	Model & Format	Prefill	Base / Cold	Spec. / W1
AMD Ryzen 9 9955HX	16C (Zen 5, AVX-512)	60G cold cg (48.35, 0 reclaim)	**Qwen3.8 MaxQ-64GB**	**72–100**	8.78 ($N = 1$)	**15.64 / 22.00**
AMD Ryzen 9 9955HX	16C (Zen 5, AVX-512)	28.5G cold cg (28.26, 0 recl.)	**Qwen3.8 MaxQ-32GB**	**101–113**	10.00 ($N = 1$)	**14.58 / 15.31**
AMD Ryzen 9 9955HX	16C (Zen 5, AVX-512)	74G host (60G cg: ARC recl.)	**DS-V4 284B MaxQ-v4**	**45–50.5**	6.80 ($N = 1$)	**7.79 / 12.31**
AMD Ryzen 9 9955HX	16C (Zen 5, AVX-512)	15G cold cg (N=1; RC2 recl.)	**Nemotron MaxQ-16GB**	**188.3**	22.64 ($N = 1$)	**26.28**‡ **/ 43.53**
Apple M5 Max	18C CPU + 40C Metal GPU	64 GB LPDDR5X (0 Swap)	DS-V4 *legacy Fractal**	169.4	27.00 (`tg128`)	20.7–23.6
Threadripper 7985WX (#1)	64C (8 CCDs, 256 MB L3)	256 GB 8-Ch DDR5	Nemotron `Q4_0` (`-t 32`)	**386.0**	73.8–80.1	**111.62 t/s**
Threadripper 9960X (#40)	24C (4 CCDs, AVX-512)	125 GB 4-Ch DDR5	Nemotron `VQ3-64`	**450.4**	49.25 t/s	**54.43 t/s**
7945HX3D + RX 9070 XT	16C + 16 GB Vulkan	94 GB DDR5 + 16 GB	Nemotron (`MXFP4`)	**563.0**	75.10 t/s	**80.80** (256K)
Apple M3/M4 Max (#24,#8)	ARM64 NEON + Metal	36–64 GB Unified	Nemotron (`Frac/Q4`)	**860.1**	71.6–78.6	**236–250** (`RC2`)
AWS Graviton3 `m7g.2xl` (#2)	8 vCPU Neoverse-V1	31 GB DDR5 (0 GPU)	Nemotron-30B (`-t 8`)	**101.7**	41.05 t/s	**44.97 t/s**

*Historical M5 Max result (`apple-silicon/results/dsv4-m5max.md`) was measured on the retired 56.56 GiB `DeepSeek-V4-Fractal` build, not `MaxQ-v4`; `MaxQ-v4` awaits fresh M5 Max re-measurement. ‡`MaxQ-16GB RC2` at 15G cgroup refaulted ≈9 GiB/run from host ZFS ARC; bare-metal 16 GB `RC2` throughput at `-c 32768` is unverified. All prefill/decode rates in tok/s.

Table 5: Complete greedy $T = 0$ common-prefix token parity vs. Pure $N = 1$ (`identity_vs_n1` from raw `*_routecrack2_v31.json` logs).

Model	Speculator	W1	W2	W3	W4
`Qwen3.8`	`MTP-1 / v3.0`	254/254	254/254	4/254	79/254
`Qwen3.8`	`RouteCrack 2.0`	**254/254**	**254/254**	4/254	48/254
`DS-V4`	`v3.0 (ngram-map)`	191/191	120/191	53/191	48/191
`DS-V4`	`RouteCrack 2.0`	**191/191**	**191/191**	34/191	48/191
`Nemotron`	`v3.0 (ngram-map)`	255/255	57/255	17/253	134/255
`Nemotron`	`RouteCrack 2.0`	**255/255**	**255/255**	17/253	173/255

parity with `v3.0 ngram-map-k` within single-run variance; its primary advantage on non-MTP models is eliminating `v3.0`'s `W2` JSON mode-bleed divergence and lifting `Nemotron W2` by +10.5% (28.83 vs. 26.08 tok/s).

Full Greedy $T = 0$ Output Parity Disclosure (`identity_vs_n1`). Table 5 reports complete exact-prefix token match counts across all four workloads against $N = 1$ greedy decoding. Across all three model families, `RouteCrack 2.0` achieves 100% greedy bit-exact parity on structured `W1+W2` (fixing `v3.0`'s `W2` divergence on `DeepSeek` 120/191 and `Nemotron` 57/255 via 28-bit Tagged Mode Refutation). On high-entropy `W3/W4`, *all* speculative methods—including `MTP-1` and `v3.0 draft-mtp`—diverge from $N = 1$ at near-tied logits, which we attribute to batched verification GEMM (`batch > 1`) vs. $N = 1$ GEMV FP32 FMA accumulation order (not yet isolated with a deterministic-reduction kernel).

7.3 Cross-Platform Multi-ISA Verification

Table 4 summarizes local reference runs and community-submitted scorecards (GitHub Issues #1, #2, #8, #16, #40, PR #24) across x86-64, Apple Silicon Metal, ARM64 Neoverse-V1, and Vulkan hybrids.

8 Limitations & Engineering Trade-offs

1. **Loader & Custom Kernel Portability**: Only `Nemotron MaxQ-16GB` ($\leq$ 42, 0 custom types, 0 ConeCollapse bias) targets unmodified upstream `llama.cpp` (`de32b268c+`) and downstream runners supporting `GGML_TYPE_Q2_0 = 42`. `Qwen3.8 MaxQ-64GB` uses standard types $\leq$ 42 (pass `--no-repack` on unpatched upstream `llama.cpp`). `Qwen3.8 MaxQ-32GB` and `DeepSeek-V4 MaxQ-v4` avoid custom tensor types for multi-backend portability but require `llama-modchip v3.1.1+` to load `ConeCollapse ffn_down_exps.bias`. Custom `FRACTAL_64` (44) ships CPU and Apple Metal shaders and `VQ3_64` (45) ships CPU SIMD kernels only (neither has CUDA/Vulkan shaders).

2. **High-Entropy Prose on 284B Non-MTP MoE**: On `DeepSeek-V4 MaxQ-v4`, `W4` free prose runs −7.4% slower than $N = 1$ (6.49 vs. 7.01 tok/s) because 256-expert DDR5 pull cost dominates whenever context n-grams miss; aggregate throughput still rises +14.5% via +90.0% `W1` and +12.7% `W2` gains.

3. **Low-Bit Sampling Hygiene on `MaxQ-32GB`**: At 1.63 bpw, greedy $T = 0$ sampling can trigger tail repetitions on long generations; applying `--temp 0.6 --top-p 0.95 --top-k 20 --presence-penalty 0.2 --repeat-penalty 1.05` (automated by `./modchip`) restores loop-free generation.

9 Conclusion

`llama-modchip v3.1.1` demonstrates that co-designing donor-free `MaxQ` compression (`PCA-Wiener + RaBitQ Q1_0` PLE, FP64 Woodbury DualGram, and `ConeCollapse`), a zero-collision multi-ISA `ggml` ABI (`44/45`), zero-copy 3D MoE `--repack`, and the `RouteCrack 2.0` 832 B L1 Micro-Arbiter enables 30B–284B Sparse MoE inference on commodity 16 GB, 32 GB, and 64 GB CPU workstations. Source code, unified patches, preflight verification tools, and raw benchmark artifacts are published under Apache 2.0 at https://github.com/synchopate/llama-modchip.

References

[1] Qwen Team, "Qwen3.8-Flash-Next Model Release," Hugging Face, https://huggingface.co/Qwen/Qwen3.8-Flash-Next, 2026.

[2] DeepSeek-AI, "DeepSeek-V4-Flash-0731 Model Release," Hugging Face, https://huggingface.co/deepseek-ai/DeepSeek-V4-Flash-0731, 2026.

[3] DeepSeek-AI, "DeepSeek-V3 Technical Report," *arXiv preprint arXiv:2412.19437*, 2024.

[4] NVIDIA, "NVIDIA-Nemotron-3.5-Lightning-30B-A3B Model Release," Hugging Face, https://huggingface.co/nvidia/NVIDIA-Nemotron-3.5-Lightning-30B-A3B-BF16, 2026.

[5] NVIDIA, "Nemotron-H: A Family of Accurate, Efficient Hybrid Mamba-Transformer Models," *arXiv preprint arXiv:2504.03624*, 2025.

[6] F. Gloeckle, B. Y. Idrissi, B. Rozière, D. Lopez-Paz, and G. Synnaeve, "Better & Faster Large Language Models via Multi-token Prediction," in *Proc. ICML*, arXiv:2404.19737, 2024.

[7] J. Gao and C. Long, "RaBitQ: Quantizing High-Dimensional Vectors with a Theoretical Error Bound for Approximate Nearest Neighbor Search," in *Proc. ACM SIGMOD*, arXiv:2405.12497, 2024.

[8] G. Gerganov et al., "`llama.cpp`: LLM Inference in C/C++," https://github.com/ggml-org/llama.cpp, 2023–2026.

[9] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers," in *Proc. ICLR*, arXiv:2210.17323, 2023.

[10] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han, "AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration," in *Proc. MLSys*, arXiv:2306.00978, 2024.

[11] A. Tseng, J. Chee, Q. Sun, V. Kuleshov, and C. De Sa, "QuIP#: Even Better LLM Quantization with Hadamard Incoherence and Lattice Codebooks," in *Proc. ICML*, arXiv:2402.04396, 2024.

[12] V. Egiazarian, A. Panferov, D. Kuznedelev, E. Frantar, A. Babenko, and D. Alistarh, "Extreme Compression of Large Language Models via Additive Quantization," in *Proc. ICML*, arXiv:2401.06118, 2024.

[13] S. Kim, C. Hooper, A. Gholami, Z. Dong, X. Li, S. Shen, M. W. Mahoney, and K. Keutzer, "SqueezeLLM: Dense-and-Sparse Quantization," in *Proc. ICML*, arXiv:2306.07629, 2024.

[14] T. Dao, "FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning," in *Proc. ICLR*, arXiv:2307.08691, 2024.

[15] Y. Song, Z. Mi, H. Xie, and H. Chen, "PowerInfer: Fast Large Language Model Serving with a Consumer-Grade GPU," in *Proc. ACM SOSP*, arXiv:2312.12456, 2024.

[16] H. Chen et al., "KTransformers: Unleashing the Full Potential of CPU/GPU Hybrid Inference for MoE Models," https://github.com/kvcache-ai/ktransformers, 2024–2026.

[17] Y. Leviathan, M. Kalman, and Y. Matias, "Fast Inference from Transformers via Speculative Decoding," in *Proc. ICML*, arXiv:2211.17192, 2023.

[18] T. Cai, Y. Li, Z. Geng, H. Peng, J. D. Lee, D. Chen, and T. Dao, "Medusa: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads," in *Proc. ICML*, arXiv:2401.10774, 2024.

[19] Y. Li, F. Wei, C. Zhang, and H. Zhang, "EAGLE: Speculative Sampling Requires Rethinking Feature Uncertainty," in *Proc. ICML*, arXiv:2401.15077, 2024.

[20] Y. Fu, P. Bailis, I. Stoica, and H. Zhang, "Break the Sequential Dependency of LLM Inference Using Lookahead Decoding," in *Proc. ICML*, arXiv:2402.02057, 2024.

[21] Z. He, Z. Zhong, T. Cai, J. D. Lee, and D. He, "REST: Retrieval-Based Speculative Decoding," in *Proc. NAACL*, arXiv:2311.08252, 2024.

[22] A. Saxena, "Prompt Lookup Decoding," https://github.com/apoorvumang/prompt-lookup-decoding, 2023.

[23] T. Romstad, M. Costalba, J. Kiiski, et al., "Stockfish: Open-Source Chess Engine (Principal Variation Search and Zobrist Transposition Tables)," https://stockfishchess.org, 2008–2026.