DEPGRAPH-U: DEPENDENCY-AWARE EXECUTION PRUNING FOR UNIFIED MULTIMODAL INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Unified multimodal models (UMMs) bring understanding and generation into a single Transformer, reducing the need for separate specialist models. A shared inference implementation can nevertheless execute operators that the current request does not consume. Even when a valid image cache exists, or a caller requests pixels rather than text, unused producers and vocabulary projections can still execute. We use output-dependent liveness to identify work that remains executable but is unnecessary for the requested output or required future state. To address this, we propose DepGraph-U, a dependency-aware execution framework that uses audited output and state dependencies to guide guarded inference rewrites, without further weight modification. Cache-Aware Dependency Dispatch (CADD) bypasses upstream producers on safe cache hits, and an image-only interface omits unused vocabulary projection. For understanding, graph-compatible KV policies distinguish preserving dynamic layer selection from enabling earlier reuse under a stable graph. We evaluate DepGraph-U across two Show-o2 scales and eight benchmarks. The complete system achieves **2.88–3.21× generation speedup** over native Show-o2, with quality trade-offs from the selected approximation policies. Against the reconstructed understanding reference, Exact-DLS preserves all 24,769 decoded responses with **1.15–2.34× speedup**. Across the six tested understanding tasks, 80.8% of Exact-DLS examples finish before incremental reuse, explaining why cache compatibility alone does not determine realized latency.

1 INTRODUCTION

Unified multimodal models (UMMs) combine understanding and generation in a shared Transformer (Vaswani et al., 2017; Chameleon Team, 2024; Zhou et al., 2025; Xie et al., 2025b; Deng et al., 2025). Sharing weights simplifies architecture but does not ensure each request executes only needed computation. In the audited image path, cached diffusion-head output is selected only after upstream features and trunk states are recomputed, and vocabulary logits the image sampler never consumes are still evaluated.

Pruning and caching address which weights, layers, tokens, or intermediates can be omitted (Sun et al., 2024; Chen et al., 2024a; Ma et al., 2024a; Liu et al., 2025a). A separate execution question remains: **which operators are still needed to produce the requested output and maintain future state?** Program slicing and dependence analysis provide the basis by reasoning backward from consumers (Weiser, 1981; Ferrante et al., 1987). For a multimodal runtime, consumers include current predictions and state a later call reads.

Figure 1 highlights two execution gaps. **Unused producers:** an accepted image-cache hit can make an upstream path unnecessary, provided that path has no required state or random-number effects. **Unreached reuse:** a compatible prefix cache saves work only if decoding continues far enough to consume it. Under dynamic layer selection, short answers can terminate during prefix rebuilding. Both gaps show why enabling a cache does not establish which computation is actually avoided.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

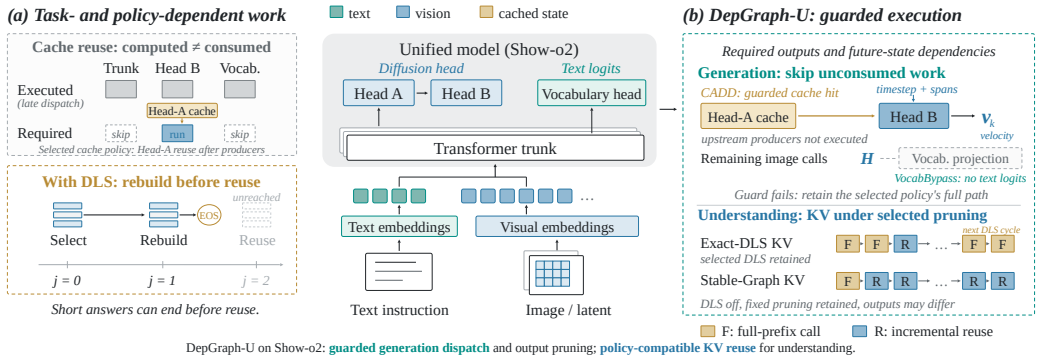
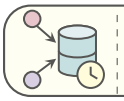


Figure 1: **DepGraph-U overview.** Unused producers and unreached reuse motivate guarded generation and graph-compatible KV policies. Exact preserves the selected DLS policy; Stable changes it.



Which computations remain live for a multimodal request, and when does valid cached state eliminate work before decoding terminates?

We introduce **DepGraph-U**, a dependency-aware execution framework that instantiates this reasoning as guarded rewrites. **Cache-Aware Dependency Dispatch (CADD)** selects safe cache hits before evaluating unused producers. **VocabBypass** omits text-vocabulary projection for image-only calls while retaining text and loss interfaces. For understanding, **Exact-DLS KV** rebuilds prefix state when the active graph changes; **Stable-Graph KV** instead fixes that graph by disabling dynamic layer skipping. These are alternative policies: Exact preserves its selected-policy reference, whereas Stable may change answers and stopping behavior.

Our implementation builds on the Show-o2 architecture, incorporating structured pruning, dynamic layer selection, and diffusion-head caching as the selected approximation policy, with a documented reconstruction of its understanding path. The complete generation system achieves 2.88–3.21× speedup over native Show-o2, with quality changes from the selected approximations. With that policy fixed, the new execution rewrites give 1.61–1.77× additional acceleration and produce pixel-identical images. Exact understanding preserves **24,769 responses** with 1.15–2.34× speedup against its own policy reference. Neither preservation claim is an equivalence claim to unmodified Show-o2.

We explain these results through component controls, execution-cost decomposition, guard checks, and a prefix-length-by-reuse-step experiment. Across six tested Exact-DLS workloads, **80.8%** of examples never reach incremental reuse; in controlled continuations, reuse savings grow between periodic rebuilds. A separate BAGEL study tests the architectural boundary: when native execution already omits a component, its removal provides little additional benefit. The contributions are:

① **Output-dependent execution pruning.** CADD and VocabBypass preserve the chosen generation trajectory while removing unused operators under explicit output and future-state contracts.

② **Graph-compatible and reachable prefix reuse.** Exact and Stable distinguish preserving dynamic selection from changing it for earlier reuse, with distinct output guarantees.

③ **Mechanistic evidence for realized savings.** Paired results, component controls, cost traces, and controlled reuse maps relate avoided operators to later calls that consume cached state.

2 Related Work

Unified models and their output paths. Show-o, Chameleon, Transfusion, and Show-o2 combine multimodal understanding and generation through shared computation (Xie et al., 2025a; Chameleon Team, 2024; Zhou et al., 2025; Xie et al., 2025b). SEED-X, LWM, ILLUME, Janus-Flow, Janus-Pro, Emu3, and VILA-U vary representations, training objectives, and modality interfaces (Ge et al., 2024b; Liu et al., 2024a; Wang et al., 2025; Ma et al., 2024b; Chen et al., 2025a; Wang et al., 2024; Wu et al., 2025b). BAGEL uses a mixture-of-transformer-experts architecture (Deng et al., 2025). UniMoD learns depth allocation; Hyper-Bagel combines speculative decoding and diffusion distillation (Mao et al., 2025; Lu et al., 2025).

Approximation and reuse policies. Weight pruning, visual-token reduction, and depth reduction remove model work (Frantar & Alistarh, 2023; Ma et al., 2023; Sun et al., 2024; Bolya et al., 2023; Chen et al., 2024a; Men et al., 2025; Raposo et al., 2024). DeepCache, FasterCache, TeaCache, TaylorSeer, LeMiCa, PAB, and AdaCache select or predict intermediate states to reuse (Ma et al., 2024a; Lv et al., 2025; Liu et al., 2025a;b; Gao et al., 2025; Zhao et al., 2025; Kahatapitiya et al., 2025).

Adaptive decoding and KV. Deja Vu reconstructs missing cache entries, CALM handles missing states under adaptive depth, and SkipDecode uses monotonic skipping compatible with caching (Liu et al., 2023; Schuster et al., 2022; Del Corro et al., 2023). LayerSkip trains early-exit behavior (Elhoushi et al., 2024). KV compression selects which history to retain (Ge et al., 2024a; Zhang et al., 2023; Li et al., 2024; Cai et al., 2025; Xu et al., 2025); our question is whether that history is compatible with the active graph and reached before termination.

Dependencies and efficient execution. Program slicing and dependence graphs motivate our output-and-state audit (Weiser, 1981; Ferrante et al., 1987). FlashAttention and MInference improve attention execution (Dao et al., 2022; Dao, 2024; Jiang et al., 2024); PagedAttention and SGLang organize KV memory and shared prefixes (Kwon et al., 2023; Zheng et al., 2024). DistServe and Sarathi optimize serving schedules (Zhong et al., 2024; Agrawal et al., 2024).

3 DepGraph-U: Dependency-Aware Execution Pruning

Audit live dependencies. The ***live set*** contains required outputs, future state effects, and their data/control ancestors. Omitted operators must not modify state or consume randomness needed by the reference execution. An operator also remains live if it updates a cache, sampling state, or selection decision that a later call must read.

3.1 Generation: dispatch before evaluating producers

Cache-aware dependency dispatch (CADD). In the selected policy, one-indexed step k computes image features, embeddings, the Qwen trunk, and diffusion-input projection before selecting fresh or cached Head-A output. CADD tests the cache guard first, then skips unused producers on an accepted hit. Head B consumes this value, timestep e_k, and modality spans m_k (Figure 2).

Let c_k denote the stored Head-A output ($\perp$ if absent), f_k indicate full-network mode, and T_c the cache interval. Set $a_k = 1$ iff CADD is enabled and the selected configuration has positive FFN pruning, an implementation-specific eligibility restriction. The caller supplies DLS-recalibration and conditional-only-guidance flags d_k, g_k. The reference predicate q_k and CADD guard b_k distinguish cache hits from safe upstream dispatch:

$$q_k = \begin{cases} \mathbf{1}[k \bmod T_c \neq 0 \ \wedge \ f_k = 0 \ \wedge \ c_k \neq \perp], & T_c > 0, \\ 0, & T_c \leq 0, \end{cases} \qquad b_k = q_k a_k (1 - d_k)(1 - g_k). \quad (1)$$

Testing b_k before the producer stack changes the decision order, not the cache contents or refresh schedule. The branch guard checks whether cached state permits bypass.

Let ρ_k extract image spans, remove the time token when enabled, and unpatchify/reshape Head B's output. CADD returns the image-latent prediction by guarded dispatch:

$$v_k^{\mathrm{CADD}} = \begin{cases} \rho_k(H_B(c_k, e_k, m_k)), & b_k = 1, \\ v_k^{\mathrm{reference}}, & b_k = 0. \end{cases} \quad (2)$$

Accepted hits preserve the cache, timestep, output transformation, and sampler. When the guard fails, the reference path runs with its original cache updates and future-layer selection decisions, preserving the reference policy during fallback.

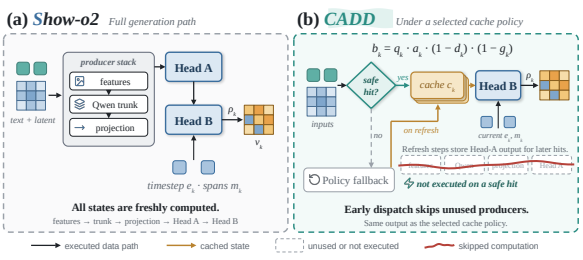


Figure 2: **CADD dispatches before unused producers.** A safe hit bypasses the producer stack; fallback preserves the selected policy.

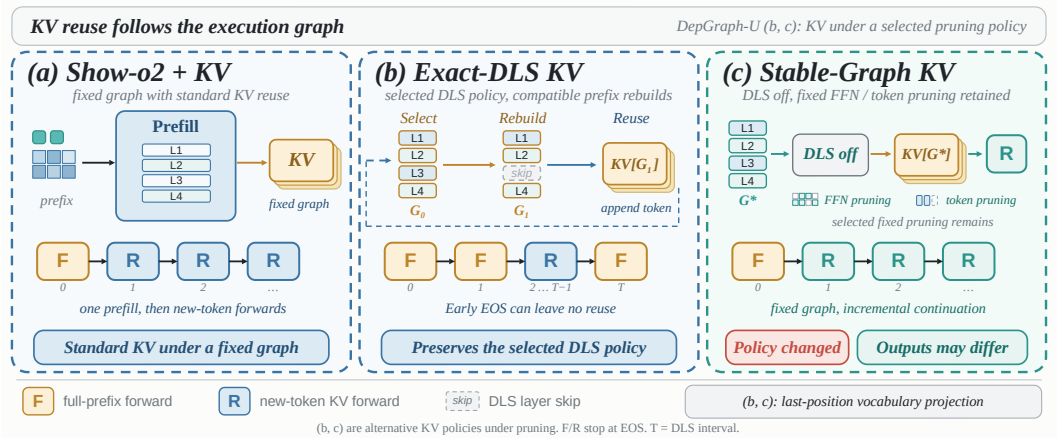


Figure 4: **Graph-compatible prefix reuse.** (a) Fixed-graph KV context. (b) Exact-DLS selects, rebuilds, then reuses. (c) Stable disables DLS; outputs may change. EOS terminates each schedule.

Omit unused logits. VocabBypass applies to the trunk calls that remain after CADD. Image synthesis consumes hidden states through the diffusion head rather than vocabulary logits. An image-only flag omits the projection and returns no text logits; the interface rejects simultaneous training labels. Text generation and text-loss calls retain the projection. The image head receives the same hidden states without first evaluating and discarding an unused matrix product (Figure 3).

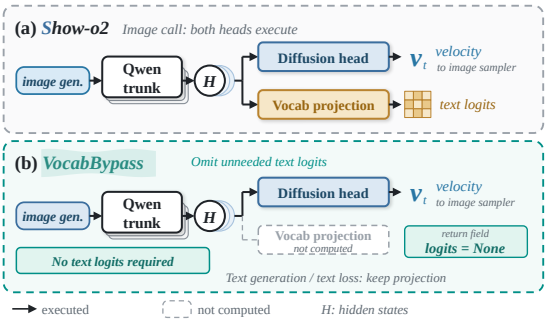


Figure 3: **VocabBypass omits image-call text logits.** Text generation and loss calls retain vocabulary projection unchanged.

Preserve the trajectory. Using the same c_k, e_k, m_k in Head B and ρ_k makes the removed producers irrelevant to v_k, provided they have no required state or random-number effect. Paired PNG hashes and guard/step traces verify the instantiated rewrite against the selected policy's approximate trajectory. Both rewrites retain the refresh schedule; callers needing an omitted value must retain its reference computation. CADD tests whether cached state supplies the image path; VocabBypass tests whether text logits are requested. A cache miss can therefore disable CADD while leaving VocabBypass applicable to the remaining image call. New consumers require dependency audits at the call site.

3.2 Understanding: Align Prefix State with Active Layers

Both KV variants retain top-1-filtered sampling: divide last-position logits by a positive temperature, mask values below the maximum, and sample their softmax with `torch.multinomial`. Tied maxima may survive together. Both use the existing *last-position projection* optimization (Hugging Face, 2025), evaluated separately from KV.

With weights and inference configuration fixed, prefix entries require compatible features/tokens, retained visual indices, positions, attention dependencies, and upstream layers. Compatibility concerns historical positions, not identity of the full growing sequence. Conservative rebuild rules and saved gather maps maintain it; no runtime signature comparison is performed. The implementation maintains layer-specific KV, stores the visual gather map from each rebuild, and gathers the incremental mask's key axis. New token positions come from the unpruned mask length because skipped layer zero need not contain valid KV. Changing an upstream layer can alter historical states even for identical tokens; both the active graph and its visual gather map must therefore remain compatible.

Exact-DLS KV. Phase 0 evaluates the full prefix to choose a skip set; phase 1 conservatively rebuilds under the selected graph even if the selection repeats. Later phases decode one token. Let $D = 1$ when the selected understanding configuration has a positive layer-skip ratio and interval

$T_{\mathrm{LS}} > 0$, and $D = 0$ otherwise. For zero-indexed step j with KV enabled, define the rebuild rule

$$B_j = \begin{cases} \mathbf{1}[j \bmod T_{\mathrm{LS}} \in \{0, 1\}], & D = 1, \\ \mathbf{1}[j = 0], & D = 0, \end{cases} \qquad R(J) = \sum_{j=0}^{J-1} (1 - B_j). \tag{3}$$

When $B_j = 1$, the entire `DynamicCache` is replaced and the current prefix processed; $B_j = 0$ processes the latest token with existing KV. J counts executed decode iterations, including the EOS-producing iteration, not decoded-text length. $R(J)$ counts incremental calls. "Exact-DLS" denotes policy preservation; paired output equality is checked experimentally under the stated decoding and RNG protocol.

Stable-Graph KV. Disabling DLS gives $D = 0$ while retaining permanent FFN and visual-token pruning. One prefill is followed by incremental decoding, $B_j = \mathbf{1}[j = 0]$. This changes the policy and may change predictions; we report it separately and exclude it from exact-output claims.

Attribute composed latency gains. CADD leaves fewer VocabBypass opportunities; their gains are not additive. Likewise, $R(J)$ counts calls, not their cost: prefix length, layers, encoding, projection, and bookkeeping also matter. We use component controls and latency distributions to separate these effects. For a paired workload with latencies t_i^{ref} and t_i^{DG}, our aggregate speedup is $S = \sum_i t_i^{\mathrm{ref}} / \sum_i t_i^{\mathrm{DG}}$. This ratio of means measures workload-time reduction; individual ratios $s_i = t_i^{\mathrm{ref}} / t_i^{\mathrm{DG}}$ instead describe sample-level variation.

4 Experiments

We evaluate three questions: how the complete system compares with native models, what the guarded rewrites contribute under a fixed policy, and when compatible prefix state avoids repeated work. The main study covers two Show-o2 scales and eight benchmarks; BAGEL supplies supplementary second-model checks in the corresponding generation and understanding tables.

Models and protocols. Show-o2 generation uses 1.5B/7B models at 432×432, batch one, on an A800-SXM4-80GB. GenEval uses 553 rewritten prompts and four seeds (2,212 images per arm); DPG-Bench uses 1,065 prompts (Ghosh et al., 2023; Hu et al., 2024). GenEval uses native/system timepoint counts of 100/70 at 1.5B and 50/35 at 7B; DPG uses 50/35. Native and complete-system outputs can differ. Fixed-policy controls instead match weights, prompts, noise, sampler, guidance, and pruning/cache settings, isolating the cost of execution under the chosen trajectory.

4.1 Generation: overall gains and policy-preserving savings

Table 1: **GenEval quality and generation latency.** Show-o2 and BAGEL blocks are local; other rows follow source protocols. The BAGEL subset is diagnostic. Subscripts give native-relative speedup.

Model	Params.	Single obj.	Two obj.	Count	Colors	Position	Color attr.	Overall ↑	Latency (s) ↓
SEED-X (2024b)	13B	0.96	0.65	0.31	0.80	0.18	0.14	0.51	—
LWM (2024a)	7B	0.93	0.41	0.46	0.79	0.09	0.15	0.47	—
ILLUME (2025)	7B	0.99	0.86	0.45	0.71	0.39	0.28	0.61	—
Show-o (2025a)	1.3B	0.98	0.80	0.66	0.84	0.31	0.50	0.68	—
JanusFlow (2024b)	1.3B	0.97	0.59	0.45	0.83	0.53	0.42	0.63	—
Janus-Pro (2025a)	7B	0.99	0.89	0.59	0.90	0.79	0.66	0.80	—
VILA-U (2025b)	7B	0.89	0.62	0.36	0.79	0.20	0.23	0.47	13.58
Show-o2 (2025b)	1.5B	0.994	0.831	0.566	0.867	0.470	0.605	0.722	10.629
+ DepGraph-U	1.5B	0.984	0.806	0.516	0.856	0.388	0.605	0.692	$3.312_{\mathbf{3.209\times}}$
Show-o2 (2025b)	7B	0.978	0.760	0.478	0.819	0.237	0.512	0.631	11.514
+ DepGraph-U	7B	0.959	0.679	0.422	0.787	0.142	0.343	0.555	$3.718_{\mathbf{3.097\times}}$
BAGEL (12-prompt subset)									
BAGEL (2025)	7B	1.000	1.000	0.500	1.000	0.500	0.500	0.750	36.521
+ DepGraph-U	7B	1.000	1.000	1.000	1.000	0.500	0.500	0.833	$32.990_{\mathbf{1.107\times}}$

Tables 1–2 show the complete system's 2.880–3.209× speedup over native Show-o2. These gains combine the selected approximation policies with avoided execution. GenEval quality declines by 0.030 at 1.5B and 0.076 at 7B; DPG declines by 1.389 and 2.177 points, respectively. The rewrites do not recover approximation losses; they reduce the cost of executing the selected trajectory. Holding that trajectory fixed gives 1.612–1.767× additional acceleration with matching paired PNG hashes and equal scores across both scales and benchmarks (Appendix C).

Table 2: **DPG-Bench quality and generation latency.** Subscripts compare local Show-o2 pairs. Literature timings use different protocols and are not directly comparable.

Model	Params.	Global	Entity	Attribute	Relation	Other	Overall ↑	Latency (s) ↓
Show-o (2025a)	1.3B	79.33	75.44	78.02	84.45	60.80	67.27	—
JanusFlow (2024b)	1.3B	87.03	87.31	87.39	89.79	88.10	80.09	—
Emu3-DPO (2024)	8B	87.54	87.17	86.33	90.61	89.75	81.60	124.8
Janus-Pro (2025a)	7B	86.90	88.90	89.40	89.32	89.48	84.19	—
Show-o2 (2025b)	1.5B	81.76	90.94	89.98	93.64	77.60	85.055	5.310
+ DepGraph-U	1.5B	81.46	89.65	89.58	93.06	73.60	83.666	$1.844_{\mathbf{2.880\times}}$
Show-o2 (2025b)	7B	82.37	91.87	89.80	93.75	80.40	86.532	11.538
+ DepGraph-U	7B	84.19	90.26	89.02	93.25	79.60	84.355	$3.723_{\mathbf{3.099\times}}$

Figure 5 varies the cache interval in the complete 1.5B system. Speedup rises from 1.765× at interval one to 3.202× at five, then saturates at 3.190× at eight. Longer intervals offer more opportunities to bypass producers, but retain image-head execution and sampler updates. The sweep reports quality–latency operating points; separate fixed-policy controls isolate execution savings.

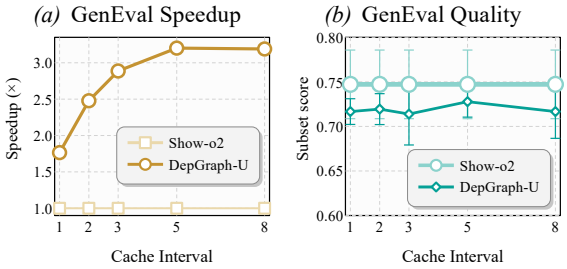


Figure 5: **Cache-interval sensitivity against native Show-o2.** A 120-prompt GenEval subset with three seeds; quality bars show seed SD. Native/system schedules differ, so outputs need not match.

Quality varies with the cache policy and remains below the native subset score. Increasing the interval changes which intermediate states are reused, so the faster operating points need not reproduce native images. The quality bars summarize variation across image seeds, not uncertainty in speedup. Native and system timings come from separate runs without paired initial noise. Their comparison includes approximation and schedule changes; the preservation result instead holds those choices fixed. CADD then changes when a valid cache hit is selected, while VocabBypass omits the unconsumed text projection. These controls explain which execution costs can be removed without attributing the entire native-relative gain to the new rewrites.

BAGEL's supplementary block in Table 1 tests the adapted complete system without TaylorSeer (Deng et al., 2025). On 12 fixed prompts at 1024 × 1024 with three timing repeats, it gives 1.107× speedup; 0.750-to-0.833 quality reflects only one additional passing image, not an established full-benchmark improvement. BAGEL already omits image-call vocabulary projection and supports native KV. Its new rewrite removes an unused image-guidance prefill, yielding only 1.0001× alone and 1.0013× on the fixed FFN-pruning policy (Appendix I). Most generation gain therefore comes from the inherited policy. This architectural boundary is consistent with the dependency audit: an unused producer can save only its existing execution cost when removed.

4.2 Understanding: Preserve a Policy or Enable Earlier Reuse

Understanding uses Show-o2-7B across MME, GQA, MMBench-English, MMMU, MMStar, and AI2D (Fu et al., 2025; Hudson & Manning, 2019; Liu et al., 2024b; Yue et al., 2024; Chen et al., 2024b; Kembhavi et al., 2016). The selected policy combines 10% FFN pruning, 50% visual-token pruning before layer 2, and 10% DLS with period 10. Its reference, FlashU-R, reconstructs the understanding path unavailable in the audited release of Ke et al. (2026); one frozen Wanda artifact is shared across tasks (Sun et al., 2024). Our reference scores 51.737 on MMStar versus the source paper's 48.3, so paired claims concern this reconstruction. Exact retains DLS; Stable disables it. MMB-EN uses circular scoring. MME-P reports perception quality, whereas MME latency includes both perception and cognition tasks in the timed workload.

Table 3 places Exact and Stable within the broader quality context. The gray Show-o2 row is a published reference, so its 1.71 s cannot be divided by the local 0.865/0.491 s to claim controlled acceleration. Our separate same-policy pairs establish Exact's 1.147–2.338× speedup and equality of all 24,769 decoded responses (Table A5). Token sequences also match on all 20,781 examples with suitable raw token logs; older MMMU/AI2D logs support response-level equality only.

Table 3: **Multimodal understanding quality and task latency.** Upper-block latencies use MMMU; the BAGEL block uses MMStar. Gray Show-o2 and literature rows are reported results. Only the local BAGEL pair gives a native-relative speedup.

Model / policy	Params.	MME-P	GQA	MMB-EN	MMMU	MMStar	AI2D	Latency (s) ↓ (MMMU)
ILLUME (2025)	7B	1445.3	—	75.1	38.2	—	71.4	—
Show-o (2025a)	1.3B	1097.2	58.0	—	26.7	—	—	—
JanusFlow (2024b)	1.3B	1333.1	60.3	74.9	29.3	—	—	—
Janus-Pro (2025a)	7B	1567.1	62.0	79.2	41.0	—	—	—
Emu3 (2024)	8B	1243.8	60.3	58.5	31.6	—	70.0	1.29
VILA-U (2025b)	7B	1317.4	58.5	63.5	30.7	37.2	48.9	0.12
Show-o2 (2025b)	7B	1620.5	63.1	79.3	48.9	56.6	78.6	1.71
+ DepGraph-U (Exact)	7B	1593.54	60.693	75.172	46.111	51.737	74.644	0.865
+ DepGraph-U (Stable)	7B	1592.39	60.677	77.234	47.111	51.919	76.490	0.491
			BAGEL					**MMStar (s)**
BAGEL (2025)	7B	—	—	—	—	68.013	—	0.271
+ DepGraph-U (adapted)	7B	—	—	—	—	66.692	—	$0.235_{\mathbf{1.154\times}}$

Stable reaches reuse after one prefill but changes the dynamic-layer policy. Some scores increase under Stable, but individual answers and stopping behavior change; Exact preserves a different reference trajectory. Fresh cap-128 MMStar/MMBench pairs yield $5.236\times$/$2.386\times$ acceleration relative to their own selected-policy references, with only 1,021/1,500 and 3,722/4,329 responses matching (Table A6). Changed stopping contributes to these gains. MME-P and GQA quality decline under Stable. The two variants use separate runs and different policies; comparing their ratios cannot isolate the execution cost of changing the layer-selection schedule.

The supplementary BAGEL block evaluates all 1,500 MMStar inputs under its native KV path. The adapted system reduces latency from 0.271 to 0.235 s ($1.154\times$), while the score falls from 68.013 to 66.692; 1,157 responses match. Unlike the Exact-DLS comparison, the BAGEL adaptation changes the pruning policy and generated answer lengths. This second-model result measures the adapted system's quality–latency tradeoff, with no claim of preserved responses (Appendix I).

Reuse opportunity. Across six Exact-DLS tasks, 20,014/24,769 examples (80.8%) finish during selection or rebuilding. These sample fractions do not measure runtime: long answers can dominate the workload even when most examples never reuse KV. This motivates examining both the opportunity to reuse and its latency effect in Figures 6 and 7. Figure 6 counts examples that execute an incremental call. Its Overall bar pools examples rather than task percentages, so larger benchmarks contribute more samples. Within MMStar, 940 examples have zero reuse and 560 reach at least one incremental call. A short answer can benefit from last-position projection even if $R(J) = 0$, whereas a long answer may amortize repeated prefix rebuilds. Cache compatibility is therefore necessary but insufficient for a workload-level speedup. Prefix length, decoding length, and the active graph determine the cost of the avoided work. An accepted state helps only if a later decoding call consumes it; ending during selection or rebuilding leaves no incremental call to accelerate.

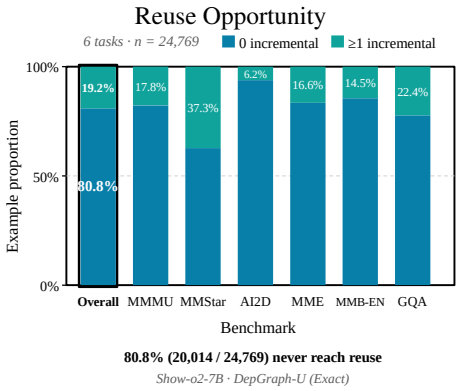


Figure 6: **Natural reuse opportunity.** Six-task example fractions; 80.8% finish before incremental reuse.

Selection and rebuilding remain necessary because they determine the current prediction and the graph used by later steps. The runtime cannot omit these calls merely because their eventual cache will go unused after EOS. This differs from an image-cache hit, where the available state already supplies the live consumer. We retain all 1,500 MMStar pairs, including the 940 zero-reuse cases, in the latency analysis below. Filtering to successful reuse would remove the dominant short-answer group and overstate the benefit for a typical request. Valid state is only an opportunity: the realized continuation determines whether later calls consume it and how much prefix computation they avoid.

Reuse benefits grow. Figure 7 groups the 1,500 MMStar pairs by incremental calls.

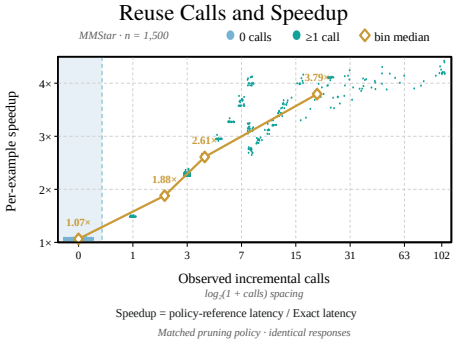


Figure 7: **MMStar reuse and speedup.**

For zero, 1–2, 3–8, and at least nine incremental calls, median paired speedups are 1.066×, 1.880×, 2.610×, and 3.795×. The full workload's 2.338× ratio of means exceeds its typical short-answer gain: median latency falls from 0.857 to 0.796 s. P95 decreases from 8.398 to 2.041 s, consistent with reduced tail latency in this workload. Prefix length and active layers also vary, so the association does not identify a causal gain for each extra reused step. Required prefill and scheduled rebuilding remain in both latency and executed-call counts. The aggregate ratio measures total workload-time reduction, whereas the median paired ratio describes a typical example. Exact retains the reference responses and stopping behavior, so these gains do not come from shorter answers. Stable must instead be assessed with its own quality and output-length results. The four groups contain 940, 121, 298, and 141 examples; each connected marker summarizes a group rather than an equal share of total runtime. These medians describe the recorded workload, not variation across repeated runs. The following controls isolate producer, projection, and KV costs while keeping the approximation policy fixed.

4.3 Component controls explain the avoided work

The controls distinguish operator removal from approximation changes by holding the policy fixed. They test producer bypass, unused projection, and compatible KV separately. Matching outputs ties each timing difference to execution; separate profiling measures the removed work and the operators that still serve the output or maintain its required future state.

Overlapping components. Table 4 isolates CADD and VocabBypass under one policy. Both together reduce 1.5B latency from 2.903 to 1.771 s and 7B latency from 6.521 to 3.671 s. CADD removes producer calls; VocabBypass removes a projection from calls that remain. Their savings therefore overlap rather than add. The 35-timepoint Euler schedule executes 34 model calls: CADD accepts 14 hits, leaving 20 trunk executions, while Head B still executes 34 times. These retained calls preserve the selected outputs and the future-state updates required by the sampler. The current timestep still enters Head B, whose prediction drives the latent update. Moving dispatch ahead of the producers removes their cost while preserving these live consumers, which limit the remaining acceleration. Each configuration uses 20 prompts per scale and three runs, with matching outputs under the fixed policy.

Table 4: **Generation ablation.** Mean latency (s).

Configuration	Show-o2-1.5B	Show-o2-7B
Reference	2.903 s	6.521 s
+ CADD	2.053 s	4.330 s
+ VocabBypass	2.429 s	5.398 s
+ Both (DepGraph-U)	1.771 s	3.671 s

Figure 8(a,b) splits time into upstream, vocabulary, and remaining image-head and sampler work.

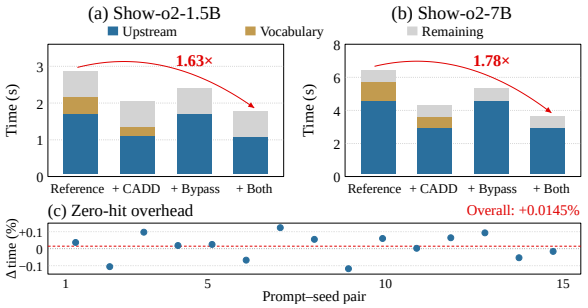


Figure 8: **Executed cost and dispatch overhead.** (a,b) Separately profiled component time at two scales. (c) Zero-hit synchronized model-call overhead across 15 prompt–seed groups.

Profiling uses nested-exclusive CUDA-stream intervals in separate runs, including launch gaps; these are not sums of isolated kernel times. Consequently, profile totals need not exactly equal Table 4's uninstrumented timing. The zero-hit control in panel (c) measures $+0.0145\%$ aggregate overhead across 510 paired model calls, with group values from -0.123% to $+0.128\%$. The measured dispatch cost is small within this scope; the control does not cover the end-to-end pipeline or establish zero overhead.

Guard and interface checks. Normal, forced conditional-only, and zero-hit replay cover 3,570 paired steps, with all recorded prediction and future-state checks matching and zero maximum prediction error. Nine additional interface records verify retained text/logit/loss behavior and rejection of conflicting image-only requests. RNG equality was checked at runtime, but raw RNG states were not exported. These generation checks do not certify all understanding masks, positions, or EOS boundaries (Appendix I).

Separate KV from projection. Table 5 reports speedup, not latency. Restricting projection to the last position gives $1.075\times/1.067\times$ with DLS on/off. With that optimization fixed, adding KV gives $1.984\times/2.042\times$. For each contrast, all responses match across 100 inputs and three repeats; the two contrasts use different denominators. These controls isolate the KV contribution; multiplying them would not replace a directly measured full-system ratio. The source records retain run-level variation separately for each contrast and DLS setting.

Table 5: **Understanding controls.** Mean paired speedup over three runs on 100 MMStar inputs; outputs match.

Isolated change	DLS on	DLS off
Last-position proj. (KV off)	$1.075\times$	$1.067\times$
KV (projection fixed)	$1.984\times$	$2.042\times$

Periodic rebuilds. Figure 9 fixes prefixes at 1,024, 1,536, or 2,048 tokens and forces 32 continuation steps on four inputs with three process repeats. Both arms use last-position projection and the same DLS policy. This removes differences in projection scope and stopping length from the timing comparison, so the recurring cost of selecting and rebuilding the prefix remains visible.

At prefix 1,536, full-prefix Qwen calls take about 789 ms versus 23.8 ms for incremental calls, but selection/rebuild phases recur at steps 1–2, 11–12, 21–22, and 31–32. Cumulative speedup is near one after two calls, rises to 4.12–$4.45\times$ after ten, and drops to 2.69–$2.80\times$ after the next rebuild pair. At the final checkpoint, the 32-call ratio is 3.43–$3.64\times$, including every scheduled rebuild.

This sawtooth pattern exposes the cost of periodic rebuilding despite valid KV between rebuilds. Checkpoints along each trajectory share earlier calls and are correlated; they are not independent repetitions. The Qwen-only timings measure this schedule under forced continuation, while natural EOS determines how much of it a benchmark request executes.

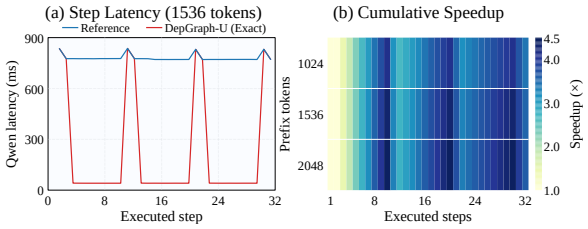


Figure 9: **Controlled prefix length and reuse.** Qwen timing and cumulative speedup over a projection-matched reference; 96 measured cells and 32 forced steps with EOS suppressed.

A response ending just after a rebuild cannot realize the savings available in the rest of that period. Suppressing EOS exposes those later calls without changing the periodic rebuild rule, making the accumulated cost of that rule visible. The controlled sweep therefore explains the schedule, while Figures 6 and 7 establish how often natural requests reach its cheaper phases. These controlled ratios cannot be assigned to every naturally terminating benchmark request. End-to-end benefit also depends on visual encoding, retained projection work, and the point at which EOS ends the response. Evaluating the realized continuation alongside the controlled schedule connects the guard's local opportunity to the work that an actual request can avoid.

5 Discussion and Conclusion

DepGraph-U removes unused generation operators and reuses compatible prefix states when decoding continues beyond rebuilding. CADD, VocabBypass, and Exact KV preserve their fixed-policy references; Stable changes dynamic selection to enable earlier reuse. Component controls explain both avoided work and the live computations that limit acceleration. The framework requires architecture-specific dependency audits, and native implementations that already omit the relevant work offer less headroom, as the BAGEL control demonstrates. The reported equality is empirical under the tested protocols, not a universal guarantee across kernels or hardware. Batch-one timings do not establish serving throughput, and repeated subset controls cannot supply uncertainty for single-run full understanding results. Evaluation must therefore report the complete system's quality tradeoff alongside the execution savings obtained under a fixed policy.

AI USE STATEMENT

Generative AI assistants supported literature discovery, problem framing, method and experiment design, implementation, execution-trace and result analysis, mathematical exposition, translation, manuscript drafting and editing, and figure preparation and typesetting. Reported measurements come from preserved experimental artifacts rather than generated or estimated results; paired outputs, raw records, and aggregate recomputation were checked. The authors remain responsible for the scientific claims, citations, and submitted content.

REPRODUCIBILITY STATEMENT

Section 3 specifies execution guards and cache rebuild rules; Section 4 gives workloads, timing scopes, and equality checks. Appendices A–I give implementation details, experimental protocols, and the evidence index. The source package separates original Show-o2 records from new controls and BAGEL records in `evidence/current_revision/`. It includes editable figure sources, protocol details, and labeled historical diagnostics. The manifest and compact verifier check saved artifacts and recompute aggregates; they do not rerun model inference.

REFERENCES

Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. Taming Throughput-Latency Tradeoff in LLM Inference with Sarathi-Serve. In *18th USENIX Symposium on Operating Systems Design and Implementation*, pp. 117–134, 2024. URL https://www.usenix.org/conference/osdi24/presentation/agrawal.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, 2023. doi: 10.18653/v1/2023.emnlp-main.298. URL https://aclanthology.org/2023.emnlp-main.298/.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token Merging: Your ViT But Faster. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.09461.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV Cache Compression based on Pyramidal Information Funneling. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=ayi7qezU87.

Chameleon Team. Chameleon: Mixed-Modal Early-Fusion Foundation Models. *arXiv preprint arXiv:2405.09818*, 2024. URL https://arxiv.org/abs/2405.09818.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An Image is Worth 1/2 Tokens After Layer 2: Plug-and-Play Inference Acceleration for Large Vision-Language Models. In *European Conference on Computer Vision*, 2024a. URL https://arxiv.org/abs/2403.06764.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are We on the Right Way for Evaluating Large Vision-Language Models? In *Advances in Neural Information Processing Systems*, 2024b.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *Transactions on Machine Learning Research*, 2024c. URL https://openreview.net/forum?id=cSimKw5p6R.

Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. Δ-DiT: A Training-Free Acceleration Method Tailored for Diffusion Transformers. *arXiv preprint arXiv:2406.01125*, 2024d. URL https://arxiv.org/abs/2406.01125.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-Pro: Unified Multimodal Understanding and Generation with Data and Model Scaling. *arXiv preprint arXiv:2501.17811*, 2025a. URL https://arxiv.org/abs/2501.17811v1.

Zhuoming Chen, Ranajoy Sadhukhan, Zihao Ye, Yang Zhou, Jianyu Zhang, Niklas Nolte, Yuandong Tian, Matthijs Douze, Leon Bottou, Zhihao Jia, and Beidi Chen. MagicPIG: LSH Sampling for Efficient LLM Generation. In *International Conference on Learning Representations*, volume 2025, pp. 44169–44190, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/6d50d824ae819d5a961c1d8edc15e833-Abstract-Conference.html.

Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.08691.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems*, 2022.

Luciano Del Corro, Allie Del Giorno, Sahaj Agarwal, Bin Yu, Ahmed Awadallah, and Subhabrata Mukherjee. SkipDecode: Autoregressive Skip Decoding with Batching and Caching for Efficient LLM Inference. *arXiv preprint arXiv:2307.02628*, 2023. URL https://arxiv.org/abs/2307.02628.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, Guang Shi, and Haoqi Fan. Emerging Properties in Unified Multimodal Pretraining. *arXiv preprint arXiv:2505.14683*, 2025. URL https://arxiv.org/abs/2505.14683.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed Aly, Beidi Chen, and Carole-Jean Wu. LayerSkip: Enabling Early Exit Inference and Self-Speculative Decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12622–12642, 2024. doi: 10.18653/v1/2024.acl-long.681. URL https://aclanthology.org/2024.acl-long.681/.

Jeanne Ferrante, Karl J. Ottenstein, and Joe D. Warren. The Program Dependence Graph and Its Use in Optimization. *ACM Transactions on Programming Languages and Systems*, 9(3):319–349, 1987. doi: 10.1145/24039.24041.

Elias Frantar and Dan Alistarh. SparseGPT: Massive Language Models Can be Accurately Pruned in One-Shot. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 10323–10337, 2023. URL https://proceedings.mlr.press/v202/frantar23a.html.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. MME: A Comprehensive Evaluation Benchmark for Multimodal Large Language Models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://arxiv.org/abs/2306.13394.

Huanlin Gao, Ping Chen, Fuyuan Shi, Chao Tan, Zhaoxiang Liu, Fang Zhao, Kai Wang, and Shiguo Lian. LeMiCa: Lexicographic Minimax Path Caching for Efficient Diffusion-Based Video Generation. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2511.00090.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model Tells You What to Discard: Adaptive KV Cache Compression for LLMs. In *International Conference on Learning Representations*, 2024a. URL https://arxiv.org/abs/2310.01801.

Yuying Ge, Sijie Zhao, Jinguo Zhu, Yixiao Ge, Kun Yi, Lin Song, Chen Li, Xiaohan Ding, and Ying Shan. SEED-X: Multimodal Models with Unified Multi-granularity Comprehension and Generation. *arXiv preprint arXiv:2404.14396*, 2024b. URL https://arxiv.org/abs/2404.14396.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An Object-Focused Framework for Evaluating Text-to-Image Alignment. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2310.11513.

Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance. *arXiv preprint arXiv:2207.12598*, 2022. URL https://arxiv.org/abs/2207.12598.

Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ELLA: Equip Diffusion Models with LLM for Enhanced Semantic Alignment. *arXiv preprint arXiv:2403.05135*, 2024. URL https://arxiv.org/abs/2403.05135.

Yushi Hu, Benlin Liu, Jungo Kasai, Yizhong Wang, Mari Ostendorf, Ranjay Krishna, and Noah A. Smith. TIFA: Accurate and Interpretable Text-to-Image Faithfulness Evaluation with Question Answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 20406–20417, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Hu_TIFA_Accurate_and_Interpretable_Text-to-Image_Faithfulness_Evaluation_with_Question_Answering_ICCV_2023_paper.html.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2I-CompBench: A Comprehensive Benchmark for Open-world Compositional Text-to-image Generation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 78723–78747, 2023. doi: 10.52202/075280-3443. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/f8ad010cdd9143dbb0e9308c093aff24-Abstract.html.

Drew A. Hudson and Christopher D. Manning. GQA: A New Dataset for Real-World Visual Reasoning and Compositional Question Answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6700–6709, 2019.

Hugging Face. Qwen2, 2025. URL https://huggingface.co/docs/transformers/v4.51.3/en/model_doc/qwen2. Transformers v4.51.3 documentation. Accessed September 10, 2026.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. MInference 1.0: Accelerating Pre-filling for Long-Context LLMs via Dynamic Sparse Attention. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://arxiv.org/abs/2407.02490.

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S. Ryoo, and Tian Xie. Adaptive Caching for Faster Video Generation with Diffusion Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15240–15252, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Kahatapitiya_Adaptive_Caching_for_Faster_Video_Generation_with_Diffusion_Transformers_ICCV_2025_paper.html.

Junlong Ke, Zichen Wen, Boxue Yang, Yantai Yang, Xuyang Liu, Chenfei Liao, Zhaorun Chen, Shaobo Wang, and Linfeng Zhang. Flash-Unified: A Training-Free and Task-Aware Acceleration Framework for Native Unified Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 9131–9142, 2026. URL https://arxiv.org/abs/2603.15271.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A Diagram Is Worth a Dozen Images. In *European Conference on Computer Vision*, pp. 235–251, 2016.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023. URL https://arxiv.org/abs/2309.06180.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM Knows What You are Looking for Before Generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 22947–22970, 2024. doi: 10.52202/079017-0722. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/28ab418242603e0f7323e54185d19bde-Abstract-Conference.html.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep Embedding Tells: It's Time to Cache for Video Diffusion Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7353–7363, 2025a.

Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World Model on Million-Length Video And Language With Blockwise RingAttention. *arXiv preprint arXiv:2402.08268*, 2024a. URL https://arxiv.org/abs/2402.08268.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From Reusing to Forecasting: Accelerating Diffusion Models with TaylorSeers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15853–15863, 2025b. URL https://arxiv.org/abs/2503.06923.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is Your Multi-modal Model an All-around Player? In *European Conference on Computer Vision*, pp. 216–233, 2024b. doi: 10.1007/978-3-031-72658-3_13.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Ré, and Beidi Chen. Deja Vu: Contextual Sparsity for Efficient LLMs at Inference Time. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 22137–22176, 2023. URL https://proceedings.mlr.press/v202/liu23am.html.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan LI, and Jun Zhu. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5775–5787, 2022. doi: 10.52202/068431-0418. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/260a14acce2a89dad36adc8eefe7c59e-Abstract-Conference.html.

Yanzuo Lu, Xin Xia, Manlin Zhang, Huafeng Kuang, Jianbin Zheng, Yuxi Ren, and Xuefeng Xiao. Hyper-Bagel: A Unified Acceleration Framework for Multimodal Understanding and Generation. *arXiv preprint arXiv:2509.18824*, 2025. URL https://arxiv.org/abs/2509.18824.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference. *arXiv preprint arXiv:2310.04378*, 2023. URL https://arxiv.org/abs/2310.04378.

Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K. Wong. FasterCache: Training-Free Video Diffusion Model Acceleration with High Quality. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.19355.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. LLM-Pruner: On the Structural Pruning of Large Language Models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.11627.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating Diffusion Models for Free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15762–15772, 2024a. URL https://openaccess.thecvf.com/content/CVPR2024/html/Ma_DeepCache_Accelerating_Diffusion_Models_for_Free_CVPR_2024_paper.html.

Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai Yu, Liang Zhao, Yisong Wang, Jiaying Liu, and Chong Ruan. JanusFlow: Harmonizing Autoregression and Rectified Flow for Unified Multimodal Understanding and Generation. *arXiv preprint arXiv:2411.07975*, 2024b. URL https://arxiv.org/abs/2411.07975v2.

Weijia Mao, Zhenheng Yang, and Mike Zheng Shou. UniMoD: Efficient Unified Multimodal Transformers with Mixture-of-Depths. *arXiv preprint arXiv:2502.06474*, 2025. URL https://arxiv.org/abs/2502.06474.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. ShortGPT: Layers in Large Language Models are More Redundant Than You Expect. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 20192–20204, 2025. doi: 10.18653/v1/2025.findings-acl.1035. URL https://aclanthology.org/2025.findings-acl.1035/.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Kadous, and Ion Stoica. RouteLLM: Learning to Route LLMs from Preference Data. In *International Conference on Learning Representations*, volume 2025, pp. 34433–34448, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5503a7c69d48a2f86fc00b3dc09de686-Abstract-Conference.html.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-Depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024. URL https://arxiv.org/abs/2404.02258.

Vijay Janapa Reddi, Christine Cheng, David Kanter, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, Ramesh Chukka, Cody Coleman, Sam Davis, Pan Deng, Greg Diamos, Jared Duke, Dave Fick, J. Scott Gardner, Itay Hubara, Sachin Idgunji, Thomas B. Jablin, Jeff Jiao, Tom St. John, Pankaj Kanwar, David Lee, Jeffery Liao, Anton Lokhmotov, Francisco Massa, Peng Meng, Paulius Micikevicius, Colin Osborne, Gennady Pekhimenko, Arun Tejusve Raghunath Rajan, Dilip Sequeira, Ashish Sirasao, Fei Sun, Hanlin Tang, Michael Thomson, Frank Wei, Ephrem Wu, Lingjie Xu, Koichi Yamada, Bing Yu, George Yuan, Aaron Zhong, Peizhao Zhang, and Yuchen Zhou. MLPerf Inference Benchmark. In *Proceedings of the 47th Annual International Symposium on Computer Architecture*, 2020. URL https://arxiv.org/abs/1911.02549.

Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2202.00512.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q. Tran, Yi Tay, and Donald Metzler. Confident Adaptive Language Modeling. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2207.07061.

Noam Shazeer. Fast Transformer Decoding: One Write-Head is All You Need. *arXiv preprint arXiv:1911.02150*, 2019. URL https://arxiv.org/abs/1911.02150.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.02502.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *arXiv preprint arXiv:2104.09864*, 2021. URL https://arxiv.org/abs/2104.09864.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A Simple and Effective Pruning Approach for Large Language Models. In *International Conference on Learning Representations*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Chunwei Wang, Guansong Lu, Junwei Yang, Runhui Huang, Jianhua Han, Lu Hou, Wei Zhang, and Hang Xu. ILLUME: Illuminating Your LLMs to See, Draw, and Self-Enhance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/papers/Wang_ILLUME_Illuminating_Your_LLMs_to_See_Draw_and_Self-Enhance_ICCV_2025_paper.pdf.

Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, Yingli Zhao, Yulong Ao, Xuebin Min, Tao Li, Boya Wu, Bo Zhao, Bowen Zhang, Liangdong Wang, Guang Liu, Zheqi He, Xi Yang, Jingjing Liu, Yonghua Lin, Tiejun Huang, and Zhongyuan Wang. Emu3: Next-Token Prediction is All You Need. *arXiv preprint arXiv:2409.18869*, 2024. URL https://arxiv.org/abs/2409.18869v1.

Mark Weiser. Program Slicing. In *Proceedings of the 5th International Conference on Software Engineering*, pp. 439–449, 1981.

Samuel Williams, Andrew Waterman, and David Patterson. Roofline: An Insightful Visual Performance Model for Multicore Architectures. *Communications of the ACM*, 52(4):65–76, 2009. doi: 10.1145/1498765.1498785. URL https://doi.org/10.1145/1498765.1498785.

Chenyuan Wu, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, Ze Liu, Ziyi Xia, Chaofan Li, Haoge Deng, Jiahao Wang, Kun Luo, Bo Zhang, Defu Lian, Xinlong Wang, Zhongyuan Wang, Tiejun Huang, and Zheng Liu. OmniGen2: Exploration to advanced multimodal generation. *arXiv preprint arXiv:2506.18871*, 2025a. URL https://arxiv.org/abs/2506.18871v1.

Yecheng Wu, Zhuoyang Zhang, Junyu Chen, Haotian Tang, Dacheng Li, Yunhao Fang, Ligeng Zhu, Enze Xie, Hongxu Yin, Li Yi, Song Han, and Yao Lu. VILA-U: a Unified Foundation Model Integrating Visual Understanding and Generation. In *International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=02haSpO453.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.17453.

Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One Single Transformer to Unify Multimodal Understanding and Generation. In *International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=o6Ynz6OIQ6.

Jinheng Xie, Zhenheng Yang, and Mike Zheng Shou. Show-o2: Improved Native Unified Multimodal Models. In *Advances in Neural Information Processing Systems*, 2025b. URL https://arxiv.org/abs/2506.15564.

Yi Xin, Qi Qin, Siqi Luo, Kaiwen Zhu, Juncheng Yan, Yan Tai, Jiayi Lei, Yuewen Cao, Keqi Wang, Yibin Wang, Jinbin Bai, Qian Yu, Dengyang Jiang, Yuandong Pu, Haoxing Chen, Le Zhuo, Junjun He, Gen Luo, Tianbin Li, Ming Hu, Jin Ye, Shenglong Ye, Bo Zhang, Chang Xu, Wenhai Wang, Hongsheng Li, Guangtao Zhai, Tianfan Xue, Bin Fu, Xiaohong Liu, Yu Qiao, and Yihao Liu. Lumina-DiMOO: An omni diffusion large language model for multi-modal generation and understanding. *arXiv preprint arXiv:2510.06308*, 2025. URL https://arxiv.org/abs/2510.06308v1.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and Evaluating Human Preferences for Text-to-Image Generation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 15903–15935, 2023. doi: 10.52202/075280-0700. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/33646ef0ed554145eab65f6250fab0c9-Abstract-Conference.html.

Yuhui Xu, Zhanming Jie, Hanze Dong, Lei Wang, Xudong Lu, Aojun Zhou, Amrita Saha, Caiming Xiong, and Doyen Sahoo. ThinK: Thinner Key Cache by Query-Driven Pruning. In *International Conference on Learning Representations*, volume 2025, pp. 56691–56709, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/8edb116d5b288b6a9bba4c16ab647702-Abstract-Conference.html.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 Technical Report. *arXiv preprint arXiv:2407.10671*, 2024. URL https://arxiv.org/abs/2407.10671.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9556–9567, 2024. URL https://arxiv.org/abs/2311.16502.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang "Atlas" Wang, and Beidi Chen. H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34661–34710, 2023. doi: 10.52202/075280-1506. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6ceefa7b15572587b78ecfcebb2827f8-Abstract-Conference.html.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-Time Video Generation with Pyramid Attention Broadcast. In *International Conference on Learning Representations*, volume 2025, pp. 3296–3319, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/092c2d45005ea2db40fc24c470663416-Abstract-Conference.html.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H. Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient Execution of Structured Language Model Programs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583, 2024. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving. In *18th USENIX Symposium on Operating Systems Design and Implementation*, pp. 193–210, 2024. URL https://www.usenix.org/conference/osdi24/presentation/zhong-yinmin.

Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the Next Token and Diffuse Images with One Multi-Modal Model. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=SI2hI0frk6.

A Algorithms and Execution Conditions

This supplement specifies DepGraph-U's dependency-aware execution contracts and algorithms (Appendix A), protocols (Appendix B), and matched results (Appendix C). Traces, component sensitivity, and native-model examples appear in Appendices D–F; historical UQP selection remains separate in Appendix G. Appendices H–I index the evidence and new controls. The full system is compared with native Show-o2 for quality and latency; output-preservation claims concern only execution rewrites under a fixed selected policy.

A.1 Generation rewrites and understanding policies

The algorithms and module names below specify the Show-o2 implementation. Auxiliary generation adaptations follow the same dependency rule but retain their own output interfaces and cache schedules (Appendix I). Table A1 distinguishes two complementary generation rewrites from two alternative understanding configurations. CADD moves the existing Head-A cache decision ahead of producers whose outputs the accepted branch does not consume. VocabBypass removes vocabulary projection from the remaining image-only calls. Exact-DLS preserves the dynamic layer-selection policy and manages KV at its boundaries. Stable-Graph disables DLS, changing that policy while retaining permanent FFN and visual-token pruning.

Table A1: **Show-o2 execution policies.** Generation rewrites combine; the two KV configurations are alternatives.

Method	Task	Reused / omitted computation	DLS policy
CADD	Generation	Reuse Head-A cache; omit unused producers	Unchanged
VocabBypass	Generation	Omit unused vocabulary projection	Unchanged
Exact-DLS KV	Understanding	Reuse layer-wise KV between rebuilds	Unchanged
Stable-Graph KV	Understanding	Reuse layer-wise KV after one prefill	Disabled

Conditional elimination argument. Fix an evaluation-mode execution region, its required outputs, and its future state effects. Include their data and control ancestors in the live set. Removing an operator outside this set cannot change a value reaching a live consumer, provided that the branch predicate does not depend on its result, mutation through aliases cannot alter a live tensor, and no required random-number or other state effect is removed. This is a conditional program argument, not a device-independent theorem of bitwise neural-network execution. The conditions must be rechecked for new hooks, consumers, kernels, or stochastic layers. These conditions follow data/control-dependence analysis (Weiser, 1981; Ferrante et al., 1987).

For CADD, the current image prediction is not the only live output. At recalibration steps, hidden states used to choose a future skip set remain live. The full-network region and conditional-only guidance also affect the reference state and batch layout. Equation 1 therefore excludes these cases, along with missing caches, disabled intervals, and unsupported configurations. On an accepted hit, Head B, the current timestep embedding e_k, modality spans m_k, output transformation ρ_k, and sampler remain active. The cache stores a Head-A output, not the final image or final latent prediction.

Image-only projection contract. An image-only caller may set `skip_lm_head=True`. The implementation then returns `logits=None` without multiplying hidden states by the vocabulary matrix; supplying labels simultaneously raises an error. With bypass disabled, text and loss calls retain their original projection. `num_logits_to_keep=0` selects all positions, while a value of one selects the last position. Bypassing vocabulary projection leaves intact the shared hidden states required by the diffusion branch. Diffusion transformers predict over image-latent patches (Peebles & Xie, 2023).

A.2 GENERATION DISPATCH

Algorithm A1 retains the reference sampler, guidance, and schedule flags. The selected policy follows FlashU (Ke et al., 2026); DepGraph-U changes producer execution. Here N counts model calls, not timepoints, c is the initially absent Head-A cache, and S is the skip set. Only scheduled refreshes store new Head-A output; an accepted hit leaves c unchanged. Appendix I distinguishes the 35-timepoint grid from its 34 calls.

Algorithm A1 Guarded generation with CADD and VocabBypass

Input: Prompt p, initial latent x_1, N reference calls, fixed weights;
CADD flag a, bypass flag u, reference schedule and sampler state
Output: Final image under the paired selected-policy execution contract

Prepare reference text conditioning and attention metadata from p
$c \leftarrow \perp$; $S \leftarrow \varnothing$ ▷ Head-A cache and layer skips
for $k = 1, \ldots, N$ **do**
  Read reference flags f_k, d_k, g_k, interval T_c, and timestep t_k
  if $f_k = 1$ **then**
    $c \leftarrow \perp$; active skips $\leftarrow \varnothing$
  else if $d_k = 1$ **then**
    active skips $\leftarrow \varnothing$
  else
    active skips $\leftarrow S$
  end if
  Prepare the reference guidance batch; compute e_k and spans m_k
  $q_k \leftarrow 0$
  if $T_c > 0$ **then**
    $q_k \leftarrow \mathbf{1}[k \bmod T_c \neq 0 \wedge f_k = 0 \wedge c \neq \perp]$
  end if
  $b_k \leftarrow q_k\, a\, \mathbf{1}[\text{selected FFN pruning} > 0]\,(1 - d_k)(1 - g_k)$
  if $b_k = 1$ **then**
    $z \leftarrow c$ ▷ Omit features, embeddings, trunk, projection, Head A
  else
    Compute reference image features and multimodal embeddings
    Run the trunk with active skips, f_k, and `skip_lm_head`$= u$
    Retain hidden states H; $U \leftarrow$ diffusion-input projection of H
    if $q_k = 1$ **then**
      $z \leftarrow c$ ▷ Reference hit still skips Head A
    else
      $z \leftarrow H_A(U, e_k, m_k)$ ▷ Compute fresh Head-A output
      if $T_c > 0$ and $k \bmod T_c = 0$ **then**
        $c \leftarrow \mathrm{detach}(z)$
      end if
    end if
  end if
  $v_k \leftarrow \rho_k(H_B(z, e_k, m_k))$
  if $d_k = 1$ **then**
    $S \leftarrow$ reference layer-selection rule applied to H
  end if
  Apply reference guidance to v_k and update all required schedule state
  $x_{k+1} \leftarrow$ reference sampler update from x_k, v_k, t_k
end for
return Reference image decoder applied to x_{N+1}

Implementation mapping. Archived code paths retain their original module names for reproducibility; these names do not rename the DepGraph-U method. Early dispatch is implemented in `flashu/patch.py`. Cache-hit and refresh predicates come from `flashu/diffusion_cache.py`; `flashu/layer_skipping.py` implements projection bypass. The caller clears the cache when entering full-network mode and computes recalibration

flags before invoking the accelerated forward function. The early branch reproduces the reference time embedding and output reshaping, including optional time-token removal and unpatchification. Diagnostic cache-perturbation hooks are excluded from the reported paired execution.

A.3 Layer-specific KV and the two decoding policies

Algorithm A2 specifies both KV configurations. At phase zero, a full-prefix call executes all layers needed for DLS selection and chooses the next skip set. Phase one rebuilds under the selected graph even if the selection repeats. Phases two through nine process one token with the compatible cache. With DLS disabled, Stable-Graph follows the same incremental path after its initial prefill. Both configurations compute last-position logits and preserve the token-selection rule rather than replacing it with greedy argmax.

The executed `mmu_generate` adapter allocates a new `DynamicCache` at every rebuild. `flashu/understanding.py` handles layer skips, visual selection, positions, and gather maps.

Algorithm A2 Understanding with Exact-DLS KV or Stable-Graph KV

Input: Image I, task-formatted question p, token cap $N_{\max}$, EOS token; fixed weights/pruning, DLS period $T = 10$, mode m, temperature $\tau > 0$
Output: Generated token sequence Y under the selected execution policy

Encode I once; prepare full embeddings E, mask M, and visual spans
Reset sample-specific layer-selection state; $K \leftarrow \varnothing$; $Y \leftarrow [\,]$
$D \leftarrow \mathbf{1}[m = \text{Exact-DLS and DLS is enabled}]$; $S \leftarrow \varnothing$
for $j = 0, \ldots, N_{\max} - 1$ **do**
  $B \leftarrow \mathbf{1}[j = 0]$
  if $D = 1$ **then**
    $\phi \leftarrow j \bmod T$; $B \leftarrow \mathbf{1}[\phi \in \{0, 1\}]$
  end if
  if $D = 1$ and $\phi = 0$ **then**
    active skips $\leftarrow \varnothing$
  else
    active skips $\leftarrow S$ ▷ Empty for Stable-Graph
  end if
  if $B = 1$ **then**
    $K \leftarrow$ new layer-wise cache; discard the old visual gather map
    $E_{\text{in}} \leftarrow E$; $M_{\text{in}} \leftarrow M$; positions $\leftarrow [0, \ldots, |E| - 1]$
    Run the full prefix with active skips and fixed pruning
    Store each executed layer's KV and the current visual gather map G
  else
    $E_{\text{in}} \leftarrow E[-1]$; position $\leftarrow |M|_{\text{key}} - 1$
    $M_{\text{in}} \leftarrow$ last query row of the full mask M
    Form deep-layer key indices from G and subsequently appended positions
    Run the new token; gather mask keys before deep-layer attention
    Read and append each executed layer's KV during that forward pass
  end if
  if $D = 1$ and $\phi = 0$ **then**
    $S \leftarrow$ reference skip selection from full-prefix states
  end if
  Retain the forward pass's last-position hidden state h_{last}
  $\ell \leftarrow W_{\text{lm}} h_{\text{last}} / \tau$ ▷ Only the last position is projected
  Set $\ell_i \leftarrow -\infty$ for every $\ell_i < \max_r \ell_r$
  $y \sim \text{Categorical}(\text{softmax}(\ell))$
  Append y to Y and its embedding to E; extend M by the reference rule
  if $y = \text{EOS}$ **then**
    break
  end if
end for
return Y; apply the unchanged task decoder and answer extractor

The fixed architecture uses Transformer attention and Qwen2 decoder components (Vaswani et al., 2017; Yang et al., 2024). KV-head sharing remains a model choice (Shazeer, 2019; Ainslie et al., 2023).

Positions, masks, and state ownership. The cache is heterogeneous: layers before and after visual pruning can have different prefix lengths, and skipped layers need not contain entries. A rebuild records the retained visual-token indices in the full sequence. Incremental decoding extends this gather map with later text positions and applies it to the mask's key axis at deeper layers. The current absolute position comes from the unpruned mask length, not layer zero's cache length. During a rebuild, visual pruning gathers hidden states, positions, and mask axes together; incremental calls reuse those indices. The operations within each branch of Algorithm A2 occur layer by layer during the forward pass, before the corresponding attention calculation. They do not correct attention after it has been computed. Rotary encodings make positions part of query–key interactions (Su et al., 2021).

The implementation does not compare abstract dependency signatures at runtime. Its conservative phase rule prevents prefix entries produced by incompatible upstream layers from being mixed. "Exact-DLS" denotes preservation of this policy; response equality is an experimental finding under the frozen protocol, not a guarantee for every prompt or floating-point backend.

FastGen, H2O, and SnapKV select retained contexts (Ge et al., 2024a; Zhang et al., 2023; Li et al., 2024); PyramidKV and ThinK vary layer budgets and key channels, respectively (Cai et al., 2025; Xu et al., 2025). Our lifecycle checks cache compatibility. LayerSkip trains for early exits and uses self-speculative verification at inference (Elhoushi et al., 2024); Mixture-of-Depths learns token routing through layers (Raposo et al., 2024). Exact-DLS instead retains the audited schedule.

B Experimental Protocols and Input Construction

B.1 Show-o2 generation workloads and timing

Table A2 records the frozen Show-o2 generation configurations. The main comparison measures the full DepGraph-U system against native Show-o2, including the quality effects of the inherited pruning/cache policy and shorter schedule. Separate component controls fix weights, prompts, seeds, timepoints, guidance, pruning, and cache policy while enabling the execution rewrites. Output equality is checked within those controls, not against native Show-o2. These runs use one A800-SXM4-80GB with batch size one and 432×432 images.

Table A2: **Show-o2 generation settings by benchmark and scale.** Auxiliary protocols appear in Appendix B.3.

Hyperparameter	Symbol	DPG-Bench 1.5B / 7B	GenEval 1.5B	GenEval 7B
Show-o2 inference steps	N_{base}	50	100	50
FlashU-family steps	N	35	70	35
Guidance scale	w	10	7.5	5
Head-cache interval	T_c	5	5	5
Final full-network steps	N_{full}	11	21	11
Image seeds	–	42	0–3	0–3
Shared FlashU / DepGraph-U settings				
FFN pruning ratio	r_p		0.20	
DLS ratio	r_{DLS}		0.20	
DLS recalculation period	T_{DLS}		10	
Early / late CFG	–		$0.75w$ / w	
Image resolution	–		432×432	
Batch size	B		1	

DPG-Bench contains 1,065 prompts and uses seed 42 for the main comparison. Each selected-policy component arm has three process-level runs, with the arm order rotated after the initial round.

Latency is the mean of the three run means, and sample standard deviation is computed across those means. Native Show-o2 has one process-level run. The first image generated in each run remains in the quality and equality checks but is excluded from latency, leaving 1,064 timed images per run.

GenEval contains 553 prompts evaluated with seeds zero through three, yielding 2,212 quality images and 2,208 timed images per arm after excluding one warm-up image per seed. The model-scale configurations in Table A2 differ in steps and guidance, so their scores do not isolate model size. Generation timing covers the recorded pipeline after initialization. Diagnostic traces and pilot runs are kept as separate artifacts and are excluded from the benchmark timing means.

Guidance and full-network region. The inherited FlashU schedule (Ke et al., 2026) uses $0.75w$ before the final full-network region and w within it. As specified in Table A2, the final 11 or 21 steps restore the full network and also disable accepted CADD hits. DLS recalibrates with period ten. Adding either generation rewrite preserves these settings; they are not retuned for the faster implementation. Classifier-free guidance combines conditional and unconditional predictions (Ho & Salimans, 2022); its schedule remains fixed here.

B.2 Understanding datasets and reconstruction

Table A3 lists all six evaluation sets and their decoding and pruning settings. FlashU-R reconstructs the understanding path unavailable in the audited release; it does not claim to reproduce an unreleased official implementation. A single Wanda artifact, calibrated on 20 disjoint DPG images, is frozen across benchmarks, unlike the original paper's MME calibration. Following the released pseudocode, the reconstruction retains the largest visual V-norm scores, although the source prose states the opposite direction. This ambiguity limits the reconstruction. Test answer labels are not used to select pruning masks or make cache decisions.

Related pruning work targets weights, structures, or layers (Frantar & Alistarh, 2023; Ma et al., 2023; Men et al., 2025); visual pruning and merging make separate choices (Chen et al., 2024a; Bolya et al., 2023). We retain the recorded Wanda mask and FlashU visual-selection rule.

All understanding arms use Show-o2-7B with batch size one. In the original six-task study, Python uses seed zero; NumPy, PyTorch, and few-shot seeds are 1234. The fresh cap-128 pairs additionally control the per-example visual RNG state, as described below. The executed adapter uses the top-1-filtered sampling described in Section 3.2; task configuration fields such as `do_sample=False` do not override its actual sampling implementation. The visual encoder runs once per example. Synchronized GPU timings include visual encoding and autoregressive computation but exclude model loading, CPU preprocessing, and scoring. Each complete understanding benchmark has one process-level run per arm. Its mean, median, and P95 therefore describe the examples in that run, not run-to-run uncertainty. Inference benchmarking likewise distinguishes latency from throughput (Reddi et al., 2020).

MMMU, AI2D, MME, and GQA use a 16-token cap. The current Exact-DLS and Stable-Graph main results use 128 for MMStar and MMBench-English. Earlier Stable-Graph pairs for these two tasks use 1,024 and are retained only as historical diagnostics in Table A7, with their own matched FlashU-R references. The cap limits decode iterations, while EOS can stop generation earlier. Output equality is checked before claiming score preservation. MMMU uses its official answer parser; AI2D and GQA use the installed exact-match rules. MMStar reports its category aggregate, MME reports perception and cognition points, and MMBench-English uses CircularEval over 1,164 groups formed from 4,329 rows rather than raw per-row accuracy.

Fresh cap-128 Stable-Graph pairs. The updated MMStar and MMBench-English comparisons pair stochastic visual inputs through per-example RNG control; the paired baseline and Stable arms use the same cap of 128. Preflight checks compare visual latents and verify incremental reuse before full evaluation. Reported means include first full-evaluation calls and cap-reaching examples, but exclude smoke timings. These runs have separate process and hardware segments from the original Exact-DLS study. Their fresh paired baseline must therefore remain attached to their quality and latency values; neither a cap-1024 baseline nor the older Exact-DLS baseline is a substitute.

Table A3: **Show-o2 understanding settings.** Current caps match the main tables; legacy cap 1,024 applies only to Table A7.

(a) Evaluation datasets

Dataset	Split	Test cases	Metric
MMMU	validation	900	Official accuracy
MMStar	val	1,500	L2-category macro
AI2D	test	3,088	Exact match
MME	test	2,374	Perception / cognition
MMBench-EN	dev v1.0	4,329	CircularEval
GQA	testdev	12,578	Exact match

(b) Decoding and pruning hyperparameters

Hyperparameter	Symbol	Exact-DLS KV	Stable-Graph KV
Token cap: MMMU / AI2D / MME / GQA	$N_{\max}$	16	16
Token cap: MMStar / MMBench (current)	$N_{\max}$	128	128
Historical MMStar / MMBench cap	$N_{\max}$	–	1,024
DLS ratio	r_{DLS}	0.10	0
DLS period	T_{DLS}	10	–
FFN pruning ratio	r_p	0.10	0.10
Visual-token pruning ratio	ρ	0.50	0.50
Visual pruning layer	L_p	2	2
Calibration images	N_{calib}	20	20
Vocabulary projection	–	Last position	Last position

B.3 Auxiliary generation protocols

The earlier Lumina-DiMOO and OmniGen2 studies test the generation rewrites, not understanding-side KV transfer. They retain their own cache policies; the later no-TaylorSeer BAGEL study is specified separately in Appendix I. Both use batch one and 1024×1024 images. GenEval uses 553 original prompts with seeds zero through three, giving 2,212 pairs per model; DPG uses 1,065 original prompts with one image per prompt. These input banks are distinct from the rewritten Show-o2 GenEval prompts described below. Within each pair, weights, prompts, seeds, step count, guidance, and cache settings are fixed; only the selected execution rewrite changes.

Lumina-DiMOO uses 64 steps, CFG 4, and ML-Cache ratio 0.9, warmup 0.3, and refresh interval 5. Its VocabBypass adaptation retains image-token logits and omits only the unused text-vocabulary projection. OmniGen2 uses 50 steps, text/image CFG 5/1, and TaylorSeer threshold 3, first-enhance 5, and order 4. CADD acts on accepted prediction steps against the last-layer-only reference, retaining the cache-refresh and feature-prediction schedule. Each adaptation preserves the outputs required by its own image sampler.

Synchronized pipeline wall time includes prompt encoding, sampling, and image decoding, but excludes loading, saving, equality checks, and warm-up. Each arm is scored independently. DPG evaluation uses the corrected CSV loader in both arms, retaining the first data row and all 14,392 question answers. The single-image protocol differs from the four-image-grid leaderboard protocol; these tables do not rank models against that leaderboard or against Show-o2. Tables A12 and A13 report ratios of unrounded mean pipeline times separately for each model and benchmark. Independent scores supplement the paired image checks: equal aggregate quality alone would not establish that individual outputs were preserved.

Recovered runs span process and hardware segments, with pairing within each segment. OmniGen2 pairs match within processes, while a preserved recovery check failed cross-process bitwise reproducibility despite matching weights. GenEval followed a predeclared timing-replication gate, and the slower pilot pair remains in the audit record. A separate Lumina 2×2 control uses eight prompts, three repeats, and four arms (96 calls) to isolate bypass with ML-Cache off and on. It records RNG resets but not poststates, so output matching does not establish RNG-state equivalence.

B.4 Inputs, extraction, and score provenance

Show-o2 generation input. The prompt uses the full text in the frozen rewritten GenEval metadata, rather than only the short object label shown in a figure. For example, the green-cup/yellow-bowl case in Figure A3 uses this recorded input (line wrapping is typographic):

> a photo of a green cup and a yellow bowl. A high-resolution photograph featuring a vibrant green ceramic cup with a smooth, glossy finish, placed beside a sunny, bright yellow round bowl on a natural wooden table. The scene is lit by soft, natural sunlight streaming in from a nearby window, casting gentle shadows and highlights that accentuate the sheen of the ceramic surfaces. The background is slightly blurred, suggesting a cozy, homey indoor space with hints of warm, neutral tones, allowing the vibrant colors of the cup and bowl to stand out prominently.

The input template combines the metadata's short request with its saved elaboration. No new elaboration is generated during evaluation. The case records preserve the exact strings, including whitespace; the quotation above normalizes only the whitespace separating the two parts.

Understanding input. The adapter combines a fixed chat wrapper, image embeddings, and the installed task's question/options. The template below preserves their ordering; bracketed fields represent embeddings or task-specific content rather than literal prompt text:

```
[BOS]system
You are a helpful assistant.<|im_end|>
<|im_start|>user
[BOI][optional time embedding][image embeddings][EOI]
[task question, answer options, and answer instruction]
<|im_start|>assistant
```

The adapter removes the literal `<image>` placeholder from the task context while retaining the image through a separate embedding branch. Within each comparison, the task-specific prompt, token cap, EOS handling, text decoding, and answer extraction are unchanged. Response matching compares decoded answer strings; token-ID matching is reported separately when both paired logs contain those IDs. An identical wrong answer counts toward response matching but not accuracy.

Binding quality to timed images. GenEval's overall score is the unweighted mean of six category rates. The evidence package includes the metadata digest and scorer revision. The newly timed native 1.5B image bank differed from the older bank in all 2,212 file and decoded-pixel comparisons. We therefore rescored the new images with the official evaluator without changing its thresholds. The resulting macro score is 0.72203406, with a mean non-warm-up latency of 10.62890507 seconds. The complete-image pass fraction, 0.71609403, is a different statistic. The older score of 0.72302 is not paired with these new timings.

In repeated fixed-policy DPG matrices, every execution-rewrite arm must match the preserved reference images before its run is accepted. Repeating a prompt checks repeatability; it does not create an independent test prompt. For 7B GenEval, both paired banks match the same 2,212-image reference, giving 4,424 reference comparisons with zero mismatches. Scores are transferred only after image-bank equality is verified. This transfer is not an independent evaluator run.

C Complete Results and Matched Controls

C.1 Category-wise generation quality

Table A4 breaks down the main GenEval results into the six scored categories. Each row includes 2,212 images, with category denominators of 320, 396, 320, 376, 400, and 400 in table order. Let s_c be the number of successful images and n_c the category size. The reported aggregate is

$$Q_{\text{GenEval}} = \frac{1}{6}\sum_{c=1}^{6}\frac{s_c}{n_c}. \tag{C.1}$$

We compute this aggregate from unrounded fractions. Averaging rounded table entries or pooling all successes into a single fraction need not reproduce the displayed score.

Table A4: **Category-wise GenEval scores.** Six category rates and their macro average for both Show-o2 model scales.

Method	Size	Single obj.	Two obj.	Count	Color	Position	Color attr.	Overall
Show-o2	1.5B	0.994	0.831	0.566	0.867	0.470	0.605	0.72203
FlashU	1.5B	0.984	0.806	0.516	0.856	0.388	0.605	0.69241
+ DepGraph-U	1.5B	0.984	0.806	0.516	0.856	0.388	0.605	0.69241
Show-o2	7B	0.978	0.760	0.478	0.819	0.237	0.512	0.63092
FlashU	7B	0.959	0.679	0.422	0.787	0.142	0.343	0.55546
+ DepGraph-U	7B	0.959	0.679	0.422	0.787	0.142	0.343	0.55546

Within each model scale, the FlashU and DepGraph-U rows are identical because their paired image files are identical. The quality gap to the native reference remains, including counting and spatial relations. For example, 1.5B position successes fall from 188/400 for native Show-o2 to 155/400 for FlashU; DepGraph-U preserves the latter count. This behavior is consistent with an execution rewrite that preserves the selected approximate trajectory without correcting its errors. It does not establish the same gap at matched native/FlashU step counts, which we did not evaluate. Related compositional criteria appear in T2I-CompBench (Huang et al., 2023); our scores follow GenEval (Ghosh et al., 2023).

C.2 Exact-DLS: equality and absolute latency

Table A5: **Same-policy Exact-DLS results.** The reference is FlashU-R; the method adds Exact KV and last-position projection. Equality concerns this reference, not native Show-o2.

Method	MME-P	GQA	MMB-EN	MMMU	MMStar	AI2D
Quality						
Policy reference	1593.54	60.693	75.172	46.111	51.737	74.644
DepGraph-U (Exact)	1593.54	60.693	75.172	46.111	51.737	74.644
Mean model latency (s) ↓						
Policy reference	0.823	0.848	1.081	1.183	2.541	1.032
DepGraph-U (Exact)	**0.718**	**0.715**	**0.817**	**0.865**	**1.087**	**0.781**
Speedup ↑	**1.147×**	**1.185×**	**1.323×**	**1.367×**	**2.338×**	**1.321×**

Table A5 reports mean model latency for each full benchmark. We define speedup as

$$\text{Speedup} = \frac{\bar{t}_{\text{reference}}}{\bar{t}_{\text{method}}}, \qquad \bar{t} = \frac{1}{n}\sum_{i=1}^{n} t_i. \tag{C.2}$$

This is not the arithmetic mean of per-example ratios. We compute ratios from unrounded measurements even when the displayed means have three decimal places.

All 24,769 decoded responses match their paired FlashU-R references, so official task scores remain unchanged under the same extraction rules. Matching token sequences are also available for all 20,781 examples from MMStar, MME, MMBench-English, and GQA. Earlier MMMU and AI2D

baseline logs lack comparable token sequences, limiting their equality claim to decoded responses. Equality does not imply that the timing logs themselves are byte-identical.

The gains combine the KV lifecycle with last-position projection. Main Table 5 holds projection fixed to estimate the incremental benefit of KV and holds KV fixed to estimate projection savings, using three repeats on 100 MMStar inputs. This attribution subset does not provide a conditional estimate for all examples that reuse KV. Full-benchmark median latency ratios are only 1.065–1.118×, compared with mean ratios of 1.147–2.338×. The accompanying data retain unrounded mean, median, and P95 records. Figure 6 shows reuse fractions, while Figure 7 shows MMStar per-example speedups with reuse-bin medians; those medians are not ratios of dataset-level latency statistics. Appendix D connects the workload-level contrast to executed call counts and measured costs.

C.3 Stable-Graph: quality and speed must be read together

Table A6: **Stable-Graph paired results.** Fresh cap-128 pairs are marked †; other tasks use cap 16. Each speedup uses its own matched reference, and outputs may change.

Method	MME-P	GQA	MMB-EN†	MMMU	MMStar†	AI2D
Quality						
Policy reference	1593.54	60.693	74.914	46.111	51.665	74.644
DepGraph-U (Stable)	1592.39	60.677	77.234	47.111	51.919	76.490
Mean model latency (s) ↓						
Policy reference	0.823	0.848	1.109	1.183	2.455	1.032
DepGraph-U (Stable)	**0.434**	**0.434**	**0.465**	**0.491**	**0.469**	**0.441**
Speedup ↑	**1.898×**	**1.953×**	**2.386×**	**2.408×**	**5.236×**	**2.341×**

Table A7 retains the original six-task Stable-Graph study, placing quality alongside latency. Percentage scores and MME point scores use different units and are not averaged together. The cap-1,024 MMStar and MMBench-English rows have their own cap-1,024 references; their speedups are not computed against the cap-128 Exact-DLS references.

Table A7: **Original Stable-Graph results.** Legacy cap-1024 rows retain their paired baselines; see Table A6 for cap 128.

Method	Score ↑	Latency (s) ↓	Speedup ↑
MMMU; cap 16			
FlashU-R	46.111	1.183	1.000×
Stable-Graph KV	47.111	0.491	2.408×
AI2D; cap 16			
FlashU-R	74.644	1.032	1.000×
Stable-Graph KV	76.490	0.441	2.341×
MME; cap 16			
FlashU-R	1593.54 P 347.86 C	0.823	1.000×
Stable-Graph KV	1592.39 P 362.86 C	0.434	1.898×
GQA; cap 16			
FlashU-R	60.693	0.848	1.000×
Stable-Graph KV	60.677	0.434	1.953×
MMStar; cap 1,024 (legacy diagnostic)			
FlashU-R	51.789	6.215	1.000×
Stable-Graph KV	51.658	0.466	13.326×
MMBench-EN; cap 1,024 (legacy diagnostic)			
FlashU-R	75.086	2.062	1.000×
Stable-Graph KV	77.062	0.463	4.451×

In this original study, 1,998 of the 24,769 responses differ from their paired references. Improvements in MMMU, AI2D, MME cognition, and MMBench-English accompany declines in GQA, MMStar, and MME perception. The available control of visual-encoding random state does not identify DLS as the sole cause of each quality difference. These historical measurements show a policy trade-off rather than lossless acceleration or a consistent improvement in quality.

In the legacy MMStar reference, eight outputs reach the 1,024-token cap and account for 63.59% of total latency. We therefore retain its $13.326\times$ mean ratio as a labeled diagnostic rather than a headline speedup. The short-answer comparisons with cap 16 yield mean ratios of 1.898–$2.408\times$. Interpreting these numbers requires considering both changes in the outputs and their stopping behavior.

The fresh cap-128 pairs in Table A6 replace the two legacy rows for the headline comparison, not for this historical diagnostic. MMStar scores change from 51.665 to 51.919, with $5.236\times$ mean speedup; MMBench-English changes from 74.914 to 77.234, with $2.386\times$ speedup. Only 1,021/1,500 and 3,722/4,329 responses match, respectively. The fresh references reach cap 128 on 10 and 7 examples, accounting for 14.93% and 7.99% of their runtime; Stable has no cap hits. Thus these gains include changed stopping behavior and do not support output-preserving KV claims. The older 1,998-response count and MMStar quality decline above belong only to the original study; they are not aggregates for the refreshed paired table.

Matching caps alone does not align the original Exact-DLS run with the fresh Stable-Graph pairs: their RNG protocols and hardware segments also differ. Main Table 5 isolates KV with projection fixed; comparing full-benchmark times across studies does not.

C.4 Additional native-KV diagnostic

The main native comparison uses the basic Show-o2 execution. The additional control in Table A8 separately enables standard native KV; it is not the denominator for the main speedups. It covers all 900 MMMU validation inputs with cap 16. Within each block, speedup and response matching are measured against the no-KV reference. Standard native KV and Exact-DLS each preserve 900/900 responses relative to their own references, while Stable-Graph preserves 776/900 relative to FlashU-R.

Table A8: **MMMU standard-KV control.** Native Show-o2 and FlashU-R use their own paired no-KV references.

Implementation	Score (%)	Latency (s)	Speedup	Match (%)
Native Show-o2-7B				
No KV	48.111	1.687	1.000×	–
+ Standard KV	48.111	0.789	2.139×	100.00
FlashU-R family: fixed pruning; DLS varies				
No KV	46.111	1.183	1.000×	–
+ Exact-DLS KV	46.111	0.865	1.367×	100.00
+ Stable-Graph KV	47.111	0.491	2.408×	86.22

Native Standard KV takes 0.789 seconds per sample at 48.111% accuracy, compared with 0.865 seconds at 46.111% for Exact-DLS. At this measured operating point, the pruned policy with its rebuild cost is neither faster nor more accurate. Stable-Graph is faster at 0.491 seconds but uses a different policy and achieves 47.111% accuracy. This control covers MMMU only and does not establish a ranking across datasets. Unlike this DLS control, long-context methods optimize token retention and attention sampling (Xiao et al., 2024; Chen et al., 2025b).

The native accuracy is the official $433/900$, verified by an independent recount. An earlier manual parser mishandled nested open-answer predictions and counted 428; we exclude that value. The correction affects the quality interpretation of the control and remains documented in the audit trail.

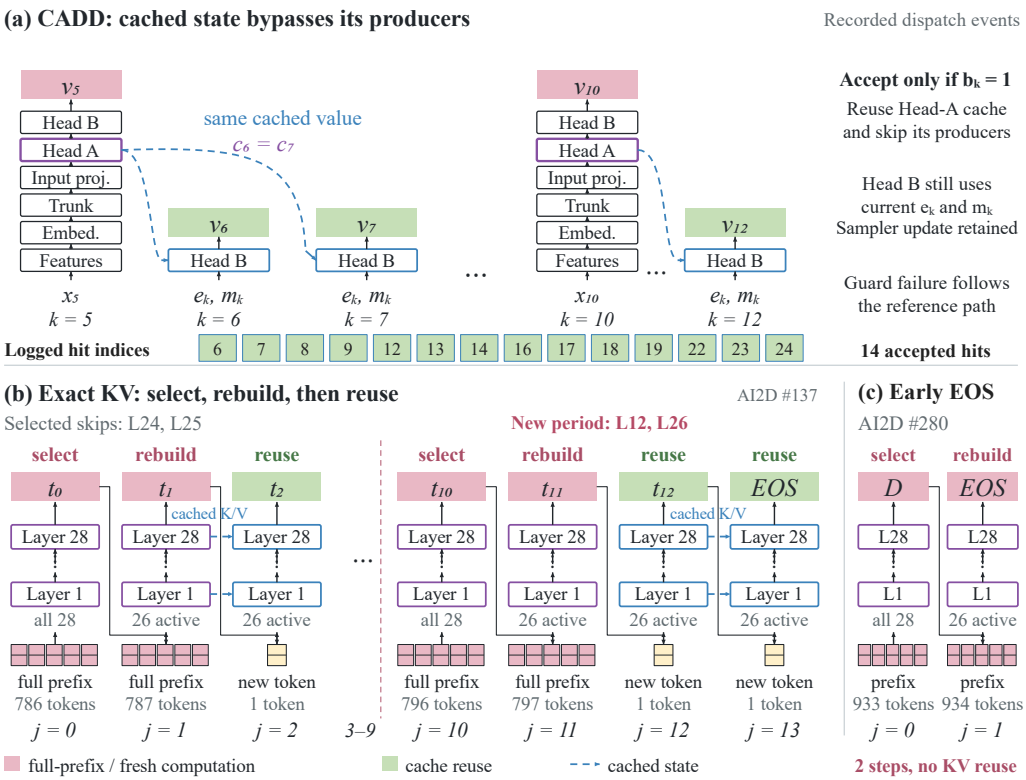


Figure A1: **Preserved execution traces.** (a) CADD reuses Head-A state and retains Head B. (b) AI2D 137 rebuilds at a DLS boundary. (c) AI2D 280 stops before reuse. Dashed arrows mark cached dependencies; spacing is not elapsed time.

D Execution Traces and the Cost of Rebuilding

D.1 From a Branch Decision to the Executed Path

Figure A1 connects the algorithms to preserved execution traces. The generation panel shows selected calls and the logged indices of accepted dispatches. CADD accepts at one-indexed steps $\{6, 7, 8, 9, 12, 13, 14, 16, 17, 18, 19, 22, 23, 24\}$, totaling 14 recorded acceptance events. These labels identify positions in the historical schedule, not a complete count of model invocations. The component audit in Appendix I separately verifies 35 timepoints and 34 model calls. Steps six and seven reuse the same cached Head-A output but evaluate Head B with their respective current time embeddings. The remaining steps follow the reference path. This count is specific to the recorded schedule, not a universal hit rate for other cache intervals or denoising lengths.

AI2D example 137 executes 14 decoding iterations, including the EOS-producing iteration. It selects layers 24 and 25 to skip in the first period, then layers 12 and 26 in the next. The full-prefix call at phase zero selects the path, and phase one rebuilds its KV. Example 280 emits `D` followed by EOS, executing only the two full-prefix calls. These examples illustrate the cache lifecycle rather than compare answer correctness. Layer labels are one-indexed, and the diagram displays selected layers rather than every intermediate block. Prefix lengths are measured at the decoder input before visual-token pruning, not at every layer's KV cache. The CADD module stacks depict source-level dependencies rather than measured per-module times.

D.2 Reachable Reuse Across the Full Workload

For period ten, write $J = 10q + r$, where $0 \le r < 10$. Equation 3 then gives

$$R(J) = 8q + \max(0, r - 2), \qquad \sum_{j=0}^{J-1} B_j = 2q + \min(r, 2). \tag{D.1}$$

The iteration count includes EOS. Across the six Exact-DLS logs, 20,014/24,769 examples (80.8%) have $R(J) = 0$. In benchmark order—MMMU, MMStar, AI2D, MME, MMBench-English, and GQA—the zero-reuse counts are 740, 940, 2,895, 1,980, 3,701, and 9,758, respectively (Figure 6). The overall rate weights examples equally instead of averaging the six dataset percentages. A reusable cache reduces a later call only when execution reaches that call.

D.3 Measured time by observed answer length

Table A9 groups the full logs by observed output length. All 65,828 recorded Qwen calls have per-call timings. Each of the 20,014 zero-reuse examples executes two iterations in these records; this length grouping contains no one-iteration examples. The longer-answer groups contain the remaining 4,755 examples.

Let $\mathcal{I}_g$ denote the examples in a length group and c_{ij} the recorded Qwen time, in milliseconds, for iteration j of example i. The two phase statistics and the trace share are

$$\bar{C}_{\text{rebuild}}(g) = \frac{\sum_{i\in\mathcal{I}_g}\sum_j B_{ij}c_{ij}}{|\mathcal{I}_g|}, \tag{D.2}$$

$$\bar{C}_{\text{incr}}(g) = \frac{\sum_{i\in\mathcal{I}_g}\sum_j (1-B_{ij})c_{ij}}{\sum_{i\in\mathcal{I}_g}\sum_j (1-B_{ij})}, \tag{D.3}$$

$$P_{\text{rebuild}}(g) = 100\frac{\sum_{i\in\mathcal{I}_g}\sum_j B_{ij}c_{ij}}{\sum_{i\in\mathcal{I}_g}\sum_j c_{ij}}. \tag{D.4}$$

Table A9: **Exact-DLS costs by answer length.** Recorded rebuild costs are per example; incremental costs are per call.

Generated tokens	Test cases	LM gen. (s)	Rebuild (ms/ex.)	Incremental (ms/call)	Rebuild / trace (%)
2	20,014	0.656	654.2	–	100.00
3–9	4,290	0.681	648.6	21.01	95.65
10–19	351	1.536	1294.6	20.96	84.89
20–49	76	2.896	2369.3	21.02	82.45
50–128	38	9.695	7992.0	21.26	83.03

The incremental mean is undefined when its denominator is zero and is displayed as a dash. "LM gen." denotes the mean recorded language-model generation time in seconds, excluding vision and setup. Rebuild time is summed per example, whereas incremental time is pooled per call; the columns therefore use different normalizations. The trace share excludes generation-loop overhead and visual encoding but includes necessary initial prefills. It must not be interpreted as entirely removable overhead or as an end-to-end saving.

These groups reflect observed outcomes across heterogeneous tasks, rather than a controlled experiment that forces different answer lengths. Prefix lengths, active layers, and task composition may differ between groups. The roughly 21 ms incremental calls help explain why reuse can benefit longer answers, although repeated prefix rebuilding still accounts for much of their traced Qwen time. The evidence supports this cost decomposition; it does not estimate the counterfactual latency of eliminating all rebuilds while keeping DLS unchanged.

Separating prefill and decoding is also important in serving (Zhong et al., 2024; Agrawal et al., 2024). Compute and memory bounds further distinguish operation counts from latency (Williams et al., 2009).

E Sensitivity to the Diffusion-Cache Interval

This appendix isolates CADD and VocabBypass under a fixed selected policy. In Figure A2, *Rewrites off/on* disables/enables both components without changing that policy. Main Figure 5 additionally uses a native Show-o2 reference (100 timepoints, mean 10.639586 s, score 0.747222)

from separate runs on the same 120-prompt/three-seed workload. Its native-relative ratios therefore differ from the fixed-policy ratios below and do not imply paired native output equality.

Changing the cache interval changes which Head-A outputs are reused and thus changes the underlying approximate generation policy. We evaluate $T_c \in \{1, 2, 3, 5, 8\}$ on the predeclared balanced subset of 120 GenEval prompts, with 20 prompts per category and seeds zero through two. All settings use Show-o2-1.5B, 70 denoising steps, CFG 7.5, 432×432 resolution, and batch size one. Within each interval, both arms share these settings and differ only in CADD and VocabBypass. The control measures their additional execution savings, separately from the full system's native-relative tradeoff. Related caches vary reuse by block or timestep (Chen et al., 2024d; Zhao et al., 2025; Kahatapitiya et al., 2025).

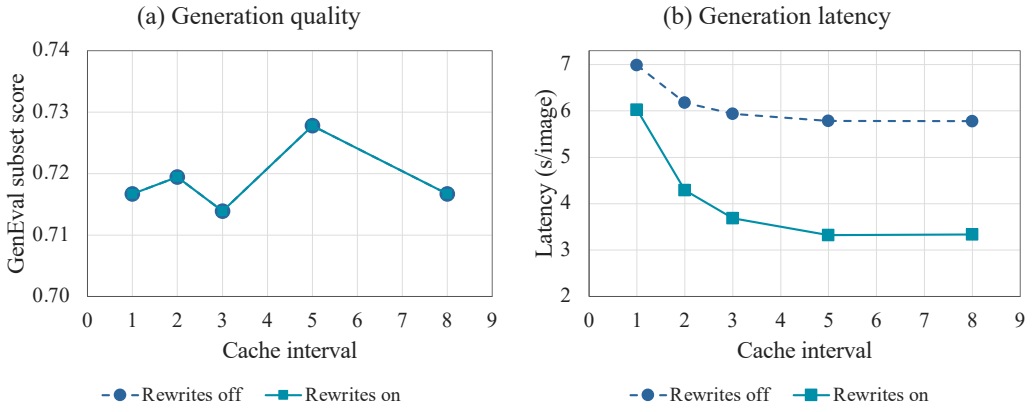


Figure A2: **Fixed-policy component sensitivity.** CADD and VocabBypass off/on: mean GenEval subset score and latency over three seeds. Quality curves coincide within this control; neither arm is native Show-o2.

Figure A2 and Table A10 draw on the same verified seed-level summaries. The 30 seed/method/interval runs contain 3,600 formal timings. Each run excludes its disjoint calibration input and warm-up; pilots and diagnostic traces are also excluded. At every interval, all 360 formal paired PNGs match, giving 1,800/1,800 matches overall. Only after this equality check do we reuse the officially scored reference images to report the rewrites-on quality. Both arms use the inherited FlashU policy (Ke et al., 2026); the rewrites only remove computations that no requested output or required state consumes. Table A10 retains seed variation and the matched latency ratios. Equality holds within each fixed interval, seed, and prompt, not across intervals.

Table A10: **Fixed-policy component control.** "Rewrites" denotes CADD and VocabBypass. Quality and latency retain their three-seed variation.

Cache interval	Rewrites off latency (s)	Rewrites on latency (s)	Speedup on vs. off	GenEval score
1	6.986 ± 0.013	6.028 ± 0.018	$1.159\times$	0.717 ± 0.014
2	6.180 ± 0.008	4.294 ± 0.004	$1.439\times$	0.719 ± 0.017
3	5.939 ± 0.011	3.685 ± 0.010	$1.612\times$	0.714 ± 0.035
5	5.785 ± 0.017	3.323 ± 0.004	$1.741\times$	0.728 ± 0.017
8	5.780 ± 0.016	3.335 ± 0.006	$1.733\times$	0.717 ± 0.030

The reported uncertainty is the sample standard deviation across three seed-level summaries. For seed means z_s, it is $\sqrt{\sum_{s=1}^{3}(z_s - \bar{z})^2/2}$. This statistic describes seed variation, not process-repeat uncertainty or a confidence interval. Because each seed has the same number of prompts, the pooled latency mean equals the mean of seed means. The accompanying CSV retains each seed's mean and quality score, including the data underlying the mean-only curves in Figure A2.

Enabling the two rewrites reduces mean latency at all five evaluated intervals. At interval one, every step is a refresh, so no CADD cache hit can be accepted; the $1.159\times$ gain must not be attributed to CADD. The remaining image-only projection bypass still applies. Larger intervals offer more reuse opportunities, although the fixed full-network region and recalibration guards limit them. The

measured ratios are $1.741\times$ at interval five and $1.733\times$ at interval eight. These observations establish neither a monotone relationship nor an optimal interval. Quality differences between intervals are small relative to the observed seed variation. These subset results do not replace the full GenEval scores in Table A4. Alternative accelerations change sampling (Song et al., 2021; Lu et al., 2022) or distill models (Salimans & Ho, 2022; Luo et al., 2023); our comparison fixes both.

F Qualitative Comparison with Native Show-o2

Figure A3 compares native Show-o2-1.5B with the complete DepGraph-U system on four GenEval prompts. Each pair uses the same rewritten prompt and seed label zero at 432×432 resolution. The native and system schedules use 100 and 70 timepoints, respectively. These are preserved outputs from separate historical runs, so the shared seed label does not establish identical initial noise across runs. All eight original PNGs are retained without image editing or regeneration.



Figure A3: **Native Show-o2 versus full DepGraph-U.** Four GenEval examples with matching prompt and seed labels from separate archived runs. Outputs differ; the selected cases do not estimate aggregate quality.

What the examples show. Prompt 00451 asks for a donut below a cat, 00514 for a green cup and yellow bowl, 00548 for a yellow bicycle and red motorcycle, and 00506 for a purple keyboard and blue scissors. The paired images differ in composition, object geometry, and local detail. Seven of the eight images pass their recorded GenEval conditions. For prompt 00514, the native image fails because the evaluator detects no bowl; the system image passes. This individual outcome does not establish an aggregate quality advantage, and detector-based checks do not certify every visual attribute or human preference. Compositional and preference metrics assess different aspects of generation (Huang et al., 2023; Hu et al., 2023; Xu et al., 2023). The display uses short prompt labels; the evidence records retain the full rewritten inputs.

System quality and component exactness. The complete system includes the inherited pruning/cache policy as well as CADD and VocabBypass. Its outputs and benchmark quality can therefore differ from native Show-o2. Table A4 reports the full-bank category scores, including the quality reductions, while the main generation tables report the corresponding native-relative latency. The fixed-policy checks in Appendix E address a separate question: whether the execution rewrites introduce additional output changes after the generation policy is fixed. Their matching images do not imply equality between the complete system and native Show-o2.

Selection and coverage. These four cases come from an existing six-case visual shortlist and are illustrative, not a random sample or an estimate of the frequency of better or worse outputs. They

neither establish that every system output is worse nor substitute for the full GenEval score. The separate-run design also prevents attributing an individual visual change to one component. No per-image speedup is assigned here. Prompt identifiers, official outcome records, and file hashes for both arms accompany the figure; the earlier fixed-policy case bank remains separately archived for its original equality checks.

G UQP: Offline Selection and Cross-Seed Transfer

G.1 A historical policy-selection diagnostic

UQP is a historical study of choosing a generation trajectory for each prompt, independent of the full DepGraph-U configuration. In Figure A4 and Table A11, the label *rewrites* denotes only CADD and VocabBypass. It changes the trajectory, whereas CADD and VocabBypass preserve the output of the selected trajectory under their contracts. We report UQP separately because its promising score-informed selection result loses quality when transferred to a different random seed. This bank is independent of the later main generation bank; neither its reference DPG score nor its latencies should be substituted for the main results. Query-dependent cost/quality selection is also studied in learned model cascades and routers (Chen et al., 2024c; Ong et al., 2025). UQP instead uses already scored candidate images and does not implement either learned routing method.

Candidate profiles. The finite bank comprises a 35-step accelerated reference following FlashU (Ke et al., 2026), an 18-step `quality_18` profile, and a 16-step `speed_16` profile. The two shorter profiles also change text guidance. Their time grid uses $u_i = 1 - i/N$ and $t_i = 3u_i/(1 + 2u_i)$, for $i = 0, \ldots, N$. The 18-step profile applies text-guidance scales of 4.20, 2.60, and 1.15 over $t \in (0.68, 1]$, $(0.34, 0.68]$, and $(0, 0.34]$, respectively. The 16-step profile uses 4.40, 2.40, and 1.10 over $(0.72, 1]$, $(0.38, 0.72]$, and $(0, 0.38]$. These ranges are converted into the executed step-count text-CFG schedule. The generic configuration also includes image-guidance values, but the Show-o2 text-to-image adapter has no independent image-guidance branch, so these values are not an executed mechanism. UQP is therefore not a step-count-only ablation.

G.2 Offline quality gate and frozen route map

For each of 1,065 DPG-Bench prompts, we generate and score all three candidates using seed zero. Let $q(p, s, r) \in [0, 1]$ denote the per-image score for prompt p, seed s, and route r, with r_0 denoting the 35-step reference. Using the stored route-level mean-latency constants $\bar{t}_r$, the selector chooses

$$r^*(p) = \operatorname*{argmin}_{r \in \mathcal{R}:\ q(p,0,r) \geq q(p,0,r_0)} \bar{t}_r. \tag{G.1}$$

The reference route is always feasible. The gate evaluates the same candidate images later used to report the selected quality, making this an offline oracle analysis of a finite bank rather than a deployable quality predictor. Replaying the saved selector reproduces all 1,065 assignments: 322 prompts take the reference route, 110 take the 18-step profile, and 633 take the 16-step profile.

Table A11(a) reports the fixed candidates, the selected mixture, and that mixture with CADD and VocabBypass enabled. The two globally fixed shorter profiles reduce the seed-zero score from 83.119 to 78.671 and 77.336. The score-informed mixture achieves 86.554 on the images chosen using their own scores. Adding CADD and VocabBypass preserves 1,065/1,065 selected image hashes. This equality is relative to the selected candidates, not the 35-step reference images.

Reported routed latency averages the measured time from each prompt's selected candidate record. It excludes the costs of generating the full candidate bank, scoring candidates, and constructing the route map. The seed-zero ratio of $2.192\times$ compares the oracle plus execution rewrites with the unoptimized 35-step reference. It includes both policy selection and execution changes and cannot isolate the contribution of routing alone.

G.3 Transfer to a new stochastic sample of the same prompts

The seed-one evaluation uses the same 1,065 prompts and freezes every seed-zero route. Seed-one scores are used only for evaluation, not to update the mapping. This tests cross-seed transfer on known prompts rather than generalization to unseen prompts. Quality uses all 1,065 prompts;

Table A11: **UQP selection and cross-seed transfer.** Historical candidate-bank results, separate from the main study.

(a) Seed 0: candidate evaluation and offline selection

Configuration	Steps	DPG ↑	Latency (s)	Speedup
35-step policy	35	83.119	2.9651	1.000×
UQP-quality	18	78.671	1.6764	1.769×
UQP-speed	16	77.336	1.5503	1.913×
Offline oracle	Mixed	86.554	1.9916	1.489×
Oracle + rewrites	Mixed	86.554	1.3525	2.192×

(b) Seed 1: frozen routes on the same prompts

Configuration	DPG ↑	Δ DPG	Latency (s)	Speedup
35-step + rewrites	83.711	0.000	1.8369	1.000×
Frozen routes + rewrites	80.421	-3.290	1.3512	1.359×

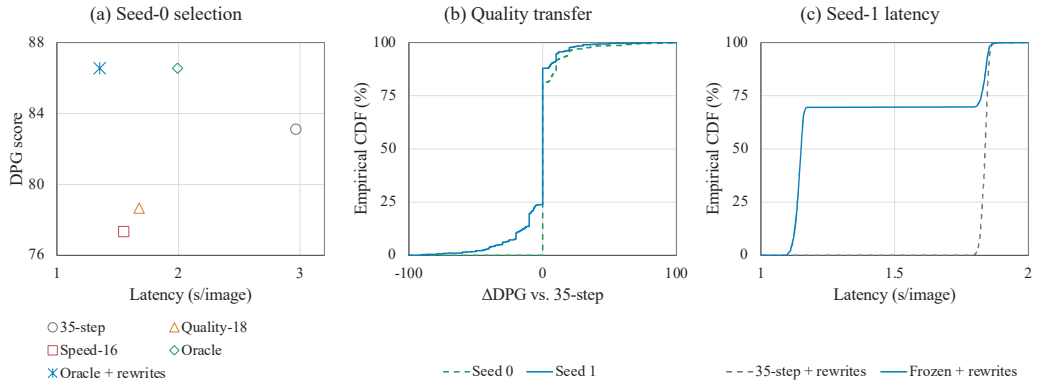


Figure A4: **Historical UQP selection and transfer.** (a) Seed-zero operating points. (b) Quality-change CDFs against the same-seed 35-step policy. (c) Seed-one latency with CADD/VocabBypass in both arms. UQP is not a DepGraph-U component.

timing excludes the common warm-up, leaving 1,064 records. Both seed-one arms in Table A11(b) use CADD and VocabBypass, keeping the execution rewrites matched.

The frozen mixture scores 80.421, compared with 83.711 for the 35-step seed-one reference, a decline of 3.290 DPG points. Its selected-inference mean is 1.3512 seconds versus 1.8369 seconds, yielding a matched ratio of 1.3595×. Frozen-route latency is reconstructed from selected candidate timings rather than measured in a separate online-router run. When a prompt selects the reference route, that timing record contributes to both distributions. Their overlapping portions therefore do not constitute independent repeat measurements.

For Figure A4(b), define $\Delta_p^{(s)} = 100[q(p, s, r^*(p)) - q(p, s, r_0)]$ in DPG points. The empirical CDF of values $z_1, \ldots, z_n$ is

$$F_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}[z_i \le x]. \tag{G.2}$$

We include all eligible records without smoothing. The fraction with lower quality is $F_n(0^-)$, not $F_n(0)$, because ties at zero are not failures. Under the selection gate, seed zero has 0 lower, 867 equal, and 198 higher scores. Seed one has 254 lower, 683 equal, and 128 higher scores, with 23.85% of prompts losing quality. The analysis records the 10^{-8} DPG-point tolerance used to distinguish numerical ties.

Interpretation and limits. The offline gate enforces nondecreasing scores on images already evaluated, but the frozen map does not preserve that property on a different stochastic sample. Output-preserving execution of a chosen route cannot remedy this selection failure. A single transfer seed

cannot support a confidence interval across seeds, and the evidence establishes neither unseen-prompt nor cross-benchmark transfer. No estimate of the full offline preparation cost is available. These limits motivate reporting UQP as an appendix diagnostic, separate from DepGraph-U's dependency-aware execution contribution.

H Reproducibility and Evidence Index

H.1 What accompanies the manuscript

The source package contains the paper, editable tables and algorithms, all main and appendix figure assets, and an `evidence/` directory with compact measurement records. The filenames below are relative to that directory. `MANIFEST.json` lists the included files and SHA256 digests; `SOURCE_INDEX.json` links derived artifacts to preserved source records through anonymous relative identifiers and hashes. Model weights, benchmark datasets, and full GPU run archives are separate from this compact manuscript package. The supplied verifier checks the included records; it neither reruns the models nor replaces the official evaluators.

Original Show-o2 records. `table_data.json` preserves the original unrounded values, benchmark sizes, response-match counts, and descriptive latency statistics. `trace_examples.csv` contains one derived record for each Exact-DLS example, while `trace_length_timing.csv` contains the five measured length-group summaries. `interval_seed_details.csv` preserves all three seed-level records for each interval and method. The CSV fields retain their specified units and normalizations; model latency is not relabeled as throughput or serving latency.

Current and historical evidence. The `current_revision/` directory contains the original six-table sources and current five-table integration, primary-source extracts, unrounded measured data, BAGEL records, and Figure 5/8/9 plot data. The controlled-reuse raw records allow recomputation of all 96 heatmap cells. The figure-selection manifests identify eight main-text and four appendix PPTX inputs; all twelve editable sources are included. Splitting the original reuse slide produces separately numbered Figures 6 and 7, so the manuscript now contains nine main figures; Figure 5 uses the latest native-comparison version rather than the superseded desktop copy.

The `main_tables/` directory retains the previous six-table evidence bank: its Exact/Stable and auxiliary-generation results now support the appendix, not the numbering of the new main tables. Its `projection_kv_controls.csv` supplies the repeated controls, and `figure6_mmstar_pairs.csv` retains all 1,500 paired MMStar observations. Original cap-1,024 diagnostics and the superseded verification manifest are preserved separately.

Images and route selection. `current_revision/native_cases/` contains the four displayed native/system pairs, full prompts, outcome records, and hashes. Its manifest identifies the separate source runs. The earlier fixed-policy examples remain in `case_records.json` and `cases/`; they are not the current Figure A3. `geneval_paired_outcomes.csv` retains the 2,212 outcomes from that historical matched bank. `uqp_paired_records.csv` and `uqp_frozen_routes.json` retain quality differences, selected timing observations, and frozen route assignments. The UQP profile configuration is stored in `uqp_profiles.json`. Main-bank, interval-subset, and historical UQP results are maintained as separate data sources.

H.2 Recomputation and a fresh experimental run

Run the compact checks from the extracted source-package root using only Python's standard library:

```
python3 evidence/verify.py
```

The verifier checks manifest hashes, figure-file provenance, historical paired case images, the current native/system case hashes, category macro averages, reported latency ratios, response-match counts, phase-derived reuse counts, interval seed summaries, and UQP frozen-transfer aggregates. It also invokes `verify_main_tables.py` to check the current displayed values, matched controls, and Figure 7 observations. Historical file names and raw model identifiers are retained to preserve provenance; the figure-number map in `review_notes/FIGURE_NUMBER_MAP.json` records

the current display numbering. Its output states how many records were checked and the scope of each check. Reproducing model outputs also requires the archived model adapters, frozen pruning artifacts, original weights and datasets, and a compatible GPU environment.

A fresh experiment should begin by freezing the workload, image seeds, stopping rules, and the relevant configuration in Tables A2 and A3, or Appendix B.3 for auxiliary generation. Keep original and fresh RNG protocols and run segments distinct. Run the reference and modified arms on identical inputs, and archive per-example identities, outputs, and synchronized timings before aggregation. Generation scores must be joined to the timed image bank; a different bank requires fresh scoring. Understanding comparisons must retain their respective task caps and official answer-extraction rules. Validate image hashes or decoded responses before transferring quality scores, and compare token IDs only when both records contain them. Apply only the recorded warm-up exclusions, then recompute means and ratios from full-precision observations. A new hardware/software stack requires new timing measurements and equality checks.

H.3 Limits of the evidence

Generation process repeats, seed variation in the interval study, and full-benchmark understanding distributions capture different sources of variation. A single full understanding run does not establish run-to-run statistical significance. Observed answer-length groups are not controlled length sweeps, qualitative cases are selected examples, and UQP's offline quality gate offers no test-time quality guarantee. The native KV counterexample covers MMMU only. Lumina-DiMOO and OmniGen2 provide auxiliary generation evidence; the BAGEL study does not establish general Exact-DLS transfer. Video models, larger serving batches, and multi-user throughput remain unevaluated. Recovery-segment pairing does not establish cross-process bitwise reproducibility, and the Lumina control does not record RNG poststates. Within the measured scope, the central conclusion remains: execution should follow the values a task actually consumes, while the quality effects of policy changes require separate evaluation.

I Dependency-Aware Execution Controls and Architecture Transfer

I.1 Generation execution-cost decomposition

The component experiment fixes the selected pruning, guidance, and cache policy and uses 20 frozen DPG prompts at each Show-o2 scale, with three independent process repeats. Four arms enable neither rewrite, CADD only, VocabBypass only, or both. Timing and profiling use separate executions: the former excludes probes and hash/image I/O; the latter records nested-exclusive CUDA-stream intervals. Smoke arms verify outputs, final latents, PNGs, RNG, and actual call counts before formal measurement. The study comprises 960 formal calls across two modes and two scales, plus disjoint warm-ups.

The validated Euler schedule has 35 timepoints but 34 model calls. CADD accepts 14 safe hits and leaves 20 trunk calls; Head B remains at 34 calls and Head A at 18. VocabBypass removes all vocabulary calls when enabled and otherwise leaves one per surviving trunk call. Figure 8 reports the separate profile totals: 2.90958, 2.06612, 2.42776, and 1.78800 s at 1.5B; 6.53239, 4.34482, 5.39877, and 3.67575 s at 7B, in the four-arm order. Stream intervals include launch gaps and are not pure kernel sums. Table 4 instead reports uninstrumented means; full-precision values and run SDs remain in the source records.

I.2 Generation guard and interface validation

Replay covers normal generation branches (1,530 paired steps), forced conditional-only guidance (1,530), and zero-hit fallback (510), totaling 3,570 comparisons. All recorded prediction and future-state checks agree, with maximum prediction error zero. Checks cover the cache, refresh, full-network, dispatch eligibility, recalibration, and conditional-only conditions in Equation 1; they are not a proof over arbitrary future callers. Nine additional text/direct-Qwen-label interface records check retained logits/loss and rejection of conflicts. RNG equality was checked at runtime, without exported raw RNG states.

A separate zero-hit experiment uses cache interval one and 510 synchronized paired model calls grouped into 15 prompt–seed conditions. Aggregate overhead is $+0.0145055\%$; group results range from -0.123212% to $+0.128184\%$. Negative overhead values are retained as measured variation. This scope excludes complete image-generation pipeline time and does not establish statistical significance of the very small difference. Understanding-side KV, visual gather, mask, position, and EOS boundary validation is separate from this generation replay.

I.3 Controlled prefix length and reuse horizon

Figure 9 uses four fixed MMStar inputs, three fresh-process repeats, and prefixes of 1,024, 1,536, and 2,048 tokens. Both arms retain the same DLS policy and last-position projection; the reference recomputes the full prefix while Exact enables compatible KV. Each trajectory is forced to 32 steps by ignoring EOS in this diagnostic. It therefore tests execution cost, not benchmark response accuracy or natural stopping. All $3 \times 32 = 96$ heatmap cells are aggregated from actual measured checkpoints without interpolation. Checkpoints within a trajectory share preceding calls and are correlated; the heatmap cells are not independent experimental repeats.

Cumulative speedup through step J is the ratio of accumulated Qwen CUDA-stream times across the reference and Exact trajectories under the specified prefix. Period-10 selection and rebuilding account for the drops after checkpoints 10, 20, and 30. Near-one and slightly sub-one initial results remain in the data. These measurements isolate the reuse horizon from natural answer length but exclude the complete multimodal pipeline and do not estimate serving throughput at larger batch sizes.

I.4 BAGEL without TaylorSeer

The generation control freezes 12 original GenEval prompts, two per category, with seed 42, 1024×1024 output, 50 timepoints, bfloat16, and batch one. Four arms cross native versus selected FFN pruning with the dead-CFG-producer rewrite off versus on. Three process repeats give 144 formal images. Quality is independently scored on the first 12 images of each arm, with pixel equality verified across repeats. No TaylorSeer or public last-layer-history optimization is active. Native/adapted means are 36.520721788/32.989770527 s ($1.107031701\times$); scores are 0.750000/0.833333. These subset values do not establish a full-GenEval improvement.

BAGEL already avoids image-call vocabulary projection and supports KV, and it lacks Show-o2's identical two-part diffusion head. The applicable new rewrite skips an unconsumed image-guidance prompt producer, reducing text prefill from two calls to one while preserving within-policy outputs and RNG. This rewrite alone gives $1.000134\times$ relative to native and $1.001262\times$ relative to the fixed pruning arm; slower repeats remain in the evidence. The complete generation saving is predominantly due to the inherited approximation policy.

Understanding uses all 1,500 MMStar inputs, greedy decoding, cap 128, fixed FFN pruning, native KV, and guarded append storage. The score is the macro mean over 18 L2 categories, not per-question accuracy. Native/adapted scores are 68.013468941/66.692480371; latencies are 0.271206171/0.235106319 s, yielding $1.153546921\times$. Only 1,157/1,500 answers match, and timing includes output-length changes. An earlier 553-prompt generation run contained TaylorSeer and other existing optimizations; its $2.488277\times$ ratio is excluded from Table 1. No complete 553-prompt, no-TaylorSeer result is claimed.

I.5 Earlier generation-transfer evidence

These earlier component studies retain their own cached references and are separate from the native-relative main tables. TaylorSeer and ML-Cache are existing state-selection or prediction policies, not DepGraph-U contributions; the new contribution is dispatching around producers whose outputs those selected states make unnecessary. Lumina-DiMOO uses discrete multimodal diffusion with ML-Cache (Xin et al., 2025); VocabBypass retains image-token logits while omitting the unused text-vocabulary part. OmniGen2 uses decoupled text/image decoding (Wu et al., 2025a); CADD bypasses upstream refiners on accepted TaylorSeer prediction steps (Liu et al., 2025b). This TaylorSeer use does not apply to the current BAGEL generation block.

Table A12: **VocabBypass on cached Lumina-DiMOO.** Paired outputs and independently scored quality match.

Method	Quality	Time (s) ↓	Speedup ↑
	GenEval		
ML-Cache	0.87753	29.311	1.000×
+VocabBypass	0.87753	**27.599**	**1.062**×
	DPG-Bench		
ML-Cache	84.140	29.598	1.000×
+VocabBypass	84.140	**27.888**	**1.061**×

Table A13: **CADD on cached OmniGen2.** The reference uses last-layer-only history and TaylorSeer.

Method	Quality	Time (s) ↓	Speedup ↑
	GenEval		
TaylorSeer	0.76254	10.695	1.000×
+CADD (Ours)	0.76254	**9.581**	**1.116**×
	DPG-Bench		
TaylorSeer	84.203	10.826	1.000×
+CADD (Ours)	84.203	**9.693**	**1.117**×

Both use batch one, 1024×1024 images, 553 original GenEval prompts with four seeds, and 1,065 original DPG prompts with one image per prompt. Lumina uses 64 steps, CFG 4, and ML-Cache ratio 0.9/warm-up 0.3/refresh interval 5. OmniGen2 uses 50 steps, text/image CFG 5/1, and TaylorSeer threshold 3/first-enhance 5/order 4. Pipeline wall time excludes loading, saving, equality checks, and warm-up. The DPG CSV loader is corrected in both arms to retain the first data row and all 14,392 answers; this differs from the four-image-grid leaderboard protocol. Recovery segments retain within-process pairing, but OmniGen2 failed an additional cross-process bitwise-reproducibility check despite matching weights. No global cross-process image-identity claim is made.

A Lumina 2×2 control (eight prompts, three repeats, 96 calls) gives $1.0635\times$ without ML-Cache and $1.0637\times$ with it; all 24 blocks in each comparison are faster. RNG resets were recorded, but poststates were not exported. These small gains and the near-zero BAGEL rewrite gain show that transfer depends on which unused operators remain in the native architecture.

I.6 Source and display provenance

The source package preserves accepted table values and primary-source extracts, as well as the native Figure 5 comparison and measured Figure 8/9 plot data. Literature quality and latency may come from separate evaluations: Emu3 MME-P is a released-checkpoint evaluation, while cross-model latency context follows the archived source measurements of Ke et al. (2026). Such rows do not provide matched speedup denominators. Missing cells remain unreported, not estimated. Params. denotes base-language-model size; GenEval scores are in $[0, 1]$, other quality scores are percentages except MME-P. Display rounding does not alter underlying ratios.