This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Accelerating Diffusion via Hybrid Data-Pipeline Parallelism Based on Conditional Guidance Scheduling

Euisoo Jung Byunghyun Kim Hyunjin Kim Seonghye Cho Jae-Gil Lee*
KAIST
{jyssys, rooknpown, hyunjin, orangingq, jaegil}@kaist.ac.kr

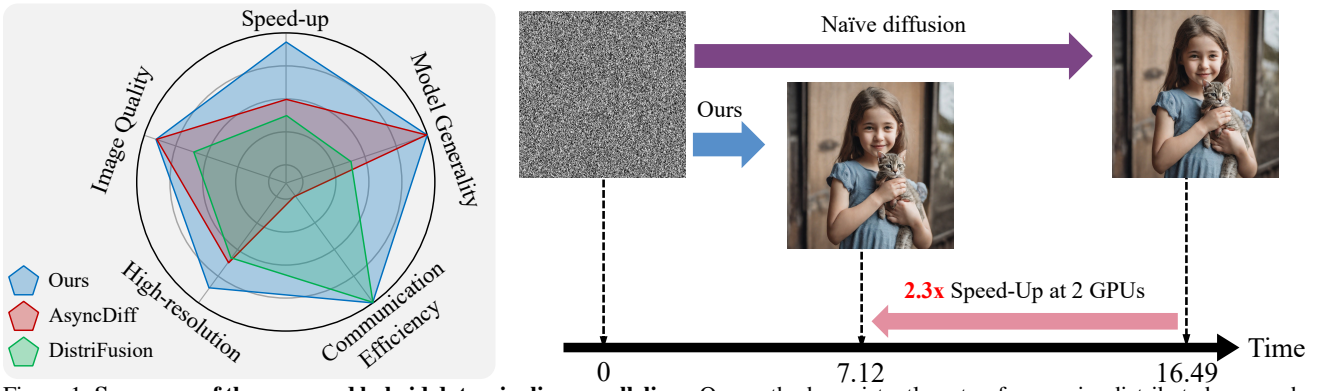

Figure 1. **Summary of the proposed hybrid data-pipeline parallelism.** Our method consistently outperforms prior distributed approaches across five key aspects: *Speed-up*, *Image Quality*, *Generality*, *High-resolution Synthesis*, and *Communication Cost*, demonstrating robust and balanced acceleration-quality trade-offs.

Abstract

Diffusion models have achieved remarkable progress in high-fidelity image, video, and audio generation, yet inference remains computationally expensive. Nevertheless, current diffusion acceleration methods based on distributed parallelism suffer from noticeable generation artifacts and fail to achieve substantial acceleration proportional to the number of GPUs. Therefore, we propose a hybrid parallelism framework that combines a novel data parallel strategy, condition-based partitioning, with an optimal pipeline scheduling method, adaptive parallelism switching, to reduce generation latency and achieve high generation quality in conditional diffusion models. The key ideas are to (i) leverage the conditional and unconditional denoising paths as a new data-partitioning perspective and (ii) adaptively enable optimal pipeline parallelism according to the denoising discrepancy between these two paths. Our framework achieves $2.31\times$ *and* $2.07\times$ *latency reductions on SDXL and SD3, respectively, using two NVIDIA RTX 3090 GPUs, while preserving image quality. This result confirms the generality of our approach across U-Net-based diffusion models and DiT-based flow-matching architectures. Our approach also outperforms existing methods in acceleration under high-resolution synthesis settings. Code is available at https://github.com/kaist-dmlab/Hybridiff.*

* indicates corresponding author.

1. Introduction

Diffusion models have emerged as a powerful family of generative models because of their superior sample quality and broad applicability. However, the inherently iterative nature of diffusion processes, which consists of many denoising steps, leads to significant inference latency and computational bottlenecks. As model sizes continue to scale, these inefficiencies become increasingly limiting, making diffusion inference acceleration a pressing research challenge. Existing approaches have focused mainly on reducing the number of sampling steps [9, 19, 20, 22, 27, 28, 35, 37, 38], designing optimal architectures [11, 13, 14, 36, 39, 40], or leveraging mathematical approximations [1, 17, 18, 23, 29, 41, 41]. Yet, these methods often require additional training or fail to deliver strong acceleration in practice, exhibiting a clear trade-off between generation quality and speed.

Distributed parallelism across multiple GPUs offers a promising alternative. Using modern parallel computing resources, one can achieve substantial throughput improvements in diffusion inference without additional training. This direction is especially appealing given the success of distributed strategies in natural language processing, where large-scale language models have already benefited from extensive parallelism research [25, 30]. As in other domains, distributed parallelism for generative model inference can be broadly classified into *data parallelism*

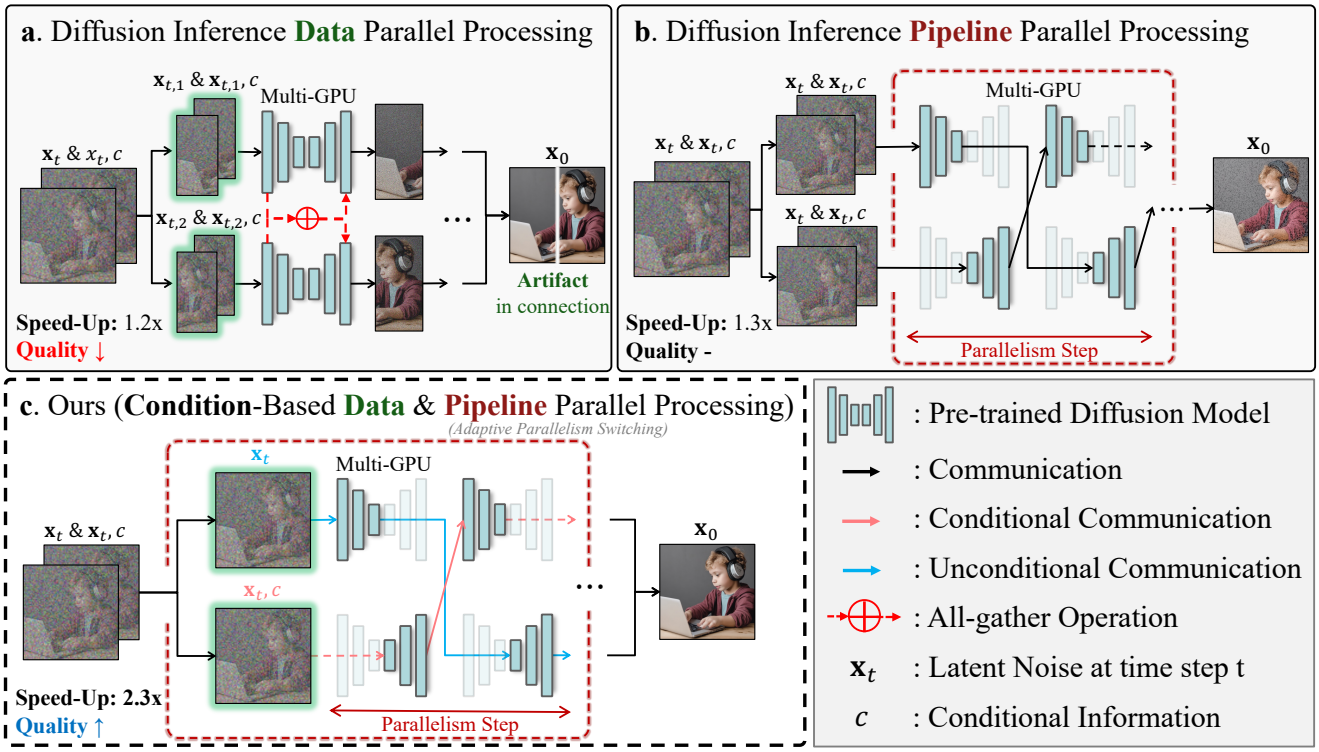

Figure 2. **Comparison of parallel strategies for diffusion inference.** (a) Patch-based data parallel frameworks suffer from bottlenecks caused by all-gather operations and artifacts at patch boundaries, leading to limited acceleration and quality degradation. (b) Pipeline parallel frameworks incur excessive asynchronous communication overhead and accumulate estimate errors. (c) Our **hybrid parallelism**, which incorporates condition-based data parallelism, adaptively combines both paradigms to achieve high fidelity and fast generation.

and *pipeline parallelism* [2, 12]. Both approaches enhance throughput by distributing either the input data or the model itself across multiple GPUs.

Representative existing studies include DistriFusion [12] for data parallelism and AsyncDiff [2] for pipeline parallelism. In DistriFusion (Figure 2a), an input image is divided into N disjoint patches, and these patches are processed in parallel across N GPUs, where each device independently handles one patch. In AsyncDiff (Figure 2b), the entire model is divided into N sequential components, where each component is assigned to a GPU, and the output from the i-th GPU is asynchronously fed as the input to the $(i+1)$-th GPU; thus, AsyncDiff enables pipelined execution across devices.

In theory, each form of parallelism can improve throughput linearly with respect to the number of GPUs, up to an ideal $N\times$speed-up, but in practice, the gains are often sublinear due to communication overhead and synchronization costs. In this paper, we propose a *hybrid* strategy that combines data and model parallelism to further increase the throughput of generative model inference, achieving *beyond-linear scaling* relative to the number of GPUs, while maintaining generation quality. That is, if there are two GPUs, we aim to obtain more than a twofold speed-up without noticeable degradation in output fidelity. In practice, when using two GPUs, data and model parallelism achieved $1.2\times$ and $1.3\times$speed-up, respectively, whereas our hybrid approach remarkably achieved a $2.3\times$speed-up under the same configuration, as shown in Figures 1 and 2.

To achieve hybrid parallelism, one could combine the aforementioned representative methods. Specifically, an image is divided into disjoint patches, and each patch is fed into a corresponding model component (not necessarily the first one). As a result, each GPU trains a $1/N$ portion of the model using a $1/N$ portion of an input image. This hybrid approach can potentially achieve beyond-linear scaling; however, it may degrade generation quality for two main reasons. First, since each GPU processes only a portion of the image, artifacts are likely to appear particularly along patch boundaries. Second, this issue is exacerbated by asynchronous communication between model components; that is, errors introduced by asynchronous rather than sequential denoising can worsen the artifacts.

In this paper, we aim to propose and further optimize the hybrid parallelism for diffusion inference from two complementary perspectives: (1) from the data parallelism perspective, transitioning from patch-based partitioning to *condition-based partitioning*; and (2) from the model parallelism perspective, advancing from static parallelism switching to *adaptive parallelism switching*.

(1) Condition-Based Partitioning. The main limitation of patch-based partitioning is that each patch represents only a *local* subregion of an image, often leading to boundary artifacts and degraded visual coherence. To address this limitation, we leverage the classifier-free guidance (CFG) [7], a technique widely adopted in diffusion models, where the model simultaneously predicts *conditional (prompted)* and *unconditional (unprompted)* noise estimates. This *dual-path* prediction naturally provides a meaningful criterion for data partitioning: as shown in Figure 2c, the conditioned $(\mathbf{x}_t, c)$ and unconditioned $(\mathbf{x}_t)$ inputs form two distinct data-parallel paths. Importantly, unlike patch-based partitioning, each image partition covers the *entire* image, thereby preserving global consistency. Consequently, condition-based partitioning yields improved visual coherence and reduced communication overhead during feature aggregation.

(2) Adaptive Parallelism Switching. Because we revise the data partitioning strategy, the pipeline parallelism must also be adapted to align with it. In the early denoising steps, the conditional and unconditional noise estimates differ substantially due to the presence or absence of the condition. Consequently, asynchronous denoising at this stage can lead to divergence between the two paths. To mitigate this issue, we defer the onset of parallel execution until the noise estimates of the two paths become sufficiently similar, beyond the conventional warm-up phase used in prior works (e.g., [2]). Similarly, toward the final denoising steps, the noise estimates from the two paths begin to diverge again; at this point, parallel execution is terminated. The specific switching points between serial and parallel execution are determined automatically based on a novel metric, called the *denoising discrepancy*, which quantifies the difference between the two noise estimates. This *adaptive* switching mechanism effectively improves generation quality by reducing error propagation, while only marginally shortening the duration of parallel processing.

This novel framework demonstrates consistent acceleration not only on conventional denoising diffusion models but also on recent state-of-the-art generative frameworks such as flow matching [16]. As long as the model follows a sequential denoising process that allows quantifying the relative influence between conditional and unconditional branches, our framework remains robust and effective. Furthermore, due to the nature of pipeline parallelism, it is not restricted to specific architectures such as U-Net or DiT, showing strong generality across diverse networks.

As summarized in Figure 1, our proposed *hybrid parallelism* achieves superior performance across the five key aspects. In fact, compared to single-GPU inference, our method achieves a **2.3×speed-up with two GPUs** (i.e., > 2), while preserving generation fidelity. See Appendix A and Section 5 for details of Figure 1. Finally, the key contributions are summarized as follows.

- **Hybrid Parallelism Framework for Diffusion Inference.** We introduce a novel diffusion inference parallelism framework that integrates condition-based partitioning and adaptive parallelism switching into a unified hybrid parallelism design.
- **Novel Condition-Based Partitioning.** At the data parallelism level, we exploit the intrinsic mechanism of diffusion by decoupling conditional and unconditional branches and performing multi-GPU denoising.
- **Adaptive Parallelism Switching.** To align pipeline parallelism with the behavior of conditional guidance, our method adaptively switches to hybrid parallelism framework during inference. Switching points are automatically determined based on the denoising discrepancy between conditional and unconditional estimates, ensuring generation efficiency throughout the denoising process.
- **Robustness across Models and Architectures.** Our framework consistently demonstrates strong acceleration and generation quality across various architectures (e.g., U-Net, DiT) and recent state-of-the-art generative frameworks, such as flow matching, even under high-resolution synthesis settings.

2. Related Work

***Single*-GPU Diffusion Acceleration.** Research on the acceleration of *single*-device diffusion inference can be classified into three categories. The first group focuses on reducing the number of sampling steps required for high-quality generation [9, 19, 20, 22, 27, 28, 31, 35, 37, 38, 41]. These approaches enable fast sampling by either reformulating the reverse process as an ordinary differential equation (ODE), distilling multi-step models into fewer steps, or directly predicting the reverse process in latent space. The second group targets model architecture optimization, aiming to reduce computational cost through network compression and efficient design [11, 13, 14, 36, 39, 40]. The third group leverages mathematical and algorithmic strategies, either exploiting the mathematical structure of diffusion processes or reusing intermediate computations to further accelerate inference [1, 17, 18, 23, 29, 34, 41]. While these methods reduce single-device inference time, they are inherently limited by the computational capacity of individual GPUs.

***Multi*-GPU Diffusion Acceleration.** Recent studies have explored various distributed parallelism strategies to accelerate diffusion inference using *multiple* GPUs [2, 4, 5, 12, 33]. DistriFusion [12] introduces a data-parallel approach that divides the input image into independent patches, performing denoising in parallel across GPUs. This work has established a foundational paradigm for parallel diffusion inference. Building on this parallelization idea, AsyncDiff,[2] introduces model parallelism by dividing the U-Net into layer-wise segments and employing a stride-

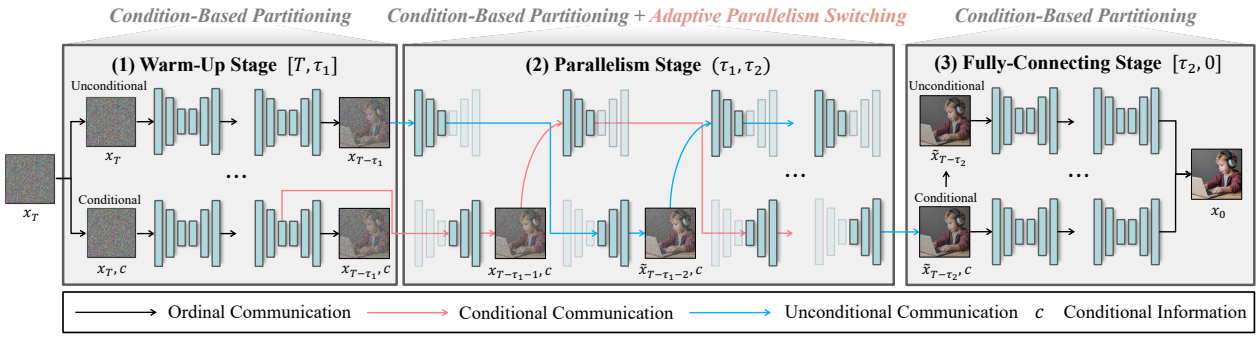

Figure 3. **Overview of the proposed diffusion inference hybrid parallel framework.** Our method adaptively switches parallelism modes at τ_1 and τ_2, optimizing the trade-off between computational efficiency and consistency of conditional guidance, and demonstrates superior inference acceleration performance while preserving high generation quality.

based scheduling strategy to balance parallel execution, achieving a notable reduction in latency.

Subsequently, PipeFusion [5] and XDiT [4] combine patch-level parallelism with transformer-oriented parallelism through ring attention. While additional adaptations such as CFG-based data parallelism have been introduced, these methods remain limited to inter-image processing and lack deeper architectural integration. Moreover, transformer-specific schemes such as ring attention exhibit limited scalability and inconsistent performance when applied to general diffusion architectures. More recently, ParaStep [33] proposes a reuse-then-predict mechanism that leverages the similarity of noise predictions between adjacent denoising steps. By reusing the noise from previous steps before re-prediction, ParaStep enables inter-step parallelization and significantly reduces communication overhead. However, because early and late diffusion steps exhibit larger discrepancies between adjacent noise states, the reuse mechanism can accumulate errors, leading to potential degradation in image quality or restricted speedup.

3. Preliminaries

Denoising Diffusion Model. Let $q(\mathbf{x}_0)$ denote the data distribution and define a forward noising process by

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\big(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \mathbf{I}\big),$$

for $t = 1, \ldots, T$, with variance schedule $\{\beta_t\}$. The model learns a parameterized reverse denoising process,

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\big(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(t)\big),$$

by optimizing the variational lower bound,

$$\mathcal{L}_{\text{VLB}} = \mathbb{E}_q\Big[\sum_{t=1}^{T} D_{\text{KL}}\big(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \,\|\, p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\big)\Big].$$

Classifier-Free Guidance (CFG). For conditional generation with a condition c, the model is trained to predict the noise $\epsilon_\theta(\mathbf{x}_t, t, c)$ and its unconditional variant $\epsilon_\theta(\mathbf{x}_t, t, \varnothing)$. At inference, the samples follow

$$\epsilon_{\text{cfg}} = \epsilon_\theta(\mathbf{x}_t, c, t) + w\big(\epsilon_\theta(\mathbf{x}_t, c, t) - \epsilon_\theta(\mathbf{x}_t, t)\big),$$

where $w > 0$ is the guidance scale. The adjusted reverse mean becomes

$$\mu_{\text{cfg}}(\mathbf{x}_t, t, y) = \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_{\text{cfg}}\Big).$$

Flow Matching. Given a target distribution $q(\mathbf{x})$ and base distribution $p_0(\mathbf{x})$, flow matching defines an ordinary differential equation,

$$\frac{d\mathbf{x}(t)}{dt} = v(\mathbf{x}(t), t),$$

where the vector field v_θ is learned by minimizing

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t, \mathbf{x}_0 \sim q}\Big\| v_\theta(\mathbf{x}_t, t) - \frac{\mathbf{x}_t - \mathbf{x}_0}{t} \Big\|^2,$$

with $\mathbf{x}_t = \mathbf{x}_0 + t\mathbf{e}$ for $\mathbf{e} \sim \mathcal{N}(0, \mathbf{I})$. Sampling proceeds by integrating $\dot{\mathbf{x}} = v_\theta(\mathbf{x}, t)$ from $t = 1$ to $t = 0$.

4. Method

4.1. Overview

Figure 3 illustrates the overall process of our proposed hybrid parallelism framework. The input isotropic noise latent $\mathbf{x}_T$ is fed simultaneously into two denoising branches: the unconditional path $f_\theta(\mathbf{x}_t, t)$ and the conditional path $f_\theta(\mathbf{x}_t, c, t)$ guided by a textual prompt c. where f_θ denotes the denoising diffusion network parameterized by θ (e.g. U-Net, DiT). To exploit both global consistency and conditional fidelity, our framework incorporates two complementary dimensions of parallelism, *condition-based partitioning*, and *adaptive parallelism switching*.

Formally, given the denoising model f_θ, the diffusion inference across N devices can be expressed as

$$\mathbf{x}_{t-1}^{(n)} = f_{\theta^{(n)}}(\mathbf{x}_t^{(n)}, c^{(b_n)}, t),$$
$$n \in \{1, \ldots, N\}, \quad b_n \in \{\text{cond}, \text{uncond}\},$$

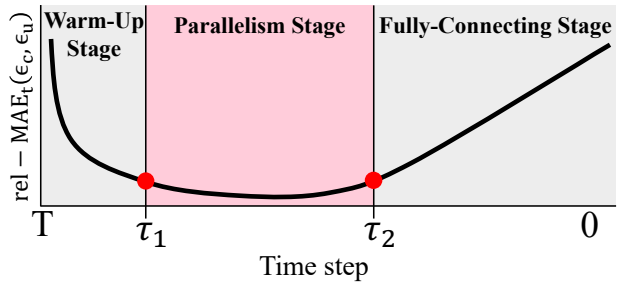

Figure 4. **Illustration of the rel-MAE$_t(\epsilon_c, \epsilon_u)$ curve.** The rel-MAE$_t(\epsilon_c, \epsilon_u)$ value is relatively large before τ_1 and after τ_2, while it converges near zero between them, indicating stable alignment between conditional and unconditional branches during the parallelism phase.

where each $\theta^{(n)}$ corresponds to the subset of model parameters assigned to the n-th device in the pipeline, reflecting adaptive parallelism switching across different network stages. Meanwhile, $b_n \in \{\text{cond}, \text{uncond}\}$ indicates whether the device n handles the conditional or unconditional branch in condition-based partitioning. Accordingly, each device processes either a conditional input with c or an unconditional input without c. This formulation jointly represents both condition-based partitioning and adaptive parallelism switching within a unified diffusion framework.

To further enhance performance, the denoising process is divided into three stages according to the temporal dynamics of conditional influence: (1) *Warm-Up Stage*, where only ordinal communication occurs between conditional and unconditional branches; (2) *Parallelism Stage*, where both branches are executed in parallel with conditional exchange; and (3) *Fully-Connecting Stage*, which merges the two branches for the final refinement. The rationale for this three-phase division and the quantitative criteria for determining the boundary points τ_1 and τ_2 are discussed in Section 4.2 and Section 4.3, respectively.

4.2. Hybrid Parallel Inference Framework

Figure 4 illustrates the relative-Mean Absolute Error of the predicted noise (relative-MAE; rel-MAE$_t(\epsilon_c, \epsilon_u)$) across the three stages of the proposed hybrid parallelism in the denoising diffusion model. To determine when the conditional and unconditional branches should interact or remain independent, we first quantify their *denoising discrepancy* during the denoising process.

Since conditional and unconditional denoisers contribute differently to generation, with one emphasizing semantic alignment to the text condition and the other stabilizing global structure, it is essential to measure how their noises ϵ_c and ϵ_u diverge over time. This discrepancy serves as a key indicator for determining the switching points between serial and parallel execution within our hybrid framework.

The *denoising discrepancy*, namely rel-MAE$_t(\epsilon_c, \epsilon_u)$, quantifies the difference in noise prediction ϵ_t between the conditional and unconditional branches at each timestep t, (where $\epsilon_c = \epsilon_\theta(\mathbf{x}_t, c, t)$ and $\epsilon_u = \epsilon_\theta(\mathbf{x}_t, t)$), and is formulated as

$$\text{rel-MAE}_t(\epsilon_c, \epsilon_u) = \frac{\mathbb{E}_{\mathbf{x},\epsilon}[\|\epsilon_\theta(\mathbf{x}_t, c, t) - \epsilon_\theta(\mathbf{x}_t, t)\|_1]}{\mathbb{E}_{\mathbf{x},\epsilon}[\|\epsilon_\theta(\mathbf{x}_t, t)\|_1]}. \quad (1)$$

Here, $\epsilon_\theta(\mathbf{x}_t, c, t)$ and $\epsilon_\theta(\mathbf{x}_t, t)$ denote the noise components predicted from the conditional and unconditional denoisers, respectively. A larger value indicates a stronger discrepancy between the two branches, reflecting a higher conditional influence on the denoising trajectory at that timestep.

According to the trend of denoising discrepancy shown in Figure 4, which exhibits a U-shaped curve over the entire denoising process, we divide the process into three stages: the *Warm-Up Stage* $[T, \tau_1]$, the *Parallelism Stage* (τ_1, τ_2), and the *Fully Connecting Stage* $[\tau_2, 0]$. The two parameters, τ_1 and τ_2, define the boundaries between these stages and are automatically determined during the middle of the denoising process. The details of how they are determined are provided in the next section. Intuitively, τ_1 marks the point where the denoising discrepancy ceases to decrease rapidly, while τ_2 indicates the point where it begins to increase.

By measuring denoising discrepancy across 5,000 prompts from the MS-COCO 2014 validation set [15], we observed that the variation of the error between the conditional and unconditional branches exhibits a clear U-shaped trend, as further demonstrated in Appendix B.

We now describe each denoising stage in detail.

(1) Warm-Up Stage $[T, \tau_1]$. This stage captures the global outline of the generated image. The conditional branch establishes the overall composition from the text prompt, while the unconditional branch stabilizes the coarse structural forms. Since both branches have distinct influences, the denoising discrepancy remains low. Therefore, each branch is processed independently using condition-based partitioning, without adaptive parallelism switching.

(2) Parallelism Stage (τ_1, τ_2). At this phase, the model refines local details within the preformed outline. The conditional and unconditional branches begin to converge, and the denoising discrepancy remains small and stable. To take advantage of this convergence, adaptive parallelism switching is activated, enabling a more powerful acceleration of the denoising process.

(3) Fully-Connecting Stage $[\tau_2, 0]$. In the final phase, fine-grained conditional cues dominate generation. The framework reverts to condition-based partitioning, integrating conditional guidance to reconstruct the final image $\mathbf{x}_0$.

A similar three stages structure has also been observed in previous studies on diffusion conditional guidance [10], which further supports the validity of our framework. Building upon this, through this three stages hybrid parallelism framework, our method achieves efficient distributed denoising while preserving generation quality.

The denoising discrepancy, rel-MAE$_t(\epsilon_c, \epsilon_u)$ can be extended to flow matching models by replacing ϵ_θ with the predicted velocity v_θ. In this case, rel-MAE$_t(v_c, v_u)$ maintains the same role in quantifying the conditional-unconditional discrepancy over the velocity field.

4.3. Adaptive Switching via Denoising Discrepancy

The core of the proposed hybrid parallelism is to dynamically determine the timesteps τ_1 and τ_2 during the real-time denoising process, sequentially switching between the Warm-Up, Parallelism, and Fully-Connecting modes, while constructing a scheduling method based on the previously defined denoising discrepancy.

(1) Determining τ_1. For each timestep t, we compute denoising discrepancy and calculate the average slope of the most recent L steps by

$$G_t = \frac{M_t - M_{t-L}}{L}. \tag{2}$$

We then select $\tau_1' = \min\{t \,|\, 0 \le G_t < g_{\text{slope}}\}$ and constrain it by safety-cap τ_{cap}. As shown in Appendix B, τ_{cap} is defined as the global minimum point of the denoising discrepancy curve, and it serves as an upper bound for τ_1 during automatic selection. The introduction of τ_{cap} ensures stability by covering cases where τ_1 is assigned too late or remains undefined due to outlier behaviors, thus maintaining generation quality while maximizing acceleration.

Consequently, τ_1 is given by $\tau_1 = \min(\tau_1', \tau_{\text{cap}})$, which marks the end of the warm-up stage where conditional influence stabilizes.

(2) Determining τ_2. During the parallelism phase, ϵ_c and ϵ_u converge to an identical value, making the denoising discrepancy measurement no longer meaningful. Therefore, τ_2 is empirically fixed to a certain number of steps k after τ_1,

$$\tau_2 = \tau_1 + k, \quad k \in \mathbb{N},\ 1 \le k < T - \tau_1. \tag{3}$$

A larger k extends the parallelism phase, resulting in faster inference but lower generation quality, while a smaller k improves fidelity at the cost of latency. A detailed analysis of quality and speed trade-offs with respect to k is presented in Section 5.4, where we empirically verify the optimal balance across various k. We also provide the algorithm of Section 4.3 in Appendix C describes the overall process.

4.4. Theoretical Analysis of Adaptive Switching

Analysis of Denoising Discrepancy by Score Decomposition. The denoising discrepancy can be theoretically interpreted as a ratio between the conditional information strength and the unconditional data prior. From the score-decomposition perspective [8, 32], can be approximated as

$$\text{rel-MAE}_t(\epsilon_c, \epsilon_u) = \frac{\|\epsilon_c - \epsilon_u\|_1}{\|\epsilon_u\|_1} \approx \frac{\|\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)\|_1}{\|s_u(\mathbf{x}_t, t)\|_1}. \tag{4}$$

$\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)$ represents the conditional information strength and $s_u(\mathbf{x}_t, t)$ denotes the unconditional score of the data distribution. Consequently, denoising discrepancy measures the relative magnitude between conditional and unconditional components.

In the score formulation of Eq. (4), the unconditional score $s_u(\mathbf{x}_t, t) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$ captures the intrinsic structure of the data distribution, while the conditional gradient $\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)$ encodes the semantic influence of the conditioning signal c. Their relative magnitudes evolve naturally along the diffusion process:

- **Warm-Up Stage:** When $\mathbf{x}_t$ is close to pure noise, $s_u(\mathbf{x}_t, t)$ carries little structural information, whereas $\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)$ dominates by guiding the global semantic layout from the prompt, leading to a large denoising discrepancy.
- **Parallelism Stage:** As denoising progresses, $s_u(\mathbf{x}_t, t)$ reconstructs meaningful local structures and becomes comparable in magnitude to $\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)$. This balance satisfies $\|s_u(\mathbf{x}_t, t)\| \approx \|\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)\|$, yielding $\frac{d}{dt}\text{rel-MAE}_t(\epsilon_c, \epsilon_u) \approx 0$ and motivates the activation of the parallel inference phase.
- **Fully-Connecting Stage:** At high SNR, most patterns have been recovered by $s_u(\mathbf{x}_t, t)$, while $\nabla_{x_t} \log p(c|\mathbf{x}_t)$ contributes to fine-grained alignment and texture refinement, causing a mild increase in denoising discrepancy.

This interpretation provides an intuitive explanation of how the relative magnitudes of the conditional and unconditional scores evolve across timesteps, theoretically supporting the three stages proposed (Warm-Up $\rightarrow$ Parallelism $\rightarrow$ Fully Connecting). Detailed derivations of Eq. (4) and the robustness analysis of τ_1 under stochastic denoising noise are shown in Appendix D and Appendix E, respectively.

4.5. Extensibility to Many GPUs Configurations

While the hybrid parallelism framework is optimized for two GPUs, it also scales well to larger even-numbered configurations. We present two extension strategies.

(1) Batch-Level Extension. In this approach, the model generates $\frac{N}{2}$ samples across N GPUs, where each pair of GPUs produces one image. This structure linearly increases acceleration with the number of GPUs while maintaining near-identical generation quality. However, it is most effective when a large number of samples are generated.

(2) Layer-Wise Pipeline Extension. This method extends the adaptive parallelism switching mechanism by dividing the optimal pipeline interval into N layer-wise segments, thereby enabling finer-grained parallel execution across multiple devices. Unlike the batch-level scheme, it can be applied to single-sample generation, though it may incur slightly reduced acceleration efficiency and minor quality degradation due to finer partitioning.

The structures and details of both strategies are provided in Appendix F. Experimental results on *single* image generations with 4 and 8 GPUs are provided in Section 5.2.

Base Model	Devices	Methods	Latency (s) ↓	Speed-Up ↑	Comm. (GB) ↓	FID ↓		LPIPS ↓		PSNR ↑	
						w/ G. T.	w/ Orig.	w/ G. T.	w/ Orig.	w/ G. T.	w/ Orig.
Stable Diffusion XL	**1**	Original Model	16.49	-	-	23.977	-	0.797	-	9.618	-
	2	DistriFusion [12]	13.53	1.22×	0.525	24.164	4.864	0.7978	0.146	9.597	24.634
		AsyncDiff [2] (stride=1)	12.54	1.31×	9.830	23.941	4.103	0.797	0.108	9.586	26.387
		Ours (k=5)	**7.12**	**2.31**×	**0.516**	**23.831**	**4.100**	**0.796**	**0.107**	**9.665**	**26.640**
	4	DistriFusion	7.13	2.31×	**0.486**	24.197	5.772	0.798	0.183	9.578	23.046
		AsyncDiff (stride=1)	9.45	1.74×	10.531	24.225	5.829	0.795	0.147	9.626	24.776
		AsyncDiff (stride=2)	6.57	2.51×	4.795	24.140	6.572	0.803	0.192	9.557	23.070
		Ours (k=5)	**4.83**	**3.41**×	0.751	**24.087**	**5.544**	**0.788**	**0.113**	**9.888**	**25.233**
Stable Diffusion 3	**1**	Original Model	19.36	-	-	33.433	-	0.810	-	8.086	-
	2	AsyncDiff [2] (stride=1)	9.82	1.97×	1.290	33.379	2.032	0.813	0.052	8.155	27.812
		xDiT-Ring [4]	14.31	1.35×	121.646	33.356	1.909	0.809	0.047	8.085	27.857
		Parastep [33]	9.98	1.94×	**0.032**	33.340	3.350	0.810	0.112	8.091	22.917
		Compact-2bit [21]	13.96	1.39×	9.049	33.560	4.712	0.810	0.196	7.999	19.242
		Ours (k=5)	**9.33**	**2.07**×	0.189	**33.322**	**1.878**	**0.780**	**0.046**	**8.229**	**27.875**
	4	AsyncDiff (stride=1)	6.51	2.97×	3.774	33.909	3.458	0.887	0.177	8.033	26.540
		AsyncDiff (stride=2)	**4.86**	**3.98**×	1.241	34.434	8.939	0.895	0.272	7.956	21.087
		xDiT-Ring	17.96	1.08×	130.792	33.374	2.232	0.861	0.127	8.001	27.101
		Parastep	6.29	3.08×	**0.038**	33.211	2.581	0.910	0.134	7.990	27.089
		Compact-2bit	9.25	2.09×	13.593	33.420	5.461	**0.811**	0.248	7.926	17.490
		Ours (k=5)	5.53	3.50×	0.572	**33.113**	**2.109**	0.857	**0.122**	**8.046**	**27.110**

Table 1. **Quantitative comparison of parallelism methods on the Stable Diffusion XL and Stable Diffusion 3 models.** We compare our method with existing distributed inference techniques under 1, 2, and 4 GPUs. We report both the baseline latency and the corresponding acceleration ratio (*Speed-Up*), Communication efficiency *(Comm.)*, and quantitative metrics assessing generation fidelity. Here, *w/ G.T.* denotes comparison with the ground-truth image, and *w/ Orig.* indicates comparison with the original (single-GPU) model output.

5. Experiments

5.1. Experimental Setup

Models. We evaluate our proposed hybrid parallelism framework on two representative diffusion backbones: Stable Diffusion XL (SDXL) [24] and Stable Diffusion 3.0 (SD3), a DiT-based flow matching model [3]. SDXL represents U-Net–based latent diffusion models [26], while SD3 reflects the transformer-based paradigm, demonstrating the generality of our approach.

Datasets. All experiments are conducted on the MS-COCO Captions 2014 benchmark [15], using 5,000 validation prompts for text-to-image generation. Generated images are compared against both the ground-truth samples and the single-GPU original model outputs.

Metrics. We evaluate inference efficiency and generative quality. Latency and speed-up ratio measure acceleration. For quality, we report FID (Fréchet Inception Distance) [6], LPIPS (Learned Perceptual Image Patch Similarity) [42], and PSNR (Peak Signal-to-Noise Ratio). Lower FID/LPIPS and higher PSNR indicate better generation quality. For implementation details, please refer to Appendix G.

5.2. Main Results

Quantitative Results. Table 1 reports a quantitative comparison across SDXL and SD3 pre-train diffusion models. On SDXL, with 2 GPUs, ours achieves a 2.31× acceleration over the single-GPU baseline while preserving image fidelity. Compared to prior methods such as DistriFusion [12] and AsyncDiff [2], our method attains the best speed–quality trade-off with minimal communication cost. Notably, our communication cost is reduced by 19.6× compared to AsyncDiff, due to adaptive parallelism switching that determines optimal parallel intervals to minimize communication cost. With 4 GPUs, ours achieves a 3.41× speed-up while preserving comparable generation quality.

Methods	Latency (s) ↓	Speed-Up ↑	FID ↓ (w/ Orig.)
Original Model	16.49	-	-
Full Condition-Based Partitioning	9.24	1.78×	**3.623**
Ours (Hybrid Parallelism)	**7.12**	**2.31**×	4.100

Table 2. **Ablation on hybrid parallel components.** All experiments are conducted on the SDXL model at 1024×1024 resolution with 2 GPUs, comparing the single-GPU baseline, full condition-based partitioning, and our hybrid parallelism framework.

For SD3, a DiT-based flow-matching model, with 2 GPUs, ours not only outperforms recent baselines, xDiT-Ring, Parastep, and CompactFusion, but also achieves a 2.07× speed-up with negligible communication cost while maintaining comparable generation quality. With 4 GPUs, ours reaches up to a 3.50× speed-up, demonstrating strong scalability across both U-Net and DiT architectures. Additional results with 8 GPUs are provided in Appendix H.

Qualitative Results. Figure 5 presents qualitative comparisons among distributed inference methods. While DistriFusion and AsyncDiff exhibit boundary artifacts or spatial inconsistency, our method preserves global coherence and fine-grained details similar to the original model. These results confirm that the proposed hybrid parallelism framework maintains high visual fidelity while achieving substantial acceleration. Further results are shown in Appendix I.

Prompt: "*A white tray with a vase holding flowers over a cat.*"

Figure 5. **Qualitative results of the main experiments.** We compare 1024×1024 image generations from the SDXL model with 2 GPUs. Our method achieves the best acceleration and FID performance, while producing visuals most similar to the original.

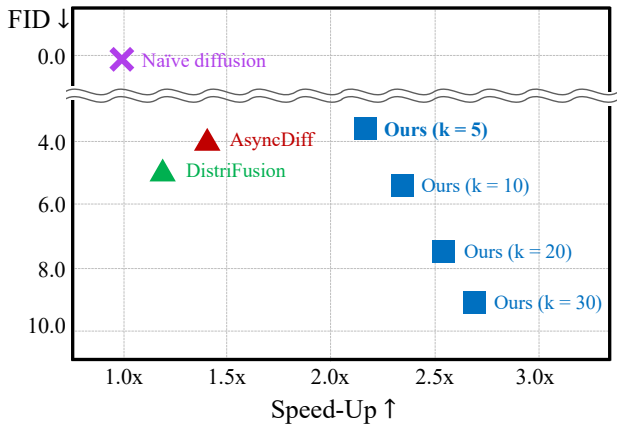

Figure 6. **Visualization of speed–quality trade-off across different parallelism intervals *k*.** Smaller k values preserve higher fidelity, whereas larger k achieve greater acceleration. Our method consistently dominates prior works across the trade-off frontier. All experiments were conducted on 2 GPUs.

5.3. Ablation Study

Ablation on Hybrid Parallel Components. Table 2 analyzes the contribution of each hybrid parallel component. We compare three settings: (1) the original single-GPU model, (2) full condition-based partitioning applied to all denoising steps, and (3) our proposed hybrid parallelism combining both condition-based partitioning and adaptive parallelism switching. Condition-based partitioning achieves a 1.78× speed-up while maintaining image quality, whereas our hybrid parallelism further improves efficiency to 2.31× with comparable quality, demonstrating that the combination of condition-based partitioning and adaptive parallelism switching maximizes acceleration while minimizing quality degradation.

5.4. Sensitivity Analysis

Impact of Different *k* Values. Figure 6 demonstrates the parallelism interval k clearly reveals a speed–quality trade-off: smaller k values preserve higher fidelity, while larger k values yield faster generation. An appropriate balance is observed at k=5, achieving both strong quality and acceleration. The interval k can be flexibly chosen by practitioners to adjust the trade-off between efficiency and fidelity. Quantitative and qualitative results across different k values are provided in Appendix J and Appendix K, respectively.

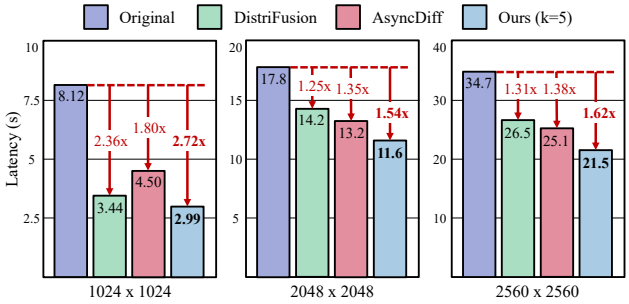

Figure 7. **Comparison of high-resolutions tasks.** We compare different parallel inference methods on the SDXL model using two NVIDIA H200 GPUs across 1024×1024, 2048×2048, and 2560×2560 high-resolutions.

High-Resolution Generation. As shown in Figure 7, our method consistently achieves superior acceleration over existing distributed inference frameworks across increasing resolutions. On the SDXL model using NVIDIA H200 GPUs, our hybrid parallelism attains up to 2.72× speed-up at 1024×1024, 1.54× speed-up at 2048×2048, and 1.62× speed-up at 2560×2560, demonstrating strong scalability for high-resolution image generation.

6. Conclusion

In this paper, we introduced a hybrid parallelism framework for diffusion inference that integrates condition-based partitioning with adaptive parallelism switching. Guided by the denoising discrepancy criterion, the method adaptively switches between parallelism modes to minimize redundant communication. It achieves 2.31× and 2.07× latency reductions on SDXL and SD3 with 2 GPUs, and up to 3.41× and 3.50× with 4 GPUs, respectively, while strongly preserving fidelity. We also generalize across the U-Net and DiT architectures and to high-resolution generation tasks, providing a unified parallelism paradigm for scalable multi-GPU diffusion inference.

Acknowledgement. This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2020-II200862, DB4DL: High-Usability and Performance In-Memory Distributed DBMS for Deep Learning, 50% and No. RS-2022-II220157, Robust, Fair, Extensible Data-Centric Continual Learning, 50%).

References

[1] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 1, 3

[2] Zigeng Chen, Xinyin Ma, Gongfan Fang, Zhenxiong Tan, and Xinchao Wang. Asyncdiff: Parallelizing diffusion models by asynchronous denoising. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 37:95170–95197, 2024. 2, 3, 7, 4

[3] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024. 7

[4] Jiarui Fang, Jinzhe Pan, Xibo Sun, Aoyu Li, and Jiannan Wang. xdit: an inference engine for diffusion transformers (dits) with massive parallelism. *arXiv preprint arXiv:2411.01738*, 2024. 3, 4, 7

[5] Jiarui Fang, Jinzhe Pan, Jiannan Wang, Aoyu Li, and Xibo Sun. Pipefusion: Patch-level pipeline parallelism for diffusion transformers inference. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 2025. 3, 4

[6] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 30, 2017. 7

[7] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop on Deep Generative Models and Downstream Applications*, 2021. 3, 2

[8] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 35:26565–26577, 2022. 6, 2

[9] Zhifeng Kong and Wei Ping. On fast sampling of diffusion probabilistic models. *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021. 1, 3

[10] Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 37:122458–122483, 2024. 5

[11] Muyang Li, Ji Lin, Chenlin Meng, Stefano Ermon, Song Han, and Jun-Yan Zhu. Efficient spatially sparse inference for conditional gans and diffusion models. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 35:28858–28873, 2022. 1, 3

[12] Muyang Li, Tianle Cai, Jiaxin Cao, Qinsheng Zhang, Han Cai, Junjie Bai, Yangqing Jia, Kai Li, and Song Han. Distrifusion: Distributed parallel inference for high-resolution diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7183–7193, 2024. 2, 3, 7, 4

[13] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 17535–17545, 2023. 1, 3

[14] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 36:20662–20678, 2023. 1, 3

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014. 5, 7, 1

[16] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023. 3

[17] Haozhe Liu, Wentian Zhang, Jinheng Xie, Francesco Faccio, Mengmeng Xu, Tao Xiang, Mike Zheng Shou, Juan-Manuel Perez-Rua, and Jürgen Schmidhuber. Faster diffusion via temporal attention decomposition. *arXiv preprint arXiv:2404.02747*, 2024. 1, 3

[18] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 1, 3

[19] 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. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 35:5775–5787, 2022. 1, 3

[20] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, pages 1–22, 2025. 1, 3

[21] Jiajun Luo, Yicheng Xiao, Jianru Xu, Yangxiu You, Rongwei Lu, Chen Tang, Jingyan Jiang, and Zhi Wang. Accelerating parallel diffusion model serving with residual compression. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 2025. 7, 4

[22] 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. 1, 3

[23] 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 (CVPR)*, pages 15762–15772, 2024. 1, 3

[24] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. 7

[25] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 3505–3506, 2020. 1

[26] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 7

[27] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 1, 3

[28] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 87–103. Springer, 2024. 1, 3

[29] Andy Shih, Suneel Belkhale, Stefano Ermon, Dorsa Sadigh, and Nima Anari. Parallel sampling of diffusion models. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 36:4263–4276, 2023. 1, 3

[30] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019. 1

[31] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 3, 4

[32] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 6, 2

[33] Kunyun Wang, Bohan Li, Kai Yu, Minyi Guo, and Jieru Zhao. Communication-efficient diffusion denoising parallelization via reuse-then-predict mechanism. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 2025. 3, 4, 7

[34] Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6211–6220, 2024. 3

[35] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 1, 3

[36] Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22552–22562, 2023. 1, 3

[37] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6613–6623, 2024. 1, 3

[38] Zongsheng Yue, Jianyi Wang, and Chen Change Loy. Resshift: Efficient diffusion model for image super-resolution by residual shifting. *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 36:13294–13307, 2023. 1, 3

[39] Dingkun Zhang, Sijia Li, Chen Chen, Qingsong Xie, and Haonan Lu. Laptop-diff: Layer pruning and normalized distillation for compressing diffusion models. *arXiv preprint arXiv:2404.11098*, 2024. 1, 3

[40] Jiale Zhang, Yulun Zhang, Jinjin Gu, Jiahua Dong, Linghe Kong, and Xiaokang Yang. Xformer: Hybrid x-shaped transformer for image denoising. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. 1, 3

[41] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023. 1, 3

[42] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018. 7