DIVIDE-AND-DENOISE: A GAME-THEORETIC METHOD FOR FAIRLY COMPOSING DIFFUSION MODELS

Abhi Gupta[1,2] † *, **Polina Barabanshchikova**[3,4] †, **Vikas Garg**[3,6], **Samuel Kaski**[3,4,5], **Tommi Jaakkola**[1,2]
[1] MIT ORC, USA [2] MIT CSAIL, USA [3] Department of Computer Science, Aalto University, Finland
[4] ELLIS Institute Finland [5] Department of Computer Science, University of Manchester, UK
[6] YaiYai Ltd
† Equal contribution *Correspondence to: abhig@mit.edu

ABSTRACT

The abundance of pre-trained diffusion models provides an opportunity for composition. Combining several models, however, runs the risk of one model dominating or models disagreeing with each other. Here, we propose Divide-and-Denoise, a method for coordinating multiple pre-trained diffusion models during sampling. Much like managing a specialized workforce, our method creates a fair but efficient division of labor across models. Central to our method is the notion of an allocation which defines the responsibility of each model to every region of the noisy sample. At every timestep, we then denoise by (i) updating the allocation by solving a fair division game, where we divide the sample into regions that maximize total utility under fairness constraints, and (ii) aligning the models with this allocation, where we guide each model to denoise within its assigned region. This leads to a new composite denoising process that evolves in tandem with a division process. We evaluate Divide-and-Denoise on conditional image generation. Across several quality metrics, including the GenEval benchmark, our method outperforms baselines and resolves common failures including missing objects and mismatched attributes. Experiments show that Divide-and-Denoise utilizes each model's expertise without neglecting any other model.

1 INTRODUCTION

Large-scale diffusion models have enabled significant advancements across robotics Xu et al. (2024); Chi et al. (2023), biomedicine Jumper et al. (2021); Corso et al. (2023), and computer vision Song & Ermon (2019); Ho et al. (2020a); Rombach et al. (2022); Sehwag et al. (2024). However, training demands significant computational resources and often requires task-specific fine-tuning Black et al. (2023); Wallace et al. (2024), creating a need for effective model reuse and composition Dhariwal & Nichol (2021); Ho & Salimans (2022).

Among the many available models, choosing which one to use is not always obvious. A collection of models may even be used together in order to generate data that no individual model could generate alone. Consider, as a running example, one model trained on images of dogs and another on cats. A common approach is to define a composite distribution as the product or mixture of the 'dog' and 'cat' densities Liu et al. (2022); Du et al. (2023). Other analytical operations include the harmonic mean and contrast Garipov et al. (2023), as well as logical operations such as AND Skreta et al. (2024). Although these operations permit tractable sampling, they are often too simple to preserve the characteristics of each model's distribution when there is conflict. For instance, if models are trained on images of animals appearing mainly in the center, sampling from their product density typically produces incoherent, overlapping dogs and cats.

Related Work. Recent work has explored composing text-to-image diffusion models to improve spatial control Bar-Tal et al. (2023); Du et al. (2023). The typical strategy is to have the user segment an image into spatial regions, assign each region a text prompt (e.g., 'dog' or 'cat'), and then denoise each region with the corresponding model. Although simple to implement, these techniques are restricted to user-defined allocations. This kind of division of labor between models is cumbersome to specify, infeasible to define in many domains (e.g., manually partitioning proteins), and does not

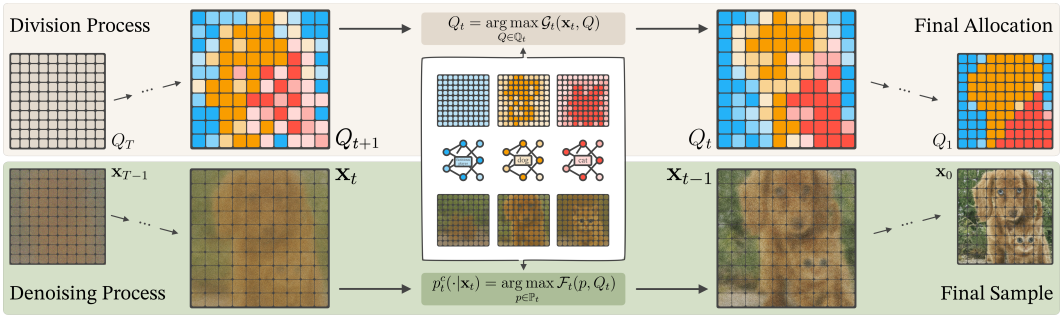

Figure 1: Divide-and-Denoise. A noisy image is iteratively refined by two coupled processes: (i) a division step that computes a fair and efficient division of the latent image given by the allocation Q and (ii) a composite denoising step that reconciles the proposals of several single concept diffusion models into a composite update p using this allocation. At every timestep, models provide utilities, the image is divided into soft regions in order to maximize total utility under fairness constraints, and each region is denoised with the assigned model.

take into account the strengths or weaknesses of each model. Bar-Tal et al. (2023) for example assume models can faithfully follow the user-prescribed layout, an assumption that often fails and requires additional forms of guidance Couairon et al. (2023); Manukyan et al. (2023).

Contributions. We propose Divide-and-Denoise: a game-theoretic framework for coordinating multiple pre-trained diffusion models. Instead of requiring ground-truth partitions, we infer them online through fair and efficient division of labor. Our method is fully compositional: models need not share weights, architectures, or training data, as long as they operate in a latent space of the same dimension.

- An inference-time algorithm for coordinating multiple pre-trained diffusion models with differing expertise.
- Coupled processes: (i) a *division process* solving a fair division game and (ii) a composite *denoising process* aligning each model with its assigned region.
- Two formulations of model utilities: a general definition for any conditional diffusion model, and a specific instantiation using attention maps for cross-attention conditioning.
- Empirical validation using GenEval Ghosh et al. (2023), CLIP-Score, VQAScore, and ImageReward showing single-concept models outperform existing composition techniques and multi-concept models.

2 Background

2.1 Diffusion Samplers

Diffusion models define a forward probability path q_t, starting from the data distribution q_0, and gradually corrupting data with Gaussian noise. These models are typically provided as time-dependent score networks $s_t(\mathbf{x}; \theta)$ approximating $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$ at each timestep t. To generate new data from diffusion models, we can simulate the denoising process with samplers including DDPM Ho et al. (2020b), DDIM Song et al. (2022), and other numerical methods. All these procedures can be expressed as sampling from a sequence of Gaussian transition kernels

$$p_t(\mathbf{x}_{t-1}|\mathbf{x}_t; \theta) \coloneqq \mathcal{N}(\mu_t(\mathbf{x}_t; \theta), \sigma_t^2 I), \quad t = 1, \dots T-1.$$

Generation begins by drawing $\mathbf{x}_T \sim \mathcal{N}(0, I)$. The sampler then iteratively produces $\mathbf{x}_{T-1}, \mathbf{x}_{T-2}, \dots, \mathbf{x}_0$ by applying these transition kernels until a final sample $\mathbf{x}_0$ is obtained. Notably, the family of samplers introduced in Song et al. (2021) allows both the total number of sampling steps T and the noise schedule σ_t to be varied while keeping the pre-trained model fixed. In this framework, the noise level is parameterized by a scalar $\eta \in [0, 1]$, where $\eta = 1$ recovers the DDPM sampler and $\eta = 0$ yields a fully deterministic trajectory.

2.2 DIFFUSION CONDITIONING

To generate samples with specific attributes, the score $s_t(\mathbf{x}_t;\theta)$ used during sampling is replaced by a surrogate function $\hat{s}_t(\mathbf{x}_t,\boldsymbol{y};\theta)$ conditioned on a concept $\boldsymbol{y}$. Classifier guidance Dhariwal & Nichol (2021) leverages

$$\nabla_{\mathbf{x}} \log q_t(\mathbf{x}_t|\boldsymbol{y}) = \nabla_{\mathbf{x}} \log q_t(\mathbf{x}_t) + \nabla_{\mathbf{x}} \log q_t(\boldsymbol{y}|\mathbf{x}_t)$$

and defines $\hat{s}_t(\mathbf{x}_t,\boldsymbol{y};\theta) = s_t(\mathbf{x}_t;\theta) + \omega\nabla_{\mathbf{x}} \log q_t(\boldsymbol{y}|\mathbf{x}_t;\theta)$, where $\log q_t(\boldsymbol{y}|\mathbf{x}_t;\theta)$ is modeled by a pre-trained classifier and ω is a scaling factor. Alternatively, one can train the diffusion model conditionally so that $s_t(\mathbf{x}_t,\boldsymbol{y};\theta) \approx \nabla_x \log q_t(\mathbf{x}|\boldsymbol{y})$. Both of these ideas are combined in a technique called classifier-free guidance Ho & Salimans (2022). It combines both unconditional and conditional models to define $\hat{s}_t(\mathbf{x}_t,\boldsymbol{y};\theta) = s_t(\mathbf{x}_t;\theta) + \omega(s_t(\mathbf{x}_t,\boldsymbol{y};\theta) - s_t(\mathbf{x}_t;\theta))$.

2.3 IMAGE GENERATION

Diffusion models for conditional image generation are trained on large datasets $\mathcal{D} \coloneqq \{\mathbf{z},\boldsymbol{y}\}$, where $\mathbf{z} \in \mathbb{R}^{H\times W\times 3}$ is a high-resolution image and $\boldsymbol{y}$ is a label or a text prompt. In practice, images are encoded into compressed representations $\mathbf{x} = \phi(\mathbf{z}) \in \mathbb{R}^{D\times D\times C}$, and the diffusion model operates in a latent space.

Attention Maps. Modern diffusion architectures rely heavily on cross-attention layers to condition generation on text prompts represented by embeddings $\tau(\boldsymbol{y}) \in \mathbb{R}^{L\times K}$ Vaswani et al. (2017). The text-to-image model can be expressed as $s_t(\mathbf{x}_t,\boldsymbol{y};\theta) = f_t(\mathbf{x}_t,\{A_t^{(\ell)}(\mathbf{x}_t,\tau(\boldsymbol{y});\theta)\};\theta)$, where $\{A_t^{(\ell)} \in \mathbb{R}^{\bar{D}_\ell\times\bar{D}_\ell\times L}\}$ is a set of per-layer cross-attention maps. For each spatial location, these maps describe how strongly the model attends to each token in the prompt. Because layers operate at different resolutions $\bar{D}_\ell$, the map sizes can vary. Generation can be controlled in creative ways by substituting attention maps $\{A_t^{(\ell)}(\mathbf{x}_t,\tau(\boldsymbol{y});\theta)\}$ with those from another pre-trained model $\{A_t^{(\ell)}(\mathbf{x}_t,\tau(\boldsymbol{y});\theta')\}$ Hertz et al. (2022). Upscaling and averaging these maps across layers leads to a saliency map $A_t(\mathbf{x}_t,\tau(\boldsymbol{y});\theta) \in \mathbb{R}^{D\times D\times L}$ as shown by Tang et al. (2023).

2.4 FAIR DIVISION

Dividing m goods among n players is a classical problem in game theory Amanatidis et al. (2023); Nishimura & Sumita (2021); Dickerson et al. (2014); Cole et al. (2017); Eisenberg & Gale (1959); Caragiannis et al. (2019). In the setting of indivisible items, each player $i \in \{1,2,\ldots,n\}$ is allocated a bundle of goods, represented by a binary assignment vector $\mathbf{M}_i \in \{0,1\}^m$, so that no two players share any goods and all goods are allocated. Each player has a utility function $u_i : \{0,1\}^m \to \mathbb{R}_+$ that measures the value of any bundle. Among all possible partitions, we typically seek solutions that are fair and efficient with respect to these utilities. The three main notions of fairness are: *envy-freeness* (no player prefers another player's bundle), *proportionality* (each player receives at least $1/n$ of their total utility), and *equitability* (all players receive bundles of equal utility). Efficiency can be measured, for example, by Nash social welfare, the product of individual utilities.

Mixed Allocations. In the case of a single good, no matter who gets it, the partition is not fair to others. This highlights that fair assignments do not always exist. One way to address this challenge is to consider randomized allocations over all possible assignments:

$$\mathbb{M}_{n,m} = \left\{ \mathbf{M} \in \{0,1\}^{n\times m} : \sum_{i=1}^{n} \mathbf{M}_{i,j} = 1 \; \forall 1 \le j \le m \right\}$$

A mixed allocation Q is a discrete distribution over $\mathbb{M}_{n,m}$. Fairness notions are defined in terms of expected utilities under Q. For example, an envy-free allocation Q satisfies $\mathbb{E}_{\mathbf{M}\sim Q} u_i(\mathbf{M}_i) \ge \mathbb{E}_{\mathbf{M}\sim Q} u_i(\mathbf{M}_{i'})$ for all $1 \le i \ne i' \le n$. Note that the uniform allocation $\mathcal{U}(\mathbb{M}_{n,m})$ is always fair. Therefore, in the randomized setting, efficiency is crucial to avoid trivial solutions.

Decomposable Allocations. When utilities are additive in the goods, $u_i(\mathbf{M}_i) = \sum_{j=1}^{m} u_{ij}\mathbf{M}_{i,j}$, the expected utility of player i simplifies to $\sum_{j=1}^{m} u_{ij}Q_{ij}$, where $Q_{ij} \coloneqq \mathbb{E}_{\mathbf{M}\sim Q}\mathbf{M}_{i,j}$ is a fractional

Algorithm 1 Divide-and-Denoise

Input: n pre-trained diffusion models, $\alpha > 0$, $\beta > 0$.
Initialize $p_T^c = \mathcal{N}(0, I)$ and $Q_T = \mathcal{U}(\mathcal{M}_{n+1,m})$.
Sample $\mathbf{x}_{T-1} \sim p_T^c$.
for $t = T-1, \ldots, 1$ **do**
  Aggregate $\{p_t^i(\cdot|\mathbf{x}_t) = \mathcal{N}(\mu_t^i(\mathbf{x}_t), \sigma_t^2 I)\},\ \{u_{ij}(\mathbf{x}_t, t)\}$
  # Update Division Process
  Solve dual problem for λ^*
  Update allocation $Q_{ij}^t \propto e^{-\langle \lambda^*, \phi_{ij} \rangle + \frac{1}{\beta} \log u_{ij}} Q_{ij}^{t+1}$
  # Update Composite Denoising Process
  Compute weights $\hat{Q}_i^t = Q_i^t + \frac{1}{n} Q_{n+1}^t$
  Update mean $\hat{\mu}_t^c = \sum_{i=1}^n \mu_t^i(\mathbf{x}_t) \odot \hat{Q}_i^t \ + \ \sigma_t \frac{\nabla_{\mathbf{x}_t} U_t(\mathbf{x}_t, Q)}{\alpha \|\nabla_{\mathbf{x}_t} U_t(\mathbf{x}_t, Q)\|}$
  Sample $\mathbf{x}_{t-1} \sim p_t^c(\cdot|\mathbf{x}_t) = \mathcal{N}(\hat{\mu}_t^c(\mathbf{x}_t), \sigma_t^2 I)$
end for

weight. We say that an allocation Q is *decomposable* if

$$Q(\mathbf{M}) = \prod_{i=1}^{n} \prod_{j=1}^{n} Q_{ij}^{\mathbf{M}_{i,j}} \quad \forall \mathbf{M} \in \mathbb{M}_{n,m}.$$

Decomposable allocations are essentially equivalent to fractional allocations of m divisible goods, where player i receives a fraction Q_{ij} of good j.

3 Divide-and-Denoise

We study the problem of coordinating n pre-trained diffusion models, each of which operates in a common latent space of dimension m. Without loss of generality, we denote the sequence of denoising kernels for diffusion model $1 \leq i \leq n$ as follows:

$$p_T^i = \mathcal{N}(0, I), \quad p_t^i(\cdot \mid \mathbf{x}_t) = \mathcal{N}(\mu_t^i(\mathbf{x}_t), \sigma_t^2 I),\ 1 \leq t < T.$$

Additionally, we assume that models have additive preferences over the latent coordinates given by $u_{ij}(\mathbf{x}, t)$, i.e. model i's value for coordinate j at latent $\mathbf{x}$ and timestep t. In Section 3.4, we show that all conditional models already possess intrinsic utilities and offer alternatives as well. Our goal is to define a composite denoising process with kernels $p_t^c(\cdot|\mathbf{x}_t)$ that best accounts for the preference of each model.

In this work, each model is conditioned on a concept $\boldsymbol{y}_i$. We expect samples from p_t^c to ideally match what a single model trained on all concepts together would generate, though such a model may not be available. Models need not share architecture or parameters.

3.1 Simulating Two Processes

The main components of our approach are outlined in Figure 1. Divide-and-Denoise generates two coupled trajectories: a sampling path of the composite denoising process, obtained by iteratively drawing $\mathbf{x}_{t-1} \sim p_t^c(\mathbf{x}_{t-1}|\mathbf{x}_t)$, and a path of the division process given by allocations Q_t, also obtained by iteratively updating in time. We define each allocation Q_t to be a distribution over $\mathbb{M}_{n,m}$, the space of partitions of the latent space of dimension m across n models. Since Q_t specifies how the latent coordinates at time t are distributed among the pre-trained models, it may be interpreted as a division of labor.

We initialize with $Q_T = \mathcal{U}(\mathbb{M}_{n,m})$ and $p_T^c = \mathcal{N}(0, I)$, and draw the first noisy latent as $\mathbf{x}_{T-1} \sim p_T^c$. At each of the remaining timesteps $1 \leq t < T$, we update the allocation and the composite process according to the bi-level optimization:

$$Q_t = \arg\max_{Q \in \mathbb{Q}_t} \mathcal{G}_t(\mathbf{x}_t, Q) \tag{1}$$

$$p_t^c(\cdot|\mathbf{x}_t) = \arg\max_{p \in \mathbb{P}_t} \mathcal{F}_t(p, Q_t) \tag{2}$$

The choice of $\mathcal{G}$, $\mathcal{F}$, $\mathbb{Q}_t$, and $\mathbb{P}_t$ will be discussed in Sections 3.2 and 3.3. Problem (1) fairly and efficiently divides the latent among models, while (2) chooses a denoising update aligning with this division. Both objectives use a common alignment score U_t with problem-specific regularization. At each timestep, we sample $\mathbf{x}_{t-1}$ from $p_t^c(\cdot|\mathbf{x}_t)$.

3.2 Computing a Fair and Efficient Division

We formulate the problem of finding the next allocation (equation 1) as a fair division game, where the goods are latent coordinates and the players are the individual diffusion models. The efficiency of the allocation is measured in terms of the expected total utility

$$U_t(\mathbf{x}, Q) = \mathbb{E}_{\mathbf{M}\sim Q} \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbf{M}_{i,j} u_{ij}(\mathbf{x}, t).$$

The objective $\mathcal{G}_t$ is the efficiency score regularized by a Kullback-Leibler (KL) divergence with positive weight β_t:

$$\mathcal{G}_t(\mathbf{x}_t, Q) = U_t(\mathbf{x}_t, Q) - \beta_t D_{\mathrm{KL}}(Q||Q_{t+1}). \tag{3}$$

The regularization term penalizes abrupt changes between consecutive allocations, encouraging temporally smooth allocation trajectories that provide a stable signal for the composite denoising update. A hyperparameter β controls the trade-off between efficiency and smoothness. For example, when $\beta \to \infty$ the allocation Q_t remains uniform throughout generation.

A solution is constrained to lie in the set of fair allocations $\mathbb{Q}_t$. We express this constraint set as

$$\mathbb{Q}_t = \left\{ Q \in \Delta(\mathbb{M}_{n,m}) : \mathbb{E}_{\mathbf{M}\sim Q} \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbf{M}_{i,j} \phi_{ij}(\mathbf{x}_t, t) \preceq \boldsymbol{b} \right\}$$

with coefficients $\phi_{ij}(\mathbf{x}_t, t) = (\phi_{ij}^1(\mathbf{x}_t, t), \ldots, \phi_{ij}^l(\mathbf{x}_t, t))$ and $\boldsymbol{b} = (b_1, \ldots, b_l)$, for all $1 \le i \le n$ and $1 \le j \le m$, specifying l linear constraints. Despite this simple form, these sets are flexible enough to represent common notions of fairness under additive utilities as shown below.

Example 1. *Using a single linear inequality $\mathbb{E}_{\mathbf{M}\sim Q} \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbf{M}_{i,j} \phi_{ij}^1(\mathbf{x}_t, t) \preceq b_1$, we can encode the following relations:*

1. *Setting $b_1 = 0$ and $\phi_{kj}^1(\mathbf{x}_t, t) = -u_{ij}(\mathbf{x}_t, t) I(k = i) + u_{ij}(\mathbf{x}_t, t) I(k = i')$ is equivalent to saying that player i is not envious of player i'.*
2. *Setting $b_1 = 0$ and $\phi_{kj}^1(\mathbf{x}_t, t) = -u_{ij}(\mathbf{x}_t, t) I(k = i) + u_{ij}(\mathbf{x}_t, t)/n$ is equivalent to constraining player i to be allocated at least $1/n$ of its total utility. Alternatively, for the normalized utilities, we can set $b_1 = -1/n$ and $\phi_{kj}^1(\mathbf{x}_t, t) = -u_{ij}(\mathbf{x}_t, t) I(k = i)$.*
3. *Setting $b_1 = 0$ and $\phi_{kj}^1(\mathbf{x}_t, t) = -u_{ij}(\mathbf{x}_t, t) I(k = i) + u_{i'j}(\mathbf{x}_t, t) I(k = i')$ is equivalent to saying that the allocated utility of player i is greater or equal to that of player i'.*

Clearly, by stacking inequalities, we can represent envy-free, proportional, and equitable constraints or their combinations for any number of players. It is worth noting that the uniform allocation is always fair, so the feasible set is not empty.

We conclude this section by introducing a generic solution to the optimization problem in equation 1.

Theorem 1. *Assume that allocation Q_{t+1} is decomposable with weights Q_{ij}^{t+1}. Then, the optimal allocation Q_t solving the fair division game (1) is also decomposable with weights*

$$Q_{ij}^t = \frac{\exp(-\langle \lambda^*, \phi_{ij}(\mathbf{x}_t, t) \rangle + u_{ij}(\mathbf{x}_t, t)/\beta) Q_{ij}^{t+1}}{Z_j(\lambda^*)}, \tag{4}$$

where $Z_j(\lambda^*) = \sum_{i=1}^{n} e^{-\langle \lambda^*, \phi_{ij}(\mathbf{x}_t, t) \rangle + u_{ij}(\mathbf{x}_t, t)/\beta} Q_{ij}^{t+1}$ is a normalization constant and λ^* is a solution of a dual problem: $\max_{\lambda \ge 0} -\langle \boldsymbol{b}, \lambda \rangle - \sum_{j=1}^{m} \log Z_j(\lambda)$.

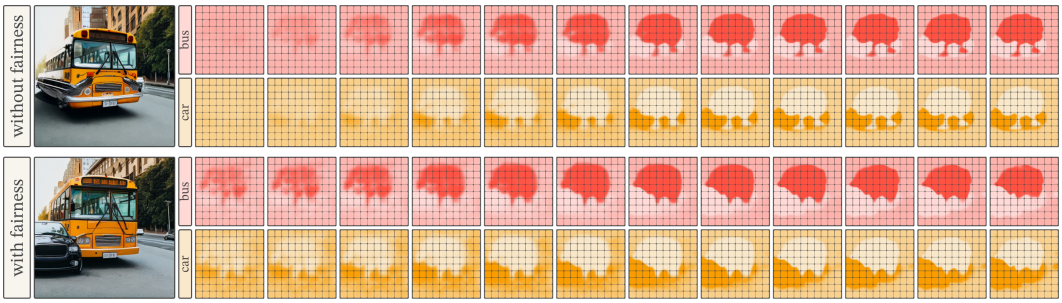

Figure 2: The role of fairness in Divide-and-Denoise using Stable Diffusion. Top: without fairness, the car model is allocated significantly fewer pixels than the bus model, resulting in a missing object. Bottom: with fairness, the allocation balances both models so the car is no longer envious of the bus. Fairness prevents a single model from dominating.

3.3 Aligning Models with their Allocations

In this section, we address the second problem (equation 2) of selecting a composite denoising kernel p_t^c from the set of feasible denoising distributions $\mathbb{P}_t$. We introduce a new objective that explicitly aligns each diffusion model's proposal with its assigned region, conditioned on the given allocation Q. Let p_j denote the marginal of a denoising kernel p at coordinate j. We define

$$\mathcal{F}_t(p, Q) = \mathbb{E}_{\mathbf{x}_{t-1}\sim p} U_{t-1}(\mathbf{x}_{t-1}, Q) - \alpha_t \mathbb{E}_{\mathbf{M}\sim Q}\left[\sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{i,j} D_{\mathrm{KL}}(p_j(\cdot|\mathbf{x}_t)||p_j^i(\cdot|\mathbf{x}_t))\right] \quad (5)$$

The aim of the KL regularization is to keep the denoising update for the coordinates allocated to model i close to its proposal. A hyperparameter $\alpha_t > 0$ controls the trade-off between alignment with the given allocation and adherence to the proposals of individual models. Notice how $u_{ij}(\mathbf{x}_{t-1}, t-1)$ contributes to the player's utility $u_i(\mathbf{M}_i)$ only if $\mathbf{M}$ assigns pixel j to model i. Maximizing U_{t-1} encourages each model to concentrate preference on its allocated region while suppressing preference outside it.

In order to encourage interaction between players for goods with low utility, we introduce a fictitious $(n+1)$-st player. This player is given a fixed uniform utility and its denoising kernel is defined as the geometric mean of the individual model kernels:

$$p^{n+1}(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto \prod_{i=1}^{n} p^i(\mathbf{x}_{t-1}|\mathbf{x}_t)^{1/n}, \quad u_{(n+1)j} \equiv 1/m,$$

for each $1 \le j \le m$. The allocation Q_t is now extended to a distribution over $\mathbb{M}_{n+1,m}$.

Since the objective in 2 can be non-linear, we cannot find an explicit solution in general. However, a solution exists under simplifying assumptions.

Theorem 2. *Consider the optimization in equation 2 with the following assumptions: (1) $\mathbb{P}_t$ is a set of all distributions on $\mathbb{R}^m$ with independent coordinates and (2) U_t is linear jointly in the first argument and t. For each $i \le n$, define the marginal weight vector $\hat{Q}_i$ as $\mathbb{E}_{\mathbf{M}\sim Q_t}(\mathbf{M}_i + \mathbf{M}_{n+1}/n)$. Then, the optimal composite denoising kernel is given by $p_t^c(\cdot|\mathbf{x}_t) = \mathcal{N}(\mu_t^c, \sigma_t^2 I)$, where $\mu_t^c = \sum_{i=1}^{n} \mu_t^i(\mathbf{x}_t) \odot \hat{Q}_i + \frac{\sigma_t^2}{\alpha_t}\nabla_{\mathbf{x}_t} U_t(\mathbf{x}_t, Q)$.*

The solution decomposes into a compositional update (first term) and a guidance term (gradient). When $\alpha_t \to \infty$, this recovers MultiDiffusion Bar-Tal et al. (2023).

In practice, we propose to use a local linearization technique. Applying a first-order Taylor expansion to linearize the reward, we approximate $p_t^c(\cdot|\mathbf{x}_t)$ with $\mathcal{N}(\hat{\mu}_t^c, \sigma_t^2 I)$, where $\hat{\mu}_t^c(\cdot|\mathbf{x}_t) = \sum_i \mu_t^i(\mathbf{x}_t) \odot \hat{Q}_i + \frac{\sigma_t^2}{\alpha_t}\sum_{i,j} Q_{ij}\nabla_{\mathbf{x}_t} u_{ij}(\mathbf{x}_t, t)$.

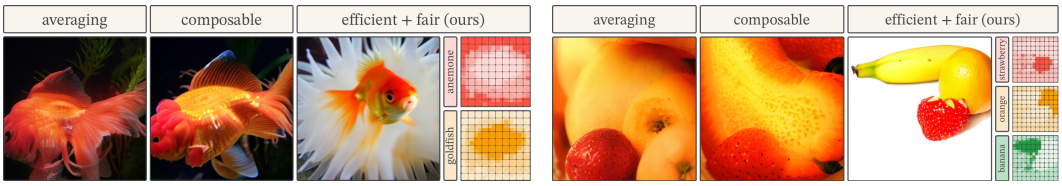

Figure 4: Left: *anemone+gold fish*. Right: *strawberry+orange+banana*.

We observe that performance is sensitive to the hyperparameter α_t. Large values of α_t suppress the influence of the guidance term, while overly small values may lead to out-of-distribution samples. We find it useful to reparameterize α_t as $\alpha_t = \frac{\sigma_t}{\alpha} \|\nabla_{\mathbf{x}_t} U_t(\mathbf{x}_t, Q)\|$, where α is a constant independent of time. The proposed approach is summarized in Algorithm 1.

3.4 Defining Player Utilities

Score-based Utility. For a vector v, let $\overline{v} = \frac{v}{\|v\|_2}$ and $\text{diag}(v)$ be the diagonal matrix with v on the diagonal. We define the utility of player i for a bundle $\mathbf{M}_{i'}$ as

$$u_i(\mathbf{M}_{i'}) = \overline{\nabla_{\mathbf{x}} q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)}^T \text{diag}(\mathbf{M}_{i'}) \overline{\nabla_{\mathbf{x}} q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)}.$$

Appendix A.2 shows that in the classifier-free guidance setup of conditional diffusion models, these utilities can be calculated from the model's trained score function by setting

$$u_{ij}(\mathbf{x}, t) = \frac{\|s_t^j(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t^j(\mathbf{x};\theta_i)\|_2^2}{\|s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t(\mathbf{x};\theta_i)\|_2^2}.$$

Attention-based Utility. In the text-to-image setting, cross-attention maps have been shown to be effective indicators of the relevance of each pixel to a target word or phrase. This motivates us to define attention-based utilities as

$$u_{ij}(\mathbf{x}, t) = \frac{A_t^j(\mathbf{x}, \boldsymbol{y}_i;\theta_i)}{\sum_{j=1}^m A_t^j(\mathbf{x}, \boldsymbol{y}_i;\theta_i)},$$

where A_t^j denotes the j-th coordinate of the attention map A_t aggregated across layers.

Score vs. Attention. Score-based utilities are available with any conditional diffusion model, eliminating the need to separately build utilities. Attention-based utilities require cross-attention layers, which many models provide. Figure 3 shows attention-based utilities behave better with less noise than score-based utilities during the generation process.

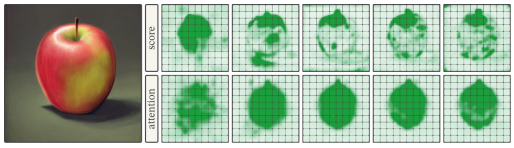

Figure 3: Evolution of score-based utilities on top and attention-based on bottom when generating an apple with Stable Diffusion.

4 Experiments

4.1 Models

Stable Diffusion. We use the Stable Diffusion 2.0 (Rombach et al., 2022), a text-to-image latent diffusion model with cross-attention conditioning. In this setting, single concepts are represented by short text descriptions. For a model i conditioned on a single concept $\boldsymbol{y}_i$, we use the prompt "`an image with` $\boldsymbol{y}_i$". In this section, we use attention-based utilities with Stable Diffusion. For quantitative comparison with score-based utilities, see Appendix A.4.

DiT. We also evaluate with the Diffusion Transformer (DiT) (Peebles & Xie, 2023), a class-conditioned diffusion model trained on ImageNet (Russakovsky et al., 2014). Here each model i

is conditioned on a single class $\boldsymbol{y}_i$ from ImageNet, and no multi-concept model is available. In this setting, we use score-based utilities.

4.2 Coordination Strategies

In all experiments, we employ a DDIM scheduler (Song et al., 2021), setting $T = 50$ sampling steps and a noise scale of $\eta = 0.015$. We use classifier-free guided models with the guidance scale $\omega = 7.5$ for Stable Diffusion and $\omega = 4$ for DiT. Divide-and-Denoise uses hyperparameters $\alpha = \eta$ in equation 3, while in equation 5 we use $\beta = 0.01$ for score-based utilities and $\beta = 0.001$ for attention-based. If not specified otherwise, proportional fairness is applied: each of n models is constrained to receive at least $1/n$ of its total utility. We compare our method against the following baselines:

Averaging. We construct a composite denoising process by averaging the scores from each single-concept diffusion model at each generation step. This represents a baseline where the division of labor is uniform.

Composable Diffusion. We employ a popular approach to compositional sampling (Liu et al., 2022) from conditional models. At each iteration, the scores are aggregated as

$$\hat{s}_t(\mathbf{x}_t;\theta) = s_t(\mathbf{x}_t;\theta) + \sum\nolimits_{i=1}^{n} \omega_i(s_t(\mathbf{x}_t,\boldsymbol{y}_i;\theta) - s_t(\mathbf{x}_t;\theta)).$$

For this baseline, we set $\omega_i = \omega$ for each i. Note that setting $\omega_i = \omega/n$ recovers the averaging baseline.

Multi-Concept Diffusion. We construct a composite denoising process with a single, multi-concept diffusion model. Since DiT models cannot simulatenously condition on several classes, this baseline applies only to Stable Diffusion where multiple concepts can be combined in a joint text prompt. We avoid enriching the multi-concept prompt with information beyond the concept set $\mathcal{Y} \coloneqq \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\}$, such as relationships between pairs of concepts, since a team of specialized models would not typically have access to this information. The multi-concept prompt is constructed as "`an image with` $\boldsymbol{y}_1$ `and` $\boldsymbol{y}_2$ `and ... and` $\boldsymbol{y}_n$". In this baseline the division of labor is implied by default through the output of a single Stable Diffusion model.

4.3 Evaluation Metrics

We evaluate our method along three axes: multi-concept image generation using single-concept models, correct attribute binding for complex concepts, and composition of intentionally conflicting concepts. We quantify first two of these criteria on a popular benchmark for image generation called GenEval (Ghosh et al., 2023). GenEval detects objects from the COCO vocabulary (Lin et al., 2014) and their color, and reports two specific metrics: **% images**: The percentage of images containing all objects given by text prompts; **% prompts**: The percentage of concept sets $\mathcal{Y}$ where at least one generated image contains all objects.

We complement GenEval with three widely-used performance metrics, each defined as $r_k(\mathbf{z}, \boldsymbol{t})$, for an image $\mathbf{z}$ and a prompt $\boldsymbol{t}$. CLIP-Score (r_1) measures text–image alignment by computing the cosine similarity between their CLIP embeddings (Radford et al., 2021). Reward (r_2) uses a learned reward model trained on human preference data to score how closely the image matches the prompt (Xu et al., 2023). VQA (r_3) assesses faithfulness by answering yes-no questions about the prompt and the generated image (Lin et al., 2024). For each metric, we report the following scores averaged across pairs $(\mathcal{Y}, \mathbf{z})$ of a set of concepts $\mathcal{Y}$ and the generated image $\mathbf{z}$: **joint**: $r_k(\mathbf{z}, \boldsymbol{t})$ where $\boldsymbol{t}$ is a multi-concept prompt for $\mathcal{Y}$; **min**: $\min_{\boldsymbol{y}_i \in \mathcal{Y}} r_k(\mathbf{z}, \boldsymbol{t}_i)$ where $\boldsymbol{t}_i$ is a single-concept prompt for $\boldsymbol{y}_i$.

4.4 Concepts as Objects

We first evaluate our method along the axis of generating more than one concept. Here each concept is defined as a distinct object. We assess how well Divide-and-Denoise works for multi-concept generation across both Stable Diffusion and DiT setups.

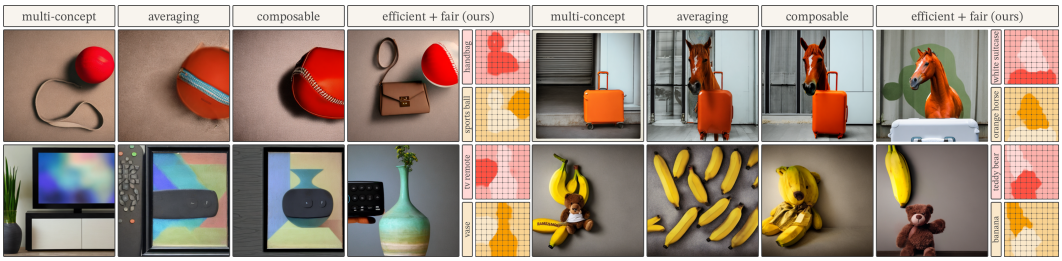

Figure 5: Divide-and-Denoise avoids object overlap, preserves all objects, and correctly attributes colors, outperforming baselines on GenEval. Concept pairs: *handbag+sports ball, tv remote+vase, white suitcase+orange horse, teddy bear+banana.*

Table 1: Performance of Divide-and-Denoise when coordinating 2 and 3 models conditioned on objects using Stable Diffusion.

Players	Coordination Strategy	GenEval ↑		CLIP ↑		Reward ↑		VQA ↑	
		%images	%prompts	joint	min	joint	min	joint	min
2	Averaging	31.25	59.00	26.26	18.64	-0.49	-1.46	0.720	0.610
	Composable Diffusion	36.50	67.00	26.85	18.92	-0.26	-1.30	0.749	0.643
	Multi-Concept Diffusion	53.75	86.00	27.05	18.77	0.28	-1.15	0.753	0.683
	Ours (without fairness)	87.00	98.00	29.91	**21.59**	1.16	-0.42	0.959	0.921
	Ours	**88.50**	**99.00**	**30.02**	21.53	**1.23**	**-0.38**	**0.960**	**0.925**
3	Averaging	1.50	5.00	25.46	16.08	-1.32	-2.06	0.461	0.296
	Composable Diffusion	3.50	13.00	26.82	16.03	-1.06	-1.97	0.472	0.304
	Multi-Concept Diffusion	14.75	43.00	28.45	15.15	-0.14	-1.82	0.532	0.384
	Ours (without fairness)	51.75	88.00	32.68	18.96	1.05	-0.92	0.872	0.773
	Ours	**59.50**	**92.00**	**33.21**	**19.09**	**1.22**	**-0.79**	**0.921**	**0.829**

Table 2: Performance using DiT.

	CLIP ↑		Reward ↑		VQA ↑	
	joint	min	joint	min	joint	min
Avg.	25.57	19.58	-1.00	-1.35	0.644	0.577
Comp.	26.67	20.43	-0.69	-1.11	0.700	0.634
Ours	**29.03**	**22.01**	**0.28**	**-0.46**	**0.868**	**0.808**

Table 3: Performance with conflicting interests.

	CLIP ↑		Reward ↑		VQA ↑	
	joint	min	joint	min	joint	min
Avg.	27.81	19.14	-0.36	-1.45	0.687	0.521
Comp.	29.00	20.20	0.08	-1.23	0.729	0.581
Multi.	29.76	19.82	0.65	-0.88	0.752	0.633
Ours	**31.13**	**21.23**	**1.12**	**-0.55**	**0.905**	**0.815**

Stable Diffusion. For each problem instance, we randomly sample n objects from the COCO vocabulary and define a single-concept prompt for each model i as "`an image with [object`$_i$`]`". In total, we construct 100 unique n-object tuples and evaluate each tuple across 4 different seeds. Results are presented in Table 1, where rows 2 and 3 correspond to $n = 2$ and $n = 3$ object-specific models, respectively. Example images can be found in Figure 5.

We find that improvement over baselines is driven by the efficient division of labor. Fairness improves most metrics further. Observe that the importance of fairness increases as more models participate, since the probability of a model being neglected by an efficient (but not fair) allocation grows. To illustrate the effect of the fairness constraint, we provide a qualitative example in Figure 2.

DiT. Each concept here is represented as a one-hot encoded ImageNet class depicting an object. We select 15 pairs of objects from the ImageNet-1K dataset (Russakovsky et al., 2014) and evaluate each pair across 20 random seeds. We test our method's ability to coordinate pairs of models to generate images containing both objects. Table 2 compares our method against averaging and composable diffusion for $n = 2$.

We note that the multi-concept baseline is unavailable here since no single DiT model can condition on multiple classes. Despite this, Divide-and-Denoise generates images that appear as though they were produced by such a multi-class model. We provide qualitative examples for $n = 2$ and $n = 3$ objects in Figure 4.

4.5 Concepts with Description

We next evaluate how our method compares to baselines when concepts contain greater detail. This simulates a scenario where a single multi-concept model would typically fail. We attach color descriptions to objects, testing whether our method can faithfully bind attributes to the correct objects. We use Stable Diffusion setup and construct single-concept prompts as earlier, but this time with each concept $\boldsymbol{y} \in \mathcal{Y}$ given by a color and object, e.g. "`an image with a orange horse`". We generate 2 object–color descriptions to define a pair of specialized models. In total, we construct 100 unique pairs and evaluate each pair across 4 different seeds. Results are presented in Table 4. Figure 5 provides a qualitative example of correct attribute binding.

Table 4: Performance with descriptive concepts.

	GenEval ↑		CLIP ↑		Reward ↑		VQA ↑	
	%image	%prompt	joint	min	joint	min	joint	min
Averaging	9.00	27.00	28.57	19.81	-0.39	-1.65	0.641	0.500
Composable Diffusion	12.25	32.00	29.65	20.37	-0.09	-1.49	0.658	0.522
Multi-concept Diffusion	12.50	30.00	29.86	19.61	0.10	-1.51	0.596	0.455
Ours	**55.75**	**86.00**	**32.65**	**22.62**	**1.34**	**-0.56**	**0.882**	**0.806**

4.6 Concepts with Conflict

Finally, we evaluate how well Divide-and-Denoise coordinates models with conflicting interests. In order to simulate this, we hand-design 40 scenarios where concepts among the models would naturally conflict. For example, we condition the first model on the concept "`desert`", while the second on the concept "`snowy mountain`". A full list of prompt combinations is provided in Appendix A.5. We use Stable Diffusion setup. As shown in Table 3, Divide-and-Denoise outperforms the coordination baselines. Examples of generated images can be found in Appendix A.6.

5 Conclusion

In this work, we introduced Divide-and-Denoise, a game-theoretic framework for coordinating several pre-trained diffusion models. Our coupled division and denoising processes resolve conflicts between models, prevent concepts or models from being neglected, and outperform baselines across a broad range of metrics including the GenEval benchmark. Notably, our formulation is not tied to a specific model architecture or conditioning mechanism. We demonstrated compelling results with both DiT and Stable Diffusion, using score-based utilities for the former and attention-based utilities for the latter. Attention-based utilities extend naturally to other domains, including text-to-graph Chang & Ye (2025), text-to-audio Liu et al. (2023), and audio-to-image Biner et al. (2024) generation. Our score-based utilities can be further applied in domains without cross-attention conditioning. An open question remains on how to empirically validate utility-driven efficient and fair divisions of non-visual objects where ground truth partitions are even harder to obtain than in images. Our results nevertheless highlight cooperative interaction between pre-trained models as a general recipe for controllable and reusable generative modeling across domains.

ACKNOWLEDGEMENTS

AG and TJ acknowledge support from the Machine Learning for Pharmaceutical Discovery and Synthesis (MLPDS) consortium, and the NSF Expeditions grant (award 1918839) Understanding the World through code.

PB and SK were supported by EU funding ELLIOT 101214398 the Research Council of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI and decision 341763. SK was supported by the UKRI Turing AI World-Leading Researcher Fellowship (EP/W002973/1).

VG acknowledges Saab-WASP (grant 411025), Research Council of Finland (grant 342077), and the Jane and Aatos Erkko Foundation (grant 7001703) for their support.

REFERENCES

Georgios Amanatidis, Haris Aziz, Georgios Birmpas, Aris Filos-Ratsikas, Bo Li, Hervé Moulin, Alexandros A. Voudouris, and Xiaowei Wu. Fair division of indivisible goods: Recent progress and open questions. *Artificial Intelligence*, 315:103841, 2023. doi: 10.1016/j.artint.2022.103841.

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation.(2023). *URL https://arxiv. org/abs/2302.08113*, 2023.

Burak Can Biner, Farrin Marouf Sofian, Umur Berkay Karakaş, Duygu Ceylan, Erkut Erdem, and Aykut Erdem. Sonicdiffusion: Audio-driven image generation and editing with pretrained diffusion models, 2024. URL https://arxiv.org/abs/2405.00878.

G. Black et al. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, 7(3):12:1–12:32, 2019. doi: 10.1145/3319729.

Jinho Chang and Jong Chul Ye. Ldmol: A text-to-molecule diffusion model with structurally informative latent space surpasses ar models, 2025. URL https://arxiv.org/abs/2405.17829.

S. Chi et al. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

Richard Cole, Nikhil R. Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay V. Vazirani, and Sadra Yazdanbod. Convex program duality, fisher markets, and nash social welfare. In *Proceedings of the 18th ACM Conference on Economics and Computation (EC '17)*, pp. 459–460. ACM, 2017. doi: 10.1145/3033274.3085119.

G. Corso et al. Diffdock: Diffusion steps, twists, and turns for molecular docking. In *International Conference on Learning Representations (ICLR)*, 2023.

Guillaume Couairon, Marlène Careil, Matthieu Cord, Stéphane Lathuilière, and Jakob Verbeek. Zero-shot spatial layout conditioning for text-to-image diffusion models. In *ICCV*, 2023.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 8780–8794, 2021.

John P. Dickerson, Jonathan Goldman, Jeremy Karp, Ariel D. Procaccia, and Tuomas Sandholm. The computational rise and fall of fairness. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI '14)*, pp. 1405–1411. AAAI Press, 2014.

Yilun Du, Conor Durkan, Robin Strudel, Joshua B Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International conference on machine learning*, pp. 8489–8510. PMLR, 2023.

Edmund Eisenberg and David Gale. Consensus of subjective probabilities: The pari-mutuel method. *Annals of Mathematical Statistics*, 30(1):165–168, 1959. doi: 10.1214/aoms/1177706379.

Timur Garipov, Sebastiaan De Peuter, Ge Yang, Vikas Garg, Samuel Kaski, and Tommi Jaakkola. Compositional sculpting of iterative generative processes. *Advances in neural information processing systems*, 36:12665–12702, 2023.

Dhruba Ghosh, Hanna Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment, 2023. URL https://arxiv.org/abs/2310.11513.

Amir Hertz, Ron Mokady, Jonathan Tenenbaum, Kfir Aberman, Daniel Cohen-Or, and Yael Pritch. Prompt-to-prompt image editing with cross-attention control. *arXiv preprint arXiv:2208.01626*, 2022.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020a.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020b. URL https://arxiv.org/abs/2006.11239.

J. Jumper et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583–589, 2021.

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 *European conference on computer vision*, pp. 740–755. Springer, 2014.

Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. In *European Conference on Computer Vision*, pp. 366–384. Springer, 2024.

Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D. Plumbley. Audioldm: Text-to-audio generation with latent diffusion models, 2023. URL https://arxiv.org/abs/2301.12503.

Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. Compositional visual generation with composable diffusion models. In *European conference on computer vision*, pp. 423–439. Springer, 2022.

Hayk Manukyan, Andranik Sargsyan, Barsegh Atanyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Hd-painter: High-resolution and prompt-faithful text-guided image inpainting with diffusion models. *arXiv preprint arXiv:2312.14091*, 2023.

Koichi Nishimura and Hanna Sumita. Envy-freeness and maximum nash welfare for mixed divisible and indivisible goods. In *Proceedings of the 22nd ACM Conference on Economics and Computation (EC '21)*, pp. 650–670, 2021. doi: 10.1145/3465456.3467644.

W. Peebles and P. Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

R. Rombach et al. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. URL http://arxiv.org/abs/1409.0575.

V. Sehwag et al. Stretching each dollar: Diffusion training from scratch on a micro-budget. *arXiv preprint arXiv:2402.12223*, 2024.

Marta Skreta, Lazar Atanackovic, Avishek Joey Bose, Alexander Tong, and Kirill Neklyudov. The superposition of diffusion models using the it\^ o density estimator. *arXiv preprint arXiv:2412.17762*, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Raphael Tang, Linqing Liu, Akshat Pandey, Zhiying Jiang, Gefei Yang, Karun Kumar, Pontus Stenetorp, Jimmy Lin, and Ferhan Ture. What the DAAM: Interpreting stable diffusion using cross attention. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5644–5659, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.310. URL https://aclanthology.org/2023.acl-long.310/.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pp. 6000–6010, 2017.

B. Wallace et al. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

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. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

Z. Xu et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

A Appendix

A.1 Proofs of the Theorems

In our proofs, we will rely on the following technical fact:

Lemma 1. *Let π and π' be probability distributions on the same domain. The solution π^* to an unconstrained optimization problem*

$$\max_{\pi} \mathbb{E}_{z\sim\pi} f(z) - \gamma D_{\mathrm{KL}}(\pi||\pi')$$

is given by

$$\pi^*(z) = \frac{\exp(f(z)/\gamma)\pi'(z)}{\int \exp(f(z)/\gamma)\pi'(z)dz}.$$

Proof. It is sufficient to notice that

$$\gamma D_{\mathrm{KL}}(\pi||\pi^*) = -\mathbb{E}_{z\sim\pi} f(z) + \gamma D_{\mathrm{KL}}(\pi||\pi') + C,$$

where C is a constant that does not depend on π. □

Proof of Theorem 2. Let us first consider the setting without fictitious player.

Recall that denoising kernels of individual models are Gaussians with the same covariance:

$$p^i(\cdot \mid \mathbf{x}_t) = \mathcal{N}(\mu_t^i(\mathbf{x}_t), \sigma_t^2 I).$$

Denote the marginal distribution of p^i at the coordinate j as

$$p_j^i(\cdot \mid \mathbf{x}_t) = \mathcal{N}(\mu_j^i(\mathbf{x}_t), \sigma_t^2).$$

For an allocation Q with weights Q_{ij}, consider a distribution p_t^Q defined as

$$p_t^Q(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto \prod_{i=1}^{n}\prod_{j=1}^{m} p_j^i(\mathbf{x}_{t-1}|\mathbf{x}_t)^{Q_{ij}}.$$

Notice that

$$\begin{aligned}
p^Q(\mathbf{x}|\mathbf{x}_t) &\propto \prod_{i=1}^{n}\prod_{j=1}^{m} p_j^i(\mathbf{x}|\mathbf{x}_t)^{Q_{ij}} \\
&= \prod_{i=1}^{n}\prod_{j=1}^{m} \exp\left(\frac{-Q_{ij}(\mathbf{x}_j - \mu_j^i(\mathbf{x}_t))^2}{2\sigma_t^2}\right) \\
&= \exp\left(\frac{-\sum_{i=1}^{n}\sum_{j=1}^{m} Q_{ij}(\mathbf{x}_j - \mu_j^i(\mathbf{x}_t))^2}{2\sigma_t^2}\right) \\
&\propto \exp\left(\frac{-\sum_{i=1}^{n}\sum_{j=1}^{m} [Q_{ij}\mathbf{x}_j^2 - 2\mathbf{x}_j\mu_j^i(\mathbf{x}_t)]}{2\sigma_t^2}\right) \\
&= \exp\left(\frac{-\sum_{j=1}^{m}\mathbf{x}_j^2 + 2\sum_{j=1}^{m}\langle \mathbf{x}_j, \sum_{i=1}^{n} Q_{ij}\mu_j^i(\mathbf{x}_t)\rangle}{2\sigma_t^2}\right) \\
&= \exp\left(\frac{-\|\mathbf{x}\|^2 + 2\langle \mathbf{x}, \sum_{i=1}^{n} Q_i \odot \mu^i(\mathbf{x}_t)\rangle}{2\sigma_t^2}\right),
\end{aligned}$$

and thus,

$$p_t^Q(\cdot|\mathbf{x}_t) = \mathcal{N}\left(\mu_t^Q(\mathbf{x}_t), \sigma_t^2 I\right), \quad \mu_t^Q(\mathbf{x}_t) = \sum_{i=1}^{n} Q_i \odot \mu_t^i(\mathbf{x}_t).$$

For any distribution $p \in \mathbb{P}_t$, we have

$$p(\mathbf{x}) = \prod_{j=1}^{m} p_j(\mathbf{x}).$$

Therefore, the following equality holds

$$\begin{aligned}
\mathbb{E}_{\mathbf{M}\in Q} \sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{i,j} D_{\mathrm{KL}}(p_j(\cdot)||p_j^i(\cdot|\mathbf{x}_t)) &= \sum_{i=1}^{n}\sum_{j=1}^{m} Q_{ij} D_{\mathrm{KL}}(p_j(\cdot)||p_j^i(\cdot|\mathbf{x}_t)) \\
&= \sum_{j=1}^{m}\left[\sum_{i=1}^{n} Q_{ij}\int p_j(\mathbf{x}_j)\log p_j^i(\mathbf{x}_j|\mathbf{x}_t)d\mathbf{x}_j - \sum_{i=1}^{n} Q_{ij} H(p_j)\right] \\
&= \int p(\mathbf{x})\log\prod_{i=1}^{n}\prod_{j=1}^{n} p_j^i(\mathbf{x}_j|\mathbf{x}_t)^{Q_{ij}} d\mathbf{x} - \sum_{j=1}^{m} H(p_j) \\
&= \int p(\mathbf{x})\log p^Q(\mathbf{x}|\mathbf{x}_t)d\mathbf{x} - H(p) + C \\
&= D_{\mathrm{KL}}(p||p_t^Q(|\mathbf{x}_t)) + C,
\end{aligned}$$

where C is a constant that does not depend on p.

By Lemma 1, the composite kernel p_t^c maximizing $\mathcal{F}_t(\mathbf{x}_t, p, Q)$ is given by

$$p_t^c(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto \exp(U_{t-1}(\mathbf{x}_{t-1}, Q)/\alpha_t) p_t^Q(\mathbf{x}_{t-1}|\mathbf{x}_t). \tag{6}$$

Since $U_t(\mathbf{x}, Q)$ is linear jointly in t and $\mathbf{x}$, we have

$$U_{t-1}(\mathbf{x}_{t-1}, Q) = U_t(\mathbf{x}_t, Q) + A(\mathbf{x}_{t-1} - \mathbf{x}_t) - b,$$

where $A = \nabla_{\mathbf{x}} U_t(\mathbf{x}_t, Q)$ and $b = \nabla_t U_t(\mathbf{x}_t, Q)$.

Substituting in equation 6, we obtain

$$\begin{aligned}
p_t^c(\mathbf{x}_{t-1}|\mathbf{x}_t) &\propto \exp(U_{t-1}(\mathbf{x}_{t-1}, Q)/\alpha_t) p_t^Q(\mathbf{x}_{t-1}|\mathbf{x}_t) \\
&= \exp\left(\frac{[U_t(\mathbf{x}_t, Q) + A(\mathbf{x}_{t-1} - \mathbf{x}_t) - b]}{\alpha_t}\right) p_t^Q(\mathbf{x}_{t-1}|\mathbf{x}_t) \\
&\propto \exp\left(\frac{A\mathbf{x}_{t-1}}{\alpha_t} + \frac{-\|\mathbf{x}_{t-1}\|^2 + 2\langle \mathbf{x}_{t-1}, \mu_t^Q(\mathbf{x}_t)\rangle}{2\sigma_t^2}\right) \\
&= \exp\left(\frac{-\|\mathbf{x}_{t-1}\|^2 + 2\langle \mathbf{x}_{t-1}, \mu_t^Q(\mathbf{x}_t) + \sigma_t^2 A/\alpha_t\rangle}{2\sigma_t^2}\right).
\end{aligned}$$

We conclude that

$$p_t^c(\cdot|\mathbf{x}_t) = \mathcal{N}(\mu_t^c, \sigma_t^2 I),$$

where

$$\mu_t^c = \sum_{i=1}^{n} \mu_t^i(\mathbf{x}_t) \odot Q_i + \frac{\sigma_t^2}{\alpha_t}\nabla_{\mathbf{x}_t} U_t(\mathbf{x}_t, Q).$$

Next, assume that the fictitious player was used. Repeating the same argument as above, we obtain

$$\mu_t^c = \sum_{i=1}^{n+1} \mu_t^i(\mathbf{x}_t) \odot Q_i + \frac{\sigma_t^2}{\alpha_t}\nabla_{\mathbf{x}_t} U_t(\mathbf{x}_t, Q).$$

Recall that $p^{n+1}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is defined as

$$p^{n+1}(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto \prod_{i=1}^{n} p^i(\mathbf{x}_{t-1}|\mathbf{x}_t)^{1/n},$$

and thus,

$$\mu_t^{n+1}(\mathbf{x}_t) = \frac{1}{n}\sum_{i=1}^{n} \mu_t^i(\mathbf{x}_t).$$

Hence, we have

$$\mu_t^c = \sum_{i=1}^{n} \mu_t^i(\mathbf{x}_t) \odot \left(Q_i + \frac{1}{n}Q_{n+1}\right) + \frac{\sigma_t^2}{\alpha_t}\nabla_{\mathbf{x}_t} U_t(\mathbf{x}_t, Q).$$

□

Proof of Theorem 1. Substituting definition of the efficiency functional $\mathcal{G}$, equation 3, in the fair division game in equation 1, we obtain the following optimization problem

$$Q_t = \underset{Q\in\mathbb{Q}_t(\mathbf{x}_t)}{\arg\max}\, \mathbb{E}_{\mathbf{M}\sim Q}\left[\sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{i,j} g_{ij}\right] - \beta_t D_{\mathrm{KL}}(Q||Q_{t+1}), \tag{7}$$

where $g_{ij} = u_{ij}(\mathbf{x}_t, t)$ and

$$\mathbb{Q}_t = \left\{ Q \in \Delta(\mathbb{M}_{n,m}) : \mathbb{E}_{\mathbf{M}\sim Q} \sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{i,j}\phi_{ij} \preceq \boldsymbol{b} \right\}.$$

By the proof of Lemma 1, the problem in equation 7 is equivalent to

$$Q_t = \underset{Q\in\mathbb{Q}_t(\mathbf{x}_t)}{\arg\min}\, D_{\mathrm{KL}}(Q||Q^*), \tag{8}$$

where

$$Q^*(\mathbf{M}) \propto \exp\left(\sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{i,j} g_{ij}/\beta\right) Q_{t+1}(\mathbf{M}).$$

□

Assuming that Q_{t+1} is decomposable with weights Q_{ij}^{t+1}, we find that

$$Q^*(\mathbf{M}) \propto \prod_{i=1}^{n}\prod_{j=1}^{m} (e^{g_{ij}/\beta} Q_{ij}^{t+1})^{\mathbf{M}_{i,j}},$$

and thus, the allocation $Q^*(\mathbf{M})$ is decomposable with weights

$$Q_{ij}^* = \frac{e^{g_{ij}/\beta} Q_{ij}^{t+1}}{\sum_{i=1}^{n} e^{g_{ij}/\beta} Q_{ij}^{t+1}}.$$

We will solve the primal optimization problem in equation 8 in its dual form. Let $\phi(\mathbf{M}) = \sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{M}_{i,j}\phi_{ij}$. Assuming the set $\mathbb{Q}(\mathbf{x}_t)$ is non-empty (which is always the case for fairness constraints), the corresponding Lagrangian is

$$\max_{\lambda\geq 0,\gamma}\ \min_{Q}\ \mathcal{L}(Q, \lambda, \gamma),$$

where

$$\mathcal{L}(Q, \lambda, \gamma) = D_{\mathrm{KL}}(Q||Q^*) + \lambda \left(\mathbb{E}_{\mathbf{M}\sim Q}\phi(\mathbf{M}) - \boldsymbol{b}\right) + \gamma \left(\sum_{\mathbf{M}\in\mathbb{M}_{n,m}} Q(\mathbf{M}) - 1\right).$$

Taking derivative with respect to $Q(\mathbf{M})$ we obtain

$$\frac{\partial \mathcal{L}(Q(\mathbf{M}), \lambda, \gamma)}{\partial Q(\mathbf{M})} = \log Q(\mathbf{M}) + 1 - \log Q^*(\mathbf{M}) + \lambda\phi(\mathbf{M}) + \gamma = 0,$$

and thus,

$$Q(\mathbf{M}) = \frac{Q^*(\mathbf{M}) \exp(-\lambda\phi(\mathbf{M}))}{\exp(\gamma + 1)}.$$

We can now plug it into the Lagrangian and take derivative with respect to γ:

$$\frac{\partial \mathcal{L}(Q(\mathbf{M}), \lambda, \gamma)}{\partial \gamma} = \sum_{\mathbf{M}} \frac{Q^*(\mathbf{M}) \exp(-\lambda\phi(\mathbf{M}))}{\exp(\gamma + 1)} - 1 = 0.$$

Hence, we have

$$Q(\mathbf{M}) = \frac{Q^*(\mathbf{M}) \exp(-\lambda\phi(\mathbf{M}))}{Z_\lambda}$$

with $Z_\lambda = \sum_{\mathbf{M}} Q^*(\mathbf{M}) \exp(-\lambda\phi(\mathbf{M}))$ and the dual problem reads

$$\max_{\lambda \geq 0} -\log(Z_\lambda) - \langle \boldsymbol{b}, \lambda \rangle.$$

Let λ^* be a solution to the dual problem. The optimal allocation is expressed as

$$Q_t(\mathbf{M}) = \frac{Q^*(\mathbf{M}) \exp(-\lambda^*\phi(\mathbf{M}))}{Z_{\lambda^*}} \propto \prod_{i=1}^{n}\prod_{j=1}^{m} (Q^*_{ij} e^{-\langle \lambda^*, \phi_{ij}\rangle})^{\mathbf{M}_{i,j}}.$$

Hence, it is also decomposable with weights

$$Q_{ij} = \frac{e^{g_{ij}/\beta} e^{-\langle \lambda^*, \phi_{ij}\rangle} Q^{t+1}_{ij}}{Z_j(\lambda^*)}$$

where

$$Z_j(\lambda^*) = \sum_{i=1}^{n} \exp\left(-\langle \lambda^*, \phi_{ij}\rangle\right) \exp(g_{ij}/\beta) Q^{t+1}_{ij}.$$

We conclude the proof by noticing that $Z_{\lambda^*} = \prod_{j=1}^{m} Z_j(\lambda^*)$.

A.2 Score-Based Utilities

In this section, we derive the simplified form of the score-based utility for coordinate j under the classifier-free guidance setup.

Recall from the main text that for a vector v, we define $\overline{v} = \frac{v}{\|v\|_2}$. The utility of player i for a bundle $\mathbf{M}_{i'}$ is defined as

$$u_i(\mathbf{M}_{i'}) = \overline{\nabla_{\mathbf{x}} q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)}^T \mathrm{diag}(\mathbf{M}_{i'}) \overline{\nabla_{\mathbf{x}} q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)}.$$

Observe that this definition satisfies the additive assumption that we posed on utilities, meaning that the utility of the bundle can be decomposed as

$$u_i(\mathbf{M}_{i'}) = \sum_{j=1}^{m} \mathbf{M}_{i',j} u_{ij}(\mathbf{x}, t),$$

where $u_{ij}(\mathbf{x}, t)$ represents the utility of player i for coordinate j at latent $\mathbf{x}$ and timestep t.

Specifically, we see that

$$u_{ij}(\mathbf{x}, t) = \left[\overline{\nabla_{\mathbf{x}} q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)}\right]_j^2,$$

where $[\cdot]_j$ denotes the j-th coordinate.

In the classifier-free guidance setup, the conditional score is approximated as

$$\nabla_{\mathbf{x}} \log q_t(\mathbf{x}|\boldsymbol{y}_i;\theta_i) \approx s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i).$$

The gradient of a density is related to its score by

$$\nabla_{\mathbf{x}} q_t(\mathbf{x};\theta_i) = q_t(\mathbf{x};\theta_i) \cdot s_t(\mathbf{x};\theta_i).$$

Using Bayes' rule on the log-densities, we have

$$\nabla_{\mathbf{x}} \log q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) = \nabla_{\mathbf{x}} \log q_t(\mathbf{x}|\boldsymbol{y}_i;\theta_i) - \nabla_{\mathbf{x}} \log q_t(\mathbf{x};\theta_i).$$

Therefore, the gradient of the posterior density is

$$\begin{aligned}\nabla_{\mathbf{x}} q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) &= q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) \cdot \nabla_{\mathbf{x}} \log q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) \\ &= q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) \cdot [s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t(\mathbf{x};\theta_i)].\end{aligned}$$

Let $\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i) = s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t(\mathbf{x};\theta_i)$ denote the score difference. After normalization, the scalar $q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)$ cancels out:

$$\begin{aligned}\overline{\nabla_{\mathbf{x}} q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)} &= \frac{q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) \cdot \Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)}{\|q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) \cdot \Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)\|_2} \\ &= \frac{q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) \cdot \Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)}{q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i) \cdot \|\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)\|_2} \\ &= \frac{\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)}{\|\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)\|_2}.\end{aligned}$$

The utility for coordinate j is therefore

$$u_{ij}(\mathbf{x}, t) = \left[\frac{\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)}{\|\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)\|_2}\right]_j^2 = \frac{[\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)]_j^2}{\|\Delta s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i)\|_2^2} = \frac{[s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t(\mathbf{x};\theta_i)]_j^2}{\|s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t(\mathbf{x};\theta_i)\|_2^2}.$$

Denoting the j-th coordinate of the score function as s_t^j, we obtain the desired expression:

$$u_{ij}(\mathbf{x}, t) = \frac{\|s_t^j(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t^j(\mathbf{x};\theta_i)\|_2^2}{\|s_t(\mathbf{x}, \boldsymbol{y}_i;\theta_i) - s_t(\mathbf{x};\theta_i)\|_2^2}.$$

This shows that the utility for each coordinate is proportional to the squared difference between the conditional and unconditional scores at that coordinate, normalized by the total squared norm of the score difference across all coordinates. The key insight is that while the gradient of the posterior density includes the posterior probability $q_t(\boldsymbol{y}_i|\mathbf{x};\theta_i)$ as a factor, this cancels out when we normalize, leaving only the score difference.

A.3 Effect of Guidance

In this section, we present additional experiments analyzing how guidance within the alignment step affects both performance metrics and computational cost. In particular, we study how results change when the parameter α is set to ∞ after the first τ iterations of the generative process.

We use the same experimental setup as in Section 4.4 with $n = 2$ players and Stable Diffusion. All experiments are run on a single AWS EC2 G6e instance with 8 vCPU, 64 GB of memory, and a single 48 GB GPU. Alongside performance metrics, we report the average wall-clock time (in seconds) required to generate a batch of four images. The results are summarized in Table 5. Experiments involving fair allocations use proportional constraints.

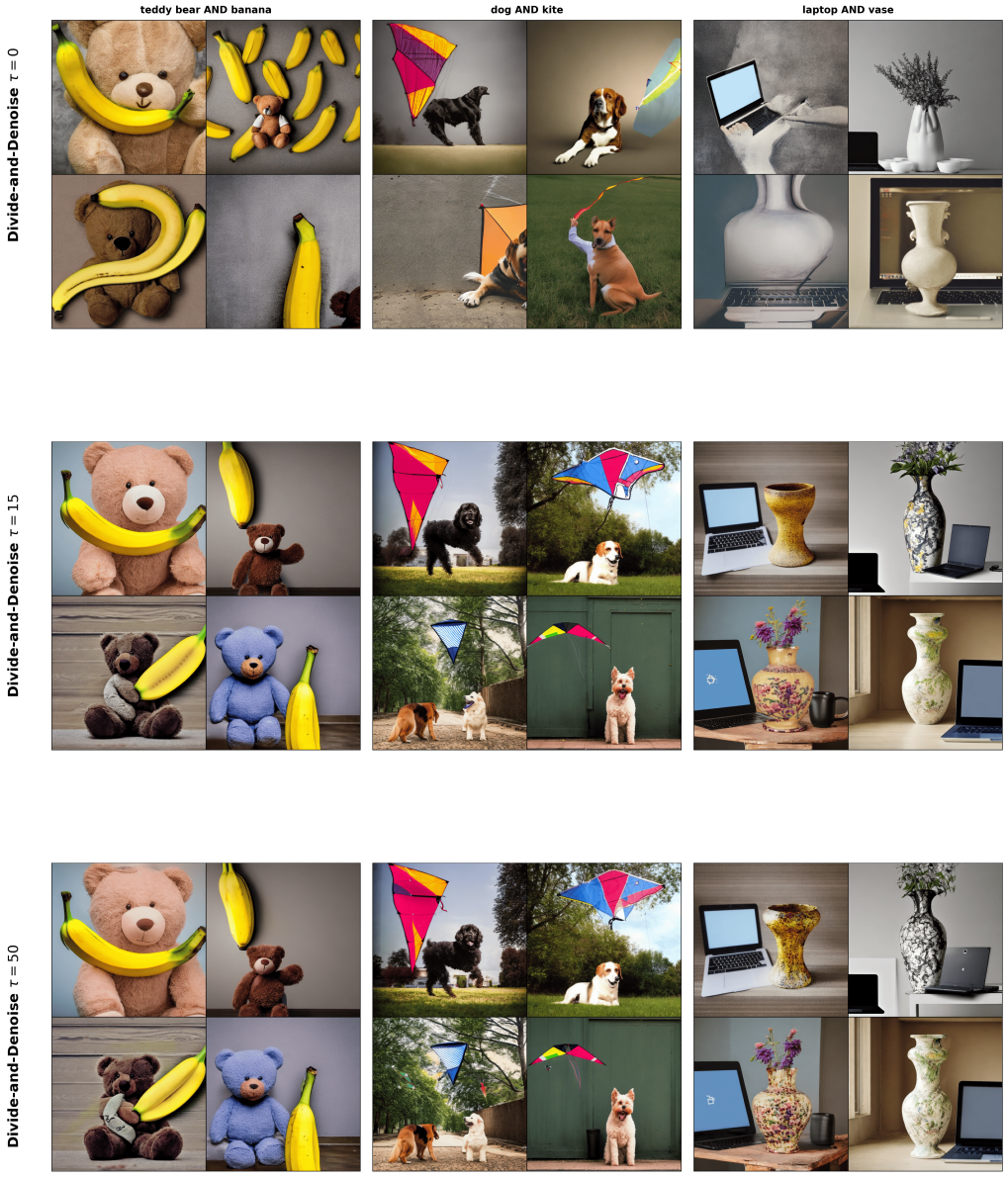

Figure 6: Samples generated by Divide-and-Denoise on GenEval benchmark with varying number of guidance steps τ.

Notably, even with $\tau = 0$ guidance steps, our method consistently outperforms the baselines. Increasing the number of guidance steps yields further improvements, though at the cost of higher computation time. Interestingly, the computational overhead of projecting onto the fair allocation set is substantially larger when no guidance is used. This occurs because explicitly steering the models toward a fair and efficient division encourages better separation of interests, often leading to subsequent allocations already being fair. Without guidance, we find that fairness often needs to be imposed at every step during generation. The qualitative results are provided in Figure 6.

Table 5: Performance of Divide-and-Denoise on coordinating 2 models conditioned on different concepts under varying numbers of guidance steps τ.

Coordination Strategy	GenEval ↑		CLIP ↑		ImageReward ↑		VQA ↑		Time
	%images	%prompts	joint	min	joint	min	joint	min	s/batch
Averaging	31.25%	59.00%	26.26	18.64	-0.49	-1.46	0.720	0.610	9.15
Composable Diffusion	36.50%	67.00%	26.85	18.92	-0.26	-1.30	0.749	0.643	9.05
Multi-Concept Diffusion	53.75%	86.00%	27.05	18.77	0.28	-1.15	0.753	0.683	8.05
Ours (no fairness, $\tau = 0$)	60.00%	93.00%	28.46	20.41	0.49	-0.87	0.883	0.811	11.48
Ours ($\tau = 0$)	62.25%	91.00%	28.41	20.50	0.57	-0.80	0.905	0.835	53.29
Ours (no fairness, $\tau = 15$)	82.50%	99.00%	29.81	21.49	1.11	-0.46	0.947	0.903	15.78
Ours ($\tau = 15$)	84.25%	99.00%	29.74	21.34	1.18	-0.42	0.953	0.909	33.42
Ours (no fairness, $\tau = 50$)	87.00%	98.00%	29.91	21.59	1.16	-0.42	0.959	0.921	26.98
Ours ($\tau = 50$)	88.50%	99.00%	30.02	21.53	1.23	-0.38	0.960	0.925	32.07

A.4 Additional Experimental Results

In addition to the experiments in section 4, we perform several supplementary tests. First, we employ our standard GenEval setup with 2 Stable Diffusion models conditioned on different objects, but this time we use score-based utilities instead of attention-based ones. Results are presented in Table 6. We notice that although this choice of utility significantly decreases performance compared to attention-based one, Divide-and-Denoise still reliably outperforms all baselines including Multi-Concept Diffusion.

Table 6: Performance of Divide-and-Denoise on coordinating 2 models conditioned on different concepts with score-based utilities on GenEval.

Coordination Strategy	GenEval ↑		CLIP ↑		ImageReward ↑		VQA ↑	
	%images	%prompts	joint	min	joint	min	joint	min
Averaging	31.25%	59.00%	26.26	18.64	-0.49	-1.46	0.720	0.610
Composable Diffusion	36.50%	67.00%	26.85	18.92	-0.26	-1.30	0.749	0.643
Multi-Concept Diffusion	53.75%	86.00%	27.05	18.77	0.28	-1.15	0.753	0.683
Ours (without fairness)	45.00%	85.00%	27.29	19.82	-0.08	-1.23	0.808	0.715
Ours (with proportional fairness)	**59.00%**	**91.00%**	**28.13**	**20.38**	**0.26**	**-1.01**	**0.863**	**0.783**

Moreover, we analyze how the performance of Divide-and-Denoise is affected by the choice of the fairness constraints. Across all tasks for the Stable Diffusion setup, we compare our method without any regularization (Efficient Allocation), with only proportional constraints (Efficient + Proportional Allocation), with proportional and equitable constraints (Efficient + Proportional + Equitable Allocation), and finally with proportional and envy-free constraints (Efficient + Proportional + Envy-Free Allocation). Note that we only use last option when coordinating 3 models, since in 2 players' case any proportional allocation is also envy-free. We report all metrics in Table 7.

A.5 Custom Datasets

We constructed a custom dataset comprising 40 examples, each designed with conflicts between individual prompts. Specifically, the first 30 prompts involve either object + semantics or semantics

Table 7: Performance of Divide-and-Denoise in coordinating 2 and 3 models on different concepts under varying fairness criteria.

Task	Allocation	GenEval ↑		CLIP ↑		Reward ↑		VQA ↑	
		%images	%prompt	joint	min	joint	min	joint	min
2 objects	Efficient	87.00%	98.00%	29.91	21.59	1.16	-0.42	0.959	0.921
	Efficient + Proportional	**88.50%**	**99.00%**	30.02	21.53	**1.23**	**-0.38**	**0.960**	**0.925**
	Efficient + Proportional + Equitable	87.00%	98.00%	**30.08**	**21.67**	1.20	-0.41	**0.960**	0.921
3 objects	Efficient	51.75%	88.00%	32.68	18.96	1.05	-0.92	0.872	0.773
	Efficient + Proportional	59.50%	92.00%	33.21	19.09	1.22	-0.79	**0.921**	0.829
	Efficient + Proportional + Equitable	**63.00%**	**94.00%**	**33.24**	**19.35**	**1.29**	**-0.74**	0.919	**0.834**
	Efficient + Proportional + Envy-Free	60.25%	92.00%	33.17	19.05	1.24	-0.77	0.915	0.824
2 objects with color	Efficient	52.50%	**86.00%**	32.42	22.53	1.22	-0.66	0.869	0.791
	Efficient + Proportional	**55.75%**	**86.00%**	**32.65**	**22.62**	**1.34**	**-0.56**	**0.882**	**0.806**
	Efficient + Proportional + Equitable	54.00%	85.00%	**32.65**	**22.62**	1.31	**-0.56**	**0.882**	0.804
concepts with conflict	Efficient	-	-	30.99	21.01	1.05	-0.60	0.896	0.796
	Efficient + Proportional	-	-	31.13	21.23	1.12	-0.55	**0.905**	0.815
	Efficient + Proportional + Equitable	-	-	**31.36**	**21.46**	**1.15**	**-0.52**	0.904	**0.819**
	Efficient + Proportional + Envy-Free	-	-	31.19	21.26	1.12	-0.54	**0.905**	**0.819**

+ semantics compositions: 10 focus on conflicting attributes (e.g., "an image with a blue lake" and "an image with violet trees"), while the remaining 20 capture conflicting semantic combinations (e.g., "an image with a desert" and "an image with a snowy mountain"). The last 10 prompts are of the form semantics + semantics + object, where at least one pairing is conflicting.

Conflicting Attributes (10 prompts):

1. `an image with a blue lake, an image with violet trees`
2. `an image with a green car, an image with a pink forest`
3. `an image with a yellow elephant, an image with a grey desert`
4. `an image with a rainbow-colored dog, an image with a black-and-white city`
5. `an image with a brown flamingo, an image with a purple swamp`
6. `an image with a transparent car, an image with a glowing forest`
7. `an image with a golden cloud, an image with a black ocean`
8. `an image with a white bus, an image with a bright orange snowfield`
9. `an image with a blue cat, an image with a black sofa`
10. `an image with a red eagle, an image with a grey sky`

Conflicting Semantic Combinations (20 prompts):

11. `an image with a desert, an image with a snowy mountain`
12. `an image with a jungle, an image with an icy glacier`
13. `an image with a burning forest, an image with a frozen river`
14. `an image with a tropical beach, an image with a volcanic eruption`
15. `an image with a futuristic city, an image with a medieval castle`
16. `an image with a stormy sky, an image with a calm lake`

17. an image with a carnival, an image with a haunted graveyard
18. an image with an underwater city, an image with a floating island
19. an image with a sunny meadow, an image with a meteor shower
20. an image with a winter tundra, an image with a blooming spring forest
21. an image with snowy mountains, an image with a camel
22. an image with a rainforest, an image with a penguin
23. an image with a rocky cliffside, an image with a telephone booth
24. an image with an iceberg, an image with a windmill
25. an image with a busy highway, an image with a deer
26. an image with a street, an image with a jaguar
27. an image with a blizzard, an image with a giraffe
28. an image with a desert oasis, an image with a moose
29. an image with a rice field, an image with a Ferris wheel
30. an image with a savanna, an image with a skyscraper

Triple Combinations (10 prompts):

31. an image with a desert canyon, an image with snowy peaks, an image with a tropical parrot
32. an image with a modern city skyline, an image with a rural farm, an image with a horse
33. an image with a sunflower field, an image with snowy mountains, an image with a polar bear
34. an image with a grassy soccer field, an image with volcanic ash clouds, an image with a motorcycle
35. an image with a frozen lake, an image with a tropical beach, an image with a palm tree
36. an image with an iceberg, an image with stormy skies, an image with a cow
37. an image with a grassy meadow, an image with a tropical sun, an image with a snowman
38. an image with a sandy beach, an image with the aurora borealis, an image with an elephant
39. an image with a wheat field, an image with a futuristic glass dome city, an image with a steam train
40. an image with a rocky cliffside, an image with a rainbow sky, an image with a boat

A.6 Additional qualitative results

We provide additional qualitative comparisons of Divide-and-Denoise with baselines. The experimental setup is described in Section 4.

For the Stable Diffusion Setup, the images are shown in Figures 7, 8, 9, and 10. Each row corresponds to one coordination mechanism: Averaging, Composable Diffusion, Multi-Concept Diffusion, and Divide-and-Denoise with fairness given by proportional constraints. Each column corresponds to a fixed set of concepts. For each combination of method and concept set, we generate a batch of 4 images.

Figure 7: Qualitative comparison on GenEval benchmark (2 objects)

For the DiT setup, the images are shown in Figure 11. We present all pairs of ImageNet classes used for the quantitative analysis (see Table 2) and plot images generated by Composable Diffusion baseline alongside images generated by our method for the same random seed. Divide-and-Denoise avoids object overlap and blending of concepts in many cases where Composable Diffusion fails to reliably represent both concepts.

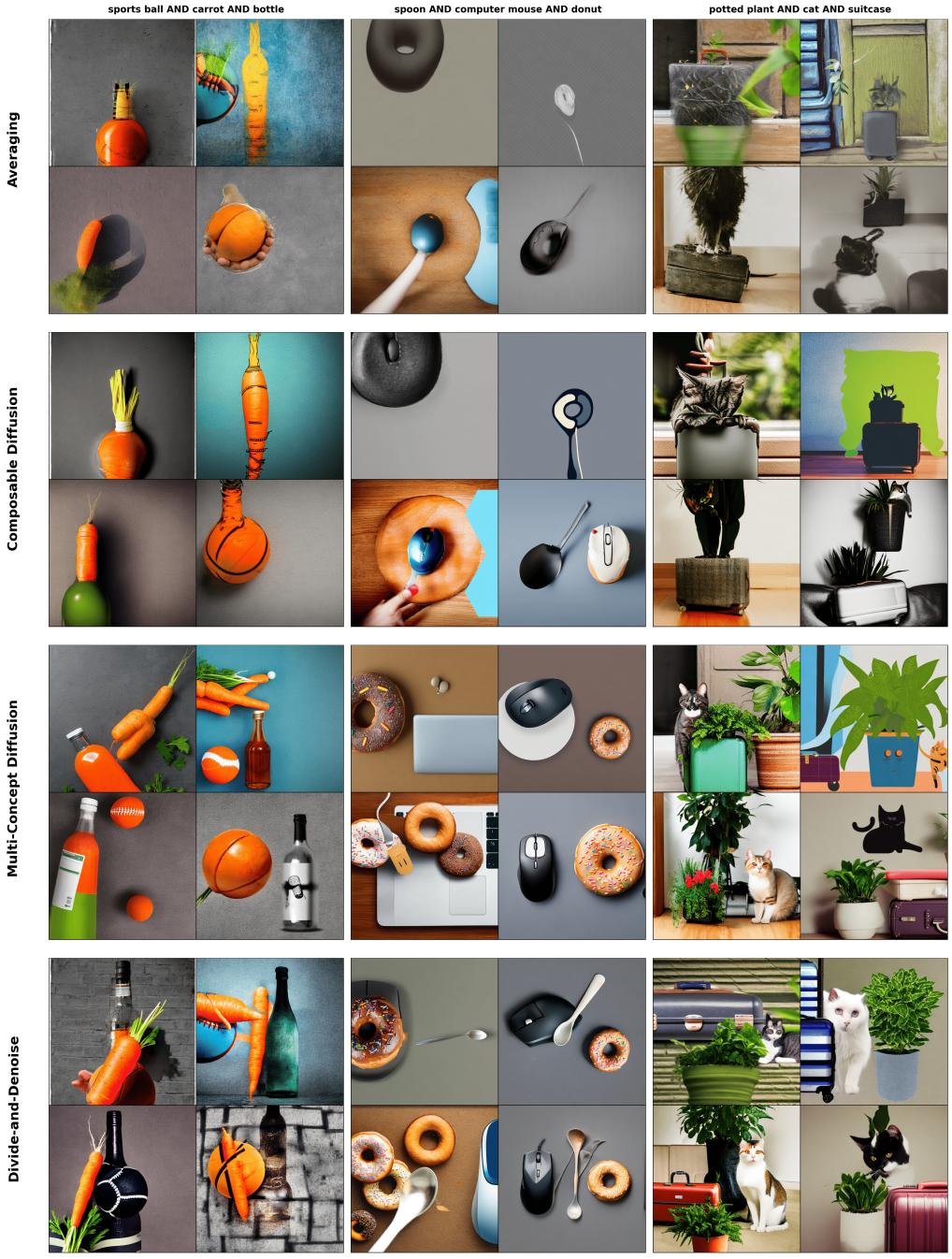

Figure 8: Qualitative comparison on GenEval benchmark (3 objects)

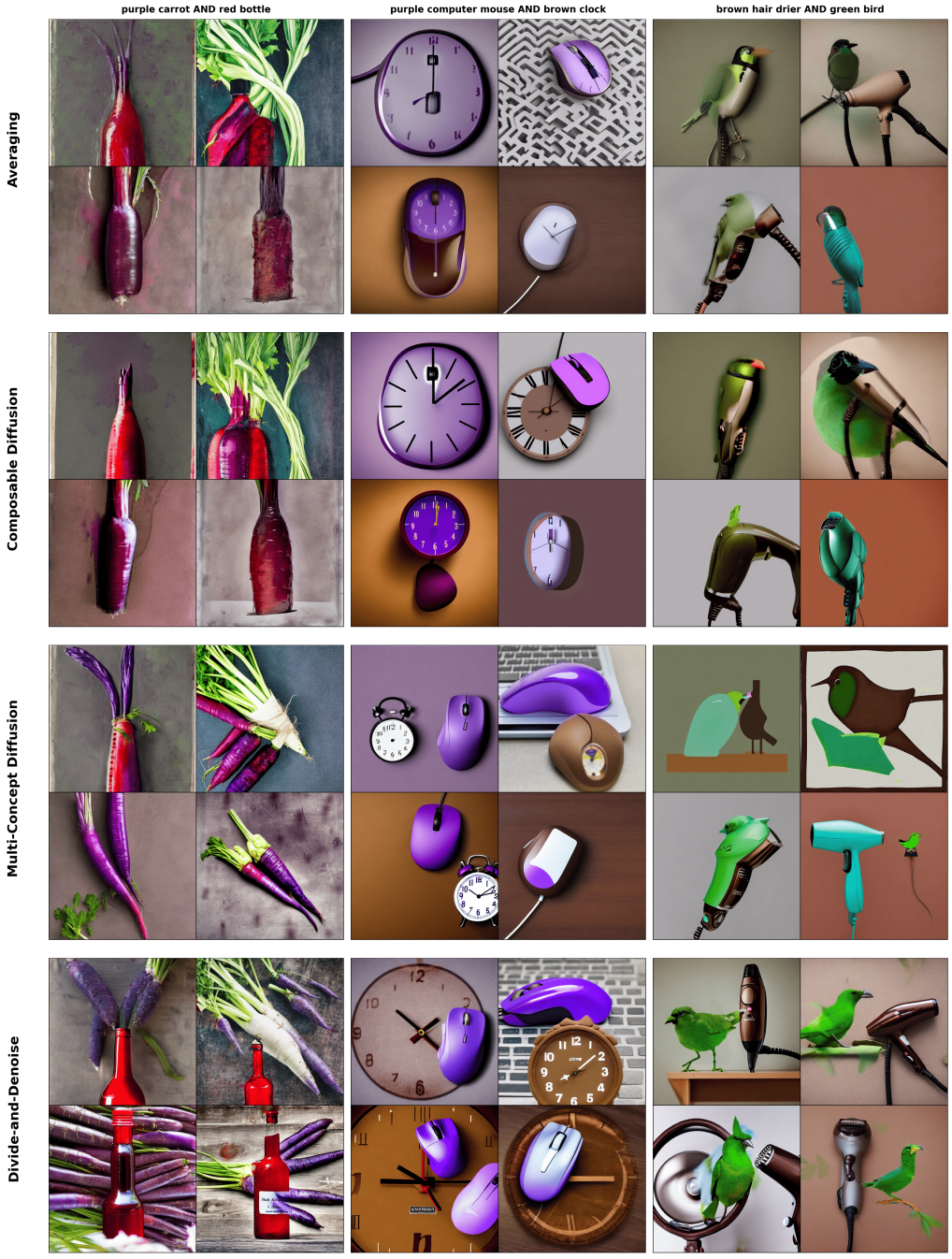

Figure 9: Qualitative comparison on GenEval benchmark (2 objects with color descriptions)

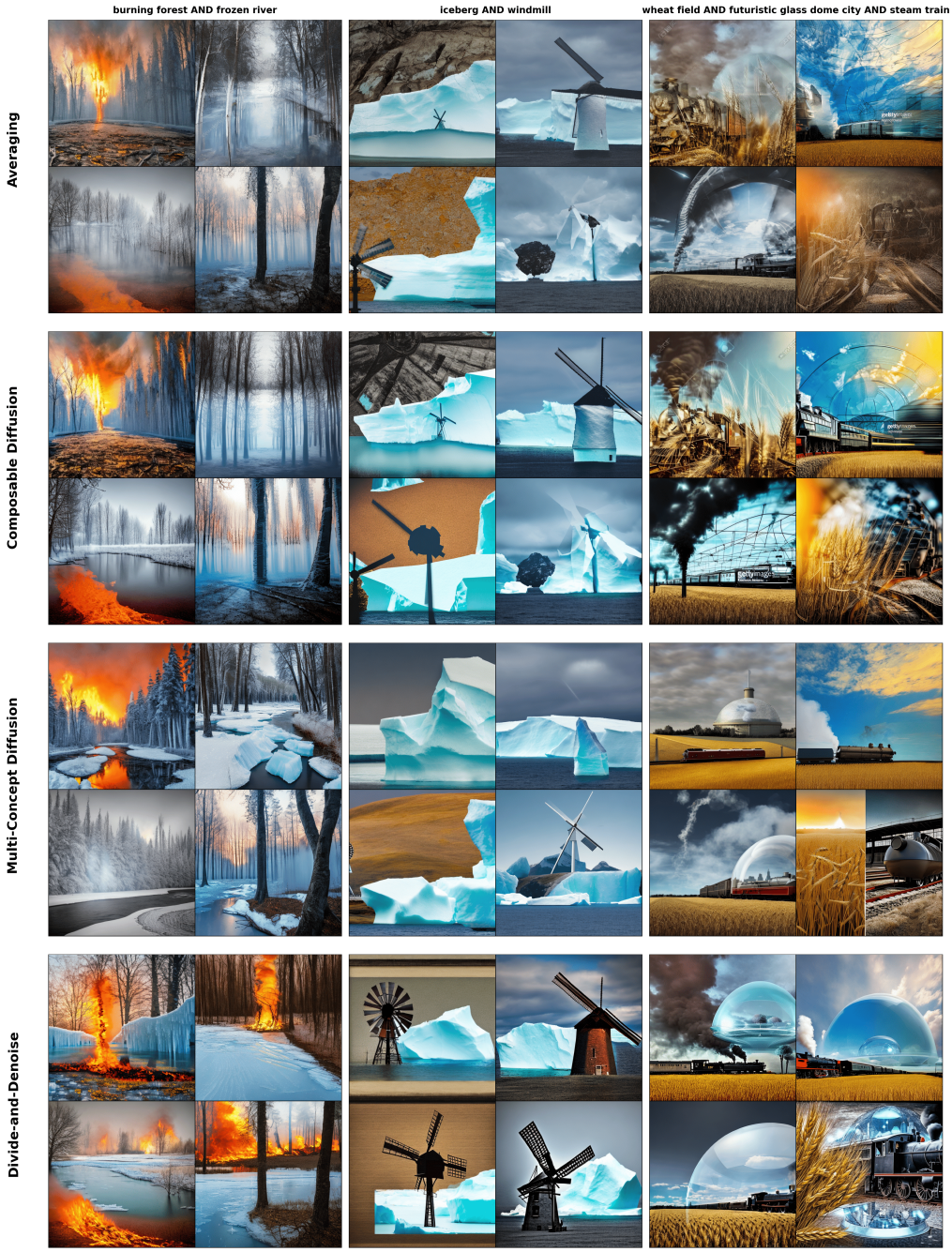

Figure 10: Qualitative comparison on Conflict dataset

Figure 11: Comparison of Composable Diffusion baseline (4 first columns) and Divide-and-Denoise (4 last columns) on a dataset of ImageNet pairs. For each pair of concepts, we show 4 images generated with random seeds. Both models use the same seeds.