SpecTr: Fast Speculative Decoding via Optimal Transport

Ziteng Sun [1] **Ananda Theertha Suresh** [1] **Jae Hun Ro** [1] **Ahmad Beirami** [1] **Himanshu Jain** [1] **Felix Yu** [1] **Michael Riley** [1] **Sanjiv Kumar** [1]

Abstract

Autoregressive sampling from large language models has shown to achieve state-of-the-art results in several natural language tasks. However, autoregressive sampling generates tokens one at a time making it slow, and even prohibitive in certain tasks. One way to speed up decoding is *speculative decoding*: use a smaller model to sample a *draft* (block or sequence of tokens), and then score all tokens in the draft by the desired large language model in parallel. The tokens in the draft are either accepted or rejected based on a statistical method to guarantee that the final output is a valid sample from the large model. In this work, we provide a principled understanding of speculative decoding through the lens of optimal transport (OT) with *membership cost*. This framework can be viewed as an extension of the well-known *maximal-coupling* problem. This new formulation enables us to generalize the sampling method to allow for a set of k candidates at the token-level, leading to an improved optimal membership cost. The optimal solution can be computed via linear programming, whose best-known runtime is exponential in k. We then propose an approximate solution whose acceptance probability is $(1 - 1/e)$-optimal multiplicatively. Moreover, it can be computed in time almost linear with size of token vocabulary. Using this new OT algorithm, we develop a new autoregressive sampling algorithm called *SpecTr*. We experimentally demonstrate that the proposed approach achieves a speedup of 3X, a further 1.36X speedup over speculative decoding on standard benchmarks.

[1]Google Research, New York. Correspondence to: Ziteng Sun <zitengsun@google.com>.

Work presented at the ES-FoMo Workshop at ICML 2023., Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Autoregressive language models have shown to achieve state-of-the-art results in several natural language tasks (Brown et al., 2020; Chowdhery et al., 2022; Thoppilan et al., 2022; Touvron et al., 2023). During inference, given a context $x^t{:=}x(1), x(2) \ldots, x(t)$, an autoregressive model $\mathcal{M}_b$ generates successive tokens $x(t+1), x(t+2), \ldots$ via temperature sampling (Ackley et al., 1985; Ficler & Goldberg, 2017), where the next token $x(t+1)$ is drawn from the temperature-scaled distribution $\mathcal{M}_b(\cdot|x^t)$. If the temperature is zero, i.e., greedy decoding, the next token is determined by the maximum likelihood method i.e., $x(t+1) = \arg\max_{x\in\Omega} \mathcal{M}_b(x|x^t)$, where Ω is the vocabulary. The sampling approach can be further combined with other sampling primitives such as nucleus sampling (Holtzman et al., 2019) and top-k sampling (Fan et al., 2018; Radford et al., 2019). All these approaches are autoregressive decoding methods, where tokens are generated serially one after another, which can be slow or even prohibitive in several applications (Stern et al., 2018). Hence, several techniques have been proposed to improve the speed of generation. Before we proceed further, we first present some notations and a simplified computational model.

Notations. We use $x^{i:j}$ to denote the sequence $x(i), x(i+1), \ldots, x(j)$ and use $x^j = x^{1:j}$. $x(i)$ denotes the i-th entry of x. Subscripts are used to distinguish between different sequences. *e.g.,* x_1^t and x_2^t denote two sequences of length t. We use $[n]$ to denote the set $\{1, \ldots, n\}$.

Computational model.

- **Standard inference.** Given a context x^t, with $O(t^2)$ computation and $O(1)$ time, an autoregressive model $\mathcal{M}_b$ can compute $\mathcal{M}_b(y|x^t)$, the (temperature-scaled) probability of all possible next tokens $y \in \Omega$.
- **Parallelization along the time axis.** Given a context x^t, with $O(t^2)$ computation and $O(1)$ time, an autoregressive model $\mathcal{M}_b$ can compute $\mathcal{M}_b(y|x^i)$, for all $y \in \Omega$ and $i \in \{1, 2, \ldots, t\}$.
- **Parallelization along time and batch axis**[1]**.** Let K be the maximum batch size that can be used during the

[1]This assumption implies that naively batching multiple queries improves decoding throughput, but not latency of a single query.

inference of the autoregressive model. Given several contexts, $x_1^t, x_2^t, \dots x_K^t$, with $O(Kt^2)$ computation and $O(1)$ time, an autoregressive model $\mathcal{M}_b$ can compute $\mathcal{M}_b(y|x_j^i)$, for all $y \in \Omega$, $i \in [t]$, and $j \in [K]$.

The above computation model assumes that parallelizing along time and batch axes does not increase the computation time. It is a simplified characterization of the typical hardware, such as TPUs and GPUs, used in neural network inference. Previous approaches also assume similar computational model to devise faster decoding algorithms (Leviathan et al., 2022; Chen et al., 2023). In practice, there will be some overhead depending on hardware, implementation and resource utilization. In Table 2 (Appendix I), we experimentally show that the assumptions roughly hold for a large transformer model. We also note that there are efficient transformer architectures, which reduces the computation cost from $O(t^2)$ to $O(t \log t)$ (see (Tay et al., 2022) for a survey). Such approaches are orthogonal to the focus of this paper, and they can be easily combined with our approach.

Broadly speaking, multiple previous approaches proposed to guess a few possible future tokens using an efficient model. They then compute several conditional probability distributions from the large model based on the guesses. Computing the distributions takes $O(1)$ time due to parallelization. The guessed tokens are then accepted or rejected based on a statistical method such that the accepted tokens are effectively samples from the large model. When the guesses are good, multiple tokens will be accepted. While this approach incurs the same computation cost as vanilla decoding (assuming computing the guess is cheap), it can significantly improve decoding latency due to parallelization.

The goal of this work is to provide a principled understanding of the above approaches and discuss optimality conditions and algorithmic improvements. We start by providing an overview of speculative decoding and related works.

2. Previous works and speculative decoding

Previous approaches make use of parallelization along the time axis to provide speed-ups. They first predict multiple tokens and validate if these multiple tokens can be generated by the model with the corresponding sampling or decoding scheme. For greedy decoding, multiple tokens can be predicted by a separate model (Stern et al., 2018), aggressive decoding (Ge et al., 2022), or retrieval augmented text (Yang et al., 2023). For sampling, recently (Leviathan et al., 2022; Chen et al., 2023) proposed an algorithm called speculative decoding. Next we provide an overview of this algorithm.

Suppose we have access to a computationally-inexpensive draft model $\mathcal{M}_s$, which also predicts the token given the context and the predictions of $\mathcal{M}_s$ is similar to that of $\mathcal{M}_b$ for many contexts. Suppose we have decoded for t steps and have obtained prefix x^t. The next step of the speculative algorithm can be broken down into three steps.

1. **Draft construction.** The draft model can be used to efficiently and "speculatively" sample L tokens, extending the context to $x(1), \dots, x(t), \tilde{x}(t+1), \dots, \tilde{x}(t+L)$. We keep the conditional probabilities on the next token $\mathcal{M}_s(y \mid x^t, \tilde{x}^{t+1:t+i})$, $\forall i < L$ and $\forall y \in \Omega$.
2. **Conditional probability computation.** After observing the samples, we compute the conditional distributions $\mathcal{M}_b(y \mid x^t, \tilde{x}^{t+1:t+i})$ for each $i < L$ and $\forall y \in \Omega$ in parallel (along time axis) in $O(1)$ time.
3. **Draft selection.** Select first L' of the L tokens and set $x(t+i) = \tilde{x}(t+i)$ for $i \le L'$ given the draft sequence and the conditional probabilities from both models.

After this step, we use $x_1^{t+L'}$ as prefix and sample the next sequence using speculative decoding iteratively. For completeness, we provide the full algorithm in Appendix A. The crux of the above three steps is draft selection, which given a draft sequence and the conditional probabilities from both models, selects a valid sequence such that the output has same distribution as that of the large model. In speculative decoding, this is achieved via recursively applying a token-level maximal coupling algorithm, which is provided in Algorithm 1.

For the draft selection, Algorithm 1 is applied where p is the conditional distribution of the draft model $\mathcal{M}_s(\cdot \mid x^t)$ and q is the conditional distribution of the large model $\mathcal{M}_b(\cdot \mid x^t)$ (which may be further conditioned on the sampled tokens).

Algorithm 1 Token-level maximal coupling

```
Input: Distributions $p, q$, Draft sample $X \sim_{i.i.d.} p$.
1: Compute $p^{\text{res}}$ where $\forall x \in \Omega, p^{\text{res}}(x) = \frac{q(x) - \min\{p(x), q(x)\}}{1 - \sum_{x'} \min\{p(x'), q(x')\}}$.
2: Sample $\eta \sim U(0, 1)$.
3: if $\eta \le \min\left(1, \frac{q(X)}{p(X)}\right)$ then
4:     $Y = X$, $accept = $ True
5: end if
6: Return $Y \sim p^{\text{res}}$, $accept = $ False.
```

Algorithm 1 returns a random variable Y which either is the accepted input X ($accept =$ `True`) or a sample from the residual distribution p^{res} ($accept =$ `False`), which is defined in Step 1 of Algorithm 1. The algorithm is recursively applied as long as the draft tokens are accepted ($accept =$ `True`) to select the first $L' \le L$ tokens from the draft model. Previous works showed that if $X \sim p$, then $Y \sim q$ (Leviathan et al., 2022; Chen et al., 2023). In the case of the draft selection this means that the output of the algorithm is distributed according to $\mathcal{M}_b(\cdot \mid x^t)$, which is exactly the desired outcome. Furthermore

$$\Pr(Y = X) = \sum_{x \in \Omega} \min(p(x), q(x)) = 1 - d_{\mathrm{TV}}(p, q),$$

where d_{TV} is the total variation distance between p and q. The more similar p and q are, the higher the chance $\Pr(Y = X)$, and fewer the number of serial calls to the larger model. In the ideal case, if $p = q$, then $\Pr(Y = X) = 1$, i.e., the draft token is always accepted, and when used for speculative decoding we have $L' = L$. In such a case, based on our computational model (Section 1), assuming the draft model is fast, the speedup is L times.

3. Our contributions

From a theoretical viewpoint, the speculative decoding algorithm raises multiple questions.

- What is the relationship between speculative decoding and the broader literature of sampling in statistics?
- Is speculative decoding optimal in an information-theoretic sense?
- Speculative decoding uses parallelization along time to speed up decoding, would it be possible to use parallelization along batch (number of drafts) to further improve decoding speed?

We provide answers to all the above questions. We first relate the problem of speculative decoding to the broader and well-studied discrete optimal transport theory (Section 4). With this connection, it becomes clear that the token-level draft selection is the optimal solution for optimal transport with indicator cost function and also related to the problem of maximal coupling (Den Hollander, 2012). Based on the connection, we show that one can further speed up the decoding by parallelizing along the batch axis by using multiple drafts from the draft model (Section 5).

More precisely, we formulate the draft selection problem as an discrete optimal transport problem with membership cost. Discrete optimal transport can be solved with a linear program, but the runtime is exponential in batch size, which can be prohibitive. To address this, we propose an approximate solution which achieves a $(1 - 1/e)$-approximation of the optimal acceptance probability (Section E).

With this theoretically motivated algorithm and guarantees, we circle back to speeding up decoding and propose a new algorithm called SpecTr and theoretically show that it can be used to derive valid sequences from the large model (Section 6). We then experimentally demonstrate the benefit of our approach on standard datasets (Section 7).

4. Token-level draft selection as an optimal transport problem

In this section, we formulate token-level draft as an optimal transport problem, where a cost function is associated with whether a draft token is accepted. To simplify notations, we assume the data comes from a discrete domain, but this can be easily generalized.

Definition 4.1 (Coupling)**.** For two probability distributions P over $\mathcal{X}$ and Q over $\mathcal{Y}$, we say a joint distribution π supported over $\mathcal{X} \times \mathcal{Y}$ is a coupling between P and Q if $\forall y \in \mathcal{Y}, \quad \sum_{x \in \mathcal{X}} \pi(x, y) = Q(y), \forall x \in \mathcal{X}, \quad \sum_{y \in \mathcal{Y}} \pi(x, y) = P(x)$. We use $\Pi(P, Q)$ to denote the set of all possible couplings between P and Q.

When it is clear from context, we will overload notation and refer to the probabilistic mapping $f_\pi : \mathcal{X} \to \mathcal{Y}$ introduced by the conditional probability $\pi(y \mid x) := \pi(x, y)/P(x)$ as a coupling, which is also referred to the transport plan from P to Q (Villani et al., 2009).

Definition 4.2 (Optimal Transport (OT) (Villani et al., 2009))**.** For a cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$, the *transportation cost* of a coupling is defined as:

$$C(\pi) = \mathbb{E}_{X,Y \sim \pi}\left[c(X, Y)\right].$$

The *optimal transport plan* is the coupling $\pi \in \Pi(P, Q)$ that minimizes the transportation cost.

With these definitions in place, we can see that with $\mathcal{X} = \mathcal{Y} = \Omega$, which is the alphabet of the tokens, we recover the speculative decoding with the cost function of *indicator cost*, which captures the resampling cost, defined below:

$$\forall x \in \Omega, y \in \Omega, \qquad c(x, y) = \mathbb{1}\left\{y \neq x\right\}.$$

The transportation cost of the coupling will be

$$C(\pi) = \mathbb{E}_{X,Y \sim \pi}\left[\mathbb{1}\left\{Y \neq X\right\}\right] = \mathbb{P}_{X,Y \sim \pi}(Y \neq X).$$

This optimal transport with this specific cost function is also called maximal coupling (Den Hollander, 2012), and the optimal cost is known to be

$$\min_{\pi : \Pi} \mathbb{P}_{X,Y \sim \pi}(Y \neq X) = 1 - d_{\mathrm{TV}}(P, Q). \tag{1}$$

This implies that Algorithm 1 achieves the optimal cost.

5. Optimal transport with multiple draft tokens

In this section, we generalize speculative decoding to allow for multiple drafts. More formally, let $\mathcal{X} = \Omega^k$ for some $k \in \mathbb{N}_+$, which is the set of k draft tokens from Ω and $\mathcal{Y} = \Omega$, which is the space of the final sampled token from the desired distribution. To characterize the resampling cost, we use the cost function of *membership cost*, defined below:

$$\forall x \in \Omega^k, y \in \Omega, \qquad c(x,y) = \mathbb{1}\left\{y \notin S(x)\right\},$$

where $S(x) = \{o \in \Omega \mid o \text{ appears in } x\}$ denotes the set of distinct elements in x. When $k = 1$, this recovers the indicator cost mentioned above. The transportation cost of the coupling will be

$$C(\pi) = \mathbb{E}_{X,Y\sim\pi}\left[\mathbb{1}\left\{Y \notin S(X)\right\}\right] = \mathbb{P}_{X,Y\sim\pi}(Y \notin S(X)). \tag{2}$$

We will also refer to the above cost $C(\pi)$ as the *rejection probability* due to its probabilistic interpretation. And similarly, $\alpha(\pi) := 1 - C(\pi) = \Pr(Y \in S(X))$ will be the *acceptance probability.*

From now on we will use membership cost as the default cost function and refer to the optimal transport solution as *optimal transport with membership cost* (OTM). We use π^* to denote the coupling that minimizes this cost $\pi^* = \arg\min_{\pi\in\Pi(P,Q)} C(\pi)$;[2] and the cost $C(\pi^*)$ is is the *optimal transport cost* between P and Q. $\alpha(P,Q) = 1 - C(\pi^*)$ is the corresponding optimal acceptance probability.

Draft selection with *i.i.d.* draft tokens. In this paper, we will mainly focus on the case when the draft tokens are *i.i.d.* samples from a base distribution. Let p, q be supported over Ω and the goal is to obtain one valid token from q given k *i.i.d.* samples from p. This applies to the practical scenario where there exists a computationally efficient model, from which we can sample multiple independent draft tokens efficiently. We set $P = p^{\otimes k}$, a product distribution whose marginals are all p, and $Q = q$. The OT problem becomes:

$$\min C(\pi) \;\; s.t. \;\; \pi \in \Pi(p^{\otimes k}, q). \tag{3}$$

In this case, we overload notation and denote the *optimal acceptance probability* as $\alpha_k(p,q) := \alpha(p^{\otimes k}, q) = 1 - C(\pi^*)$. To better understand the quantity, we present a few properties and examples of α_k in Appendix B. In particular, we show that $\alpha_k(p,q)$ increases as k increases, implying that multiple drafts can help increase the acceptance probability.

Optimal transport in discrete domain has been studied extensively (Kantorovich, 1942; Pele & Werman, 2009; Guo et al., 2020), and it is shown that the optimal transport problem can be solve in time exponential in k (Appendix D).

Lemma 5.1. *OTM (3) can be solved in time $O(|\Omega|^{O(k)})$.*

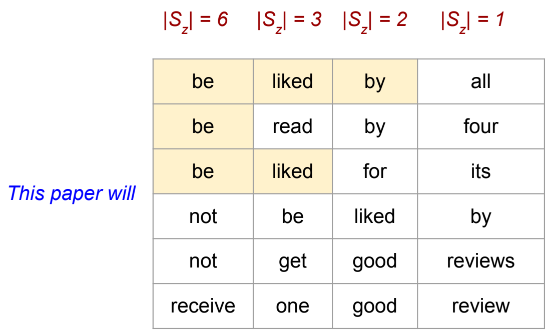
We refer to the optimal coupling obtained above as OTM-k and denote it as $\pi^{\text{OTM}-k}$. For the case of $k = 1$, we have a closed form expression for the optimal acceptance cost (see Equation (1)), whereas for larger values of k, we don't have a general closed form expression. We provide an upper bound on the acceptance probability in Appendix C.

While this solution gives the optimal transportation cost, if we use generic linear program solver to solve (6), to the best of our knowledge, the runtime will be exponential in k, which can be prohibitive when either the vocabulary size Ω or the number of draft tokens k is large. In Appendix E, we will present an approximate solution to the OTM problem and show that for any pair of distributions, it gives a $(1-1/e)$ approximation to the *optimal acceptance probability* α_k.

6. SpecTr: Application of OTM in autoregressive sampling

In this section, we describe how OTM can be used to speed up auto-regressive sampling, which we refer to as SpecTr sampling. Similar to speculative decoding, each step of SpecTr can be decomposed into three phases:

1. **Draft set construction.** Given context x^t, use the draft model sample a set of draft sequences with length L, denoted by $S = \{z^L \sim \mathcal{M}_s(\cdot \mid x^t)\}$. We keep the conditional probabilities $\mathcal{M}_s(y \mid x^t, z^i)$ for all $y \in \Omega, i \le L$ and $z^L \in S$.
2. **Conditional probability computation.** Compute the conditional probabilities on the next token for the large model $\mathcal{M}_b(y \mid x^t, z^i)$ for all $y \in \Omega, i \le L$ and $z^L \in S$ in parallel.
3. **Draft selection.** Select first L' of the L tokens and set $x(t+i) = z(i)$ for $i \le L'$ and some $z \in S$ given the set of draft sequences and the conditional probabilities from both models.



	$\|S_z\|$ = 6	$\|S_z\|$ = 3	$\|S_z\|$ = 2	$\|S_z\|$ = 1
This paper will	be	liked	by	all
	be	read	by	four
	be	liked	for	its
	not	be	liked	by
	not	get	good	reviews
	receive	one	good	review

Figure 1. An example of draft selection in SpecTr with $L = 4$ and $K = 6$. At each step $k = |S_z|$ is the number of draft tokens for OTM. In the first step, we compute the transport plan with $|S_z| = K = 6$ and the sequential selection algorithm will select '*be*', which appeared thrice in our samples. We then compute the transport plan with $|S_z| = 3$ and the sequential selection algorithm will select '*liked*'. We then compute the transport plan with $|S_z| = 2$ and the sequential selection algorithm will select '*by*'. Finally, we compute the transport plan with $|S_z| = 1$ and the sequential selection algorithm will not select any of the drafts.

[2]The existence of optimal coupling in discrete domain is well-known, *e.g.,* see (Villani et al., 2009). When the optimal coupling is not unique, we use π^* to denote one of the optimal couplings.

Draft set with *i.i.d.* draft sequences. Gvien context x^t, a natural way to come up with a set of K drafts is to independently sample K draft sequences from the conditional distribution $\mathcal{M}_s(\cdot \mid x^t)$, *i.e.*,

$$z_1^L, z_2^L, \ldots, z_K^L \sim_{i.i.d.} \mathcal{M}_s(\underbrace{\cdot, \cdot, \ldots \cdot}_{L \text{ dots}} \mid x^t). \tag{4}$$

The draft set construction method in (4) can be generalized to a prefix-tree based algorithm. However, this generalized version did not perform better in experiments. We include this construction in the Appendix H for completeness.

Draft selection with multiple candidates. At a high-level, the algorithm proceeds in a recursive fashion. Given prompt x^t and a candidate set S sampled from $\mathcal{M}_s(\cdot \mid x^t)$, the algorithm first computes the optimal transport plan $f_\pi : \Omega^{|S|} \to \Omega$ from $\mathcal{M}_s(\cdot \mid x^t)^{\otimes |S|}$ to $\mathcal{M}_b(\cdot \mid x^t)$. Then f_π is applied to the first token in each sequence in S to obtained a valid token Y from $\mathcal{M}_b(\cdot \mid x^t)$. If Y is not the last token ($L \geq 2$), we filter out sequences in S whose first token is not Y and denote the remaining sequences as S_{next} and feed it to the algorithm with context (x^t, Y) and draft length $L - 1$. This goes on until we have $L = 1$ or $S_{\text{next}} = \emptyset$.

A sample run of the algorithm is presented in Figure 1. The detailed algorithm (Algorithm 4) and its performance (Theorem G.1) is presented in Appendix G.

7. Experiments

We evaluate the performance of our algorithm and compare it to speculative decoding by following a recipe provided in (Leviathan et al., 2022). We train decoder-only transformer models on the one-billion language benchmark (LM1B) (Chelba et al., 2013) based on the example provided in the FLAX library (Heek et al., 2023). For the draft model, we use a $6M$ parameter transformer model, and for the large model we use a 97M parameter transformer model.

The results of different decoding algorithms are shown in Table 1. The baseline method decodes one token from the large model per serial call, and speculative decoding improves this to ≈ 2.3. The proposed method SpecTr improves upon speculative decoding and increases the number of decoded tokens per serial call as we increase the number of drafts K. We further note that for both Speculative decoding and SpecTr, the number of decoded tokens increases as we increase the block length from 4 to 8. We also note that based on our current implementation, generating the drafts using the small models adds about 10%-15% latency under settings in Table 1. Due to space constraints, we provide additional experiments and details in Appendix I.

Table 1. Experimental results on the LM1B dataset. All results are over 1000 test prompts averaged over three different random seeds.

Algorithm	K	L	Number of decoded tokens per serial call
Baseline	-	-	1.0
Speculative	1	4	2.2
SpecTr	2	4	2.4
SpecTr	4	4	2.7
SpecTr	8	4	**3.0**
Speculative	1	8	2.3
SpecTr	2	8	2.6
SpecTr	4	8	3.0
SpecTr	8	8	**3.3**

References

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. A learning algorithm for boltzmann machines. *Cognitive science*, 9 (1):147–169, 1985.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., Koehn, P., and Robinson, T. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

Chen, C., Borgeaud, S., Irving, G., Lespiau, J.-B., Sifre, L., and Jumper, J. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Dantzig, G. B. Linear programming. *Operations research*, 50(1):42–47, 2002.

Den Hollander, F. Probability theory: The coupling method. *Lecture notes available online (http://websites. math. leidenuniv. nl/probability/lecturenotes/CouplingLectures. pdf)*, 2012.

Fan, A., Lewis, M., and Dauphin, Y. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

Ficler, J. and Goldberg, Y. Controlling linguistic style aspects in neural language generation. *arXiv preprint arXiv:1707.02633*, 2017.

Ge, T., Xia, H., Sun, X., Chen, S.-Q., and Wei, F. Lossless acceleration for seq2seq generation with aggressive decoding. *arXiv preprint arXiv:2205.10350*, 2022.

Guo, W., Ho, N., and Jordan, M. Fast algorithms for computational optimal transport and wasserstein barycenter. In Chiappa, S. and Calandra, R. (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2088–2097. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/guo20a.html.

Heek, J., Levskaya, A., Oliver, A., Ritter, M., Rondepierre, B., Steiner, A., and van Zee, M. Flax: A neural network library and ecosystem for JAX, 2023. URL http://github.com/google/flax.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

Kantorovich, L. V. On the translocation of masses. In *Dokl. Akad. Nauk. USSR (NS)*, volume 37, pp. 199–201, 1942.

Leviathan, Y., Kalman, M., and Matias, Y. Fast inference from transformers via speculative decoding. *arXiv preprint arXiv:2211.17192*, 2022.

Pele, O. and Werman, M. Fast and robust earth mover's distances. In *2009 IEEE 12th international conference on computer vision*, pp. 460–467. IEEE, 2009.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Stern, M., Shazeer, N., and Uszkoreit, J. Blockwise parallel decoding for deep autoregressive models. *Advances in Neural Information Processing Systems*, 31, 2018.

Tay, Y., Dehghani, M., Bahri, D., and Metzler, D. Efficient transformers: A survey. *ACM Computing Surveys*, 55(6): 1–28, 2022.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Villani, C. et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Yang, N., Ge, T., Wang, L., Jiao, B., Jiang, D., Yang, L., Majumder, R., and Wei, F. Inference with reference: Lossless acceleration of large language models. *arXiv preprint arXiv:2304.04487*, 2023.

A. Speculative decoding

Algorithm 2 Speculative Sampling SpecSample.

```
Input: Input sequence $x^t$. Access to a small model $\mathcal{M}_s$ and a large model $\mathcal{M}_b$, block length $L$, end of sequence symbol $eos$.
1: Autoregressively sample $\mathcal{M}_s$ with context $x^t$ to get $L-1$ subsequent samples denoted by $\tilde{x}_{t+1}, \dots, \tilde{x}_{t+L-1}$.
2: Let $\tilde{x}_i = x_i$ for $i \le n$.
3: In parallel compute $p_i = \mathcal{M}_s(\cdot|\tilde{x}^{t+i-1})$ and $q_i = \mathcal{M}_b(\cdot|\tilde{x}^{t+i-1})$ for $1 \le i \le L$.
4: for $i = 1, \dots, L-1$ do
5:     Compute $Y_i, accept = Algorithm\ 1(p_i, q_i, \tilde{x}_{t+i})$
6:     $x_{t+i} = Y_i$.
7:     if $x_{t+i} = eos$ then
8:         Return $x^{t+i}$
9:     end if
10:    if $accept = $ True then
11:        Continue.
12:    else
13:        Return SpecSample($x^{t+i}, \mathcal{M}_s, \mathcal{M}_b, L$).
14:    end if
15: end for
16: Draw $x_{t+L}$ from $q_L$.
17: Return SpecSample($x^{t+L}, \mathcal{M}_s, \mathcal{M}_b, L$).
```

B. Analysis and examples on the optimal acceptance probability α_k.

Lemma B.1. *(Appendix B.1) The optimal acceptance probability satisfies the following properties.*

- ***Monotonicity.*** *For any p, q and $k \ge 1$, $\alpha_k(p,q) \le \alpha_{k+1}(p,q)$.*

- ***Consistency.*** *If $q(x)/p(x)$ is bounded $\forall x \in \Omega$, we have*

$$\lim_{k\to\infty} \alpha_k(p,q) = 1.$$

  Else,

$$\lim_{k\to\infty} \alpha_k(p,q) = \sum_{x\in\Omega} \mathbb{1}\{p(x) > 0\}\, q(x).$$

With the above result, it is clear that increasing k could increase the acceptance probability, particularly when the draft model satisfies $p(x) > 0$ for all $x \in \Omega$.

We illustrate for few simple cases and plot them in Figures 2, 3, 4 and provide analysis for these simple distributions in Appendix B.2. Let $U(d)$ denote a uniform distribution over $[d]$. In Figure 2, we plot the optimal acceptance probability for different uniform functions q as a function of k. Observe that all acceptance probabilities are monotonically increasing and tend to one as $k \to \infty$, however the rate of convergence is vastly different. Furthermore if $\alpha_1(p,q) > \alpha_1(p,q')$, that does not necessarily mean $\alpha_k(p,q) > \alpha_k(p,q')$. In Figure 3, we plot the optimal acceptance probability for different Bernoulli distributions q as a function of k when $p = \text{Ber}(0.25)$. Note that when $p = q$, the acceptance probability is always one (green line), but as we increase / decrease q the acceptance probability decreases. Finally, in Figure 4, we plot the acceptance probability for different values of k as a function of q, when $p = \text{Ber}(0.25)$. In this scenario, note that if k is sufficiently large, say 8, then for most values of q, the acceptance probability is one, however if k is small, then the acceptance probability depends on how close p and q are.

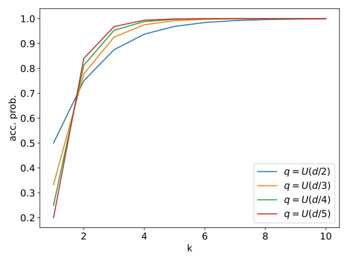


Figure 2. Optimal acc. prob. vs k when $p = U(d)$ for $d = 120$.

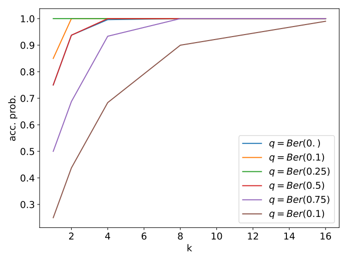


Figure 3. Optimal acc. prob. as a function of k when $p = \text{Ber}(0.25)$.

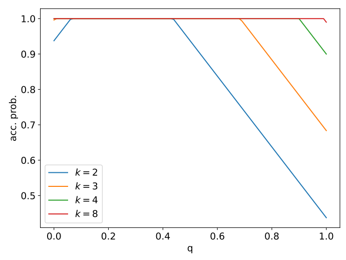


Figure 4. Optimal acc. prob. as a function of q when $p = \text{Ber}(0.25)$.

B.1. Proof of Lemma B.1

We first prove *monotonicity*. By definition,

$$\begin{aligned}\alpha_k(p,q) &= 1 - \min_{\pi \in \Pi(p^{\otimes k}, q)} \Pr_{X^k, Y \sim \pi}\big(Y \notin S(X^k)\big)\\ &= \max_{\pi \in \Pi(p^{\otimes k}, q)} \Pr_{X^k, Y \sim \pi}\big(Y \in S(X^k)\big)\end{aligned}$$

Moreover, for any $\pi \in \Pi(p^{\otimes k}, q)$, we can construct $\pi' \in \Pi(p^{\otimes k+1}, q)$ by setting

$$\forall x^{k+1} \in \Omega^{k+1}, y \in \Omega, \pi'(x^{k+1}, y) = \pi(x^k, x(k+1), y)p(x(k+1)),$$

i.e., adding and independent sample from p to X^k.

Hence we have

$$\begin{aligned}\alpha_k(p,q) &= \max_{\pi \in \Pi(p^{\otimes k}, q)} \Pr_{X^k, Y \sim \pi}\big(Y \in S(X^k)\big)\\ &= \max_{\pi \in \Pi(p^{\otimes k}, q)} \Pr_{X^{k+1}, Y \sim \pi'}\big(Y \in S(X^k)\big)\\ &\leq \max_{\pi \in \Pi(p^{\otimes k}, q)} \Pr_{X^{k+1}, Y \sim \pi'}\big(Y \in S(X^{k+1})\big)\\ &\leq \max_{\pi' \in \Pi(p^{\otimes k+1}, q)} \Pr_{X^{k+1}, Y \sim \pi'}\big(Y \in S(X^{k+1})\big)\\ &= \alpha_{k+1}(p,q).\end{aligned}$$

Next we prove *consistency*. We start with the case when $\forall x \in \Omega, q(x)/p(x) < \infty$. To prove this, we will show that Algorithm 3 with $\gamma_{\max} = \max_{x \in \Omega} q(x)/p(x)$ statisifies

$$\lim_{k \to \infty} \alpha(\pi^{\text{K-SEQ}}_{\gamma_{\max}}) = 1.$$

Notice that by Lemma E.2 and Theorem E.1, $\pi^{\text{K-SEQ}}_{\gamma_{\max}}$ is a valid coupling, and

$$\alpha(\pi^{\text{K-SEQ}}_{\gamma_{\max}}) = 1 - (1 - \beta_{p,q}(\gamma_{\max}))^k,$$

where $\beta_{p,q}(\gamma) = \sum_{x \in \Omega} \min(p(x), \frac{q(x)}{\gamma}) \geq 1/\gamma_{\max} > 0$. Taking $k \to \infty$ concludes the proof.

For the case when $q(x)/p(x)$ is unbounded, there exists $x \in \Omega$ such that $q(x) > 0$ and $p(x) = 0$. Let

$$p_{\text{off}} = \sum_{x \in \Omega} \mathbb{1}\left\{p(x) = 0\right\} q(x).$$

Let x_0 be such that $p(x_0) > 0$. We define q' such that

$$q' = \begin{cases} 0, & \text{if } p(x) = 0, \\ q(x), & \text{if } p(x) > 0 \text{ and } x \neq x_0, \\ q(x) + p_{\text{off}} & \text{if } x = x_0. \end{cases}$$

Then we have $d_{\text{TV}}(q, q') = p_{\text{off}}$, and hence by subadditivity of transport cost,

$$\alpha_k(p, q) \geq \alpha_k(p, q') - p_{\text{off}}.$$

Moreover, we have $\forall x \in \Omega, q'(x)/p(x) < \infty$. Hence

$$\lim_{k \to \infty} \alpha_k(p, q) \geq \lim_{k \to \infty} \alpha_k(p, q') - p_{\text{off}} = 1 - p_{\text{off}} = \sum_{x \in \Omega} \mathbb{1}\{p(x) > 0\} q(x).$$

B.2. Optimal acceptance probability calculations

In this section, we provide a sketch of optimal acceptance probability calculations for results in Figures 2, 3, and 4.

Figure 2: $p = U(d)$ and $q = U(d/r)$. The optimal acceptance probability is

$$\alpha_k(U(d), U(d/r)) = 1 - (1 - 1/r)^k.$$

We first prove $\alpha^k(U(d), U(d/r)) \geq 1 - (1 - 1/r)^k$ by a construction. Let $S(X^k)$ be the set of unique symbols in X^k. Consider the following transport plan, where Y is drawn uniformly from $S(X^k) \cap [d/r]$ and draws a new uniform sample from $[d/r]$ if $S(X^k) \cap [d/r] = \emptyset$. Observe that since $U(d)$ is uniform over $[d]$, this is a valid transport plan and furthermore,

$$\alpha_k(U(d), U(d/r)) \geq \Pr(S(X)^k \cap [d/r] \neq \emptyset) = 1 - (1 - 1/r)^k.$$

The upper bound follows by setting $\Omega_0 = [d] \setminus [d/r]$ in Theorem C.1.

$$\alpha_k(U(d), U(d/r)) \leq \Pr(S(X^k) \cap [d/r] \neq \emptyset) = 1 - (1 - 1/r)^k.$$

Figure 3 and 4: Ber(p) and Ber(q). The optimal acceptance probability is

$$\alpha_k(\text{Ber}(p), \text{Ber}(q)) = \min(q, 1 - (1-p)^k) + \min(1 - q, 1 - p^k). \tag{5}$$

Setting $\Omega_0 = \{0, 1\}$ in Theorem C.1 yields the upper bound. For the lower bound observe that since $\Omega = \{0, 1\}$, $\mathbb{1}\left\{y \notin S(x^k)\right\} < 1$ if and only if x^k is 0^k or 1^k. Hence,

$$\begin{aligned} \alpha_k(\text{Ber}(p), \text{Ber}(q)) &= \pi(X^k \notin \{0^k, 1^k\}) + \max_\pi \{\pi(Y = 0, X^k = 0^k) + \pi(Y = 1, X^k = 1^k)\} \\ &= 1 - p^k - (1-p)^k + \max_\pi \{\pi(Y = 0, X^k = 0^k) + \pi(Y = 1, X^k = 1^k)\}. \end{aligned}$$

Consider the transport plan π given by $\pi(1^k, 1) = \min(p^k, q)$, $\pi(1^k, 0) = p^k - \min(p^k, q)$, $\pi(0^k, 0) = \min((1-p)^k, 1-q)$, and $\pi(0^k, 1) = (1-p)^k - \min((1-p)^k, 1-q)$. It can be checked that this is a valid transport plan. To see this matches the upper bound on the optimal cost from Theorem C.1, notice that

$$\begin{aligned} &1 - p^k - (1-p)^k + \max_\pi \{\pi(Y = 0, X^k = 0^k) + \pi(Y = 1, X^k = 1^k)\} \\ &= 1 - p^k - (1-p)^k + \min(p^k, q) + \min((1-p)^k, 1-q). \end{aligned}$$

If $p^k \leq q$ and $(1-p)^k \leq 1 - q$, then the above equation simplifies to 1 and (5) also simplifies to 1. If $p^k > q$ and $(1-p)^k \leq 1 - q$, then the above equation simplifies to $1 + q - p^k$ and (5) also simplifies to the same quantity. Similarly, the proof applies for $p^k \leq q$ and $(1-p)^k > 1 - q$.

C. Upper bound on α_k

Even though, we don't have a closed form solution for general k, we provide an information-theoretic upper bound in the next theorem. For the case of $k = 1$, this upper bound matches the optimal acceptance probability of previous results.

Theorem C.1. *For any two distributions p, q and $\forall k \geq 1$, we have*

$$\alpha_k(p,q) \leq \min_{\Omega_0 \subset \Omega} \left\{ \sum_{y \in \Omega_0} \min\left\{q(y), 1-(1-p(y))^k\right\} + \sum_{x^k \in \Omega^k} \min\{\prod_{i=1}^k p(x_i), \sum_{y \in x^k \cap \Omega_0^c} q(y)\} \right\}.$$

Proof. It would be enough to show that for any $\pi \in \Pi(p^{\otimes k}, q)$, and any $\Omega_0 \subset \Omega$, we have

$$\Pr\left(Y \in S(X^k)\right) \leq \sum_{y \in \Omega_0} \min\left\{q(y), 1-(1-p(y))^k\right\} + \sum_{x^k \in \Omega^k} \min\{\prod_{i=1}^k p(x_i), \sum_{y \in S(x^k) \cap \Omega_0^c} q(y)\}.$$

$$\begin{aligned}
&\Pr\left(Y \in S(X^k)\right) \\
&= \sum_{y \in \Omega} \sum_{x^k \in \Omega^k} \Pr\left(X^k = x^k, Y = y\right) \cdot \mathbb{1}\left\{y \in S(x^k)\right\} \\
&= \sum_{y \in \Omega_0} \sum_{x^k \in \Omega^k} \Pr\left(X^k = x^k, Y = y\right) \cdot \mathbb{1}\left\{y \in S(x^k)\right\} \\
&+ \sum_{y \in \Omega_0^c} \sum_{x^k \in \Omega^k} \Pr\left(X^k = x^k, Y = y\right) \cdot \mathbb{1}\left\{y \in S(x^k)\right\} \\
&= \sum_{y \in \Omega_0} \Pr\left(y \in S(x^k), Y = y\right) + \sum_{x^k \in \Omega^k} \sum_{y \in S(x^k) \cap \Omega_0^c} \Pr\left(X^k = x^k, Y = y\right) \\
&\leq \sum_{y \in \Omega_0} \min\{\Pr\left(y \in S(x^k)\right), q(y)\} + \sum_{x^k \in \Omega^k} \min\{\Pr\left(X^k = x^k\right), \sum_{y \in S(x^k) \cap \Omega_0^c} q(y)\} \\
&= \sum_{y \in \Omega_0} \min\{1-(1-p(y))^k, q(y)\} + \sum_{x^k \in \Omega^k} \min\{\prod_{i=1}^k p(x_i), \sum_{y \in S(x^k) \cap \Omega_0^c} q(y)\}.
\end{aligned}$$

□

D. Details on solving OT with linear program.

Optimal transport in discrete domain has been studied extensively (Kantorovich, 1942; Pele & Werman, 2009; Guo et al., 2020), and it is shown that the optimal transport problem is equivalent to the following linear programming problem:

$$\begin{aligned}
&\min \sum_{x \in \Omega^k} \sum_{y \in \Omega} \pi(x,y) \mathbb{1}\left\{y \notin S(x)\right\} \\
&s.t. \quad \forall y \in \Omega, \sum_x \pi(x,y) = Q(y) \\
&\qquad \forall x \in \Omega^k, \sum_y \pi(x,y) = P(x) \\
&\forall x \in \Omega^k, y \in \Omega, \pi(x,y) \geq 0.
\end{aligned} \tag{6}$$

Linear programming can be solved in time polynomial in the number of variables and constraints (Dantzig, 2002; Pele & Werman, 2009). Linear program in (6) has $|\Omega|^{k+1}$ variables and $|\Omega|^k + |\Omega|$ equality constraints.

E. Approximate OTM via k-sequential selection

In this section, we present sequential selection algorithm (K-SEQ), an approximate solution to the optimal transport problem in Equation (3), which can be efficiently computed in time almost linear in $|\Omega|$ and logarithmic in k. The algorithm is presented in Algorithm 3.

Algorithm 3 k-sequential selection algorithm (K-SEQ).

```
Input: Distributions $p, q$, samples $X_1, \ldots, X_k \sim_{i.i.d.} p$. $\gamma \in [1, k]$ : division factor.
1: Let $\beta_{p,q}(\gamma) = \sum_{x \in \Omega} \min(p(x), q(x)/\gamma)$ and $p_{\text{acc}} = 1 - (1 - \beta_{p,q}(\gamma))^k$. Compute $p^{\text{res}}$ where
```

$$\forall x \in \Omega, p^{\text{res}}(x) = \frac{q(x) - \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{p_{\text{acc}}}{\beta_{p,q}(\gamma)}}{1 - p_{\text{acc}}}. \tag{7}$$

```
2: for $i = 1, 2, \ldots, k$ do
3:     Sample $\eta_i \sim U(0, 1)$.
4:     if $\eta_i \leq \min\left(1, \frac{q(X_i)}{\gamma \cdot p(X_i)}\right)$ then
5:         $Y = X_i$.
6:         Return $Y = X_i$.
7:     end if
8: end for
9: Return $Y \sim p^{\text{res}}$.
```

At a high-level, the algorithm goes over all k samples $X_1, \ldots, X_k$ generated from p sequentially, and decides on whether to accept each X_i based on the ratio $q(X_i)/p(X_i)$. The algorithm output the first accepted sample or result from a residual distribution p^{res} if none of the samples is accepted. To control the probability of accepting an $x \in \Omega$ with probability larger than $q(x)$. We choose an appropriate $\gamma \in [1, k]$ and accept X_i with probability $\min(1, q(X_i)/(\gamma \cdot p(X_i)))$ instead of $\min(1, q(X_i)/(p(X_i)))$ as in the single-draft case. Further, notice that Algorithm 3 recovers Algorithm 1 when $\gamma = k = 1$.

In Theorem E.1, we show that family of joint distributions induced by Algorithm 3 is indeed valid transportation plans from $p^{\otimes k}$ to q. Moreover, to find the best transportation plan within the family, we only need to search over a single parameter γ, which reduces the computation cost significantly. We also show that searching over this sub-family of couplings won't decrease the optimal acceptance probability by a multiplicative constant. The performance of Algorithm 3 is stated in Theorem E.1.

Theorem E.1. *Let $\beta_{p,q}(\gamma) = \sum_{x \in \Omega} \min(p(x), \frac{q(x)}{\gamma})$ and γ^* be the solution to the identity below.*

$$1 - (1 - \beta_{p,q}(\gamma))^k = \gamma \beta_{p,q}(\gamma). \tag{8}$$

When $\gamma \geq \gamma^*$, the coupling $\pi_\gamma^{\text{K-SEQ}}$ introduced by Algorithm 3 is a valid transport plan from $p^{\otimes k}$ to q and

$$\alpha(\pi_\gamma^{\text{K-SEQ}}) \geq p_{\text{acc}} = 1 - (1 - \beta_{p,q}(\gamma))^k.$$

And when $\gamma = \gamma^*$, we have

$$\alpha(\pi_{\gamma^*}^{\text{K-SEQ}}) \geq (1 - e^{-1}) \alpha_k(p, q).$$

Moreover, γ^* can be computed in time $O(|\Omega| \log k)$.

We provide the proof in Appendix E.1 To see why γ^* can be computed efficiently, we notice that the function $f(\gamma)$ defined below has a root in $[1, k]$. Moreover it is continuous and monotonically increasing when $\gamma \in [1, k]$:

$$f(\gamma) = 1 - (1 - \beta_{p,q}(\gamma))^k - \gamma \beta_{p,q}(\gamma).$$

Hence the solution to Equation (8) can be efficiently computed using binary search over the set $[1, k]$.

In fact, although Theorem E.1 only guarantees that Algorithm 3 can achieve an acceptance rate at least a $(1 - e^{-1})$ factor of the optimal acceptance rate, empirically we observe that the acceptance probabilities are much closer for certain distributions. For example, for the example in Figure 2, the proposed algorithm is in fact optimal. We list few more comparisons in the Appendix F.

E.1. Proof of Theorem E.1

We start by proving the following lemma on γ^*.

Lemma E.2. *Let*

$$f(\gamma) = 1 - (1 - \beta_{p,q}(\gamma))^k - \gamma\beta_{p,q}(\gamma).$$

Then we have Let γ^* be the solution to Equation (8). Then when $d_{\mathrm{TV}}(p, q) \in (0, 1)$,

- *$f(\gamma)$ is monotone in γ in $[1, \infty)$;*
- *$\gamma^* \in [1, \min\{k, \max_x \frac{q(x)}{p(x)}\}]$.*

Proof. It would enough to prove the followings: (1) $f(\gamma)$ is monotone in γ in $[1, \infty)$; (2) $f(1) \geq 0$; (3) $f(k) \leq 0$; (4) $f(\max_x \frac{q(x)}{p(x)}) \leq 0$.

To see (1), since $\beta_{p,q}(\gamma)$ is decreasing in γ, so is $1 - (1 - \beta_{p,q}(\gamma))^k$. Moreover, $\gamma\beta_{p,q}(\gamma) = \sum_x \min\{\gamma p(x), q(x)\}$, which is non-decreasing in γ. Hence we have $1 - (1 - \beta_{p,q}(\gamma))^k - \gamma\beta_{p,q}(\gamma)$ is decreasing.

To see (2), note that when $\gamma = 1$, $\beta_{p,q}(\gamma) = 1 - d_{\mathrm{TV}}(p, q)$. Hence we have

$$1 - (1 - \beta_{p,q}(1))^k = 1 - d_{\mathrm{TV}}(p, q)^k \geq 1 - d_{\mathrm{TV}}(p, q).$$

When $\gamma = k$, (3) holds since for $x \in [0, 1]$, we have $1 - (1 - x)^k \leq kx$. Moreover, when $\gamma = \max_x \frac{q(x)}{p(x)} > 1$, we have $\beta_{p,q}(\gamma) = 1/\gamma$ and (4) holds since

$$1 - (1 - \beta_{p,q}(\gamma))^k = 1 - (1 - 1/\gamma)^k < 1 = \gamma \cdot 1/\gamma.$$

$\square$

Next we prove Theorem E.1, we will break the proof into four parts: (1) computation efficiency; (2) $\pi_\gamma^{\text{K-SEQ}}$ is a valid transport plan; (3) acceptance probability; (4) optimality guarantee of $\pi_{\gamma^*}^{\text{K-SEQ}}$.

Computation efficiency. Note that the lemma immediately implies that γ^* can be computed up to arbitrary accuracy δ in time $|\Omega| \log(k/\delta)$ using binary search over $[1, k]$.

Valid transport plan. We next prove that $\pi_\gamma^{\text{K-SEQ}}$ is a valid transport plan when $\gamma \geq \gamma^*$. By Lemma E.2, when $\gamma \geq \gamma^*$, we have $1 - (1 - \beta_{p,q}(\gamma))^k \geq \gamma\beta_{p,q}(\gamma)$. Recall that $p_{\mathrm{acc}} = 1 - (1 - \beta_{p,q}(\gamma))^k$, and

$$\forall x \in \Omega, p^{\mathrm{res}}(x) = \frac{q(x) - \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{p_{\mathrm{acc}}}{\beta_{p,q}(\gamma)}}{1 - p_{\mathrm{acc}}}.$$

$\forall x \in \Omega$, we have

$$\min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{p_{\mathrm{acc}}}{\beta_{p,q}(\gamma)} \leq \frac{1 - (1 - \beta_{p,q}(\gamma))^k}{\gamma\beta_{p,q}(\gamma)} q(x) \leq q(x),$$

this implies $p^{\mathrm{res}}(x) \geq 0$ for all $x \in \Omega$. Moreover,

$$\sum_{x \in \Omega} p^{\mathrm{res}}(x) = \sum_{x \in \Omega} \frac{q(x) - \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{p_{\mathrm{acc}}}{\beta_{p,q}(\gamma)}}{1 - p_{\mathrm{acc}}} = 1.$$

Hence p^{res} is a valid distribution. It remains to show that the marginal of Y is q. We first compute the probability of the output $Y = x$. Note that probability that $Y = X_1$ is

$$p(X_1) \min\left(1, \frac{q(X_1)}{\gamma p(X_1)}\right) = \min\left(p(X_1), \frac{q(X_1)}{\gamma}\right).$$

Hence

$$\Pr(Y = X_1 = x) = \min\left(p(x), \frac{q(x)}{\gamma}\right).$$

Therefore,

$$\Pr(Y = X_1) = \sum_x \min\left(p(x), \frac{q(x)}{\gamma}\right) = \beta(\gamma).$$

Similarly, probability that

$$\Pr(Y = X_2 = x) = \Pr(Y \neq X_1)\Pr(Y = X_2 | Y \neq X_1) = (1 - \beta_{p,q}(\gamma))\min\left(p(x), \frac{q(x)}{\gamma}\right).$$

Hence,

$$\begin{aligned}
&\Pr\big(Y = x, \text{one of } X^k \text{ is accepted}\big) \\
&= \sum_{i=0}^{k-1} \Pr\left(X_1, \ldots, X_i \text{ are rejected}, X_{i+1} \text{ is accepted, and } X_{i+1} = x\right) \\
&= \sum_{i=0}^{k-1} (1 - \beta_{p,q}(\gamma))^i \cdot p(x) \cdot \min\left\{1, \frac{q(x)}{p(x)\gamma}\right\} \\
&= \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \cdot \sum_{i=0}^{k-1} (1 - \beta_{p,q}(\gamma))^i \\
&= \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{1 - (1 - \beta_{p,q}(\gamma))^k}{\beta_{p,q}(\gamma)}
\end{aligned}$$

Summing over all symbols x yields

$$\Pr(\text{one of } X^k \text{ is accepted}) = \sum_x \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{1 - (1 - \beta_{p,q}(\gamma))^k}{\beta_{p,q}(\gamma)} = 1 - (1 - \beta_{p,q}(\gamma))^k.$$

Hence we have

$$\begin{aligned}
\Pr(Y = x) &= \Pr\big(Y = x, \text{one of } X^k \text{ is accepted}\big) + (1 - \text{one of } X^k \text{ is accepted}) p^{\text{res}}(x) \\
&= \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{1 - (1 - \beta_{p,q}(\gamma))^k}{\beta_{p,q}(\gamma)} \\
&\qquad + \big(1 - (1 - \beta_{p,q}(\gamma))^k\big) \frac{q(x) - \min\left\{p(x), \frac{q(x)}{\gamma}\right\} \frac{1 - (1 - \beta_{p,q}(\gamma))^k}{\beta_{p,q}(\gamma)}}{1 - (1 - \beta_{p,q}(\gamma))^k} \\
&= q(x).
\end{aligned}$$

Acceptance probability. The acceptance probability holds since

$$\alpha(\pi_\gamma^{\text{K-SEQ}}) \geq \Pr(\text{one of } X^k \text{ is accepted}) = 1 - (1 - \beta_{p,q}(\gamma))^k.$$

Optimality guarantee of $\pi_{\gamma^*}^{\text{K-SEQ}}$. It can be seen that $\beta(\gamma)$ is decreasing in γ, and so is $1 - (1 - \beta_{p,q}(\gamma))^k$. Hence we have

$$\alpha(\pi_{\gamma^*}^{\text{K-SEQ}}) \geq 1 - (1 - \beta_{p,q}(\gamma^*))^k \geq 1 - (1 - \beta_{p,q}(k))^k = c_k(p, q) \cdot \min\{kp(x), q(x)\},$$

where

$$c_k(p, q) = \frac{1 - (1 - \beta_{p,q}(k))^k}{k\beta_{p,q}(k)} \in [1 - (1 - 1/k)^k, 1).$$

The inclusion holds since$f(x) = \frac{1-(1-x)^k}{kx}$ in monotonically decreasing when $x \in (0, 1/k]$ and $f(1/k) = 1-(1-1/k)^k, \lim_{x\to 0^+} f(x) = 1$.

Moreover, $\forall x \geq 0, kx \geq 1-(1-x)^k$. Hence we have

$$\begin{aligned}\alpha(\pi^{\text{K-SEQ}}_{\gamma^*}) &\geq \left(1-(1-1/k)^k\right) \cdot \min\{kp(x), q(x)\} \\ &\geq \left(1-(1-1/k)^k\right) \min\{1-(1-p(x))^k, q(x)\} \\ &\geq \left(1-(1-1/k)^k\right) \alpha_k(p,q),\end{aligned}$$

where the last inequality is due to the upper bound in Theorem C.1 with $\Omega_0 = \Omega$.

F. Comparisons between OTM-k and K-SEQ

F.1. Examples where the approximate algorithm is optimal

In this section, we show that for the example in Figures 2, K-SEQ achieves the optimal acceptance accuracy. In this case, $p = U(d)$ and $q = U(d/r)$. Recall that the optimal acceptance probability is

$$\alpha_k(U(d), U(d/r)) = 1-(1-1/r)^k.$$

For $U(d)$ and $U(d/r)$, we have

$$\beta(\gamma) = \sum_{x\in[d]} \min\{p(x), q(x)/\gamma\} = \frac{1}{\max\{r,\gamma\}}.$$

And hence solving $1-(1-\beta(\gamma))^k = \gamma\beta(\gamma)$ gives $\gamma^* = r(1-(1-1/r)^k)$. And be Theorem E.1, we have

$$\alpha(\pi^{\text{K-SEQ}}_{\gamma^*}) \geq 1-(1-\beta(\gamma^*))^k = 1-(1-1/r)^k.$$

And the equality holds since this an upper bound for any coupling.

F.2. Gap between OTM-k and K-SEQ

To see how OTM-k and K-SEQ compare in general, we numerically compute the acceptance probability for a pair of compressed conditional distributions. We feed the prompt

"He said he also has asked prosecutors to"

to both large and small models used in Section 7 and obtain the conditional distributions p, q. To make the computation feasible for OTM-k, we take the set of top 10 elements from p, q respectively and set the support S to be the union of the two sets. Then we set p' and q' to be the normalized distribution of p and q over the set S.

We then numerically compute the acceptance probability for the optimal transport solution in Section 5 and the approximate solution in Appendix E with different k's. The result in shown in Figure 5. When $k = 1$, the acceptance probability is equal to $1 - d_{\text{TV}}(p', q')$ for both solutions. The acceptance probability increases for both methods as k increases and there exists a gap between the optimal and approximate solution. We would expect the gap to exist for general conditional distributions from language models. We leave exploring computationally efficient ways to close this gap as an interesting future direction.

G. Detailed draft selection algorithm and its performance

The detailed algorithm is presented in Algorithm 4. We assume the condition probabilities on the next token is available given any prefix in the candidate set since they are computed parallelly in the second phase, and won't list them as inputs explicitly in Algorithm 4. The algorithm proceeds in a recursive fashion. Given prompt x^t and a candidate set S sampled from $\mathcal{M}_s(\cdot \mid x^t)$, the algorithm first computes the optimal transport plan $f_\pi : \Omega^{|S|} \to \Omega$ from $\mathcal{M}_s(\cdot \mid x^t)^{\otimes|S|}$ to $\mathcal{M}_b(\cdot \mid x^t)$. Then f_π is applied to the first token in each sequence in S to obtained a valid token Y from $\mathcal{M}_b(\cdot \mid x^t)$. If Y is not the last token ($L \geq 2$), we filter out sequences in S whose first token is not Y and denote the remaining sequences as S_{next} and feed it to the algorithm with context (x^t, Y) and draft length $L-1$. This goes on until we have $L = 1$ or $S_{\text{next}} = \emptyset$.

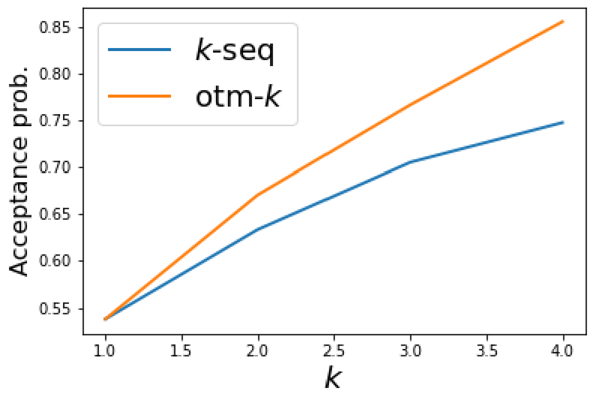


Figure 5. Acceptance probability comparison OTM-k and K-SEQ with compressed conditional distributions.

Algorithm 4 Draft selection with multiple candidates (DraftSelection).

```
Input: Input sequence $x^t$; candidate length: $L$; a set of candidates $S = \{z_i^L \mid i = 1, \ldots, |S|\}$ with length $L$.
 1: Compute a transport plan (using linear programming in Lemma 5.1 for an exact solution or Algorithm 3 for an
    approximate solution) from $\mathcal{M}_s(\cdot \mid x^t)^{\otimes|S|}$ to $\mathcal{M}_b(\cdot \mid x^t)$, denoted by $\pi_t$.
 2: Get the multi-set of next token-level drafts: $S_z = \{z_i(1)\}_{i\in[|S|]}$ and compute $Y = f_{\pi_t}(S_z)$.
 3: if $L = 1$ then
 4:    if $Y \in S_z$ then
 5:       Sample $Y' \sim \mathcal{M}_b(\cdot \mid (x^t, Y))$
 6:       Return $(x^t, Y, Y')$.
 7:    else
 8:       Return $(x^t, Y)$
 9:    end if
10: end if
11: Let $S_{\text{next}} = \{z^{2:L} \mid z \in S \text{ and } z(1) = Y\}$ be the set that consists of sub-sequences of the candidates that agree with
    the selected next token.
12: if $S_{\text{next}} = \emptyset$ then
13:    Return $(x^t, Y)$
14: else
15:    Return DraftSelection$((x^t, Y), L-1, S_{\text{next}})$
16: end if
```

In this case when Y is the last token (*i.e.,* $L = 1$) and $Y \in S$, we have the choice to sample an additional token $\mathcal{M}_b(\cdot \mid (x^t, Y))$ since this conditional probability is already computed in the second phase. Due to the property of the transport plan, we know that Y is always a valid sample from $\mathcal{M}_b(\cdot \mid x^t)$. The overall performance of the algorithm is stated in Theorem G.1. We needed to take care in the statement and the proof to deal with the fact that the length τ of the output sequence Y^τ is itself a random variable.

Theorem G.1. *Assume all drafts in set S are generated from the small model with input x^t, or more precisely, $\forall z \in S$,*

$$\forall i \in [1, L], \qquad z(i) \sim \mathcal{M}_b(\cdot \mid x^t, z^{i-1}). \tag{9}$$

Let Y^τ be the output of Algorithm 4 where τ is the length of the output, and $Z^{\tau+1:L} = (Z(\tau+1), \ldots, Z(L)) \sim \mathcal{M}_b(\underbrace{\cdot, \cdot, \ldots \cdot}_{(L-\tau)\text{ dots}} \mid x^t, Y^\tau)$, then it satisfies that $(Y^\tau, Z^{\tau+1:L}) \sim_{\text{prob}} \mathcal{M}_b(\underbrace{\cdot, \cdot, \ldots \cdot}_{L\text{ dots}} \mid x^t)$. More precisely, For any length-L sequence $o^L = (o(1), \ldots, o(L)) \in \Omega^L$, we have

$$\Pr\left((Y^\tau, Z^{\tau+1:L}) = o^L\right) = \Pi_{i=1}^L \mathcal{M}_b(o(i) \mid x^t, o^{i-1}).$$

G.1. Proof of Theorem G.1

We prove the theorem via induction. When $L = 1$, $\Pr(\tau = 1) = 1$, the theorem follows directly since f_π in Algorithm 4 is a valid transport plan. Suppose the theorem holds for $L = \ell \geq 1$, we next prove that it holds for $L = \ell + 1$. Let $\bar{Y}^{\tau'}$ be the output sequence when $L = \ell + 1$ and $\bar{Z}^{\tau'+1:\ell+1}$ be the subsequent samples from $\mathcal{M}_b$. Note that compared to the case when $L = \ell$, extending the block length of the tree by one only changes the probability of Y^τ when $\tau = L$, *i.e.*, $\forall j < \ell$ and length-j sequence $o^j \in \Omega^j$, we have

$$\Pr\left(Y^j = o^j, \tau = j\right) = \Pr\left(\bar{Y}^j = o^j, \tau' = j\right)$$

For any length-ℓ sequence o^ℓ, let

$$\delta(o^\ell) := \Pr\left(Y^\ell = o^\ell, \tau = \ell\right) - \Pr\left(\bar{Y}^\ell = o^\ell, \tau' = \ell\right).$$

Then by definition, we have

$$\delta(o^\ell) = \sum_{o(\ell+1) \in \Omega} \Pr\left(Y^{\ell+1}) = (o^\ell, o(\ell+1)), \tau = \ell+1\right)$$

For any length-$(\ell+1)$ sequence $o^{\ell+1} \in \Omega^{\ell+1}$, we have

$$\begin{aligned}
&\Pr\left((\bar{Y}^j, \bar{Z}^{\tau'+1:\ell+1}) = o^{\ell+1}\right) && (10)\\
=&\sum_{j=1}^{\ell-1} \Pr\left(\bar{Y}^j = o^j, \tau' = j\right)\mathcal{M}_b(o^{j+1:\ell+1} \mid x^n, o^j)\\
&\quad + \Pr\left(\bar{Y}^\ell = o^\ell, \tau' = \ell\right)\mathcal{M}_b(o(\ell+1) \mid x^n, o^\ell) + \Pr\left(\bar{Y}^{\ell+1} = o^{\ell+1}, \tau = \ell+1\right) && (11)\\
=&\sum_{j=1}^{\ell-1} \Pr\left(Y^j = o^j, \tau' = j\right)\mathcal{M}_b(o^{j+1:\ell+1} \mid x^n, o^j)\\
&\quad + \left(\Pr\left(Y^\ell = o^\ell, \tau' = \ell\right) - \delta(o^\ell)\right)\mathcal{M}_b(o(\ell+1) \mid x^n, o^\ell) + \Pr\left(\bar{Y}^{\ell+1} = o^{\ell+1}, \tau = \ell+1\right) && (12)\\
=&\mathcal{M}_b(o^{\ell+1} \mid x^n) - \delta(o^\ell)\mathcal{M}_b(o(\ell+1) \mid x^n, o^\ell) + \Pr\left(\bar{Y}^{\ell+1} = o^{\ell+1}, \tau = \ell+1\right). && (13)
\end{aligned}$$

Hence it would enough to show that

$$\delta(o^\ell)\mathcal{M}_b(o(\ell+1) \mid x^n, o^\ell) = \Pr\left(\bar{Y}^{\ell+1} = o^{\ell+1}, \tau = \ell+1\right) \tag{14}$$

Note that the event $\bar{Y}^{\ell+1} = o^{\ell+1}, \tau' = \ell+1$ only happens when o^ℓ are all accepted samples from $\mathcal{M}_s$ in the sampling process and when proceeding, the next obtained token is $o(\ell+1)$.

On the other hand, the $\delta(o^\ell)$ is the probability of the event that the sampling process stops at o^ℓ when $L = \ell$ and proceeds when $L = \ell+1$, which, by definition of the algorithm, happens if and only if o^ℓ are all accepted samples from $\mathcal{M}_s$. Moreover, when proceeding, since f_π is a valid transport plan, we have that the next sample is generated from $\mathcal{M}_b(\cdot \mid x^t, o^\ell)$. And hence Equation (14) holds.

This concludes the proof.

H. Construct a candidate set by sampling from a prefix-tree

As discussed in Section 1, the size of the draft set S is constrained by the number of parallel computations that can be supported in the hardware. Hence it is important to design the draft set carefully to allow for a longer sequence of accepted candidate sets. In addition to the *i.i.d.* draft set selection approach listed in Section 6, we present an algorithm that samples a draft set that forms the leaves of a prefix tree. Given a draft set size K, the algorithm can be specified by a sequence of parameter $(k_1, k_2, \ldots, k_L)$ satisfying $\prod_{i=1}^{L} k_i = K$.

At a high-level, the algorithm starts with a root node with sequence $x^{1:t}$ and forms a prefix tree of depth L. At depth $i \in [1 : L-1]$, each node is expanded by a factor of k_{i+1} and each of its children will contain a sequence that satisfies: (1) Its prefix agrees with the sequence in the parent node; (2) The next token is sampled from the conditional probability given the prefix in small model. These child nodes will be at depth $i+1$ and the process goes until it hits depth L. We give a detailed description of the algorithm in Algorithm 5.

Algorithm 5 Draft set selection via prefix-tree.

```
Input: Input sequence $x^t$; expansion factors at each level: $(k_1, k_2, \dots, k_L)$.
1: $S_0 = \{x^t\}$.
2: for $i = 0, 1, 2, \dots, L-1$ do
3:     $S_{i+1} = \emptyset$.
4:     for all seq $\in S_i$ do
5:         Sample $k_{i+1}$ i.i.d. tokens $X_1, X_2, \dots, X_{k_{i+1}}$ from $\mathcal{M}_b(\cdot \mid \text{seq})$.
6:         $S_{i+1} = S_{i+1} \cup \{(\text{seq}, X_i), i = 1, 2, \dots k_{i+1}\}$.
7:     end for
8: end for
9: Return $S_L$.
```

I. Additional experiments

We first provide a verification of the computational model introduced in Section 1 by reporting the latencies of using the large model to compute the probabilistic distributions with parallelization over time and batch axes. As shown in Table 2, the latency stays roughly constant in these setting.

Table 2. Average latency with parallelization along the time axis and batch axis. We report average latency with standard deviation over 1,000 runs using a $97M$ transformer relative to length = 4 and batch = 1 on GPU.

Relative latency	batch = 1	batch = 2	batch = 4	batch = 8
length = 4	1.00 ± 0.16	1.01 ± 0.15	1.06 ± 0.10	1.10 ± 0.16
length = 8	1.01 ± 0.18	1.09 ± 0.25	1.10 ± 0.09	1.42 ± 0.4

Similar to Table 2, we report relative latency when parallelizing across the time and batch axes using the small $6M$ draft model in Table 3. In Table 3, the reported relative latencies are relative to the large $97M$ model to get a sense of the relative cost of sampling multiple drafts with the small model compared to the large model. We also include results for varying draft model sizes with the same $97M$ large model for LM1B in Table 4. These additional draft models were produced by either halving ($2M$) or doubling ($20M$) the original $6M$ draft model's number of layers, embedding dimension, MLP dimension, and number of attention heads. As expected, the larger draft models improve all speculative methods' number of decoded tokens per large model serial call with SpecTr maintaining the best performance across all draft model sizes.

Table 3. Average latency with parallelization along the time axis and batch axis. We report average latency with standard deviation over 1,000 runs using a $6M$ transformer relative to the $97M$ transformer with length = 4 and batch = 1 on GPU.

Relative latency	batch = 1	batch = 2	batch = 4	batch = 8
length = 4	0.18 ± 0.02	0.19 ± 0.04	0.18 ± 0.09	0.20 ± 0.13
length = 8	0.17 ± 0.04	0.19 ± 0.05	0.16 ± 0.02	0.18 ± 0.04

Table 4. Experimental results on the LM1B dataset with varying draft model sizes and the $97M$ transformer as the large model. All results are over 1000 test prompts averaged over three different random seeds and sampling temperature of 1.0 for both the draft and large models.

Draft model	Algorithm	K	L	Number of decoded tokens per serial call
$2M$ Transformer	Baseline	-	-	1.00
	Speculative	1	4	1.86 ± 0.02
	SpecTr	2	4	2.07 ± 0.01
	SpecTr	4	4	2.32 ± 0.00
	SpecTr	8	4	$\mathbf{2.56 \pm 0.01}$
	Speculative	1	8	1.91 ± 0.01
	SpecTr	2	8	2.15 ± 0.01
	SpecTr	4	8	2.41 ± 0.00
	SpecTr	8	8	$\mathbf{2.68 \pm 0.01}$
$6M$ Transformer	Baseline	-	-	1.00
	Speculative	1	4	2.21 ± 0.01
	SpecTr	2	4	2.43 ± 0.01
	SpecTr	4	4	2.74 ± 0.01
	SpecTr	8	4	$\mathbf{2.99 \pm 0.02}$
	Speculative	1	8	2.33 ± 0.01
	SpecTr	2	8	2.61 ± 0.02
	SpecTr	4	8	2.96 ± 0.03
	SpecTr	8	8	$\mathbf{3.27 \pm 0.02}$
$20M$ Transformer	Baseline	-	-	1.00
	Speculative	1	4	2.71 ± 0.01
	SpecTr	2	4	2.96 ± 0.00
	SpecTr	4	4	3.28 ± 0.02
	SpecTr	8	4	$\mathbf{3.49 \pm 0.03}$
	Speculative	1	8	3.12 ± 0.02
	SpecTr	2	8	3.48 ± 0.04
	SpecTr	4	8	3.85 ± 0.05
	SpecTr	8	8	$\mathbf{4.15 \pm 0.04}$