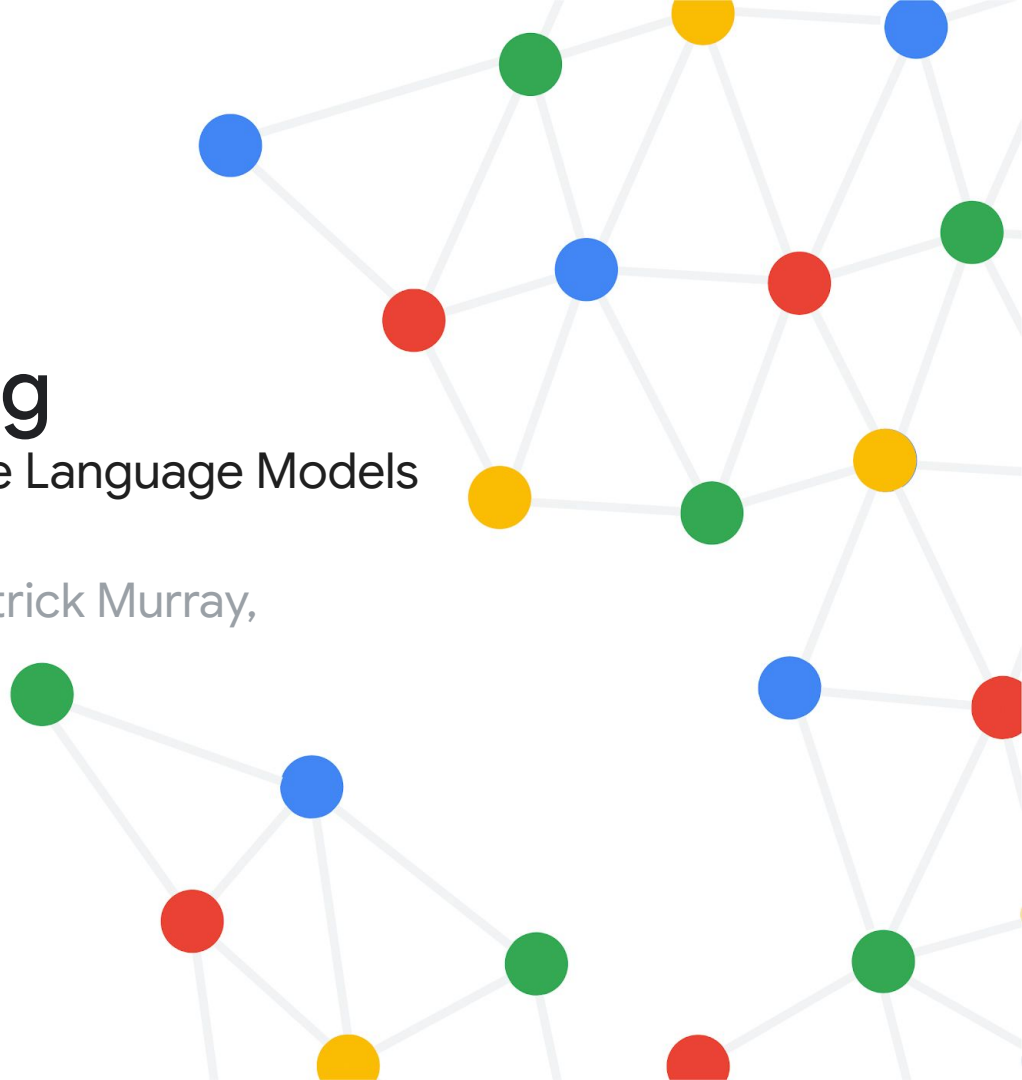
Arithmetic Sampling

Parallel Diverse Decoding for Large Language Models

Luke Vilnis*, Yury Zemlyanskiy*, Patrick Murray,
Alexandre Passos, Sumit Sanghai



Google Research

Motivation

- Sampling from large language models is computationally expensive, and samples are often **very similar** to one another.
- We want to **make each sample count**!
- Methods to **avoid duplication** or **promote diversity**, such as
  - Beam search
  - Gumbel-top-k [Kool et al. 2019]
  - UniqueRandomizer [Shi et al. 2020]
  - Determinantal beam search [Meister et al. 2021]
  - ...

  are **difficult to parallelize**.

What might we want from a sampling algorithm?

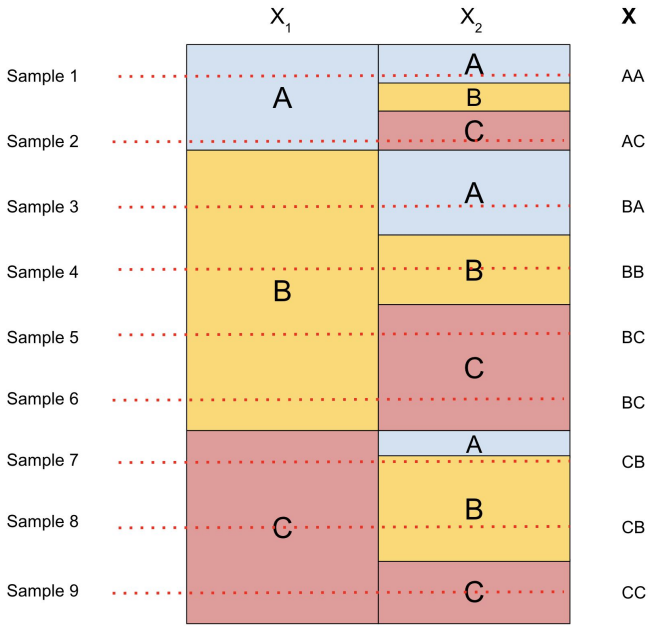
Diverse Faithful Efficient Parallel

- **Desiderata**
  - **Diverse** samples**:** sampled sentences should be diverse by some metric.
  - **Faithful** to the underlying probabilistic model**:** if a sequence gets high probability under the language model, we should see it.
  - Computationally **efficient:** no more complex than normal decoding
  - **Parallel:** # of samples should not depend on platform HBM

Idea: arithmetic sampling

Diverse Faithful Efficient Parallel

- **Arithmetic sampling**
  - Reinterpret ancestral sampling as lazily constructing an **arithmetic codebook**
  - Pick codes that are **spread out in code space**
  - Replace IID Monte Carlo samples w/ **dependent samples**



$$\text{Each prefix } (X_1 = v_{i_1}, ..., X_L = v_{i_L})$$
$$\text{falls in subinterval } (w_{i_1 ... i_L}, w_{i_1 ... i_L + 1})$$
$$w_{i_1 ... i_L} = w_{i_1 .. i_{L-1}} + \sum_{j < i_L} P(X_1 = v_{i_1}, ..., X_{L-1} = v_{i_{L-1}}, X_L = v_j)$$

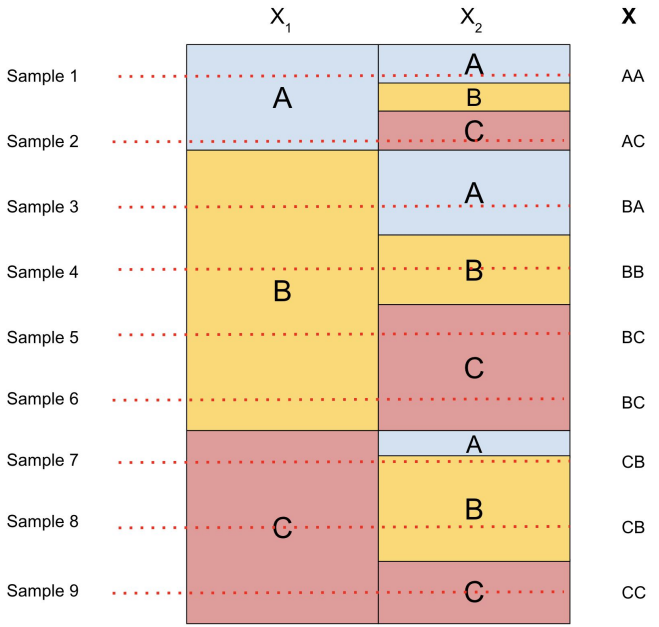
Idea: arithmetic sampling

Diverse Faithful **Efficient** Parallel

- **Efficient sampling**
  - Samples are generated by an algorithm very similar to standard ancestral sampling

Algorithm 1 Sampling from a Code Point

Input: code point $c \in [0, 1]$, ordered vocabulary V, sequence model $P(X_1, ..., X_T)$
set $c_0 = c$, $X = \{\}$, $t = 0$
repeat
  set $w_i = \sum_{j<i} P(X_t = v_j | X_{<t})$
    for $v_j \in V$
  set $X_t = v_i$
    s.t. $w_i < c < w_{i+1}$
  set $(m, M) = (w_i, w_{i+1})$
    s.t. $w_i < c < w_{i+1}$
  set $c_{t+1} = \frac{c_t - m}{M - m}$
  set $t = t + 1$
until $X_{t-1} = \text{EOS}$
return X

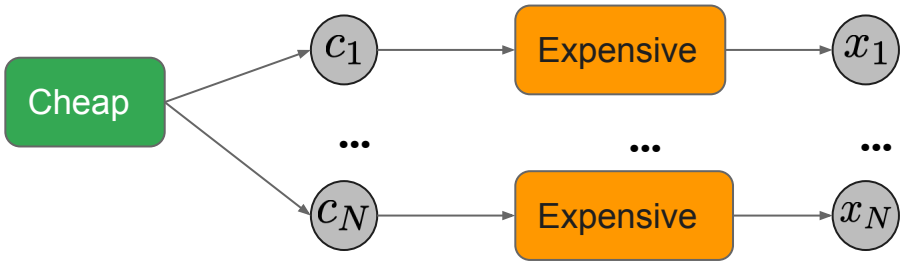


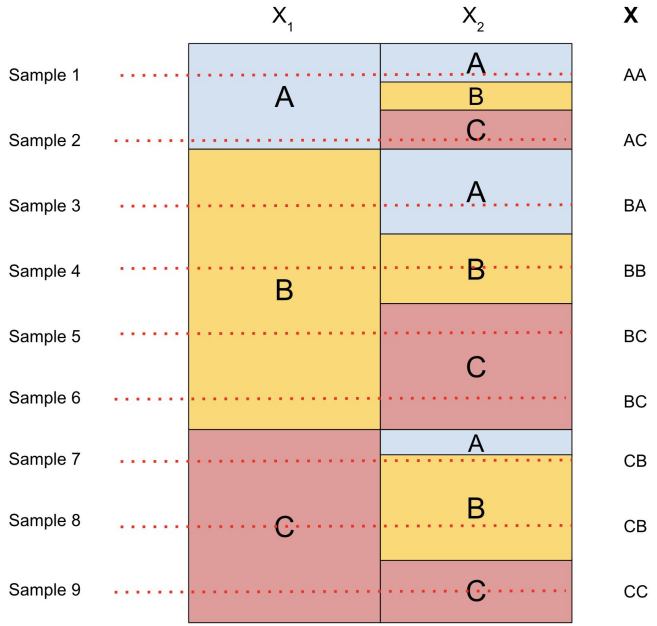
Idea: arithmetic sampling

Diverse Faithful **Efficient** **Parallel**

- **Parallel sampling**
  - LLM sampling is embarrassingly parallel conditioned on a set of 1-d codes sampled evenly from [0,1]

$$\{x_i\} = f(\{c_i\}) \text{ s.t. } c_i \text{ are cheap}$$

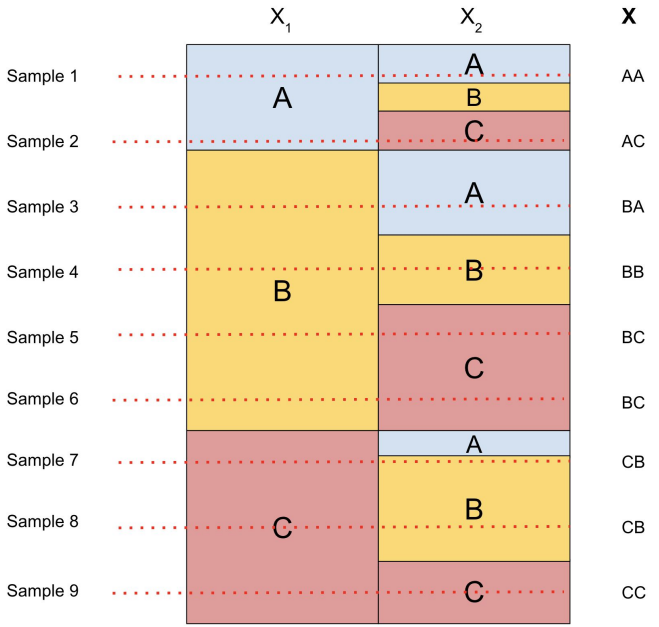




Idea: arithmetic sampling

Diverse Faithful **Efficient** **Parallel**

- **Unbiased & Consistent**
  - Because randomly shifted lattice points have marginal uniform distribution, estimators are **unbiased**
  - Because samples occur on a grid, estimators are **consistent**

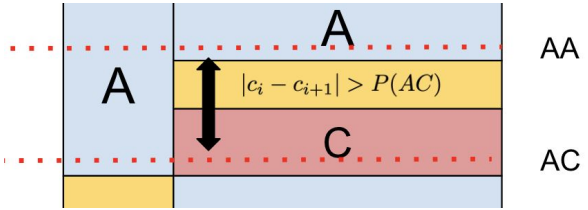


Idea: arithmetic sampling

Diverse **Faithful** **Efficient** **Parallel**

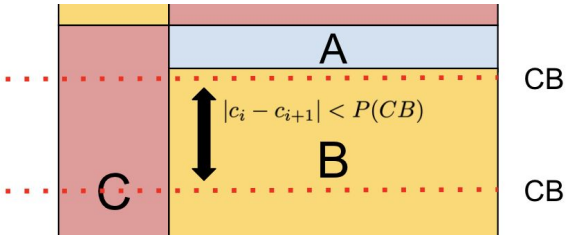
- Arithmetic sampling does not sample any prefix **"too much"**

Proposition 5. *Arithmetic sampling with size N will never sample the same prefix x more than n times if $P(X = x) < n/(N + 1)$.*



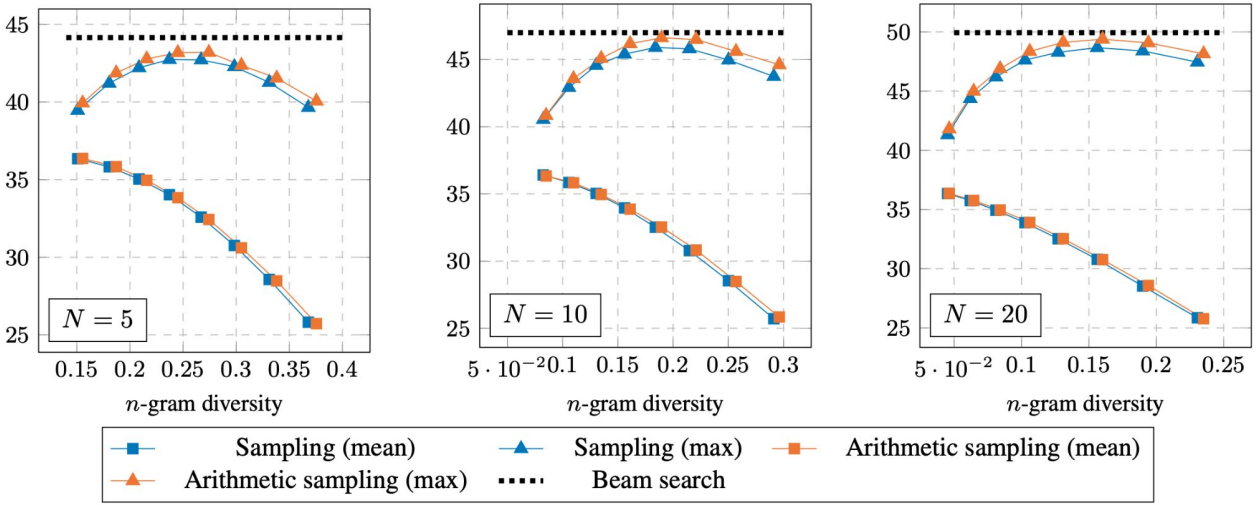
- ... or **"too little"**

Proposition 6. *Arithmetic sampling with size N must always sample a prefix x at least n times if $P(X = x) > (n + 1)/(N + 1)$.*



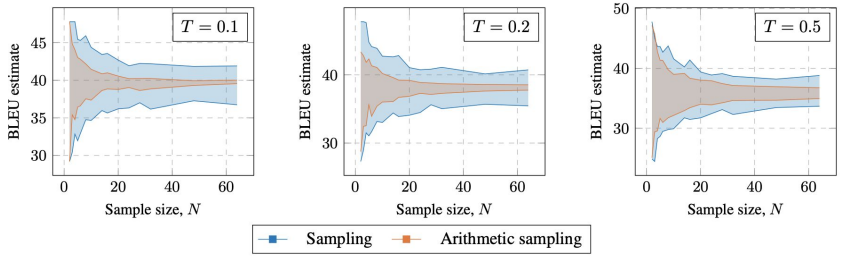
Experiments

- WMT EnFr reranking
  - BLEU vs. n-gram diversity
  - Oracle: max BLEU within beam, closes most of the gap with beam search

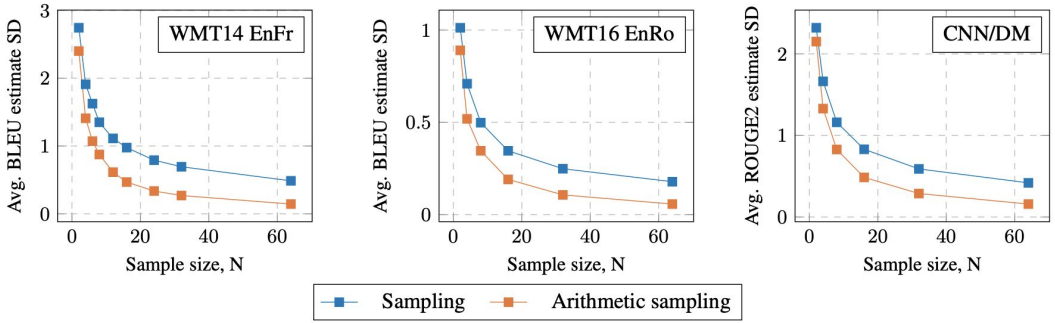


Experiments

- Variance reduction
  - Sentence BLEU variance (WMT EnFr)



  - Estimator variance (BLEU for WMT14 EnFr and WMT16 EnRo, ROUGE2 for CNN/DailyMail)



Conclusion

- Arithmetic sampling
  - Parallel, efficient, unbiased estimator
  - Diverse decoding
  - Implemented in T5X: github.com/google-research/google-research/tree/master/arithmetic_sampling

Conclusion

- Arithmetic sampling
  - Parallel, efficient, unbiased estimator
  - Diverse decoding
  - Implemented in T5X: github.com/google-research/google-research/tree/master/arithmetic_sampling

Thanks!