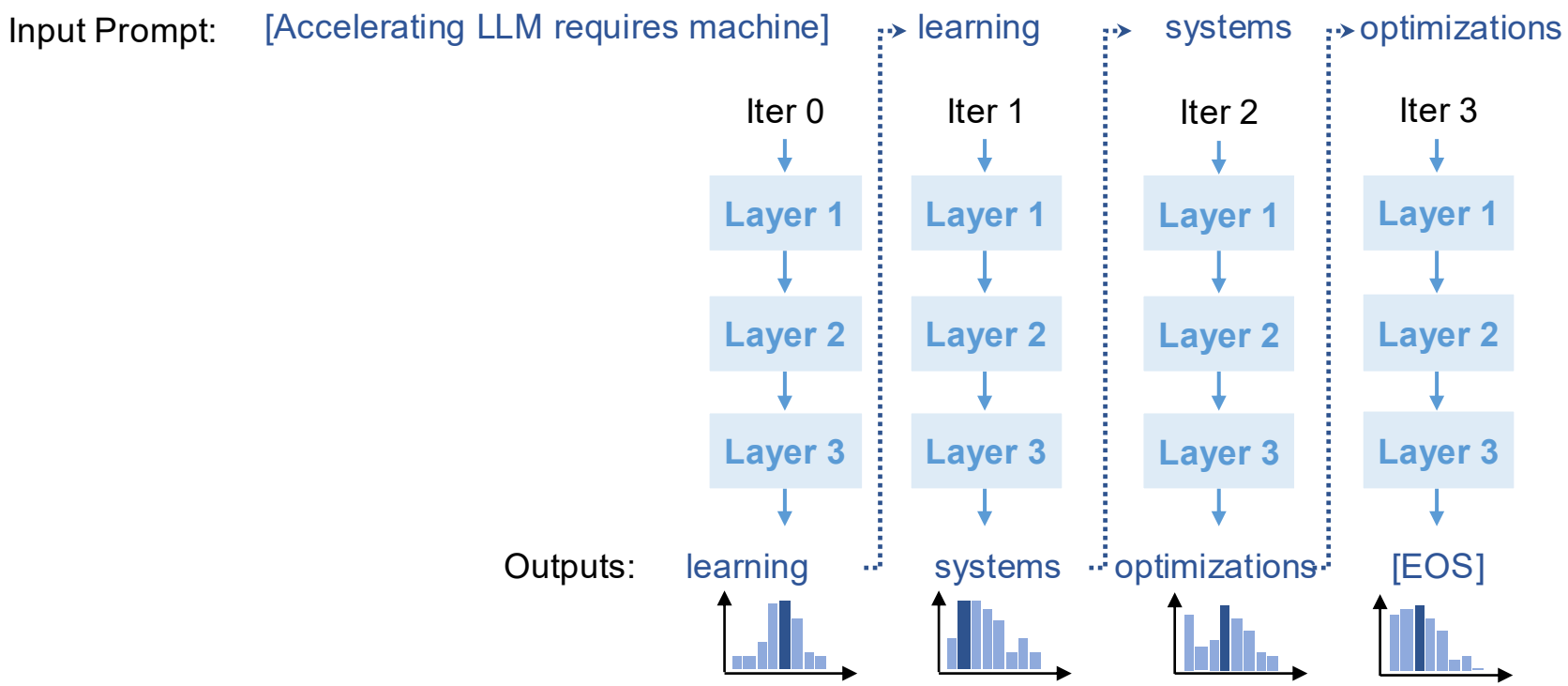
15-442/15-642: Machine Learning Systems

LLM Serving Techniques Part 2: Speculative Decoding

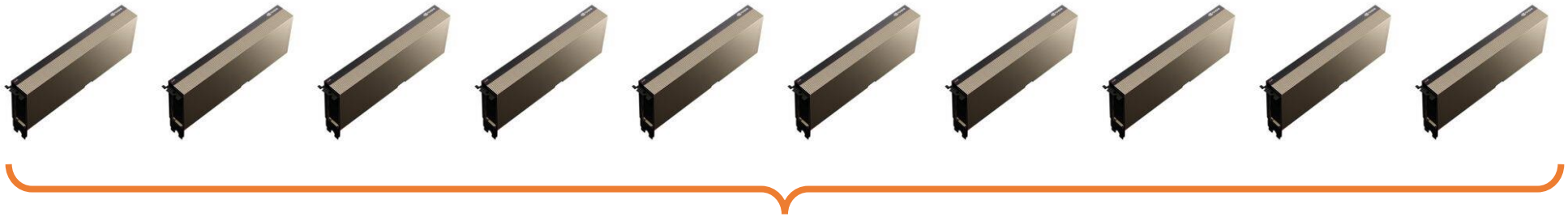
Tianqi Chen and Zhihao Jia

Carnegie Mellon University

Generative LLM Inference: Autoregressive Decoding



LLMs are Slow and Expensive to Serve



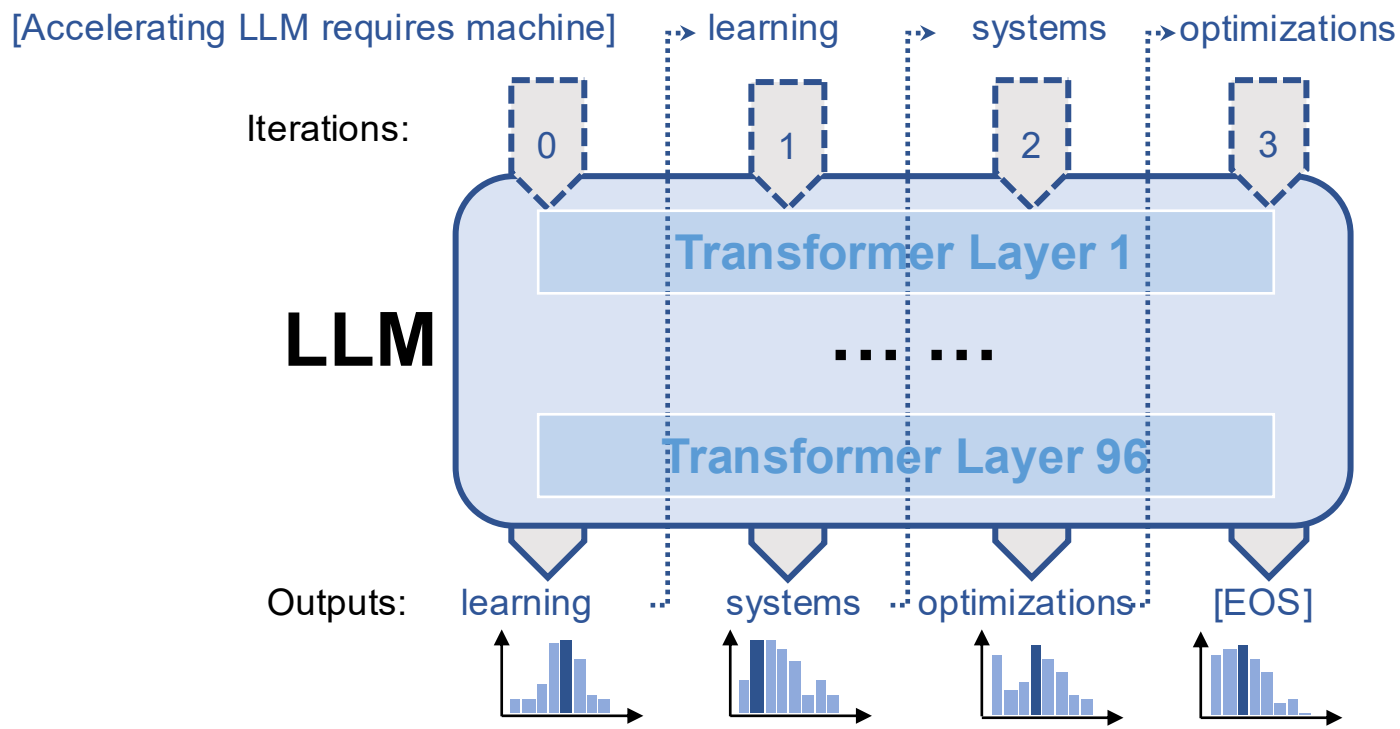
- **At least ten** A100-40GB GPUs to serve 175B GPT-3 in half precision
- Generating 256 tokens takes **~20 seconds**
- Cannot process many requests in parallel
  - Per-request key/value cache takes **3GB GPU memory**

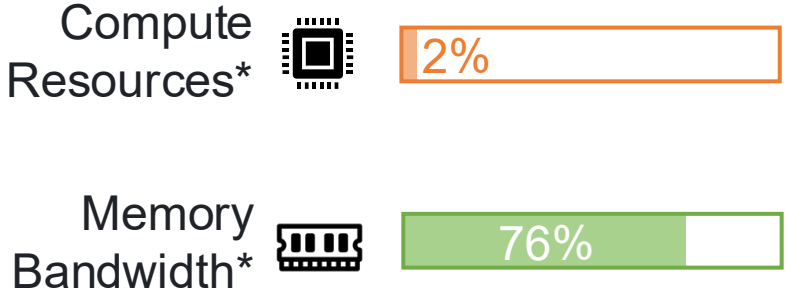
Today's Lecture: Speculative Decoding

- **Model-based speculative decoding**
  - Using a separate draft model to predict LLM's output

- Model-free speculative decoding
  - Using previously generated tokens to predict future tokens

Recall: Incremental Decoding Issues





- Limited degree of parallelism → underutilized GPU resources
- Need all parameters to decode a token → bottlenecked by GPU memory access

* Measured by serving LLAMA-2-70B on 4 A100 GPUs with 4K sequence length

Tradeoffs between Different Language Models

# Parameters	175B	13B	2.7B	760M	125M
TriviaQA	**71.2**	57.5	42.3	26.5	6.96
PIQA	**82.3**	79.9	75.4	72.0	64.3
SQuAD	**64.9**	62.6	50.0	39.2	27.5
latency	20 s	7.6s	2.7s	1.1s	**0.3s**
# A100s	10	1	1	1	**1**

Comparing multiple GPT-3 models*

Large models

Pro: better generative performance

Con: slow and expensive to serve

Small models

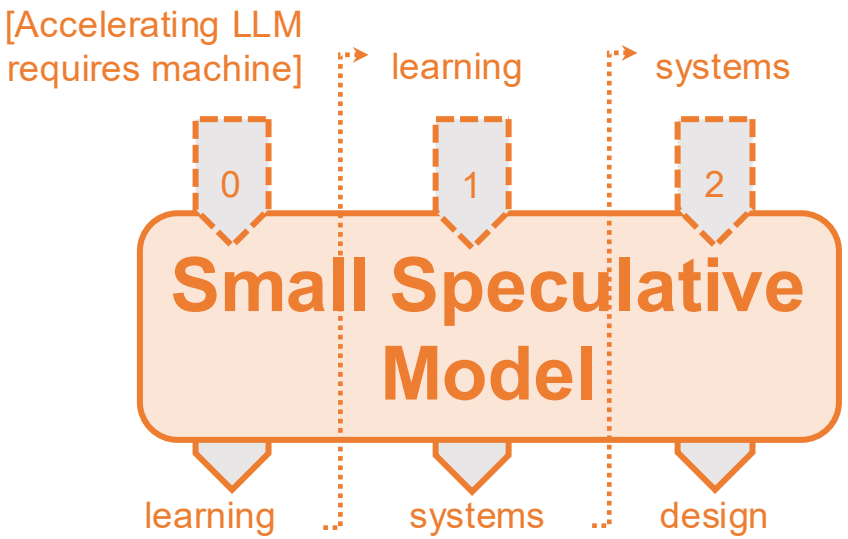
Pro: cheap and fast

Con: less accurate

* Language Models are Few-Shot Learners. Arxiv. 2005.14165

Speculative Decoding

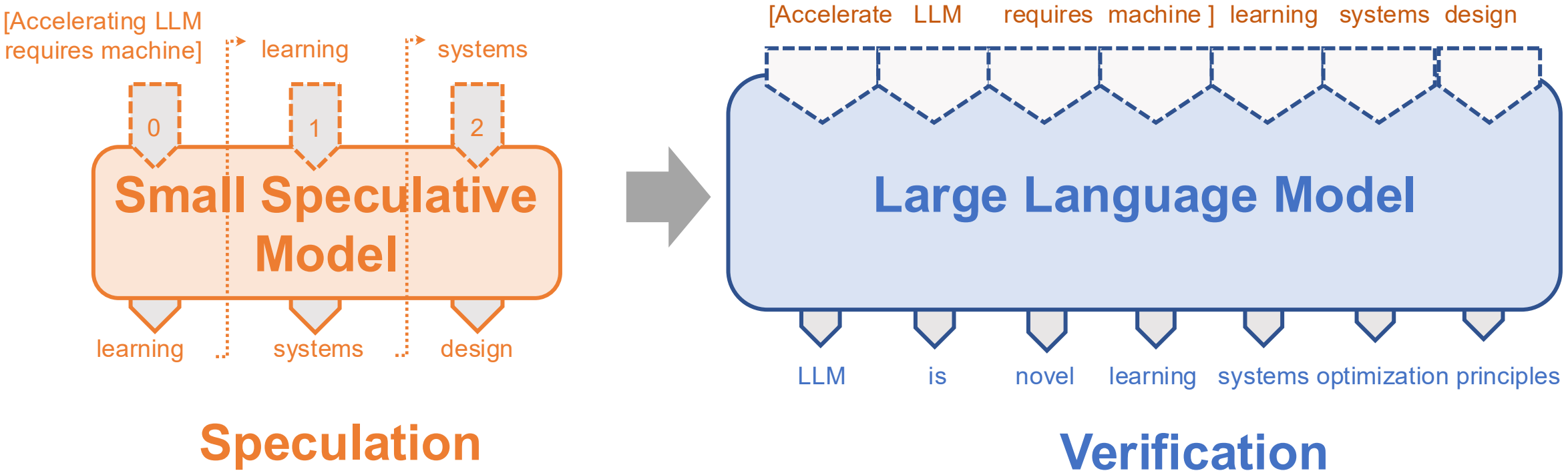
1. Use a small speculative model (SSM) to predict the LLM's output
   - SSM runs much faster than LLM



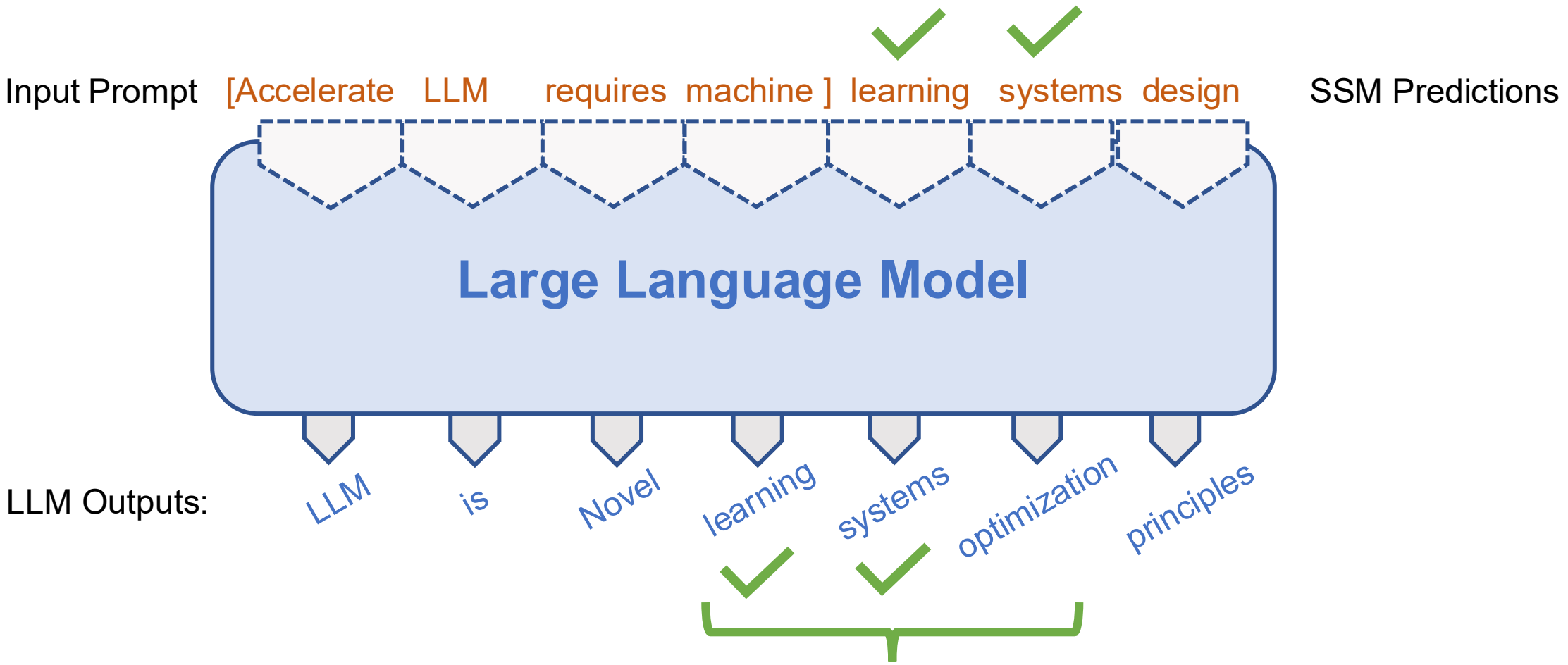
Speculation

Speculative Decoding

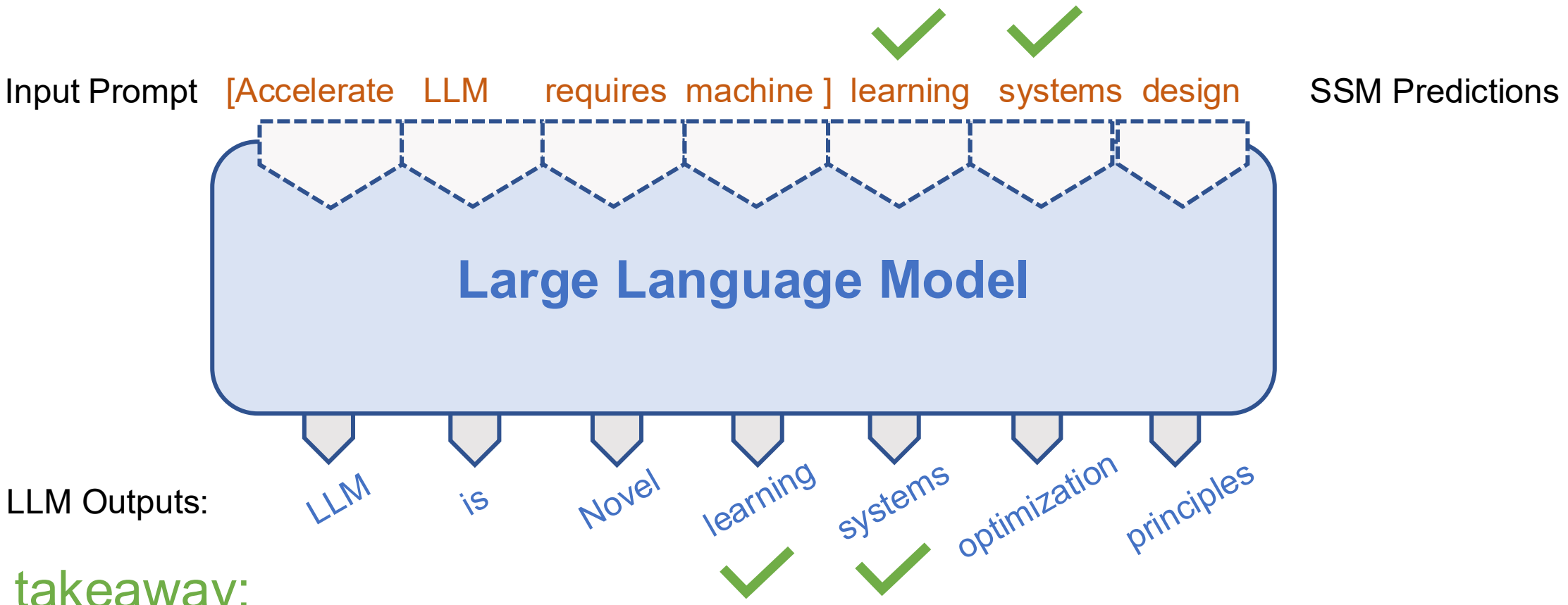
1. Use a small speculative model (SSM) to predict the LLM's output
   - SSM runs much faster than LLM
2. Use the LLM to verify the SSM's prediction



Verifying Speculative Decoding Results



Verifying Speculative Decoding Results



Key takeaway:

- LLM inference is bottlenecked by accessing model weights
- using LLM to decode multiple tokens to improve GPU utilization

SpecInfer: Tree-based Speculative Inference & Verification

Key idea: not use LLMs as incremental decoder, use them as **parallel token tree verifier**

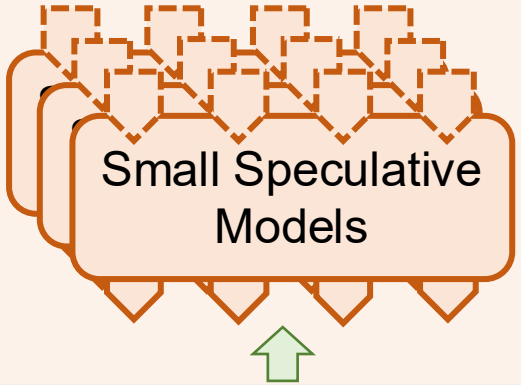
- **Better performance:** outperform existing LLM systems by **1.3-2.4x**
- **Higher efficiency:** reduce GPU memory access by **2.5-4.4x**
- **Correctness:** verification guarantees end-to-end equivalence

* SpecInfer: Accelerating LLM Serving with Tree-based Speculative Inference and Verification. ASPLOS'24

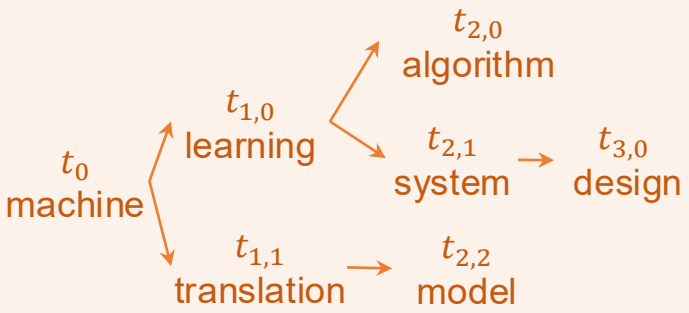
SpecInfer Workflow

Speculator

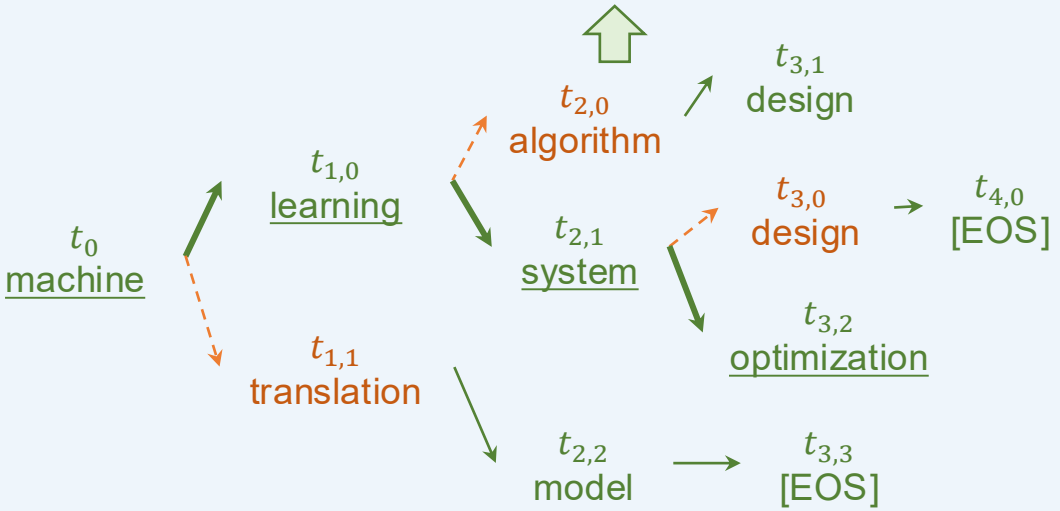
[Accelerating LLM requires machine]

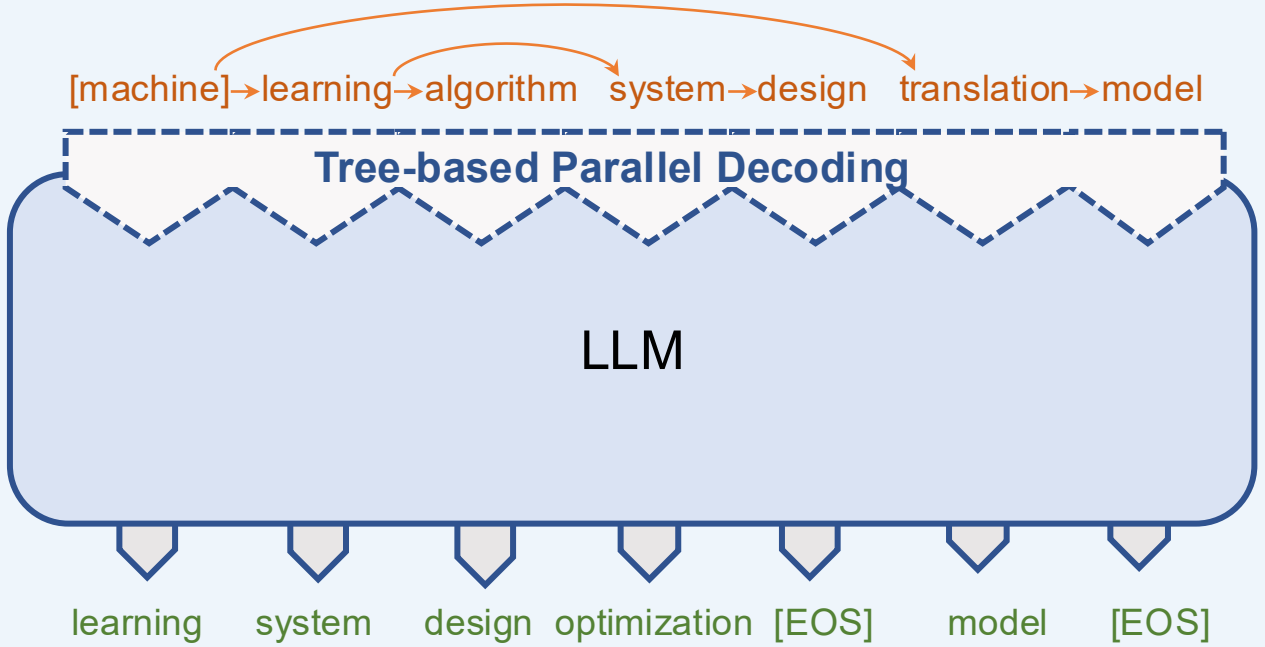


Output 0: [machine] learning algorithm
Output 1: [machine] learning system design
Output 2: [machine] translation model

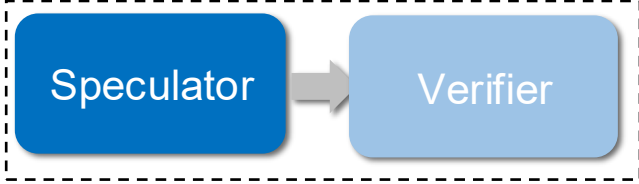


Verified output: machine learning system optimization





Verifier



Learning-based Speculator

[machine]

Prompt

- Small speculative models

Quantized LLM

Distilled LLM

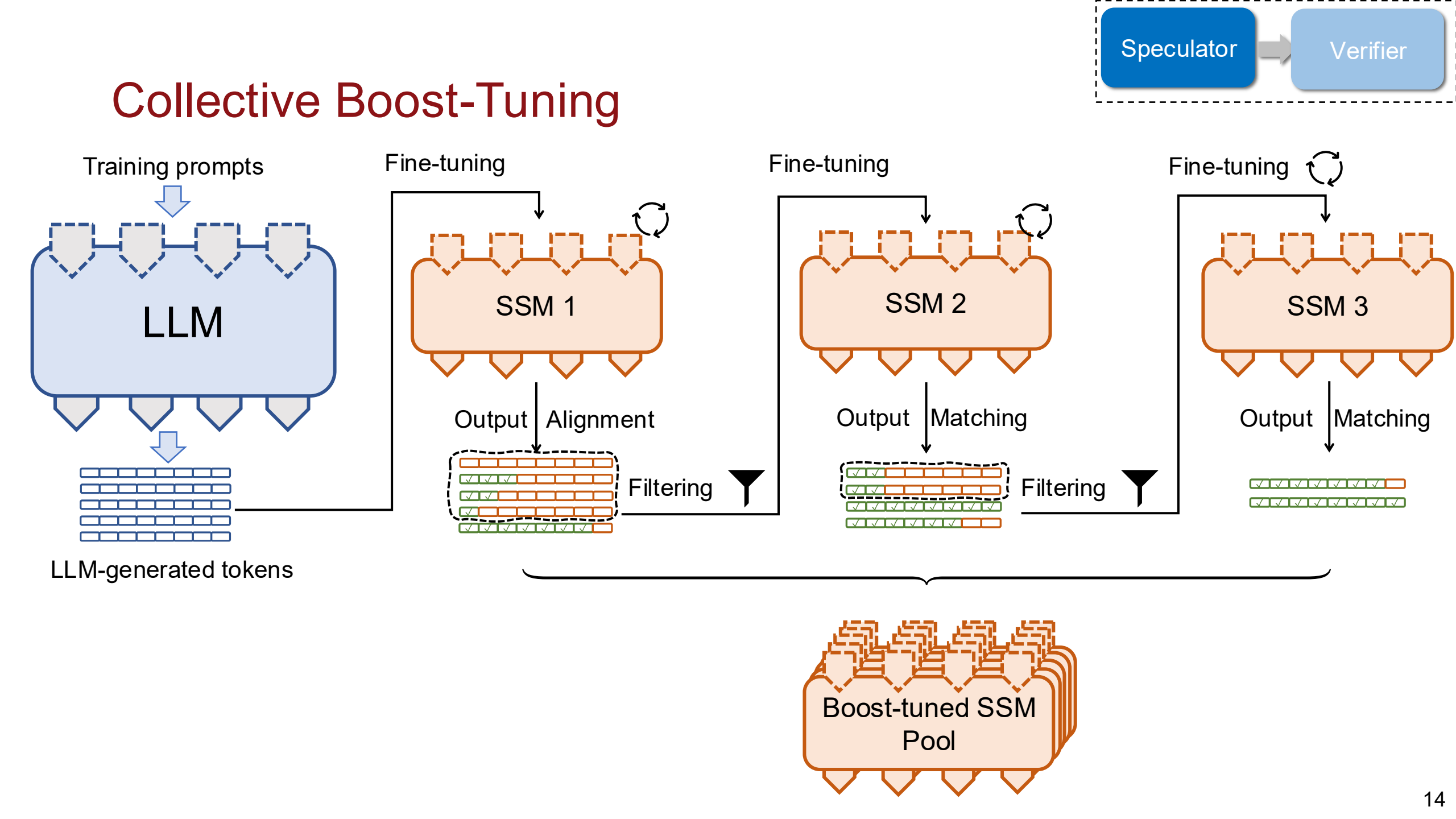
Fine-tuned SSM

SSM 0: [machine] → intelligence

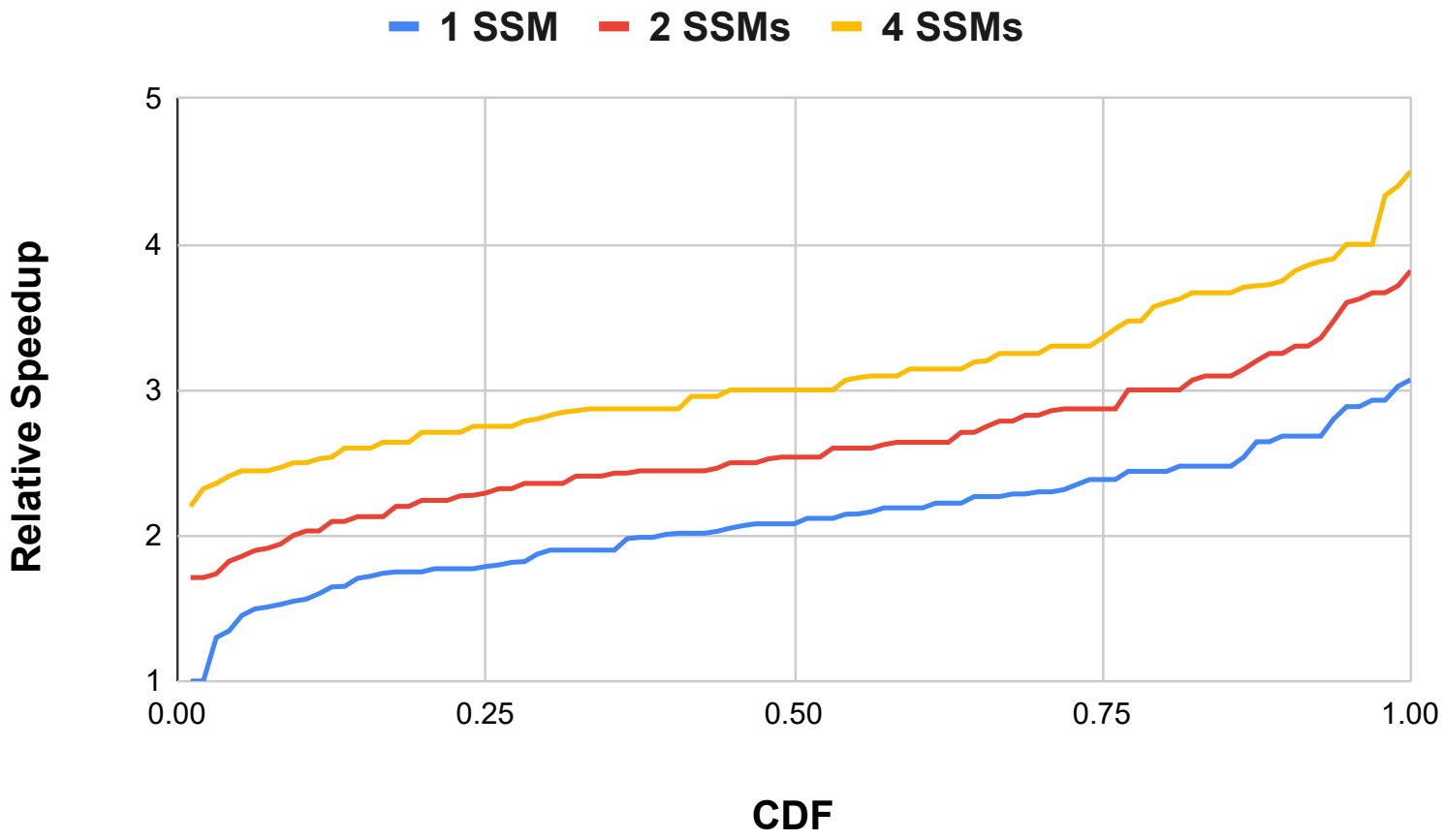
SSM 1: [machine] → translation → model

SSM 2: [machine] → learning → algorithm
learning → system → design

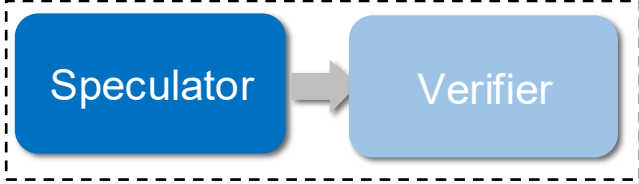
Speculated Tokens


Speculator
Verifier
Collective Boost-Tuning
Training prompts
LLM
LLM-generated tokens
Fine-tuning
SSM 1
Output
Alignment
Filtering
Fine-tuning
SSM 2
Output
Matching
Filtering
Fine-tuning
SSM 3
Output
Matching
Boost-tuned SSM Pool

Collective Boost-Tuning Consistently Improves Performance



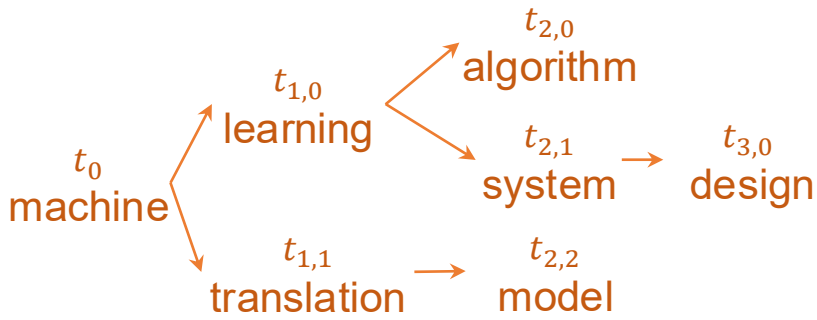
* LLM: LLAMA-7B, SSM: LLAMA-160M, dataset: Alpaca, GPU: NVIDIA A10



Token Tree Merge

- A compact way to represent speculated tokens

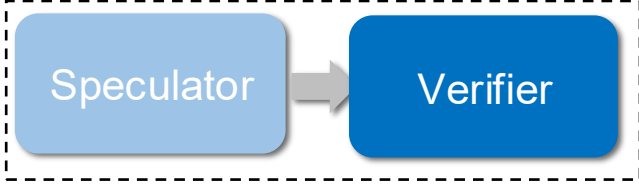
SSM 0: [machine] learning algorithm
SSM 1: [machine] learning system design
SSM 2: [machine] translation model

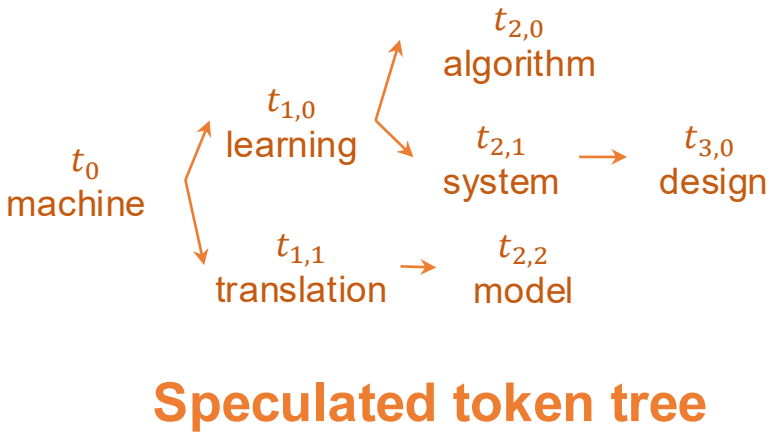


Token Sequences

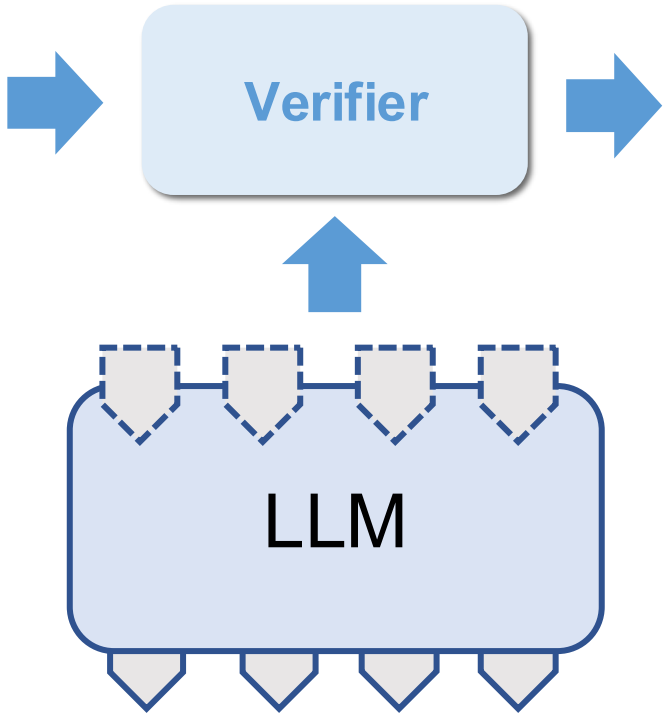
Token Tree

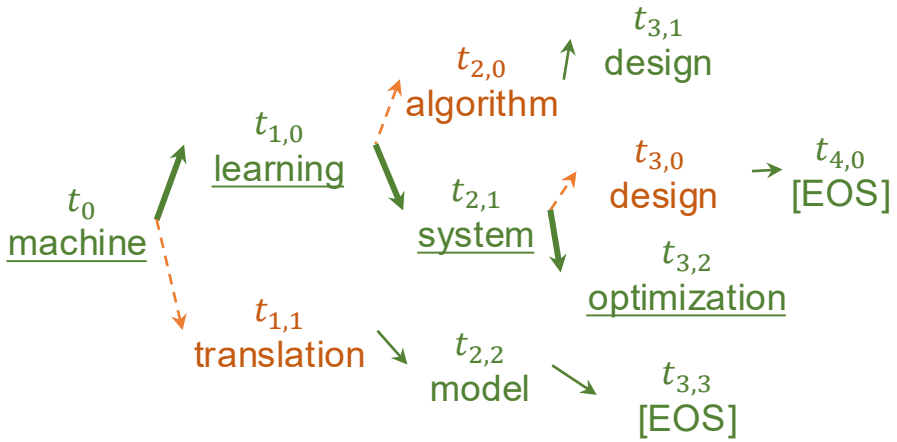
Token Tree Verifier



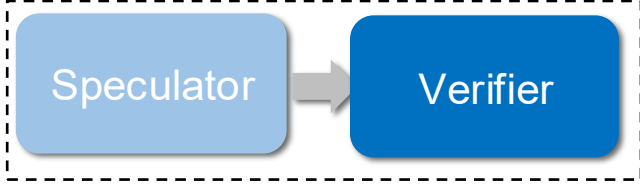


Speculated token tree

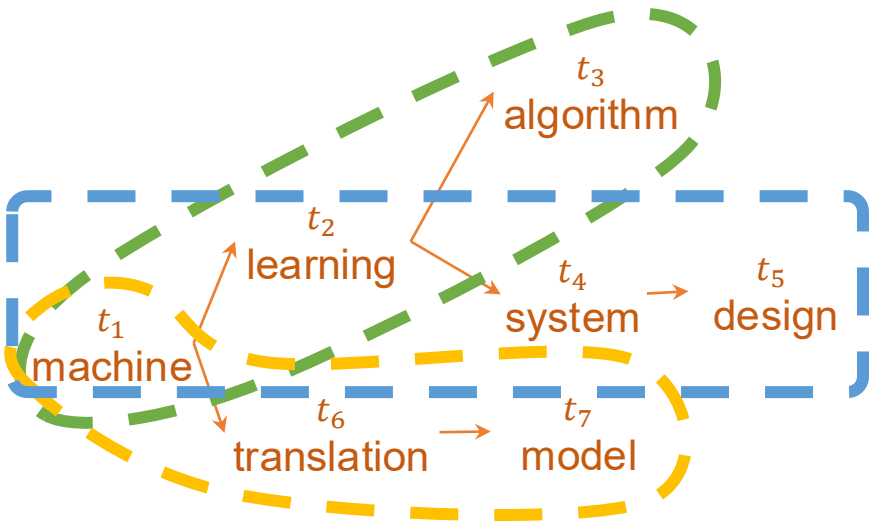


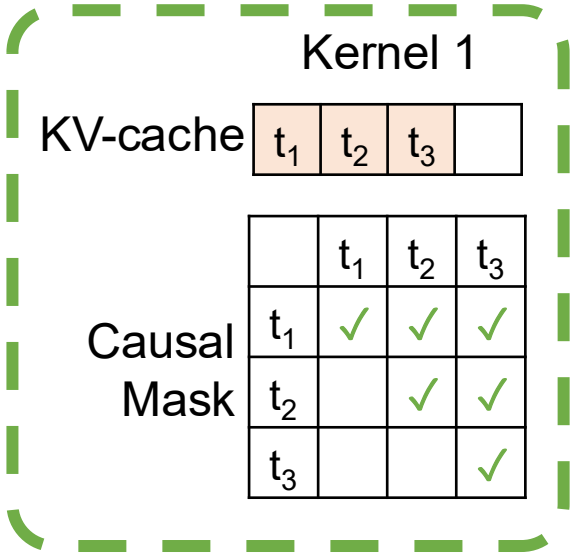


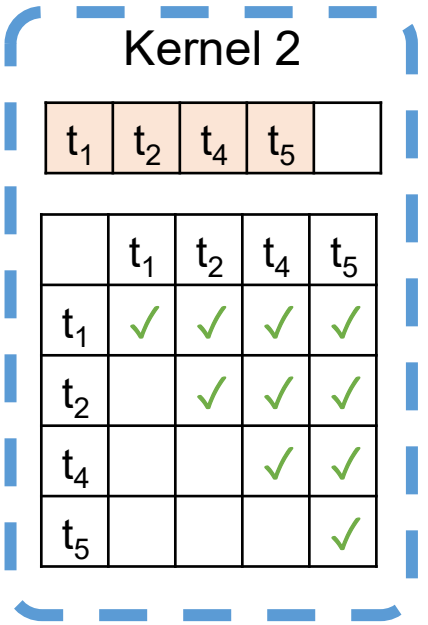
Verified token tree

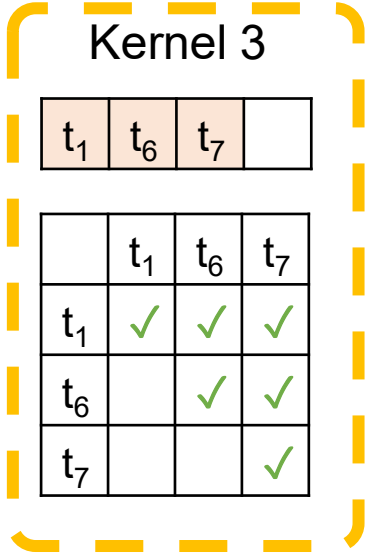


Sequence-based Decoding









Issues:

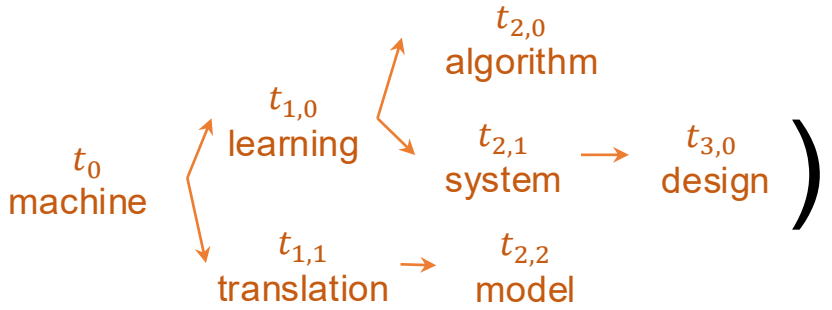
- Redundant decoding computation
- More requests → more GPU memory for key/value cache

Tree Attention

- same output as sequence attention for each token; no redundancy

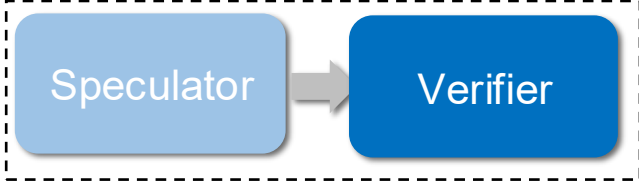
Token Sequences

Attention([machine] learning algorithm
[machine] learning system design
[machine] translation model)

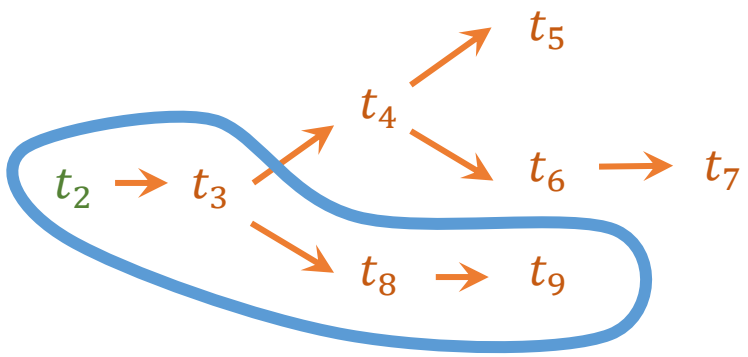
$=$ Tree-Attention(

)

Token Tree



Tree-based Parallel Decoding



KV-cache

t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	…

Topology-aware causal mask

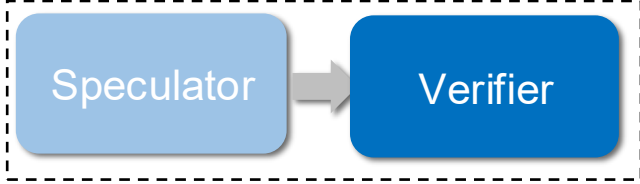
	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
t_2	✓	✓	✓	✓	✓	✓	✓	✓
t_3		✓	✓	✓	✓	✓	✓	✓
t_4			✓	✓	✓	✓		
t_5				✓				
t_6					✓	✓		
t_7						✓		
t_8							✓	✓
t_9								✓

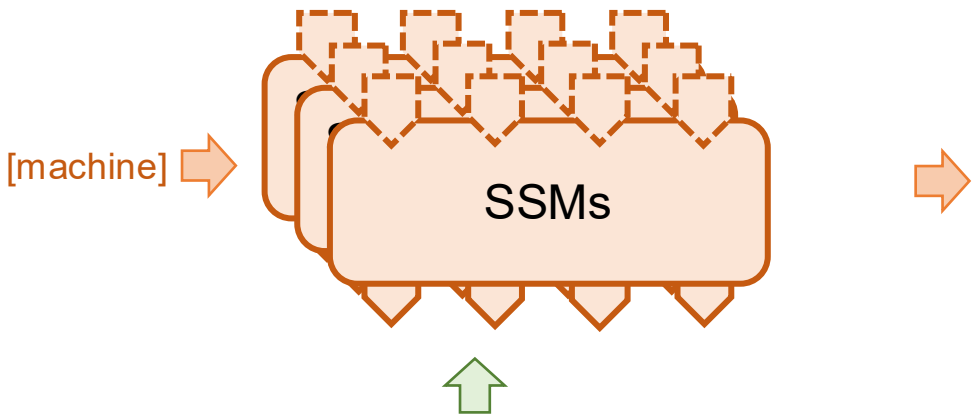
Key optimizations:

- A DFS-based approach to linearizing a token tree
- Tree topology-aware causal mask
- Decoding all tokens in a single GPU kernel

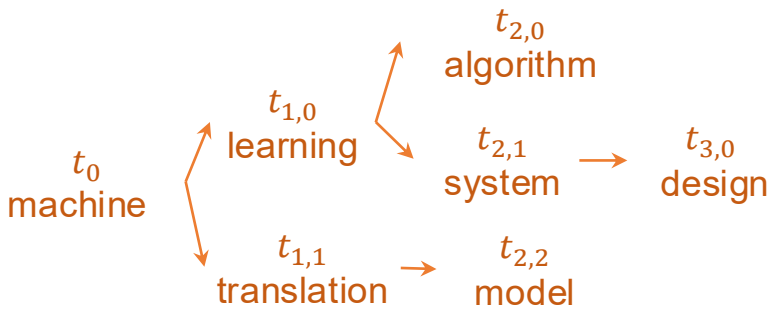
* SpecInfer: Accelerating LLM Serving with Tree-based Speculative Inference and Verification. ASPLOS'24

Verification Workflow

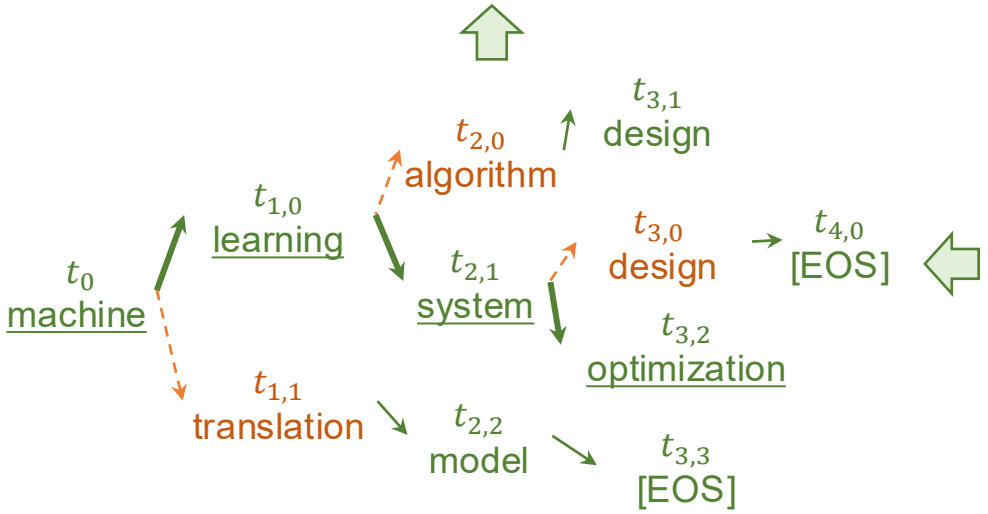




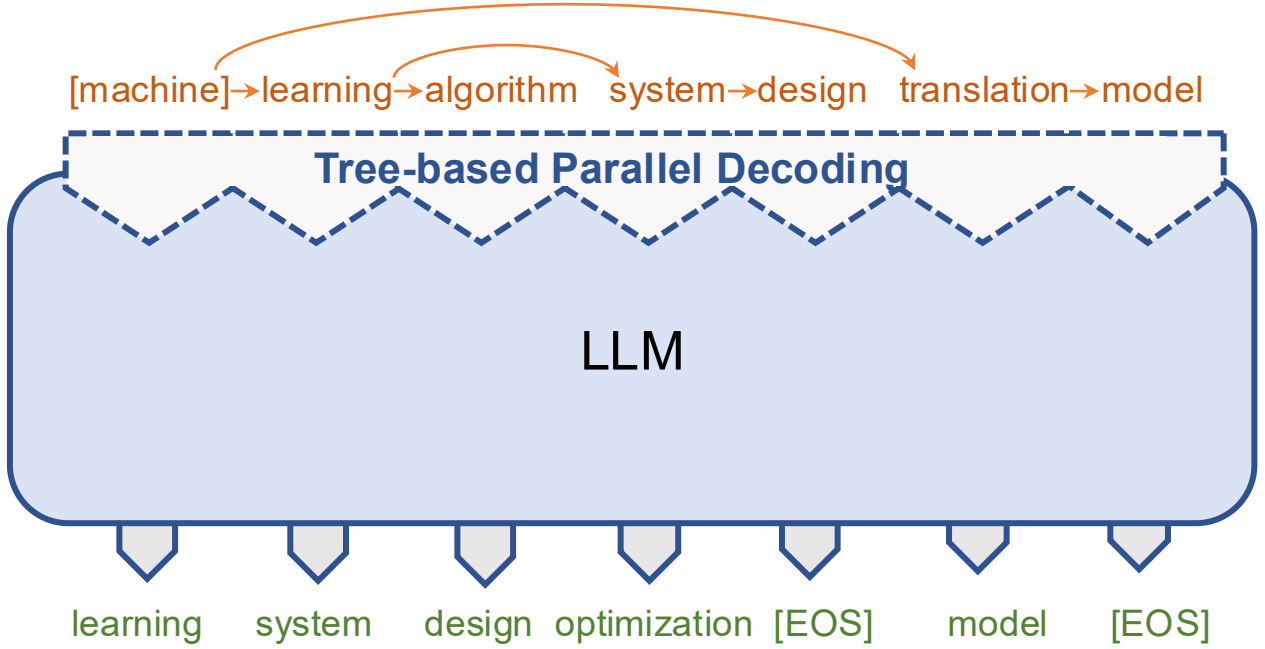
Speculated Token Tree



Verified output: machine learning system optimization



Verified token tree



Stochastic Decoding

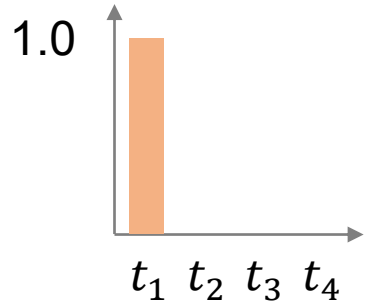
- **Challenge:** verifying stochastic equivalence

$$P_{\text{IncrDecode}}(\cdot \mid x_{<i}, \text{LLM}) = P_{\text{SpecInfer}}(\cdot \mid x_{<i}, \text{LLM}, \{\text{SSM}_i\})$$

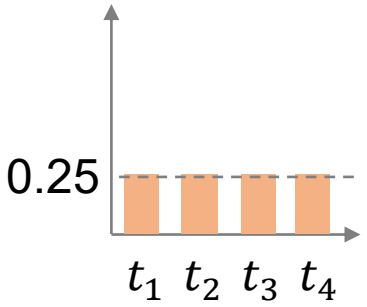
- A strawman approach: **naïve sampling**
  - Use LLM to sample $x_i \sim P_{\text{IncrDecode}}(\cdot \mid x_{<i}, \text{LLM})$
  - Verify if x_i is in the token tree

Naïve Sampling can be Suboptimal

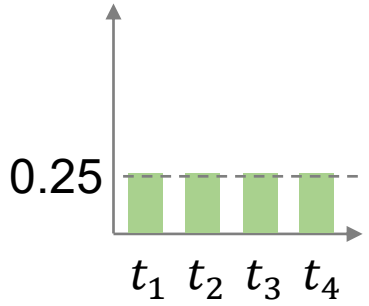
- Assume one LLM, two SSMs, and four possible tokens: t_1, t_2, t_3, t_4



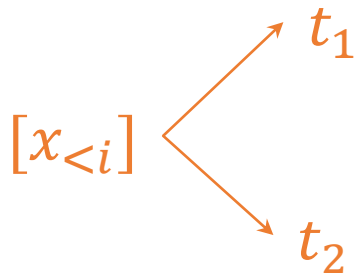
SSM 1: $P(\cdot\,|x_{<i}, \text{SSM}_1)$



SSM 2: $P(\cdot\,|x_{<i}, \text{SSM}_2)$



LLM: $P(\cdot\,|x_{<i}, \text{LLM})$



Token Tree

Naïve sampling's verification prob. = **50%**

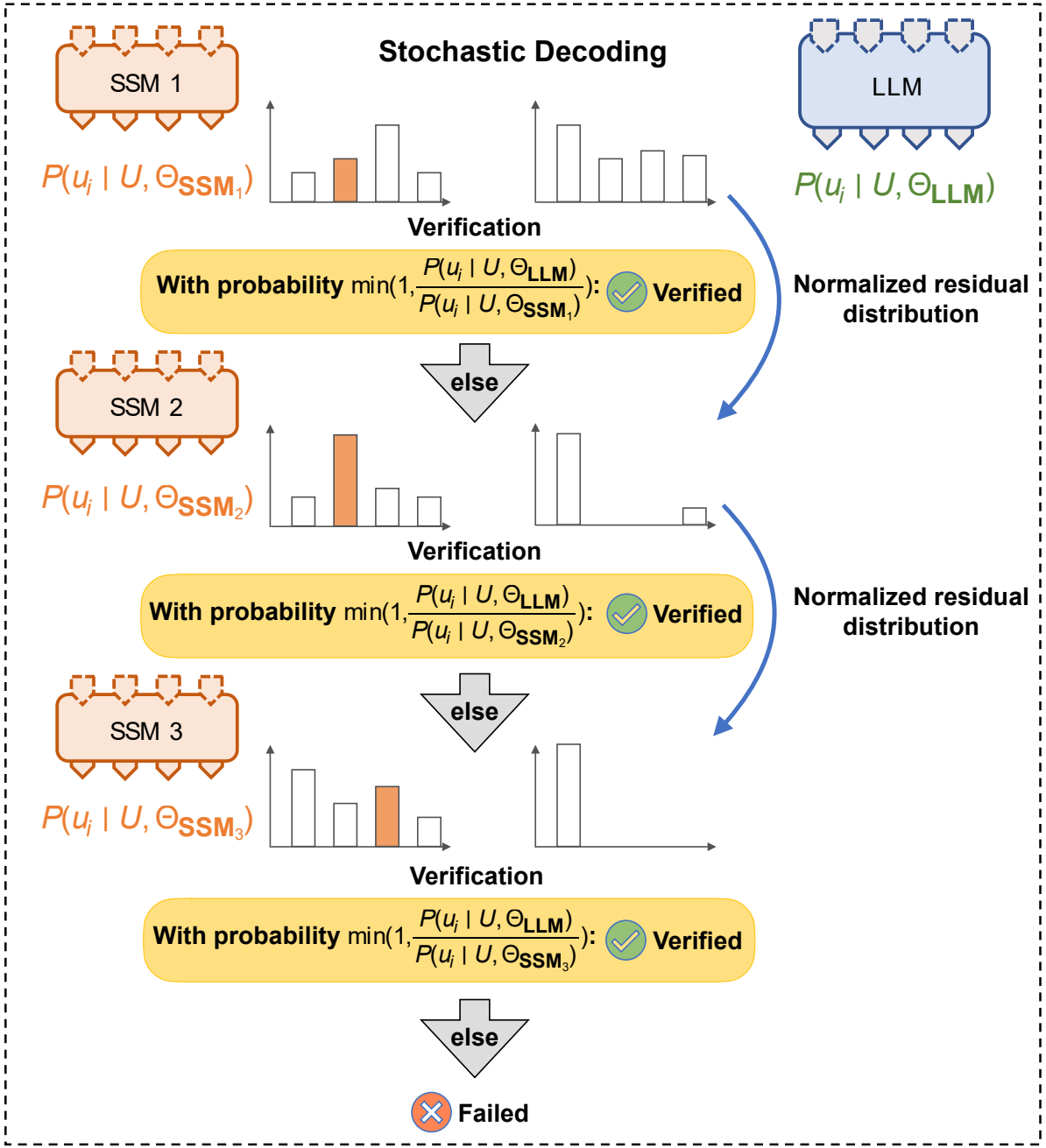
But we can do better by directly accepting SSM 2; verification prob. = 100%

Key issue: naïve sampling ignores correlation between $P(\cdot\,|x_{<i}, \text{SSM})$ and $P(\cdot\,|x_{<i}, \text{LLM})$

Speculative Sampling

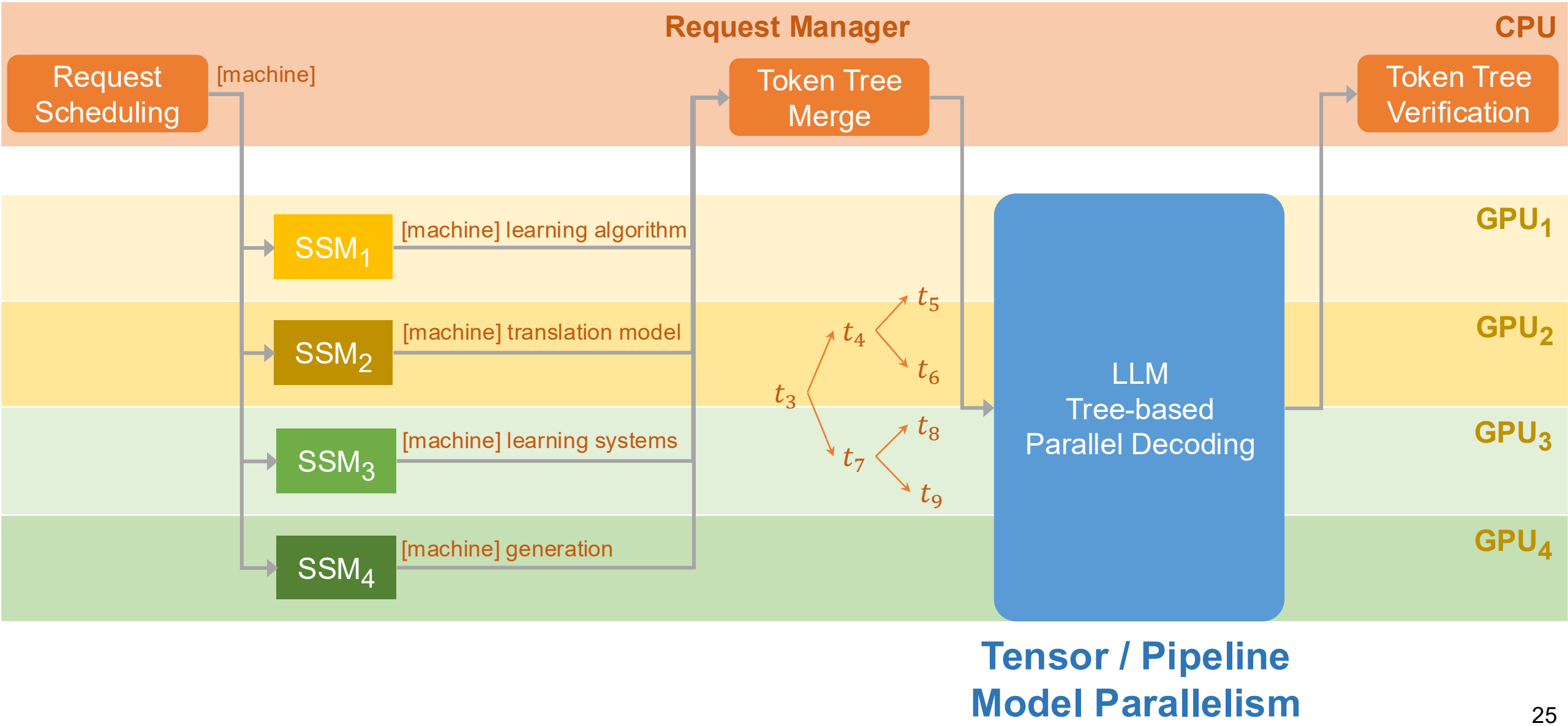
1. Sample a token $x \sim P(u_i|U, \Theta_{SSM})$ using SSM
2. If $P(x|U, \Theta_{SSM}) \leq P(x|U, \Theta_{LLM})$, directly accept x
3. If $P(x|U, \Theta_{SSM}) > P(x|U, \Theta_{LLM})$, accept x with prob. $\frac{P(x|U,\Theta_{LLM})}{P(x|U,\Theta_{SSM})}$
4. If reject x, normalize residual distribution

$$P'(x|U, \Theta_{LLM}) = norm(\max(0, P(x|U, \Theta_{LLM}) - P(x|U, \Theta_{SSM})))$$

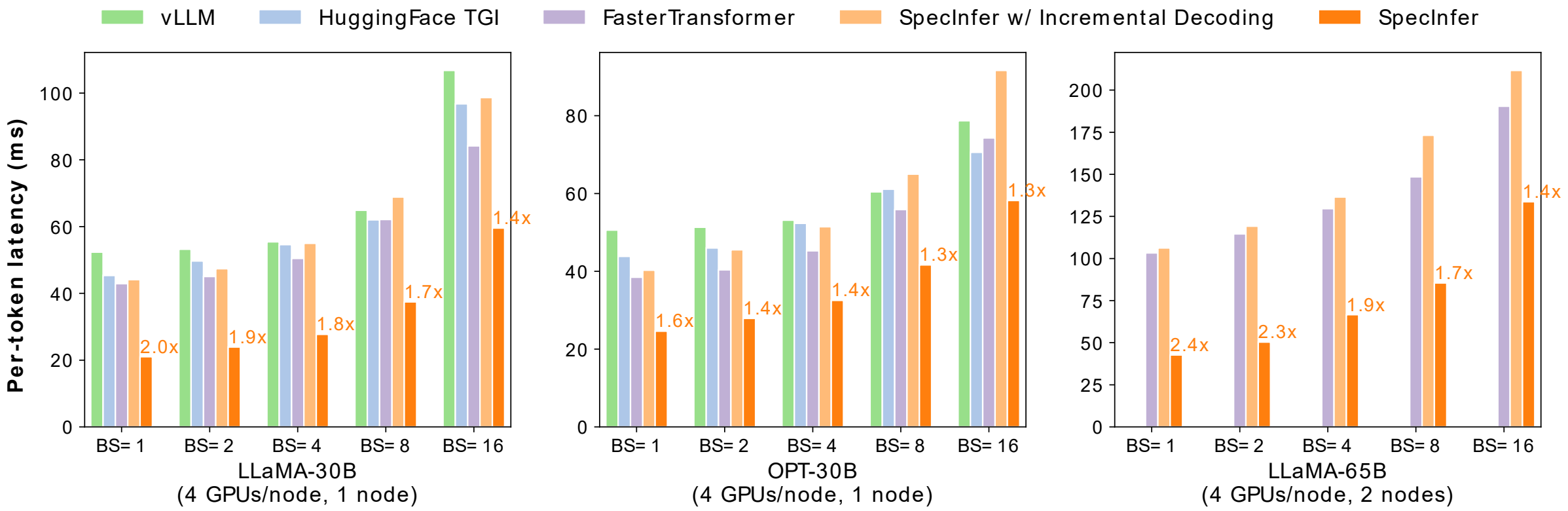


* Details in SpecInfer: Accelerating Large Language Model Serving with Tree-based Speculative Inference and Verification. ASPLOS'24

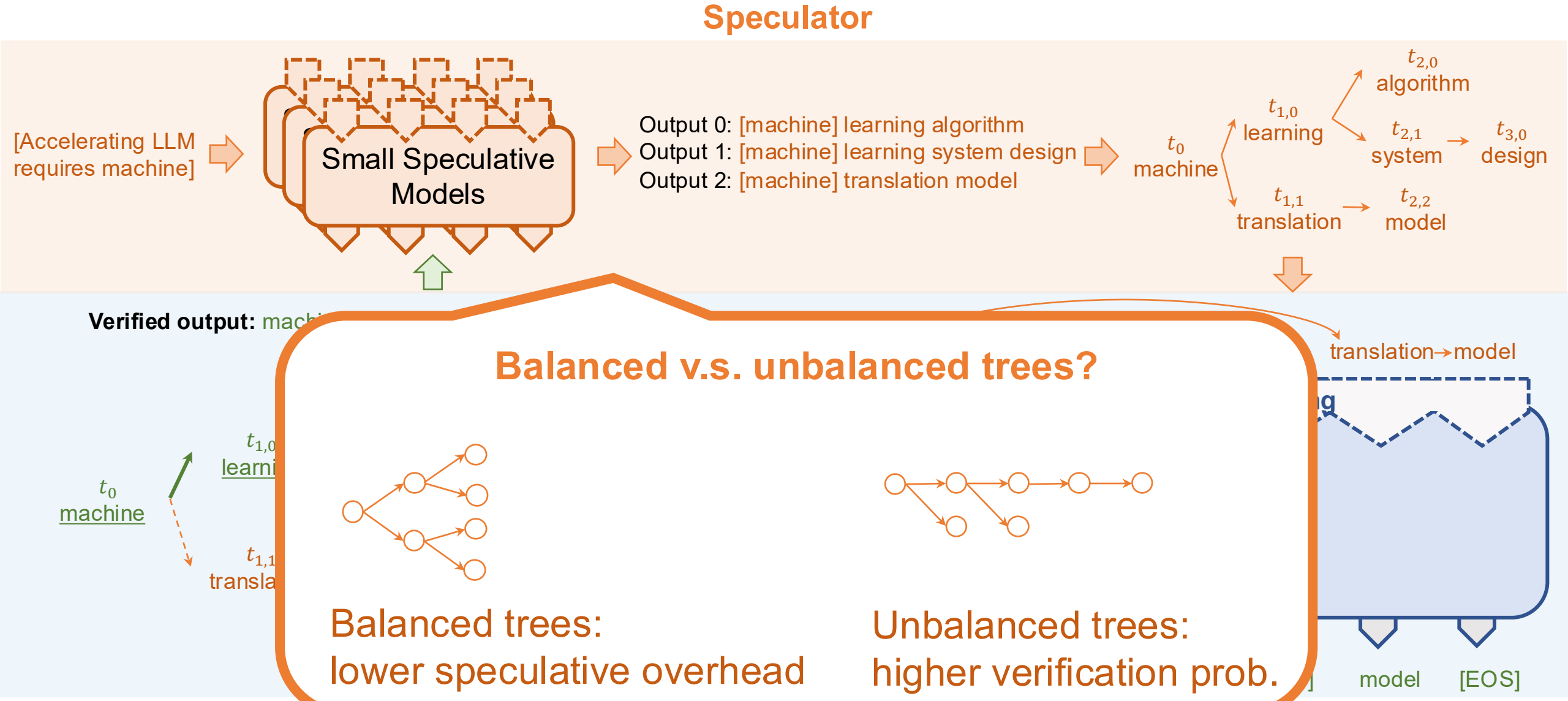
Distributed LLM Serving



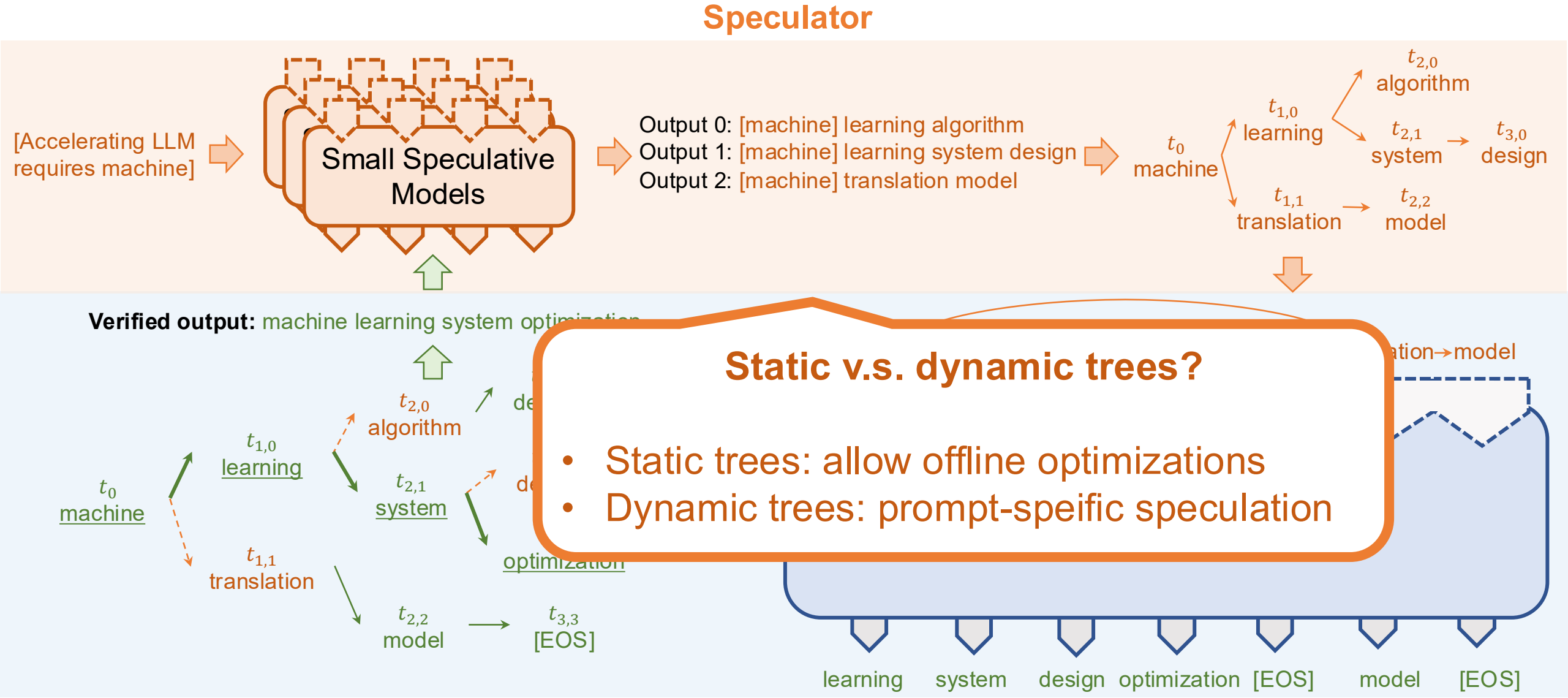
SpecInfer Accelerates LLM Inference by 1.3-2.4x



Open Research Questions



Open Research Questions



Static v.s. dynamic trees?

- Static trees: allow offline optimizations
- Dynamic trees: prompt-speific speculation

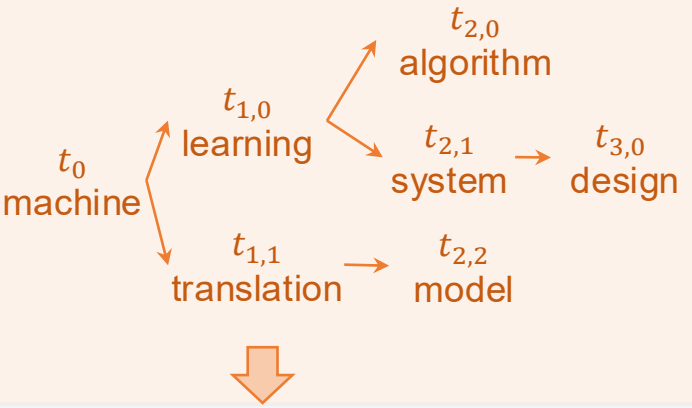
Open Research Questions

Speculator

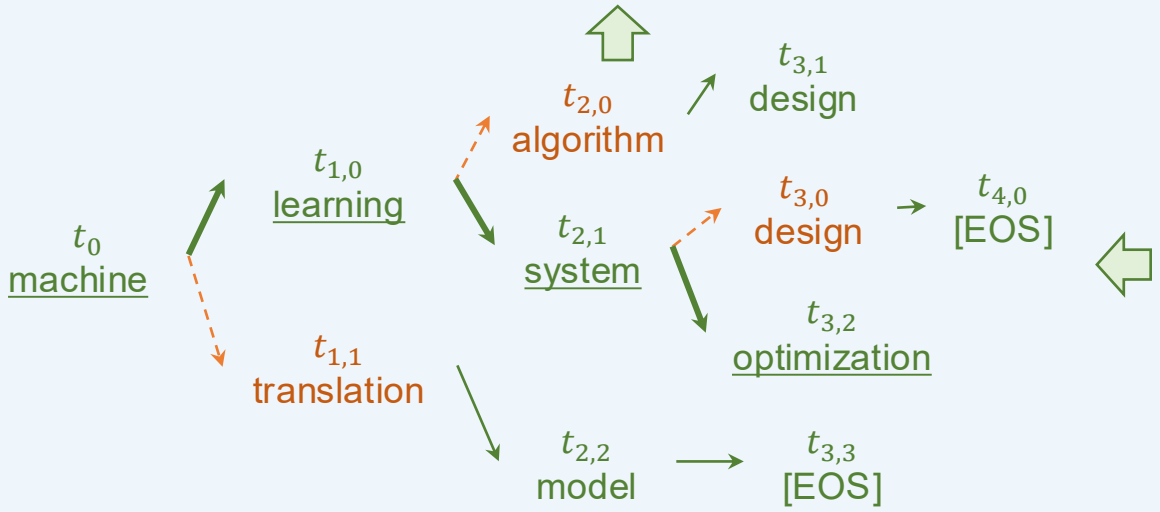
[Accelerating LLM requires machine]

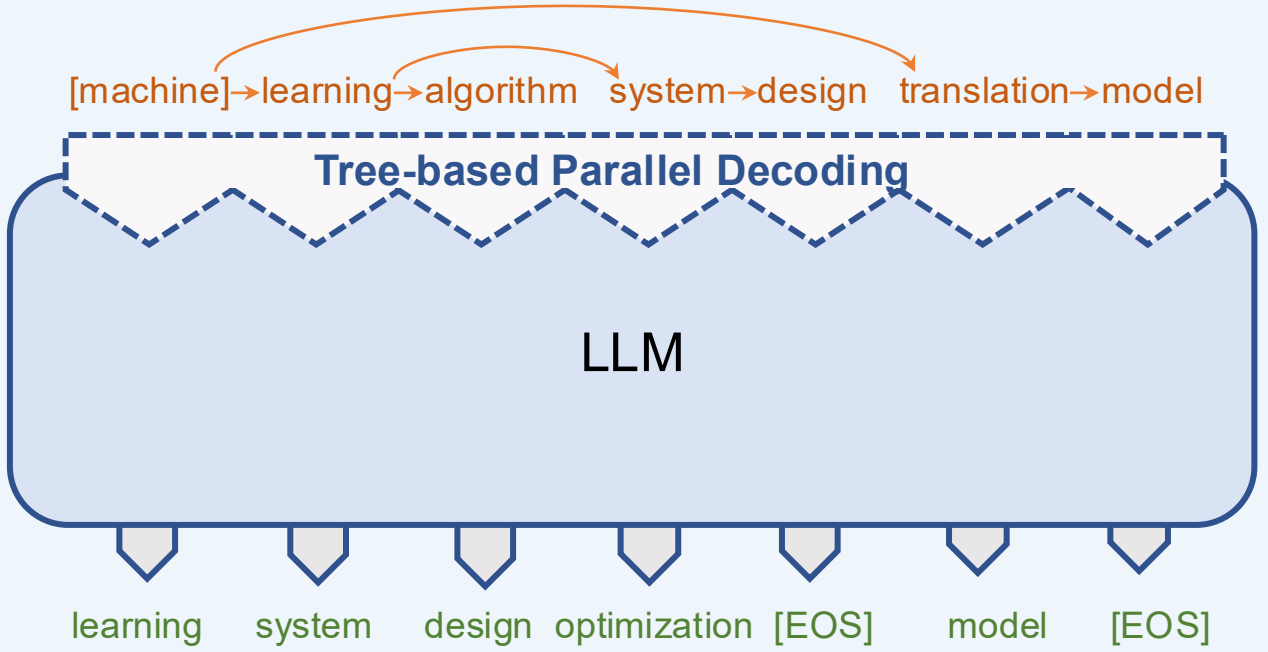
Sm

How to verify stochastic decoding?
- Multi-step speculative sampling
- Naïve sampling
- Greedy sampling
- Mixture?



Verified output: machine learning system optimization

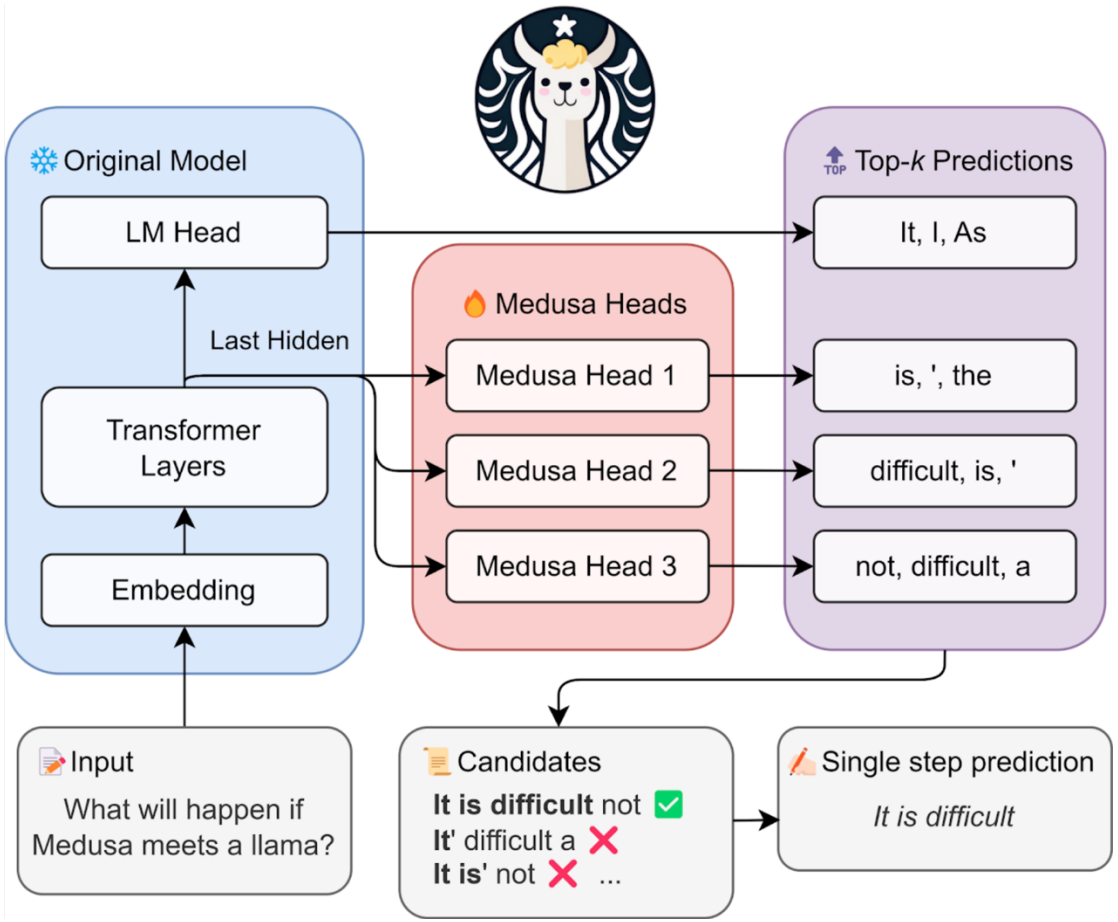


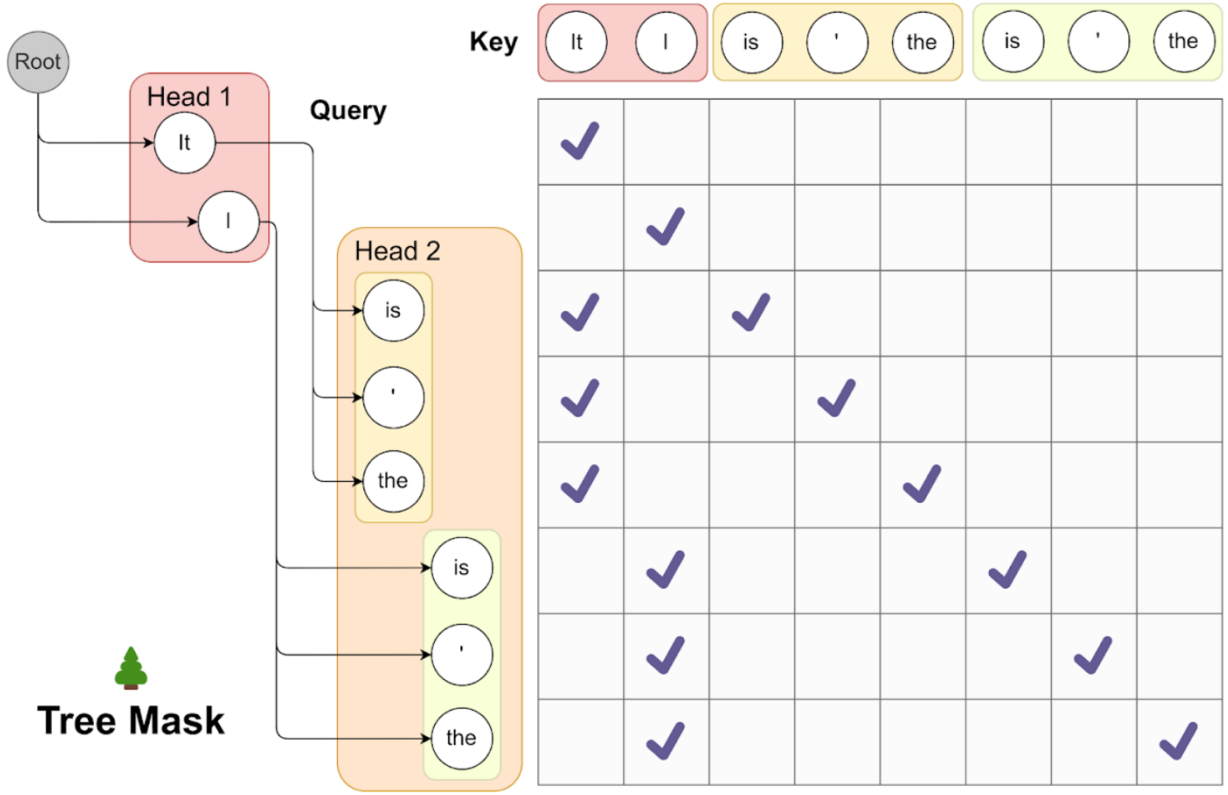


Verifier

Medusa: Speculative Decoding with Multiple Decoding Heads

- Introduce trainable attention heads to predict future tokens

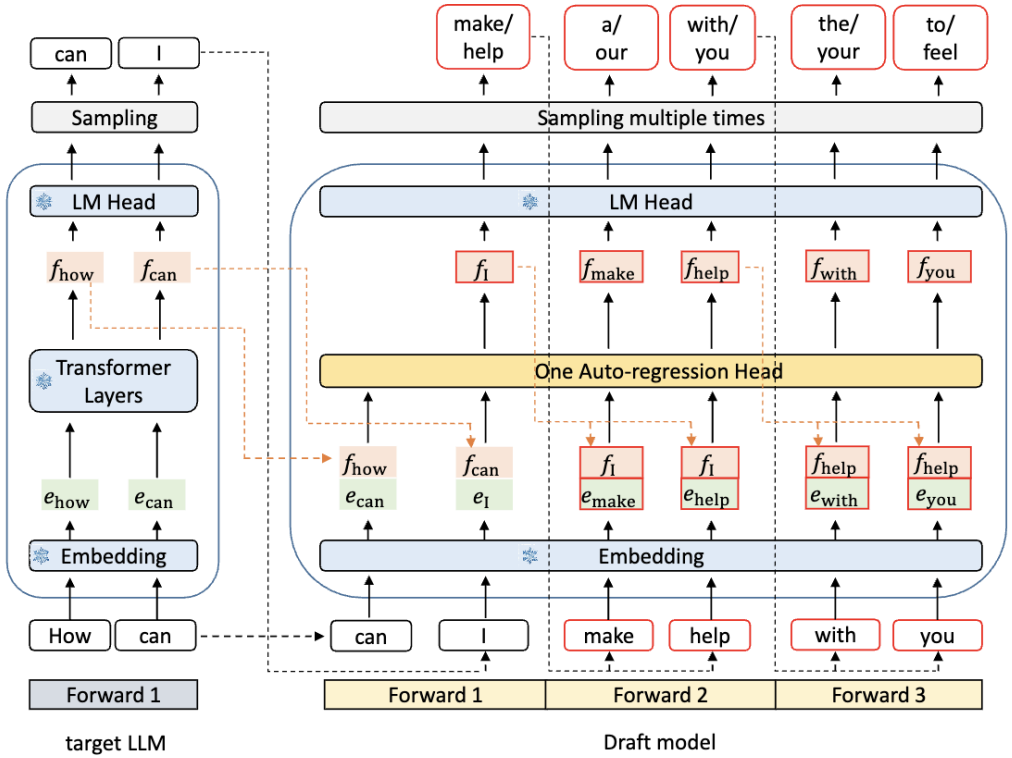


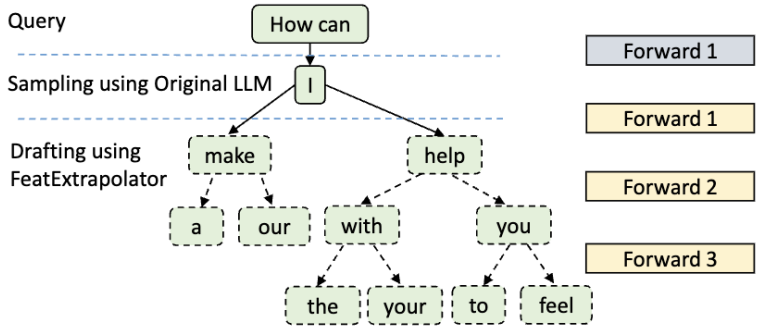


https://www.together.ai/blog/medusa

Eagle: Dynamic Speculative Token Trees

- Draft model reuses embedding and LM head layers of the LLM, and introduces a trainable **attention layer**
- **Expand phase**: draft model selects K nodes to decode in each iteration

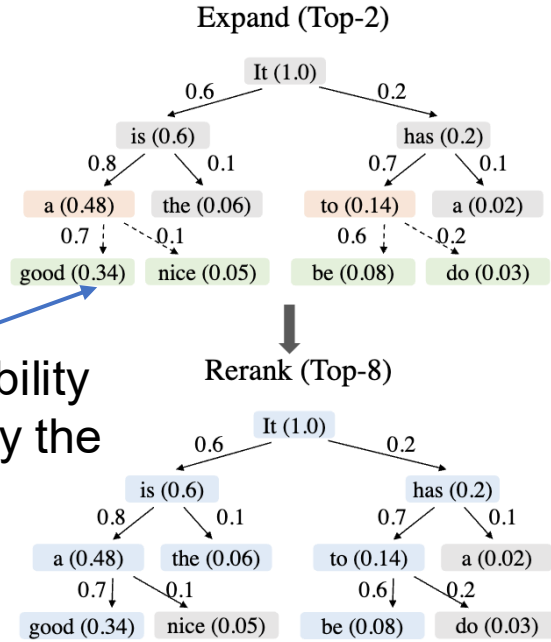




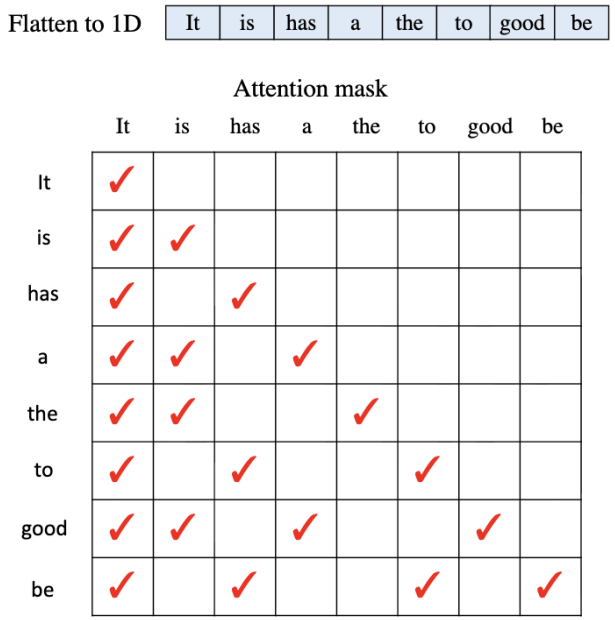
EAGLE: Speculative Sampling Requires Rethinking Feature Uncertainty
EAGLE-2: Faster Inference of Language Models with Dynamic Draft Trees

Eagle: Dynamic Speculative Token Trees

- Draft model reuses embedding and LM head layers of the LLM, and introduces a trainable **attention layer**
- **Expand phase**: draft model selects K nodes to decode in each iteration
- **Rerank phase**: select N nodes for verification based on accumulated likelihood



Accumulated probability to select that path by the draft model



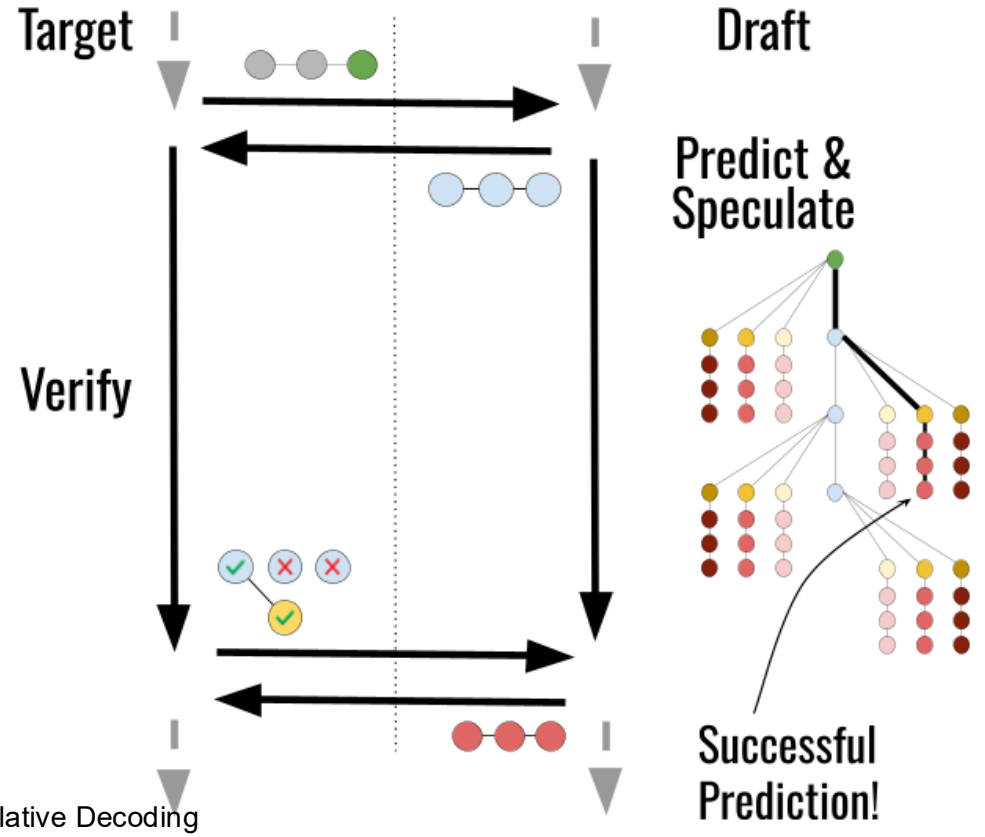
EAGLE: Speculative Sampling Requires Rethinking Feature Uncertainty
EAGLE-2: Faster Inference of Language Models with Dynamic Draft Trees

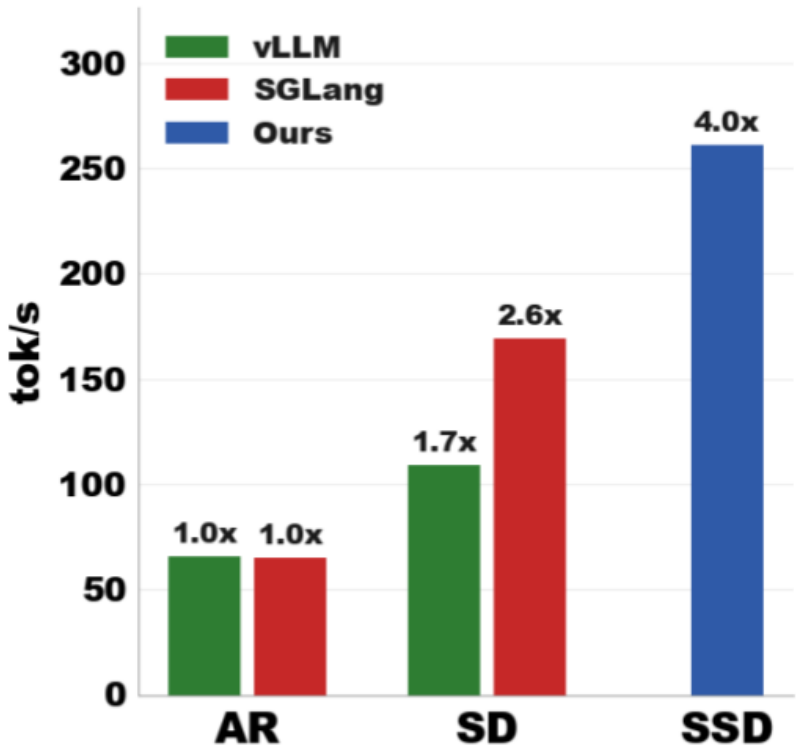
Speculative Speculative Decoding (SSD)

Issue: verifier must wait idly for the draft to speculate

Speculative Speculative Decoding (SSD)

- Speculation runs on a separate device in parallel with verification
- Draft model precomputes speculations for many possible verification outcomes and returns the speculated tokens immediately if one occur





Today's Lecture: Speculative Decoding

- Model-based speculative decoding
  - Using a separate draft model to predict LLM's output

- **Model-free speculative decoding**
  - Using previously generated tokens to predict future tokens

Prompt Lookup Decoding

- **Prompt:** What is the capital of South Korea?

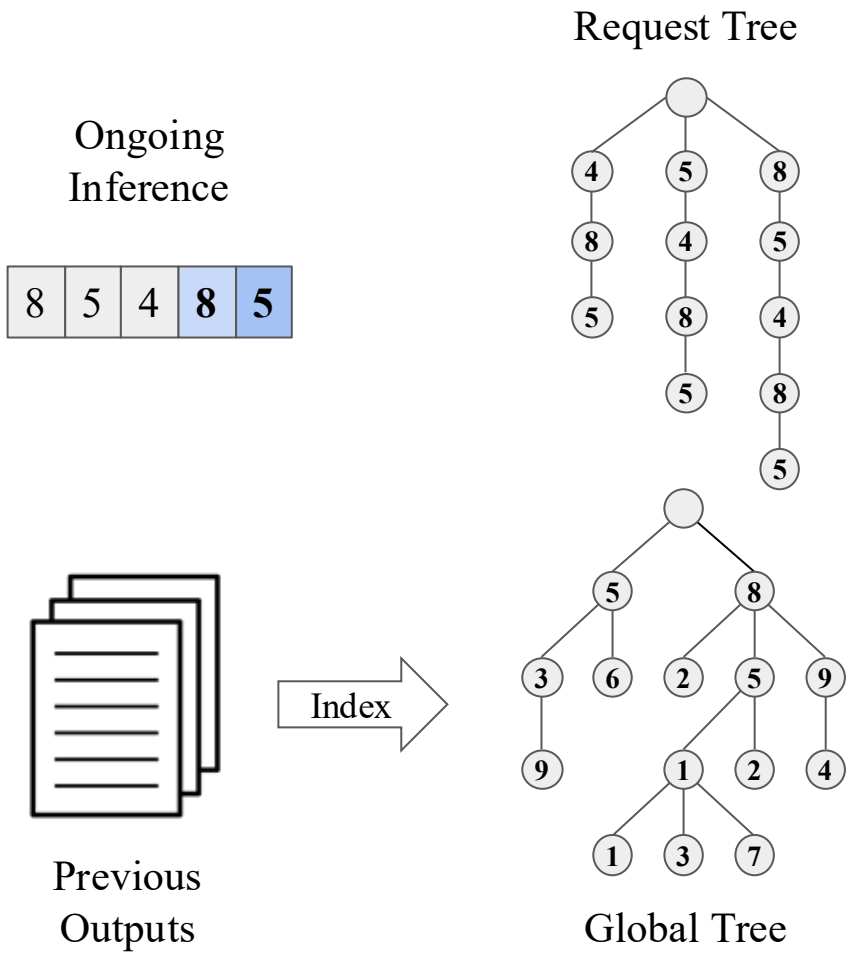
2-gram	3 speculate tokens
What is	the capital of
is the	capital of South
the capital	of South Korea
capital of	South Korea ?

During generation:

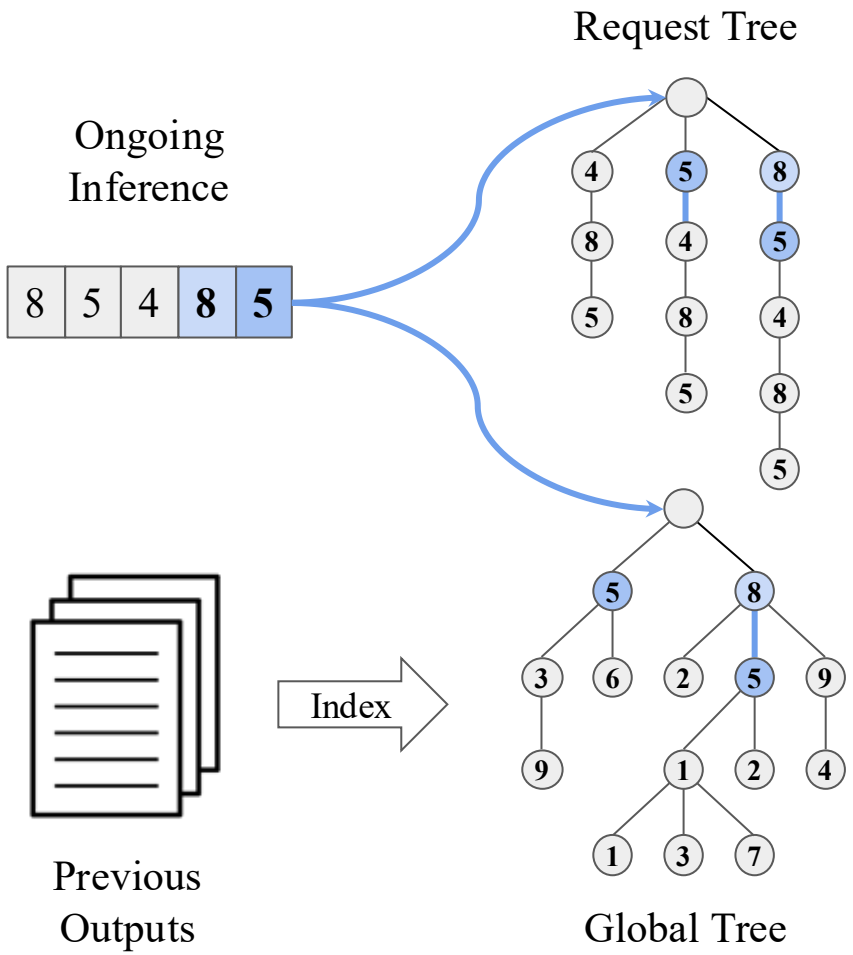
the capital of South Korea

✅ ✅ ✅

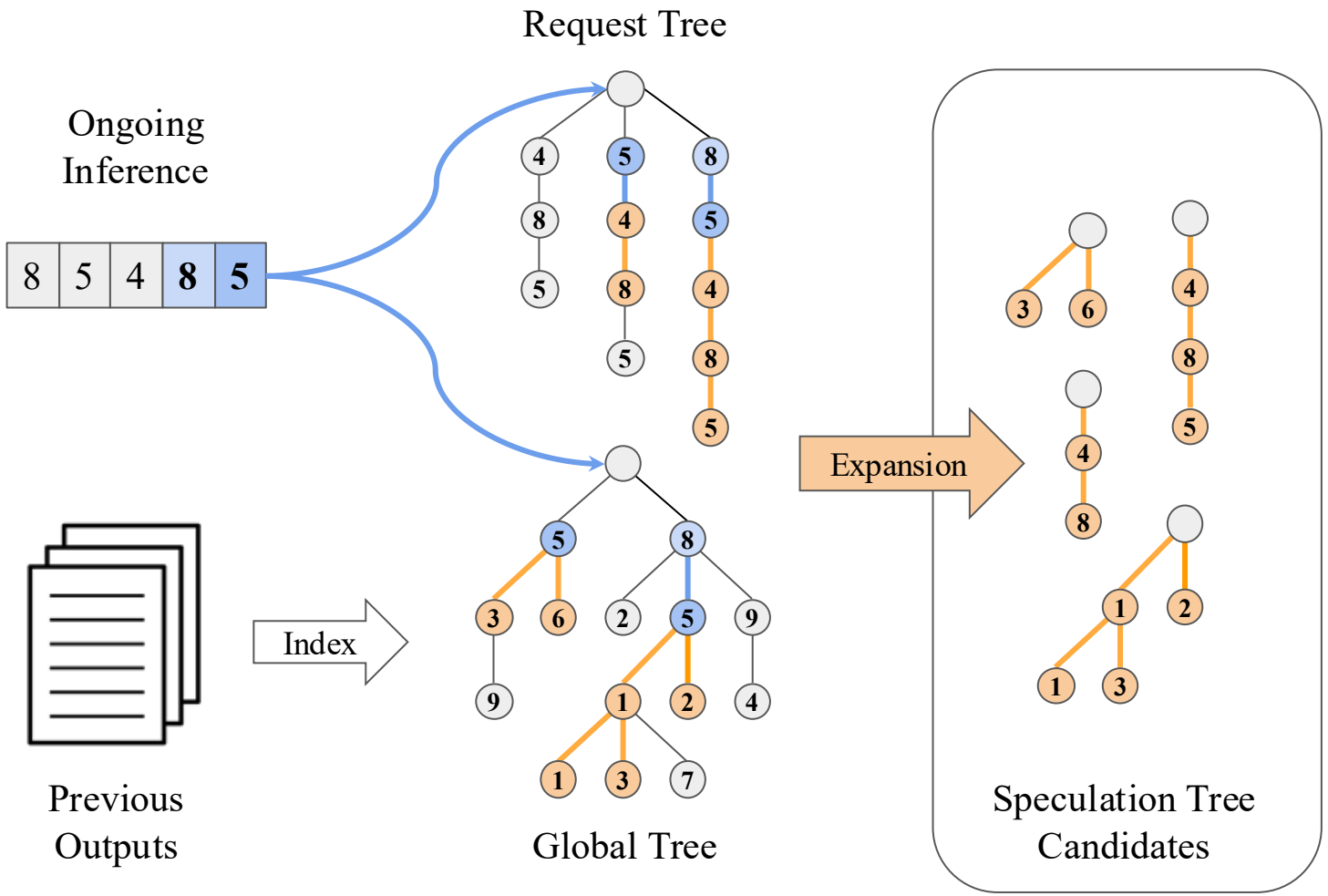
SuffixDecoding: Two-Tier Suffix Tree Speculation



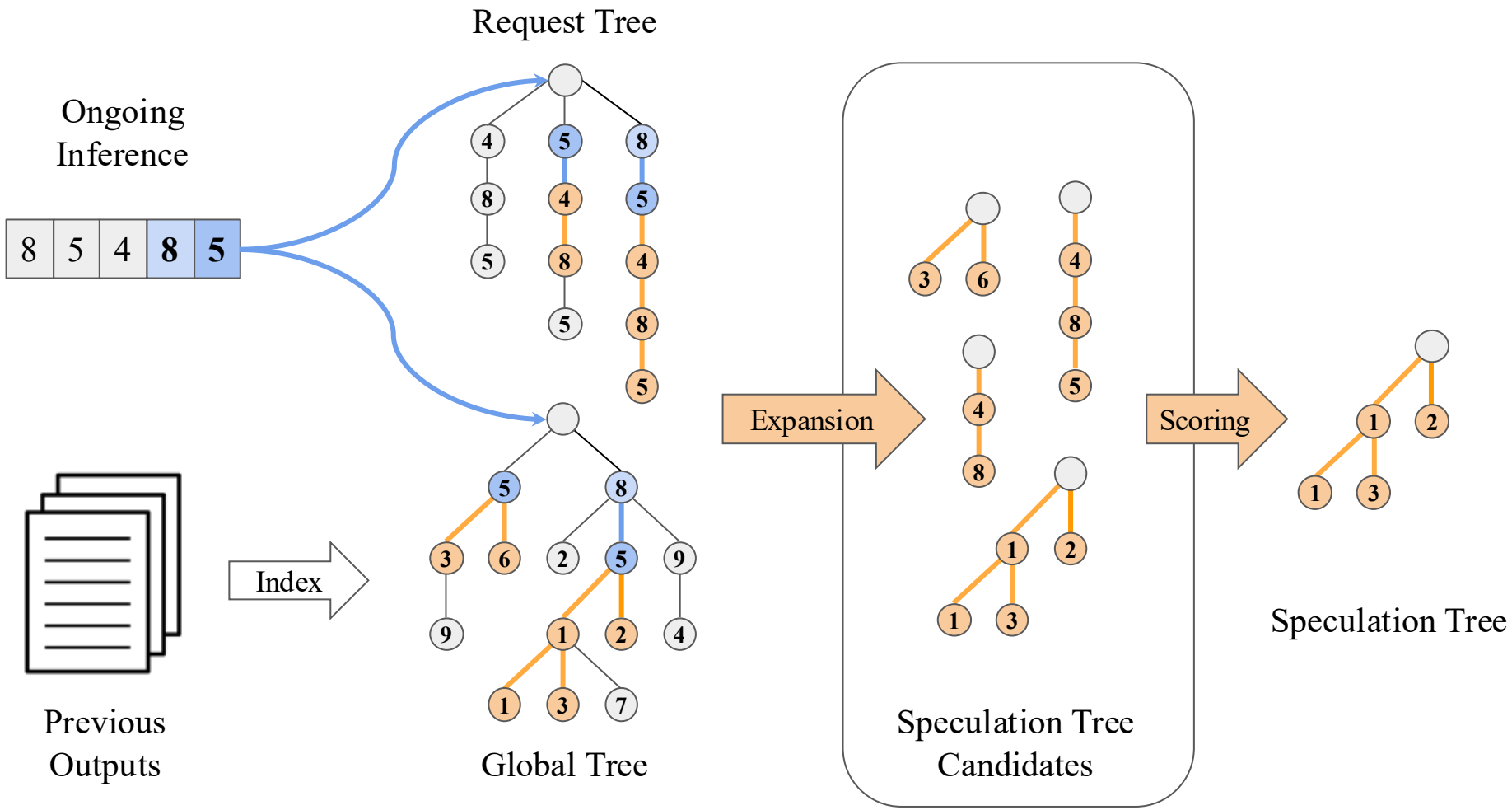
SuffixDecoding: Two-Tier Suffix Tree Speculation



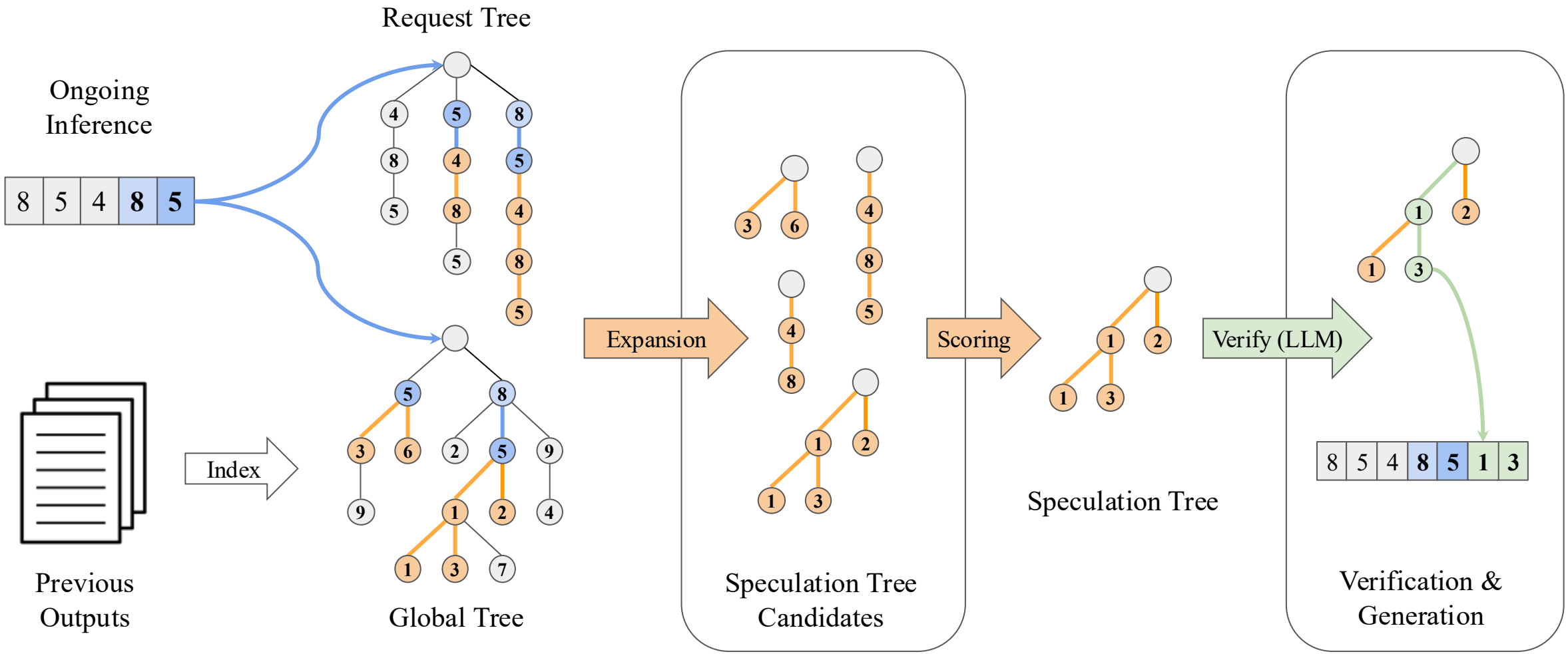
SuffixDecoding: Two-Tier Suffix Tree Speculation



SuffixDecoding: Two-Tier Suffix Tree Speculation

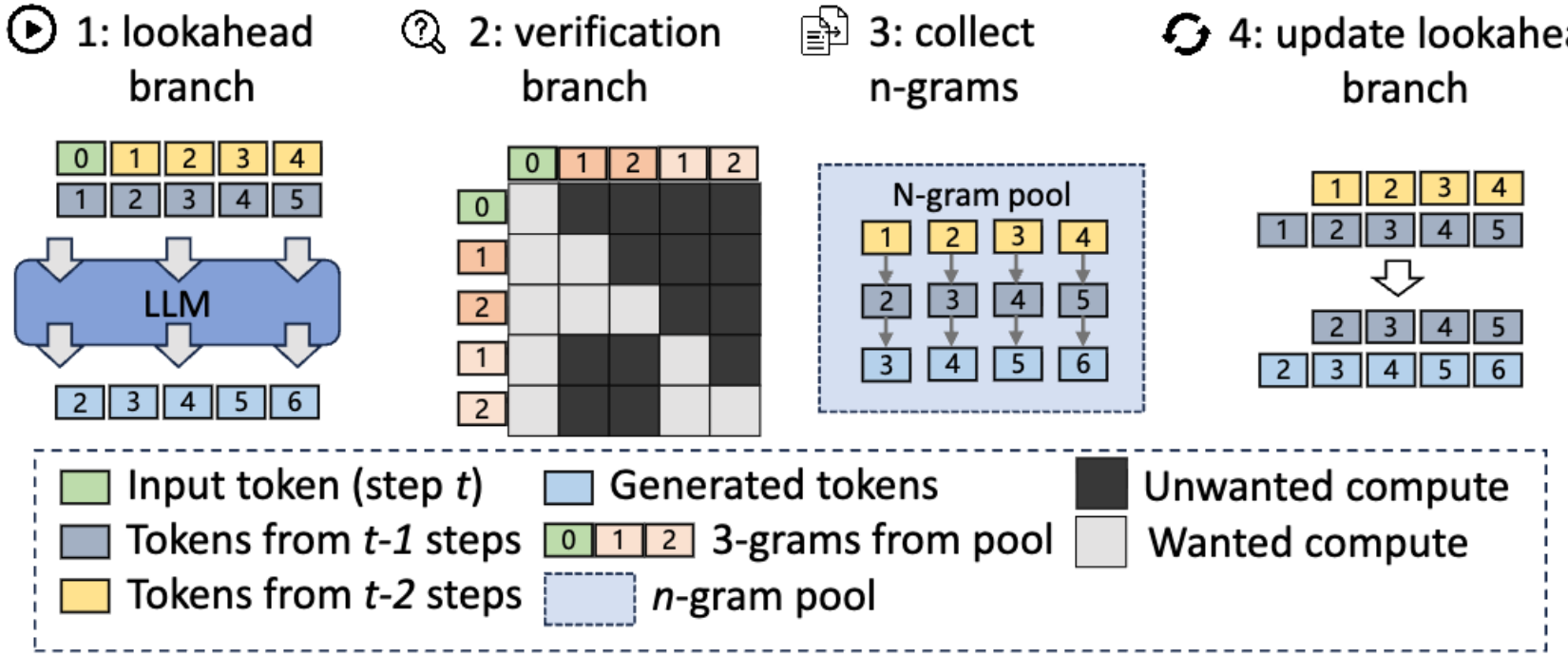


SuffixDecoding: Two-Tier Suffix Tree Speculation



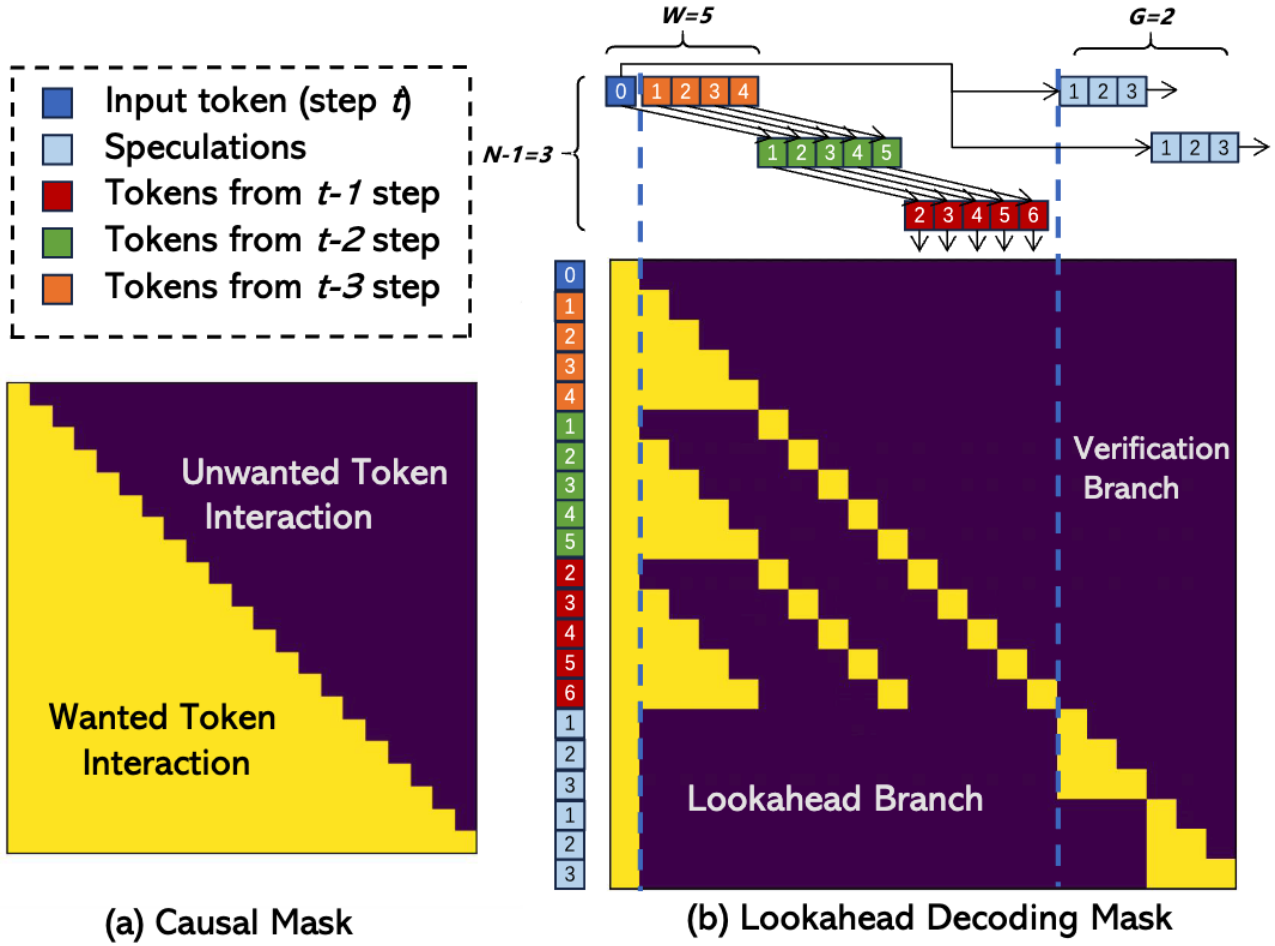
Lookahead Decoding

1. Lookahead branch: generate one token at each position
2. Verification branch: verify multiple 3-grams (searched from the 3-gram pool)
3. Collect and cache newly generated 3-grams from lookahead branch
4. Update lookahead branch



Lookahead Decoding

- Batch verification and lookahead branches in a single pass



Today's Lecture: Speculative Decoding

- Model-based speculative decoding
  - Using a separate draft model to predict LLM's output

- Model-free speculative decoding
  - Using previously generated tokens to predict future tokens