ModelRunner and Forward Batches

Deep Dive into SGLang — Chapter 12

Yin Yang

Foundation Books Project

Where this chapter sits

This is the chapter where scheduling bookkeeping becomes executable.

- The scheduler has already chosen *which* requests run this step.
- ModelRunner must turn that decision into tensors and metadata that model layers can consume.
- Two requests can share one model call and still mean different things.

Goal: by the end you can look at any forward-batch field and say which request, which token row, and which cache slot it must describe — and what breaks if it lies.

The failure this chapter is about

The scheduler hands over request order $[A, B, C]$, with A the continuing pancake request r_{pan}.

- Suppose one metadata vector silently swaps the first two rows.
- The model still returns a *valid* tensor of the right shape.
- The runtime then commits a row to the wrong request.

The central failure is not an invalid tensor shape. It is a valid tensor whose rows attach to the wrong request, cache slot, or output obligation. Every section is one defense against that.

Roadmap

The batch ladder

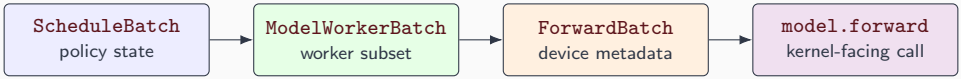
Two index spaces and a mode

From concept to code

Padding, output, and repair

The batch ladder

Scheduler state narrows into a model call



- Each rung *removes* scheduler-only state or *adds* model-runner metadata.
- No rung may change which request row or token row a field describes.

The batch ladder is why SGLang follows `ScheduleBatch` all the way into `model.forward`.

Two index spaces and a mode

Every field lives in one of two index spaces

The running step: a `MIXED` batch, request order $[A, B, C]$.

Index space	Fields	Running value
Request rows R	`seq_lens`, `req_pool_indices`	$[129, 81, 30]$, $[7, 12, 19]$
Token rows $\mathcal{T}$	`input_ids`, `positions`, `out_cache_loc`	8 rows $[A_{128}, B_{80}, C_{24..29}]$

A, B decode at positions $128, 80$; C contributes prefill suffix $24, \ldots, 29$. So $b = 3$ request rows but $n = 8$ token rows.

R-indexed fields name requests; $\mathcal{T}$-indexed fields name token rows. A shape check cannot tell you which is which.

The forward mode tags how to read the bundle

Mode	Contract carried by the batch
`EXTEND`	prefix len, extend len, flattened suffix locations
`DECODE`	one new token row per resident request
`MIXED`	chunked-prefill suffix rows + decode rows (extend-like)
`IDLE`	no real token rows, but an explicit empty contract
`PREBUILT`	KV-ready disaggregated decode, prepared elsewhere

The mode is more precise than a prefill/decode boolean: position construction, cache writes, speculative masks, and graph eligibility all depend on the exact tag.

From concept to code

From concept to code: the forward-batch bridge

`ForwardBatch.init_new` copies scheduler state into the object the model and backends consume.

from `python/sglang/srt/model_executor/forward_batch_info.py`

```
ret = cls(
    forward_mode=batch.forward_mode,
    req_pool_indices=batch.req_pool_indices,
    seq_lens=batch.seq_lens,
    out_cache_loc=batch.out_cache_loc,
    ...
    req_to_token_pool=model_runner.req_to_token_pool,
    token_to_kv_pool=model_runner.token_to_kv_pool,
    attn_backend=model_runner.attn_backend,
)
```

Everything downstream code reads enters here: mode, rows, lengths, output slots, the pools, and the backend handle.

Padding, output, and repair

Three obligations after assembly

Padding Graph buckets and collective sync may add *dummy* rows. `num_token_non_padded` marks the only rows allowed to survive.

Output `LogitsMetadata.from_forward_batch` decides which hidden rows become logits, logprobs, and sampled tokens — in the same order.

Repair After the step, one *keep vector K* slices every row-indexed field at once, so survivors keep their identity.

Padding must never become visible work; output and repair must keep the same row contract the model executed under.

What the measurements separate

An RTX 5090 run instruments the executable batch so cost is not one opaque "forward" number:

Row	Raw tok	Padded tok	Padding tok
Graph padding default	6,207	6,890	683
Graph padding disabled	6,204	6,204	–
Decode-heavy mode mix	967	1,100	133

The padding account is directly visible; buffer population and attention-metadata init are emitted as separate phase timers. The executable batch has measurable adaptation work *around* the model call.

The one invariant to carry forward

Executable-batch alignment
A forward batch F is valid only when its R-indexed fields, $\mathcal{T}$-indexed fields, forward mode m_{fwd}, cache locations, logits metadata, padding, lifetime, and batch order all describe the same scheduler-selected model step.

- The running trace, graph padding, the logits handoff, and post-step filtering are all *the same rule*, applied to different fields.
- A single misaligned field can attach a logprob to the wrong row, write KV into the wrong slot, or let padding leak into output.

Next: with this alignment contract in place, the model-architecture registry can ask which model class is allowed to consume the batch.