

PR #39000 完整报告

sgl-project/sglang

[Router] Fleet-wide sampling contract 1/3: the config surface

合并时间: 2026-09-16 12:39

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/39000>

执行摘要

本 PR 是 sgl-router 的「Fleet-wide sampling contract」三栈实现的第一栈 (Stack 1/3)，新增 `--override-sampling-params` 与 `--sampling-param-conflict` 两个 opt-in 配置项，为托管端点提供运营方可控的采样契约：一个固定配置适用于路由器的所有客户端，不受部署镜像引擎默认值漂移影响。本 PR 只落地配置面 (类型、解析器、启动校验)，请求路径零改动；默认未设置 flag 时行为完全不变。核心是新增 `experimental/sgl-router/src/config/sampling.rs` (812 行)，以手写解析器 + 严格域校验 + fail-fast 启动错误的方式定义了 `SamplingOverrides`、`SamplingField`、`ParamSpec`、`ConflictPolicy` 等类型，并由 `Config::validate` 统一兜底校验。

功能与动机

PR body 的核心表述是: "An operator running a hosted endpoint often needs the sampling contract to be theirs, not the engine's: one fixed configuration for every client of a router, independent of what the deployed image's defaults happen to be." 具体场景是像 Moonshot API 那样服务 Kimi 模型，由 Kimi-Vendor-Verifier 的 `tests/params` 套件校验受管参数 (可省略、可发送精确配置值、否则 400)。

设计上刻意采用三栈拆分: 本 PR 只加配置面; #39001 做请求路径上的强制与注入、错误码和计数器; #39002 做零拷贝注入、e2e 代理测试与 README。三个 PR 合计 +939 / +839 / +579 行，边界清晰，每栈可独立审阅。

实现拆解

1. 新增 `sampling.rs` 数据模型与解析器 (核心, +812 行) - `SamplingOverrides`: `params: BTreeMap<SamplingField, ParamSpec> + conflict: ConflictPolicy`。 `BTreeMap` 保证注入顺序稳定，转发体跨运行字节一致。 - `SamplingField`: 8 个字段枚举 (`temperature`、`top_p`、`top_k`、`min_p`、`repetition_penalty`、`frequency_penalty`、`presence_penalty`、`n`)，`from_wire_name()` 使 `typo` 成为启动错误而非静默不匹配。 - `ParamSpec`: `Exact(serde_json::Number)` 保留运营方字面量 (`"n": 1` 不变成 `1.0`)；`Range{lo,hi}` 为 inclusive band。 - 手写解析器 (`parse_sampling_overrides`) 使用自定义 `ObjectEntries` 按有序 `entries` 解码，拒绝重复 key，避免 `serde_json::Map` 静默覆盖问题。 - `validate_spec` 是唯一校验入口，被解析器和 `SamplingOverrides::validate` 共享；包含域检查、`top_k` 不连续域 (`{-1} ∪ [1, inf)`) 特判、band 在 `allow` 下报错等。
2. CLI 接入 (`cli.rs`, +96 行) - 新增 `--override-sampling-params` (`Option<String>`) 与 `--sampling-param-conflict` (`Option<ConflictPolicy>`，`requires = "`

override_sampling_params" 防止单独使用)。 - Cli::into_config 解析 flag 后写入 ModelConfig.sampling_overrides; 未设置时用 Default::default(), 零开销。

3. 类型挂接与校验兜底 (types.rs / mod.rs) - ModelConfig 新增 sampling_overrides 字段; Config::validate 调用 SamplingOverrides::validate(), 确保测试 fixture、未来 config 文件、admin API 等任何路径构造的配置都满足与 flag 相同的域约束。
4. 全仓 20 处 ModelConfig 字面量补齐 - 源码侧: policies/factory.rs、server/app_context.rs、server/routes/models.rs、server/routes/tokenize.rs、tokenizer/mod.rs、workers/manager.rs。 - 测试侧: tests/component/discovery/static_urls.rs 及 tests/proxy/ 下 8 个测试文件, 统一加 sampling_overrides: Default::default()。
5. 测试配套 - sampling.rs 内 12 个单元测试: 覆盖全部参数、band、整数归一化、top_k 哨兵、i64 边界、ALL/index 一致性、validate 拒绝手工构造 spec, 以及 31 例畸形输入表 (重复 key、未知 key、非数、band 缺边、倒置 band 等)。 - cli.rs 3 个测试: flag 到达 ModelConfig、reject 默认模式、allow 模式选择、单独使用 --sampling-param-conflict 被 clap 拒绝、畸形 JSON 启动失败。

experimental/sgl-router/src/config/cli.rs

CLI 入口: 新增两个 flag 的定义、requires 约束以及在 into_config 中把解析结果写入 ModelConfig 的接线逻辑, 同时承载 3 个验证 flag 到达配置层的测试。

```
// ---- fleet-wide sampling contract (opt-in) ----
/// 采样参数固定 fleet-wide, 以一个 JSON 对象给出, key 为请求体字段名。
///
/// 配置值在请求省略该字段时注入; `--sampling-param-conflict` 决定请求显式
/// 发送不同值时的行为。未知/重复 key、越界值、`allow` 下的 band 都会在启动
/// 时报错并点名违规 key。
#[arg(long, value_name = "JSON")]
pub override_sampling_params: Option<String>,
/// 冲突策略: `reject` (默认) 准入前 400; `allow` 原样转发客户端值。
///
/// clap 的 `requires` 保证该模式单独设置 (没有 `--override-sampling-params`
/// 时) 会被拒绝——模式本身不治任何东西。
#[arg(
    long,
    value_enum,
    value_name = "MODE",
    requires = "override_sampling_params"
)]
pub sampling_param_conflict: Option<ConflictPolicy>,

// 在 `Cli::into_config` 中解析 flag 并写入 `ModelConfig`。
let sampling_overrides = match &self.override_sampling_params {
    None => SamplingOverrides::default(), // 未设置 = 旧行为, 零开销
    Some(raw) => {
        // 解析失败直接返回 Err, 启动即失败而不是带病运行。
        parse_sampling_overrides(raw, self.sampling_param_conflict.unwrap_or_default())?
    }
}
```

```

    }
};

let config = Config {
    // ... 其它字段省略 ...
    model: ModelConfig {
        // ... 其它字段省略 ...
        sampling_overrides,
    },
    // ... 其它字段省略 ...
};
config.validate()?;

```

experimental/sgl-router/src/config/types.rs

ModelConfig 新增 **sampling_overrides** 字段，是采样契约从 CLI 到运行时的数据载体，所有手工构造 **ModelConfig** 的点都必须补齐该字段。

```

// `ModelConfig` 结构体（节选）：新增字段放在所有既有可选字段之后，
// 默认空值保证未启用时行为完全不变。
#[derive(Debug, Clone)]
pub struct ModelConfig {
    pub id: String,
    pub tokenizer_path: String,
    pub policy: PolicyKind,
    pub decode_policy: DecodePolicyKind,
    pub bucket_config: Option<BucketConfig>,
    pub circuit_breaker: Option<CircuitBreakerConfig>,
    pub cache_aware: Option<CacheAwareConfig>,
    pub sticky: Option<StickyConfig>,
    pub affinity: Option<AffinityConfig>,
    pub fused: Option<Vec<FusedTerm>>,
    pub eligibility: Option<EligibilityConfig>,
    /// 采样参数 fleet-wide 固定，以及请求发送不同值时的行为：准入前 400，
    /// 或客户端值原样转发。任一模式下，配置值都会在请求省略字段时注入——
    /// 见 `SamplingOverrides`。空值（默认）保持现状。
    pub sampling_overrides: SamplingOverrides,
}

```

评论区精华

本 PR 无 review 评论线程，sherlockwu 直接 APPROVED。最有价值的「讨论」其实内嵌在代码注释和 PR body 中，特别是以下几点设计论证：

- 为什么手写解析器而不是 **serde(deny_unknown_fields)**： "the flag is read once, at startup, on a router that crash-loops if it is wrong, so the message an operator reads out of kubectl logs is the whole debugging session." 每个拒绝都必须点名违规 key、值和违反的域。
- 为什么用有序 entries 而非 **serde_json::Map**： "a map keeps only the last of a repeated key, so {"temperature": 0, "temperature": 1} would start cleanly and enforce a value

the operator did not write."

- 为什么 `ParamSpec::Exact` 持有 `serde_json::Number` 而非 `f64`: 注入要重新发出运营方的字面量, "n": 1 不能到引擎变成 1.0。
- 为什么 `band` 在 `allow` 下是启动错误: `band` 从不注入、`allow` 从不拒绝, 两者并存就是 `dead config`。

这些决策全部有对应测试钉住 (重复 key、倒置 `band`、`band+allow`、`i64` 边界、`top_k` 哨兵等)。

风险与影响

技术风险:

1. 配置域与引擎实际的偏差: 接受的域刻意采用 OpenAI 客户端库普遍校验的区间, 比 `SamplingParams.verify` 更窄。若未来引擎支持更宽范围 (如 `temperature` 负值或 `top_k: -1` 哨兵), 当前 `check_domain` 可能误拒绝合法配置, 需要回头放宽。
2. `band` 的 `f64` 精度: `Range { lo, hi }` 使用 `f64`, 对接近 `i64` 边界的大整数 (如 `n`) 可能丢精度; 当前文档已知悉, `band` 主要面向连续域字段。
3. 契约扩展的兼容性: `ConflictPolicy` 若未来新增模式 (如 `rewrite`), `validate_spec` 中 "band 在 `allow` 下报错" 的规则需要同步扩展, 否则可能放行半吊子配置。
4. 构造点遗漏风险: 虽然本 PR 补全了全部 20 处 `ModelConfig` 字面量, 但未来新增构造点时若走 `..Default::default()` 之外的方式, 仍有漏配风险; 编译器会强制检查, 实际风险可控。

影响评估:

- 对用户: 默认配置零行为变化; 启用后, 托管端点可获得与 Moonshot API 一致的采样参数治理 (省略即注入、冲突即 400 或穿透)。
- 对系统: 改动全部在 `experimental/sgl-router` 配置层, 请求路径和引擎无接触, 无性能影响 (仅在启动时解析一次)。
- 对团队: 为 Stack 2/3 提供了稳固的类型与校验地基; `Config::validate` 的统一兜底使未来的配置来源 (文件、admin API) 自动继承相同约束。

关联脉络

- #39001 (Stack 2/3, 同一作者): 本 PR 的直接后继, 请求路径上的强制与注入、错误码与计数器, 将消费 `SamplingOverrides` 的 `params` 与 `conflict`。
- #39002 (Stack 3/3): 零拷贝注入、e2e 代理测试与 README 文档, 其中 README 是 `--override-sampling-params` 的 `clap help` 指向的文档目标。
- #39004 (近期 router 配置演进): 同一模块的「按 worker 解析转发协议」改动, 说明 `sgl-router` 在配置面持续扩展, 本 PR 是其中一条功能线的一部分。
 - 更宏观地看, 这组 Stack 与 Kimi-Vendor-Verifier 的 `tests/params` 套件呼应, 是 SGLang 为托管服务提供商构建「运营商级治理能力」的一个环节——此前 HiCache 分层存储路由 (#39109、#39110、#39111 系列) 也是同一方向的铺垫。