

PR #39396 完整报告

sgl-project/sglang

[docs] DeepSeek-V4: MI355X PD disaggregation recipes for all three strategies

合并时间: 2026-09-14 17:11

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/39396>

DeepSeek-V4 MI355X PD 三策略配方与 Playground 支持 (PR #39396)

执行摘要

本 PR 为 DeepSeek-V4 cookbook 补上 MI355X 的 prefill/decode 分离部署配方，并按低延迟、均衡、高吞吐三档策略分别给出角色级 sizing；为让配方可组合，文档站共享引擎 `_playground.jsx` 新增 `roleOverrides`、`forceOff`、`notes`、`resolveRouter` 等扩展点，顺带修掉 DP-Attention 关闭后遗留 `--enable-dp-attention-local-control-broadcast`、以及投机解码 sizing 提示被抑制两个问题。改动全部落在文档与 Playground 配置层（4 个文件，+544/-40），不触碰任何运行时、内核或模型代码，因此不存在服务侧回归；风险集中在共享生成器的连带影响与配方尚未端到端验证两点。

功能与动机

PR body 给出的动机很直接：

The DeepSeek-V4 cookbook had no prefill/decode disaggregation recipe for MI355X, so the MORI-IO transport, the per-role sizing it needs, and the cache tiers that pair with it were not reachable from the Playground.

也就是说，MI355X 用户此前既拿不到 ROCm 专属的 MORI-IO 传输配置，也无法从 Playground 组合出可复制的 PD 命令。作者同时强调这次不是加一个开关：

Disaggregation is not one operating point, so each serving strategy gets its own role sizing rather than a single "PD on" toggle.

三档策略在并行度（TP-only 对 TP8/DP8）、`--max-running-requests`（8 / 96 / 256）、decode graph ladder（1-8 / 1-96 / 1-32）以及卸载层级（无 / HiCache / UMBP）上都不不同，因此必须按角色、按策略表达。

实现拆解

1. 共享生成器：把业务约束编码进 Playground (`docs/src/snippets/_playground.jsx`)

扩展点	位置	作用与动机
<code>resolveRouter(fc, sel)</code>	模块级	允许平台用 <code>routerOverrides: [{when, port?, command}]</code> 覆写 PD 路由器，首个命中项生效，未提供字段回落默认，避免某平台改写默认 router 影响其他 cell
<code>HICACHE_HEADS</code>	模块级常量	HiCache 与 UMBP 互斥， <code>umbp</code> 轴需要整族剥离 HiCache 参数，故集中共享 flag 头
<code>forceOff: [{when, stripEnv, reason}]</code>	attention 轴 <code>apply()</code> / <code>render()</code>	命中时强制清空 DP flag 家族并按白名单剥 env；UI 同步置灰并显示 <code>reason</code> ，避免展示一个 apply 会丢弃的取值
<code>roleOverrides + role 级 when</code>	pdDisagg 轴 <code>apply()</code>	同一 role 在不同 operating point 下整体替换 flags/env，不必在 Mode 下拉里堆重复项； <code>when</code> 让 ROCm 专属 sizing 不泄漏到 CUDA
<code>notes: [{when, mode, text}]</code>	pdDisagg 轴 <code>render()</code>	在卡片内提示依赖跨轴事实的前置条件

受影响的下游逻辑：`apply()` 中 DP 分支被改写后，任何携带 `--enable-dp-attention-local-control-broadcast` 的 cell（不限本 PR 新增配方）渲染结果都会变化。

2. 配方声明：三档 PD 角色 (<docs/src/snippets/configs/deepseek-ai/deepseek-v4.jsx>)

- `attention.dpAttn.forceOff`：低延迟与均衡的 PD role 强制 TP-only，并剥离 SGLANG_DP_SHARED_EXPERT_LOCAL、SGLANG_DP_USE_GATHERV、SGLANG_DP_USE_REDUCE_SCATTER。理由写在 `reason` 里——`--max-running-requests` 是 server 级并向下整除 `attn_dp_size`，打开 DP 会让 per-rank 批量低于已捕获的 graph，而命令里两个 flag 都没变。
- `pdDisagg.modes`：prefill/decode 两个 role 带 `when: {hw: ["mi355x"], strategy: ["low-latency"]}`，各自携带 MORI per-rank dispatch 预算（16384 / 128）；prefill 走 `--disable-cuda-graph`，decode 捕获 1-8 的 ladder。
- `pdDisagg.roleOverrides`：balanced 重新描述为 ceiling 96、chunked-prefill 16384、ladder 到 96；high-throughput 继承 TP8/DP8、ceiling 256、mem-fraction 0.92、ladder 截到 32。
- `transferBackends.mori`：补 transport 级 env（SGLANG_MORI_COMBINE_DTYPE、MORI_IO_SQ_BACKOFF_TIMEOUT_US、MORI_IO_QP_MAX_SEND_WR）并限定 `envWhen.hw`；ibDevices 拆成 mlx5 与 rdma3,rdma0,...,rdma5 两族互相 hide。
- `speculative`：mtp-314 从 pro-official 的 hide 列表移除（0813 保留 MTP head，且 DSpark 无法在 PD 下运行），mtp-112 仍隐藏。

- `hicache.roleOverrides`: `balanced` 的 `prefill` 配 `ratio 2.5 / page_first / write-through / best-effort` 预取, 条件比 `modes` 更严 (额外锁定 `variant`、`quant`、`nodes`) 。

3. 部署面板: AMD RDMA 容器参数 ([docs/src/snippets/_deployment.jsx](#))

抽出共享常量 `AMD_RDMA_DOCKER_FLAGS` (`--device /dev/infiniband`、`--cap-add IPC_LOCK`、`memlock/stack/nofile` 上限), 挂到 `MI300X/MI325X/MI350X/MI355X` 四张卡的 `multiNodeDockerFlags`, 让多节点 PD 的 Docker 命令自带 RDMA 直通。

4. Cookbook 正文 ([docs/cookbook/autoregressive/DeepSeek/DeepSeek-V4.mdx](#))

新增 §3.8 讲清三档 `operating point`、角色差异、卸载层级如何区分均衡与高吞吐, 以及 `ceiling / graph ladder` 的算术; §3.9 说明 `UMBP` 是 `HiCache` 的兄弟而非其后端; 同时改写 `MTP/DSpark` 段落, 指出 PD 下 `DSpark` 不可用、应回落 `MTP 3-1-4`。

5. 测试与验证配套

无单元测试 (作者标注 `N/A`, 理由是未触碰运行时)。验证方式为 [docs/scripts/check_cookbook_configs.mjs](#) 加浏览器逐策略 × 角色渲染; 所有新 `cell` 保持 `verified: false`, 等待 `cookbook` 的 "Submit verified cell" 流程确认。

[docs/src/snippets/_playground.jsx](#)

文档站 `Playground` 的共享命令生成引擎, 本 PR 的主要逻辑改动集中在此: 新增 `roleOverrides / forceOff / notes / resolveRouter` 四组扩展点, 并修复 `DP flag` 家族与 `local-control-broadcast` 的连带剥离。它被所有 `cookbook` 复用, 改动会外溢到其他模型的 `cell`。

[docs/src/snippets/_playground.jsx](#) — 角色解析与 DP-Attention 强制关闭

```
// 约束匹配器: 所有带 when 的配置 (routerOverrides / modes / roleOverrides /
// notes / forceOff) 都复用它 —— key 之间是 AND, 单个 key 的取值数组内是 OR。
const matchConstraint = (base, constraint) => {
  if (!constraint || typeof constraint !== "object") return false;
  const entries = Object.entries(constraint);
  if (entries.length === 0) return false;
  return entries.every(([k, vs]) =>
    Array.isArray(vs) && vs.includes(base[k]));
};

// PD 路由器是所有 cell 共享的, 所以想换路由策略的平台声明
// routerOverrides: [{when, port?, command}], 而不是把默认值改掉;
// 首个命中项生效, override 未提供的字段回落到默认 router。
const resolveRouter = (fc, sel) => {
  if (!fc || !fc.router) return null;
  const hit = (fc.routerOverrides || []).find(
    (r) => r && matchConstraint(sel, r.when));
  return hit ? { ...fc.router, ...hit } : fc.router;
};
```

```

// Attention 轴 apply(): forceOff 表示取值由选择本身决定, 用户显式选择与
// base cell 都不再有发言权 —— 用于 PD role 与其派生 cell 拓扑不同的场合。
const dpForced = (knobEntry("dpAttn").forceOff || []).find(
  (r) => r && h.matchConstraint(factsNow(), r.when));
if (dpForced) {
  flags = h.stripFlagsByFirstToken(flags, [
    ...h.DP_HEADS, "--enable-dp-attention",
    "--enable-dp-attention-local-control-broadcast",
  ]);
  // 只服务于 DP 的 env 必须随 flag 一起撤掉, 否则会留在一个已经没有 DP 的命令里。
  const stripEnv = dpForced.stripEnv || [];
  if (stripEnv.length) {
    env = env.filter((e) => !stripEnv.includes(e.split("=")[0]));
  }
} else if (value.dpAttn !== null && value.dpAttn !== undefined
  && !blocked("dpAttn", value.dpAttn)) {
  const dpHead = h.flagSpelling(flags, h.DP_HEADS, "--dp-size");
  // local-control-broadcast 只在 DP attention 打开时有意义, 必须随它下架;
  // apply 每次从 base cell 重新播种, 因此重新调整 DP 大小 (而非关闭) 时会恢复。
  const hadLocalBroadcast =
    h.hasFlag(flags, "--enable-dp-attention-local-control-broadcast");
  flags = h.stripFlagsByFirstToken(flags, [
    ...h.DP_HEADS, "--enable-dp-attention",
    "--enable-dp-attention-local-control-broadcast",
  ]);
  if (typeof value.dpAttn === "number" && value.dpAttn > 0) {
    flags = h.insertAfter(flags, h.ANCHOR_NEAR_TP, [
      `${dpHead} ${value.dpAttn}`,
      "--enable-dp-attention",
      ...(hadLocalBroadcast
        ? ["--enable-dp-attention-local-control-broadcast"] : []),
    ]);
  }
}
}

```

```

// pdDisagg 轴 apply(): role 自带的 when 让它只对一部分选择生效;
// roleOverrides 把同一条 role 在更窄的选择下重新描述, 于是低延迟与高吞吐
// 两种 operating point 不必在 Mode 下拉里各占一项。首个命中项整体替换该
// role 的 flags/env, 此时 role 自己的 when 门控被跳过。
const roleOverride = (fc.roleOverrides || []).find((r) =>
  r && r.mode === mode && r.when && h.matchConstraint(sel, r.when));
const roleSpec = roleOverride || modeMeta;
const modeGate = roleOverride ? null : (modeMeta && modeMeta.when);
const modeOk = !modeGate || Object.keys(modeGate).every(
  (k) => (modeGate[k] || []).includes(sel[k]));
if (modeOk && roleSpec && roleSpec.flags && roleSpec.flags.length) {
  // 先剥同头 flag, 再追加 role 自己的版本 —— 让 role 的取值压过 base cell。
  flags = h.stripFlagsByFirstToken(
    flags, roleSpec.flags.map((f) => f.split(/\s=/)[0]));
}

```

```

    adds.push(...roleSpec.flags);
}

```

```

// render() 里把 forceOff 的结果镜像到 UI: 展示命令真正会取到的值并置灰,
// 而不是留一个看似可点、apply 却会丢弃的选项。
const forced = (knob.forceOff || []).find(
  (r) => r && h.matchConstraint(base, r.when));
renderSelect(forced ? false : knobDisplay(knob), entriesFor(knob),
  (nv) => setKnob(knob.id, nv), base, labelFor(knob),
  { hideValues: hideNullFor(knob),
    disabled: kc.disabled || !!forced,
    disabledReason: forced ? forced.reason : kc.disableReason });

```

docs/src/snippets/configs/deepseek-ai/deepseek-v4.jsx

DeepSeek-V4 的 Playground 配方定义文件，本 PR 的核心交付物：MI355X 低延迟 / 均衡 / 高吞吐三档 PD role 的 sizing、MORI transport env、rdmaN NIC 列表、HiCache 与 UMBP 的层级划分以及 DP-Attention 的 forceOff 约束都在这里声明。

docs/src/snippets/configs/deepseek-ai/deepseek-v4.jsx — 三档 PD 配方与 DP 强约束

```

// Attention 卡：低延迟与均衡两个 PD role 只跑 TP-only。--max-running-requests
// 是 server 级、按 attn_dp_size 向下整除的，只有 dp_size == 1 时才同时等于
// per-rank batch；读者若自行打开 DP，per-rank 批量会掉到已捕获 graph 之下，
// 而命令里两个 flag 都没变。所以这里直接 forceOff（高吞吐本身就是 DP 点，
// 故意不列在 when 里），并把 DP 专属 env 一并清掉。
{ id: "dpAttn", label: "DP-Attention",
  forceOff: [
    { when: { hw: ["mi355x"], strategy: ["low-latency", "balanced"],
      pdMode: ["prefill", "decode"] },
      stripEnv: ["SGLANG_DP_SHARED_EXPERT_LOCAL",
        "SGLANG_DP_USE_GATHERV",
        "SGLANG_DP_USE_REDUCE_SCATTER"],
      reason: "低延迟 / 均衡 PD role 是 TP-only，这正是 --cuda-graph-bs-decode 能等于完整
        ceiling 的前提。" },
    ],
  values: [null, false, 4, 8, {
    value: 16, disable: { nodes: ["single"] },
    disableReason: "DP-Attention=16 需要 16 个 rank，请先把 Deploy 面板切到多节点。" }],
  labels: { "auto": "Auto", "false": "Off" } },

```

```

// PD 卡：role 自带 when，因此 ROCm 专属 sizing 不会泄漏到 CUDA。
// MORI 的 per-rank dispatch 预算按 role 定（prefill 16384 / decode 128），
// 所以它挂在 role 上而非 transport 上；prefill 跑 eager（dsv4 indexer 的
// prefill 路径未做 graph 捕获）并整批派发 chunked-prefill，decode 则捕获
// 小批量 ladder、每步最多派发一个 step 的 token 量。
modes: [
  { id: "prefill", label: "Prefill role",
    when: { hw: ["mi355x"], strategy: ["low-latency"] },

```

```

env: ["SGLANG_MORI_NUM_MAX_DISPATCH_TOKENS_PER_RANK=16384"],
flags: ["--load-balance-method round_robin", "--mem-fraction-static 0.86",
"--max-running-requests 8", "--swa-full-tokens-ratio 0.1",
"--disable-cuda-graph", "--context-length 1048576"] },
{ id: "decode", label: "Decode role",
when: { hw: ["mi355x"], strategy: ["low-latency"] },
env: ["SGLANG_MORI_NUM_MAX_DISPATCH_TOKENS_PER_RANK=128"],
flags: ["--load-balance-method round_robin", "--mem-fraction-static 0.86",
"--max-running-requests 8",
"--cuda-graph-bs-decode 1 2 3 4 5 6 7 8"] },
],

// 均衡档：仍是 TP-only，但 ceiling 提到 96、chunked-prefill 预算 16384。
// TP-only 让 server 级 ceiling 同时就是 per-rank batch，decode ladder 因此
// 能一路铺到 96；高吞吐档的 256 在 DP 8 下每 rank 只得 32，ladder 相应只到 32。
// 下面只保留各 role 相对 base cell 真正变化的 flag，其余条目（tokenizer-worker-num、
// stream-interval、watchdog-timeout、enable-metrics 等）与原配方一致，此处省略。
roleOverrides: [
{ mode: "prefill", when: { hw: ["mi355x"], strategy: ["balanced"] },
env: ["SGLANG_MORI_NUM_MAX_DISPATCH_TOKENS_PER_RANK=16384"],
flags: ["--mem-fraction-static 0.86", "--max-running-requests 96",
"--chunked-prefill-size 16384", "--disable-cuda-graph"] },
{ mode: "decode", when: { hw: ["mi355x"], strategy: ["balanced"] },
env: ["SGLANG_MORI_NUM_MAX_DISPATCH_TOKENS_PER_RANK=128"],
flags: ["--mem-fraction-static 0.86", "--max-running-requests 96",
// 实际为 1..96 的完整 ladder，这里省略中间项。
"--cuda-graph-bs-decode 1 2 ... 96"] },
],

```

```

// 均衡档的卸载层级：prefill 挂 HiCache 主机层（ratio 2.5、page_first、
// write-through、best-effort 预取）；decode 不挂，因为 MORI 下 HiCache
// 在 decode role 上不被推荐。高吞吐档改用 UMBP，二者互斥。
// 该 override 的条件比 modes 更严：额外锁定 variant / quant / nodes。
{
mode: "prefill",
when: { hw: ["mi355x"], variant: ["pro"], quant: ["fp4"],
strategy: ["balanced"], nodes: ["single"] },
memLayout: "page_first",
ioBackend: "direct",
ratio: 2.5,
writePolicy: "write_through",
prefetchPolicy: "best_effort",
}

```

评论区精华

该 PR 的评审记录非常干净：0 条 review comment、0 条普通 comment，只有 reviewer HaiShaw 的一条空 body APPROVED。因此有信息量的讨论全部来自 PR body 的自我论证，需要按原话理解：

"Disaggregation is not one operating point, so each serving strategy gets its own role sizing rather than a single 'PD on' toggle."

"The decode graph ladders differ because `--max-running-requests` is server-wide and floor-divided by `attn_dp_size`"

"Left to the reader, switching DP on would cut the per-rank batch below the captured graphs without changing either flag in the command, so the roles force DP-Attention off and gray the control."

"Turning DP-Attention off stripped `--dp` and `--enable-dp-attention` but left `--enable-dp-attention-local-control-broadcast` orphaned in the command. ... This affected any cell carrying that flag, not just the new recipes."

可以看到，本 PR 最重要的设计权衡（role 为何强制 DP 关闭、高吞吐为何被排除在 forceOff 之外、roleOverrides 为何优于下拉重复项）由作者单方面给出并直接获批，缺少第三方对 " 把业务约束编码进文档生成器 " 这一做法的挑战。

风险与影响

风险	具体位置	说明
共享生成器连带变更	<code>_playground.jsx</code> DP 分支	该文件被所有 cookbook 复用；孤儿 flag 的修复作者自述会影响任何携带 <code>--enable-dp-attention-local-control-broadcast</code> 的 cell，而 PR 无自动回归手段
恢复逻辑的隐式耦合	<code>hadLocalBroadcast</code>	恢复依赖 "apply 每次从 base cell 重新播种 " 这一契约，未来若引入缓存或增量渲染会静默失效
env 去重粒度偏粗	pdDisagg 轴 env 合并	role env 以 <code>!env.includes(e)</code> 精确字符串去重，同 key 不同值可能生成两条； <code>stripEnv</code> 是白名单，未列入的 DP env 会残留
门控粒度不一致	<code>modes / roleOverrides</code> 的 <code>when</code>	balanced/high-throughput 的 <code>when</code> 仅 <code>hw + strategy</code> ，而 hicache 的 <code>when</code> 用了五个维度，前者更宽松
配方未验证	<code>DeepSeek-V4.md</code> §3.8	所有新 cell <code>verified: false</code> ，PR body 记录 Extra CI（Run #34826174449）失败；若 sizing 与实机漂移会误导用户
运行时无影响	全量 diff	未改动 kernel、模型或运行时文件，服务行为不变

影响范围：MI355X 运维者可直接从 Playground 复制含 MORI-IO、rdmaN NIC 列表与容器 fabric flags 的完整 PD 命令；文档站所有 cookbook 共享 `_playground.jsx`，因此新扩展点与 DP 剥离修复会作用于其他模型的 cell；团队侧需要 AMD/DeepSeek 维护者在 flag 或拓扑变化时同步 `when` 条件，并补齐三档配方的端到端验证。

关联脉络

- 同硬件文档线：#39358 把 Qwen3.5 MI355X cookbook 与 HiCache page_first_direct 对齐，与本 PR 的 page_first / write-through 卸载配置共用同一套约定；#39360、#39172、#39356 则显示 MI355X/ROCm 侧的注意力后端与 CI 仍在快速迭代，本 PR 的配方需要跟随其变化。
- 同模型运行时线：#37413 在 gfx950 上启用 FP8 双池统一 KV，#37382 为 DSV4 补齐主机内存缓存管理，本 PR 的 HiCache/UMBP 层级划分正是这些运行时能力在文档侧的出口。
- PD 与统一内存线：#37506 打通统一内存池全形态 PD 传输，#37418 启用 prefill CUDA Graph 捕获；本 PR §3.9 关于 "UMBP 是 HiCache 的兄弟而非其后端" 的表述，以及 prefill role 显式 --disable-cuda-graph 的理由，都直接引用这条演进线。
- 从更宏观的角度看，这一系列 PR 反映出一个趋势：cookbook 的 Playground 正在从 "参数展示" 演变为 "约束执行器"，forceOff、roleOverrides、notes 这类扩展点把原本写在正文里的部署经验固化成可校验的配置，这是文档质量收敛的积极信号，但也意味着文档生成的命令开始需要测试来兜底。