FLINK FORWARD # ASIA 2022 ONLINE | 实时即未来

Flink OLAP Improvement of Resource Management and Runtime

曹帝胄 | 字节跳动基础架构工程师

01 Flink OLAP in ByteDance

02 作业调度

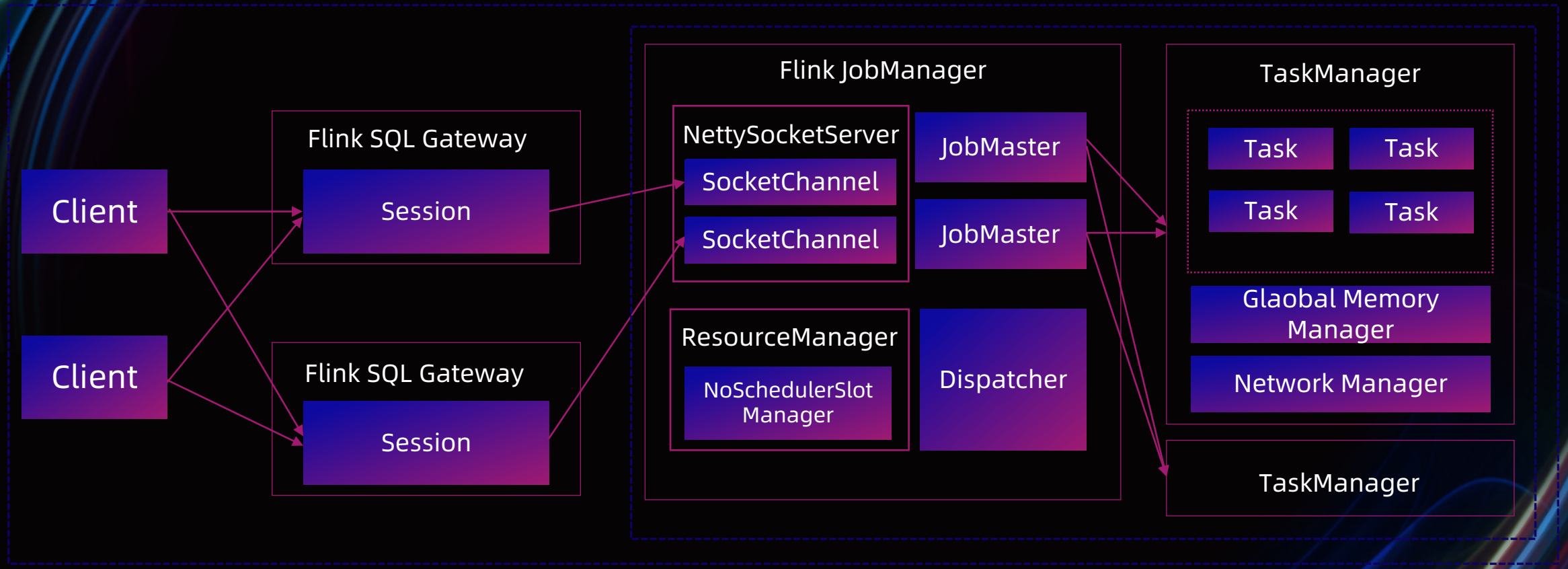
03 Runtime 执行

04 收益

05 未来规划

01 Flink OLAP in ByteDance

Flink OLAP Architecture in ByteDance



Flink OLAP in ByteDance

20+ HTAP 线上集群

16,000+ Cores

Query 50W+/天

挑战

QPS

Lantency

调度 Benchmark

测试用例

- 根据复杂度设计三组测试
- Flink 默认并发度128，数据量极小

Join

```
select val1, count(*) from table1
left join table2 on val1=val2
left join table3 on val2=val3
group by val1
```

WordCount

```
select val1, count(*) from table1
group by val1
```

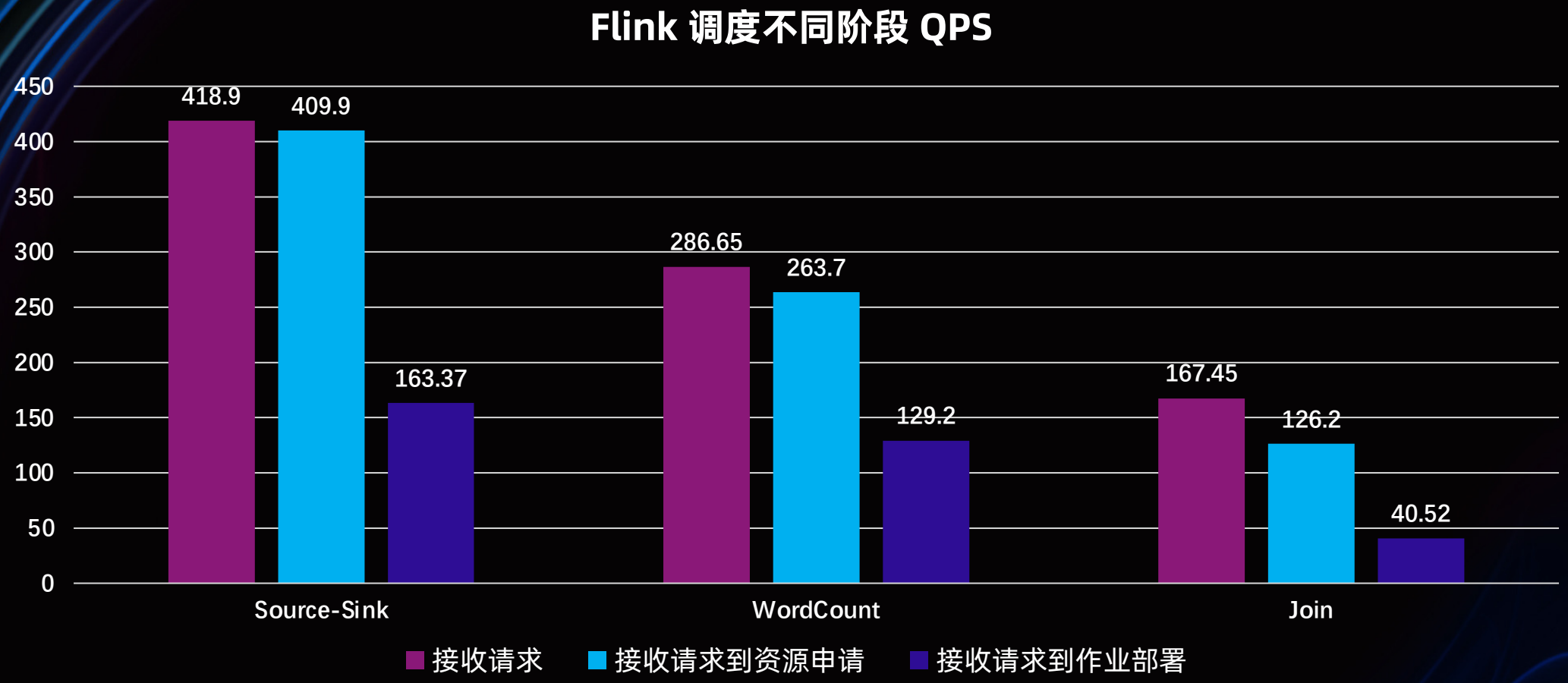
Source-Sink

```
select val1 from table1;
```

测试环境

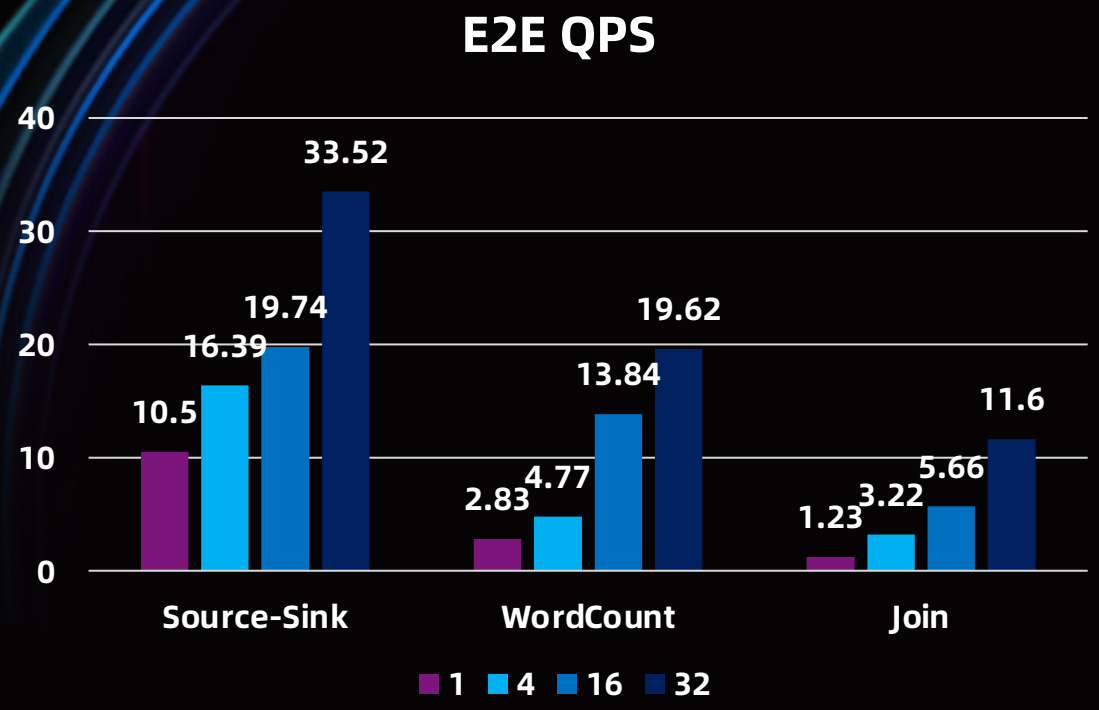
- 5台物理机，共 500 Cores，1.25W 个 Slot
- 统计10分钟内完成的作业及作业平均 Latency

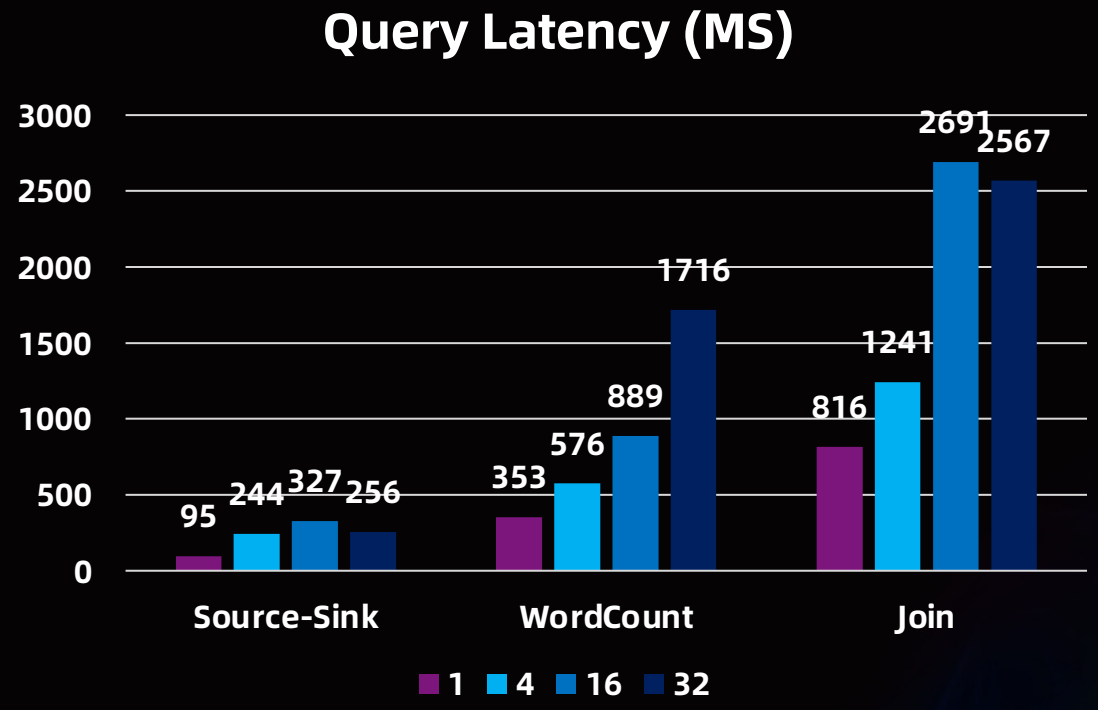
问题分析



部署阶段性能下降明显

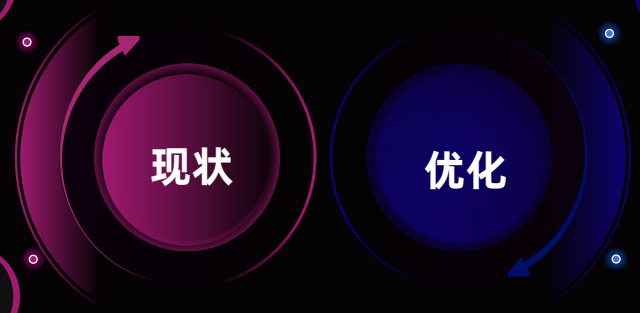
问题分析





高 QPS 下，Latency 上涨严重

现状和优化



作业资源管理和部署瓶颈

作业资源管理流程复杂

作业部署交互瓶颈

任务运行延迟瓶颈

结果拉取瓶颈

作业运行时资源创建

高并发下锁抢占问题

作业调度优化

作业资源管理流程优化

任务部署结构优化

模块交互优化

任务运行时优化

结果拉取优化

Task 运行优化

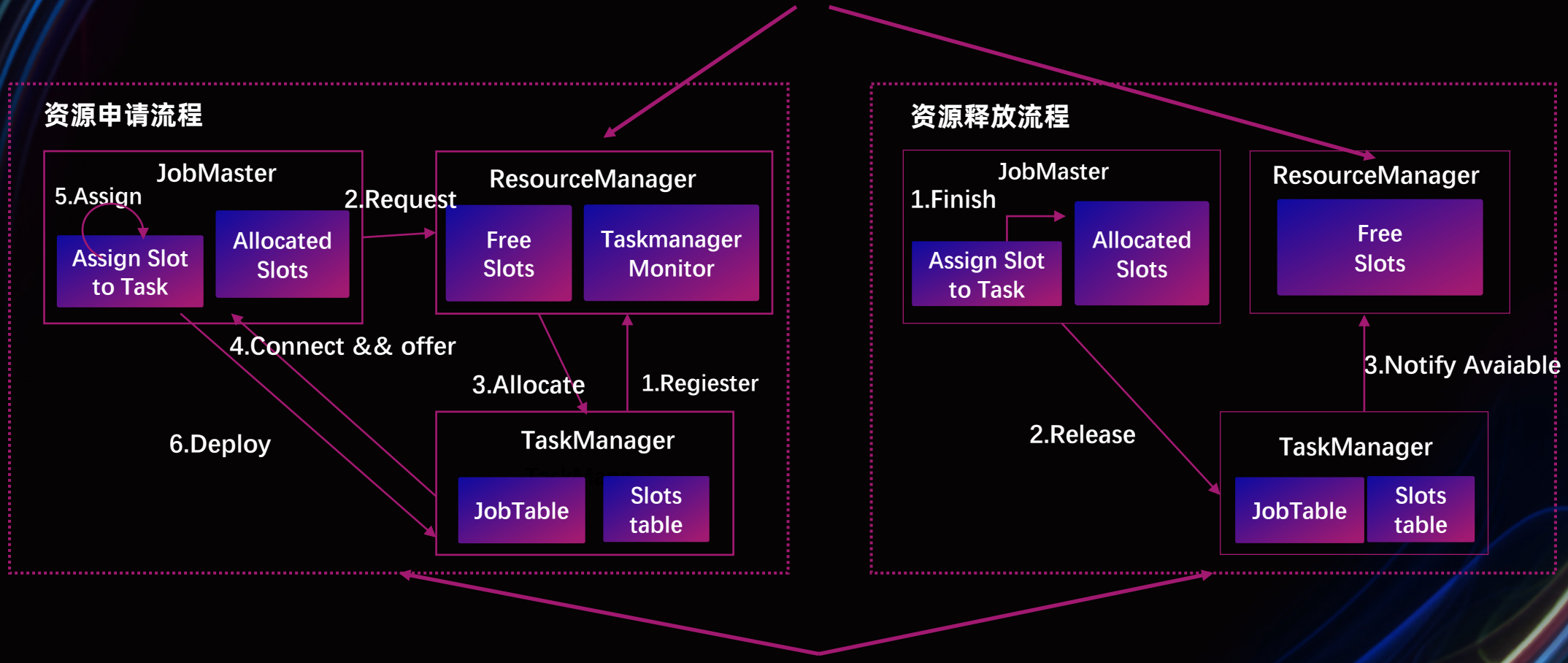
02 作业调度

1. 资源管理流程优化
2. 任务部署结构优化
3. 模块交互优化

资源管理流程优化

现状

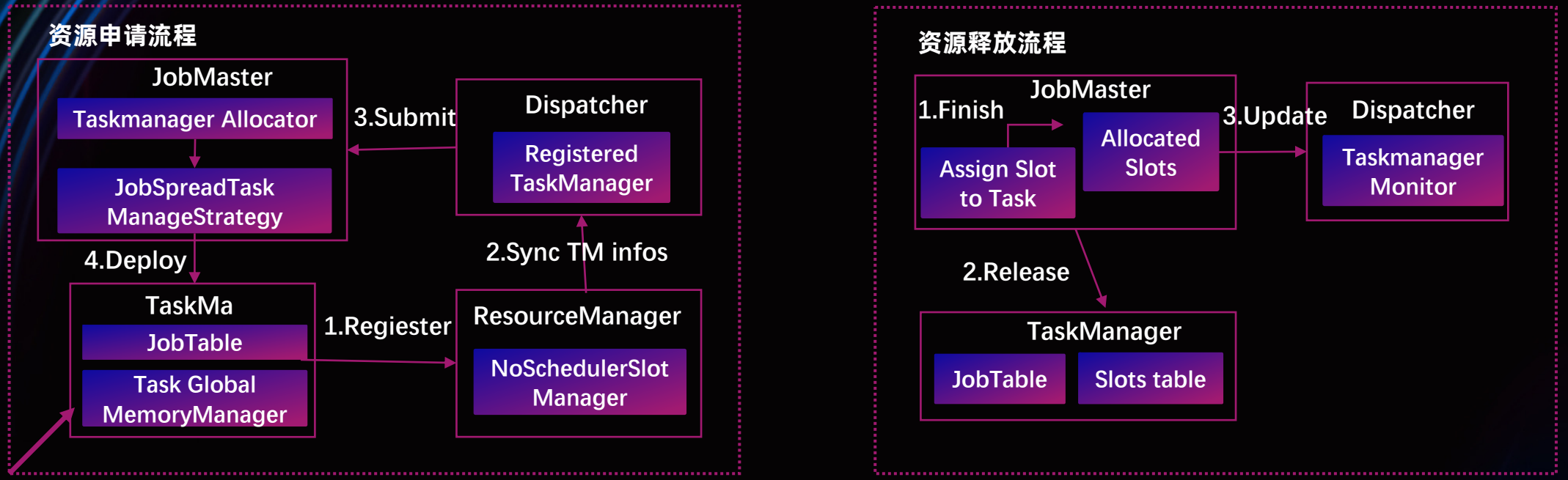
每次资源申请释放都需要经过 ResourceManager，形成单点瓶颈



资源申请释放需要经过多次交互

资源管理流程优化

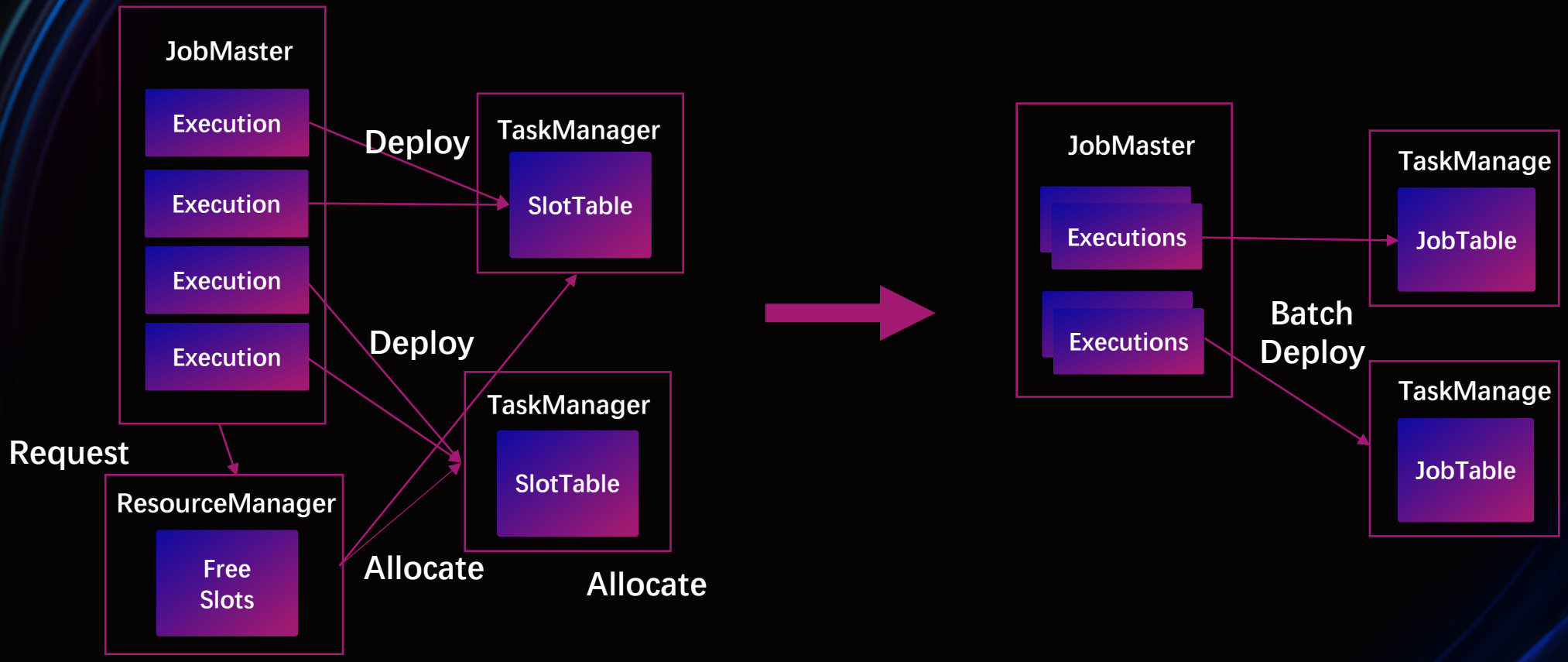
优化



Source-sink 调度部署阶段 QPS 测试

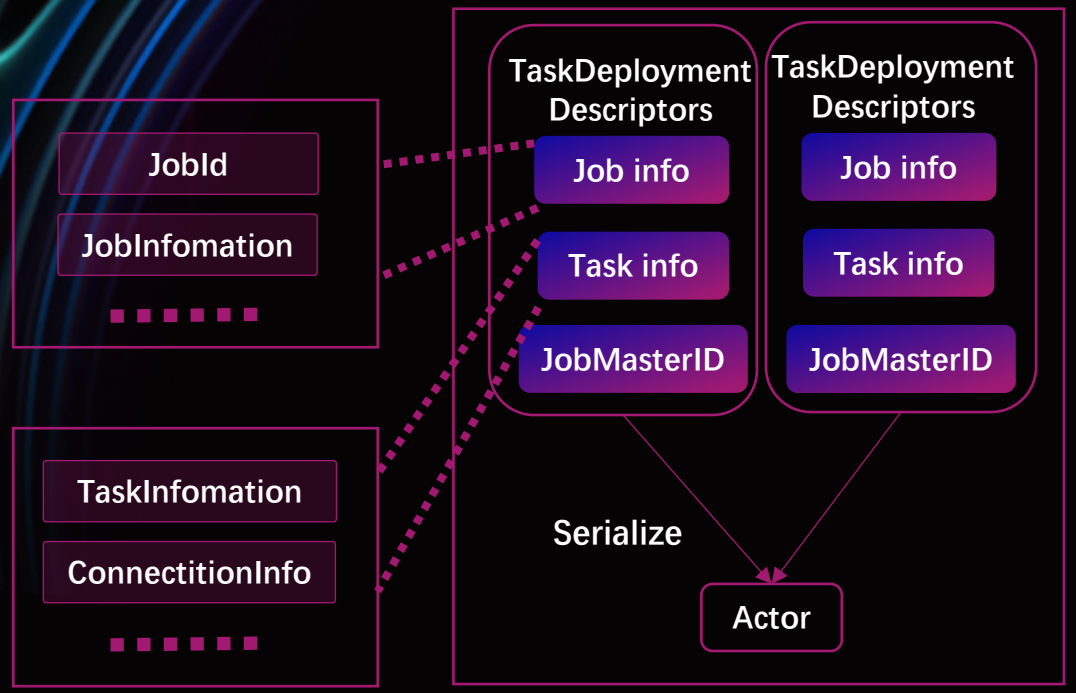
	Client 提交线程数	QPS
优化前	32	160.42
优化后	32	359.67

按 TM 维度作业部署

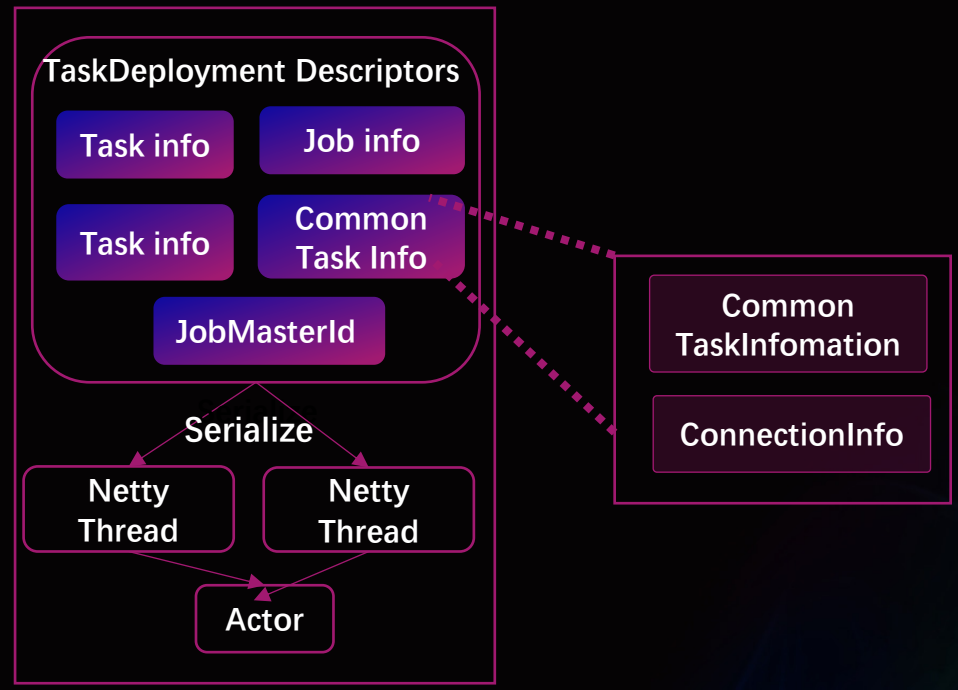


任务部署结构优化

现状



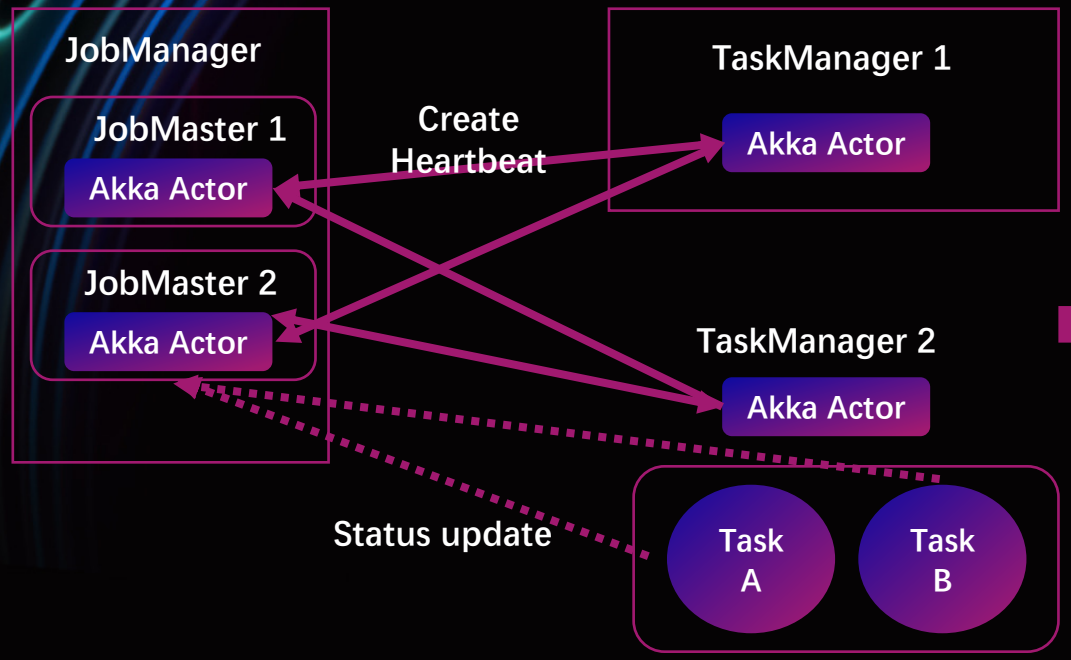
优化



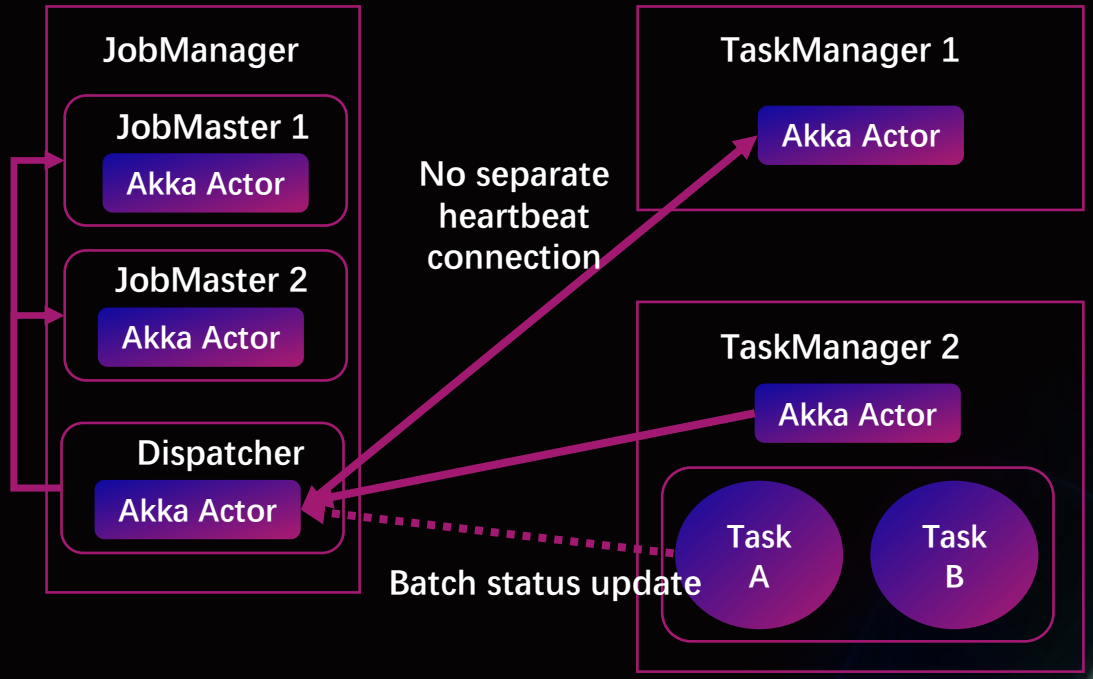
作业	计算任务数量	优化前		优化后	
		序列化后数据大小（KB）	序列化总耗时（ms）	序列化后数据大小（KB）	序列化总耗时（ms）
Source	13	63.1	5462	5.6	644
WordCount	26	317.3	122546	11.1	940
Join	30	557.5	219189	28.3	2830

模块交互优化

现状



优化



以 Task 数为128的测试作业在 QPS 100, TM 100的环境下为例

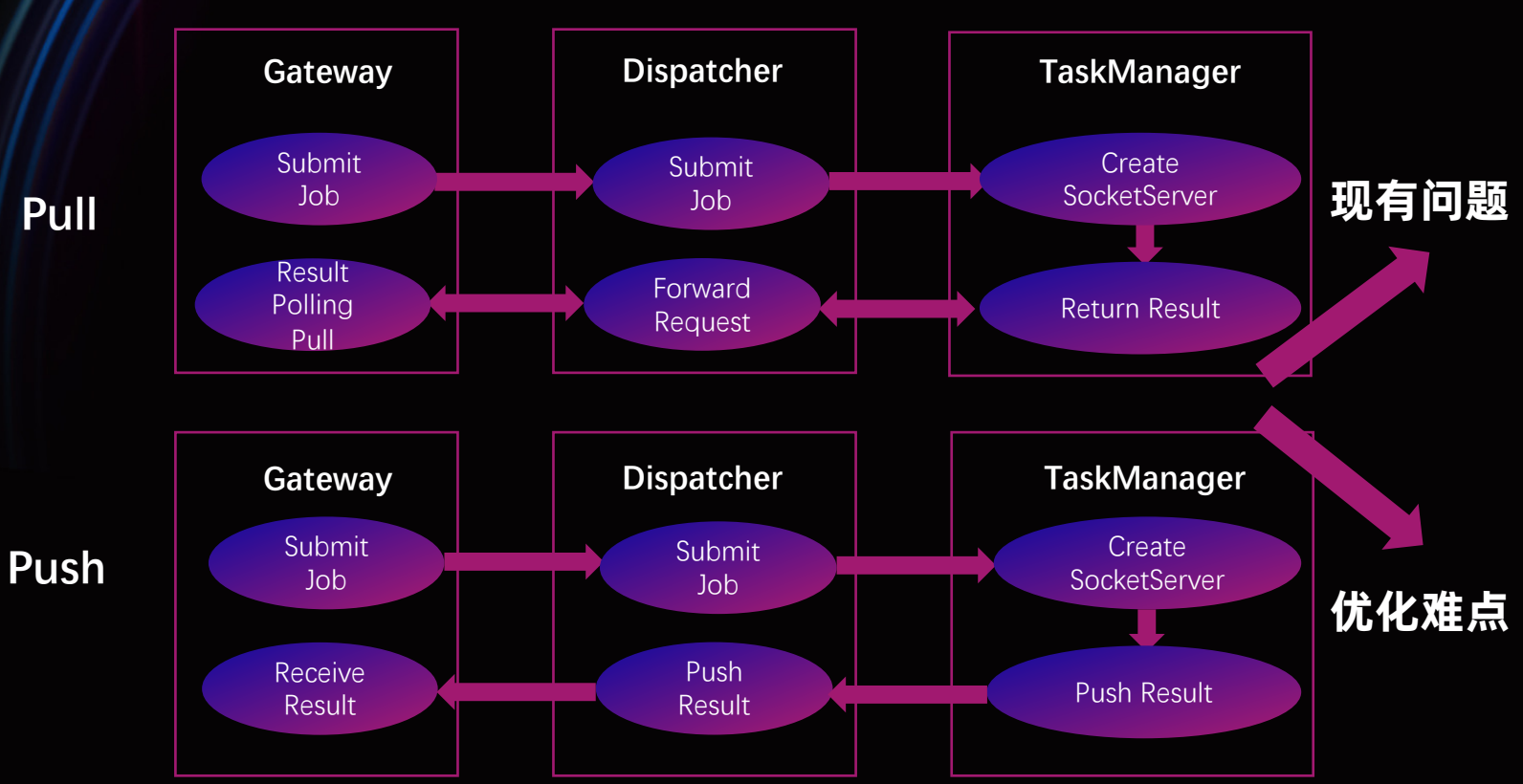
优化前		优化后	
连接连接/销毁 QPS	Task 任务更新请求数	连接连接/销毁 QPS	Task 任务更新请求数
10,000	2,560,000	0	25,600

03 Runtime 优化

1. 作业结果拉取优化
2. Task 运行优化

作业结果拉取优化

Pull VS Push



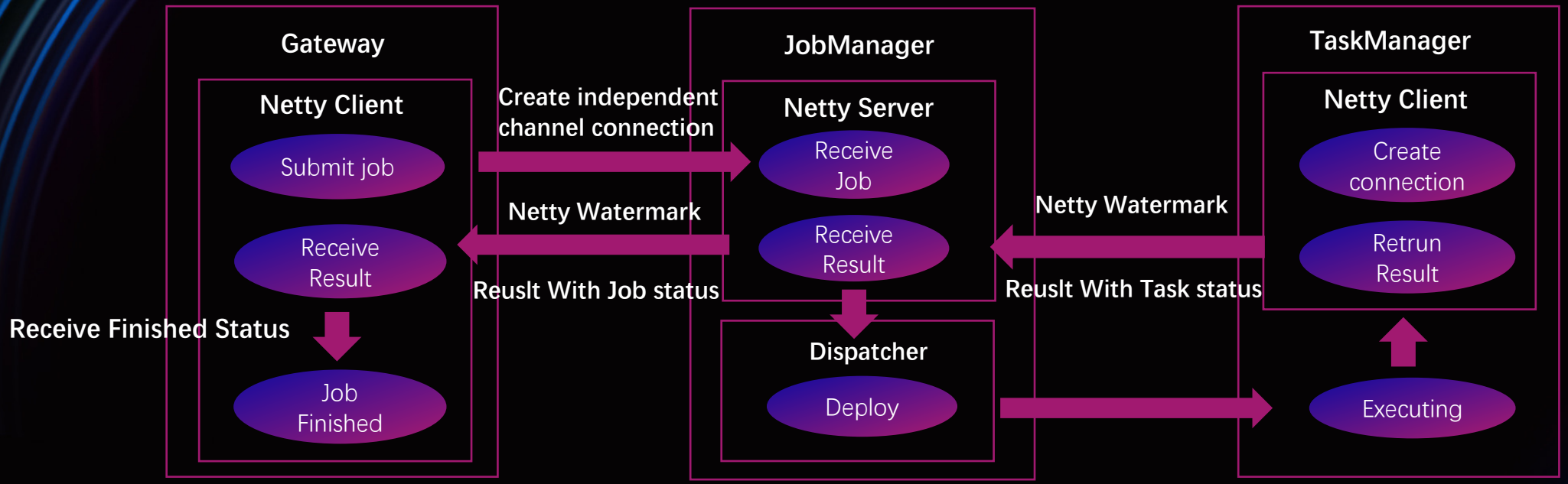
Pull 模式

- 存在轮询时间，增加了 Lateancy
- 需要创建临时资源（网络、线程等）
- Dispatcher 存在单 Actor 线程瓶颈

Push 模式

- 返回结果的流量控制
- 作业结束状态处理

作业结果 Push 架构

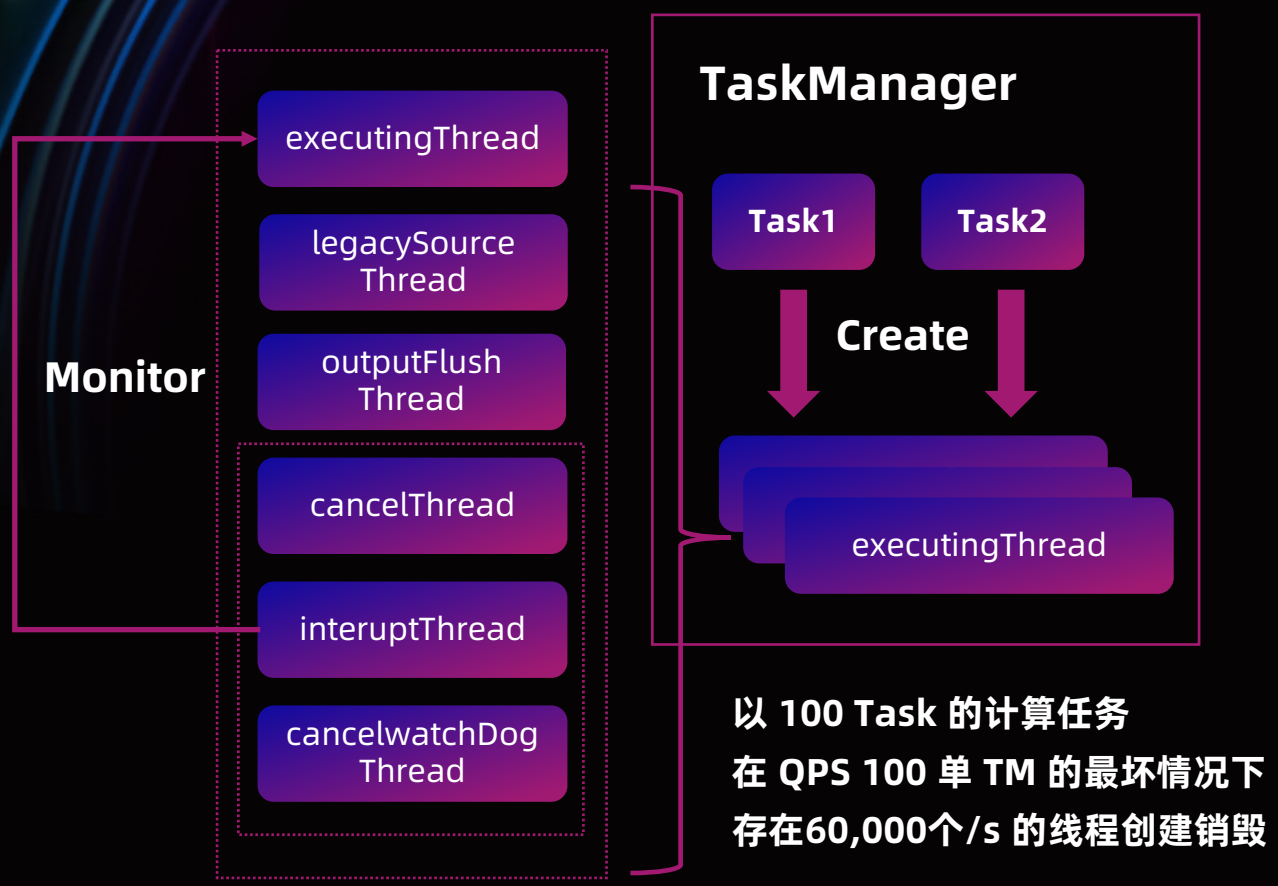


50TM 下空数据 Query QPS 测试

提交协议	Client 提交线程数	查询完成 QPS
Http Rest (pull)	128	850
Netty Socket (push)	128	4096

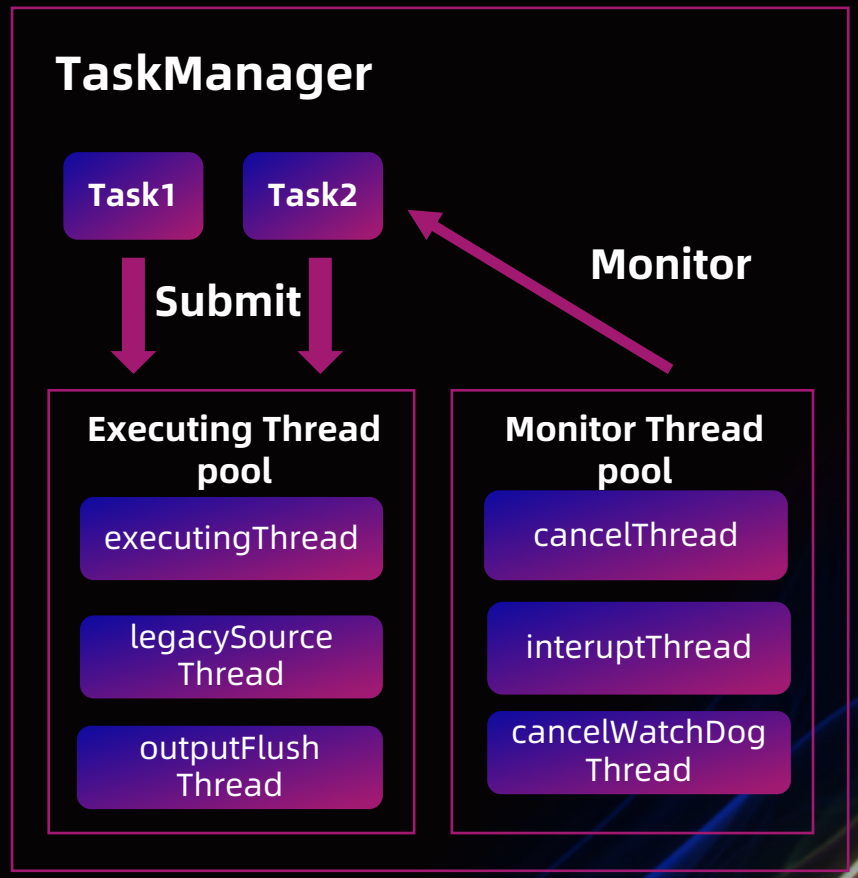
线程优化

现状



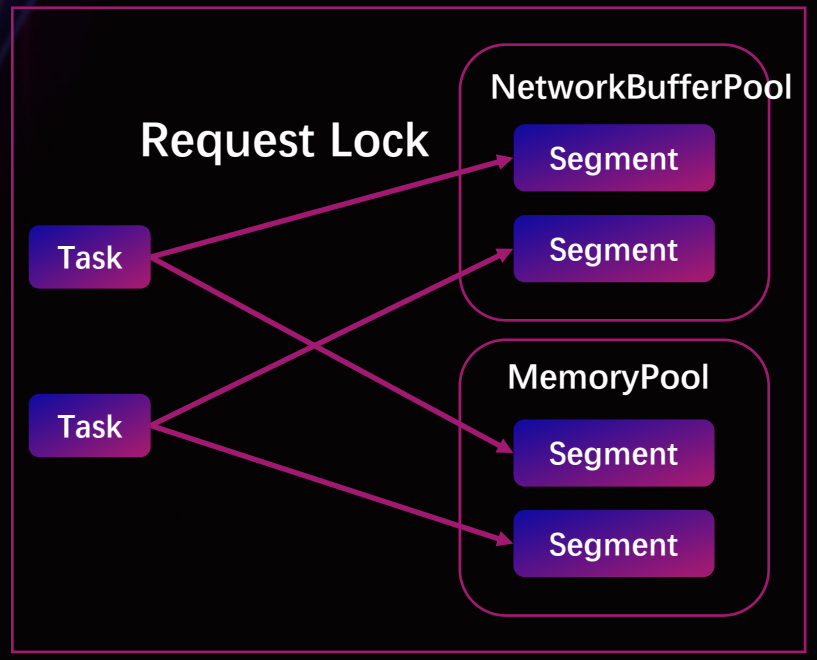
以 100 Task 的计算任务
在 QPS 100 单 TM 的最坏情况下
存在60,000个/s 的线程创建销毁

优化和实现



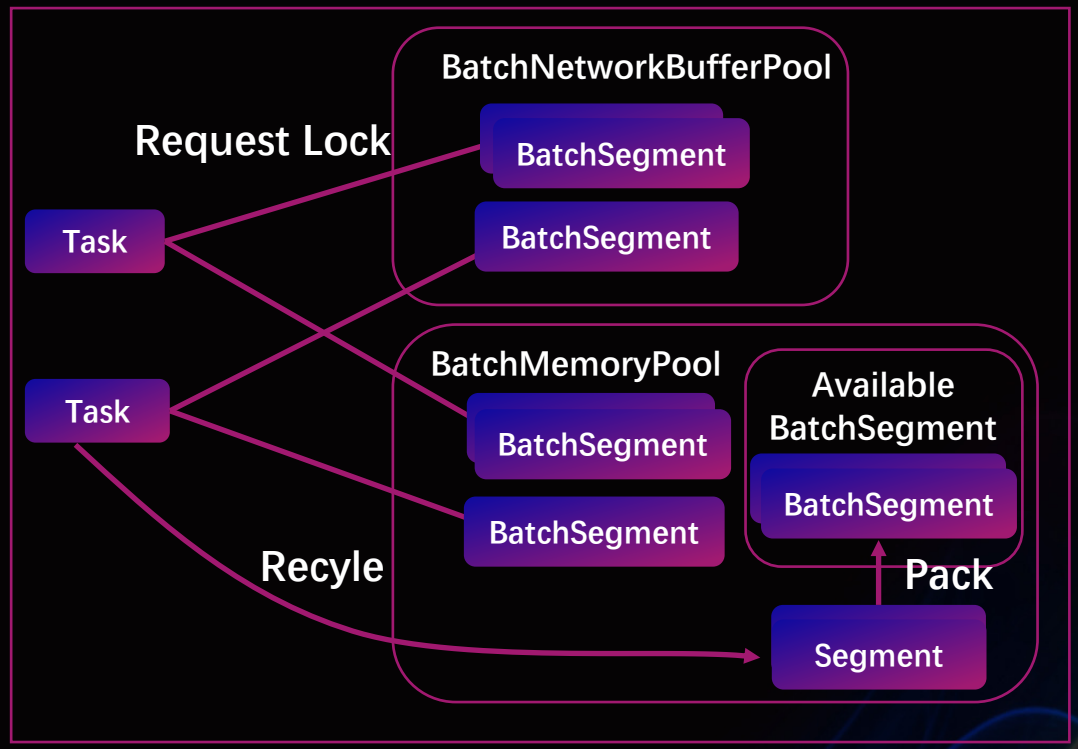
锁优化

现状



10并发, Task100 ， TM 为1，batchSegment size 为10
申请 segment 的锁占用请求由1000/s -> 100/s
模拟申请10w个 segment 总耗时由4353ms -> 793 ms

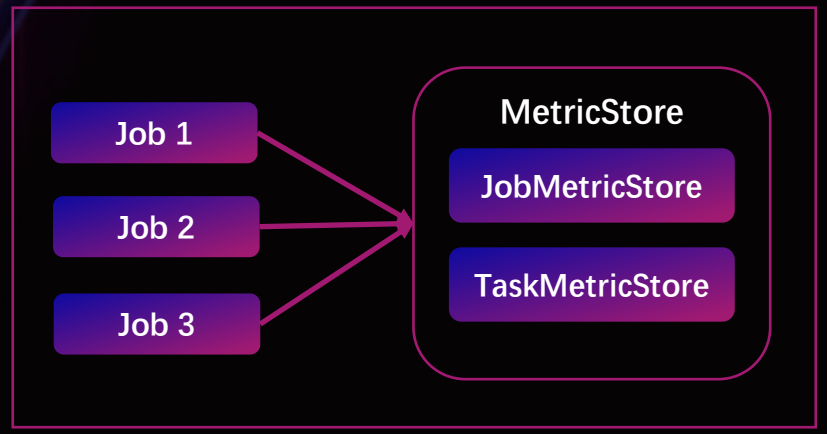
优化



最大 segment 碎片 < BatchSegment size

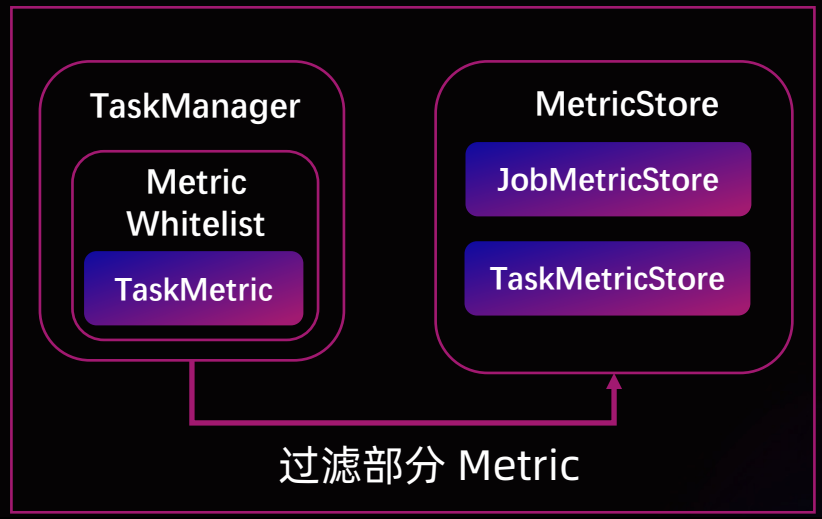
内存优化

现状



Name	Count	Size
char[]	25,692,152 (38.4%)	1,671,033,238 B (48.2
java.lang.String	25,690,669 (38.4%)	719,338,732 B (20.8%)
java.util.concurrent.ConcurrentHashMap$Node	13,288,984 (19.8%)	584,715,296 B (16.9%)
java.util.concurrent.ConcurrentHashMap$Node#1		44 B (0%)
<fields>		
next = null		-
val = java.lang.String#135230 : 0.0		28 B (0%)
key = java.lang.String#1 : Shuffle.Netty.Output.numBuff		28 B (0%)
hash = int 2034987183		-
static <classLoader> = null		-
<references>		
[47] in java.util.concurrent.ConcurrentHashMap$Node		1,048 B (0%)
table in java.util.concurrent.ConcurrentHashMap#1		100 B (0%)
metrics in org.apache.flink.runtime.rest.handler.le		24 B (0%)

优化



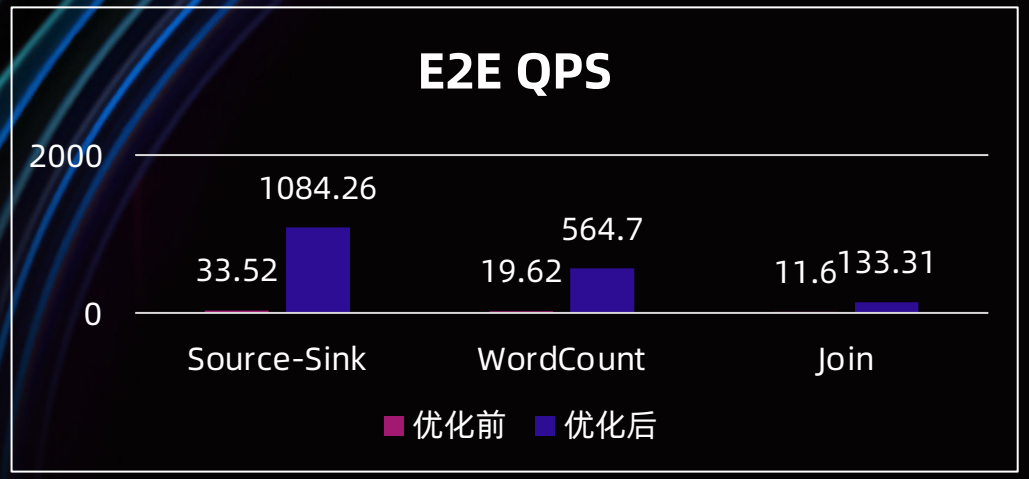
num	#instances	#bytes	class name
1:	7262797	1032501800	[C
2:	5368079	995963816	[Ljava.lang.Object;
3:	691648	842363656	[B
4:	3155236	227176992	java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask
5:	509466	205810696	[I
6:	1919162	168532912	[J
7:	1960498	160478536	[Ljava.util.HashMap$Node;
8:	6303200	151276800	java.lang.String
9:	2538445	121845360	java.util.HashMap
10:	2767442	88558144	java.util.HashMap$Node
11:	3505117	84122808	java.util.ArrayList
12:	1510617	72509616	org.apache.flink.runtime.executiongraph.IOMetrics
13:	1189876	66633056	org.apache.flink.runtime.executiongraph.ArchivedExecution
14:	1915568	61298176	org.apache.flink.runtime.executiongraph.ExecutionAttemptID
15:	1189876	57114048	org.apache.flink.runtime.executiongraph.ArchivedExecutionVertex
16:	1204656	48186240	org.apache.flink.runtime.util.EvictingBoundedList
17:	654457	41885248	java.util.concurrent.ConcurrentHashMap

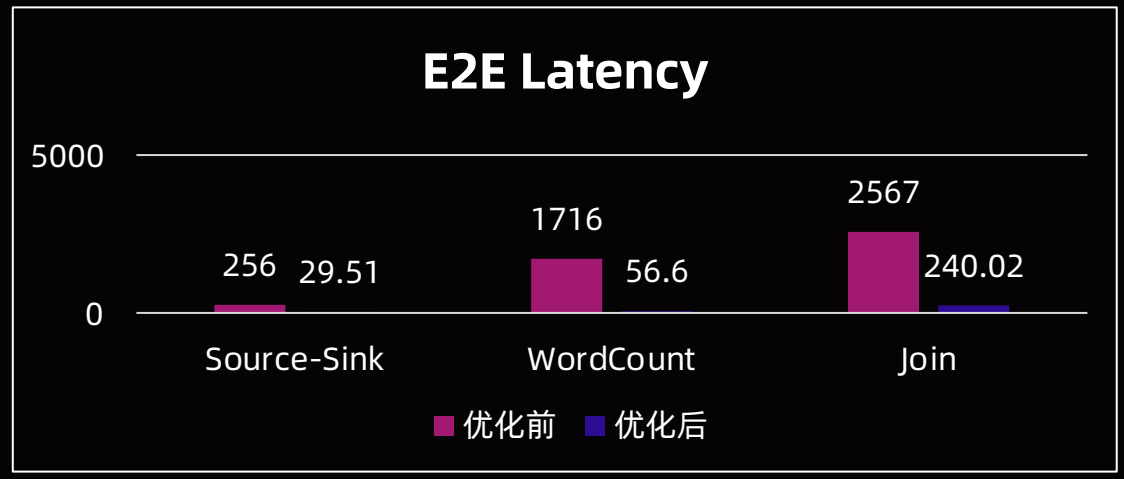
Full GC 频率降低88%

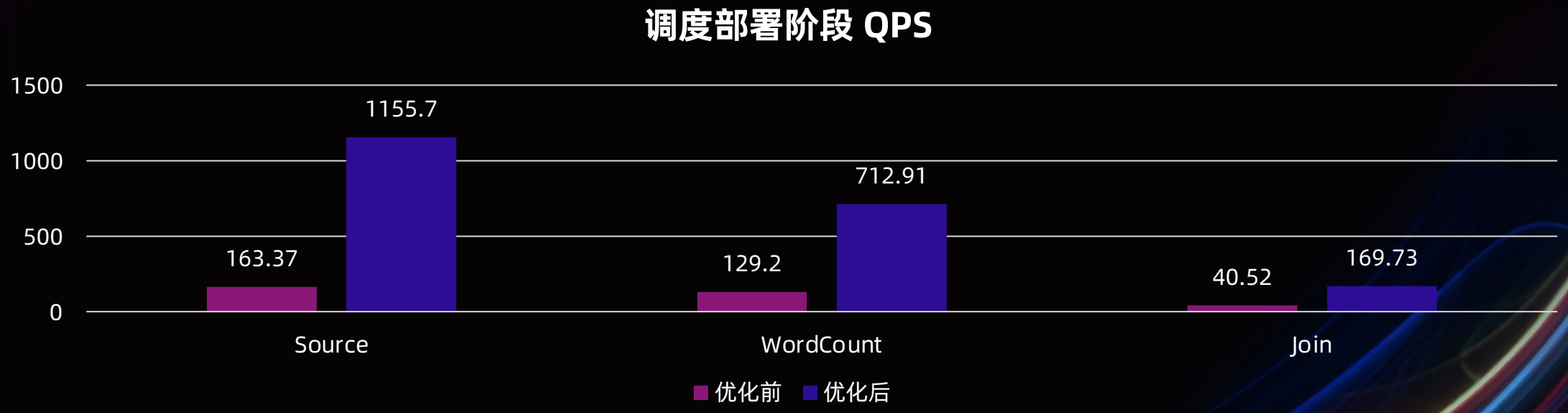
04 收益

1. Benchmark
2. 业务收益

Benchmark


E2E QPS
2000
0
33.52
1084.26
19.62
564.7
11.6
133.31
Source-Sink
WordCount
Join
优化前
优化后


E2E Latency
5000
0
256
29.51
1716
56.6
2567
240.02
Source-Sink
WordCount
Join
优化前
优化后


调度部署阶段 QPS
1500
1000
500
0
163.37
1155.7
129.2
712.91
40.52
169.73
Source
WordCount
Join
优化前
优化后

业务收益

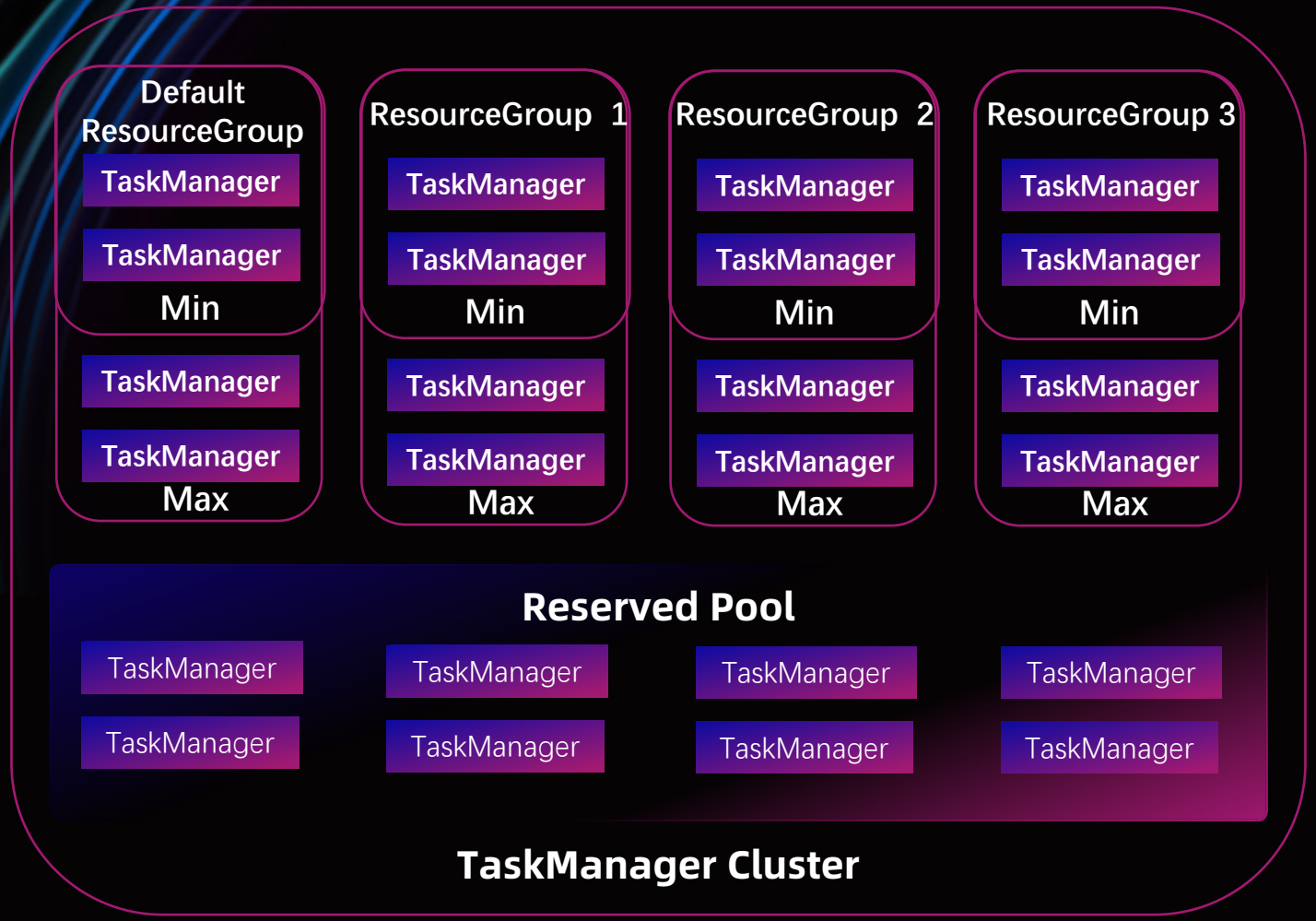
业务接入 12+
User Growth、电商、
幸福里、飞书等核心业务

单集群复杂查询
QPS > 200
Latency P99 <5s

05 未来规划

1. Serveless 能力建设
2. 性能提升

Serveless 能力建设



- 集群维度支持资源组隔离
- 集群支持弹性扩缩容
- 单节点维度支持计算任务分优先级调度

性能提升

集群负载性能

1. JobManager 支持水平扩展
2. 集群资源利用率优化
3. Task 调度性能优化

运行时优化

1. 运行时网络消息优化
2. 资源申请流程优化

冷启动优化

1. Gateway 冷启动优化
2. 网络初始化优化
3. 内存申请流程优化

FLINK FORWARD # ASIA 2022 ONLINE | 实时即未来

THANK YOU

谢　谢　观　看