

39. Distributed Publish/Subscribe Query Processing on the Spatio-Textual Data Stream

姚婉薇-51184501174 刘骞-51184501130

Content

- 本文主要目标
- 分布式发布订阅系统
- 工作负载分区策略
- 动态负荷调整方法
- 实现结果
- 总结

背景

A) 什么是空间文本数据流上的发布订阅系统？

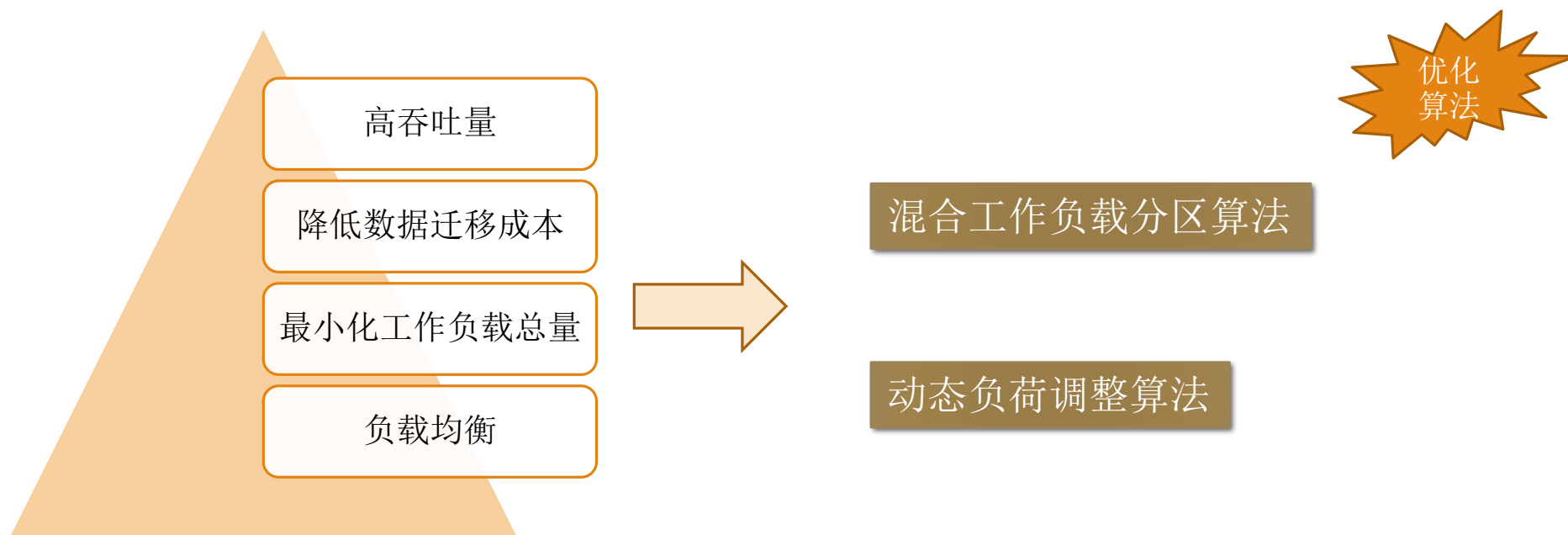
通过允许用户注册具有空间和文本约束的连续查询，实现高效和有效的信息分发。

B) 为什么使用分布式？

计算工作量增加，单个服务器的容量满足不了空间文本信息和注册查询的增长，所以需要使用分布式解决方案。

本文主要目标

在空间文本数据流上构建分布式的发布订阅系统



Content

- 本文主要目标
- 分布式发布订阅系统
- 工作负载分区策略
- 动态负荷调整方法
- 实现结果
- 总结

定义

Spatio-Textual Object（空间文本对象）：

$$O = \langle \text{text}, \text{loc} \rangle$$

$O.\text{text}$ ：对象的文本内容， $O.\text{loc}$ ：对象的位置（包括经纬度）。

Spatio-Textual Subscription (STS) Query（空间文本订阅查询）：

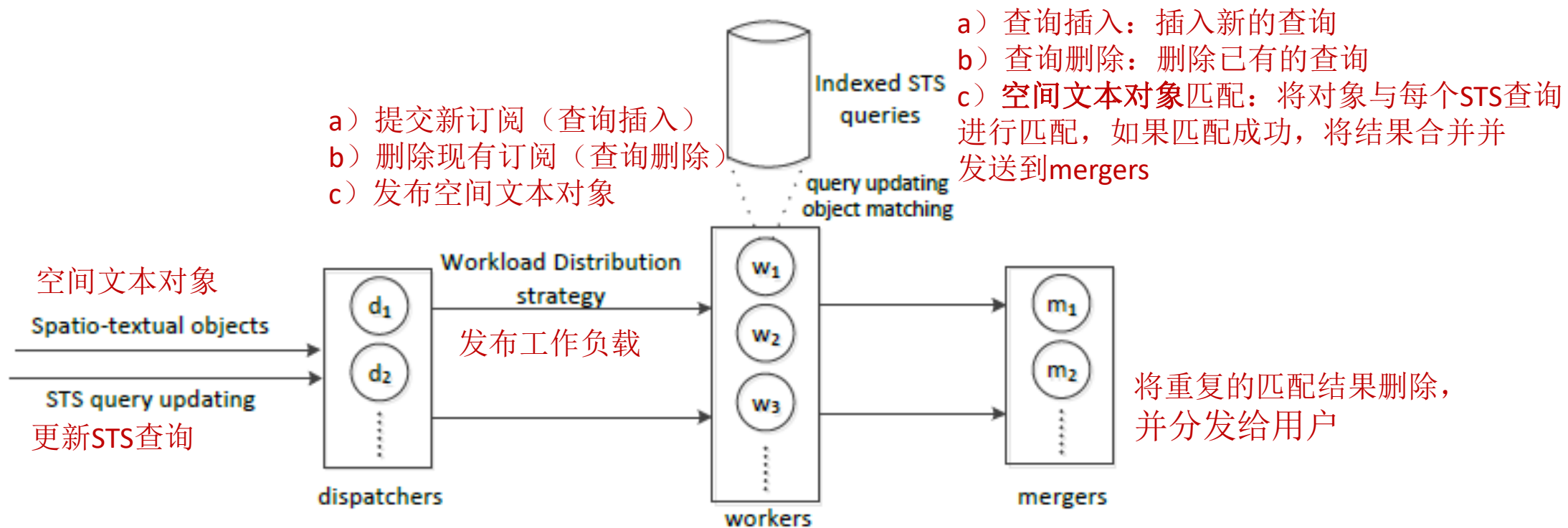
$$Q = \langle k, r \rangle$$

$Q.k$ ：关键字序列， $Q.r$ ：区域范围。

Spatio-Textual Object Matching（空间文本对象匹配）：

如果 $O.\text{text} \models Q.k \wedge O.\text{loc} \in Q.r$ 为真，则称 O 与STS查询 Q 匹配成功。

PS²Stream系统架构



优化算法

A) 工作负载分区策略

目标:

最小化工作负载总量和
工作负载均衡

方法:

结合数据的**文本**属性和
空间属性分配工作负载

B) 动态负荷调整方法

意义:

数据流的动态变化导致
工作负载变化, **Worker**
的负载也会改变

目标:

减少总工作负载, 使得
调用和迁移成本降低。

Content

- 本文主要目标
- 分布式发布订阅系统
- 工作负载分区策略
- 动态负荷调整方法
- 实现结果
- 总结


工作负载分区策略

- 目标

- 最小化工作负载总量
- 工作负载均衡

- 优点

- 结合数据的文本属性和空间属性分配工作负载



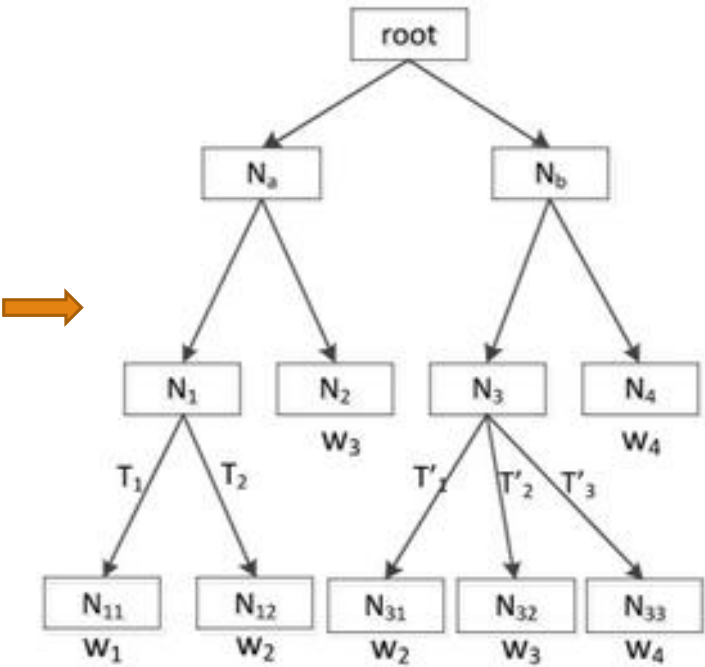
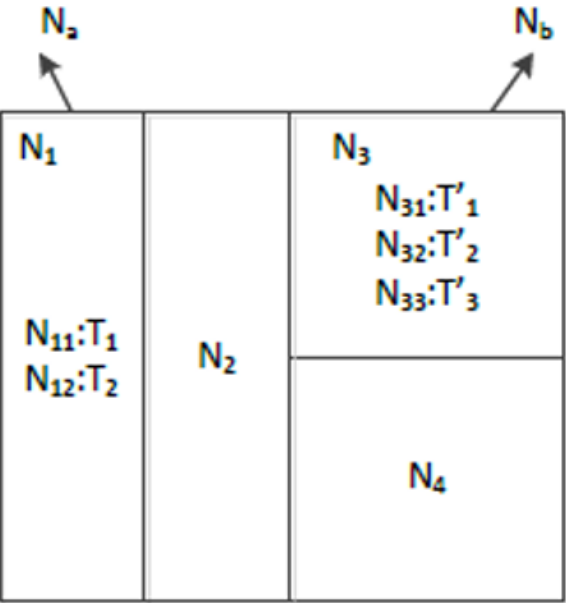
混合分区

混合分区原理及Dispatchers的索引结构

worker: w1 w2 w3 w4

N_i : 根据O与Q的文本相似度分区

T: 关键字



g1 T1:w1 T2:w2	g3 w3	g5 T' ₁ :w2 T' ₂ :w3 T' ₃ :w4	g7 W4
g2 T1:w1 T2:w2	g4 w3	g6 T' ₁ :w2 T' ₂ :w3 T' ₃ :w4	g8 W4

H1

H2

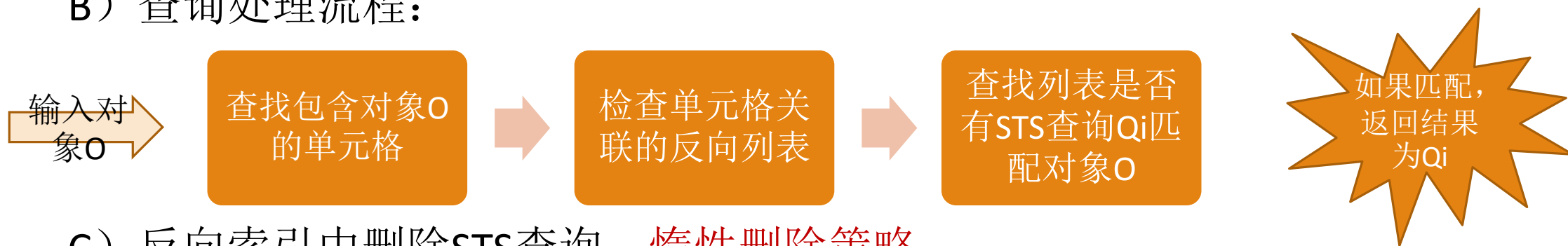
Dispatchers的索引结构

Workers的查询处理

A) 网格反向索引GI2:

STS (And): 最不频繁关键字的反向列表 STS (Or): 最不频繁的关键字的倒排列表

B) 查询处理流程:



C) 反向索引中删除STS查询, 惰性删除策略。

把删除的查询的id记录到哈希表中, 并在对象匹配过程中删除查询。

Content

- 本文主要目标
- 分布式发布订阅系统
- 工作负载分区策略
- 动态负荷调整方法
- 实现结果
- 总结

动态负荷调整方法

A) 局部

- 当dispatcher检测到**负载均衡约束被违反**，立即通知负载最大的worker把工作负载的一部分转换到负载最小的worker。

B) 全局

- **定期检查**最近的数据样本是否需要工作负载**重新分区**。

局部动态负荷调整方法

- 第一阶段，我们检查 w_o 和 w_l 中的一些单元是否可以拆分或合并，以减少总工作量，如果存在这样的单元，我们会进行相关的迁移操作。
- 第二阶段，如果仍然违反负载均衡约束，我们继续选择 w_o 中的一组单元迁移到 w_l ，最小化迁移成本，使得系统满足负载均衡约束。

全局动态负荷调整方法

当数据分布发生重大变化时，当前工作负载分区策略的性能将会下降。

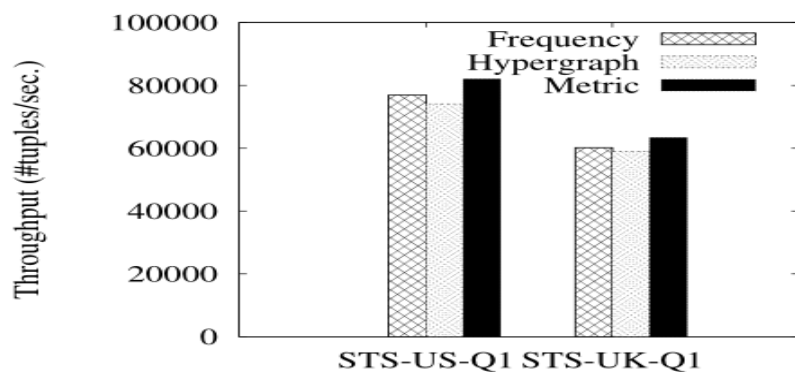
我们采用定期检查最近的数据样本是否需要工作负载重新分区的方法，如果需要重新分区，我们使用之前提出的算法进行工作负载重新划分。

Content

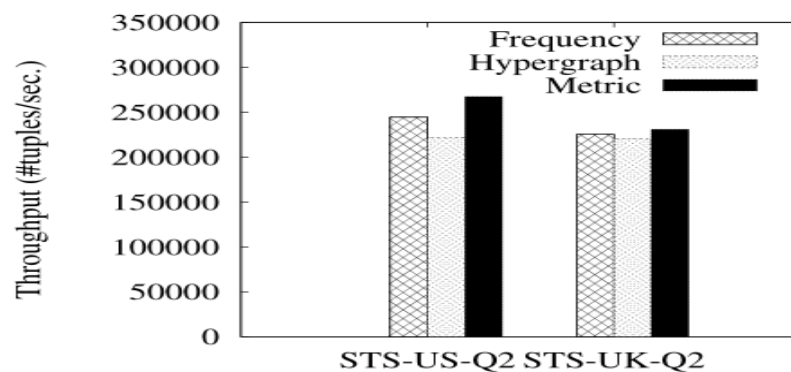
- 本文主要目标
- 分布式发布订阅系统
- 工作负载分区策略
- 动态负荷调整方法
- 实现结果
- 总结

负载分区结果对比

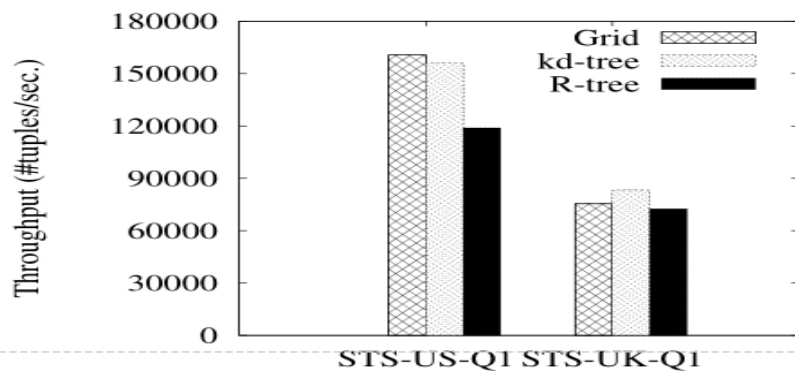
文本分区，空间分区的实验结果对比。



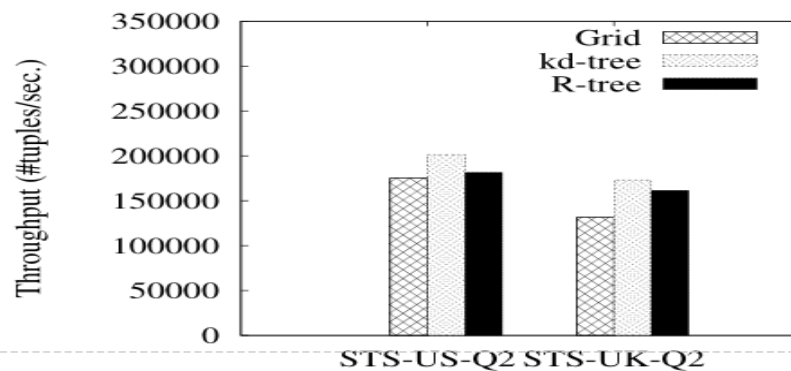
(a) Text-Partitioning (#Q1=5M)



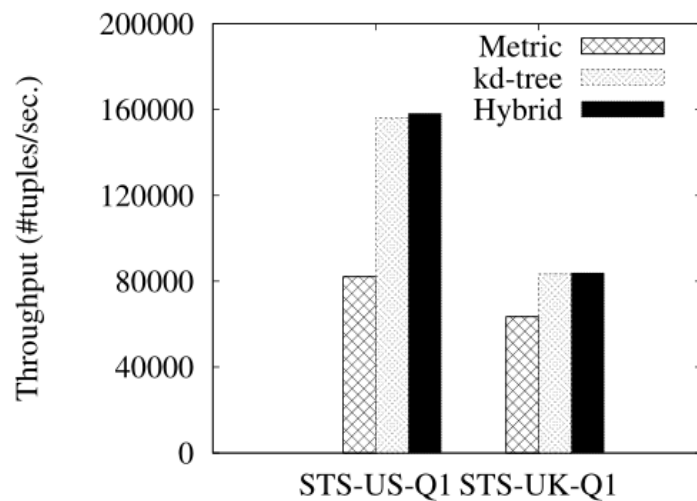
(b) Text-Partitioning (#Q2=10M)



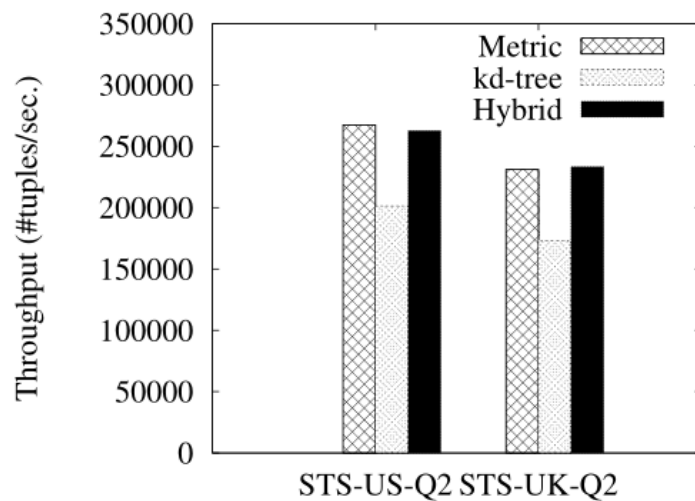
(c) Space-Partitioning (#Q1=5M)



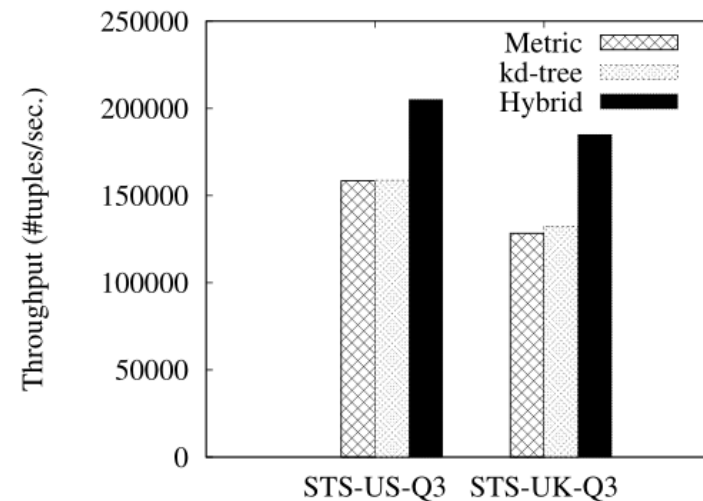
(d) Space-Partitioning (#Q2=10M)



(a) #Queries=5M (Q1)

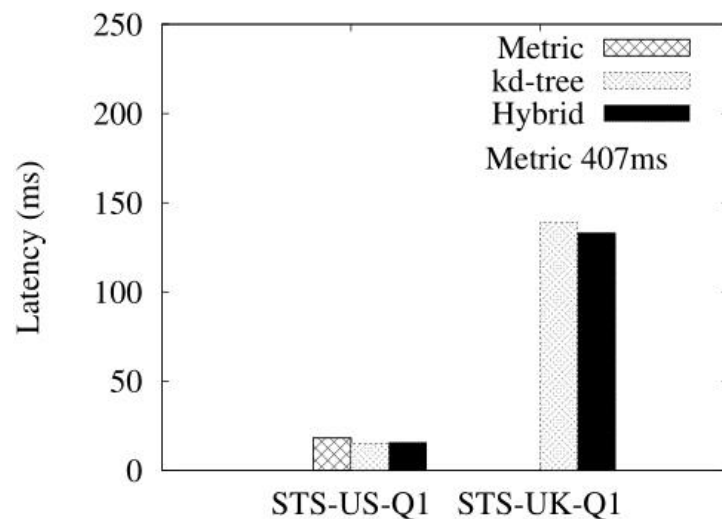


(b) #Queries=10M (Q2)

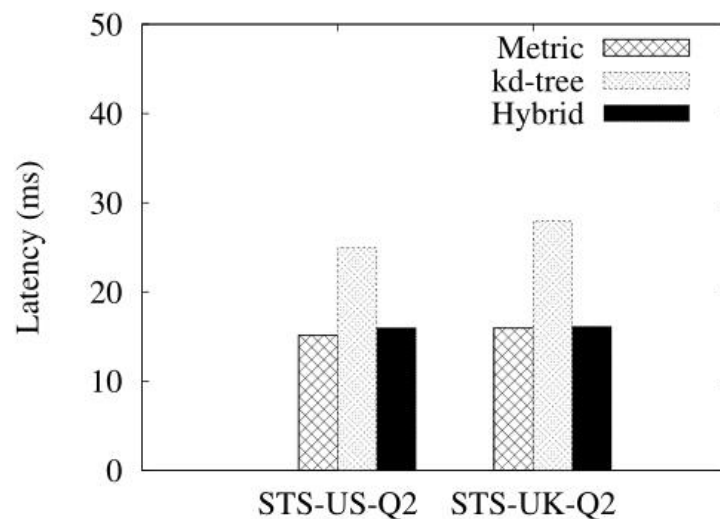


(c) #Queries=10M (Q3)

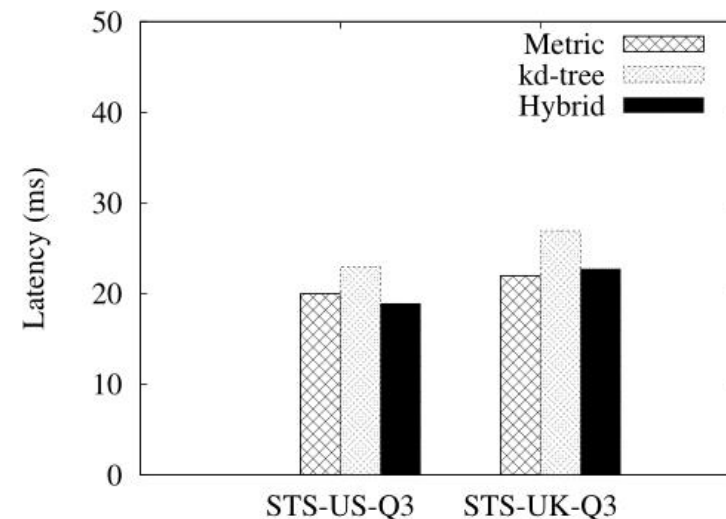
6种不同数据集吞吐量对比：纯黑色为混合分区，对比吞吐量较高



(a) #Queries=5M (Q1)



(b) #Queries=10M (Q2)



(c) #Queries=10M (Q3)

6种不同数据集延时对比：纯黑色为混合分区算法，对比延迟时间较低

Content

- 本文主要目标
- 分布式发布订阅系统
- 工作负载分区策略
- 动态负荷调整方法
- 实现结果
- 总结

总结

工作负载分区策略

- 对比单一的文本分区或者空间分区，混合分区方法结合两者的优点：综合考虑了工作负载总量最小化和工作负载均衡

动态负荷调整方法

- 满足负载均衡约束
- 最小化了数据流的迁移成本

Thank you !!
