目录

[动态图上高效缓存的并发处理点对点查询策略 2](#_Toc146296639)

[摘要 2](#_Toc146296640)

[前言 4](#_Toc146296641)

[背景和动机 4](#_Toc146296642)

[系统概述 4](#_Toc146296643)

[基于xxxxx的优化策略 4](#_Toc146296644)

[实验评估 4](#_Toc146296645)

[相关工作 4](#_Toc146296646)

[结论 4](#_Toc146296647)

[参考文献 5](#_Toc146296648)

动态图上高效缓存的并发处理点对点查询策略

摘要

随着图处理技术在地图导航、网络分析等领域的大范围应用，大量点对点查询作业在同一个底层图上并发运行。然而现有的图上点对点查询系统致力于加快对单个查询的响应速度，在面对高并发的点对点查询需求时，由于冗余的数据访问，处理效率很低。我们观察到并发查询任务之间存在着数据访问相似性，这启发我们提出了一种高效缓存的并发点对点查询方法。具体地，我们将图查询过程中的数据分为“图结构数据”和“任务特定数据”，前者记录了图的拓扑信息，后者记录了查询任务所要访问的图结构数据分块，不同查询所需访问的图结构数据分块可能重叠。

GraphCPP: A Cache-Efficient Approach for Concurrent Point-to-Point Queries in Dynamic Graphs

Abstract

As graph processing techniques become increasingly prevalent in domains such as map navigation and network analysis, the demand for concurrent point-to-point query tasks on dynamic graphs has surged. While many contemporary systems prioritize optimizing the response time of individual queries, their efficiency diminishes in high-concurrency query scenarios due to recurrent redundant data accesses. Identifying a consistent pattern in data access across concurrent query tasks, we propose a novel caching mechanism designed specifically for concurrent point-to-point queries. We classify data accessed during graph querying into two distinct types: "graph structural data," which encapsulates the graph's topology, and "task-specific data," which monitors the portions of the graph structural data accessed during queries. Noting that data chunks from separate queries may intersect, our methodology adopts a data-driven scheduling approach. In this context, a single instance of the graph structural data is preserved in memory/LLC throughout the execution of concurrent tasks. These tasks access data cohesively at a detailed level, founded on graph data segments, ensuring a single data access serves multiple tasks. This strategy reduces data access overhead and boosts the throughput of concurrent graph queries.

因此，我们采用了一种数据驱动的调度方法：在执行并发点对点查询任务时，内存/LLC中只保留一份图结构数据。多任务之间以细粒度的图数据分块为单位共享数据。一次访问，多个任务处理，以此分摊数据访问的开销，提高并发图查询的吞吐量。为了展示GraphCPP的效率，我们将其与最先进的点对点查询系统进行对比，包括SGraph[x]、Tripoline[x]、Pnp[x]，实验表明，GraphCPP将并发点对点查询的效率提升了xxxx倍。

当前问题：摘要只写了数据共享这一个创新点，显得单薄。

To ascertain the effectiveness of GraphCPP, we conducted comparative evaluations against leading point-to-point query systems, specifically SGraph[x], Tripoline[x], and Pnp[x]. Our experimental data reveals that GraphCPP enhances the efficiency of concurrent point-to-point queries by a significant factor of xxxx.

前言

背景和动机

系统概述

基于xxxxx的优化策略

实验评估

相关工作

结论

致谢**INTRODUCTION**

BACKGROUND AND MOTIVATION

OVERVIEW OF GraphCPP

xxxxx STRATEGY xxxxx

EXPERIMENTAL EVALUATION

RELATED WORK

CONCLUSION

ACKNOWLEDGMENTS

参考文献

1. Chen H, Zhang M, Yang K, et al. Achieving Sub-second Pairwise Query over Evolving Graphs[C]//Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2. 2023: 1-15.
2. Jiang X, Xu C, Yin X, et al. Tripoline: generalized incremental graph processing via graph triangle inequality[C]//Proceedings of the Sixteenth European Conference on Computer Systems. 2021: 17-32.
3. Xu C, Vora K, Gupta R. Pnp: Pruning and prediction for point-to-point iterative graph analytics[C]//Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. 2019: 587-600.