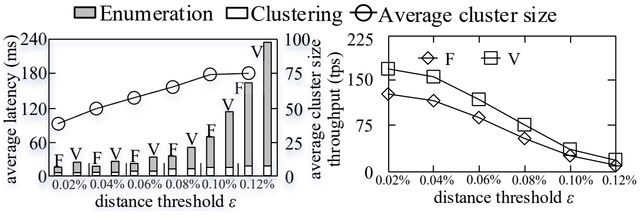
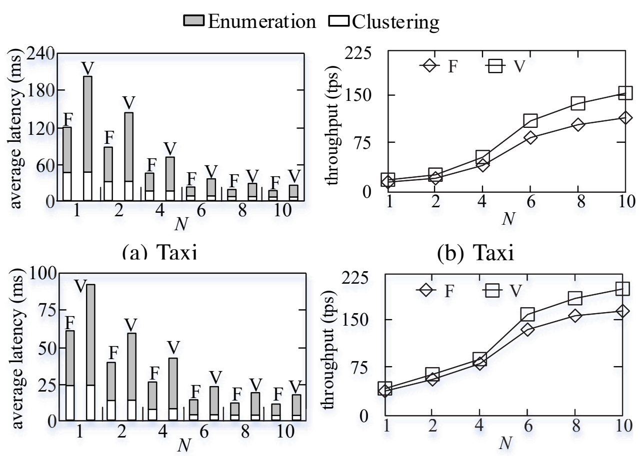



(a) Taxi (b) Taxi

(c) Brinkhoff (d) Brinkhoff

**Figure 13: Pattern Detection Performance vs. $\epsilon$**




(a) Taxi (b) Taxi

(c) Brinkhoff (d) Brinkhoff

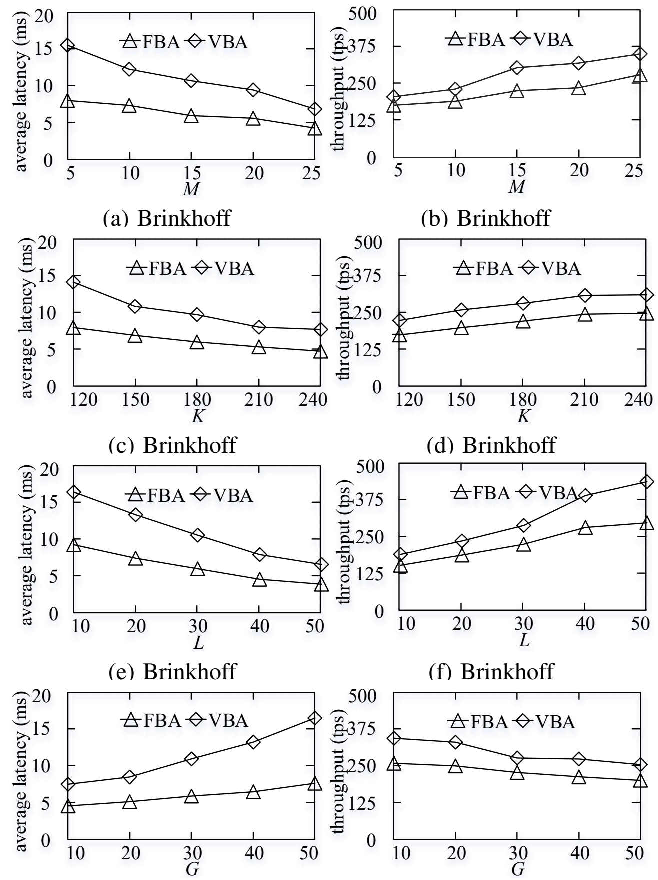
**Figure 14: Pattern Detection Performance vs. $N$**

## 7.3 Pattern Enumeration Performance

Next, we consider the effects of the constraints $M$, $K$, $L$, and $G$ on the performance of pattern enumeration methods VBA and FBA. Here, BA is omitted as it cannot run on large datasets, and clustering is omitted as its performance is not affected by the constraints. Fig. 15 shows the latency and throughput when increasing $M$, $K$, $L$, and $G$ on Brinkhoff. As expected, VBA has better throughput than FBA, while FBA has better latency. In addition, the average latency decreases (while the throughput increases) as $M$ and $K$ or $L$ grow, as fewer valid candidates are generated or the pruning ability of Lemma 5 increases. However, the average latency increases (while the throughput decreases) as $G$ grows. The reason is that, as $G$ grows, more valid patterns are generated.

## 7.4 Summary

We can conclude that for clustering, our method RJC is more efficient on both latency and throughput when compared with the existing methods SRJ and GDC. When considering pattern enumeration, the scalability of our methods FBA and VBA are better than that of BA. In addition, FBA has the best latency, while VBA has the best throughput. The overall finding is that the ICPE framework and its corresponding methods are scalable, and are able to achieve low latency and high throughput. In addition, we recommend FBA if the throughput achieved is able to keep up with the incoming workload, and we recommend VBA if this is necessary to keep up with the incoming workload or if the higher latency is not critical.




(a) Brinkhoff (b) Brinkhoff

(c) Brinkhoff (d) Brinkhoff

(e) Brinkhoff (f) Brinkhoff

(g) Brinkhoff (h) Brinkhoff

**Figure 15: Pattern Enumeration Performance vs. $M$, $K$, $K$, $G$**

# 8. CONCLUSIONS

In this paper, we investigate real-time distributed co-movement pattern detection over streaming trajectories, which is useful in future movement prediction, trajectory compression, and a variety of location-based services. We develop a Flink-based framework, called ICPE. Flink is used because of its suitability for stream processing and its high efficiency and reliability. The framework encompasses two phases, i.e., clustering and pattern enumeration. To accelerate the clustering, we utilize a GR-index based range join, together with effective pruning techniques. To support efficient pattern enumeration, an id-based partitioning method, two bit compression techniques, and candidate based enumeration are utilized to reduce the storage and processing costs from exponential to linear. Extensive experiments using both real and synthetic datasets suggest that the proposed framework and its constituent data structures and algorithms are efficient and scalable. It is observed that, ICPE can achieve low latency and high throughput, and thus, it can support real-time co-movement pattern detection over streaming trajectories. In future research, it is of interest to extend ICPE to support the detection of additional types of advanced patterns.

# 9. ACKNOWLEDGMENTS

This work was supported in part by the National Key R&D Program of China under Grant No. 2018YFB1004003, the 973 Program under Grant No. 2015CB352502, the NSFC under Grant No. 61522208, the NSFC-Zhejiang Joint Fund under Grant No. U1609217, and the ZJU-Hikvision Joint Project. Yunjun Gao is the corresponding author of the work.