**Richard Kershner**

Phone/txt: 303-898-8757

info@RichardKershner.com

https://github.com/Richard-Kershner/RideShare\_OnDemand\_MicrosTransit\_Algorithms

**11/01/2023**

**Testing Scheduling Algorithms for a Ride Share System**

Through my involvement in a basic ride-sharing system (on-demand, curb-to-curb or micro-transit), I encountered various challenges related to ride scheduling. This concept revolves around providing curb-to-curb transportation within specified areas, such as small towns or municipalities, where small buses or vans can efficiently pick up multiple passengers from designated addresses. Unlike conventional city buses, this approach allows passengers to enjoy door-to-door transportation, enhancing accessibility and convenience. These systems, when integrated with city transit services, play a vital role in extending transportation options to more rural areas where establishing a fixed-route bus service is cost-prohibitive. Notably, this model is already in use in several regions, including FlexRide by the Regional Transportation District (RTD) in Denver, CO; On-Demand services by the Utah Transit Authority (UTA); Go-Glades offered by Palm Tran in Palm Beach, Florida; Metro Micro services provided by the Los Angeles County Metropolitan Transportation Authority; and Your-Ride-Plus operated by the Flint Mass Transit Authority (MTA). Moreover, there are many more specialized services that cater to specific needs, such as medical transportation. This industry has witnessed significant growth over the past couple of decades, becoming an integral part of a robust transportation systems. Understanding scheduling algorithms as well as the concept of an algorithm test bed is an integral to the future of this industry.

**What are Scheduling Algorithms and Why are they Important?**

Scheduling algorithms serve as the foundation of an efficient ride-sharing system. When a passenger requests a ride via a mobile app or by phone, they provide specific details, including pick-up and drop-off locations, as well as preferred times. Scheduling algorithms play a pivotal role in determining how these requests can be seamlessly integrated into existing routes, significantly impacting the system's overall efficiency.

Efficient scheduling is vital as it ensures passengers reach their destinations promptly and drivers follow logical routes. This not only creates a safer and more reliable ride-sharing experience but also reduces the consumption of resources. Drawing from my extensive six years of industry experience, I've observed recurring issues. These include instances where routes pass the passenger's drop-off location multiple times, leading to unnecessary delays when rides are canceled during convoluted routes. Additionally, passengers are often routed through extensive detours, causing substantial delays, especially in the presence of construction, traffic, and train crossings, making them late for critical appointments.

The current system I have encountered, which primarily relies on finding the closest pick-up time to the requested time and fitting in the closest drop-off time to the pick-up, yields less than a 25% chance of logical scheduling when two rides are scheduled simultaneously. This leads to mounting frustration for both drivers and passengers, increasing distractions and raising concerns about safety as well as unnecessarily wasting resources. The optimization of scheduling algorithms is, therefore, a paramount consideration for the future of ride-sharing programs.

**What is an Algorithm Testing Bed and Why Should It Be Used?**

An algorithm testing bed, in simple terms, involves taking various groups of ride requests and subjecting each group to different scheduling algorithms. It collects data on how these algorithms perform based on key indicators, such as the distance covered by the vehicle, the number of rejected requests, and the impact on each passenger's ride time.

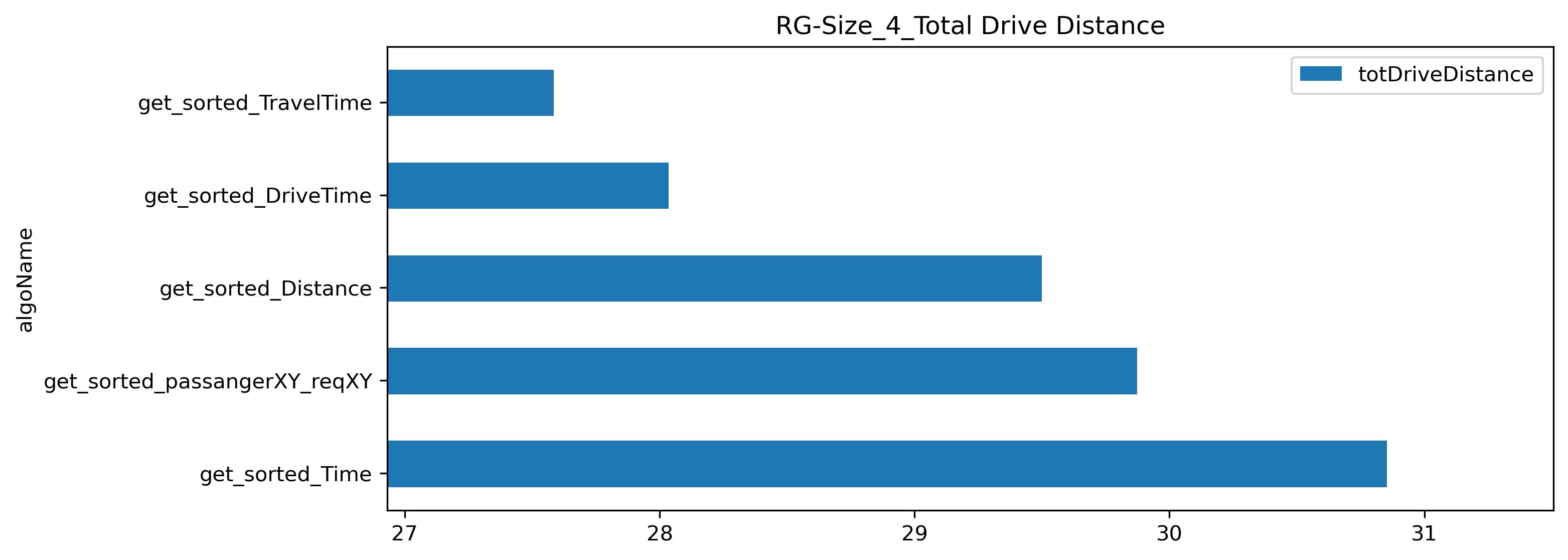
Various ride scheduling methods exist, including proximity to the request time, minimizing driving distance, and minimizing ride duration, each with its own resource requirements. For instance, request-time-based algorithms typically explore alternative routes only when the initial choice doesn't fit, whereas vector-based locations change the order of route testing with minimal mathematical computations. On the other hand, minimizing distance or time involves evaluating all possible route fits using mapping modules, which can impose substantial resource demands, especially in a large fleet of vehicles. Future scheduling algorithms may even incorporate neural networks, which require even more resources and fine-tuning.

Markers help identify opportunities for resource optimization while considering other factors that impact the passenger experience. Implementing an algorithm testing bed can lead to significant benefits, such as reducing drive distance (and related expenses like fuel) and decrease requests by more than 10%. Additionally they can cut a passenger's ride time on average by over 50%. Detailed examples of preliminary testing on a small data source can be found in Appendix A.

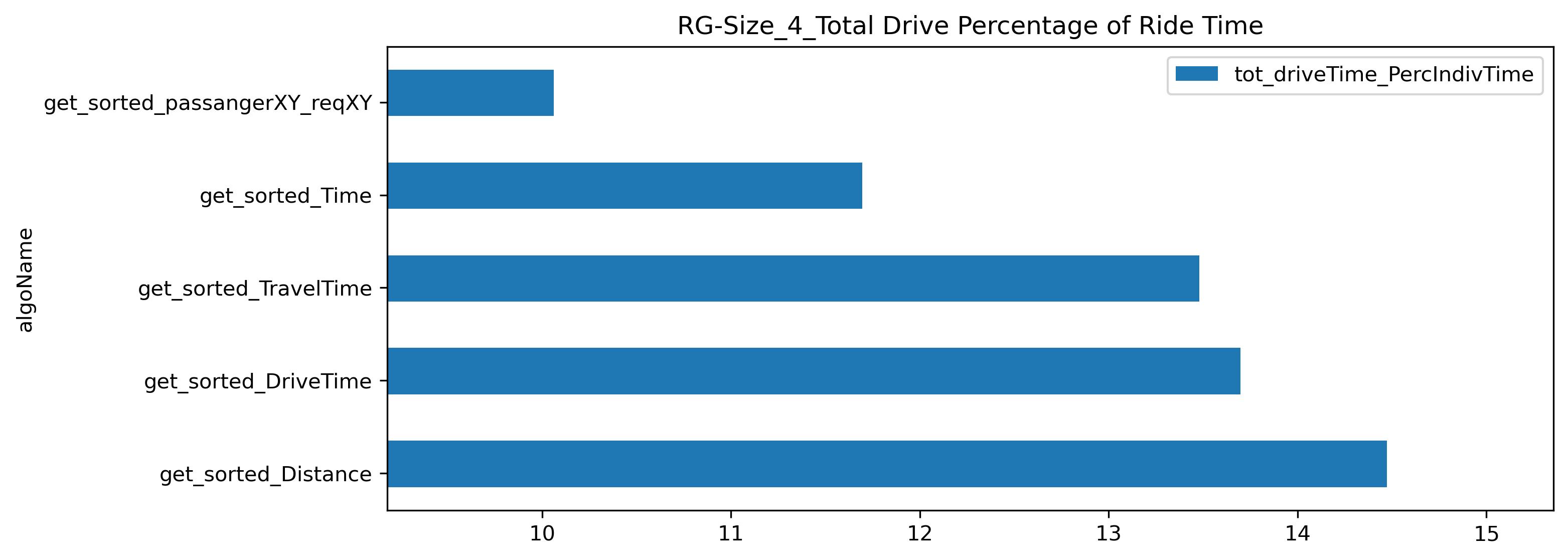
A testing bed can be applied and utilized in numerous areas before embarking on new software development or expanding existing systems. In addition to the aspects mentioned earlier, it's crucial to address various scenarios, including how the system handles ride cancellations and the resulting waiting times for passengers at pick-up points due to circuitous routes. Other factors like accommodating delays and efficiently managing single-location stops for multiple passengers must also be considered. Furthermore, it's essential to acknowledge that all algorithms mentioned and tests shown in Appendix A are based on limited data. To ensure the robustness and reliability of the software, a comprehensive testing bed should be executed using extensive real-life data samples. This research phase is integral to inform and refine the software design stage.

**Appendix A**

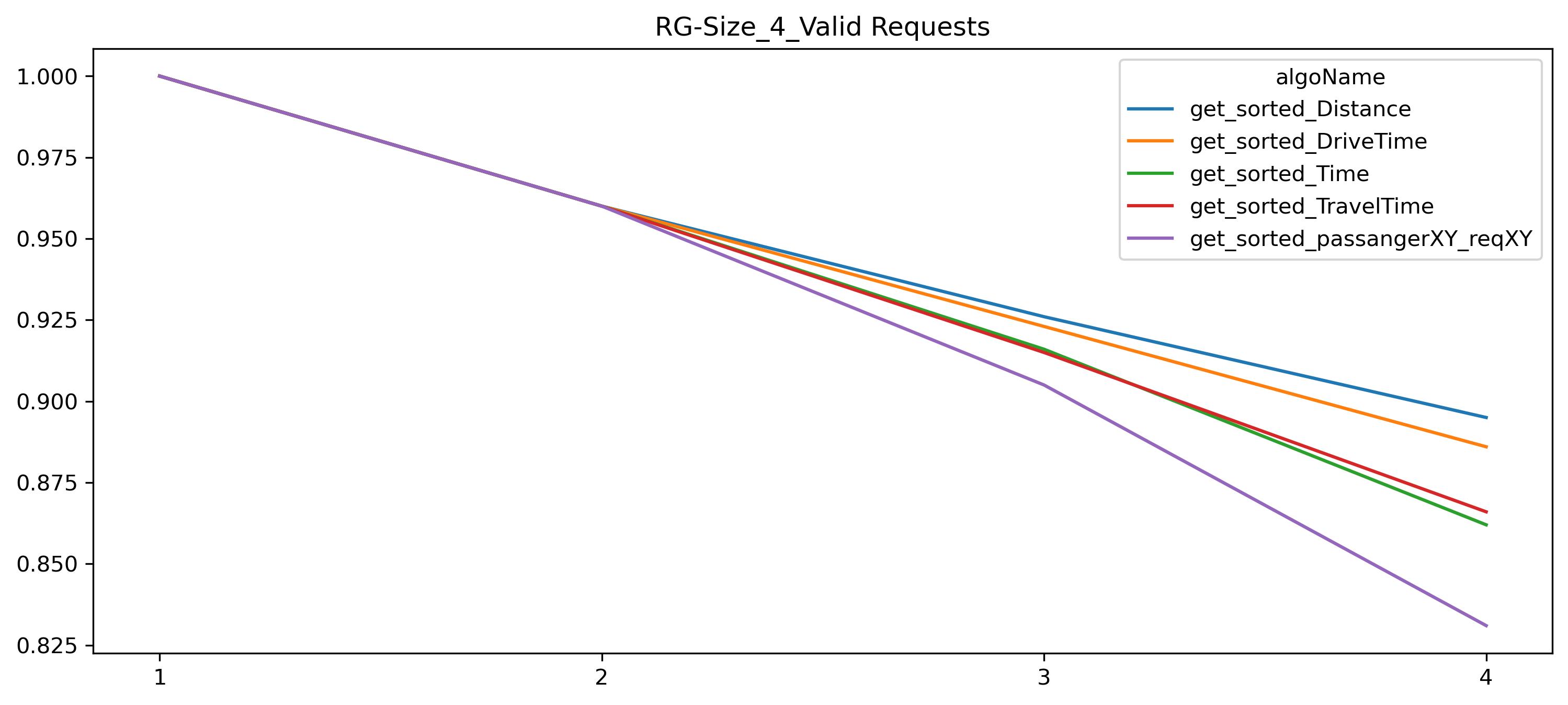
**Total Drive Distance \***

Total drive distance is based on 4 requests (pick-up and drop-off) which generate 8 stops. As shown on the graph to the right and based on all requests being valid, the ride distance varied from just under 28 miles to almost 31 miles, a difference of 27.5 to 31.5 or 10%.

**Client’s Average Ride Time \***

As seen on the graph to the right, passenger’s time spent riding on the bus varied from 10 minutes to 14 ½ minutes, almost 50% difference on average! For shorter rides, this would appear to make little difference but for a request from one end of town to the other, this can easily turn into a 45 minute ride, if not longer, for a small town. Specially when traffic, railroads and construction are included.

**Request Validation \***

A ride is restricted by a set total ride time. In this example, as more rides are added on, there is an increase in rides that are rejected. In this example it varies on the 4th request by .825 to .925 or a difference of roughly 10%.

\* All data is generated using limited source of addresses (69) randomly chosen as well as limited time frames.