Visualising the **Impact** of **High-dimensional** Configuration **Parameters** on the **Performance** of Data Distribution Service

Kaleem Peeroo, Peter Popov, Vladimir Stankovic, Tillman Weyde

*Data Distribution Service (**DDS**) is a specification for middleware in **real-time and mission-critical systems** such as **energy management systems** and **medical devices**. Such systems have specific **performance requirements** to **function properly** and **avoid catastrophe**. However, the **performance** of the system can **vary depending** on the various **parameters** defined before or during run-time. **Understanding how** the different **parameters impact performance helps DDS** system **designers** set up distributed **systems** in **critical domains**. This **work introduces** a **workflow** for **understanding how parameter** values can **affect** system **performance**, as seen in their **distribution form**. It **applies to DDS** and **any configurable system** with **multiple parameters** that can take **multiple values** that may result in **different system performance**. This **visualisation workflow** is **demonstrated** on a **dataset** created by running tests on an **actual DDS system**. **We argue** that they **help understand non-linear relationships between** the **parameters** and DDS **performance** and **help identify anomalous results** that may **need investigation**.*

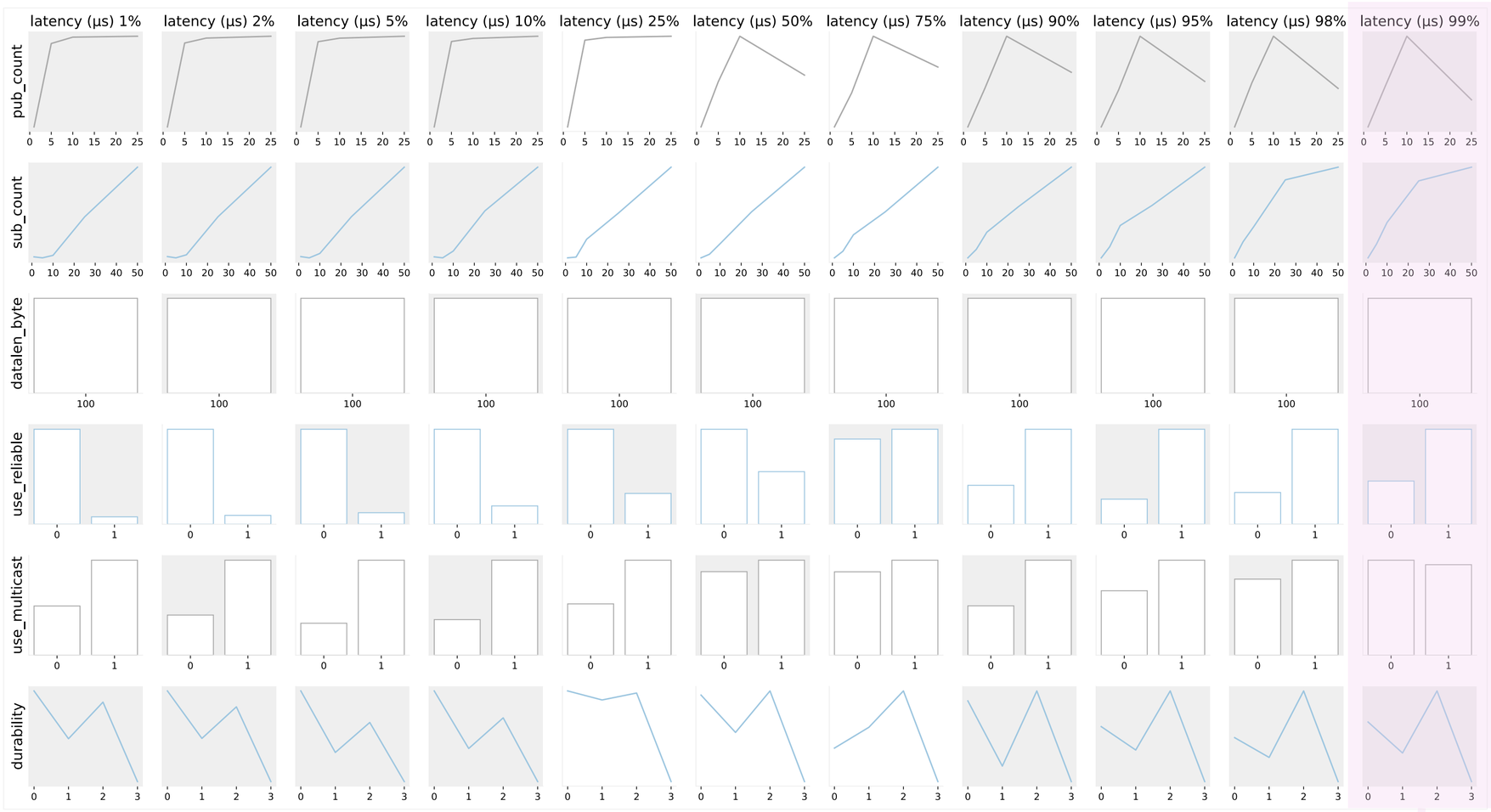
1. Problem

The performance of DDS systems depends on the values of the parameters configured. Examples of parameters include:

- **Publisher Count** How many publishers (data senders).
- **Subscriber Count** How many subscribers (data receivers).
- **Data Length** How big are the messages.
- **Durability** Send past messages to newly joined subscribers?
- **Reliability** Wait for message acknowledgement?
- **Multicast Usage** Send to multiple recipients at once?

Different values for the **parameters** lead to a **combinatorial explosion**.

How do we **visualise** the **impact** of different configuration **parameters** and **their interdependencies** on the **performance** of a DDS system?



2. Solution

Let's assume we are interested in how the stated parameters affect the latency of a DDS system.

1. Explore the global view using small multiples.
We **start** with the **small multiples view** seen at the **top right**. **Parameters** vary per **row**. Cumulative **quantiles** increase towards the right per **column**. Depending on the parameter value type, both line graphs and bar charts are used. Here, we **look for irregularities** in trends.
Given that we are interested in latency, the **worst-case latency** can be seen in the **last four columns**. We would **expect** the **latency** to **increase linearly** as the **number** of **publishers** or **subscribers increases**. The **top right cell** (publisher count for the 99th percentile) **shows otherwise**.

2. Zoom into a specific quantile and generate median box plots.
We "**zoom**" into this **column** by **generating box plots** of the **median latencies** for **all tests**, as seen in the figure outlined in pink.

3. Filter out values for specific parameters.
Let's **isolate** all tests that use **10 publishers** to see what is happening. We **replot** the **box plots** as seen in the figure surrounded in blue.

4. Generate new median box plots.
The newly created blue figure shows a **new irregularity**: **25 subscribers** on top of 10 publishers have **higher latencies than 50 subscribers**.

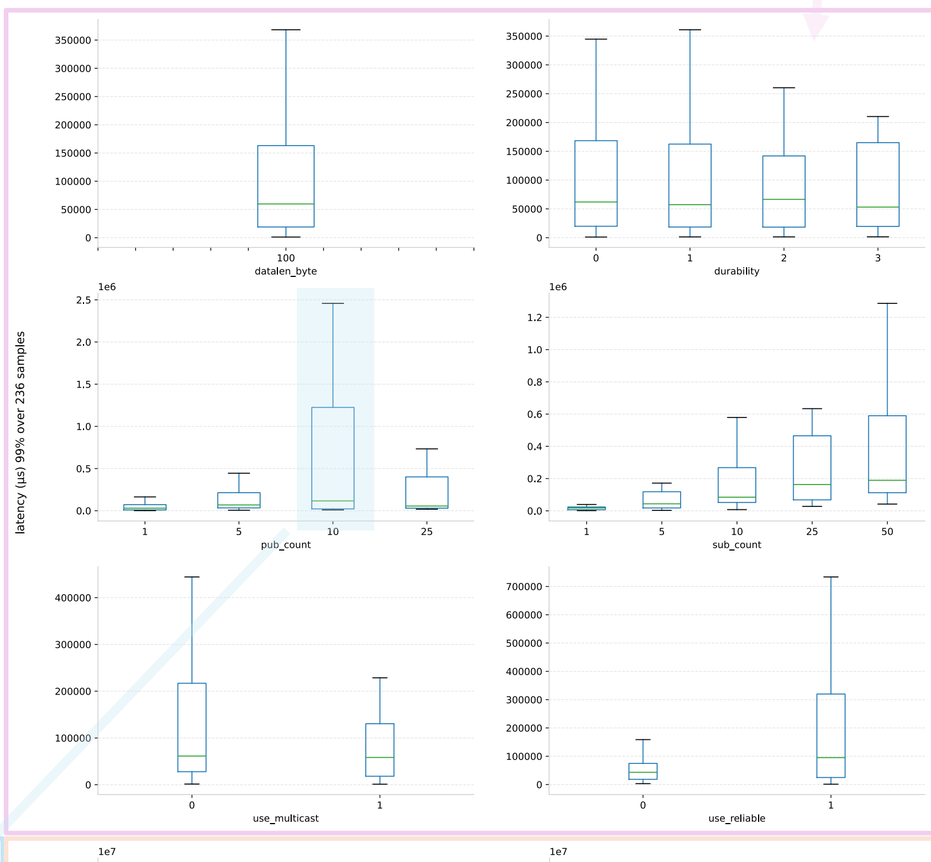
5. Go back to step 3 (filter out values for specific parameters) until a conclusion is made.
Let's **isolate** all **tests** with **10 publishers and 25 subscribers** and generate the box plots outlined in orange.
We notice that **8 tests** produced these values. That **confirms** that there were **not** a few **anomalous results** that skewed the data. The thing that pops out here is seen when we compare the blue figure with the orange figure. **Focus on** the use of **multicast**. The blue figure shows that **using multicast** (giving a value of 1) **leads** to **lower latency**. Meanwhile, the orange figure shows that **using multicast leads** to a **higher latency**. What is the difference between the two figures? The specific usage of **25 subscribers alongside 10 publishers.**

6. Conclusion is made.
If a DDS system **designer wants** to **design** a **system** specifically, it is best **not to use multicast** to get **more performant latency** readings when using **10 publishers and 25 subscribers.**

We introduce a global-to-local visualisation workflow that's only 5 steps:

1. Explore the global view using small multiples.
2. Zoom into a specific quantile and generate median box plots.
3. Filter out values for specific parameters.
4. Generate new median box plots.
5. Go back to step 3 until a conclusion is made.

3. Demo



4. Future Work

- Creating a **tool** to **let users interact** in **real-time**.
- Deal with **limitations**:
  - **Increasing** the number of **parameters** or **x-axis values** makes the **small multiples** view **visually overwhelming**.
  - If tests with one parameter value produce **really small latencies** while tests with another parameter value produces **really big latencies** the **box plots** will be **squashed out of proportion**.
- **Explore** other **options** of **multivariate visualisation** that **involve user interaction**.