
Hi, I'm Manik

Intern,
Student,
Web dev,

Sometimes, I contribute to Prometheus

<https://www.manikrana.dev/>

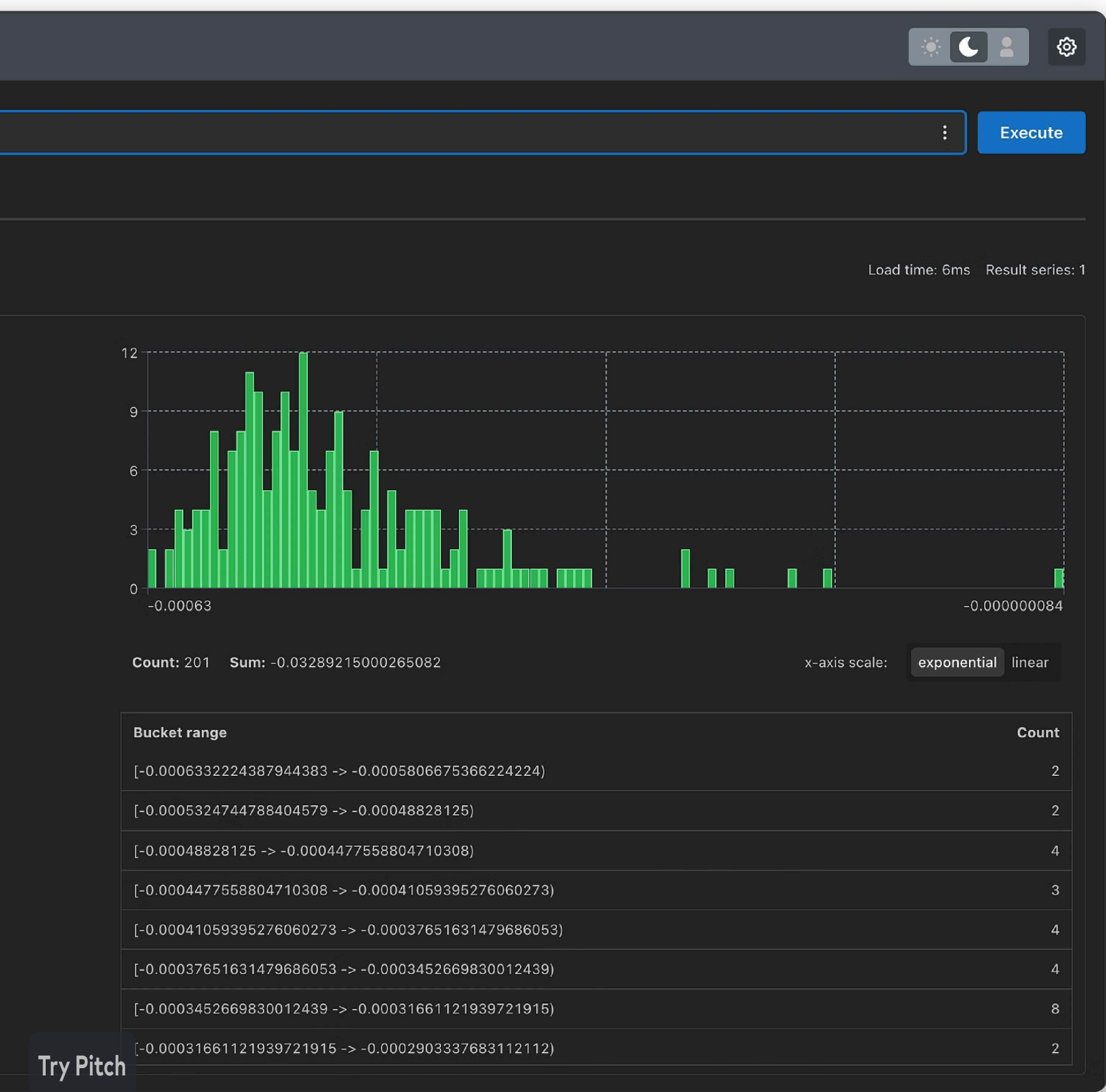


Sometimes, I contribute to Prometheus

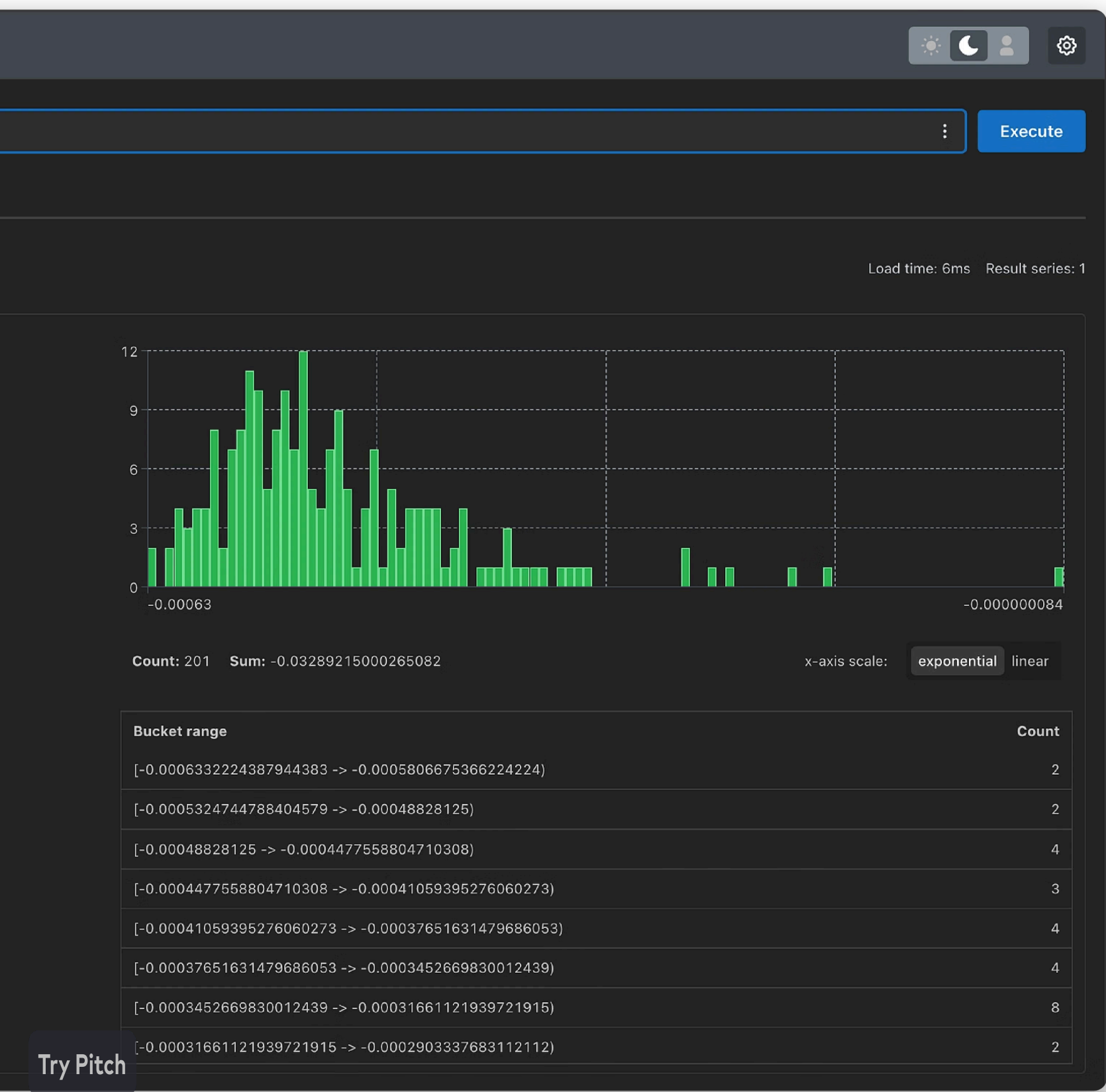
"Like what?" you may ask



1) Prometheus UI: Native Histogram Rendering



1) Prometheus UI: Native Histogram Rendering



Update histogram math #13680

Merged beorn7 merged 61 commits into prometheus:main from Maniktherana:update-histogram-math on Jul 4

Conversation 104 Commits 61

+427 -53

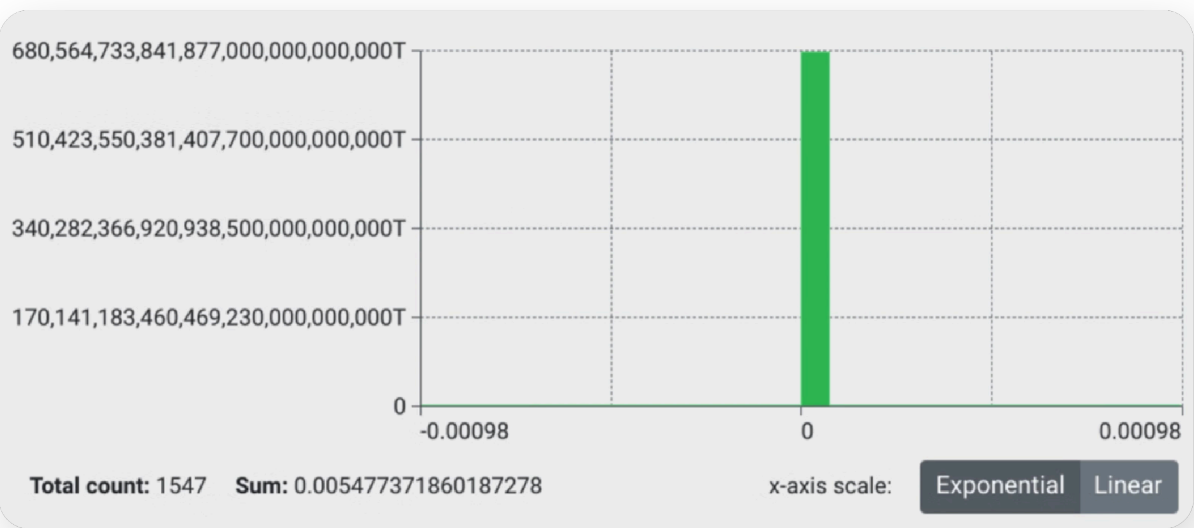


Some Challenges

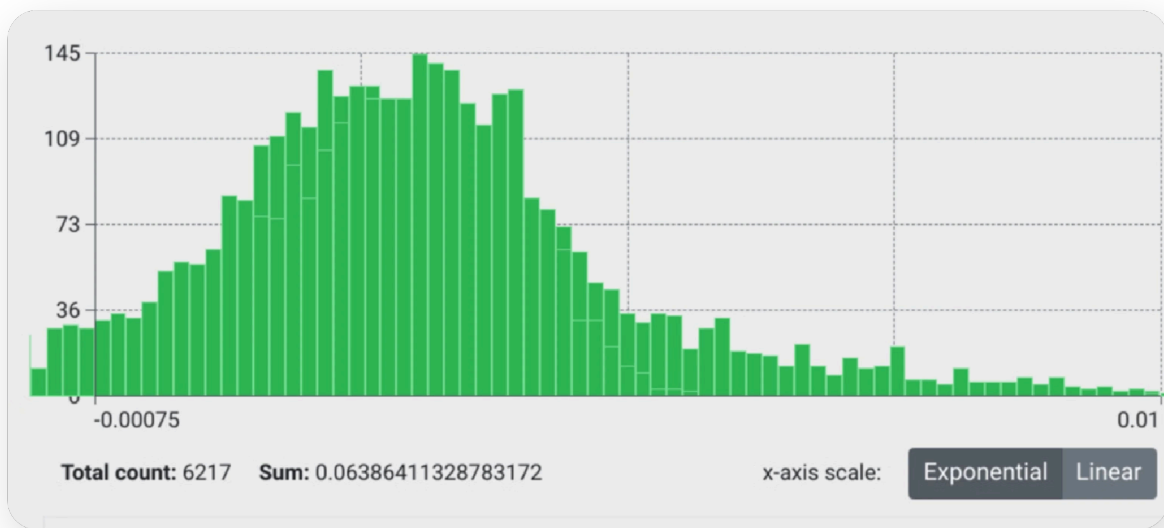
It's easy to use something off the shelf, but we used raw HTML/CSS/JS to render histograms

Positioning a couple of bars on a plane is relatively straightforward. Things get tricky when exponential scales and zeros are involved

My PRs (and long slack messages with Björn) were mostly just getting the math right



Consistent bucket scaling



Consistent bucket positioning

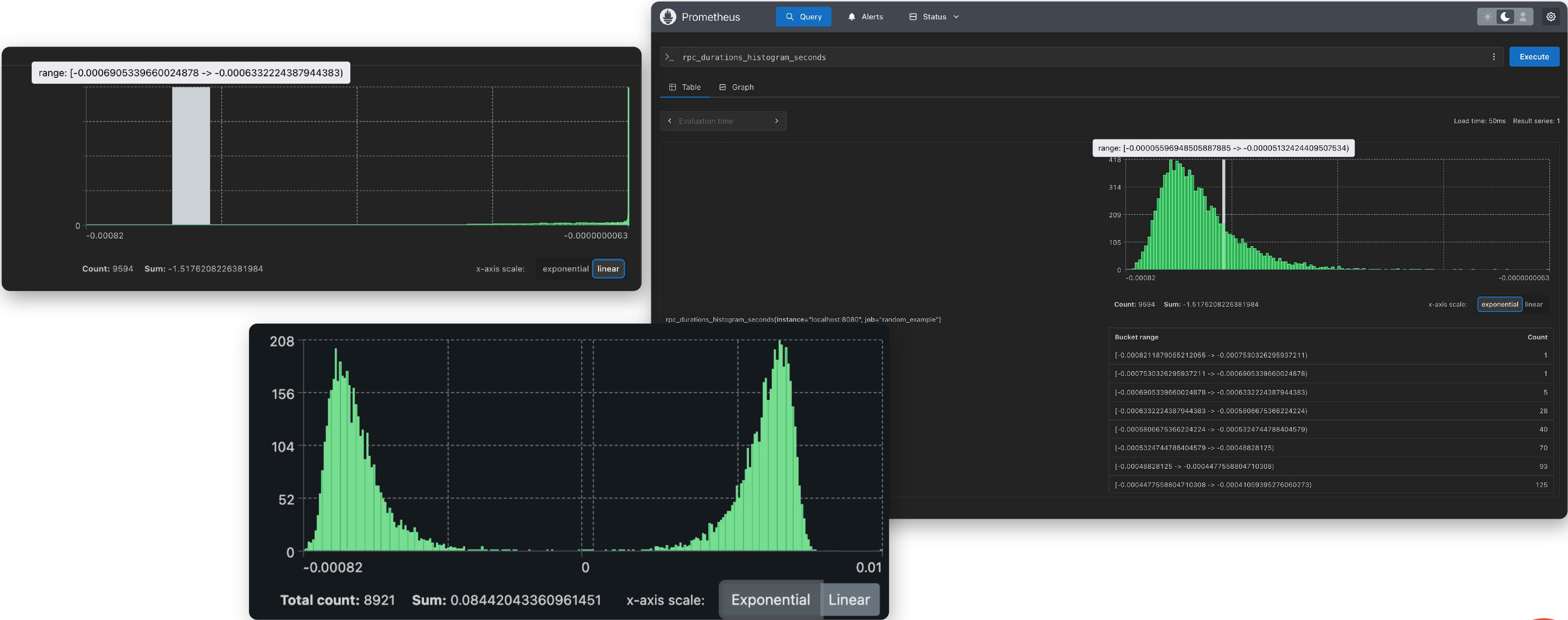


Zero-bucket handling



The Result

Over the course of many weeks of iteration, we were finally able to render beautiful histograms 🎉



2) Prometheus Parser: Implementing Created Timestamps

KEY TECHNICAL DETAILS

- It utilizes an ephemeral deep copy of the main parser to "peek ahead" at upcoming lines without altering the original parser's state.

The function verifies if the "_created" line belongs to the same metric series by comparing labels.

- The CreatedTimestamp() function specifically targets metrics of type counter, summary, and histogram.

```
func deepCopy(p *OpenMetricsParser) OpenMetricsParser {  
    newB := make([]byte, len(p.l.b))  
    copy(newB, p.l.b)  
  
    newLexer := &openMetricsLexer{  
        b:      newB,  
        i:      p.l.i,  
        start:  p.l.start,  
        err:    p.l.err,  
        state:  p.l.state,  
    }  
  
    newParser := OpenMetricsParser{  
        l:      newLexer,  
        builder: p.builder,  
        mtype:   p.mtype,  
        val:     p.val,  
        skipCTSeries: false,  
    }  
    return newParser  
}
```

It was the process of multiple syncs and pair programming sessions



This is not ideal

```
func deepCopy(p *OpenMetricsParser) OpenMetricsParser {  
    newB := make([]byte, len(p.l.b))  
    copy(newB, p.l.b)  
  
    newLexer := &openMetricsLexer{  
        b:      newB,  
        i:      p.l.i,  
        start:  p.l.start,  
        err:    p.l.err,  
        state:  p.l.state,  
    }  
  
    newParser := OpenMetricsParser{  
        l:      newLexer,  
        builder: p.builder,  
        mtype:   p.mtype,  
        val:     p.val,  
        skipCTSeries: false,  
    }  
    return newParser  
}
```



As we learned the hard way

OpenMetrics using gigabytes more memory #14808

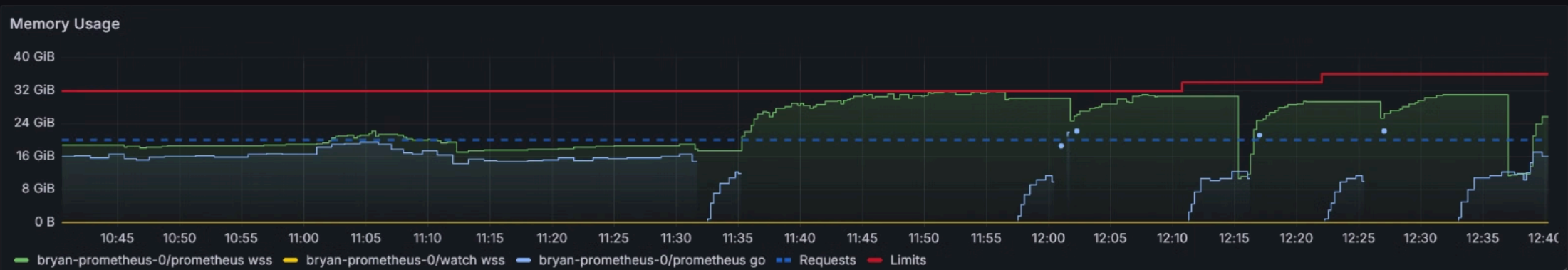
Closed bboreham opened this issue last week · 3 comments

OOM at 32GB limit; also CPU usage went way up.

CPU Usage



Memory Usage



This is not ideal

```
func deepCopy(p *OpenMetricsParser) OpenMetricsParser {
    newB := make([]byte, len(p.l.b))
    copy(newB, p.l.b)

    newLexer := &openMetricsLexer{
        b:      newB,
        i:      p.l.i,
        start:  p.l.start,
        err:    p.l.err,
        state:  p.l.state,
    }

    newParser := OpenMetricsParser{
        l:      newLexer,
        builder: p.builder,
        mtype:   p.mtype,
        val:     p.val,
        skipCTSeries: false,
    }

    return newParser
}
```

oops 🙈



Whats next for me?

I hope to stick around and contribute more

Develop a heatmap view for Native Histograms

Benchmark our OM/CT related code

Optimize and ship a better CT parser for OM



Thank you

Bartek, Arthur, Daniel, Björn, Julius & The Prometheus Team

<https://www.manikrana.dev/>

