libTSDB

Goutham Veeramachaneni
Student @ IIT Hyderabad, India
ex-intern @ CoreOS

putadent gouthamve

TSDB: github.com/prometheus/tsdb

- Prometheus 2.0’s storage engine
- A lib. vendored by Prometheus.

Why?

- Time-series is everywhere!
- A nice API for large datasets
- Awesome compression: 1 Billion points in ~1.2GB

Simple use-case: Prometheus with PUSH!

- Lots of requests.
- Several people (including me) built “aggregators” which expose push data to Prometheus.

- Let’s build a native Prometheus server with push functionality!

Introducing PromFlux

- Ingest using Influx line protocol - pre-built client libs!

- Query using PromQL <3

Introducing PromFlux

- Ingest using Influx line protocol - pre-built client libs!

- Query using PromQL <3

Umm, NO

- This is not how Prometheus works.
- These stunts are performed by an amateur, don't try this in production.

Umm, NO

- This is not how Prometheus works.
- These stunts are performed by an amateur, don't try this in production.

Umm, NO

- This is not how Prometheus works.
- These stunts are performed by an amateur, don't try this in production.

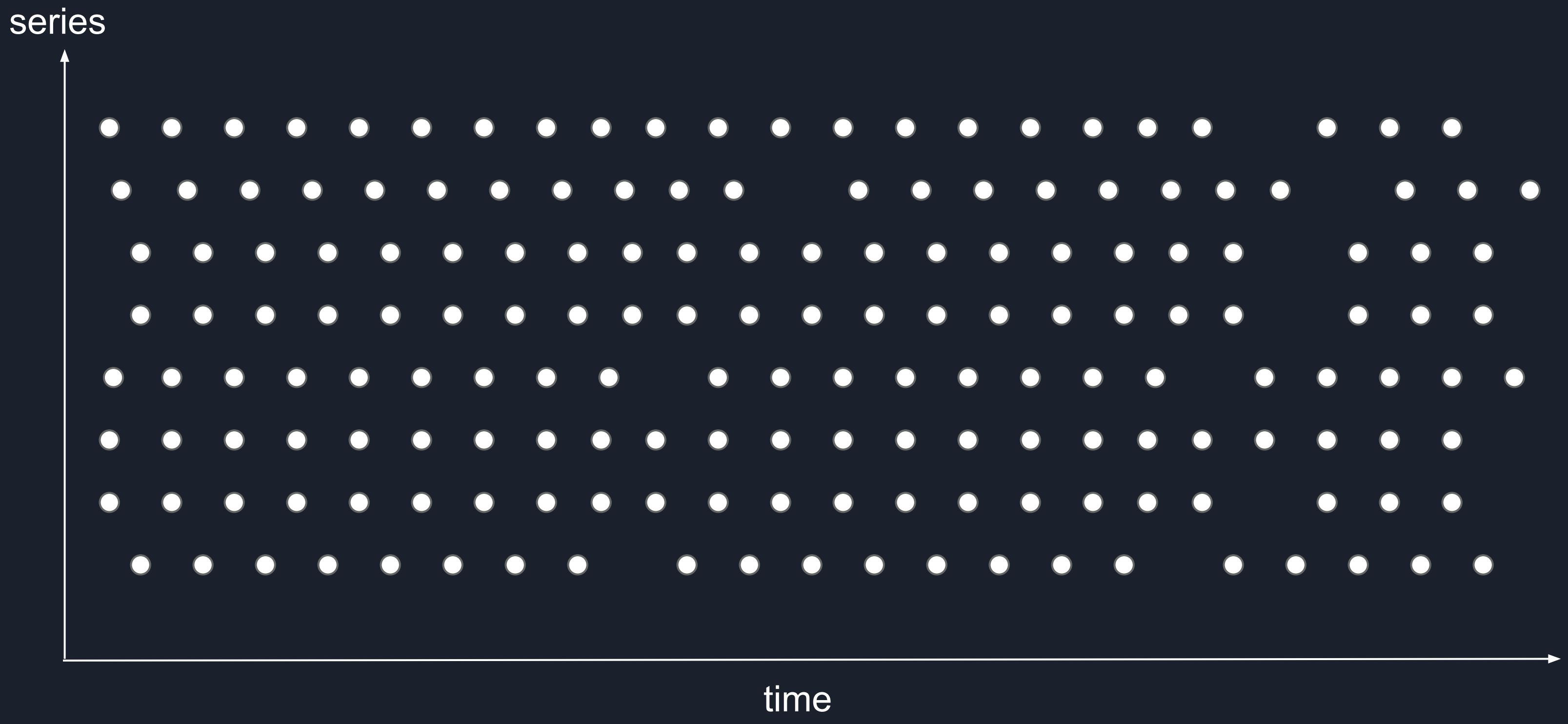
YOLO!

Some basics

- A time series:

```
(t0, v0), (t1, v1), (t2, v2), (t3, v3), ....
```

Some basics



Some basics

```
requests_total{path="/status", method="GET", instance="10.0.0.1:80"}

requests_total{path="/status", method="POST", instance="10.0.0.3:80"}

requests_total{path="/", method="GET", instance="10.0.0.2:80"}

...
```

Some basics

```
{
    __name__=”requests_total”,
    pod=”nginx-34534242-abc723
    job=”nginx”,
    path=”/api/v1/status”,
    status=”200”,
    method=”GET”,
}
```

Some basics

```
{
        __name__=”requests_total”,
        pod=”nginx-34534242-abc723
        job=”nginx”,
        path=”/api/v1/status”,
        status=”200”,
        method=”GET”,
}
```

```
{
        name=”requests_total”,
        pod=”nginx-34534242-abc723
        job=”nginx”,
        path=”/api/v1/status”,
        status=”200”,
        method=”GET”,
}
```

Some basics

```
{
    __name__=”requests_total”,
    pod=”nginx-34534242-abc723
    job=”nginx”,
    path=”/api/v1/status”,
    status=”200”,
    method=”GET”,
}
```

```
{
    pod=”nginx-34534242-abc723
    job=”nginx”,
    path=”/api/v1/status”,
    status=”200”,
    method=”GET”,
}
```

Some basics

```
requests_total{path="/status", method="GET", instance="10.0.0.1:80"}
```

Some basics

```
requests_total{path="/status", method="GET", instance="10.0.0.1:80"}
```

```
{name="requests_total", path="/status", method="GET", instance="10.0.0.1:80"}
```

Some basics

```
requests_total{path="/status", instance="10.0.0.1:80"}

requests_total{path="/status", instance="10.0.0.3:80"}

requests_total{path="/", instance="10.0.0.2:80"}
```

Select: *requests_total*

Some basics

```
{name="requests_total", path="/status", instance="10.0.0.1:80"}

{name="requests_total", path="/status", instance="10.0.0.3:80"}

{name="requests_total", path="/", instance="10.0.0.2:80"}
```

```
Select: {name="requests_total"}
```

Some basics

```
requests_total{path="/status", instance="10.0.0.1:80"}

requests_total{path="/status", instance="10.0.0.3:80"}

requests_total{path="/", instance="10.0.0.2:80"}
```

Select: *requests_total{path="/status"}*

Some basics

```
{name="requests_total", path="/status", instance="10.0.0.1:80"}

{name="requests_total", path="/status", instance="10.0.0.3:80"}

{name="requests_total", path="/", instance="10.0.0.2:80"}
```

Select: {*name="requests_total", path="/status”*}

Line Protocol (*simplified*)

```
cpu,host=server01,region=uswest value=1

cpu,host=server02,region=uswest value=3
```

```
{name=”cpu”, host=”server01”, region=”uswest”} 1

{name=”cpu”, host=”server02”, region=”uswest”} 3
```

Code

Creation

Creation

```go
func Open(dir string, l log.Logger, r prometheus.Registerer, opts *Options) (*DB, error)

type Options struct {
    // The interval at which the write ahead log is flushed to disc.
    WALFlushInterval time.Duration
    // Duration of persisted data to keep in milliseconds.
    RetentionDuration uint64
    // The sizes of the Blocks in milliseconds.
    BlockRanges []int64
}
```

Code

Insertion

Insertion

```
func (db *DB) Appender() Appender

type Appender interface {
    Add(series labels.Labels, t int64, v float64) (ref string, err error)
    // Add adds a sample pair for the referenced series. It is generally faster
    // than adding a sample by providing its full label set.
    AddFast(ref string, t int64, v float64) error
    // Commit submits the collected samples and purges the batch.
    Commit() error
    // Rollback rolls back all modifications made in the appender so far.
    Rollback() error
}
```

Insertion

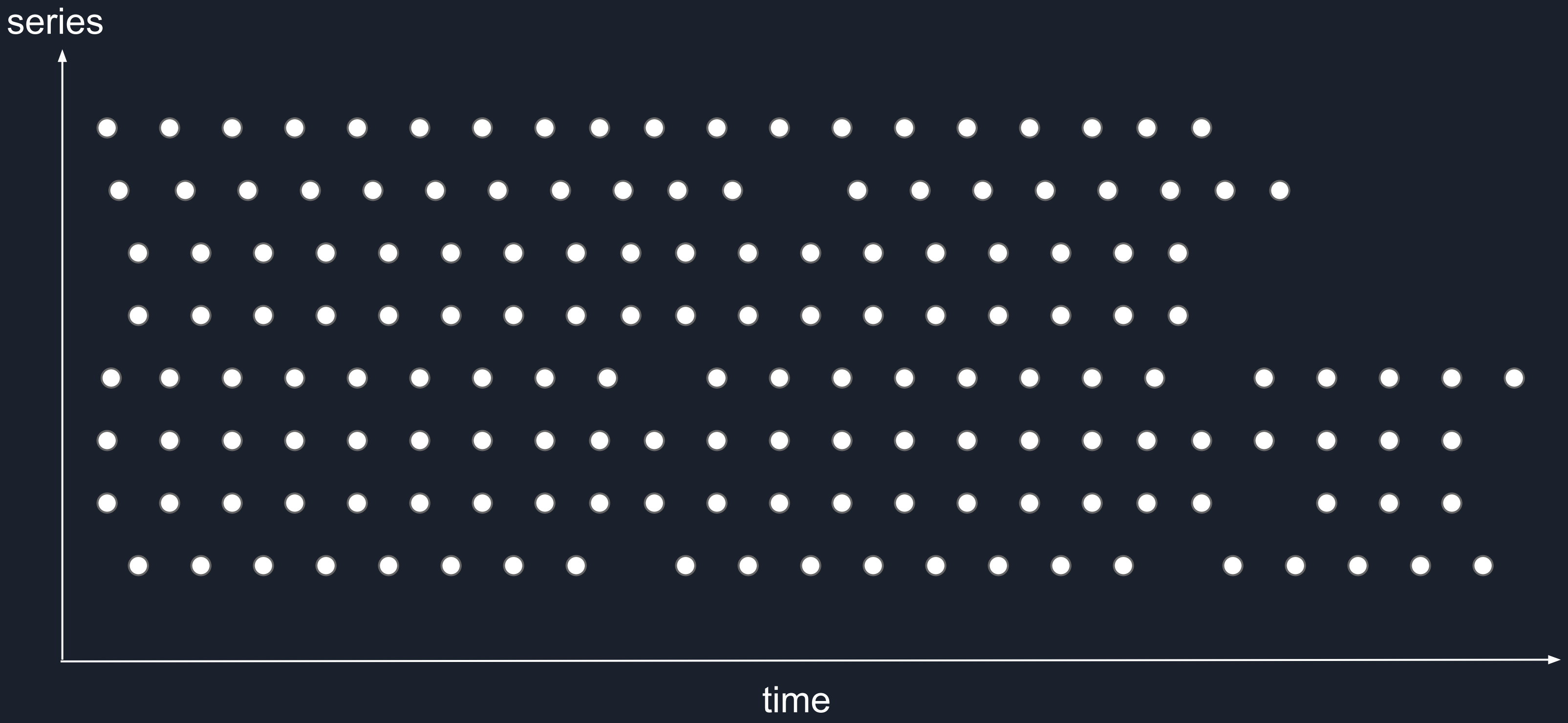
```
func (db *DB) Appender() Appender

type Appender interface {
    Add(series labels.Labels, t int64, v float64) (ref string, err error)
    // Add adds a sample pair for the referenced series. It is generally faster
    // than adding a sample by providing its full label set.
    AddFast(ref string, t int64, v float64) error
    // Commit submits the collected samples and purges the batch.
    Commit() error
    // Rollback rolls back all modifications made in the appender so far.
    Rollback() error
}
```

Appender: Ordering

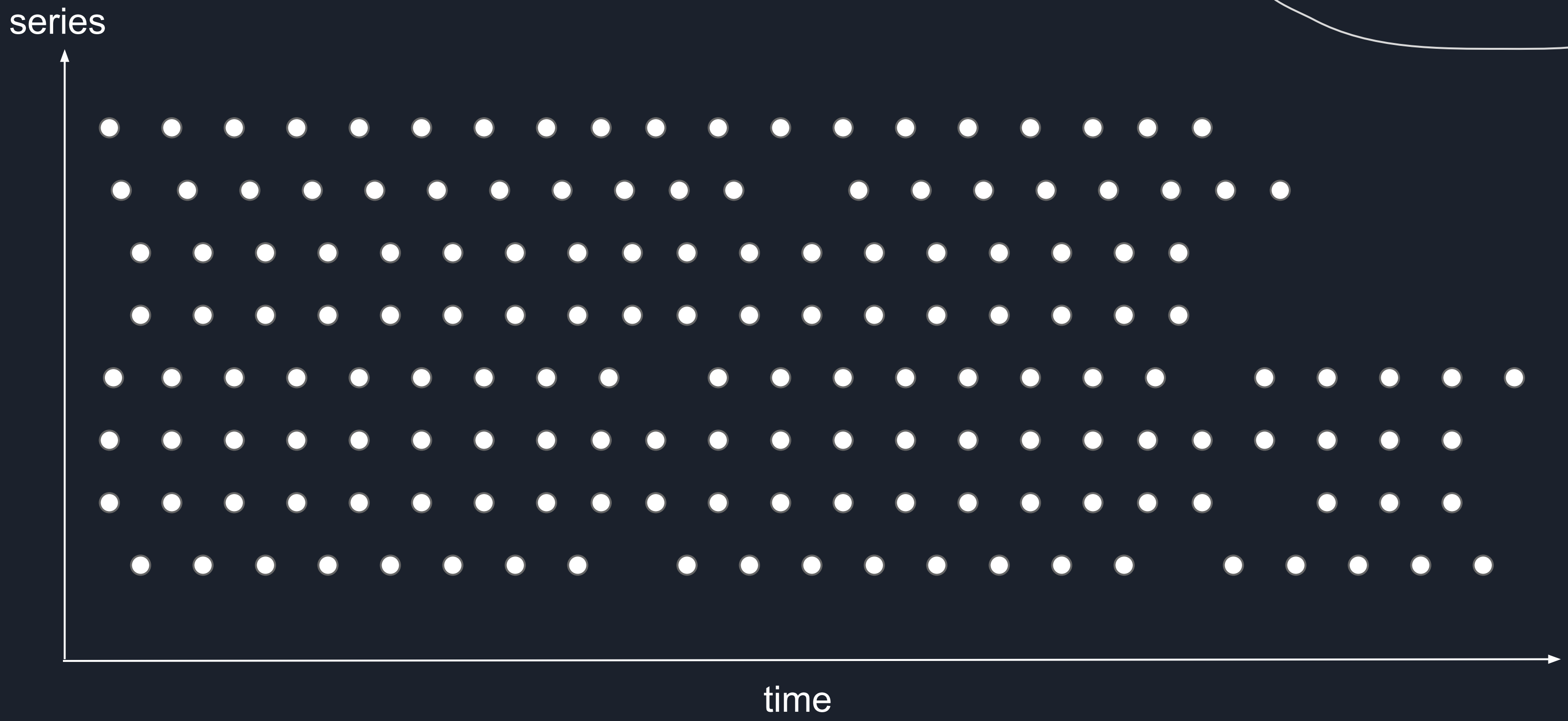
The samples of **each series** need to be ordered.

```
Add(ser1,  10, 4)  → ✔

Add(ser1,  15, 7)  → ✔

Add(ser2,  10, 7)  → ✔

Add(ser1,  12, 7)  → ✘
```

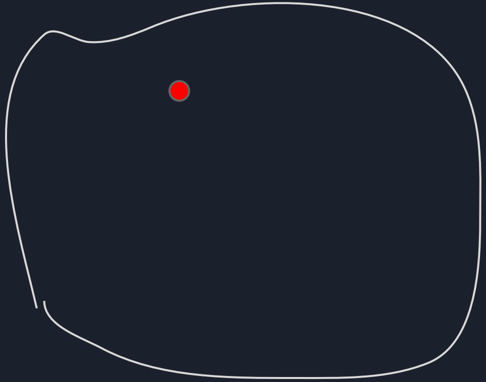
Appender

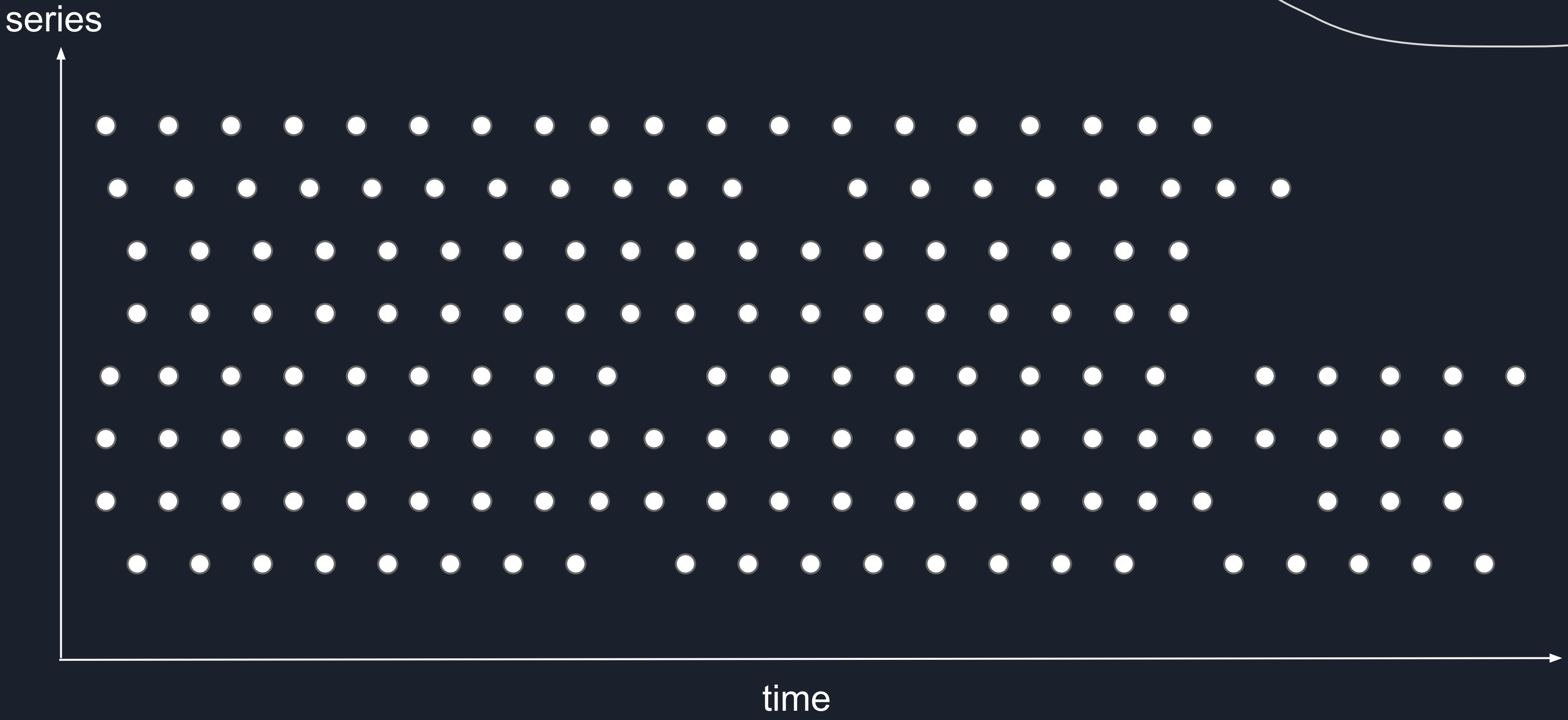


Appender

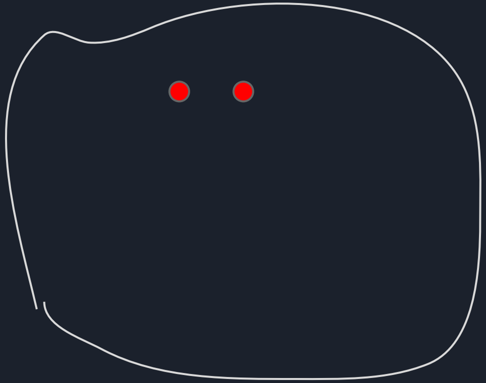


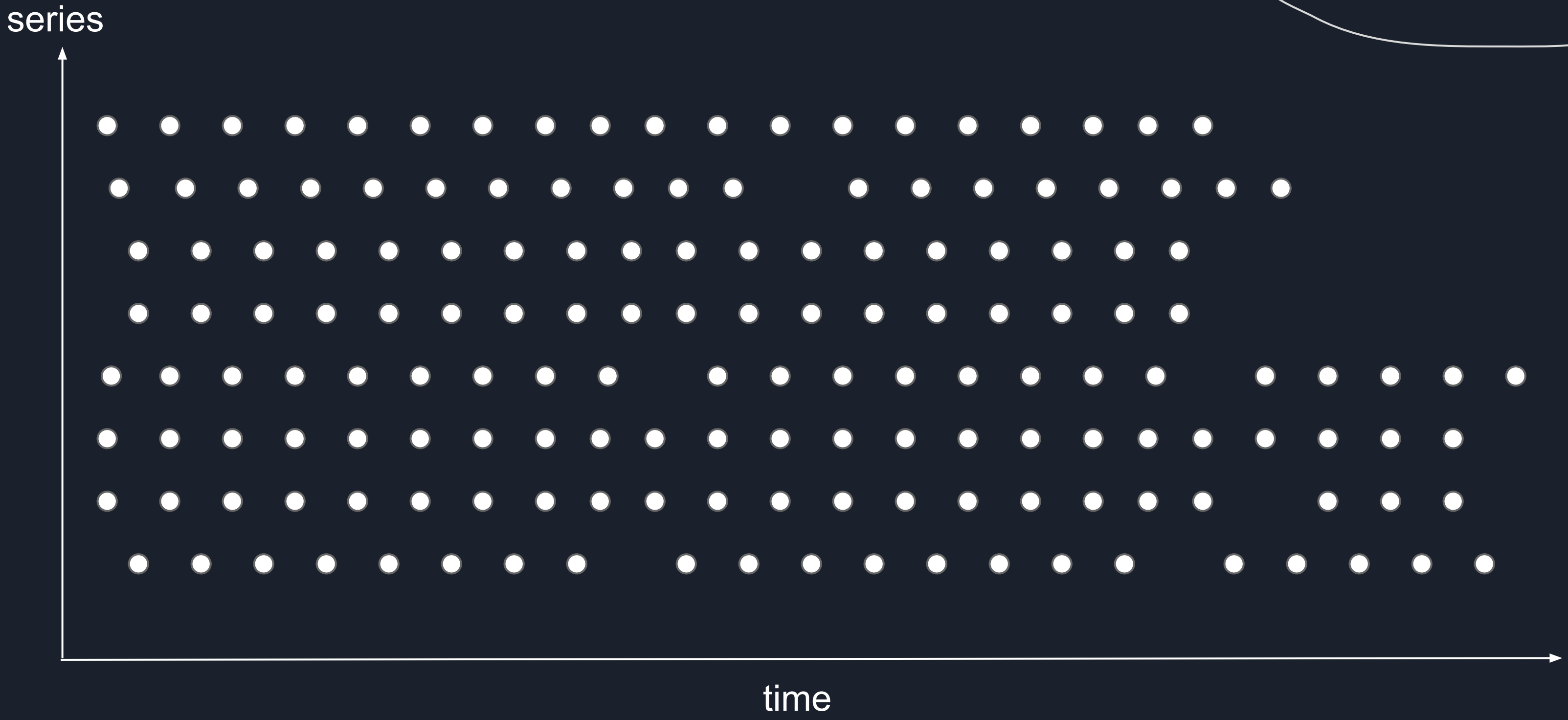
Appender



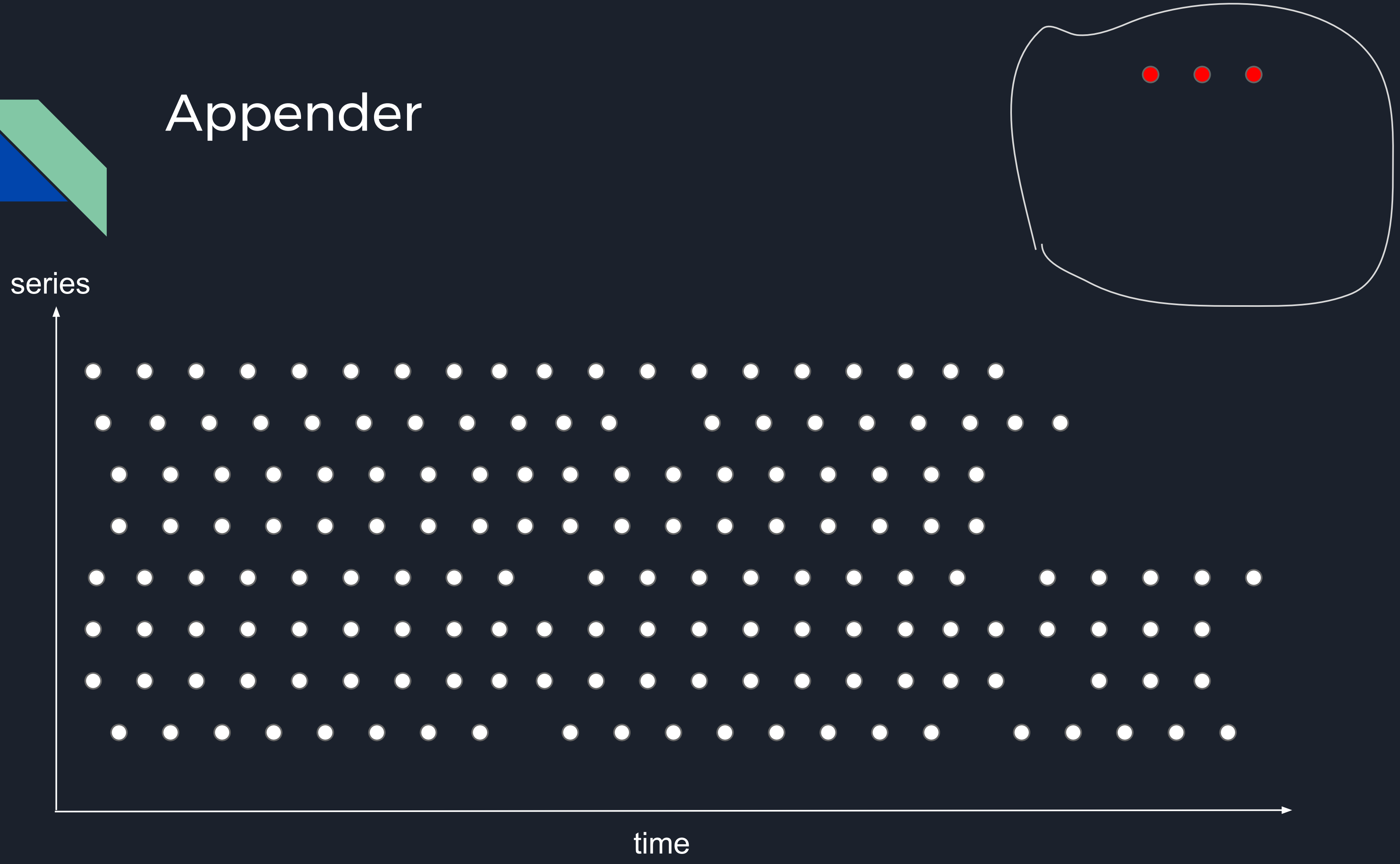


Appender

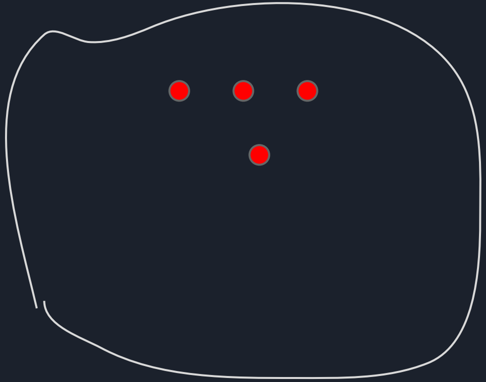


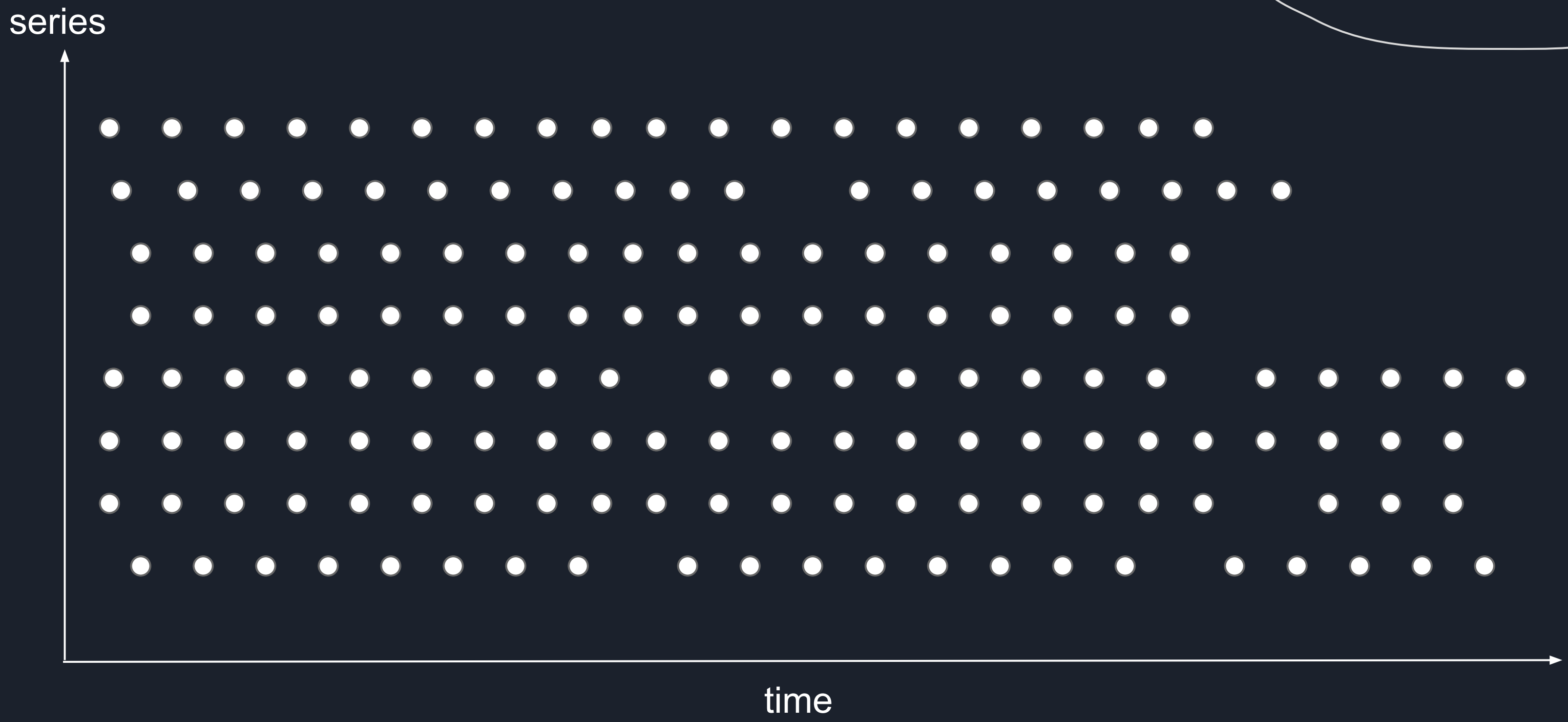


Appender

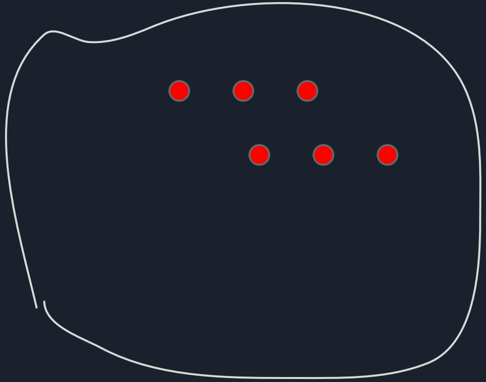


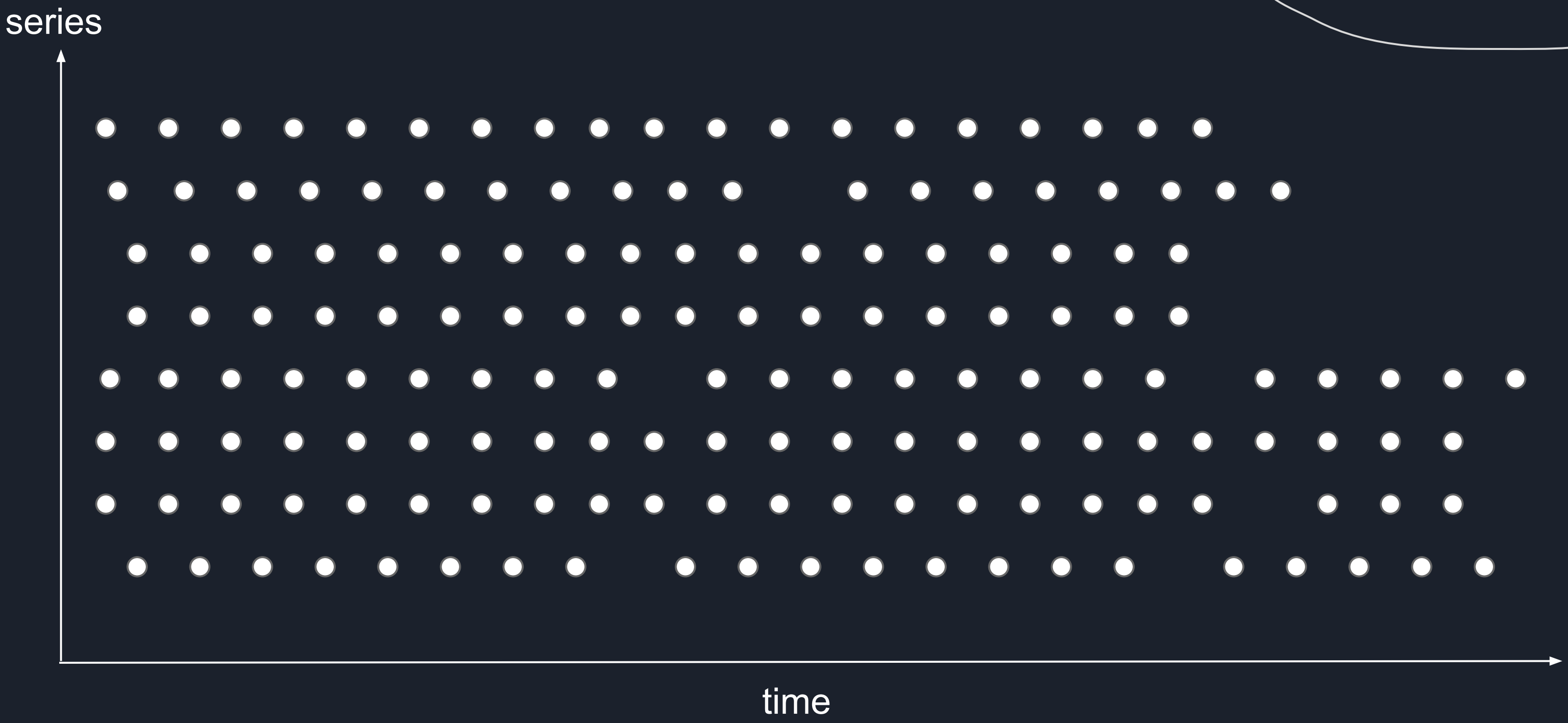
Appender



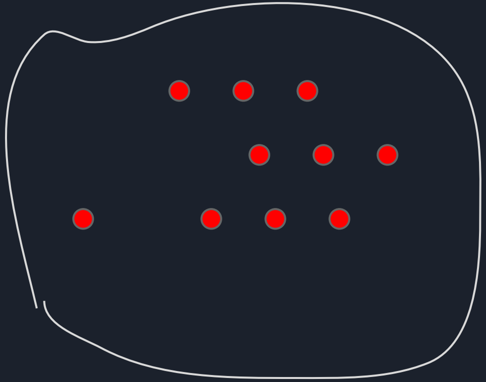


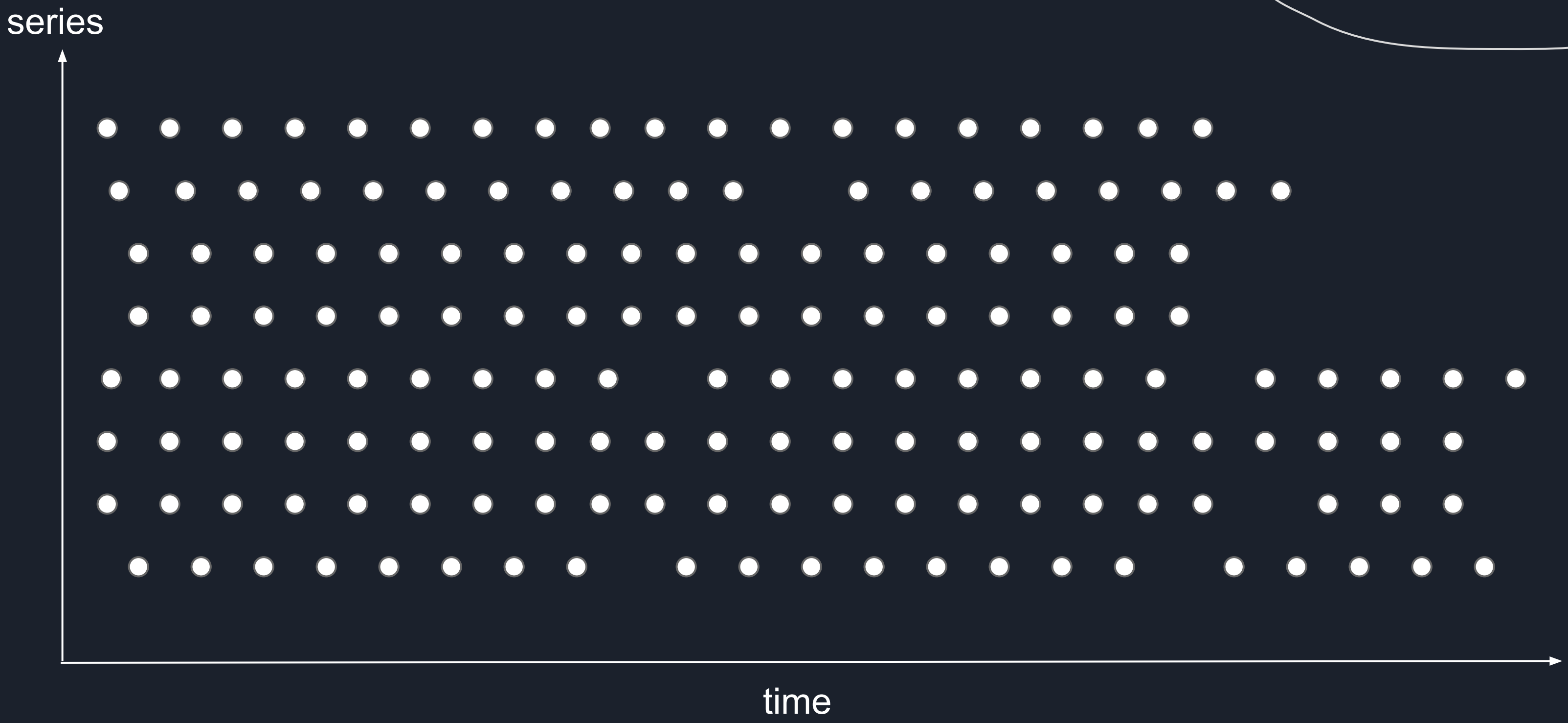
Appender



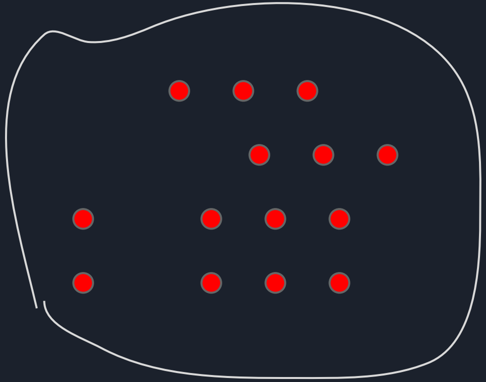


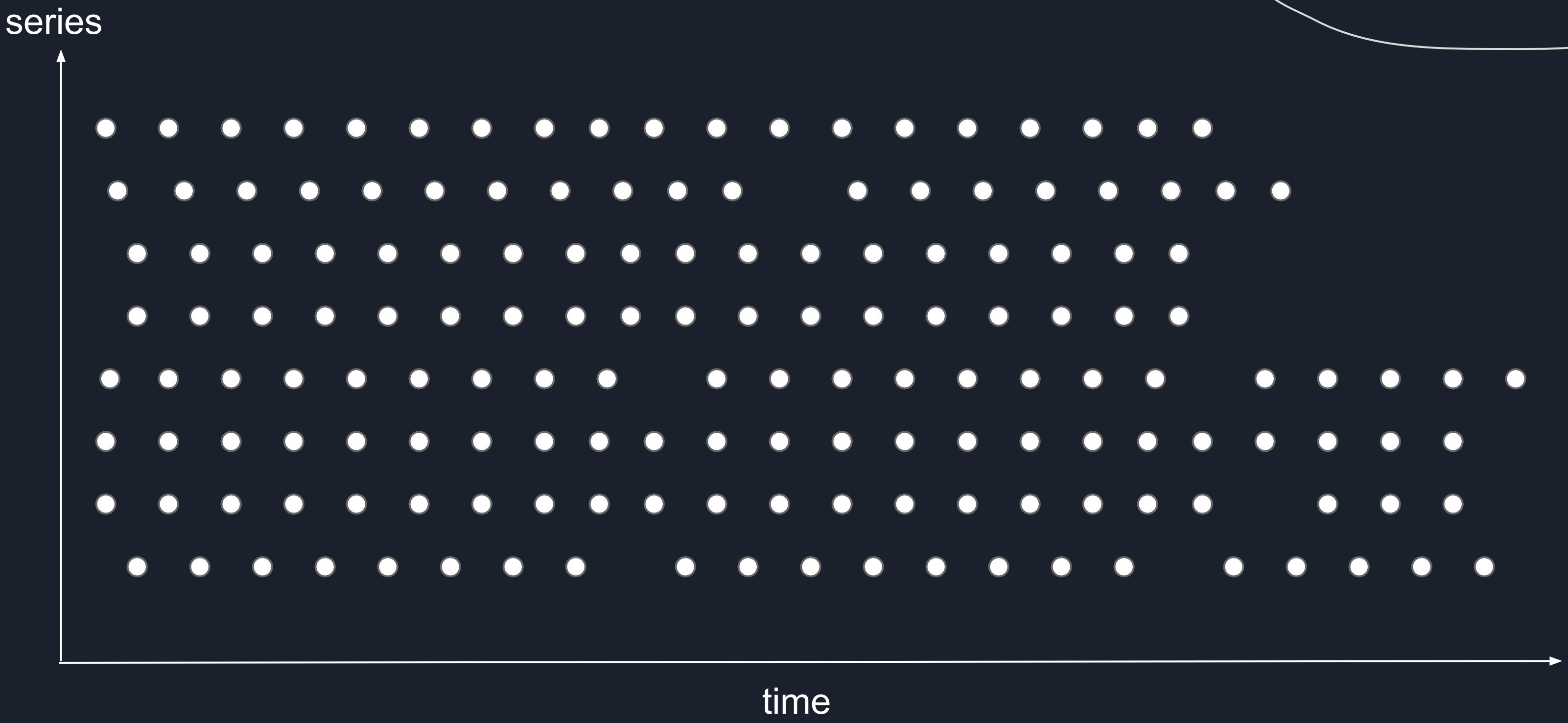
Appender



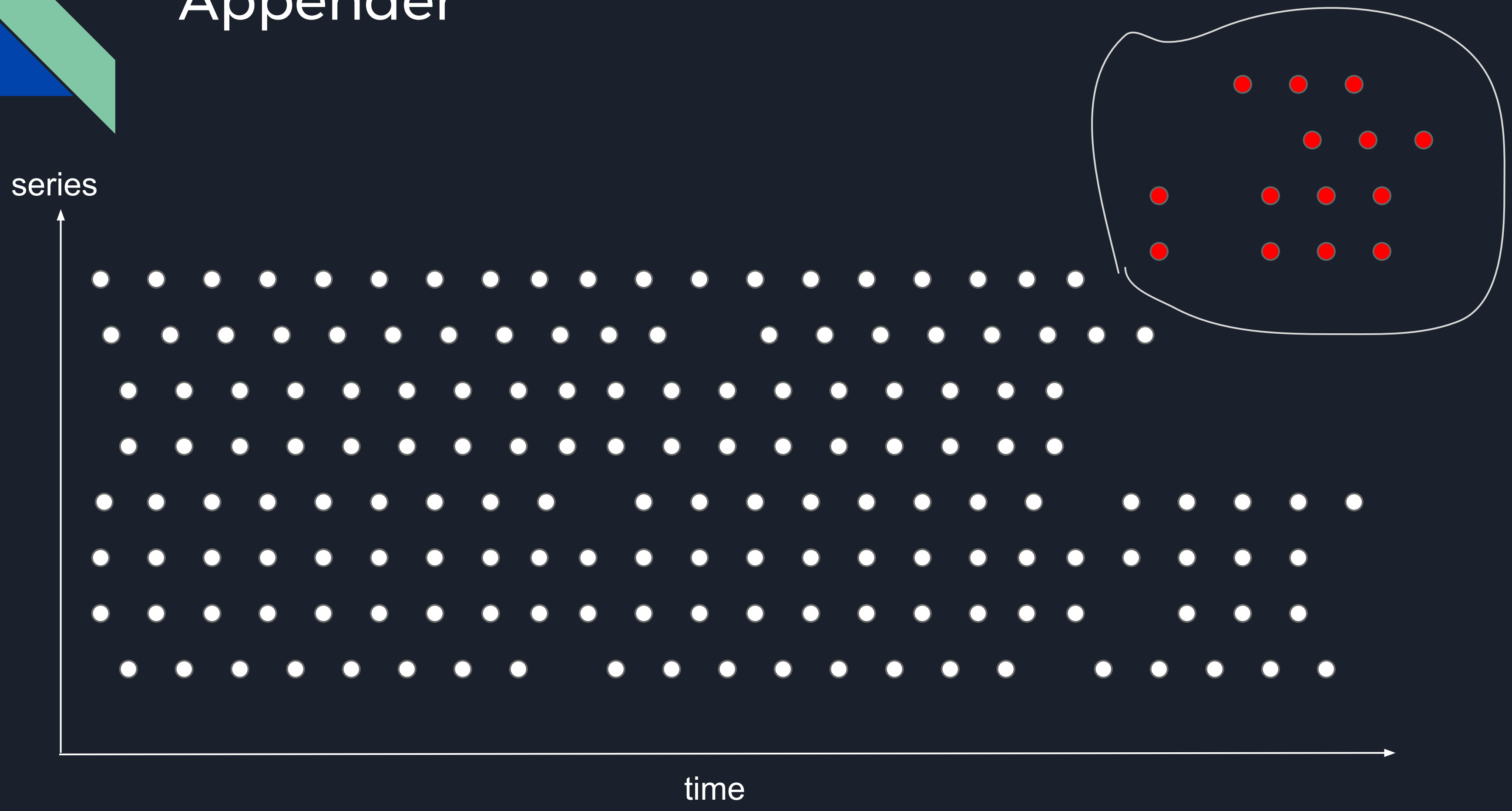


Appender

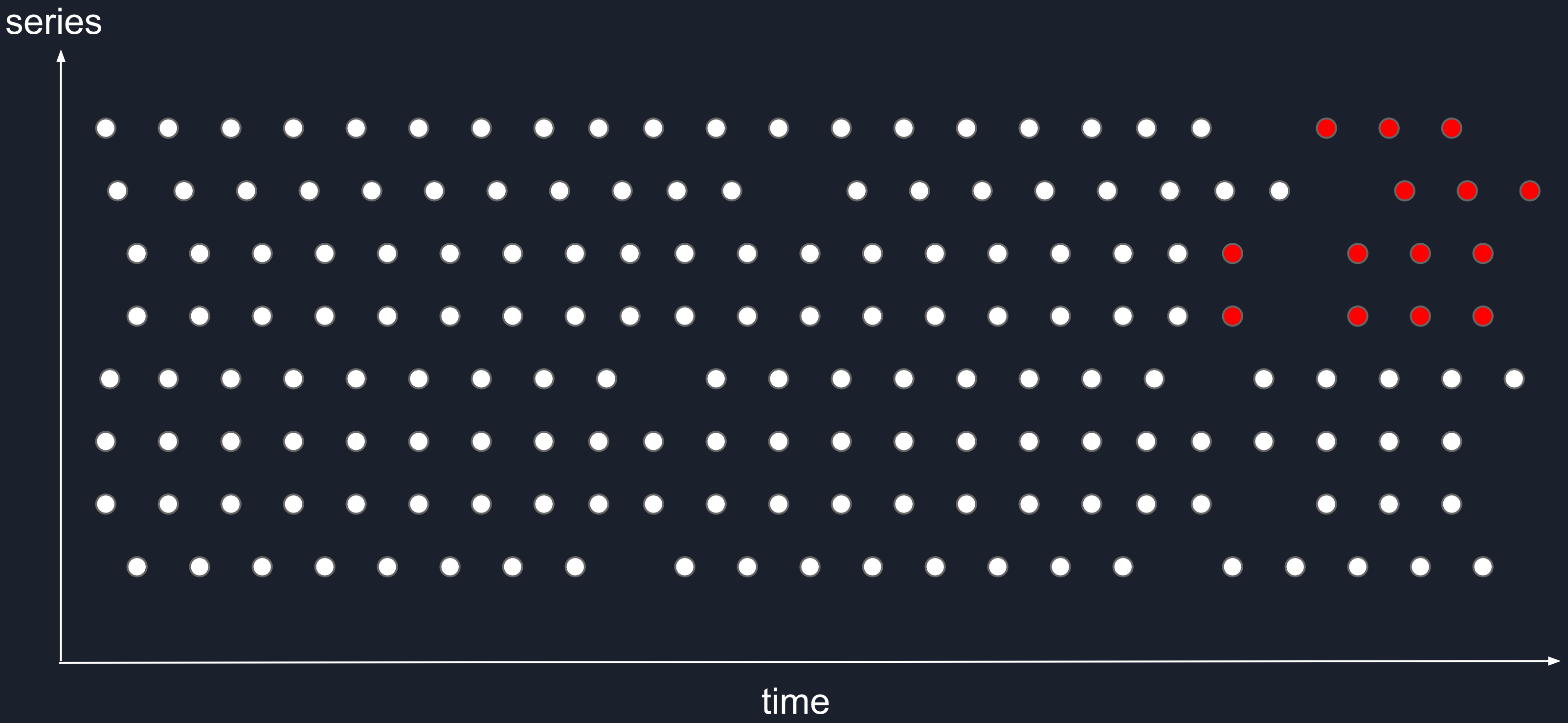




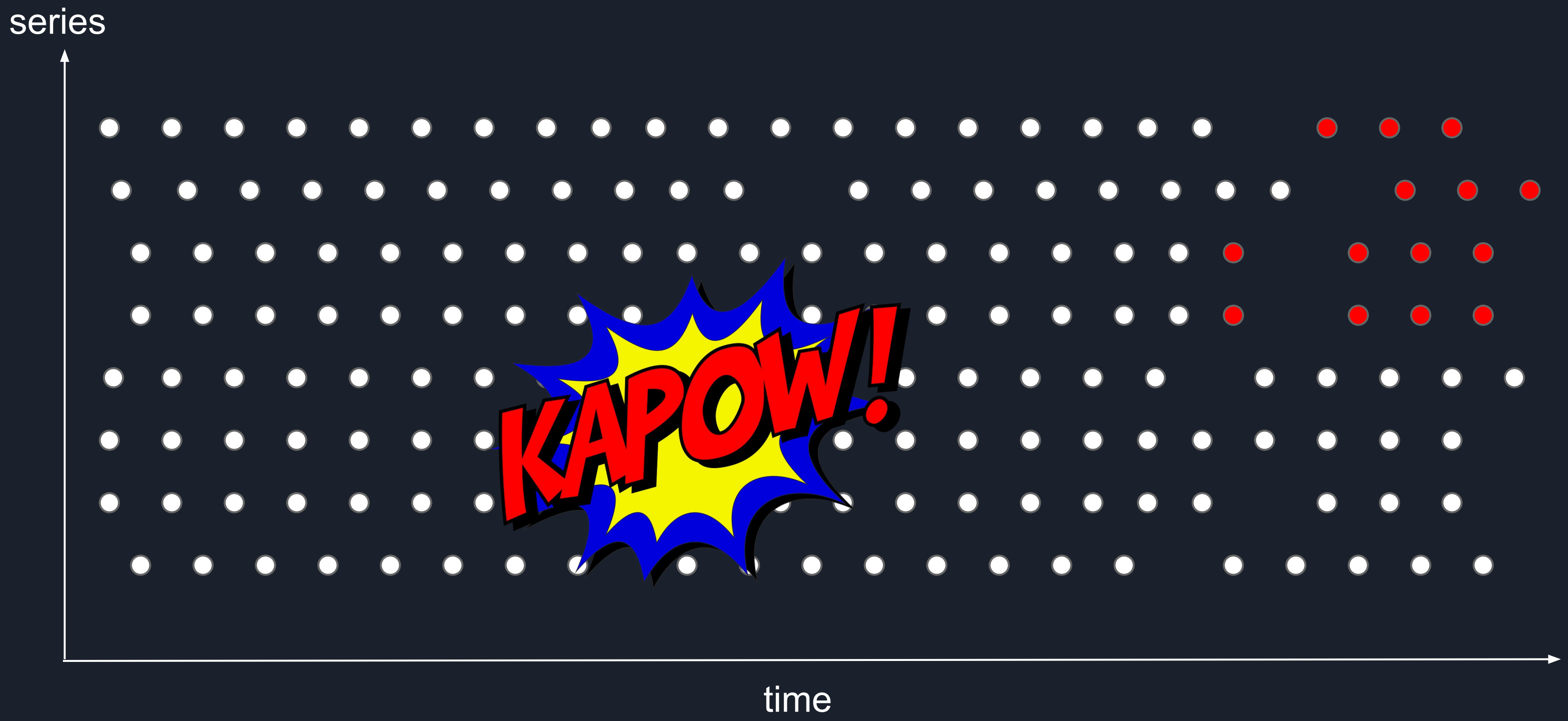
Appender



Appender



Appender



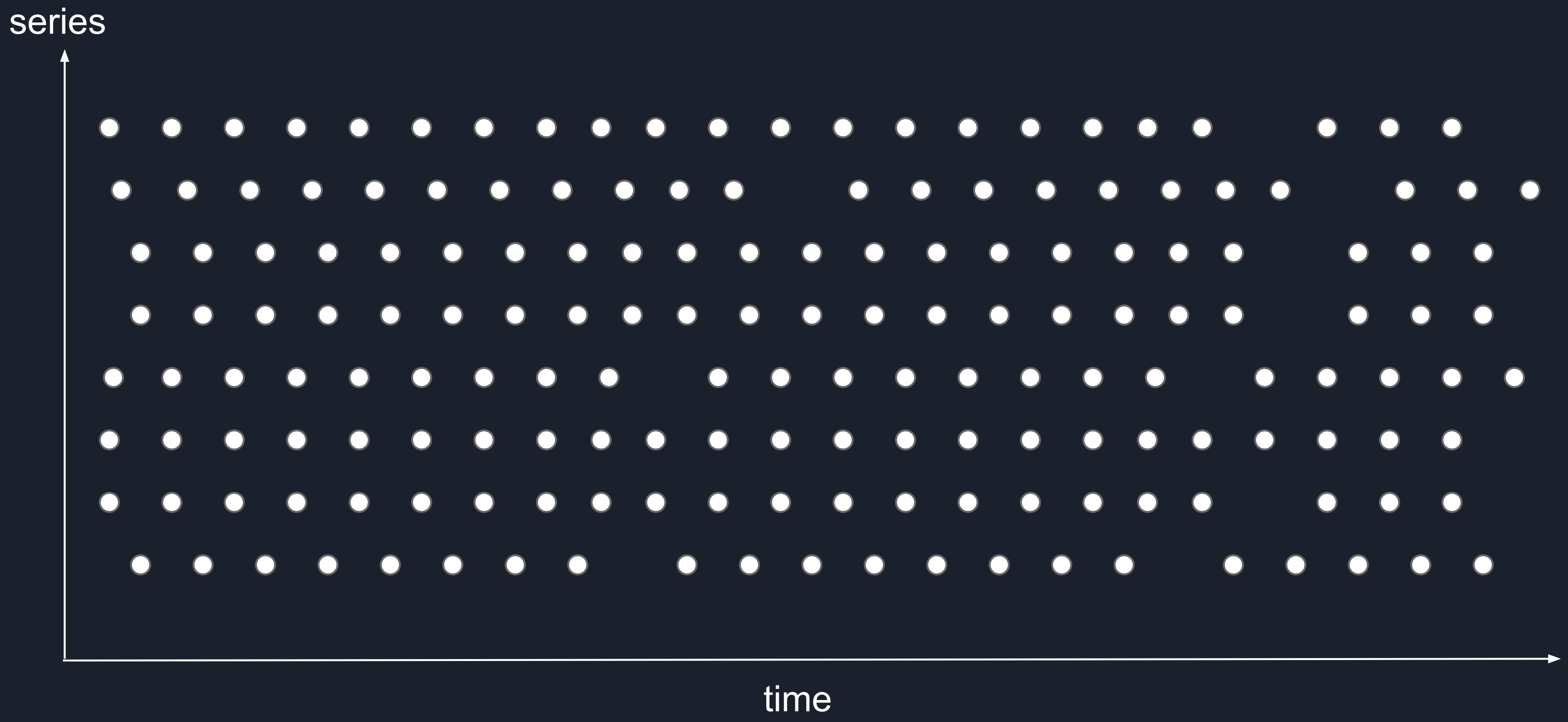
Util

```
func LineToMetrics(buf []byte) ([]Metric, error)

type Metric struct {
    Series labels.Labels


    Timestamp int64
    Value float64
}
```

Code

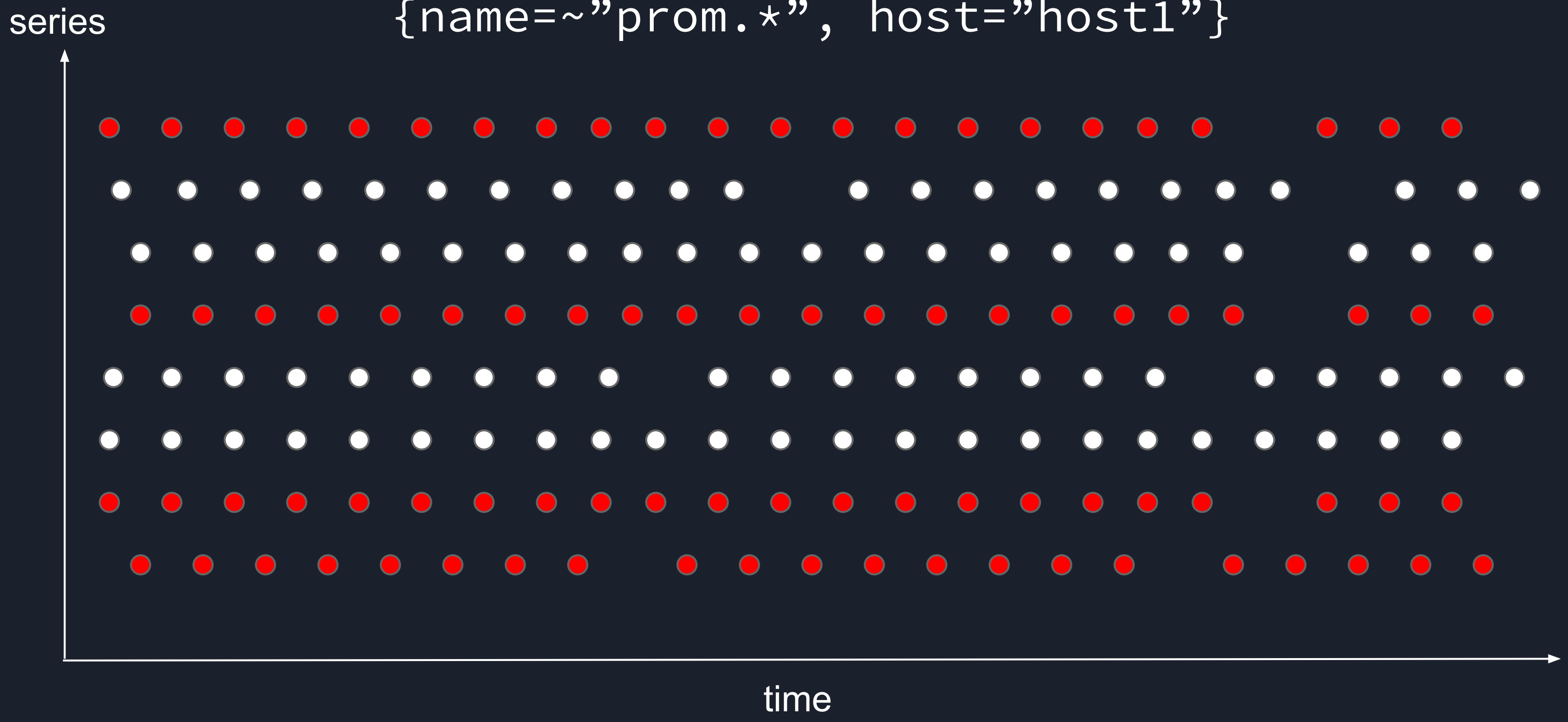
Querying

Querying



Querying

```
{name=~"prom.*", host="host1"}
```



Querying: Matcher

```
// {name=~”prom.*”, host=”host1”}

type Matcher interface {
    // Name returns the label name the matcher should apply to.
    Name() string
    // Matches checks whether a value fulfills the constraints.
    Matches(v string) bool
}
```

Querying: Matcher

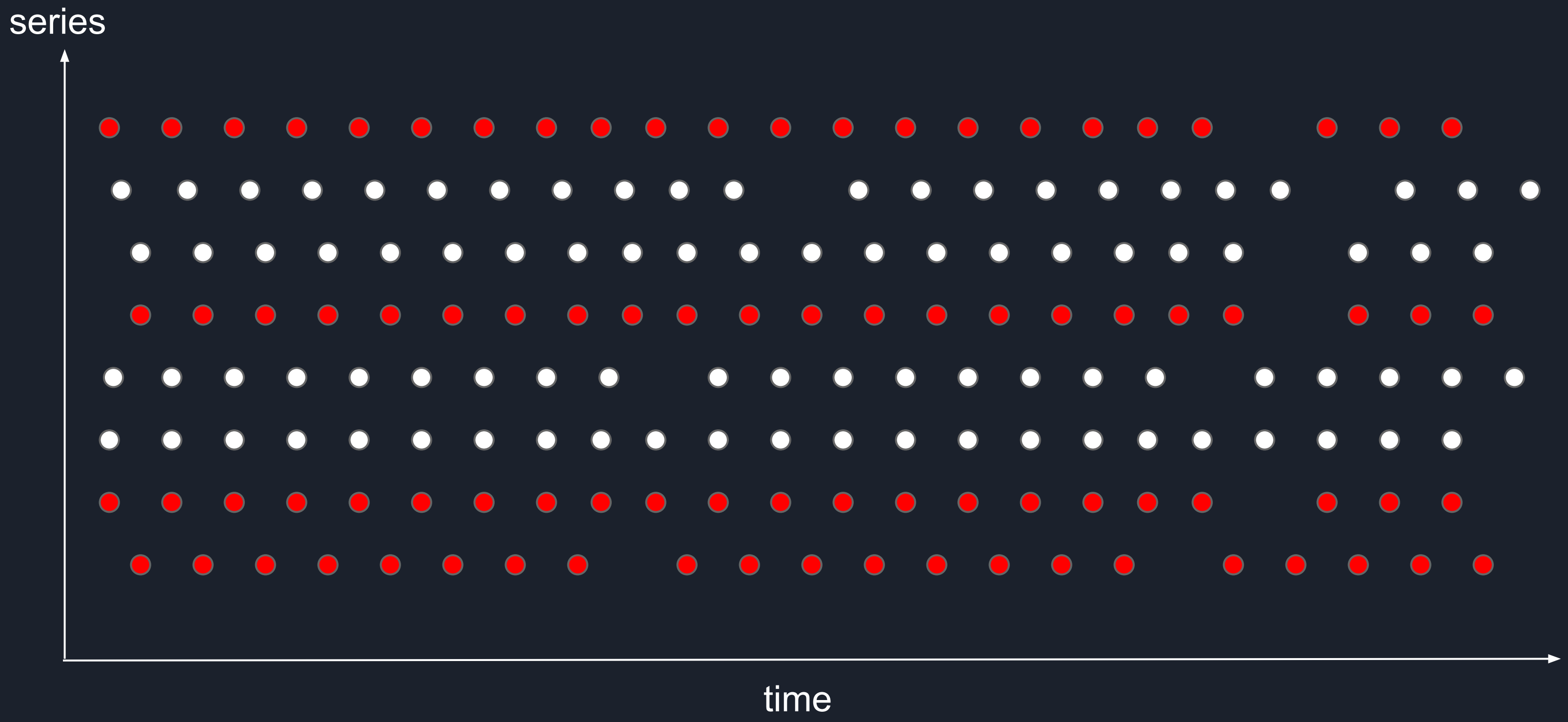
```
em := labels.NewEqualMatcher(“name”, “prometheus”)  // {name=”prometheus”}
em.Matches(“prometheus”) // → true
em.Matches(“influx”)  // → false

// Check if a series has a label m.Name(), if yes, then call m.Matches() on
label value. If it matches then the series is Matched.



// So em matches all series that have {name=”prometheus”} as a label.
```

Querying: Matcher

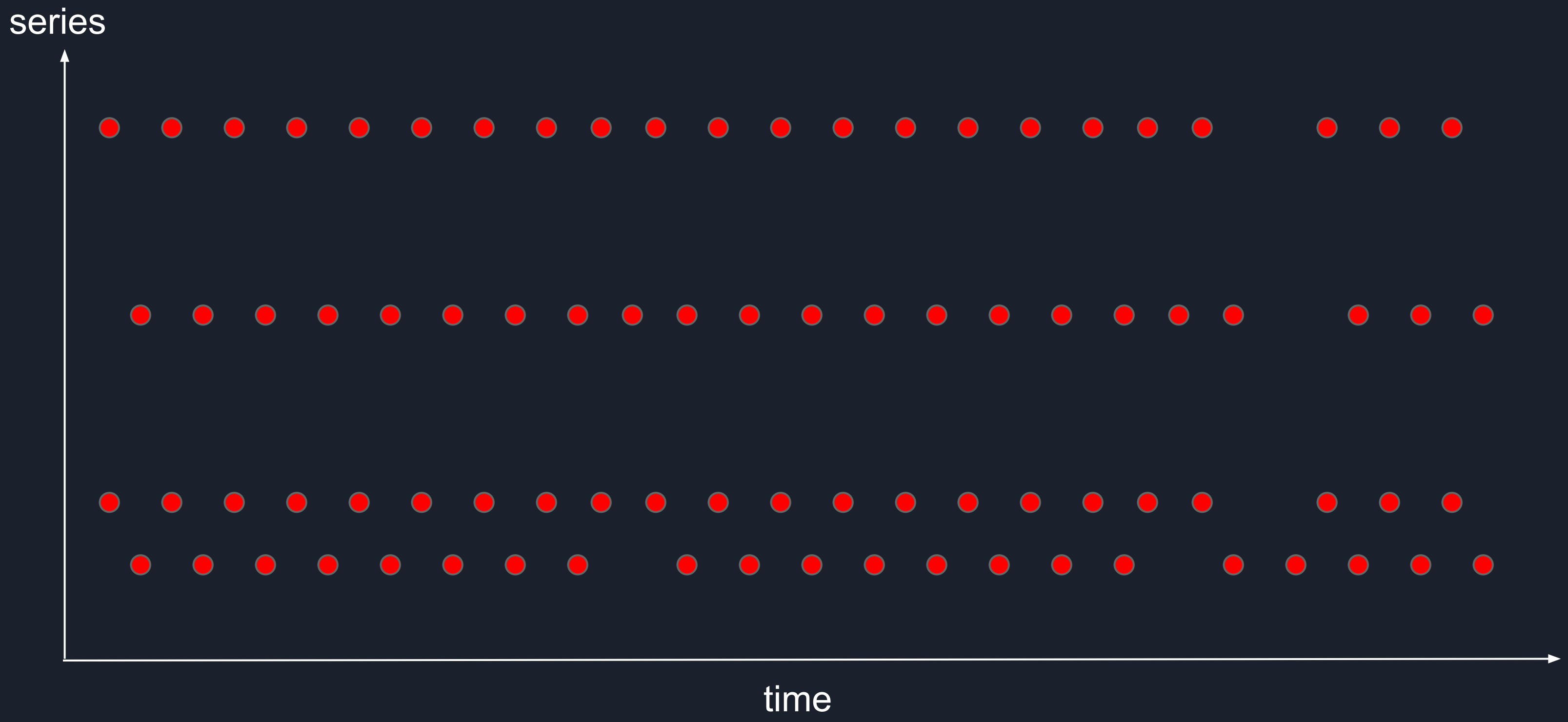
```
regM := labels.NewRegExpMatcher(“name”, “prom.*”)  // {name=~”prom.*”}
regM.Matches(“prometheus”) // → true
regM.Matches(“promflux”)  // → true
regM.Matches(“influx”)  // → false
```

```
Select([]labels.Matcher) SeriesSet // The set of series that match all
the given Matchers
```

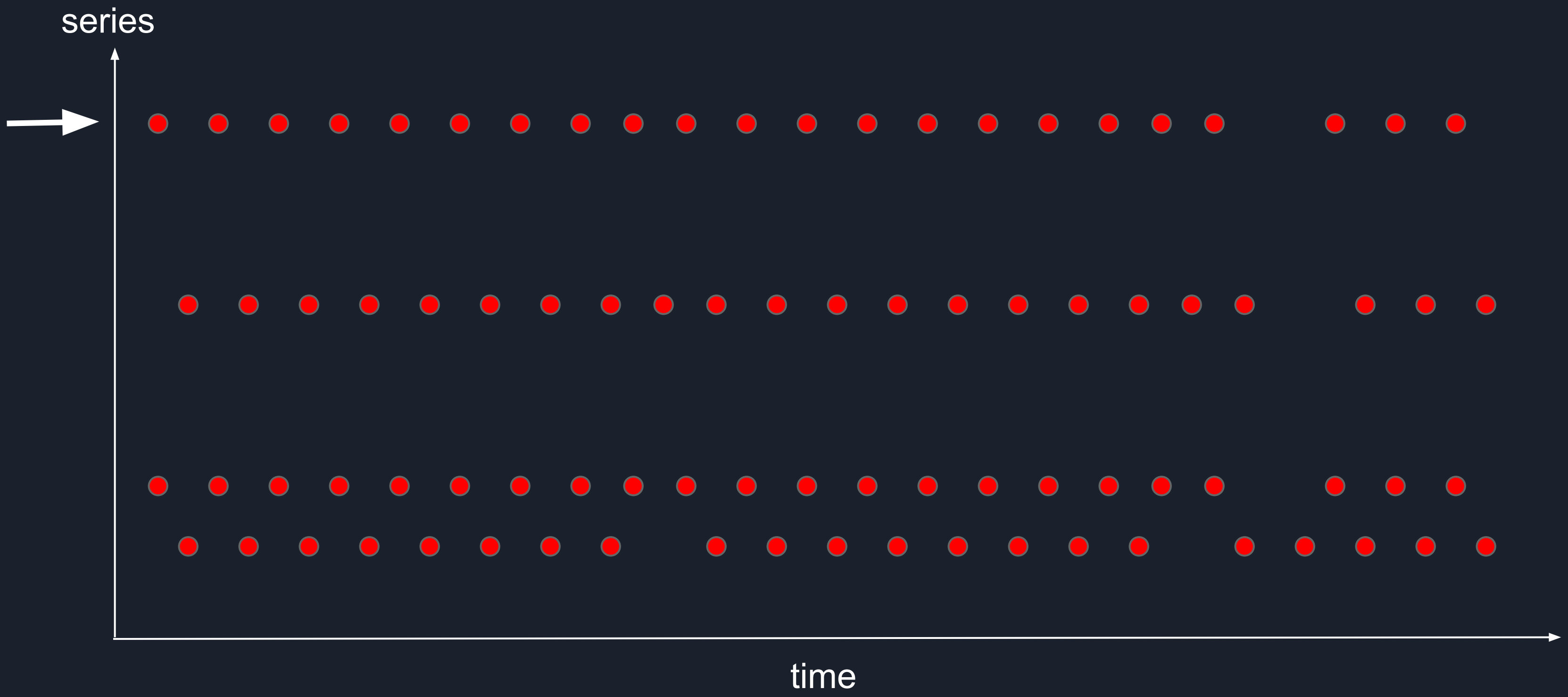
Querying



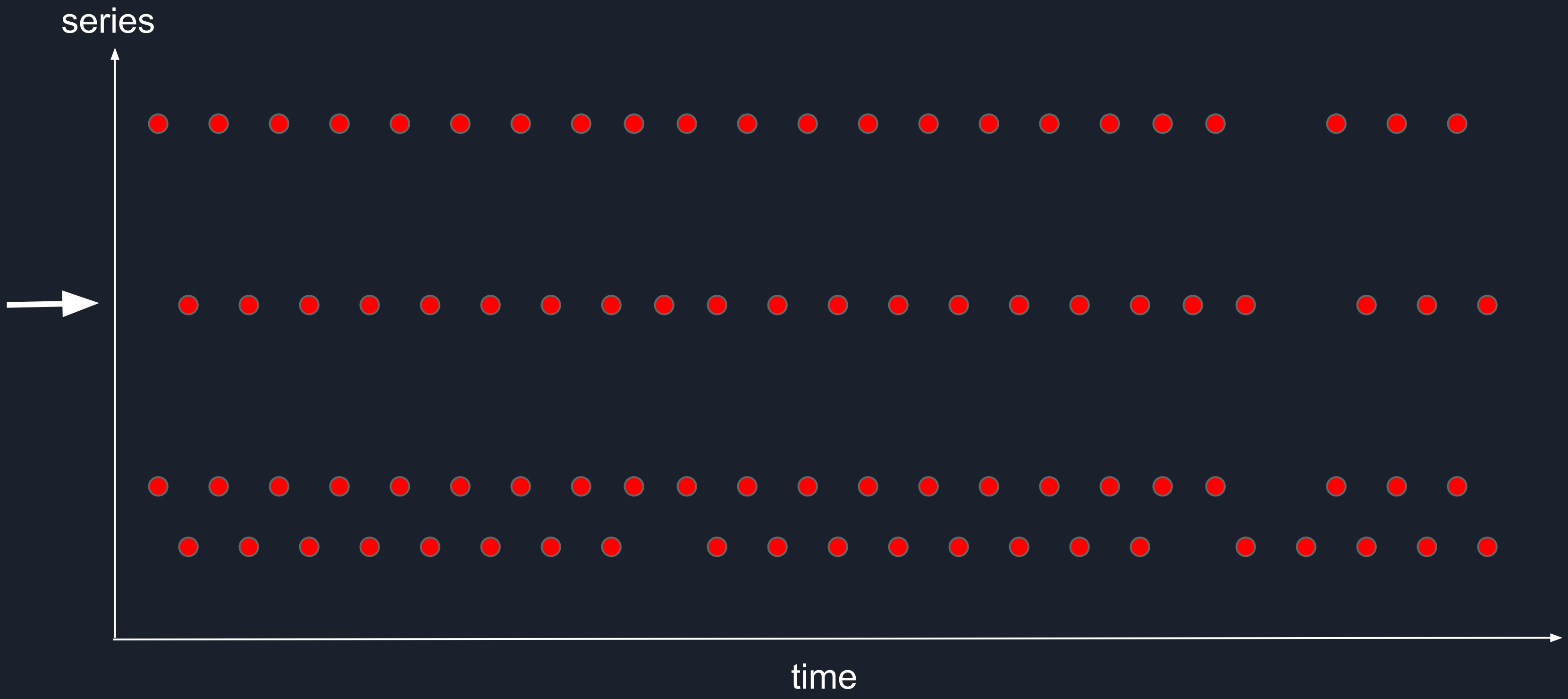
Querying



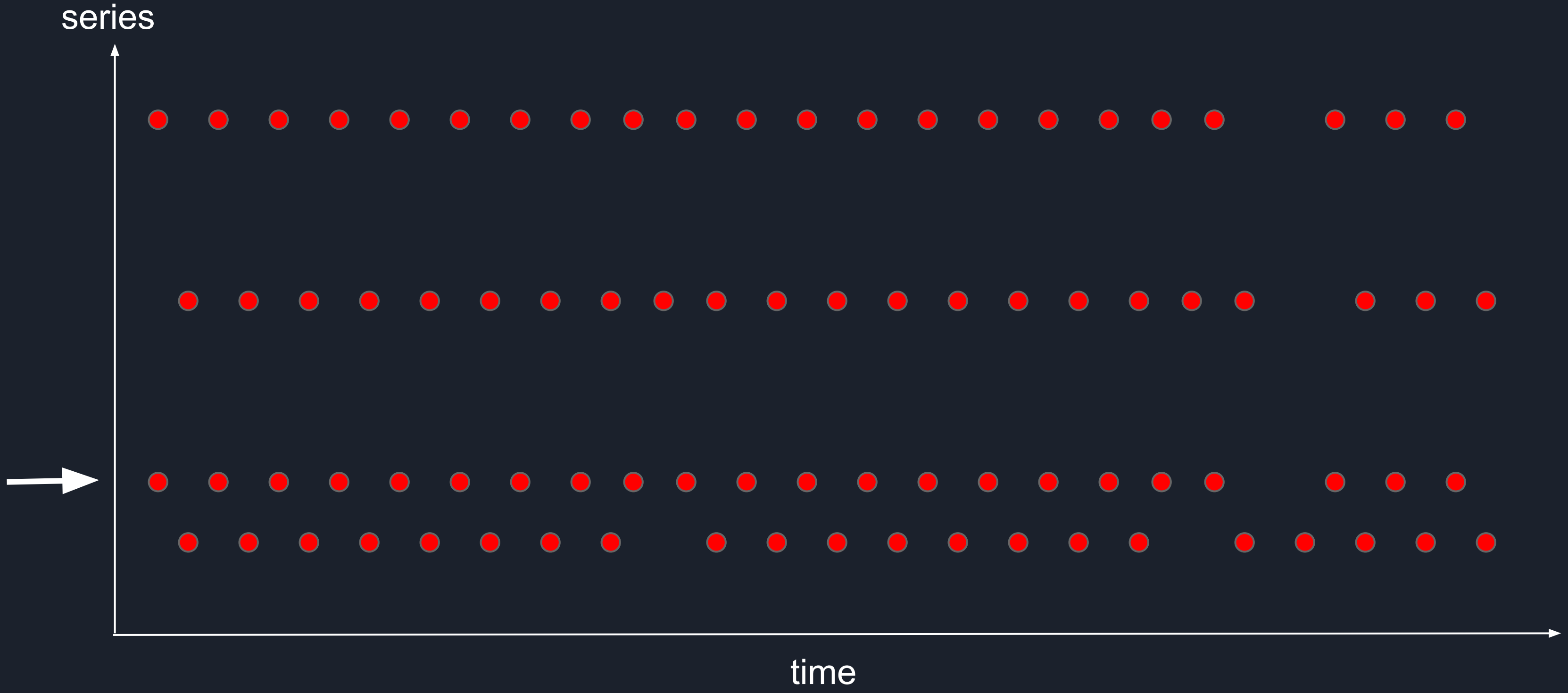
Querying



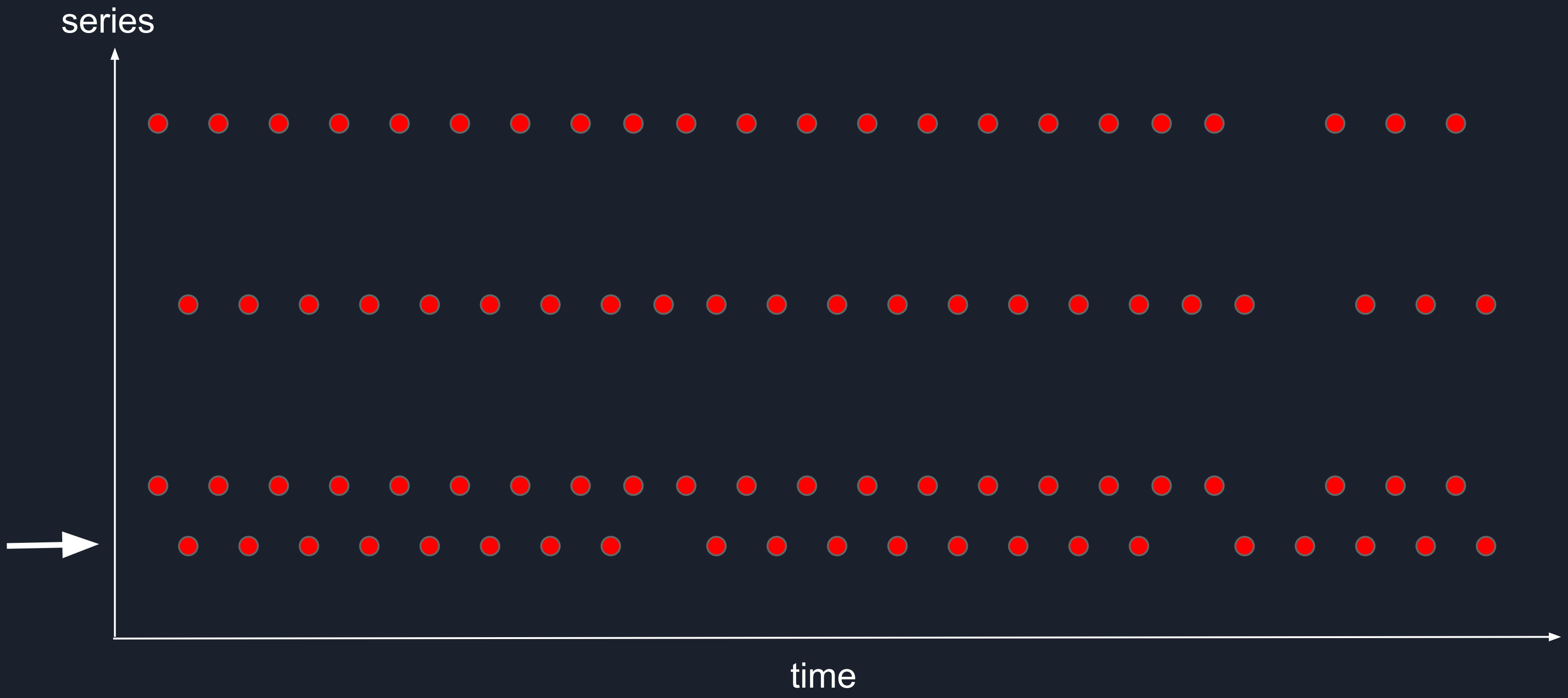
Querying



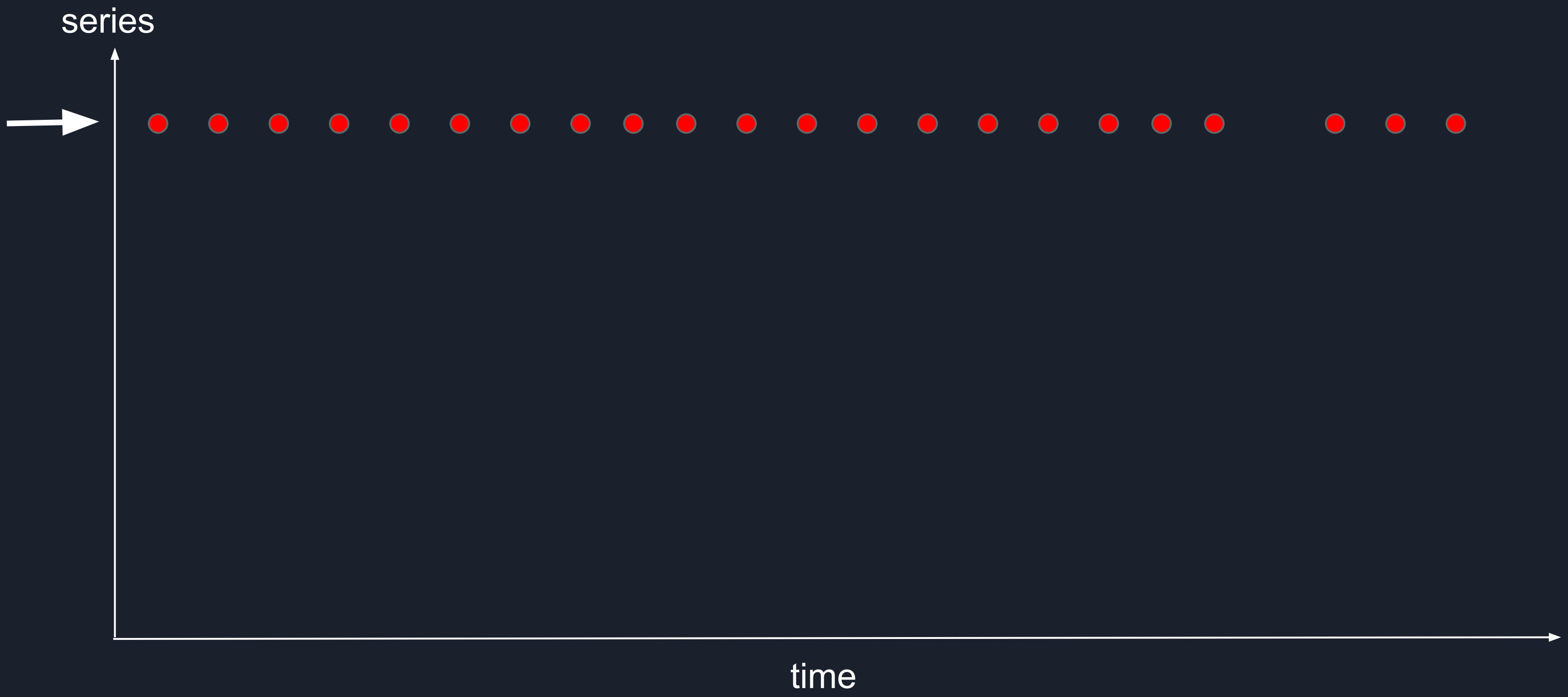
Querying



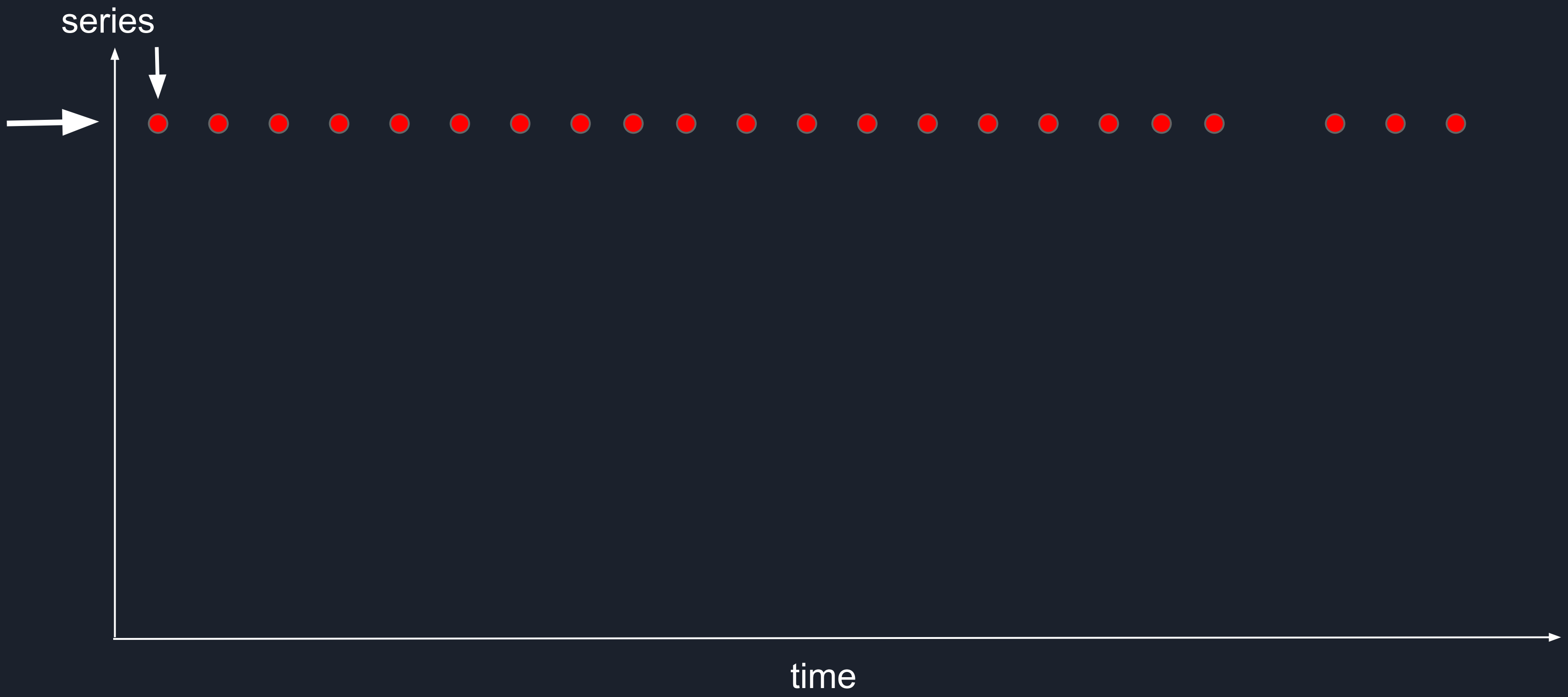
Querying



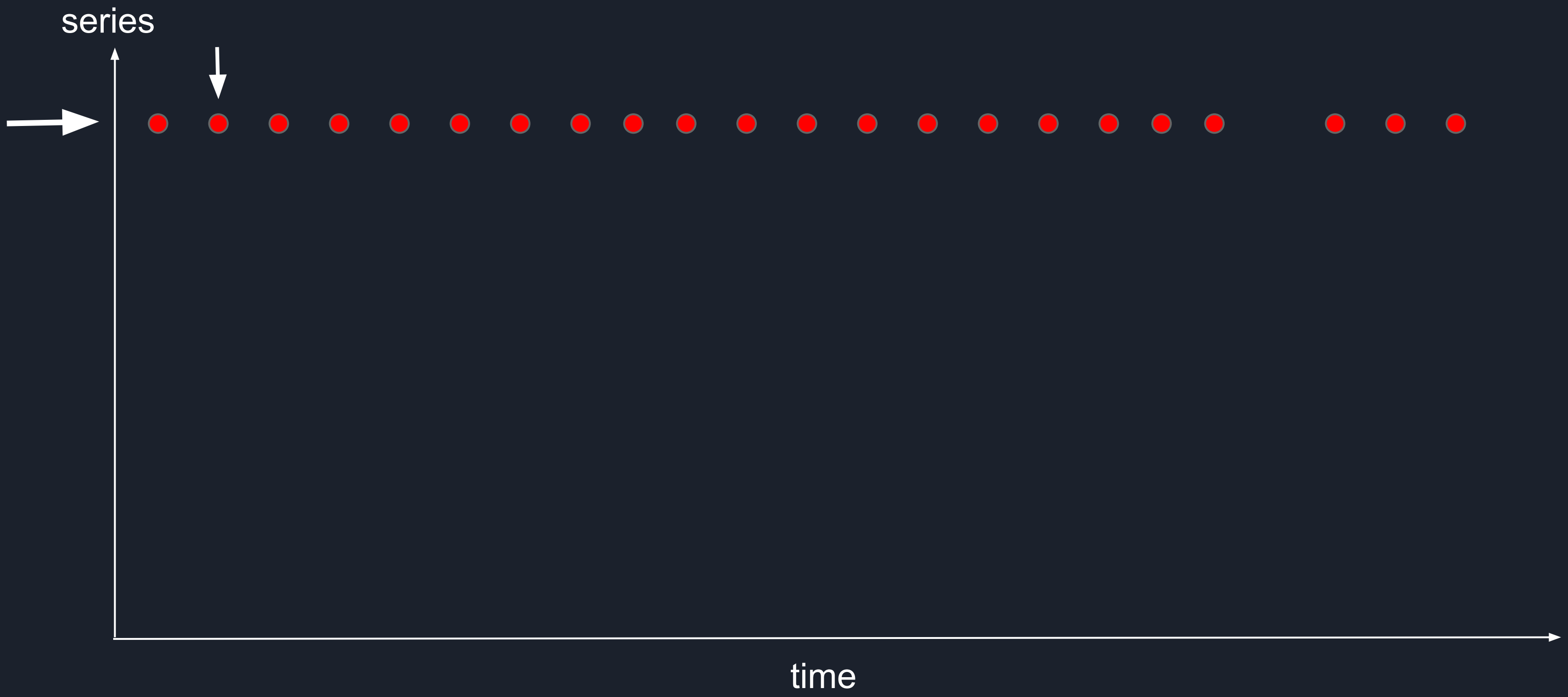
Querying



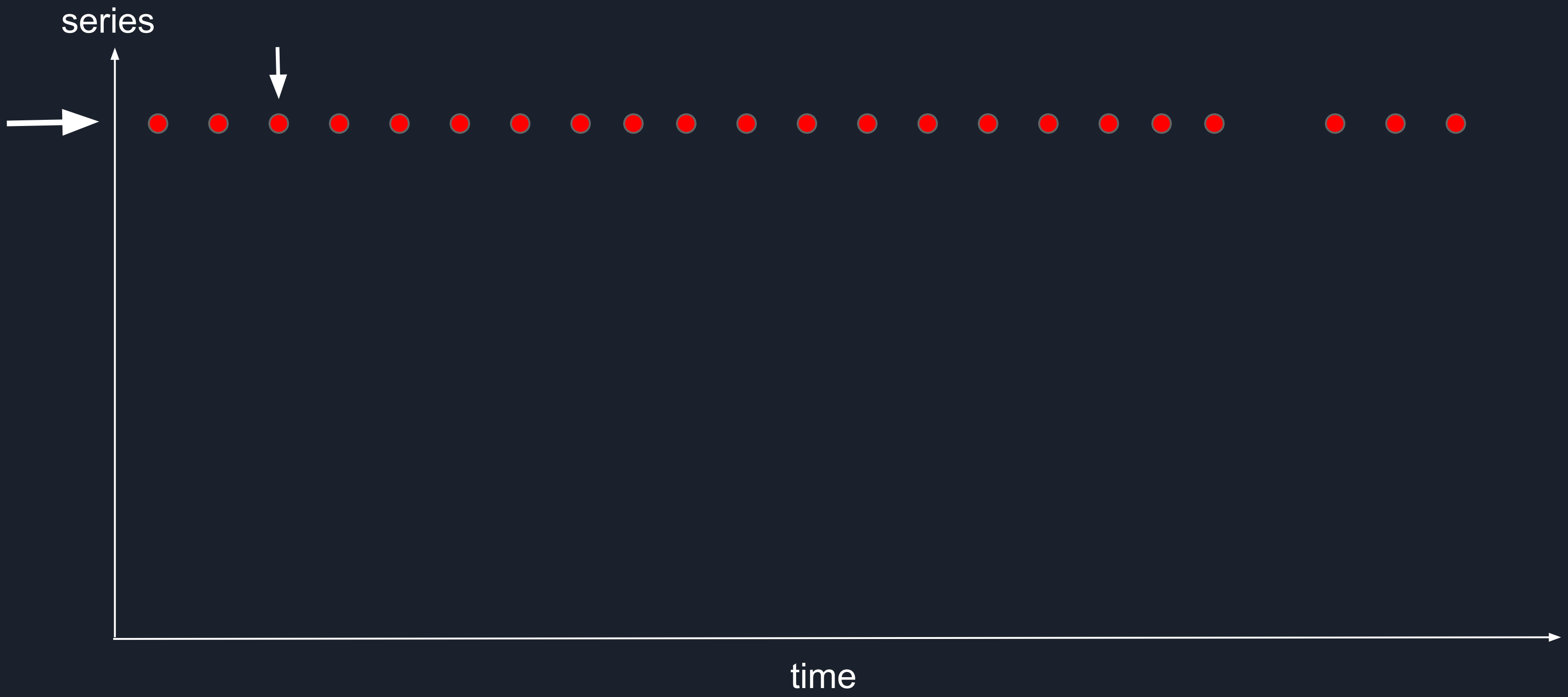
Querying



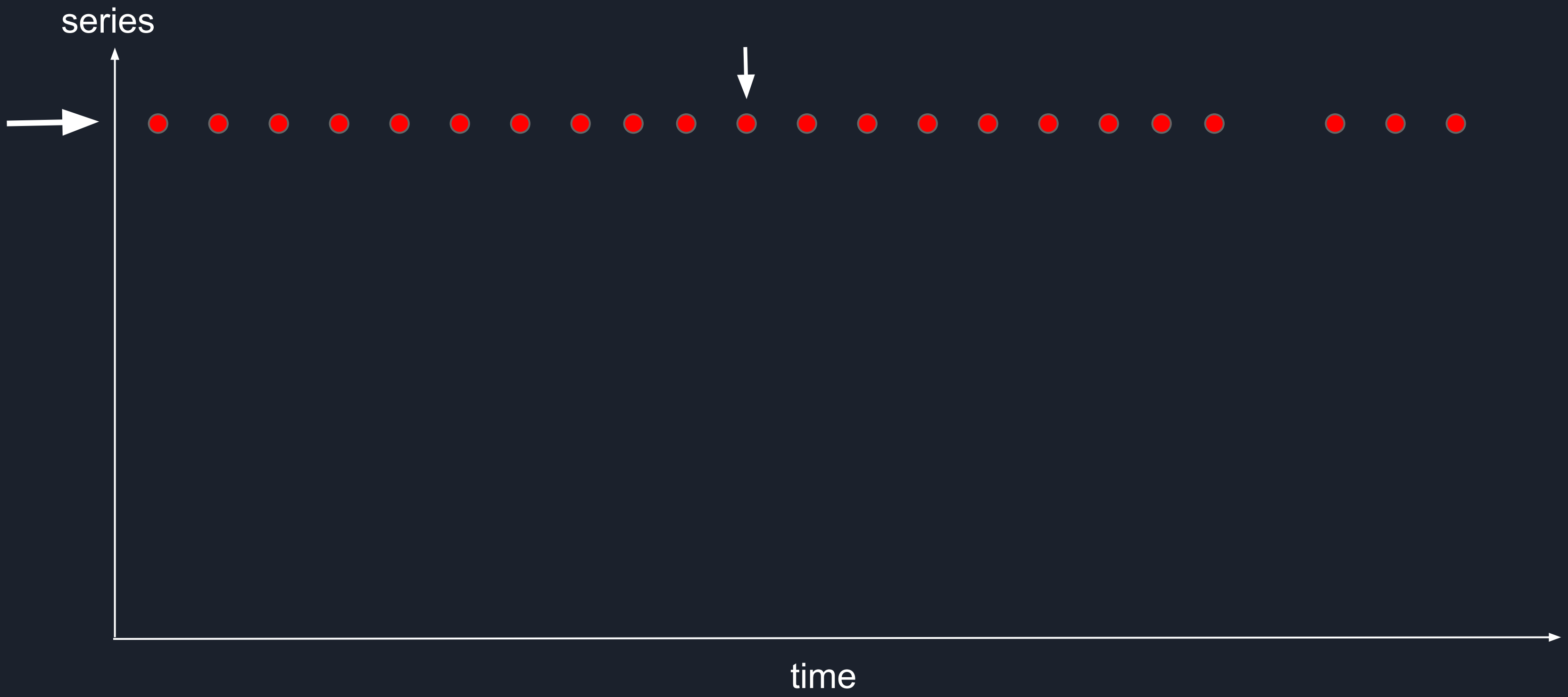
Querying



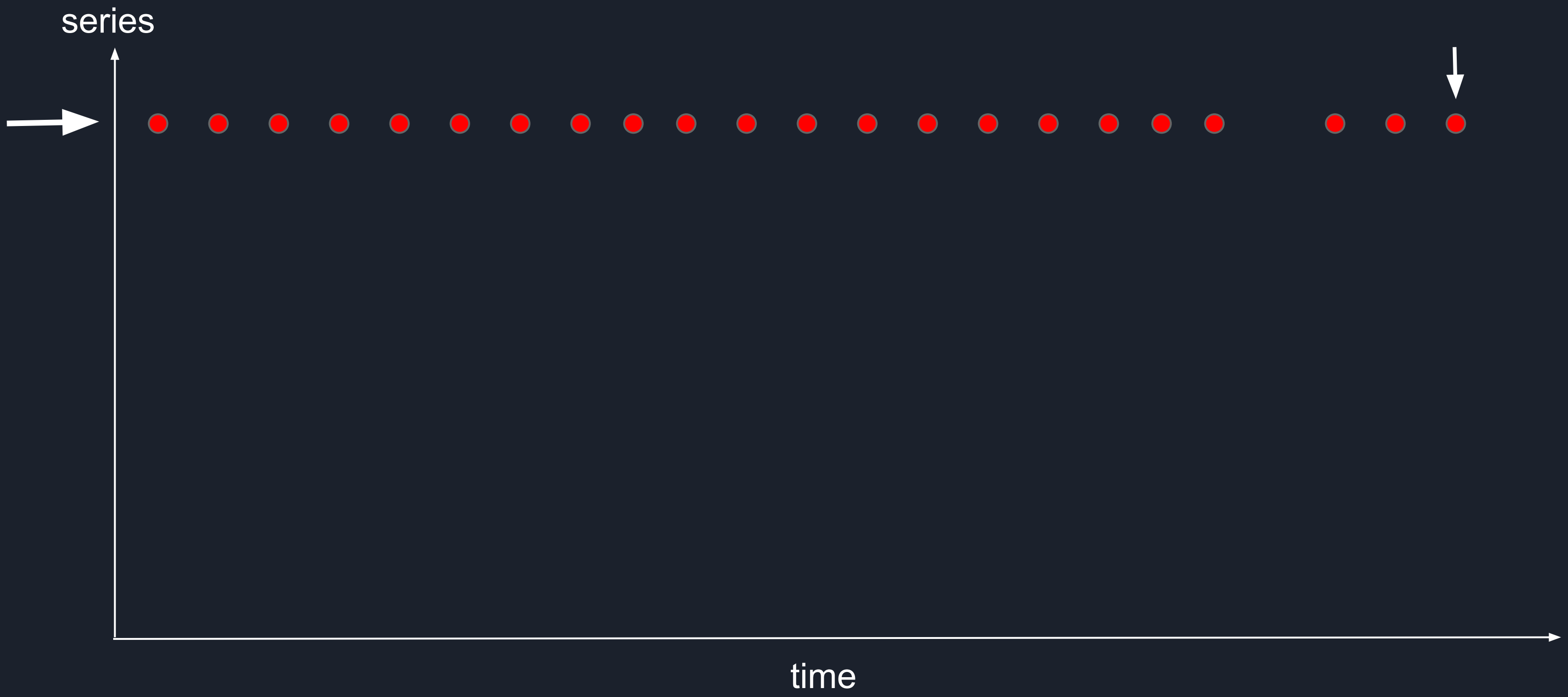
Querying



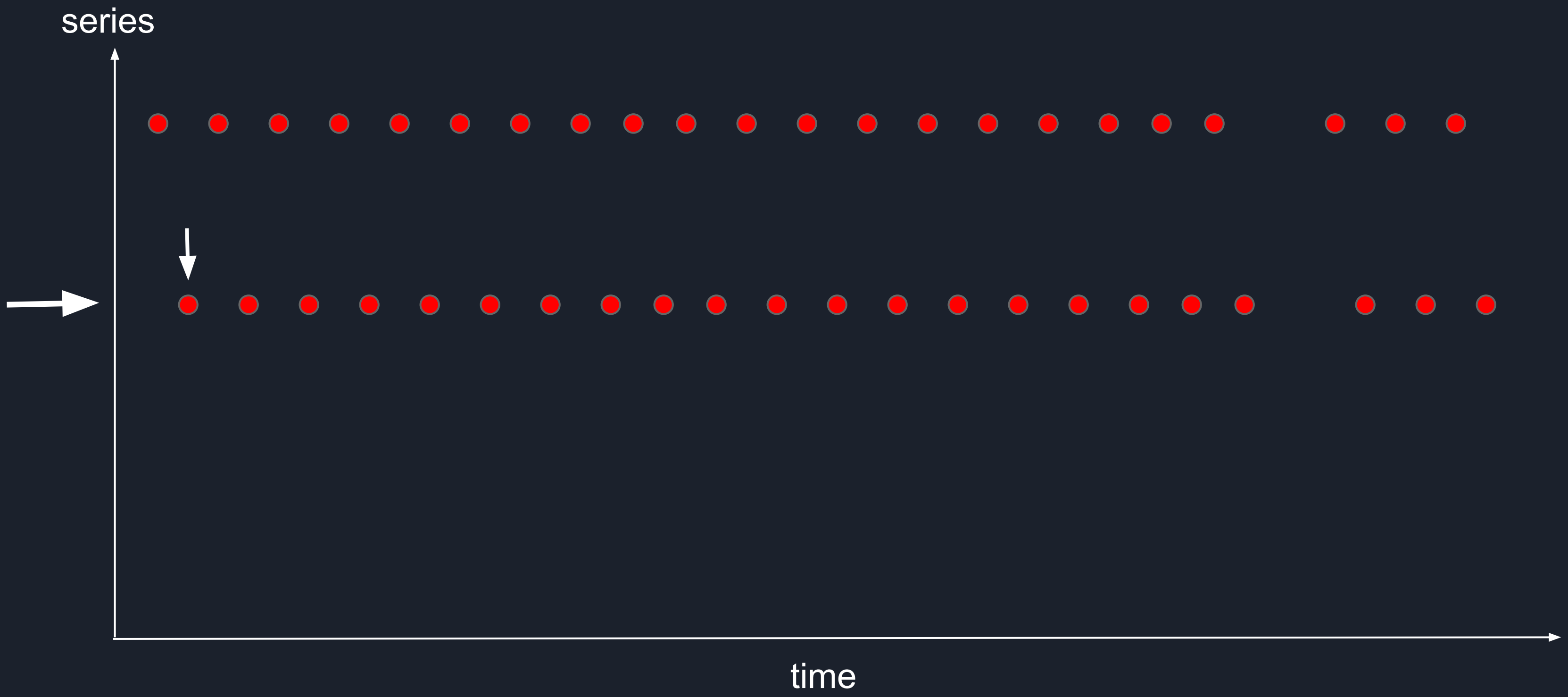
Querying



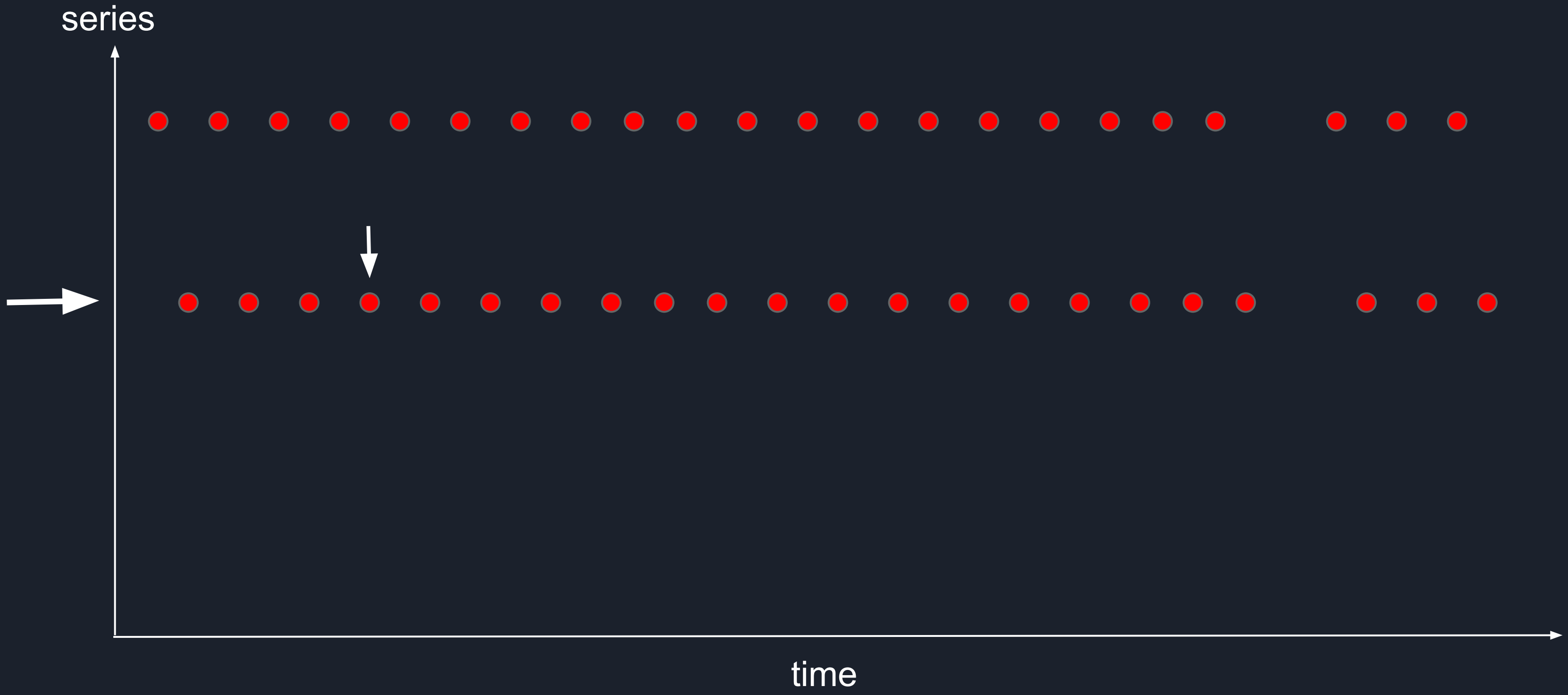
Querying



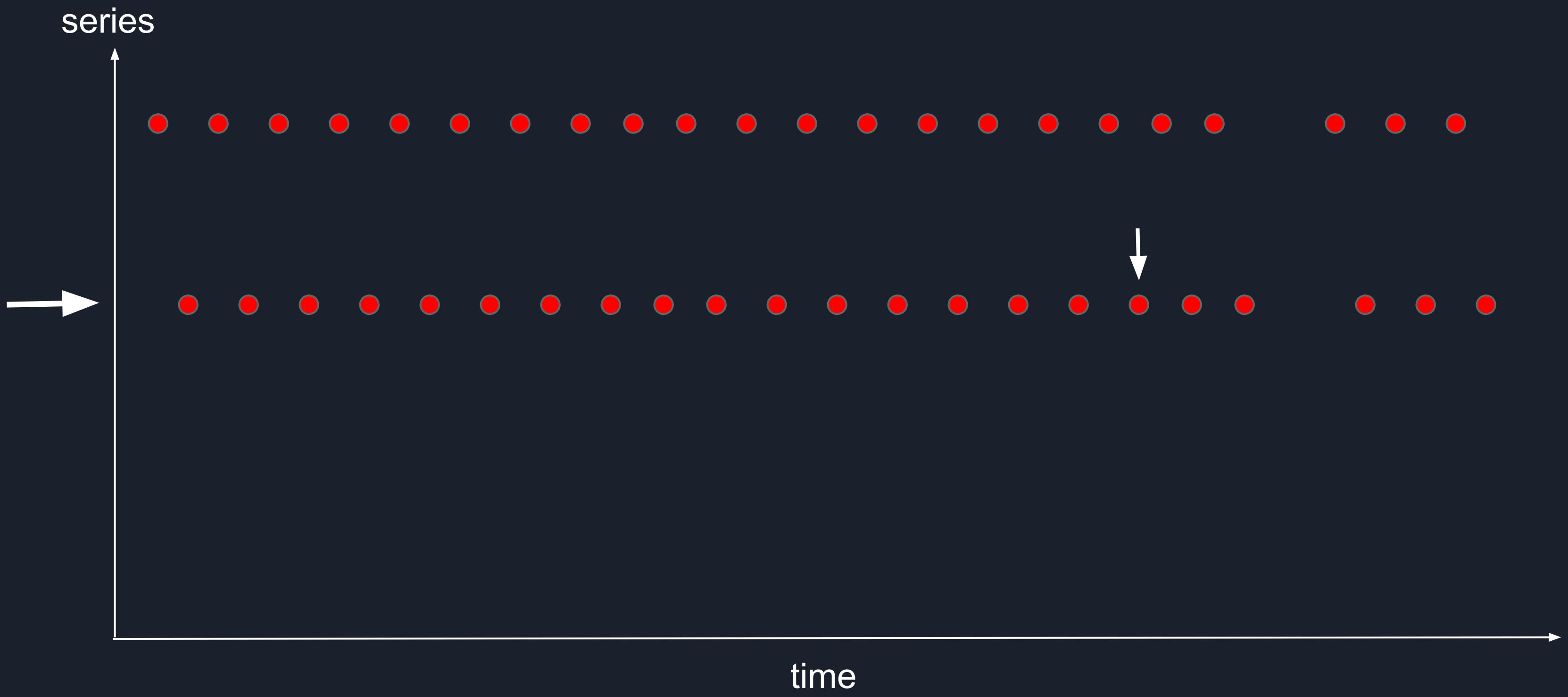
Querying



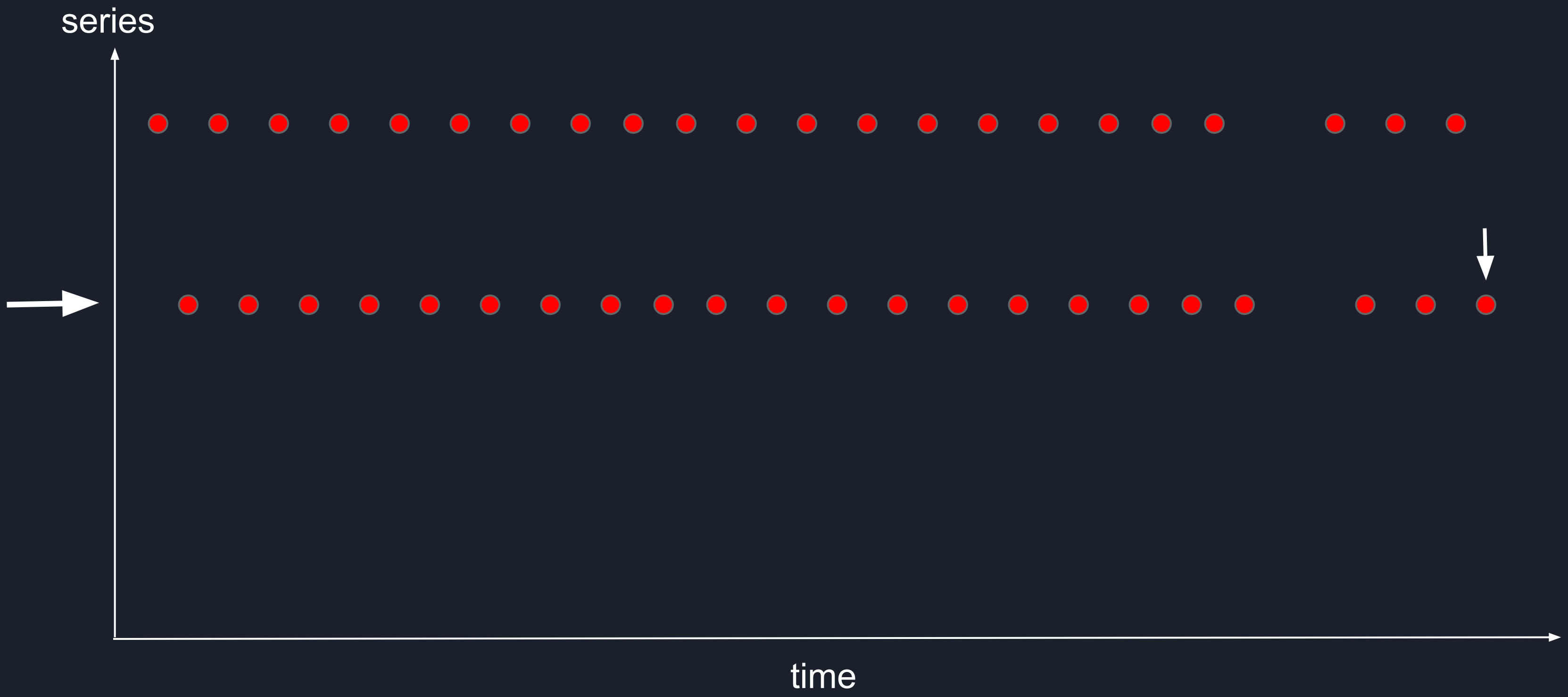
Querying



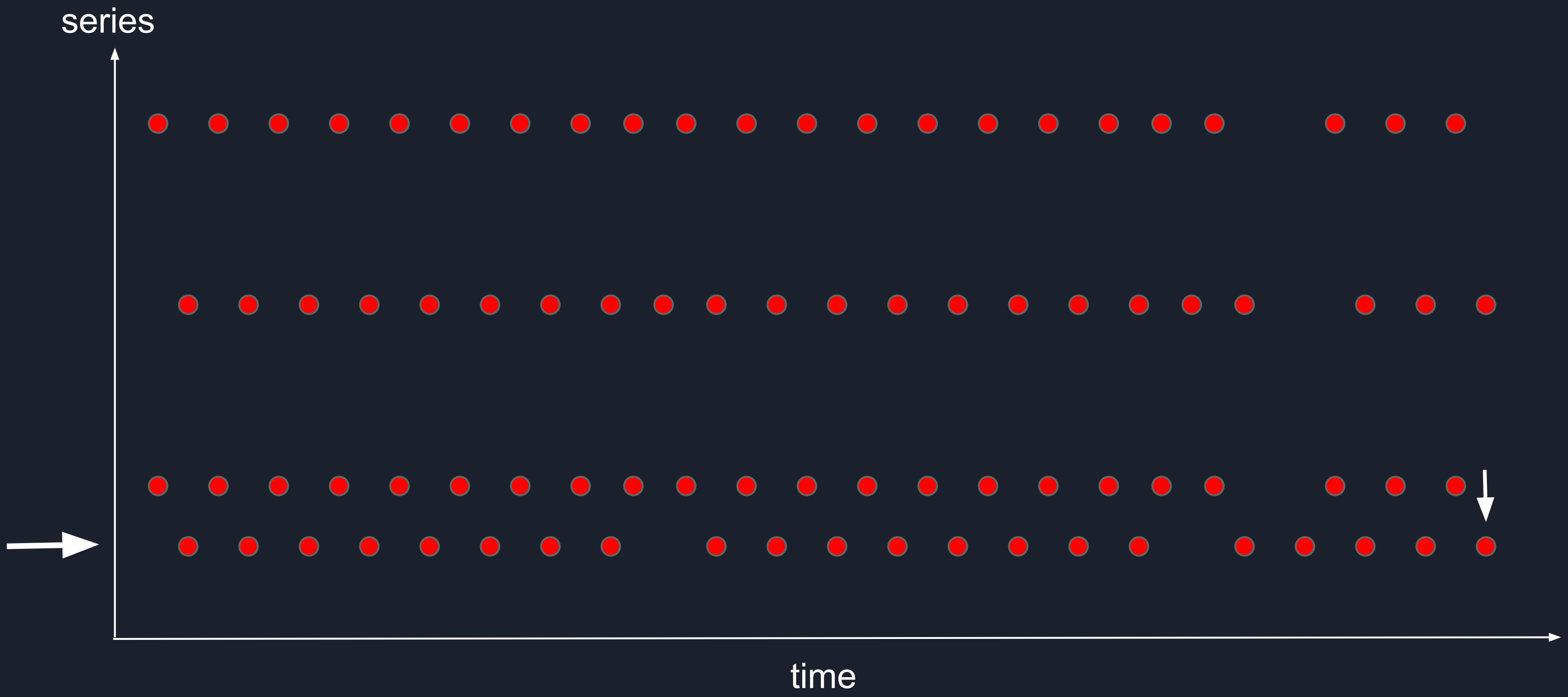
Querying



Querying



Querying



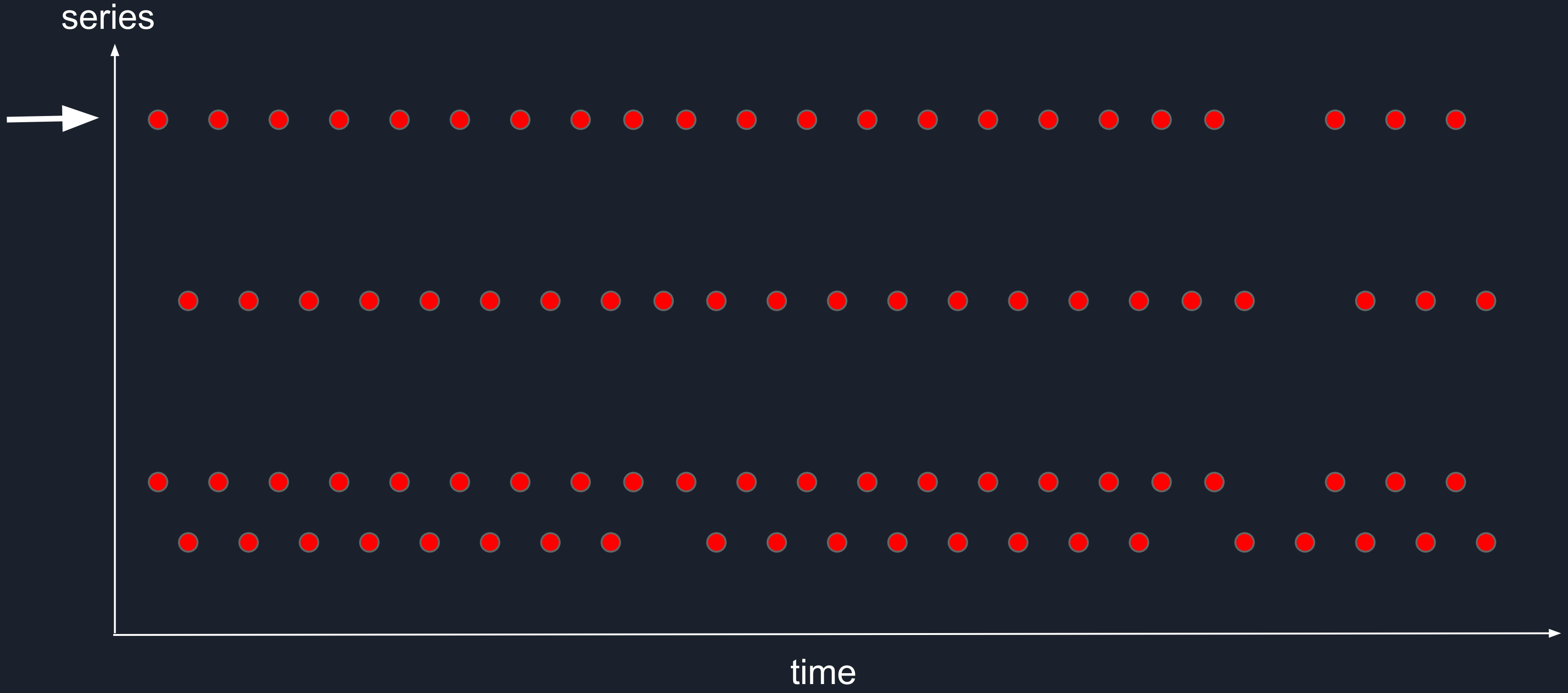
Querying

```
func (s *DB) Querier(mint, maxt int64) Querier

type Querier interface {
    // Select returns a set of series that matches the given label matchers.
    Select(...labels.Matcher) SeriesSet
    // LabelValues returns all potential values for a label name.
    LabelValues(string) ([]string, error)
    // Close releases the resources of the Querier.
    Close() error
}
```

Querying

```
Select(...labels.Matcher) SeriesSet

type SeriesSet interface {
    Next() bool
    At() Series
    Err() error
}
```

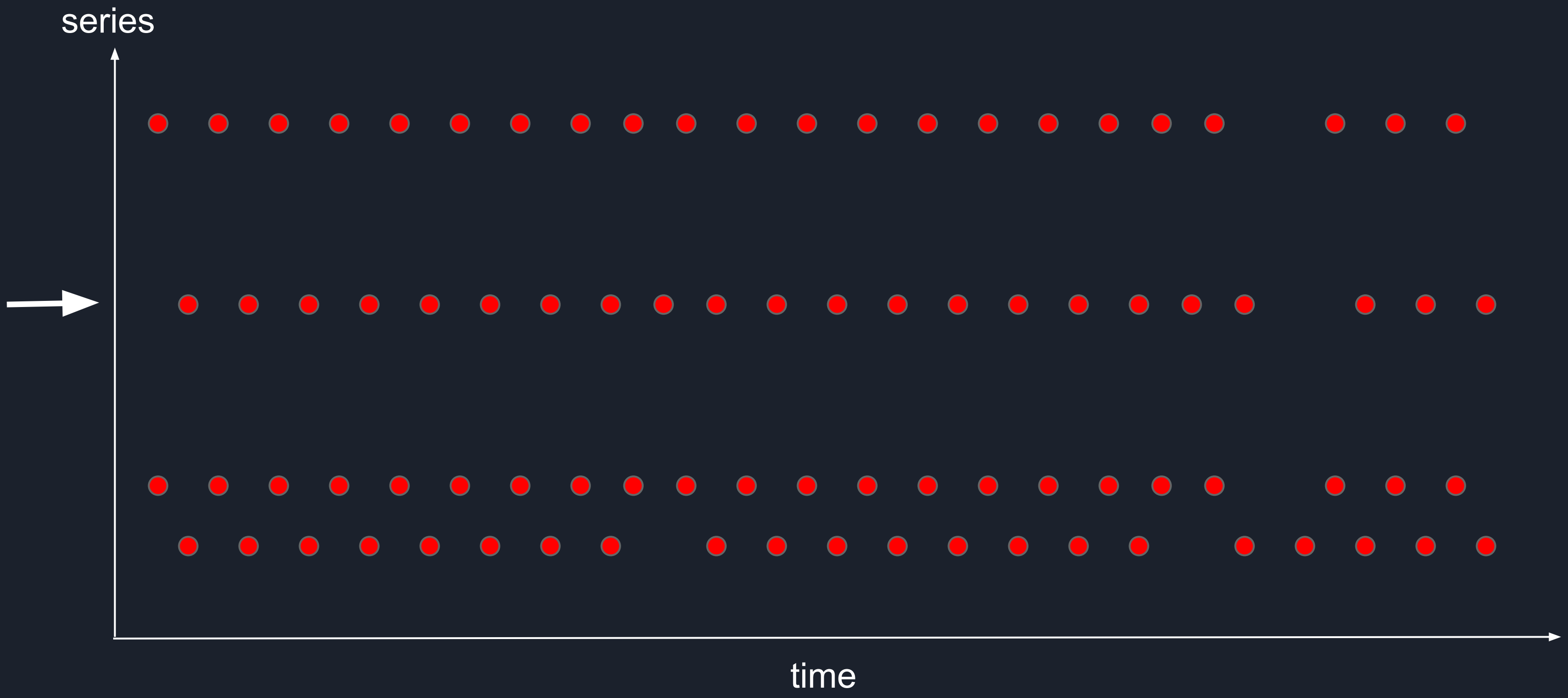
Querying



Querying

```
Select(...labels.Matcher) SeriesSet

type SeriesSet interface {
    Next() bool
    At() Series
    Err() error
}
```

Querying



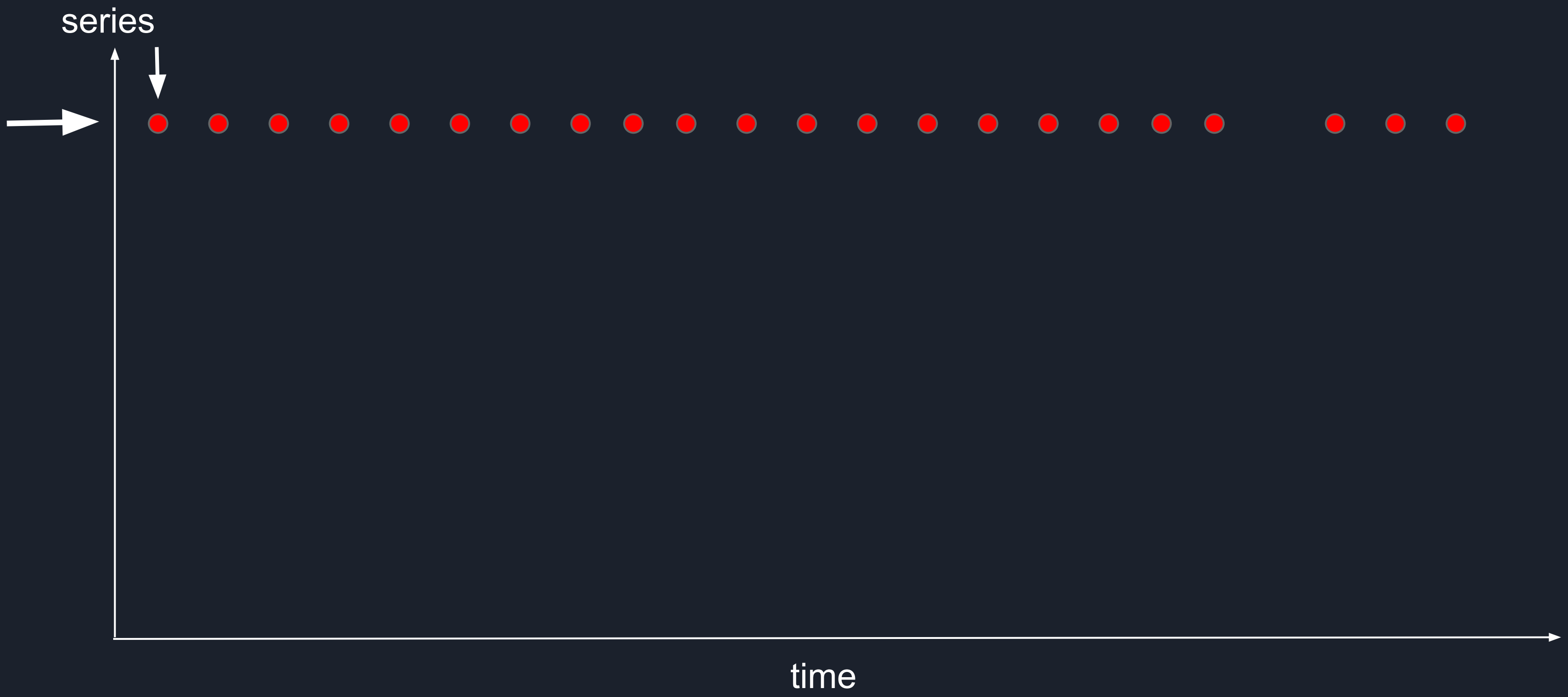
Querying

```
At() Series

type Series interface {
    // Labels returns the complete set of labels identifying the series.
    Labels() labels.Labels

    // Iterator returns a new iterator of the data of the series.
    Iterator() SeriesIterator
}
```

Querying

```
type SeriesIterator interface {
    // Seek advances the iterator forward to the given timestamp.
    // If there's no value exactly at t, it advances to the first value
    // after t.
    Seek(t int64) bool
    // At returns the current timestamp/value pair.
    At() (t int64, v float64)
    // Next advances the iterator by one.
    Next() bool
    // Err returns the current error.
    Err() error
}
```

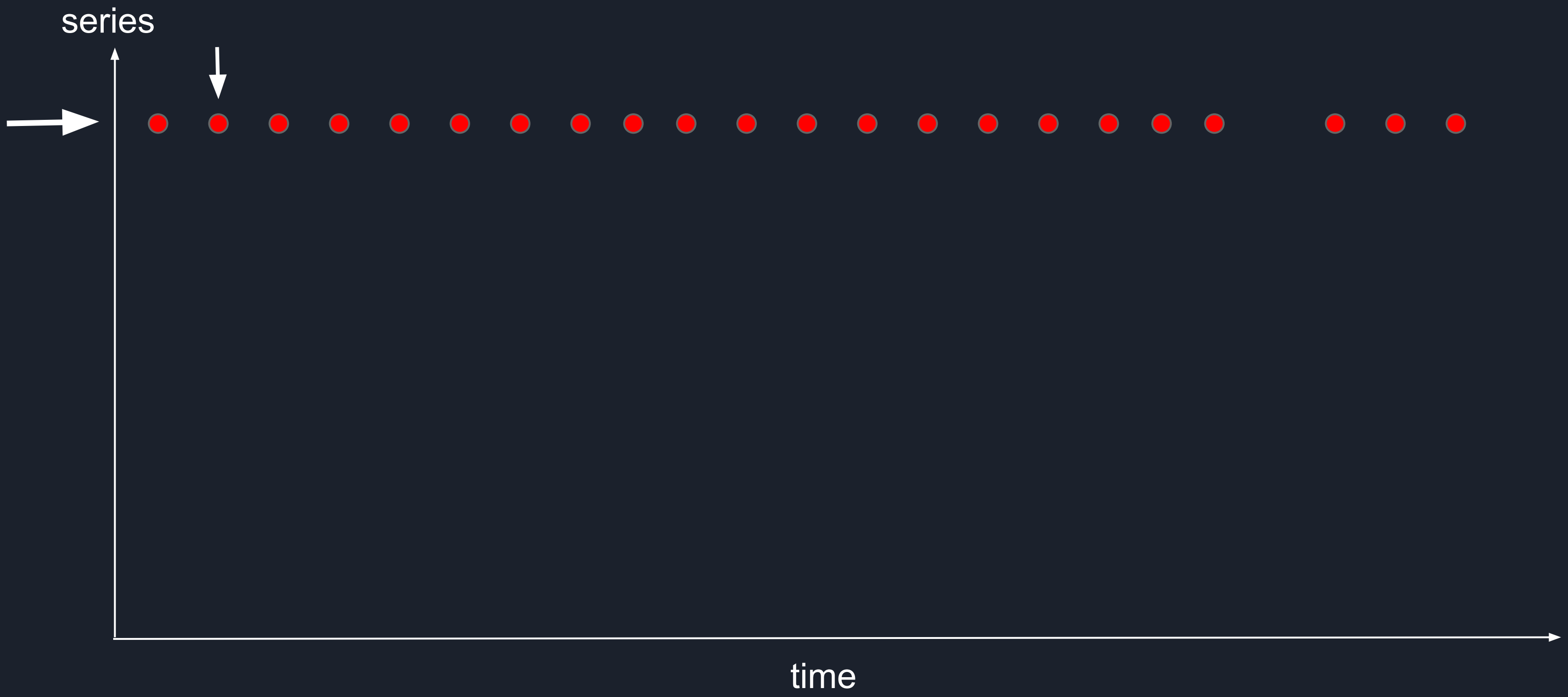
Querying



Querying

```
type SeriesIterator interface {
    // Seek advances the iterator forward to the given timestamp.
    // If there's no value exactly at t, it advances to the first value
    // after t.
    Seek(t int64) bool
    // At returns the current timestamp/value pair.
    At() (t int64, v float64)
    // Next advances the iterator by one.
    Next() bool
    // Err returns the current error.
    Err() error
}
```

Querying



Util

```
func PromQLToMatchers(buf []byte) ([]labels.Matcher, error)

PromQLToMatchers({name=~”prom.*”, host=”123”}) // → []Matcher

type response struct {
    Series []series
}

type series struct {
    Labels labels.Labels
    Points []point                // point → struct{ t v }
}
```

Code

https://github.com/gouthamve/promflux

Demo

Questions?

Goutham Veeramachaneni
Student @ IIT Hyderabad, India
ex-intern @ CoreOS

putadent

gouthamve