IBM Research

ConfAdvisor: A Performance-centric Configuration Tuning Framework for Containers on Kubernetes

Tatsuhiro Chiba, Rina Nakazawa, Hiroshi Horii
Sahil Suneja and Seethalami Seelam

IBM Research
IBM T.J. Watson Research Center

Agenda

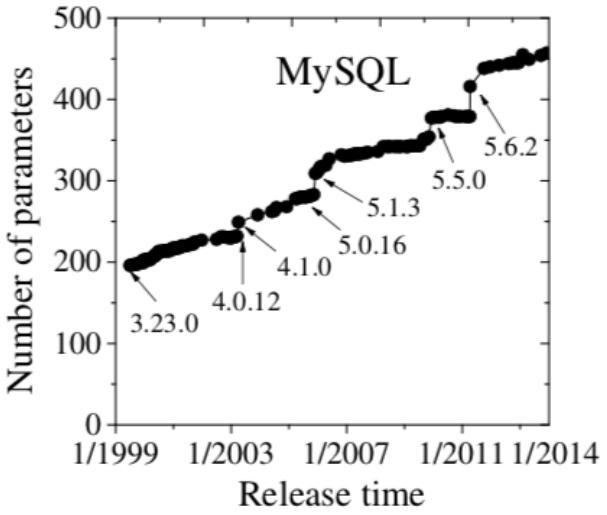
- Motivation, Problems and Challenges
- Design & Implementation
- Evaluation
- Summary

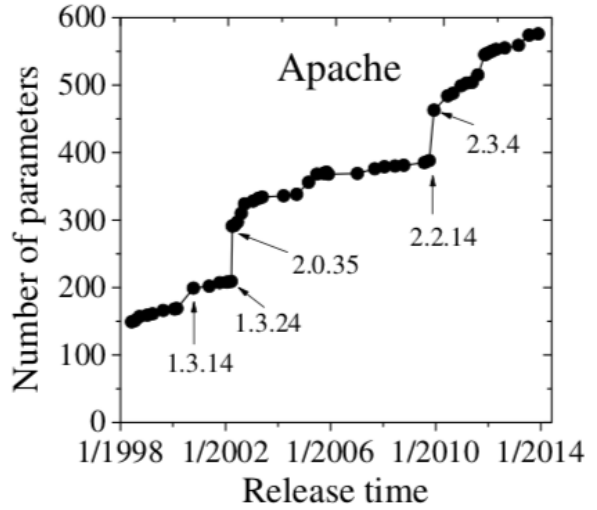
Performance Related Configs for Software

- **What kind of configs?**
  - the configs which can change software behavior
    - heap size, algorithm, # threads, enabling/disabling futures...
- **How many configs can we tune?**
  - Increasing tunable configs year by year [Xu+, FSE '15]
  - Apache HTTP Server: 550+
  - MySQL: 460+
  - OpenJDK (JVM): 700+

```
$> java -XX:+PrintFlagsFinal
[Global flags]
    uintx AdaptiveSizeDecrementScaleFactor          = 4    {product}
    uintx AdaptiveSizeMajorGCDecayTimeScale         = 10   {product}
    uintx AdaptiveSizePausePolicy                   = 0    {product}
    uintx AdaptiveSizePolicyCollectionCostMargin    = 50   {product}
    uintx AdaptiveSizePolicyInitializingSteps       = 20   {product}
...

$> java -XX:+rPintFlagsFinal | wc
723     3758   72844
```

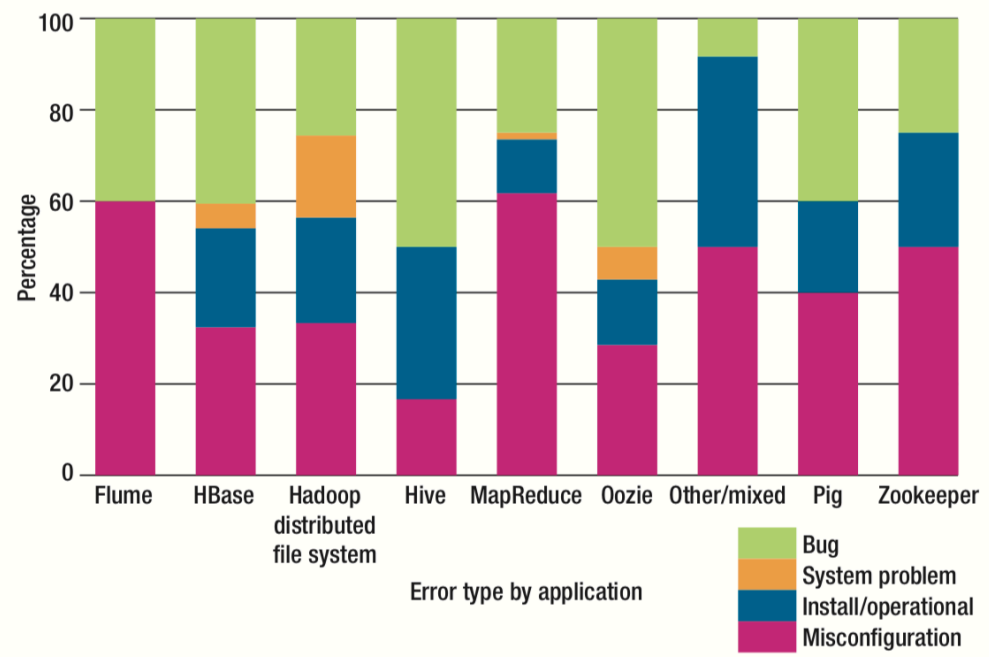




(ref) Hey, You Have Given Me Too Many Knobs!, Xu et al, ESEC/FSE 2015

Misconfig makes big performance impact

- **What is the result after config tuning?**
  - success: performance improvement
  - failure: performance drawback
  - config influences each other in complex manner
- **system anomalies due to misconfig**
  - up to 31% of cloud outage are caused by misconfiguration [Yin+, SOSP'11]
  - misconfig is a top reason of hadoop system outage [Ariel+, IEEE Software '13]



(Ref.) How Hadoop Clusters Break, IEEE Software, Vol.30, No.4

Example of Configs – Liberty and Nginx

Liberty (server.xml)

```
<server description="new server">
  <featureManager>
    <feature>microProfile-2.0</feature>
  </featureManager>
  <!-- optimal thread size depends on workload -->
  <executor coreThreads="1" maxThreads="10"/>
</server>
```

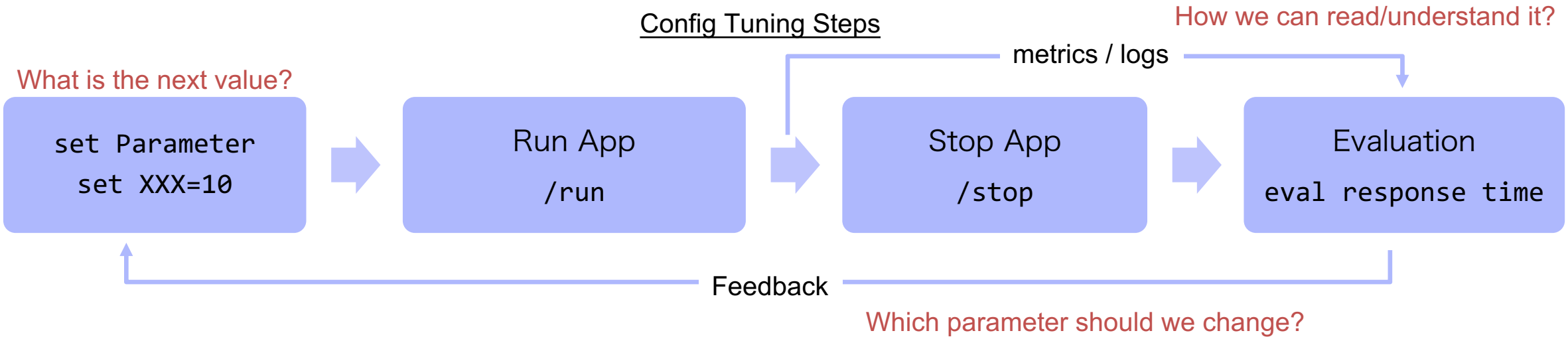
```
worker_processes  100;              # depends on env
events {
  worker_connections  1024;         # depends on env
}
http {
  sendfile      off;                # better to set on
  tcp_nopush    off;                # better to set on
  keepalive_timeout  60;            # depends on role
}
```

Nginx (nginx.conf)

- **Can change those configs, but …**
  - Not easy task to set proper config manually
  - Need to know app role detail
- **What about automation?**
  - Ansible or Kubernetes Operator can fix wrong operational processes automatically
  - PerfOps cycle still remains out of the automation loop..

Why config tuning is difficult?

- **Everyone cannot always make best config tuning, because**
  - exists too many knobs
  - require deep knowledge about the system and apps
  - application programmer != config tuning specialist
- **Need to know what happens in the apps from various logs/metrics**
  - Java Apps → GC, JIT, Method profiling, lock contention..
  - MongoDB → query stats, perf counter, i/o monitoring, index...



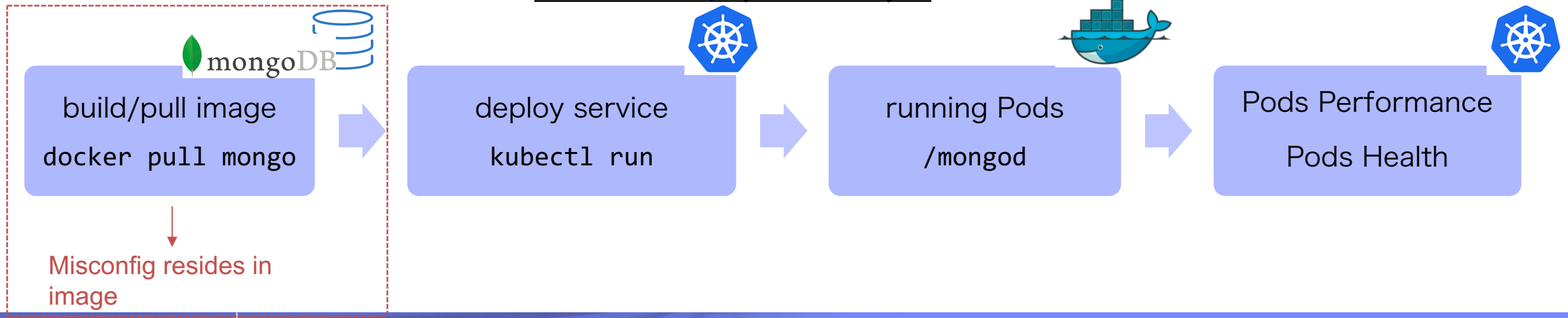
Config Tuning Difficulties in Container/Kubernetes (1/3)

- **Anyone can publish container images**
  - pre-tuned config image
  - default config image
  - misconfig image
- **Official images have few preset cfg**
  - still remains enough tuning space

TABLE I
THE NUMBER OF CONFIGS IN OFFICIAL DOCKER IMAGES

Software & Config	No. of tuning knobs default	No. of tuning knobs available	image name
Nginx (nginx.conf)	20	732	nginx:1.15
Apache2 (httpd.conf)	72	1011	httpd:2.4.37
Redis (redis.conf)	0	103	redis:5.0
MongoDB (mongod.conf)	8	127	mongo:4.1.4
Open Liberty (jvm.options)	0	146	open-liberty:javaee8
Cassandra (jvm.options)	32	763	cassandra:3.3.1

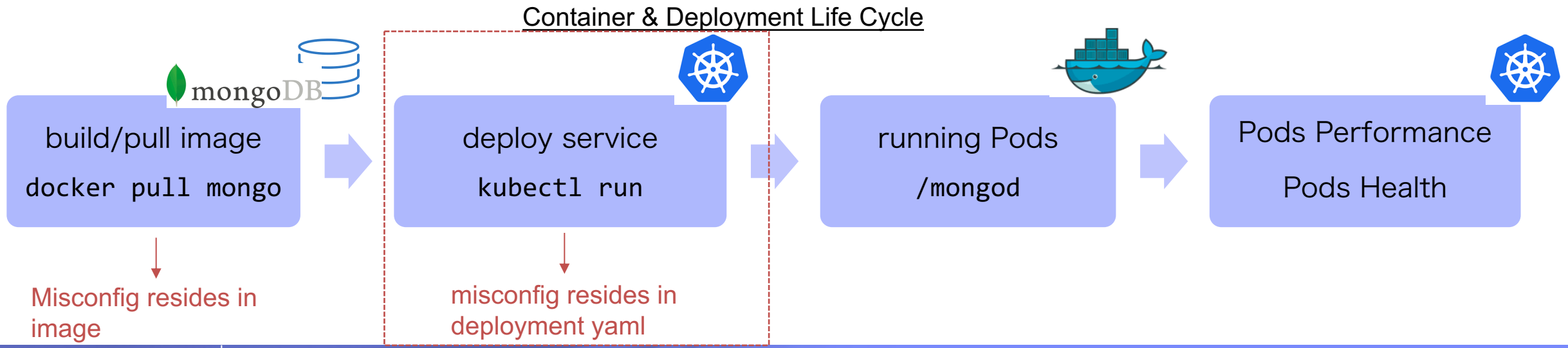
Container & Deployment Life Cycle



Config Tuning Difficulties in Container/Kubernetes (2/3)

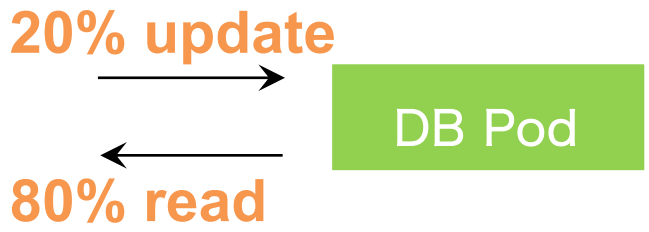
- **Config Violation at Deployment Timing**
  - Resource Quota (requests, limits)
  - Service mesh policies (load balancing, network capacity)
  - Storages (Persistent Volume or temporal space)

```
spec:
  containers:
  - image: cassandra:3.11.2
    resources:
      limits:
        memory: 1Gi
        cpu: 2
```

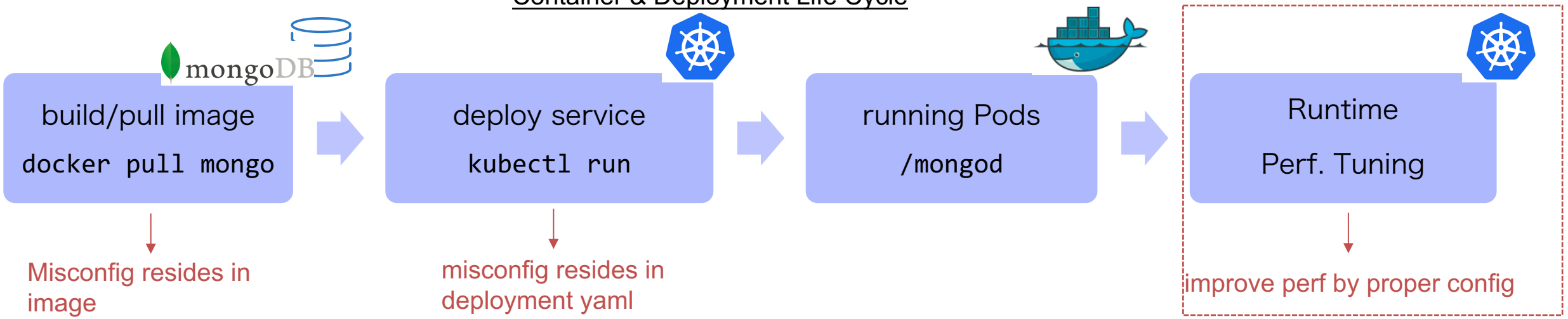


Config Tuning Difficulties in Container/Kubernetes (3/3)

- **Dynamic config tuning after running for a while**
  - Based on app usage characteristics or workloads
    - Read heavy vs. Write heavy
    - Response Time vs. Throughput
  - Detecting failed request, etc.
  - Horizontal pod scaling vs. vertical pod scaling



Container & Deployment Life Cycle



Related Works

- No generalized framework to make a config advice statically or dynamically
- No hybrid approach to use rule based and ML based tuning
- No continuous PerfOps config tuning system to support image, container, and Kubernetes

Name	Approach	Target System		Limitation	dedicated	k8s
Ansible	Rule base	OS, init setup	static	Need Python/SSH	-	✓
Dr.Erephant [1] Starfish [2]	Rule base	Hadoop Spark	static	Heuristics	✓	-
OtterTune [3]	Model base	MySQL PostgreSQL	static	model update	✓	-
CherryPick [4] BOAT [5]	Search base (Bayesian Opt.)	Hive, Spark, JVM VM Instance type	dynamic	Sampling convergence	✓	-
Our Framework	**Hybrid**	**Various Apps**	**static dynamic**		-	✓

[1] Dr. Erephant, Spark Summit '16

[2] Startfish, CIDR '11

[3] OtterTune, SIGMOD '17

[4] CherryPick, NSDI '17

[5] BOAT, WWW '17

Challenges and Contributions

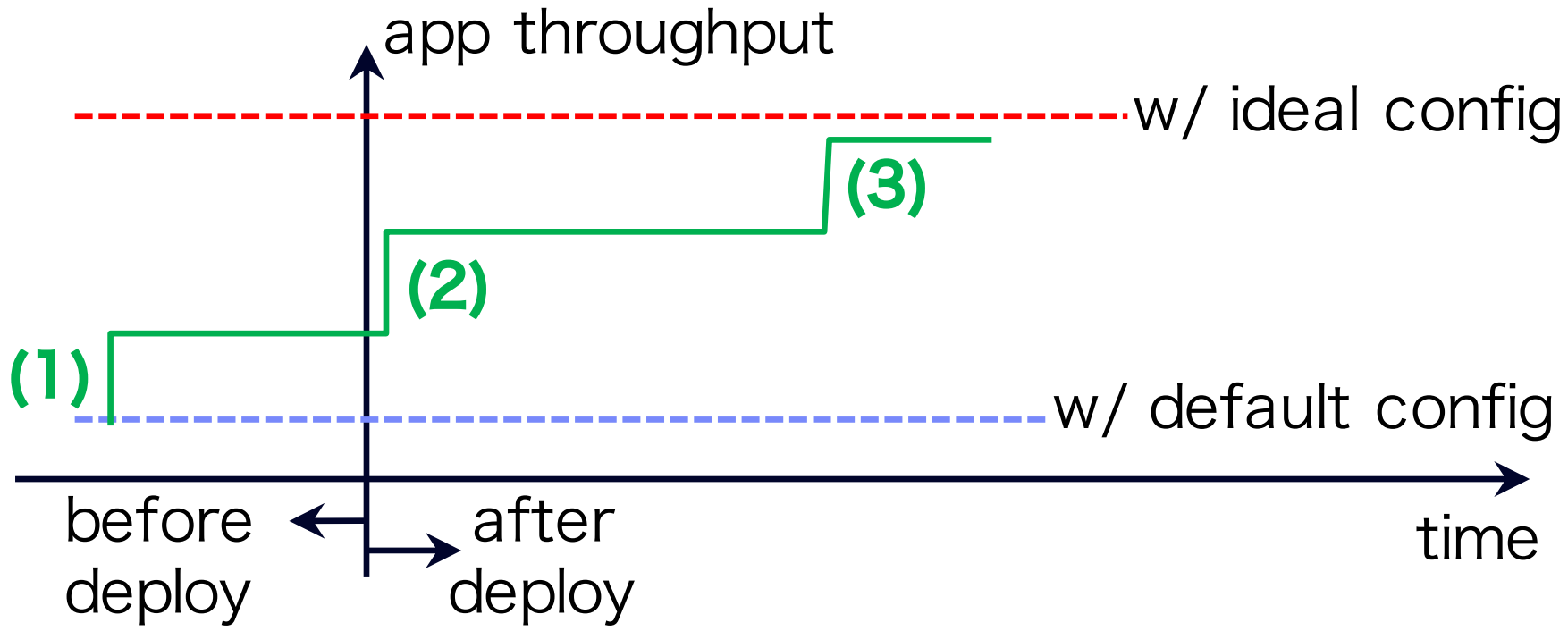
- **Challenges**
  - Achieve static/dynamic config tuning for various types of resources on k8s
  - Apply an optimized config continuously, aligned with cloud native app lifecycle
  - Support not only one specific app but many apps
  - Give a reasonable advice why my container is slow, relying on my container's config and metrics

- **Contributions**
  - Building a config tuning framework, ConfAdvisor, on Kubernetes
  - Extendable/customizable plugin system which can include tuning logic as a code
  - Case study：config tuning for Cassandra/ Liberty / MongoDB

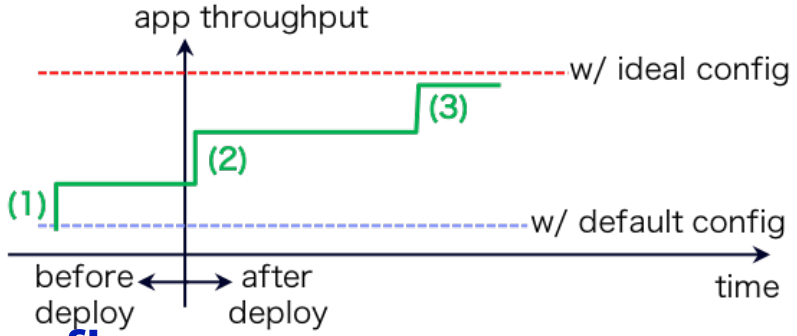
Agenda

- Motivation, Problems and Challenges
- **Design & Implementation**
- Evaluation
- Summary

Config tuning opportunities in cloud-native app lifecycle



ConfAdvisor – Definition of Advice Levels



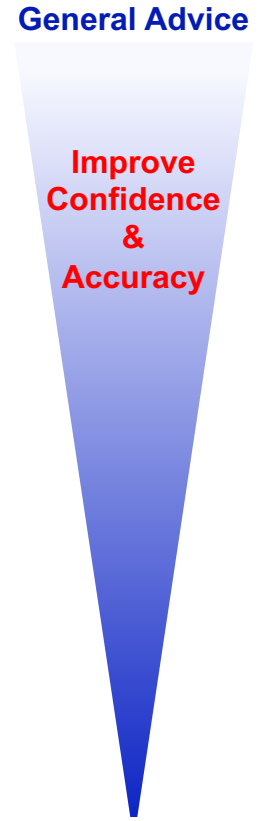
All advices are personalized result, which is based on its config or metrics that system can observe

Before Deployment (image)

After Deployment (Container)

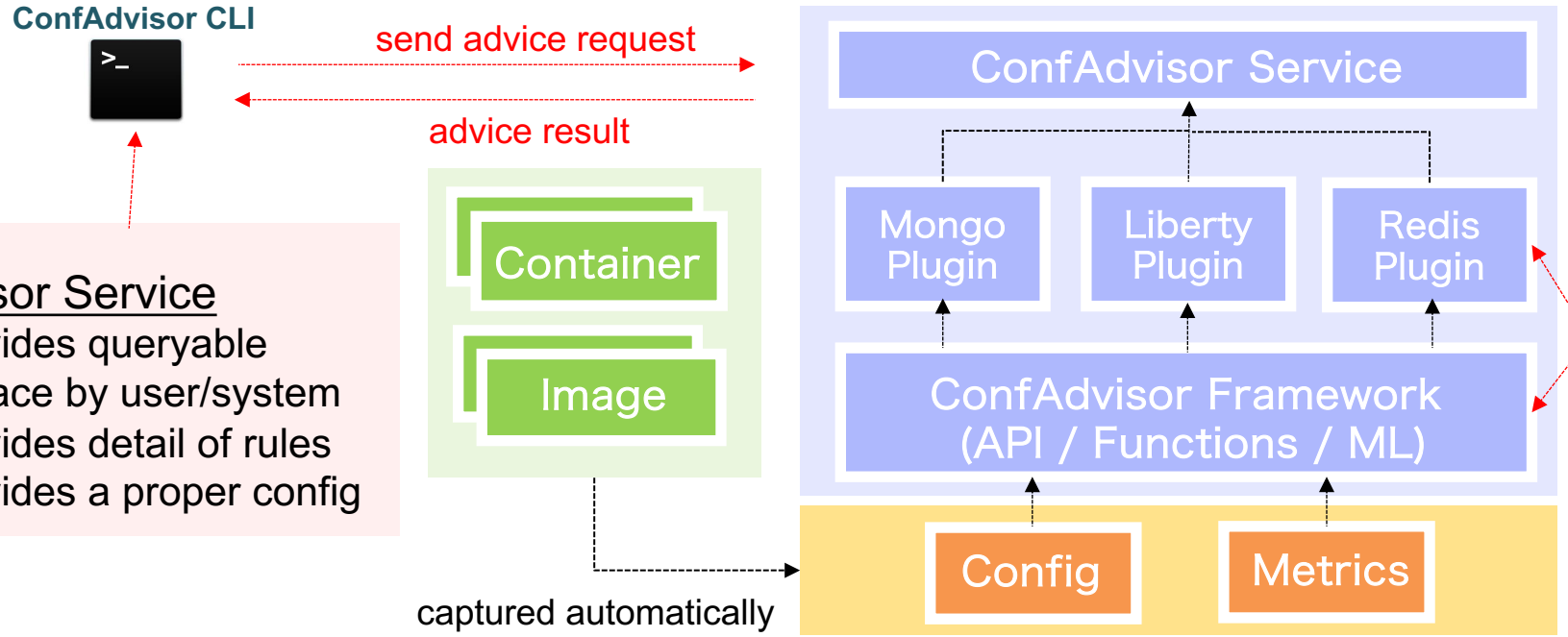
After Deployment for a while (Container)

- **Level 1. Advice based on app internal default config**
  - Liberty: jvm.options, server.xml, persistent.xml
  - mongodb: mongod.conf
- **Level 2. Advice based on app env config such as k8s quota or architecture**
  - Liberty: heap size, app threads size
  - Spark: executor heap size, executor threads
- **Level 3. Advice based on observed app runtime metrics**
  - Liberty: lock contention, optimal heap size, optimal GC algorithm
  - mongodb: throughput aware settings / latency aware settings



ConfAdvisor – System Overview

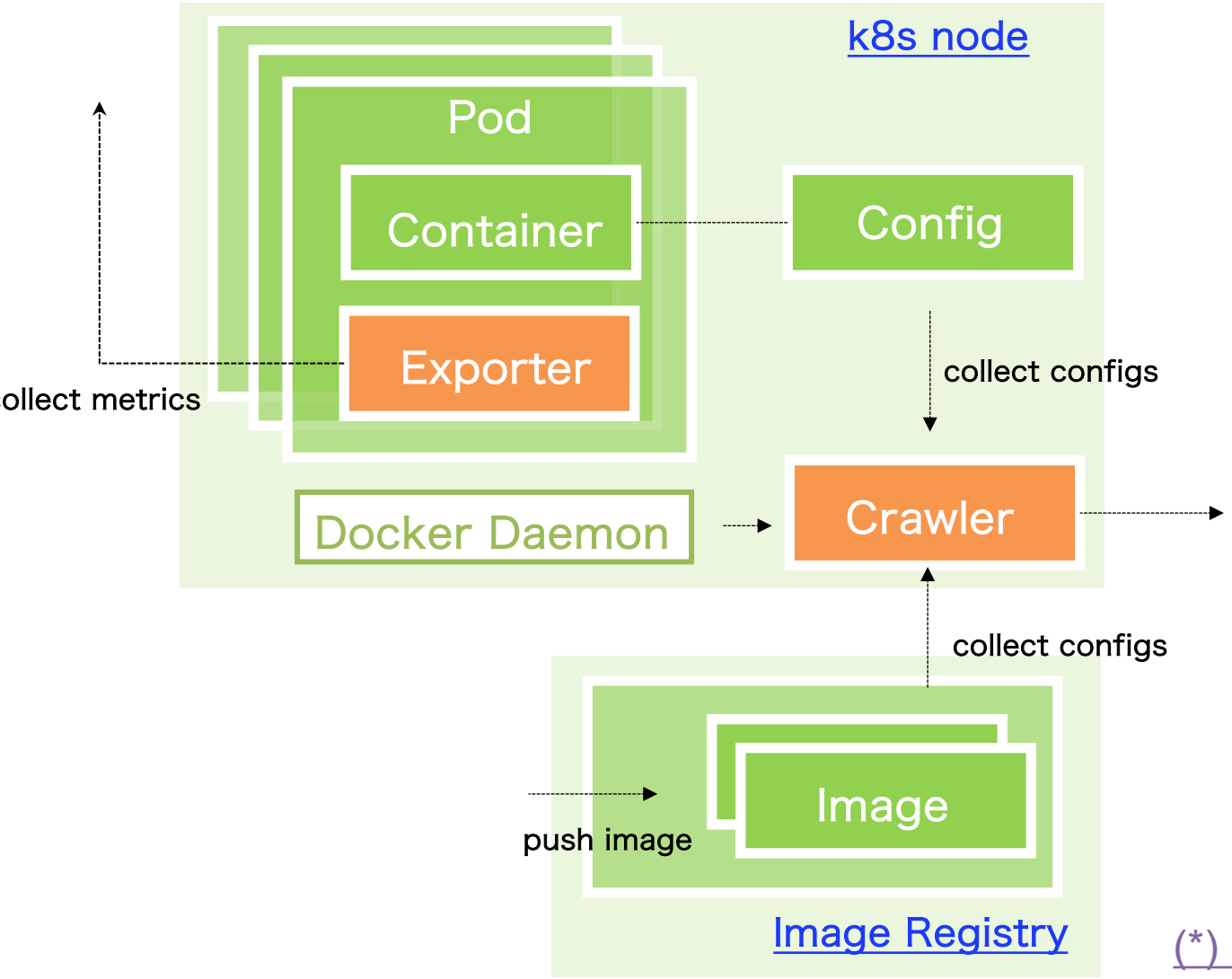
- **provides a service which gives perf-sensitive conf advice**
  - continuous automated PerfOps pipeline by user or system
- **pluggable/programable/declarative rule framework**
  - customize/extend advice rule by perf engineers
  - install plugin if we need



Advisor Service
- provides queryable interface by user/system
- provides detail of rules
- provides a proper config

Add Plugins and Rules
- simple customizable what-if rule
- python and jinja2 template
- config/metrics are available

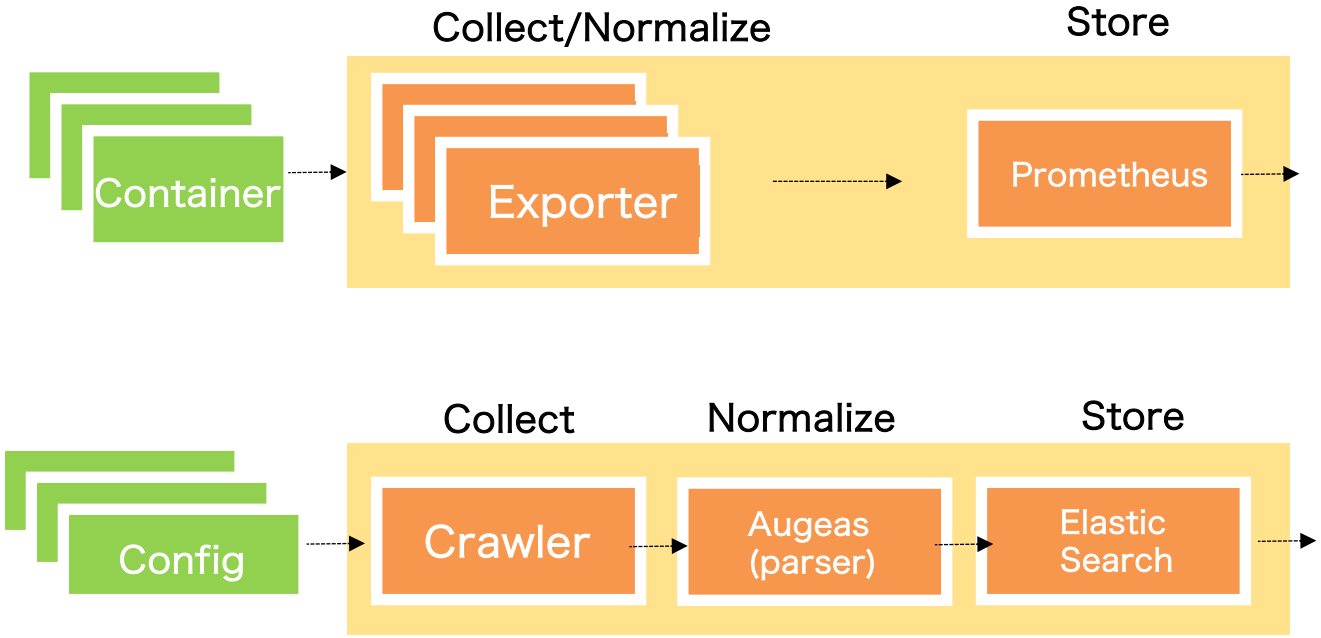
Implementation: Capturing config and metrics



- **Metrics collector**
  - deploy app exporter as a side car
  - periodically captured by prometheus
- **Container config collector**
  - periodically captured by crawler(*) daemonset
  - crawls config immediately by detecting container creation event
- **Image config collector**
  - crawls config inside of image immediately when finding a new image in registry

(*) https://github.com/cloudviz/agentless-system-crawler

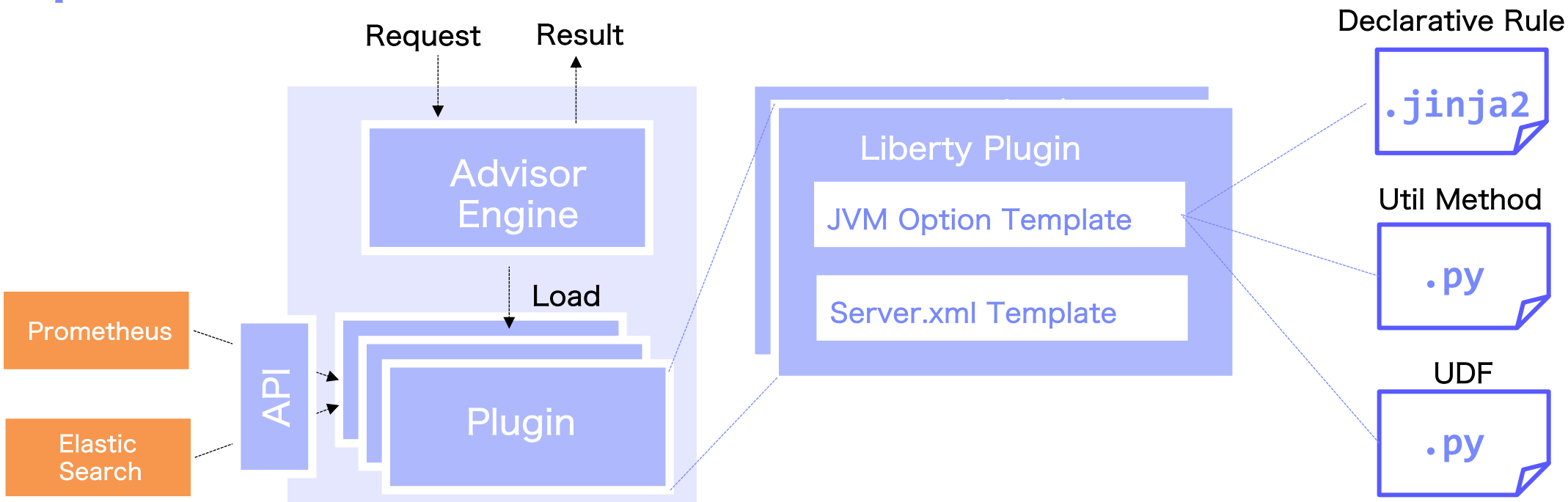
Implementation: Storing configs and metrics



- **Metrics storage**
  - stores all metrics into Prometheus
- **Config storage**
  - Data normalization by using Augeas (*)
  - Store them into ElasticSearch

(*2) http://augeas.net/

Implementation: Advisor Framework



- **Advisor Service & Framework**
  - handle advice request for an app
  - load a specified plugin
  - calc. optimized config for the app
  - give back the result

- **Plugin**
  - Written in Python
  - Implemented logic and rules per each config
    - Liberty, MongoDB, K8S, Cassandra, etc..
  - can use predefined functions by framework
  - can access various data from API

Implementation : Declarative Rules (1/2)

- **Declarative Config Template**
  - apply it when **what-if** rule is matched
  - **current** is a placeholder to access the current value
  - advice will be generated from a formula in **advice**
- **Customizable**
  - append any rules for a specified config if you want

```
"jvm.options": [
{
  "name": "mx",
  "what-if": "current.mx > current.limits_memory",
  "advice": "current.limits_memory * 0.75",
  "order": 0,
  "message": "should keep mx less than memory limits"
},
{
  "name": "ms",
  "what-if": "current.ms != advice.mx",
  "advice": "current.limits_memory * 0.75",
  "order": 1,
  "message": "should keep ms same as mx"
}
]
```

Implementation : Declarative Rules (2/2)

- **Extendable plugins**
  - can write any tuning logic as a code in a plugin
  - prepare it as an UDF function (e.g. `rate_cpu_usage`)
  - the function is automatically available inside of declarative rule
- **Framework API supports**
  - exposing time-series data as a pandas dataframe
  - utilizing prebuilt ML model

```
"server.xml": [
{
  "key": "maxThreads",
  "what-if": "current.maxThreads < (current.cpu * 5)",
  "advice" : "current.cpu * 5",
  "order": 0,
  "message": "should start with 5 * vcpu"
}],
"k8s": [
{
  "key": "scale_pod",
  "what-if": "rate_cpu_usage(vars.interval) > 0.8"
  "advice": "current.replica += 1",
  "order": 0,
  "message": "check average cpu usage in last interval
}]
```

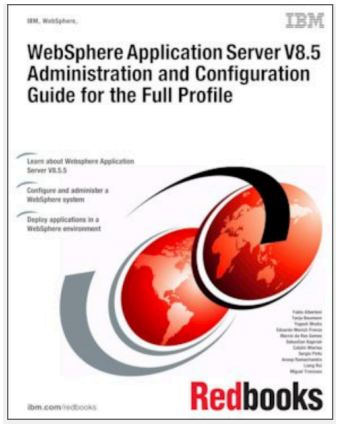
Agenda

- Motivation, Problems and Challenges
- Design & Implementation
- **Evaluation**
- **Summary**

Example of Implemented Declarative Rules

App	Config	what-if rule	Advice
Liberty	server.xml	maxPoolSize != coreThreads	maxPoolSize = coreThreads
Redis	redis.conf	maxmemory == None	maxmemory = mem_limit * 0.8
Node.js	k8s yaml spec	rate_cpu_usage(30m) > 0.9	replica++
MongoDB	mongod.conf	cacheSizeGB > mem_limit	cacheSizeGB = mem_limit / 2
Cassandra	jvm.options	-XX:MaxHeapSize > 16GB	-XX:+UseG1GC
Cassandra	jvm.options	jdk_version < 1.8.0_192-b01	-XX:+UseCGroupMemoryLimitForHeap

- **Implemented advisor plugin for apps**
  - Liberty, Httpd, Nginx, Node.js, Redis, MongoDB, Cassandra
- **Advisor source coming from,**
  - Official Tuning Guide (RedBook, official site)
  - JIRA, Github Issue, Quora, etc.
  - manually ported from those docs



Runtime Overhead – querying performance

- **Evaluated end-to-end ConfAdvisor performance**
  - L1/L2: simple rule and advice processing
  - L3: including data loading from Prometheus
  - Make it scale by putting ConfAdvisor replicas more

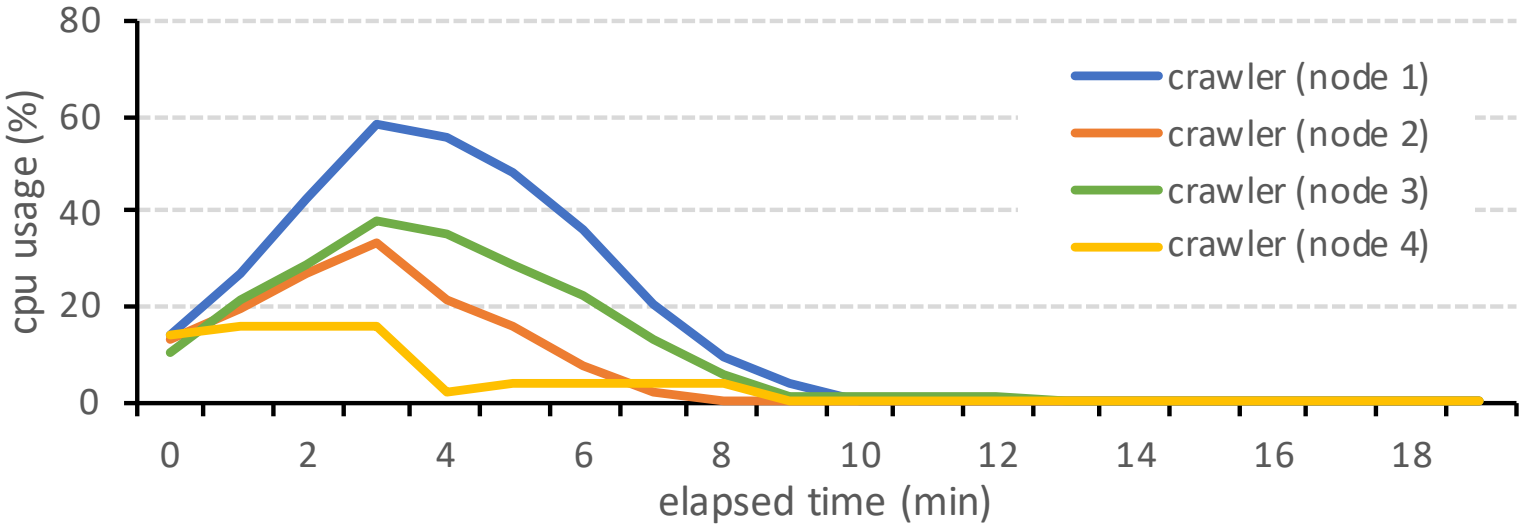
CONFADVISOR QUERY PROCESSING THROUGHPUT

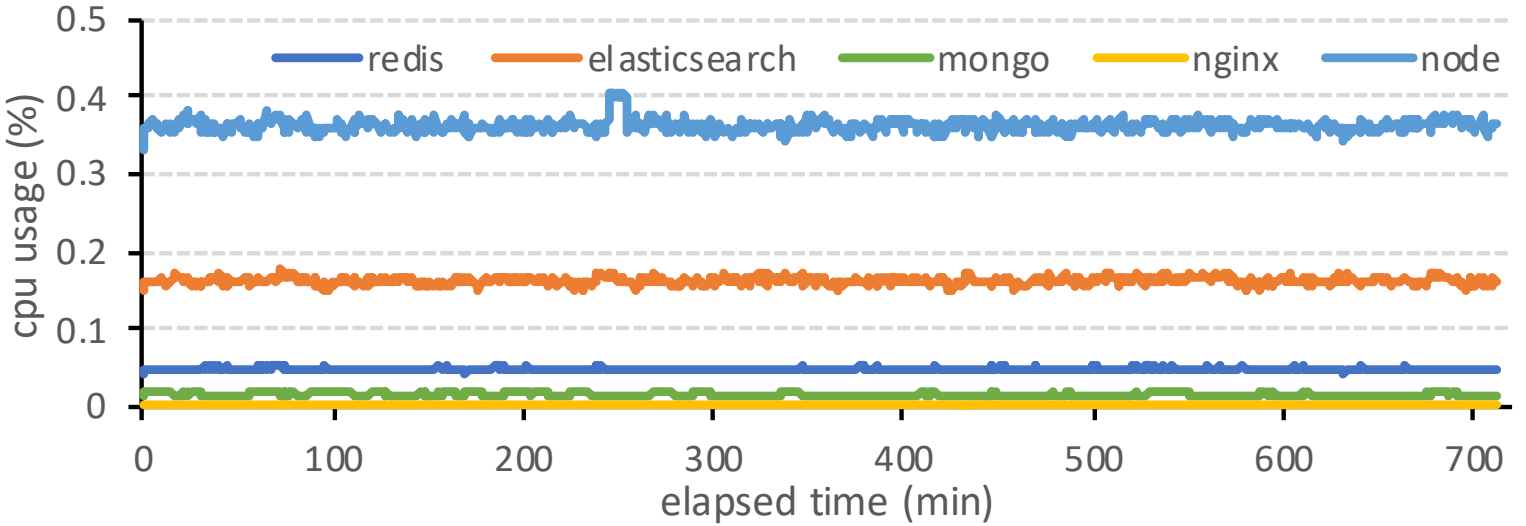
	Rule throughput (query/sec)				
category	**avg**	**stdev**	**min**	**max**	**median**
L1/L2	25.6	1.93	20.1	29.7	25.7
L3	3.32	1.92	0.56	5.55	4.39

Environment	
Machine	Xeon E5 2683 v3 (Xen), SMT2
VM	8 vCPU, 16GB RAM, Ubuntu 16.04
Instance	4
Kubernetes	1.11.1
Docker	18.06.1-ce
Prometheus	2.3.1

Runtime Overhead – crawling performance

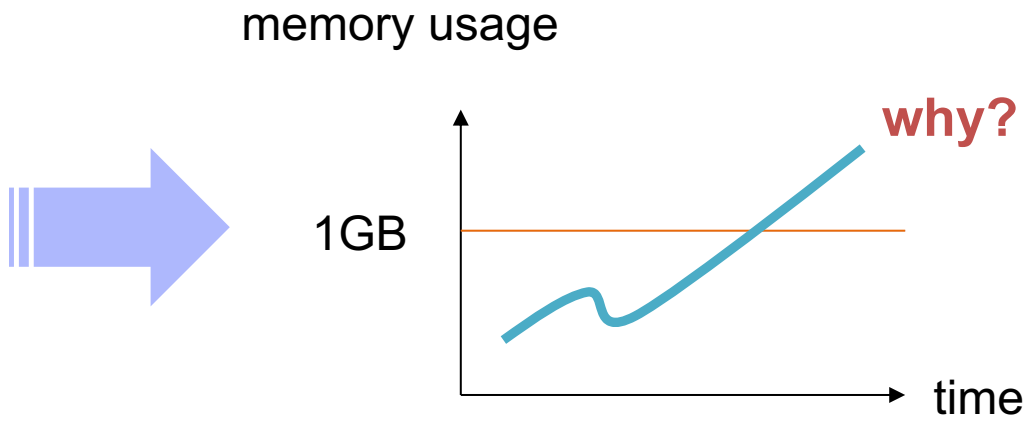
- **Evaluated config crawling overhead**
  - depends on number of containers on each node
  - not so big overhead
- **Evaluated metrics crawling overhead**
  - depends on the number of metrics and frequency
  - seems to be low





Case Study: Cassandra + YCSB benchmark (1/2)

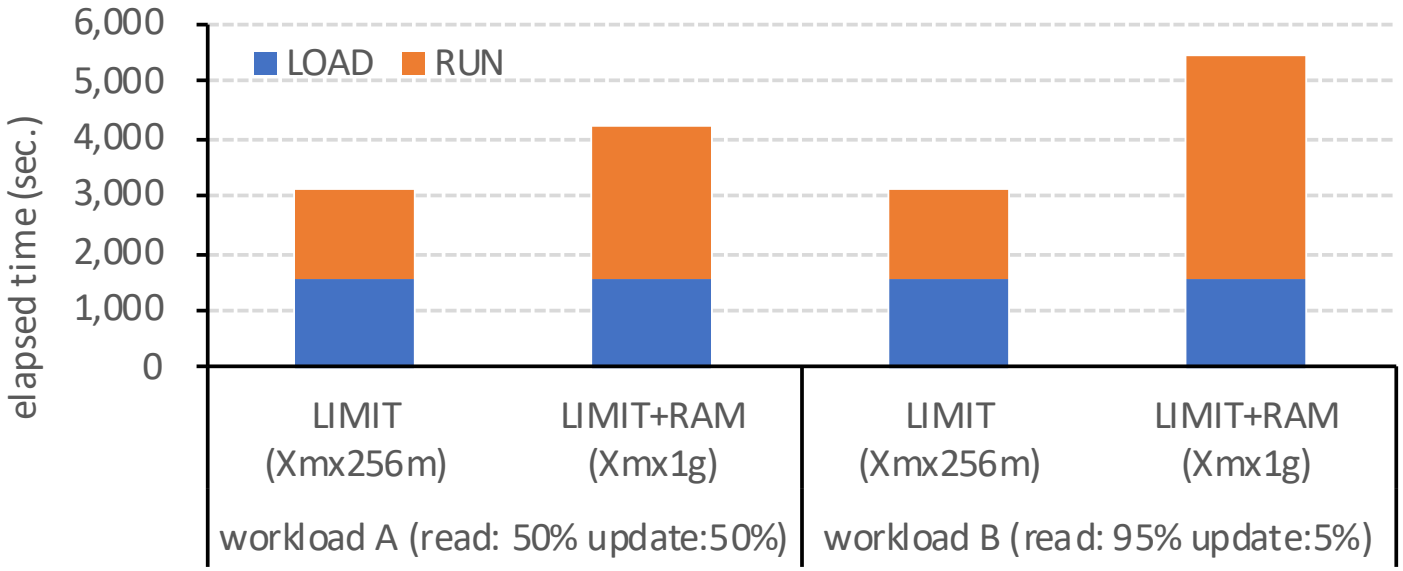
```
spec:
  containers:
  - image: cassandra:3.11.2
    resources:
      limits:
        memory: 1Gi
        cpu: 2
```



- **particular OpenJDK do not take cgroup limit into consideration**
  - `-XX:+UseCGroupMemoryLimitForHeap`
  - `what-if: version > 1.8.0_192-b01`: no need to set
  - `what-if: version < 1.8.0_192-b01`: must set the option explicitly
- **recommends to append the option**
  - w/o advice: can not finish workload (KILLED by the system)
  - w/ advice: keep smaller heap than 1GB limitation

Case Study: Cassandra + YCSB benchmark (2/2)

```
spec:
  containers:
  - image: cassandra:3.11.2
    resources:
      limits:
        memory: 1Gi
        cpu: 2
```



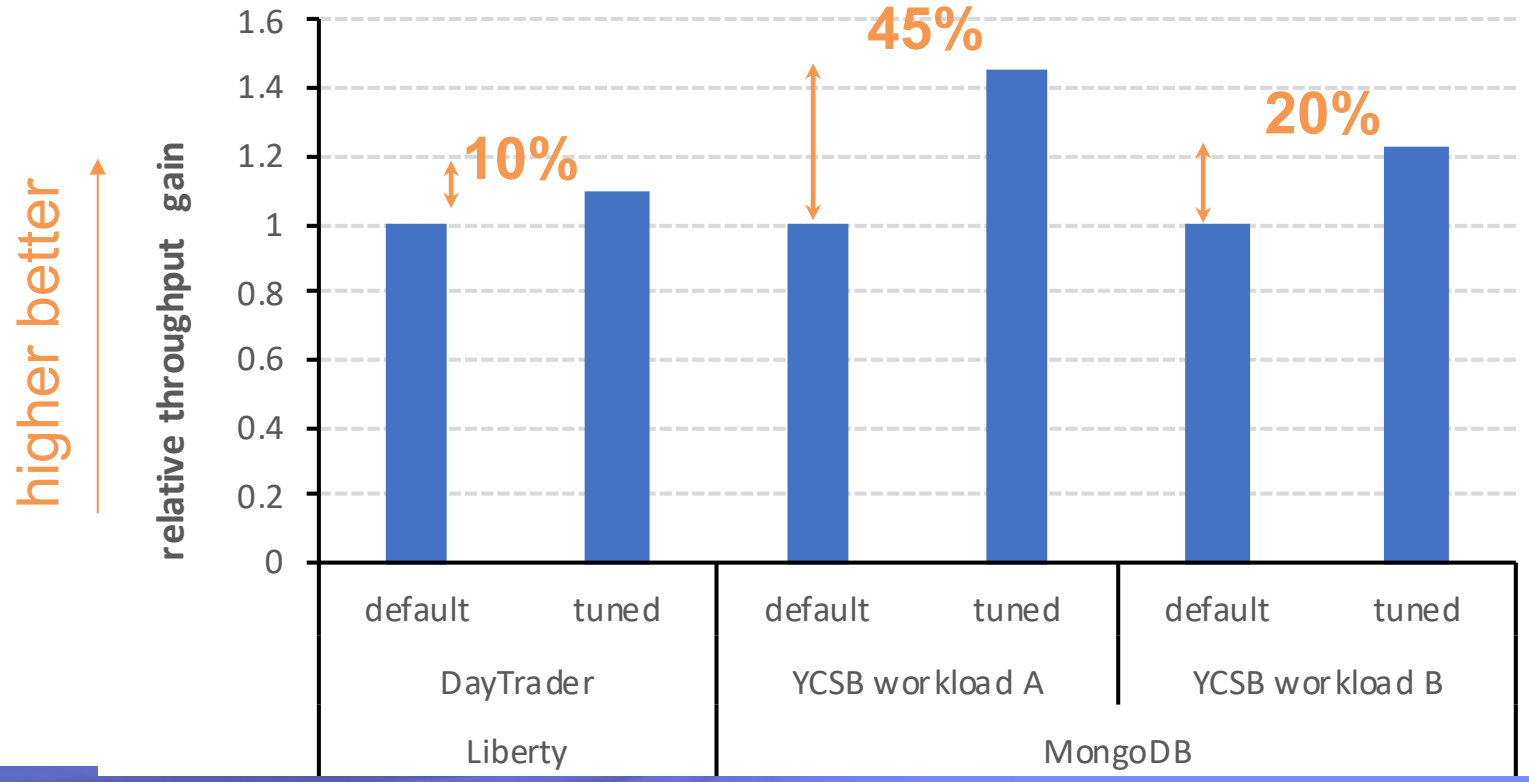
- **recommends to append one more JVM option**
  - LIMIT: `-XX:+UseCGroupMemoryLimitForHeap` (`-Xmx256m`)
  - LIMIT+RAM: `-XX:+UseCGroupMemoryLimitForHeap -XX:MaxRAMFraction=1` (`-Xmx1g`)
- **By fixing the JVM option misconfig**
  - successfully run workload
  - achieved 1.7x or 2.5x improvement

Case Study: memory sizing

- **Liberty + DayTrader**
  - Nursery heap/tenured heap size optimization
- **MongoDB + YCSB**
  - WiredTiger cache size optimization

DayTrader
- duration : 1800 sec
- ramp up : 300 sec
- # client thread : 60

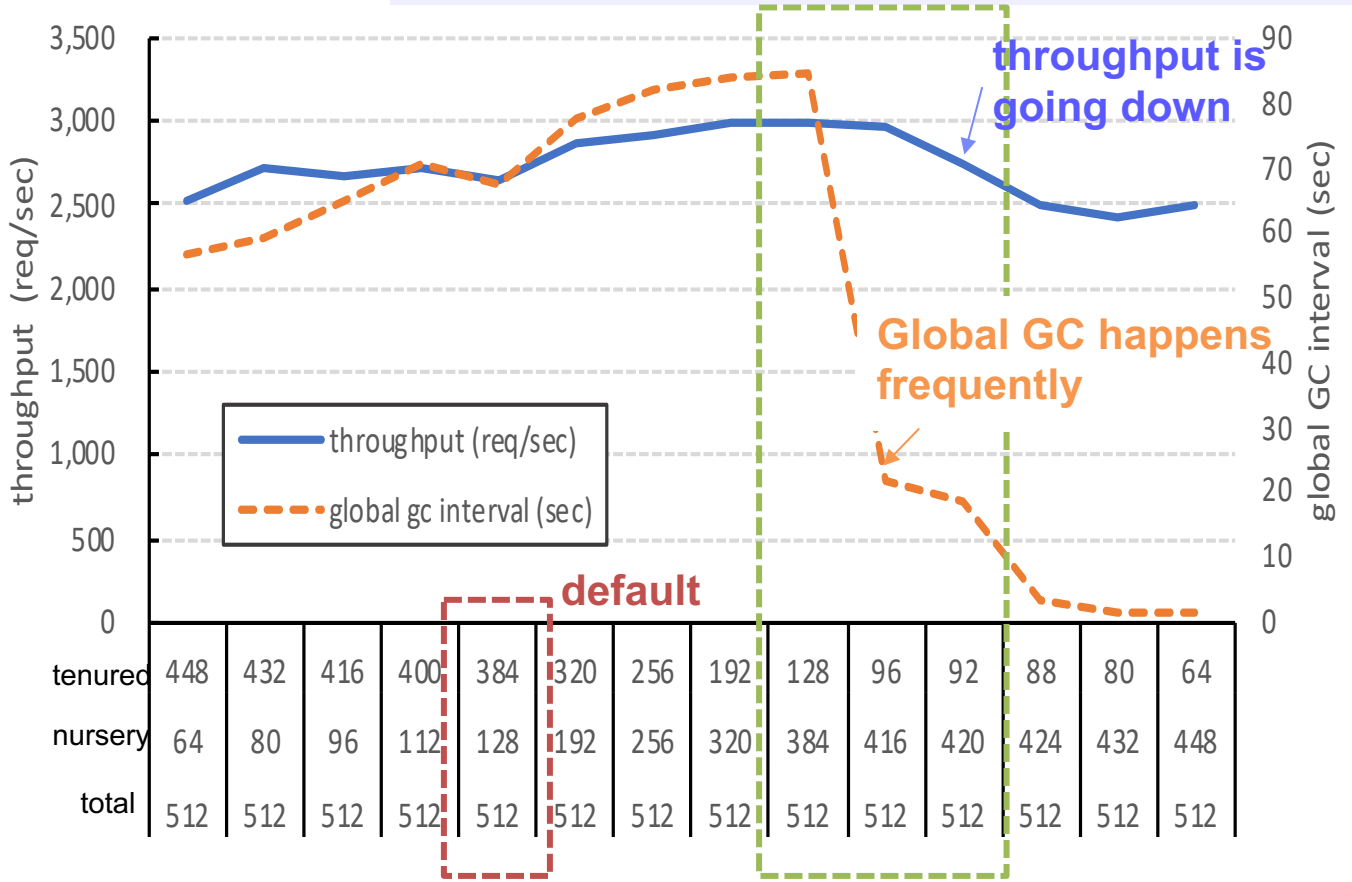
YCSB
- data size : 1KB
- record : 1M
- dist. : Uniform
- # client thread : 8
- workload A: read/update = 50/50
- workload B: read/update = 95/5



Memory Sizing: Liberty + DayTrader

```
default opt: -Xmx:512m -Xms:128m
```

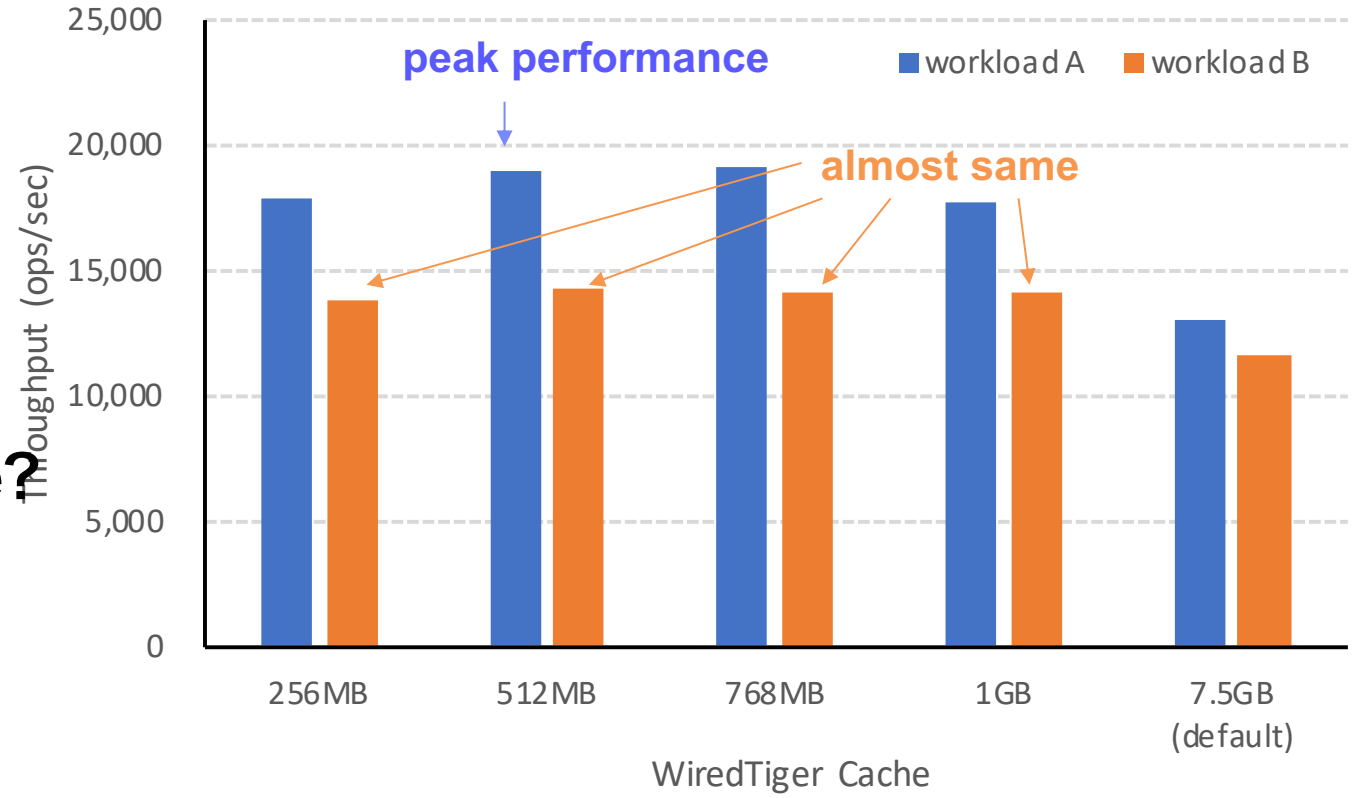
- **Heap size ratio affects performance**
  - Large nursery heap improves throughput
  - Too small tenured heap causes long gc pause time
- **What kind of advice can we make?**
  - what-if: `3 * current GC interval < latest GC interval`
  - advice: `nursery heap ++`



Memory Sizing: MongoDB + YCSB

```
spec:
  containers:
  - image: mongo:4.1.4
    resources:
      limits:
        memory: 1Gi
        cpu: 1
```

- **Mongo does not consider memory limit**
  - tries to reserve half of system memory as a cache..!
- **In-memory cache size affects performance**
  - workload A: 1.45x improvement
  - workload B: 1.20x improvement
- **What kind of advice can we make?**
  - what-if: `read/write ratio ≒ 1.0`
  - advice: `cache = total memory / 2`



Summary

- **Summary**
  - Built a sustainable config tuning framework on Kubernetes
  - Provided extendable/programmable plugin system
  - confirmed perf improvement for the apps on Kubernetes
- **Future**
  - Imports various expert knowledge (build a ecosystem)
  - Prepares Bayesian Optimization feature
  - Imports various ML model