运行k8s & fission

简单例子

- 步骤
  - 下载minikube，kubectl
    - minikube：本地运行k8s cluster
    - kubectl：用于与k8s通信的命令行
  - 运行minikube
    - minikube start
  - 下载fission
  - 运行fission
    - 创建函数
    - 运行函数
      - 运行函数时间花费大概有2～3秒；

```
~/Doc/1/202205/fission    fission function test --name hello-js
hello, world!
```

运行k8s & fission

使用newdeploy运行

- ```
  ~/Doc/1/202205/fission  fission fn create --name hello --env py
  thon --code hello.py --executortype newdeploy --minscale 1 --maxsca
  le 3 --targetcpu 50
  Package 'hello-d1b97f2f-03f9-44d4-b72b-8b9e25f8731a' created
  function 'hello' created
  ```

- ```
  ~/Doc/1/202205/fission  fission function test --name hello
  Hello, world!
  ```

ServerlessBench

简介

- 2.0提供了**测试serverless平台**的功能；
  - github上暂时还没有相关的测试平台的文档（只有1.0的相关测试用例）
  - 测试文章被删除

    华为云联合上海交大发布Serverless基准测试平台 - 知乎专栏

    1 day ago — 摘要：华为云联合上海交大重磅推出ServerlessBench 2.0，为社区提供涵盖12类 ...

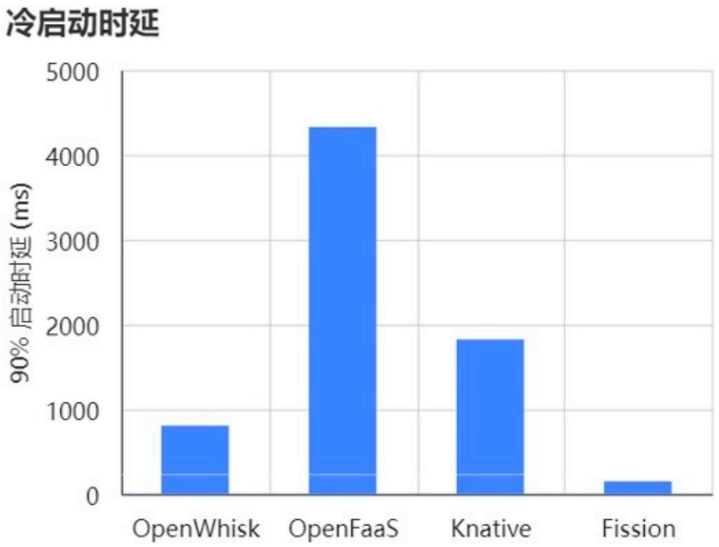
    - 平台OpenWhisk，OpenFaaS, Knative，Fission的入口函数都不相同。
- 测试结果

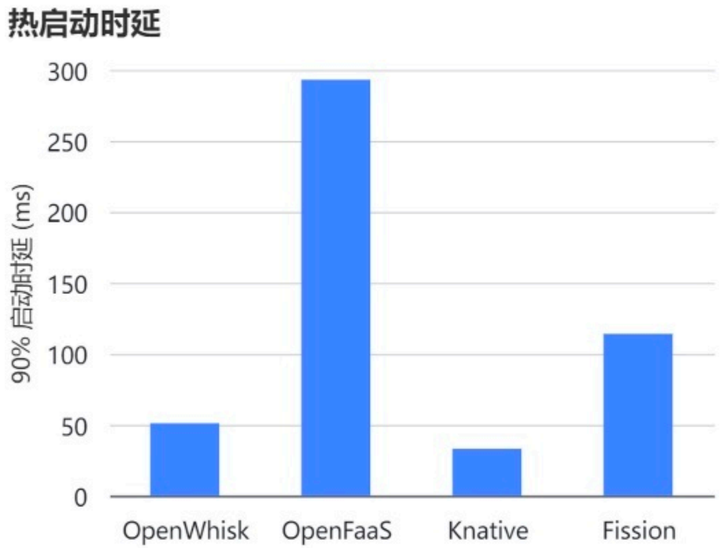
测试名称	应用范例	平台			
		OpenWhisk	OpenFaaS	Knative	Fission
TC1: Cold/Warm Startup	Float Operation	✓	✓	✓	✓
	Numpy matmul	✓	✓	✓	✓
	Gzip Compression	✓	✓	✓	✓
TC2: Cold/Warm Execution	Float Operation	✓	✓	✓	✓
	Numpy matmul	✓	✓	✓	✓
	Gzip Compression	✓	✓	✓	✓
TC3: QoS guaranteed Concurrency Execution	Float Operation	✓	✓	✓	✓
	Numpy matmul	✓	✓	✓	✓
TC4: Scaling Speed	Float Operation	✓	✓	✓	✓
TC5: Price	Float Operation	✓	✓	✓	✓
	Numpy matmul	✓	✓	✓	✓
More in future (e.g., chained apps, ...)					

ServerlessBench

测试结果

- 冷、热启动时延





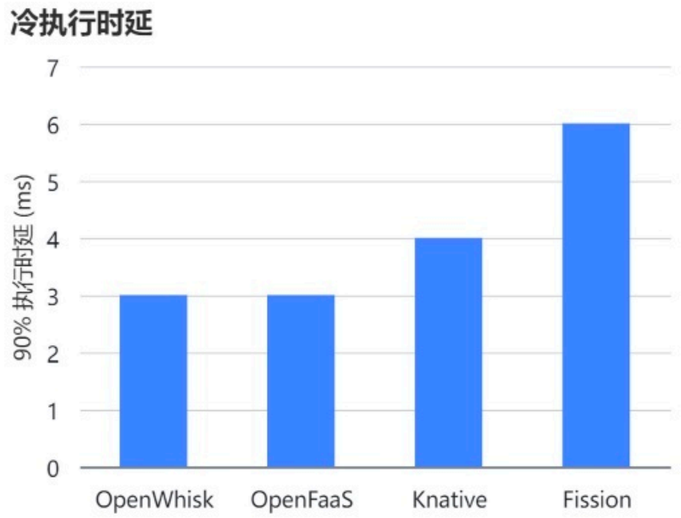
- 冷启动时延　　热启动时延

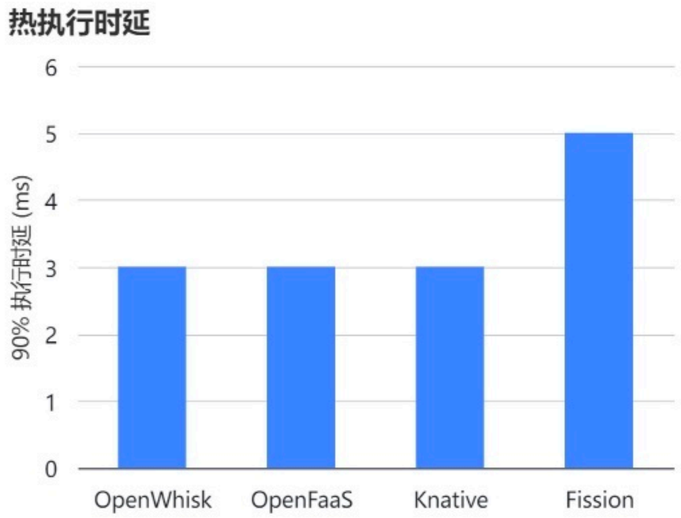
- fission冷启动时间最短，归功于preheating technology

ServerlessBench

测试结果

- 冷、热**执行**时延



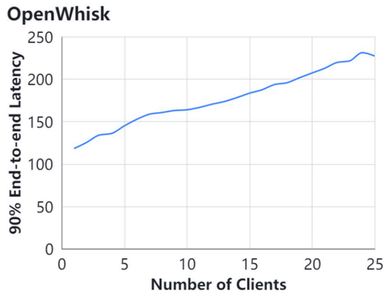


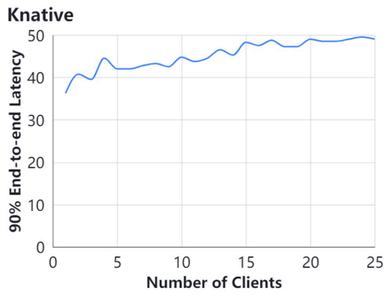
- 冷执行时延　　热执行时延
- 执行函数的时延

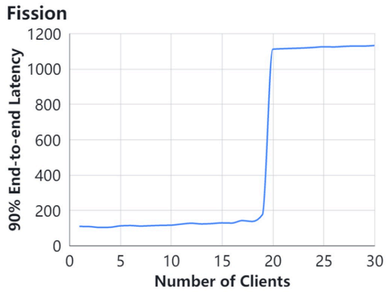
ServerlessBench

测试结果

- 并发请求对QoS(quality of service)的影响

- 

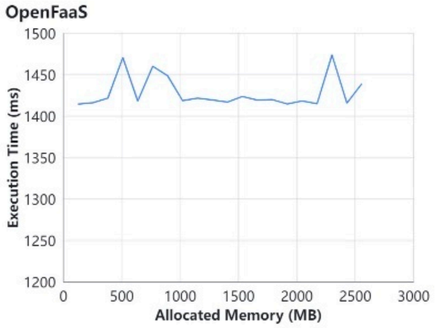


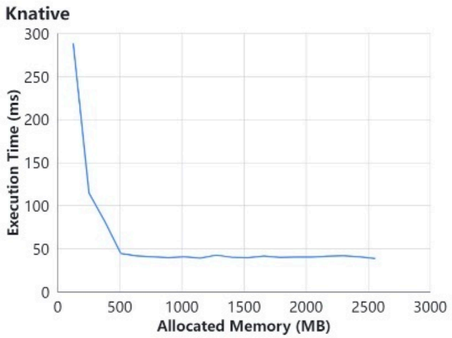


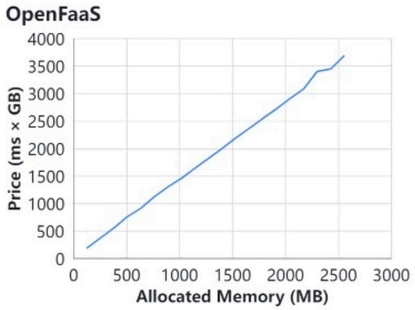
ServerlessBench

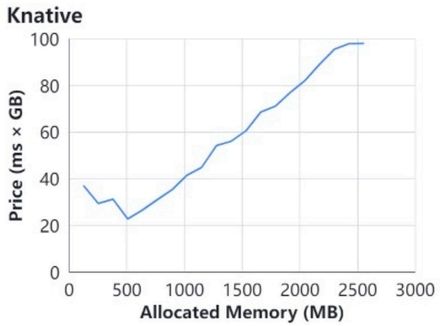
测试结果

- Platform cost performance





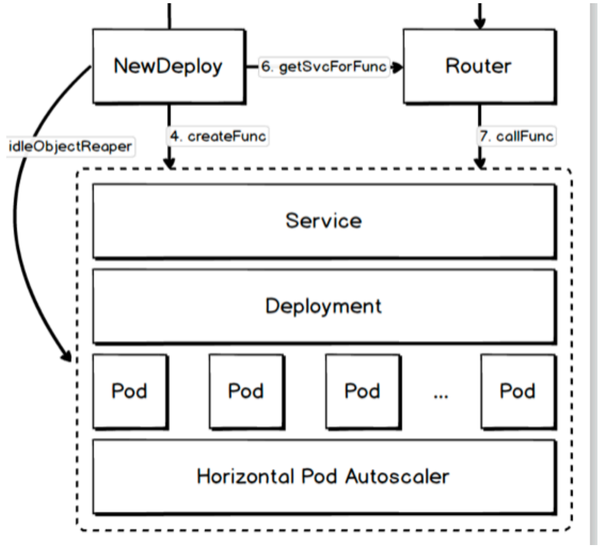




-
- 逐渐调整分配给该函数的资源，测量执行延迟与分配资源数量的关系

Fission createFunc

- 没有找到相关的更深入的文档；
- k8s Deployment - abstract over pods
  - deployment之下的操作都是由k8s来自动处理
    - 包括：pod、container、replica
  - 如果要更改deployment之下的状态（如pod、container），修改deployment的配置文件，k8s会自动响应配置文件
    - kubectl apply -f nginx-deployment.yaml




```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

```
~/Doc/1/202205/fission  kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx-deployment created
```

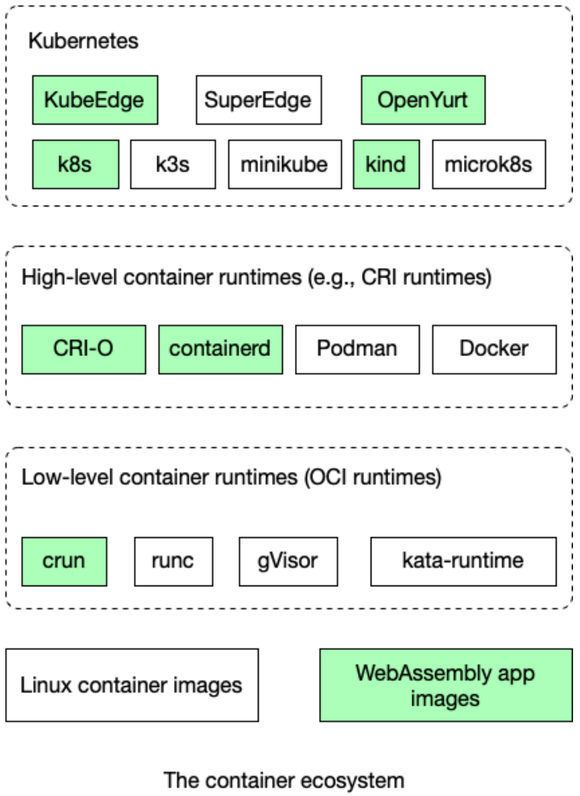
```
~/Doc/1/202205/fission  kubectl get deployment
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           31s
```

```
~/Doc/1/202205/fission  kubectl get pod
NAME                               READY   STATUS    RESTARTS   AGE
nginx-deployment-6c8b449b8f-4q5tk  1/1     Running   0          40s
nginx-deployment-6c8b449b8f-n6rq6  1/1     Running   0          40s
```

Run WebAssembly container images in Kubernetes

wasmEdge in k8s

- Container runtimes
  - crun
    - Has **WasmEdge** support baked in
    - Load and run **Wasm OCI images**
- CRI runtimes
  - configure and use high level container runtimes, such as **CRI-O** and **containerd**, to load and run **WebAssembly OCI images on top of low level container runtimes**

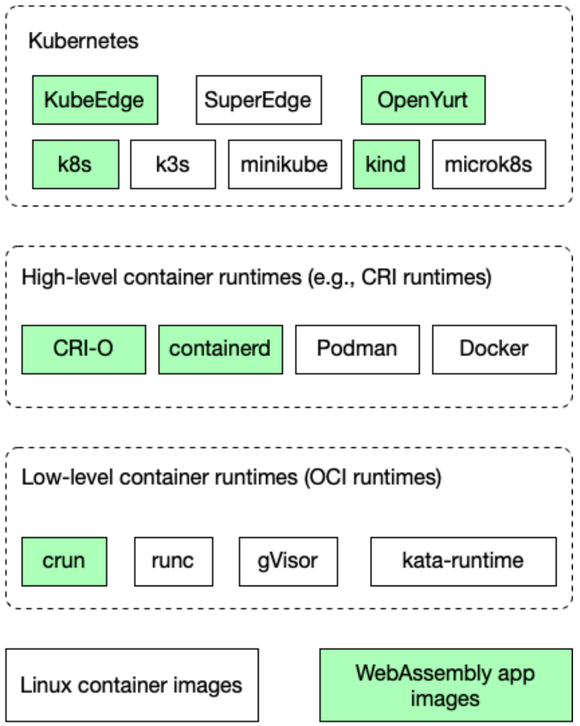


The container ecosystem

Run WebAssembly container images in Kubernetes

wasmEdge in k8s

- CRI runtimes
  - pulls container images from registries (e.g., Docker Hub),
  - manages them on disk
  - **launches a lower-level runtime to run container processes**
- K8s example
  - A simple WebAssembly **app**
  - A WebAssembly-based **HTTP service**



The container ecosystem

参考

- Kubernetes Tutorial for Beginners
- **WasmEdge in Kubernetes**
- Serverlessbench 2.0: Huawei cloud and Shanghai Jiaotong University released the serverless benchmark platform