On expanding the Kubernetes scheduler extender

Asser N Tantawi

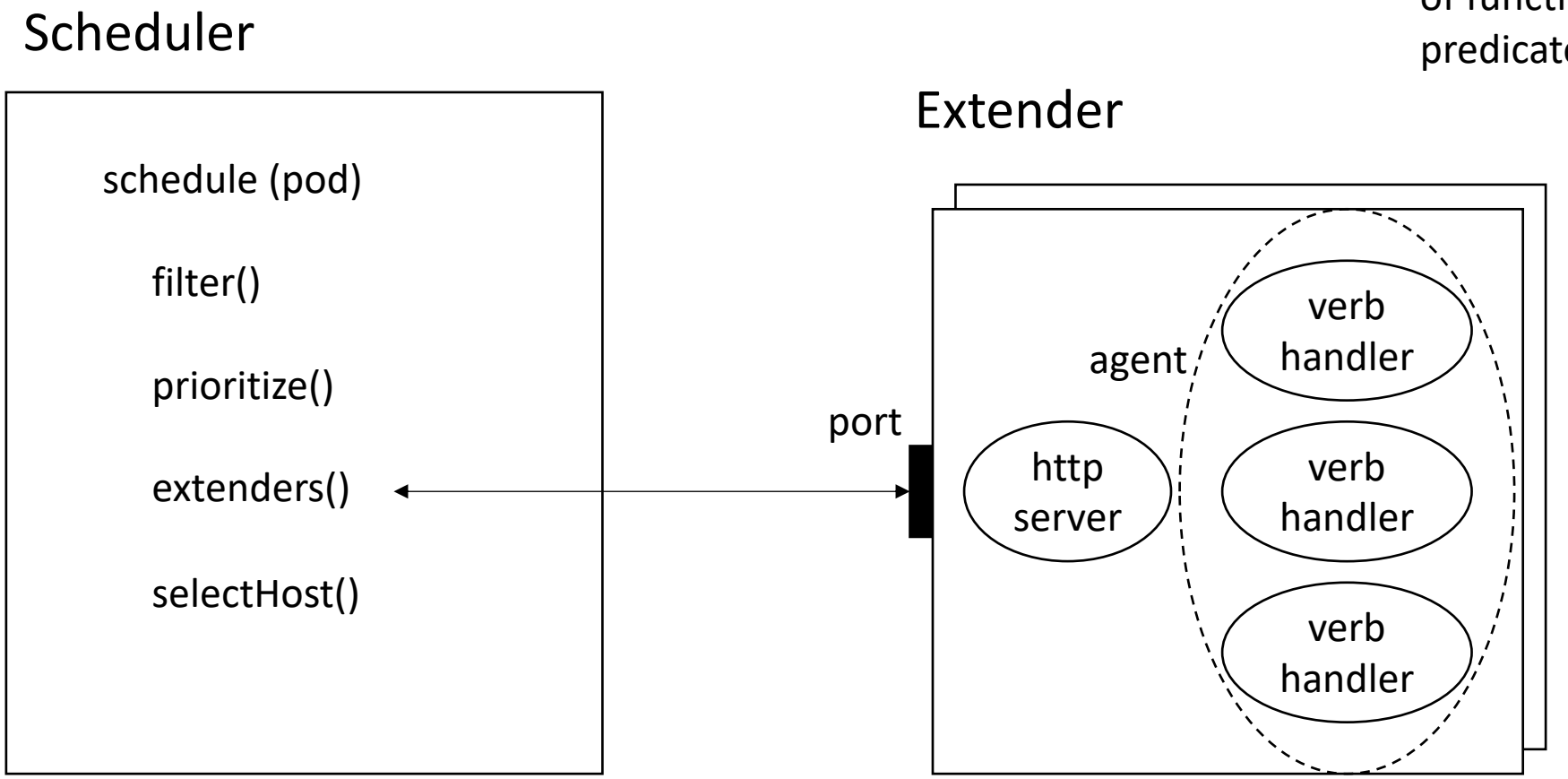
IBM Research

Background

- We have developed a k8s scheduler extender
  - schedules pods based on observed resource usage (CPU and memory)
  - called the kube-safe-scheduler
  - open-sourced at https://github.com/IBM/kube-safe-scheduler
- Now, we need to expand the extender with additional functionalities
  - adaptive bin packing, policy objective based, risk-aware, network-aware, …
- We need to produce an architecture for the extender to
  - allow for ease of expandability
  - provide a development environment isolation
  - enable selective functional deployment

Overview of the Kubernetes scheduler extender

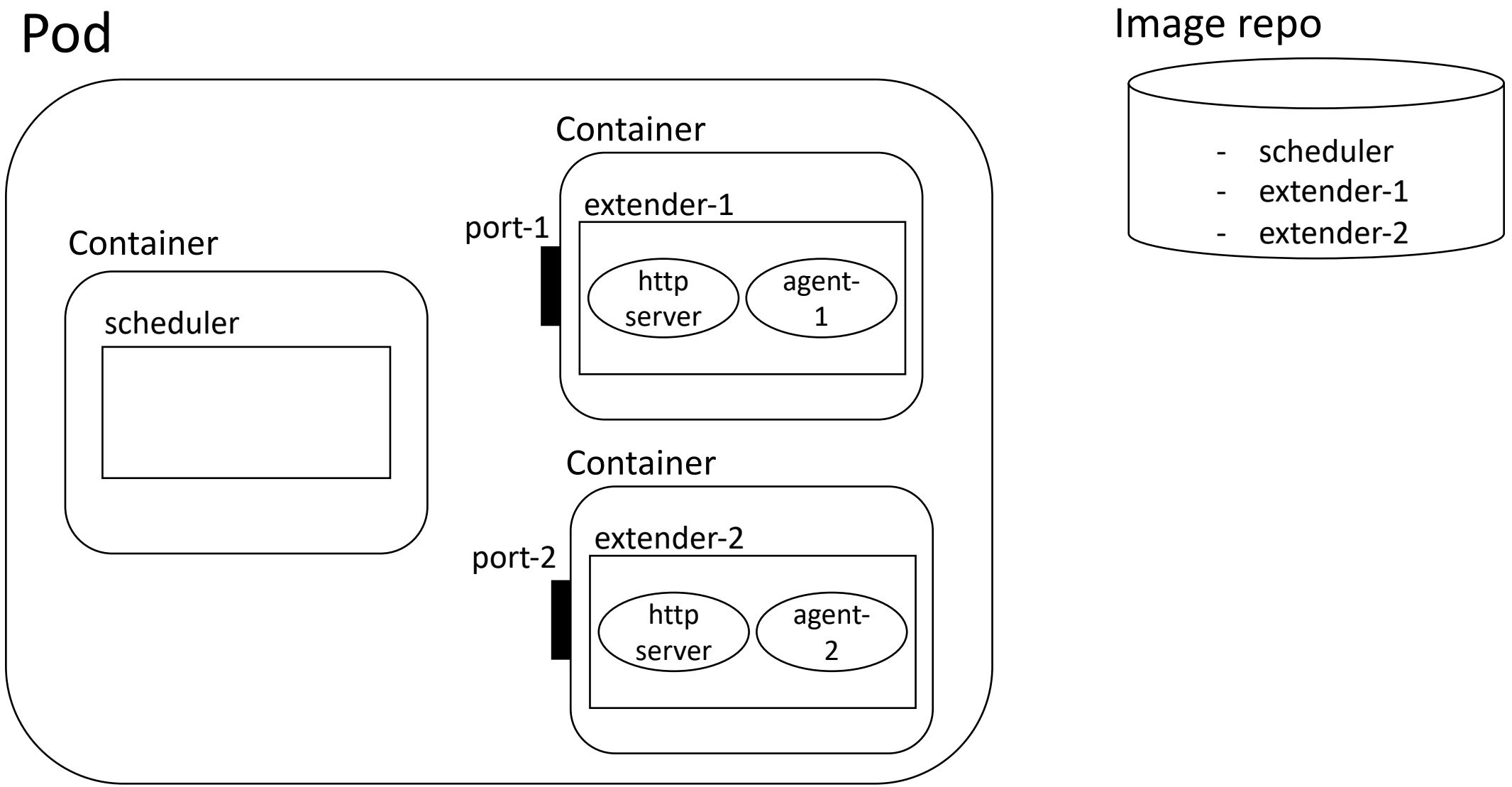
Define *Agent* as the implementation of a set of functionally-related predicates/priorities



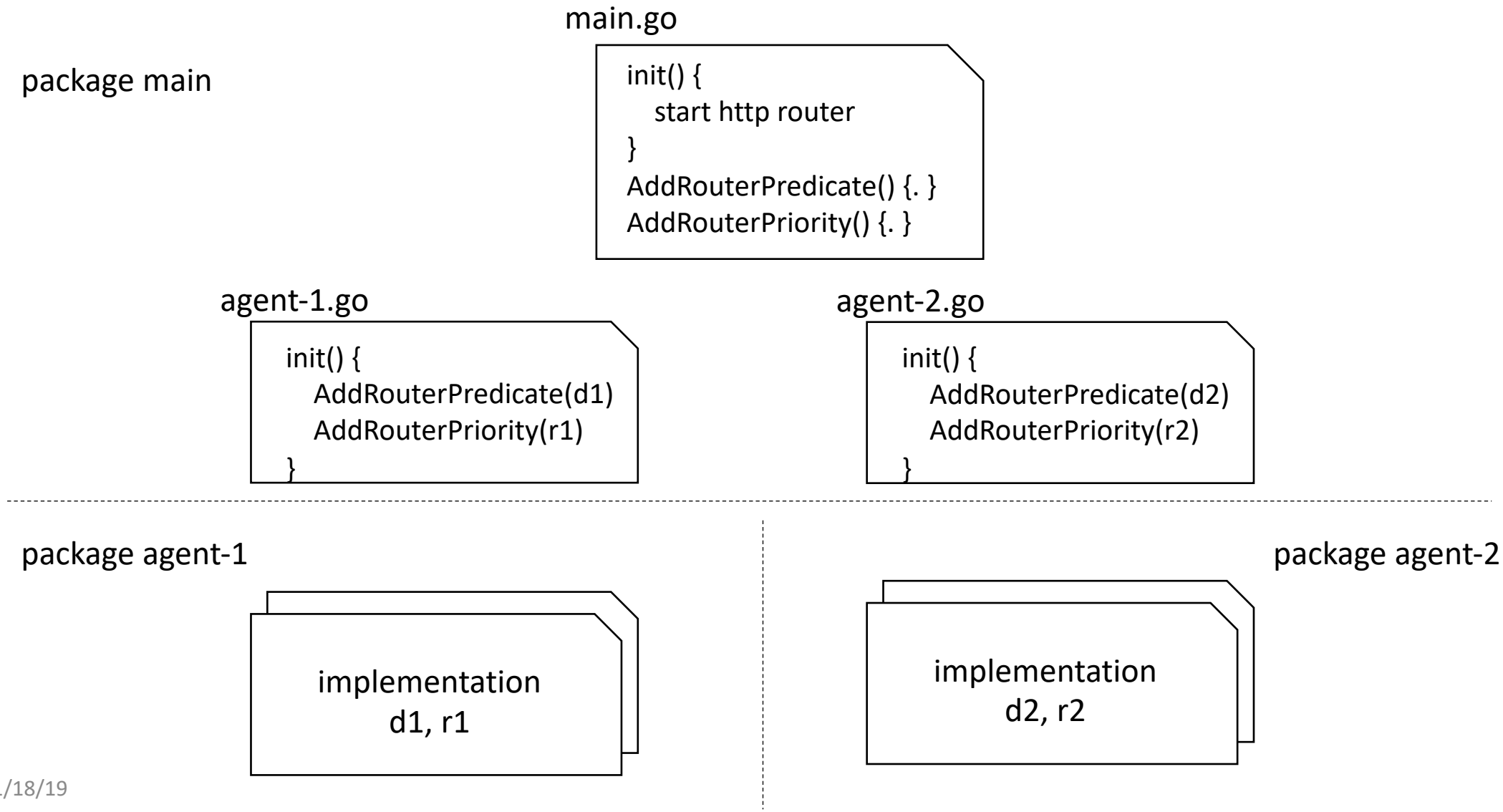
Config

```
"extender": {
    "urlPrefix": "http://127.0.0.1:12346/scheduler",
    "filterVerb": "filter",
    "prioritizeVerb": "prioritize",
    "weight": 5,
    "enableHttps": false,
    "httpTimeout": 5
    }
```

Deployment of scheduler and multiple extenders in a pod



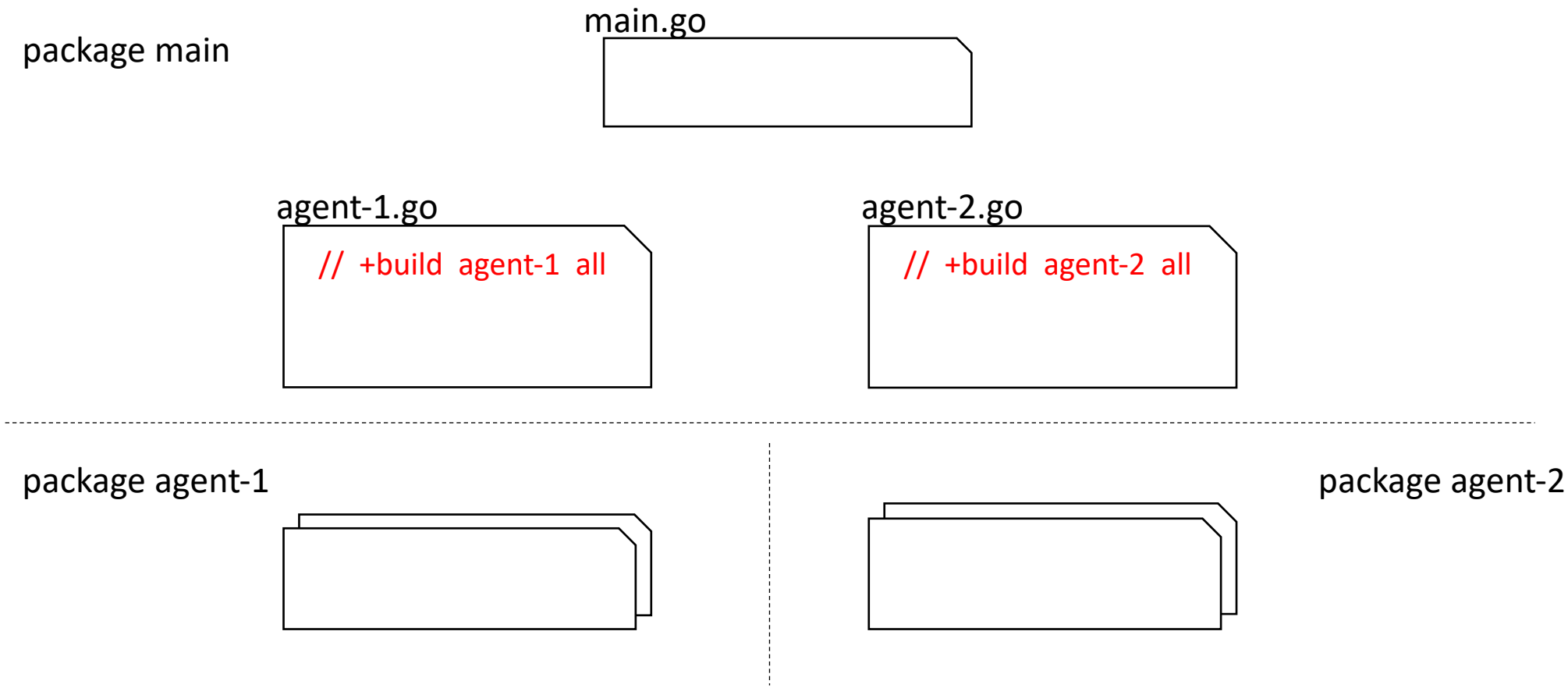
Code structure for easy agent development and expandability



Easy extender building

optional:

- *build all agents in one extender container using the all tag*
- *or subset of agents using subset of tags*



go install –tags agent-1 …

go install –tags agent-2 …

Example: Scheduler with two extenders (one agent each)

- Agent-1
  - Name: **safe**
  - Description: usage-based safe scheduling
  - Predicates: safe-overload
  - Priority functions: safe-overload, safe-balance
  - Stateless (between subsequent calls to extender)
- Agent-2
  - Name: **pigeon**
  - Description: bin packing objective scheduling
  - Priority functions: pigeon-holing
  - Stateful: keep handle to state model

package main
main.go

```
func init() {
    InitMain()
}

// InitMain : initialize main - should be done before any other init()
func InitMain() {
    if !initialized {
        // init logger
        initLogger()

        // init router
        router = httprouter.New()
        AddVersion(router)

        initialized = true
    }
}

// AddRouterPredicate : add a predicate to the router
func AddRouterPredicate(p Predicate) {
    AddPredicate(router, p)
}

// AddRouterPriority : add a priority function to the router
func AddRouterPriority(p Prioritize) {
    AddPrioritize(router, p)
}
```

package main
safe.go

```go
// +build safe all

package main

import (
    v1 "k8s.io/api/core/v1"
    schedulerapi "k8s.io/kubernetes/pkg/scheduler/api"

    safe "kube-safe-scheduler/safe"
)

// Initialize agent
func init() {
    InitMain()

    // add agent predicates
    AddRouterPredicate(SafeOverloadPredicate)

    // add agent priority functions
    AddRouterPriority(SafeBalancePriority)
    AddRouterPriority(SafeOverloadPriority)
}

/**
 * Define safe overload predicate
 */

// SafeOverloadPredicate : safe node overload predicate
var SafeOverloadPredicate = Predicate{
    Name: safe.PredicateSafeOverloadName,
    Func: func(pod v1.Pod, node v1.Node) (bool, error) {
        return safe.PredicateFunc(pod, node)
    },
}
```

package safe implementation

```go
package safe

import (
    "bytes"
    "fmt"
    "log"

    v1 "k8s.io/api/core/v1"
    schedulercache "k8s.io/kubernetes/pkg/scheduler/cache"
)

// PredicateFunc : safe predicate function
func PredicateFunc(pod v1.Pod, node v1.Node) (bool, error) {

    log.Printf("==> SafeOverloadPredicate: applying SafePredicate for pod %s on node %s ... \n",
        pod.Name, node.Name)

    var buf bytes.Buffer

    podRequest := getResourceRequest(&pod)
    okCPU, outCPU, _ := CPUSafePredicate(&node, podRequest)
    fmt.Fprintf(&buf, "%s", outCPU)

    okMemory, outMemory, _ := MemorySafePredicate(&node, podRequest)
    fmt.Fprintf(&buf, "%s", outMemory)

    okOverall := okCPU && okMemory

    fmt.Fprintf(&buf, "okCPU = %v; okMemory = %v; okOverall = %v; \n", okCPU, okMemory, okOverall)
    log.Print(buf.String())

    return okOverall, nil
}
```

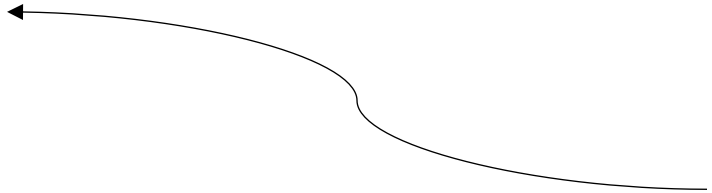
package main
pigeon.go

```go
// +build pigeon all

package main

import (
    pigeon "kube-safe-scheduler/pigeon"
    "log"

    v1 "k8s.io/api/core/v1"
    schedulerapi "k8s.io/kubernetes/pkg/scheduler/api"
)

// Initialize agent
func init() {
    InitMain()

    // create a pigeon client and register it (for persistence)
    log.Println("info: creating pigeon instance ...")
    pigeon.PigeonAgent = pigeon.NewAgent()

    // add agent priority functions
    AddRouterPriority(PigeonPriority)
}

/**
 * Define pigeon priority functions
 */

// PigeonPriority : pigeon priority function
var PigeonPriority = Prioritize{
    Name: pigeon.PriorityPigeonName,
    Func: func(pod v1.Pod, nodes []v1.Node) (*schedulerapi.HostPriorityList, error) {
        return pigeon.PriorityFunc(pod, nodes)
    },
}
```

create a handle for stateful operations

package pigeon implementation

```go
package pigeon

import (
    "fmt"
    "math"

    v1 "k8s.io/api/core/v1"
    schedulerapi "k8s.io/kubernetes/pkg/scheduler/api"
)

var (
    // PigeonAgent :
    PigeonAgent *Agent
)

// Agent :
type Agent struct {
    client *Client
}

// NewAgent creates a pigeon agent
func NewAgent() *Agent {

    configFile := "pigeon.cfg"
    client := NewClient(configFile)

    return &Agent{
        client: client,
    }
}
```



keep a handle for stateful operations

package pigeon implementation

```go
package pigeon

import (
    v1 "k8s.io/api/core/v1"
    schedulerapi "k8s.io/kubernetes/pkg/scheduler/api"
)

// PriorityFunc : compute pigeon priority function
func PriorityFunc(pod v1.Pod, nodes []v1.Node) (*schedulerapi.HostPriorityList, error) {
    var priorityList schedulerapi.HostPriorityList
    priorityList = make([]schedulerapi.HostPriority, len(nodes))

    agent := PigeonAgent
    if agent != nil {
        agent.UpdateState(&nodes)
        rankMap, err := agent.ComputeRank(&pod)
        if err == nil {
            var i int = 0
            for k, v := range *rankMap {
                priorityList[i] = schedulerapi.HostPriority{
                    Host:  k,
                    Score: v,
                }
                i++
            }
        }
    } else {
        for i := 0; i < len(nodes); i++ {
            priorityList[i] = schedulerapi.HostPriority{
                Host:  nodes[i].GetName(),
                Score: 0,
            }
        }
    }
    return &priorityList, nil
}
```

use the handle

A Dockerfile for each extender/agent(s)

Dockerfile.safe

Build safe extender image

```
FROM golang:1.10-alpine as builder
ENV CGO_ENABLED=0
ENV GOOS=linux
ENV GOARCH=amd64

ARG VERSION=0.0.1

# build
WORKDIR /go/src/kube-safe-scheduler
COPY . .
RUN go install -tags safe -ldflags "-s -w -X main.version=$VERSION" kube-
safe-scheduler

# runtime image
FROM gcr.io/google_containers/ubuntu-slim:0.14
COPY --from=builder /go/bin/kube-safe-scheduler /usr/bin/kube-safe-
scheduler
ENTRYPOINT ["kube-safe-scheduler"]
```

Build pigeon extender image

Dockerfile.pigeon

```
# Build image
FROM golang:1.10-alpine as builder
ENV CGO_ENABLED=1
ENV GOOS=linux
ENV GOARCH=amd64

ARG VERSION=0.0.1

# (note: alpine-sdk is needed to avoid segmentation fault issue with cgo libraries)
# (additional: rsync linux-headers mercurial)
RUN apk add --update git alpine-sdk build-base bash\
    && apk update && apk upgrade

ENV GOPATH=/go
ENV GOBASE=$GOPATH/src/kube-safe-scheduler
ENV GOPIGEON=$GOBASE/pigeon

ENV CPATH=/c
ENV CBASE=$CPATH/pigeon
ENV CSRC=$CBASE/src
ENV CBIN=$CBASE/Debug
ENV OUT=/out

RUN mkdir -p $GOBASE              \
    && mkdir -p $CBASE            \
    $$ mkdir -p $OUT

COPY . $GOBASE

# build C code
RUN cp -r $GOBASE/pigeon_c/* $CBASE                \
    && make -C $CBIN all

RUN ln -sf $CBIN/libpigeon.so  $GOPIGEON/pigeonmodule/libs/libpigeon.so   \
    && ln -sf $CSRC/* $GOPIGEON/pigeonmodule/include

# build Go code (https://golang.org/cmd/link/)
RUN go install -tags pigeon -ldflags "-s -w -X main.version=$VERSION" kube-safe-scheduler

RUN cp $GOPATH/bin/kube-safe-scheduler $OUT/kube-safe-scheduler     \
    && cp $GOBASE/pigeon_c/pigeon.cfg $OUT/pigeon.cfg                                    \
    && cp $CBIN/libpigeon.so $OUT/libpigeon.so


# Runtime image
#FROM gcr.io/google_containers/ubuntu-slim:0.14

FROM alpine
RUN apk add --update bash         \
    && apk update && apk upgrade

ENV OUT=/out
ENV TARGET /usr/local/bin
ENV LD_LIBRARY_PATH $TARGET
WORKDIR $TARGET

COPY --from=builder $OUT/kube-safe-scheduler kube-safe-scheduler
COPY --from=builder $OUT/pigeon.cfg pigeon.cfg
COPY --from=builder $OUT/libpigeon.so libpigeon.so

ENTRYPOINT ["kube-safe-scheduler"]
```

Image repository

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
kube-sched-ext-pigeon	0.0	a130400bfaa3	24 hours ago	25.5MB
kube-sched-ext-safe	0.0	ff4d4e34596e	24 hours ago	56.6MB
kube-sched-ext-congestion	0.0	3f055aa74979	24 hours ago	56.5MB
k8s-assessor	0.0	b521ea60ba27	8 days ago	149MB
nginx	latest	540a289bab6c	3 weeks ago	126MB
python	3-alpine	204216b3821e	3 weeks ago	111MB

Deploy scheduler
Deploy safe extender
Deploy pigeon extender

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-scheduler
  namespace: kube-system
  labels:
    app: my-scheduler
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-scheduler
  template:
    metadata:
      labels:
        app: my-scheduler
    spec:
      volumes:
      - name: my-scheduler-config
        configMap:
          name: my-scheduler-config
      containers:
      - name: my-scheduler-ctr
        image: gcr.io/google_containers/hyperkube:v1.13.5
        imagePullPolicy: IfNotPresent
        args:
        - kube-scheduler
        - --config=/my-scheduler/config.yaml
        - -v=4
        volumeMounts:
        - name: my-scheduler-config
          mountPath: /my-scheduler
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-scheduler-policy
  namespace: kube-system
data:
 policy.cfg : |
  {
    "kind" : "Policy",
    "apiVersion" : "v1",
    "extenders" : [{
      "urlPrefix": "http://localhost:5401/scheduler",
      "filterVerb": "predicates/safe-overload",
      "prioritizeVerb": "priorities/safe-overload",
      "preemptVerb": "",
      "bindVerb": "",
      "weight": 1,
      "enableHttps": false,
      "nodeCacheCapable": false
    },
    {
      "urlPrefix": "http://localhost:5402/scheduler",
      "filterVerb": "",
      "prioritizeVerb": "priorities/pigeon-holing",
      "preemptVerb": "",
      "bindVerb": "",
      "weight": 1,
      "enableHttps": false,
      "nodeCacheCapable": false
    }],
    "hardPodAffinitySymmetricWeight" : 10
  }
```

```yaml
    - name: my-extender-ctr-1
      image: kube-sched-ext-safe:0.0
      imagePullPolicy: IfNotPresent
      livenessProbe:
        httpGet:
          path: /version
          port: 5401
      readinessProbe:
        httpGet:
          path: /version
          port: 5401
      ports:
        - containerPort: 5401
      env:
        - name: SAFEUTILIZATION
          value: "90"
        - name: SAFEPERCENTILE
          value: "30"
        - name: SAFEPRINTTABLE
          value: "false"
        - name: SAFEFORECASTWEIGHT
          value: "20"
        - name: HTTP_PORT
          value: "5401"
    - name: my-extender-ctr-2
      image: kube-sched-ext-pigeon:0.0
      imagePullPolicy: IfNotPresent
      livenessProbe:
        httpGet:
          path: /version
          port: 5402
      readinessProbe:
        httpGet:
          path: /version
          port: 5402
      ports:
        - containerPort: 5402
      env:
        - name: NUM_RESOURCES
          value: "2"
        - name: POLICY_OBJECTIVE
          value: "A_BINPACK"
        - name: HTTP_PORT
          value: "5402"
```

Container startup

scheduler

```
tantawi@assers-mbp kube-safe-scheduler % kc logs -f my-scheduler-799789869c-vs5j4  my-scheduler-ctr -n kube-system


I1115 14:30:19.347267       1 factory.go:1167] Creating scheduler from configuration: {{ } [] [] [{http://localhost:5401/scheduler predicates/safe-overload  priorities/safe-overload 1  false <nil> 0s false [] false}
{http://localhost:5402/scheduler   priorities/pigeon-holing 1  false <nil> 0s false [] false}] 10 false}
I1115 14:30:19.347331       1 factory.go:1176] Using predicates from algorithm provider 'DefaultProvider'
I1115 14:30:19.347338       1 factory.go:1191] Using priorities from algorithm provider 'DefaultProvider'
I1115 14:30:19.347341       1 factory.go:1208] Creating extender with config {URLPrefix:http://localhost:5401/scheduler FilterVerb:predicates/safe-overload PreemptVerb: PrioritizeVerb:priorities/safe-overload
Weight:1 BindVerb: EnableHTTPS:false TLSConfig:<nil> HTTPTimeout:0s NodeCacheCapable:false ManagedResources:[] Ignorable:false}
I1115 14:30:19.347357       1 factory.go:1208] Creating extender with config {URLPrefix:http://localhost:5402/scheduler FilterVerb: PreemptVerb: PrioritizeVerb:priorities/pigeon-holing Weight:1 BindVerb:
EnableHTTPS:false TLSConfig:<nil> HTTPTimeout:0s NodeCacheCapable:false ManagedResources:[] Ignorable:false}
I1115 14:30:19.347368       1 factory.go:1256] Creating scheduler with fit predicates 'map[MaxEBSVolumeCount:{} MaxAzureDiskVolumeCount:{} NoDiskConflict:{} PodToleratesNodeTaints:{}
```

safe extender

```
tantawi@assers-mbp kube-safe-scheduler % kc logs -f my-scheduler-799789869c-vs5j4 -c  my-extender-ctr-1 -n kube-system
[  info ] 2019/11/15 14:30:18 main.go:87: Log level was set to INFO
[  info ] 2019/11/15 14:30:18 main.go:131: server starting on port 5401
```

pigeon extender

```
tantawi@assers-mbp kube-safe-scheduler % kc logs -f my-scheduler-799789869c-vs5j4  my-extender-ctr-2 -n kube-system
[  info ] 2019/11/15 14:30:18 main.go:87: Log level was set to INFO
parameters in config file pigeon.cfg:
[  info ] 2019/11/15 14:30:18 pigeon.go:18: creating pigeon instance ...
policyResourceIndex = 0
policyObjective = A_BINPACK

===> Created pigeon client
Version=0.0.0
setting number of resources from environment to 2
numResources=2
setting policy objective from environment to A_BINPACK
policyObjective=A_BINPACK
[  info ] 2019/11/15 14:30:18 main.go:131: server starting on port 5402
```

Scheduling a test pod

scheduler

```
I1115 14:32:01.493892       1 factory.go:1392] About to try and schedule pod default/test-pod
I1115 14:32:01.493920       1 scheduler.go:525] Attempting to schedule pod: default/test-pod
I1115 14:32:01.540317       1 scheduler_binder.go:207] AssumePodVolumes for pod "default/test-pod", node "worker-node1"
I1115 14:32:01.540339       1 scheduler_binder.go:217] AssumePodVolumes for pod "default/test-pod", node "worker-node1": all PVCs bound and nothing to do
I1115 14:32:01.540403       1 factory.go:1604] Attempting to bind test-pod to worker-node1
```

Scheduling a test pod

safe extender

```
[  info ] 2019/11/15 14:32:01 routes.go:30: safe-overload ExtenderArgs =
[  info ] 2019/11/15 14:32:01 predicate.go:15: ==> SafeOverloadPredicate: applying SafePredicate for pod test-pod on node master-node ...
[  info ] 2019/11/15 14:32:01 predicate.go:30: Checking cpu fit based on usage history :
meanFreeCPU = 900; stdFreeCPU = 200
freeAvg = 900.000000; freeStdev = 200.000000
capacity = 2000.000000; demand = 200.000000
usedAvg = 1100.000000; usedStdev = 200.000000
BetaDistribution: alpha = 14.137500; beta = 7.612500; mean = 0.650000; var = 0.010000; m1 = 0.650000; m2 = 0.432500; m3 = 0.293872;
mu = 0.650000; sigma = 0.100000; risk(0.900000) = 0.001099      (<= 0.300000) accepted!
Checking memory fit based on usage history :
No valid memory statistics.
okCPU = true; okMemory = true; okOverall = true;
[  info ] 2019/11/15 14:32:01 predicate.go:15: ==> SafeOverloadPredicate: applying SafePredicate for pod test-pod on node worker-node1 ...
[  info ] 2019/11/15 14:32:01 predicate.go:30: Checking cpu fit based on usage history :
meanFreeCPU = 1600; stdFreeCPU = 50
freeAvg = 1600.000000; freeStdev = 50.000000
capacity = 2000.000000; demand = 200.000000
usedAvg = 400.000000; usedStdev = 50.000000
BetaDistribution: alpha = 100.500000; beta = 234.500000; mean = 0.300000; var = 0.000625; m1 = 0.300000; m2 = 0.090625; m3 = 0.027564;
mu = 0.300000; sigma = 0.025000; risk(0.900000) = 0.000000      (<= 0.300000) accepted!
Checking memory fit based on usage history :
No valid memory statistics.
okCPU = true; okMemory = true; okOverall = true;
[  info ] 2019/11/15 14:32:01 predicate.go:15: ==> SafeOverloadPredicate: applying SafePredicate for pod test-pod on node worker-node2 ...
[  info ] 2019/11/15 14:32:01 predicate.go:30: Checking cpu fit based on usage history :
meanFreeCPU = 1700; stdFreeCPU = 200
freeAvg = 1700.000000; freeStdev = 200.000000
capacity = 2000.000000; demand = 200.000000
usedAvg = 300.000000; usedStdev = 200.000000
BetaDistribution: alpha = 4.437500; beta = 13.312500; mean = 0.250000; var = 0.010000; m1 = 0.250000; m2 = 0.072500; m3 = 0.023631;
mu = 0.250000; sigma = 0.100000; risk(0.900000) = 0.000000      (<= 0.300000) accepted!
Checking memory fit based on usage history :
No valid memory statistics.
okCPU = true; okMemory = true; okOverall = true;
```

Scheduling a test pod

safe extender

```
[  info ] 2019/11/15 14:32:01 priority.go:24: ==> SafePriority: calculating priority for pod test-pod on node master-node ...
[  info ] 2019/11/15 14:32:01 priority.go:53: Calculating priority based on usage history ...
CPU usage statistics:
meanFreeCPU = 900; stdFreeCPU = 200
freeAvg = 900.000000; freeStdev = 200.000000
capacity = 2000.000000; demand = 200.000000
usedAvg = 1100.000000; usedStdev = 200.000000
BetaDistribution: alpha = 14.137500; beta = 7.612500; mean = 0.650000; var = 0.010000; m1 = 0.650000; m2 = 0.432500; m3 = 0.293872;
mu = 0.650000; sigma = 0.100000; risk(0.900000) = 0.001099; riskFraction = 0.003662
CPUScore = 9
Memory usage statistics:
No valid memory statistics.
overallScore = 9
[  info ] 2019/11/15 14:32:01 priority.go:24: ==> SafePriority: calculating priority for pod test-pod on node worker-node1 ...
[  info ] 2019/11/15 14:32:01 priority.go:53: Calculating priority based on usage history ...
CPU usage statistics:
meanFreeCPU = 1600; stdFreeCPU = 50
freeAvg = 1600.000000; freeStdev = 50.000000
capacity = 2000.000000; demand = 200.000000
usedAvg = 400.000000; usedStdev = 50.000000
BetaDistribution: alpha = 100.500000; beta = 234.500000; mean = 0.300000; var = 0.000625; m1 = 0.300000; m2 = 0.090625; m3 = 0.027564;
mu = 0.300000; sigma = 0.025000; risk(0.900000) = 0.000000; riskFraction = 0.000000
CPUScore = 10
Memory usage statistics:
No valid memory statistics.
overallScore = 10
[  info ] 2019/11/15 14:32:01 priority.go:24: ==> SafePriority: calculating priority for pod test-pod on node worker-node2 ...
[  info ] 2019/11/15 14:32:01 priority.go:53: Calculating priority based on usage history ...
CPU usage statistics:
meanFreeCPU = 1700; stdFreeCPU = 200
freeAvg = 1700.000000; freeStdev = 200.000000
capacity = 2000.000000; demand = 200.000000
usedAvg = 300.000000; usedStdev = 200.000000
BetaDistribution: alpha = 4.437500; beta = 13.312500; mean = 0.250000; var = 0.010000; m1 = 0.250000; m2 = 0.072500; m3 = 0.023631;
mu = 0.250000; sigma = 0.100000; risk(0.900000) = 0.000000; riskFraction = 0.000000
CPUScore = 9
Memory usage statistics:
No valid memory statistics.
overallScore = 9
[  info ] 2019/11/15 14:32:01 routes.go:81: safe-overload hostPriorityList = [{"Host":"master-node","Score":9},{"Host":"worker-node1","Score":10},
{"Host":"worker-node2","Score":9}]
```

Scheduling a test pod

pigeon extender

```
[  info ] 2019/11/15 14:32:01 routes.go:63: pigeon-holing ExtenderArgs =
===> StateUpdateNode: Updating node master-node with [(Capacity,Ovf,Usage)]=[ (2000,false,1000) (1396,false,140) ]
===> StateUpdateNode: Updating node worker-node1 with [(Capacity,Ovf,Usage)]=[ (2000,false,250) (1396,false,0) ]
===> StateUpdateNode: Updating node worker-node2 with [(Capacity,Ovf,Usage)]=[ (2000,false,250) (1396,false,0) ]
stateupdater: updateTime=0ms;
addedPEs:[master-node,worker-node1,worker-node2]
deletedPEs:[]
updatedPEs:[]
addedLEs:[]
deletedLEs:[]
updatedLEs:[]

CloudSystem:
PEs:
PE: name=master-node; idx=0; resCap=[2000,1396]; resUsed=[1000,140]; ovf=[0,0]; hostedLEs=[];
PE: name=worker-node1; idx=1; resCap=[2000,1396]; resUsed=[250,0]; ovf=[0,0]; hostedLEs=[];
PE: name=worker-node2; idx=2; resCap=[2000,1396]; resUsed=[250,0]; ovf=[0,0]; hostedLEs=[];

Evaluating objective for pod test-pod
===> demandCPU=200, demandMemory=209715200, demandPod=1, demandGPU=0, demandStorage=0
===> dcpu=200, dmem=200, dpod=1, dgpu=0, dstg=0
podRequest={0959d429-bac8-4692-8483-819bcc26268d [200 200]}
Node master-node:
+++++ Begin abpfn_getsystemvalue(): +++++
system matrix: [[0.600000, 0.125000, 0.125000][0.243553, 0.000000, 0.000000]]
system average: [0.283333, 0.081184]
system covariance: [[0.050139, 0.025708][0.025708, 0.013182]]
demand average: [200.000000, 200.000000]
demand relative average : [0.100000, 0.143266]
demand average2: [[40000.000000, 40000.000000][40000.000000, 40000.000000]]
demand relative average2: [[0.010000, 0.014327][0.014327, 0.020525]]
demand covariance: [[-0.000000, 0.000000][0.000000, 0.000000]]
covS=0.837634; covD=0.000244;
value=0.837389
+++++ End abpfn_getsystemvalue(): +++++
Node worker-node1:
+++++ Begin abpfn_getsystemvalue(): +++++
system matrix: [[0.500000, 0.225000, 0.125000][0.100287, 0.143266, 0.000000]]
system average: [0.283333, 0.081184]
system covariance: [[0.025139, 0.004457][0.004457, 0.003603]]
demand average: [200.000000, 200.000000]
demand relative average : [0.100000, 0.143266]
demand average2: [[40000.000000, 40000.000000][40000.000000, 40000.000000]]
demand relative average2: [[0.010000, 0.014327][0.014327, 0.020525]]
demand covariance: [[-0.000000, 0.000000][0.000000, 0.000000]]
covS=0.545667; covD=0.000244;
value=0.545423
+++++ End abpfn_getsystemvalue(): +++++
Node worker-node2:
+++++ Begin abpfn_getsystemvalue(): +++++
system matrix: [[0.500000, 0.125000, 0.225000][0.100287, 0.000000, 0.143266]]
system average: [0.283333, 0.081184]
system covariance: [[0.025139, 0.004457][0.004457, 0.003603]]
demand average: [200.000000, 200.000000]
demand relative average : [0.100000, 0.143266]
demand average2: [[40000.000000, 40000.000000][40000.000000, 40000.000000]]
demand relative average2: [[0.010000, 0.014327][0.014327, 0.020525]]
demand covariance: [[-0.000000, 0.000000][0.000000, 0.000000]]
covS=0.545667; covD=0.000244;
value=0.545423
+++++ End abpfn_getsystemvalue(): +++++
Node objective values = [ (master-node,0.837389) (worker-node1,0.545423) (worker-node2,0.545423)  ]
rankMap: map[master-node:1 worker-node1:10 worker-node2:10]
[  info ] 2019/11/15 14:32:01 routes.go:81: pigeon-holing hostPriorityList = [{"Host":"master-node","Score":1},{"Ho
st":"worker-node1","Score":10},{"Host":"worker-node2","Score":10}]
```

test pod scheduled

```
NAME       READY   STATUS    RESTARTS   AGE   IP                NODE
test-pod   1/1     Running   0          88s   192.168.180.225   worker-node1
```

Host selection based on

- standard predicates and priority functions
- safe predicate and priority function
- pigeon-holing priority function
- configured weighted sum of priority values
- all done within the scheduler extender pod