A Comparison of Scheduling on Open Source Cloud Platforms

Dr. Qiming Teng (滕启明)
tengqim@cn.ibm.com
IBM Research

Agenda

- How Clouds do Scheduling Today?
  - OpenStack Nova
  - Kubernetes
  - oVirt
  - Mesos
- Discussion
  - Goals
  - Solutions

OpenStack Nova

OpenStack Nova Scheduler

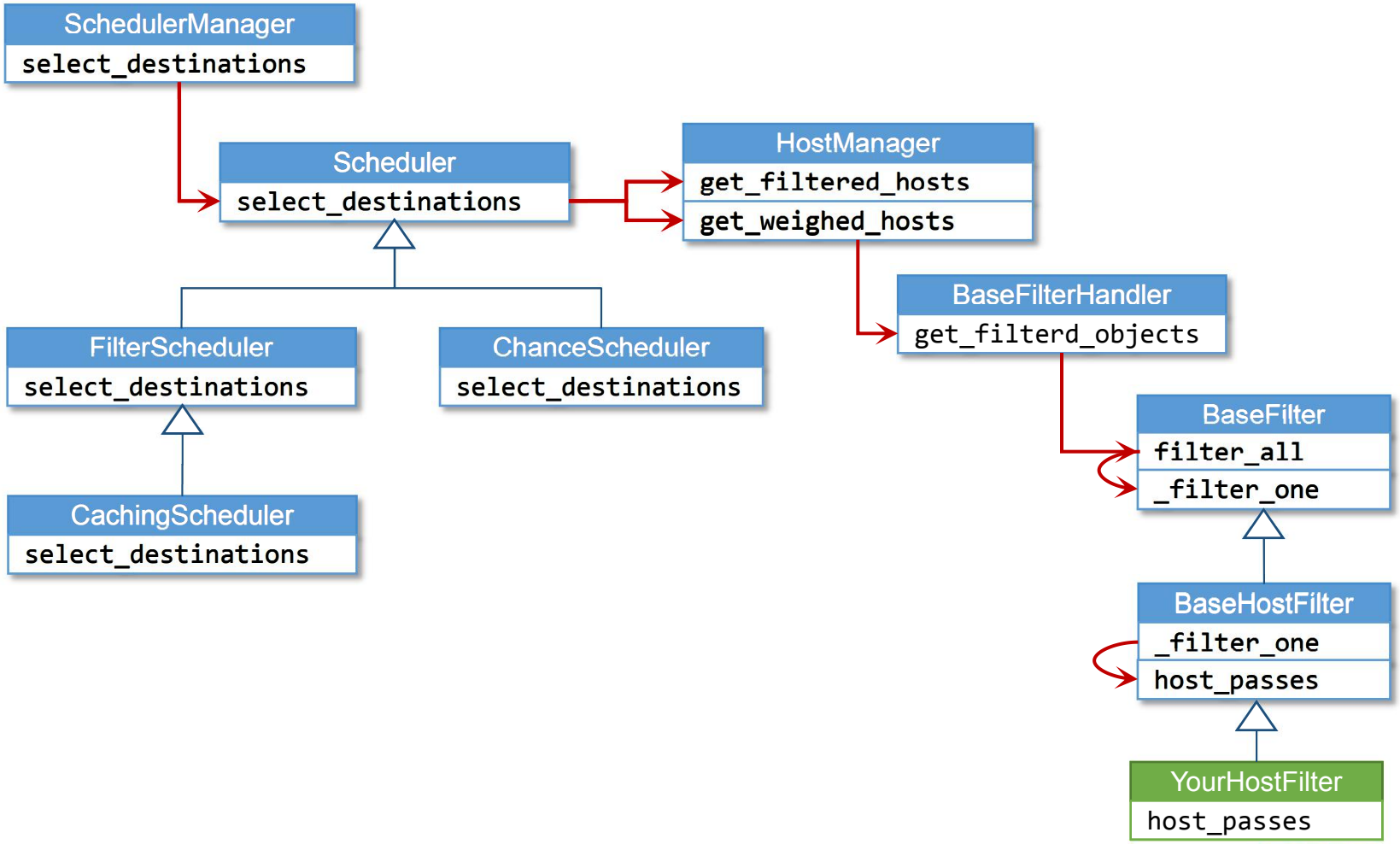
Category	Filter	Description	Type
Global	AvailabilityZoneFilter	AZ support	L
	ComputeFilter	Compute active	L
	ComputeCapabilitiesFilter	CPU topology, features, migration support etc.	L
	TrustedFilter	Trusted Computing Pools	L
	ImagePropertiesFilter	architecture, hypervisor, vm mode	L
	NumInstancesFilter, AggregateNumInstancesFilter	Instances count per host/aggregate	C
Resource	CoreFilter, AggregateCoreFilter, ExactCoreFilter	VCPU count	C
	RamFilter, AggregateRamFilter, ExactRamFilter	RAM size	C
	DiskFilter, AggregateDiskFilter, ExactDiskFilter	Disk capacity	C
	IoOpsFilter, AggregateIoOpsFilter	I/O bandwidth	C
	NUMATopologyFilter	NUMA topology (image) or extra spec (flavor)	L
	PciPassthroughFilter	PCI vendor and product ID	L
	MetricsFilter	Monitor metrics available?	C
Affinity	SameHostFilter, DifferentHostFilter		L
	SimpleCIDRAffinityFilter		L
	ServerGroupAffinityFilter, ServerGroupAntiAffinityFilter		L
	TypeAffinityFilter, AggregateTypeAffinityFilter	Instance type affinity	KV
Freeform	JsonFilter	more powerful combination, via ‘query’ scheduler hints	KV
Misc	RetryFilter	Skip hosts attempted	-

OpenStack Nova Scheduler

Weigher

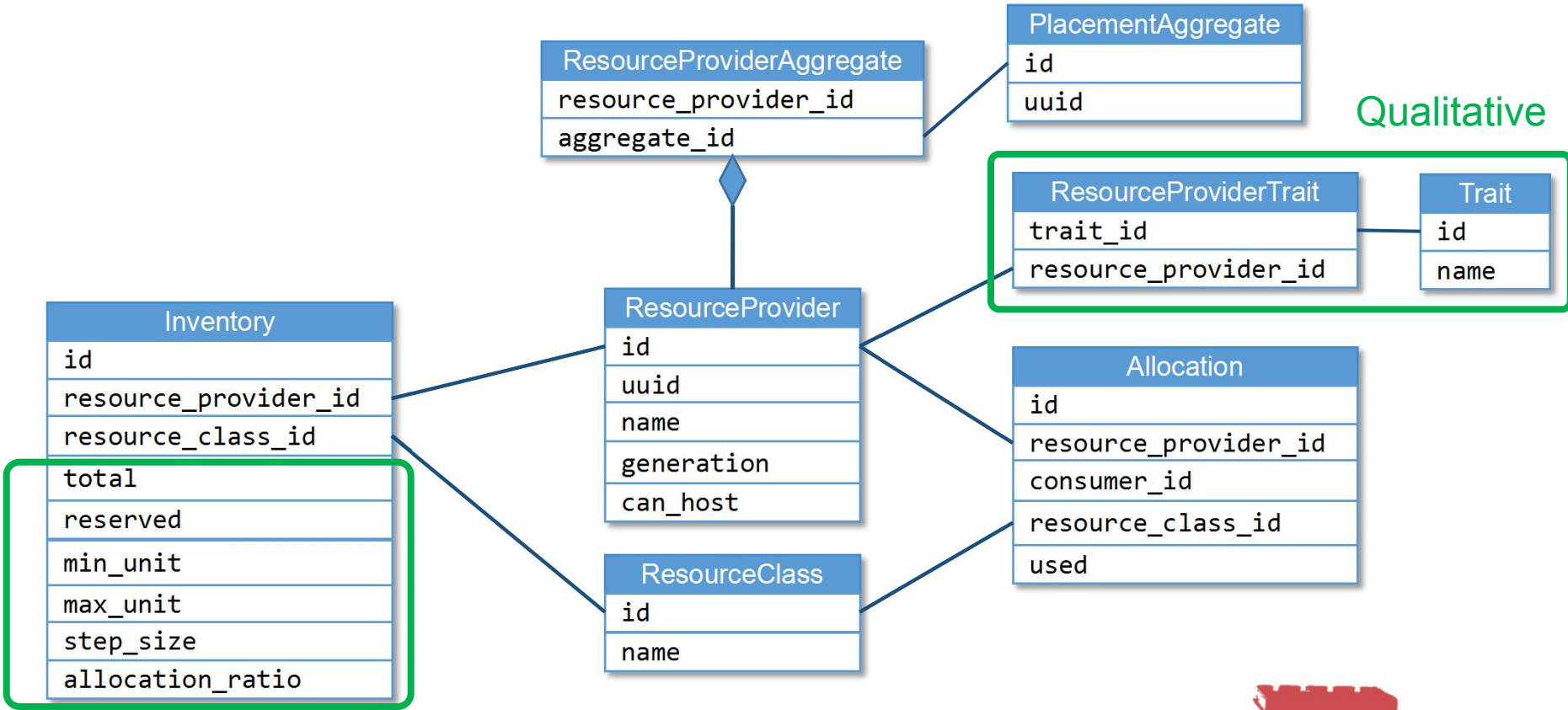
Category	Weigher	Description
Affinity	SeverGroupSoftAffinityWeigher	Based on instances on host
	ServerGroupSoftAntiAffinityWeigher	Based on instances on host
Resource	RamWeigher	Free RAM
	DiskWeigher	Free Disk
	IoOpsWeigher	Num I/O Ops
	MetricsWeigher	Metric (CPU) value

Nova Scheduler Extension



OpenStack Nova Placement

- A step towards more general/practical usage scenarios



Kubernetes

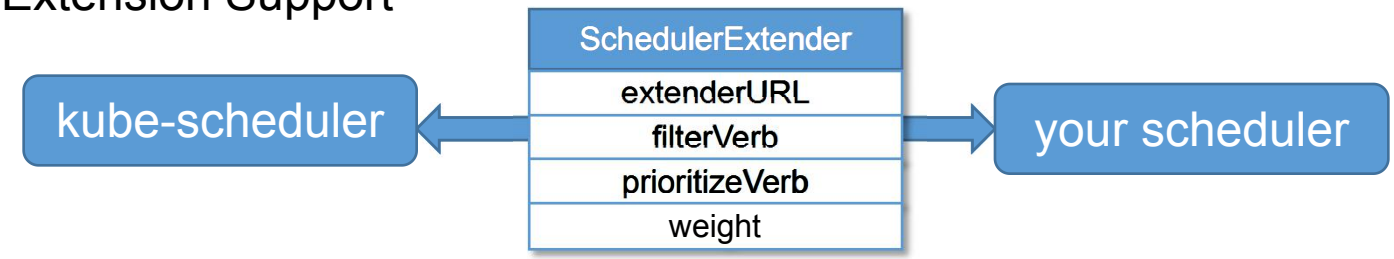
K8S Scheduler Predicates

Category	Filter	Description	Type
Label	PodTolerateNodeTaints	Tolerate tainted nodes	L
	MatchNodeSelector	• Label matches (Essential) • Node affinity checking	L
	NodeLabel	Builtin	L
	HostName	Host/node name (Essential)	L
	LabelPreference	Check lables	L
Resource	PodFistsResources	• Allowed pod number • Resources: CPU, Memory, NvidiaGPU, OpaqueIntResources	C
	CheckNodeMemoryPressure	Memory, BestEffort	C
	CheckNodeDiskPressure	Disk	C
	NoDiskConflict	Non-conflicting disk volumes (GCE, EBS and RBD specific)	L
	NoVolumeZoneConflict	Volume zone	L
	MaxEBSVolumeCount	Max volume count	C
	MaxGCEPDVolumeCount	Max volume count	KV
	MaxAzureDockVolumeCount	Max volume count	C
	GeneralPredicates	• Non-critical: PodFitsResource • Essential	L
	PodFitsHostPorts	Port available (Essential)	KV
Affinity	MatchInterPodAffinity	(Anti-)Affinity among pods to deploy and against existing pods	KV
	ServiceAffinity, ServiceAntiAffinity	Pods from same service scheduled to same node	

K8S Scheduler Priorities

Category	Priority Function	Default	Description
Spread	SelectorSpreadPriority	Y	Minimize homogeneous #pods on the same node
	ServiceSpreadingPriority	N	Favor nodes with fewer matching nodes
Affinity	InterPodAffinityPriority	Y	(Anti-)Affinity, by weighting affinity scores
	NodeAffinityPriority	Y	Prefer NodeAffinity label matches, counter matched terms
Label	NodeLabelPriority	Y	Prefer nodes having specified labels
	NodePreferAvoidPodsPriority	Y	Respect node 'preferAvoidPods' annotation
	TaintTolerationPriority	Y	Check number of intolerable taints on nodes
Resource	BalancedResourceAllocation	Y	Try balance resource usage, must use with LeastRequestedPriority
	LeastRequestedPriority, MostRequestedPriority	Y	Prefer nodes least/most utilized (CPU + MEM)/2
	ImageLocalityPriority	N	Prefer nodes with more local pkgs
Misc	EqualPriority	N	The default

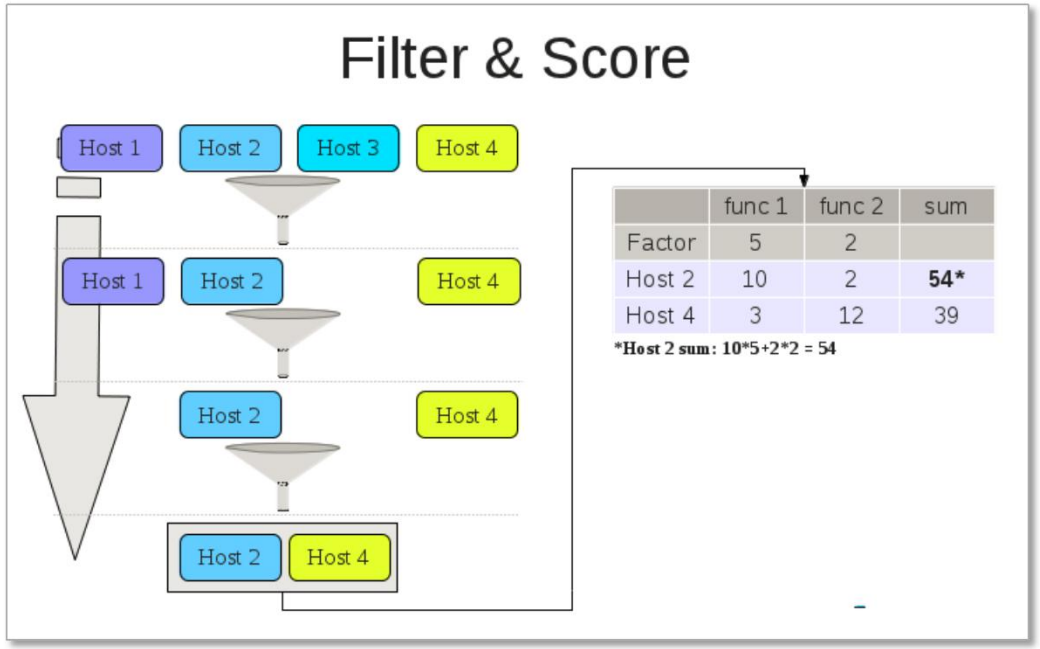
Extension Support



oVirt

oVirt Scheduler

- Filter + Score
- Load-Balancing
  - policy + migration
- Extensibility
  - Java
  - Python



source: https://www.ovirt.org/develop/release-management/features/sla/ovirtscheduler/

oVirt Filters/Weighers

Category	Filter	Status	Description	Type
Spread	PinToHost	OK	Subset of hosts a VM can pin to	KV
	Migration	OK	Filter current VM ⇔ encourage migration	KV
Resource	CPU	OK	CPU cores	C
	Memory	N/A	Filter by memory usage	-
	CPU-Level	OK	CPU level, manufacture, model etc	KV
	Network	OK	Filter by necessary network interfaces	KV
Affinity	VMAffinityGroups	OK	Same or different hosts	KV
HA	HA	OK	Filter by HA scores	KV
Misc	Emulated-Machine	OK	Filter emulators	L
	None	OK	Filter nothing	-

Category	Weight Policy	Status	Description
Spread	OptimalForEvenGuestDistribution	OK	Balanced distribution wrt #VM
	OptimalForEvenDistribution	OK	Balanced distribution wrt CPU utilization
Affinity	VMAffnityGroups	OK	Check VM from the same group
HA	OptimalForHaReservation	OK	Number of HA VMs
	HA	OK	Check number of HA VMs
POWER	OptimalForPowerSaving	OK	Minimize number of hosts needed, considering CPU utilization
Misc	None	Gone	Use even distribution policy instead

Mesos Family

Scheduling on Mesos

- Mesos doesn't do scheduling, it only manages resources
  - Mesos allocates resources to frameworks
  - Each frameworks decides how to "schedule"
- Mesos Agent ⇔ OpenStack Nova ResourceProvider
  - Resource ::= key-value pairs
  - For an agent:
    - `--resources='cpus:24;gpus:2;mem:24576;disk:409600;...'`
    - `or via resources.txt`

```
[
  {
    "name": "cpus",
    "type": "SCALAR",
    "scalar": {
      "value": 24
    }
  },
  {
    "name": "gpus",
    "type": "SCALAR",
    "scalar": {
      "value": 2
    }
  },
```

```
{
   "name": "mem",
   "type": "SCALAR",
   "scalar": {
     "value": 24576
   }
 },
 {
   "name": "disk",
   "type": "SCALAR",
   "scalar": {
     "value": 409600
   }
 },
```

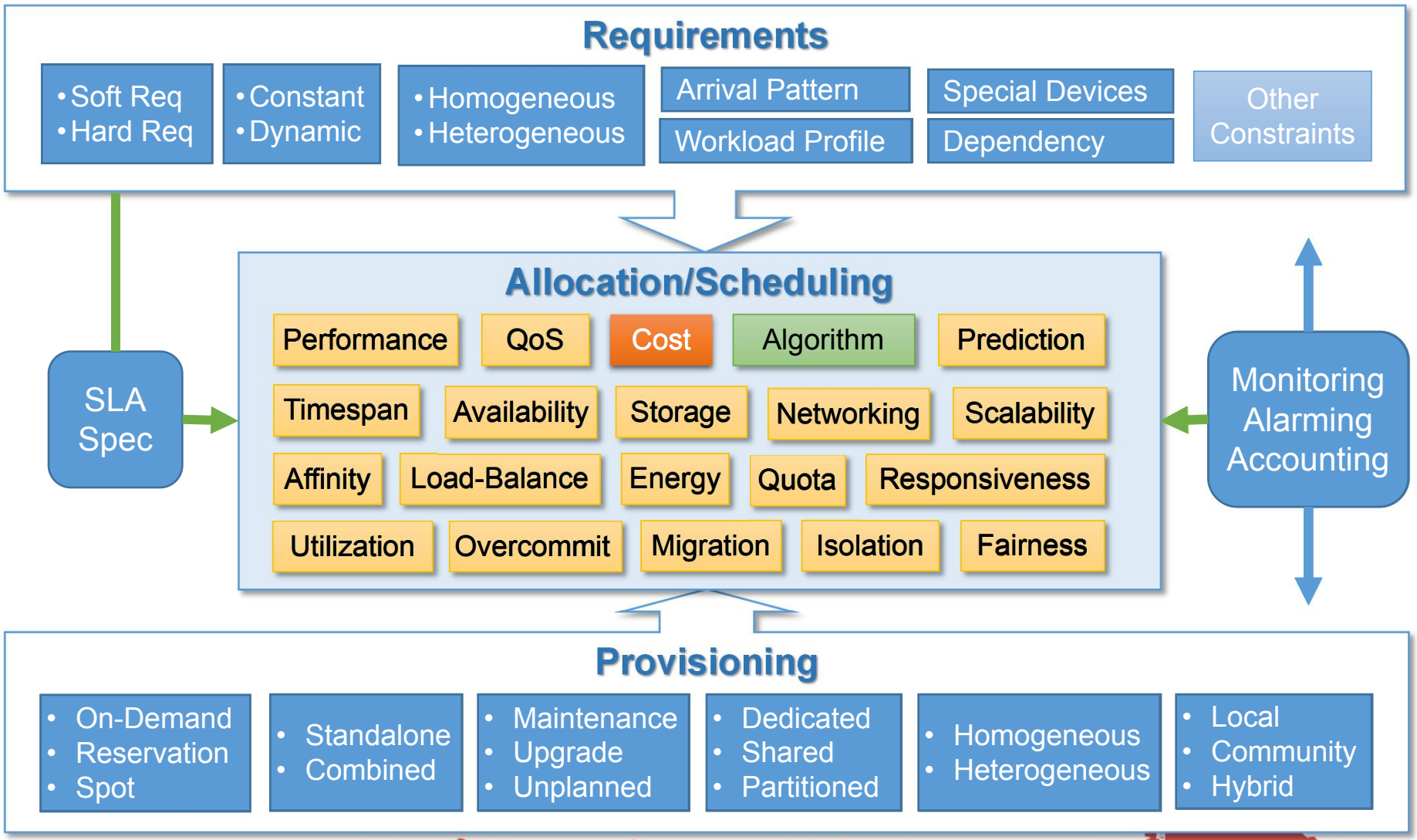
```
  {
    "name": "ports",
    "type": "RANGES",
    "ranges": {
      "range": [
        {
          "begin": 21000,
          "end": 24000
        },
        {
          "begin": 30000,
          "end": 34000
        }
      ]
    }
  },
]
```

Scheduling on Mesos

- Mesos Schedulers:
  - Marathon: long running services
  - Chronos: batch, periodic tasks
  - both respect resource specs (offers)
- Marathon Constraints: <field, operator, parameter>
  - anti-affinity:
    - `"constraints": [["hostname": "UNIQUE"]]`
  - host-bind:
    - `"constraints": [["rack_id", "CLUSTER", "rack-1"]]`
  - high-availability:
    - `"constraints": [["rack_id", "GROUP_BY"]]`
  - filter-host:
    - `"constraints": [["rack_id", "LIKE", "rack-[1-3]"]]`

Discussion

The Scheduling Problem



Other Factors/Technologies

- 1600+ papers, 90+ algorithms, during the past 10 years
- From Economy: Profit Driven
  - Supply and Demand
  - Marketing, Bargaining, Auction, Pricing
- From Mathematical Programming
  - Multi-goal, multi-constraint optimization
  - Heuristics, Meta-heuristics
  - Queueing: M/G/1
  - Ant Colony
  - Genetic algorithms
  - Bag of tasks, Group scheduling
  - …
- Extensibility, Complexity, …

Summary

- there are many academic research work on cloud scheduling
  - most of which are not landed in real world deployments
- a well-designed scheduling solution can give you a competitive edge
- there is no solution ready for your environment
  - and there will never be one if you don't know what you need
- we give and we take