IBM

Open by design™

IBM开源技术微讲堂

Kubernetes系列

第三讲

Kubernetes中的资源调度与资源管理

更多信息，请访问：http://ibm.biz/opentech-ma

“Kubenetes”系列公开课

每周四晚8点档

1. Kubernetes 初探
2. 上手 Kubernetes
3. **Kubernetes 的资源调度**
4. Kubernetes 的运行时：Kubelet
5. Kubernetes 的网络管理
6. Kubernetes 的存储管理
7. Kubernetes 的日志与监控
8. Kubernetes 的应用部署
9. 扩展 Kubernetes 生态
10. Kubernetes 的企业实践

课程Wiki：http://ibm.biz/opentech-ma

日期	主题	视频回放	讲义
10月19日	Kubernetes初探	Kubernetes初探	Kubernetes第一讲.pdf
10月26日	上手Kubernetes：基本概念、安装和命令行工具kubctl	Kubernetes上手	Kubernetes第二讲.pdf
11月2日	**Kubernetes的资源调度**		
11月9日	Kubernetes的运行时：Kubelet		

讲师介绍—马达

IBM 软件架构师,

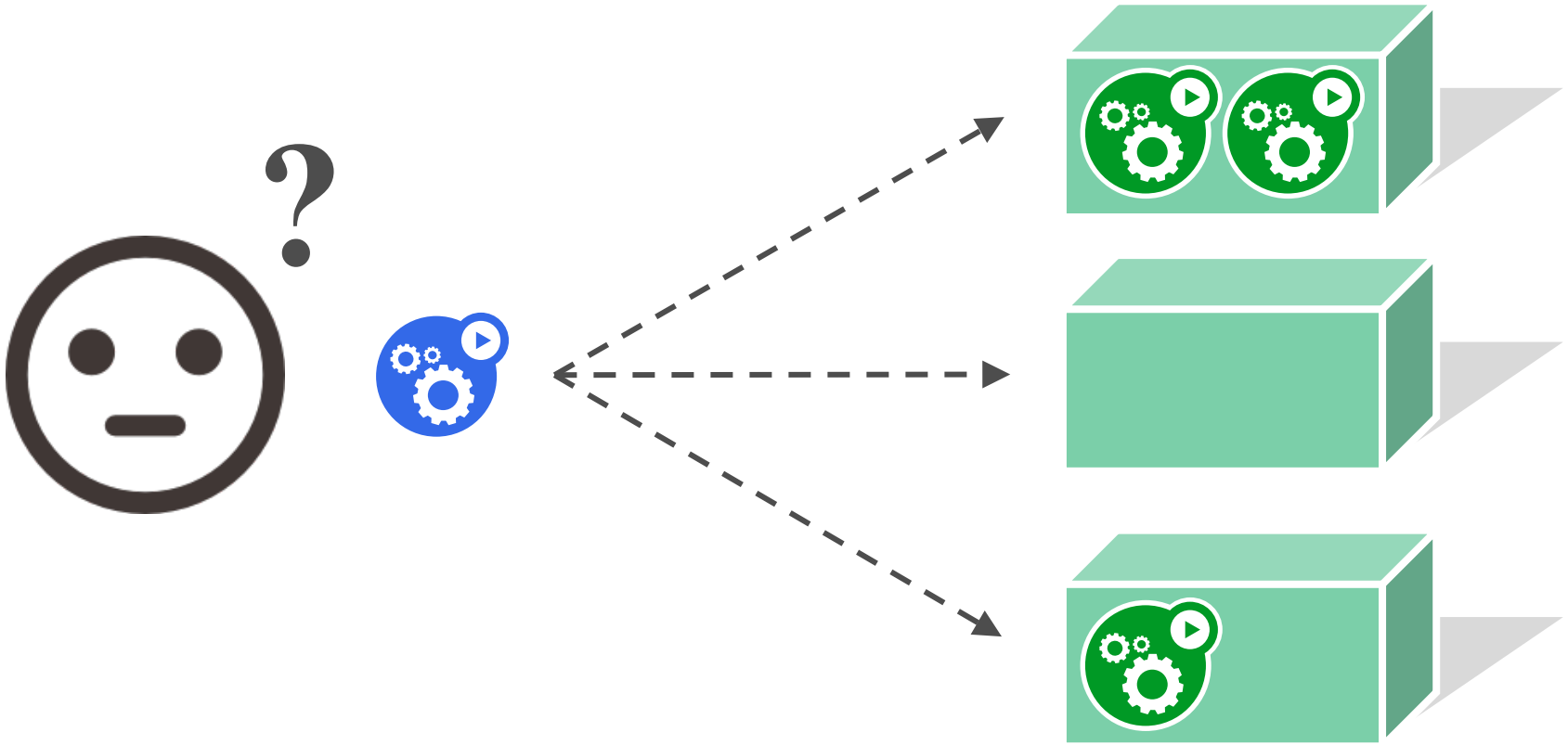
Kubernetes Maintainer,

Mesos Contributior。

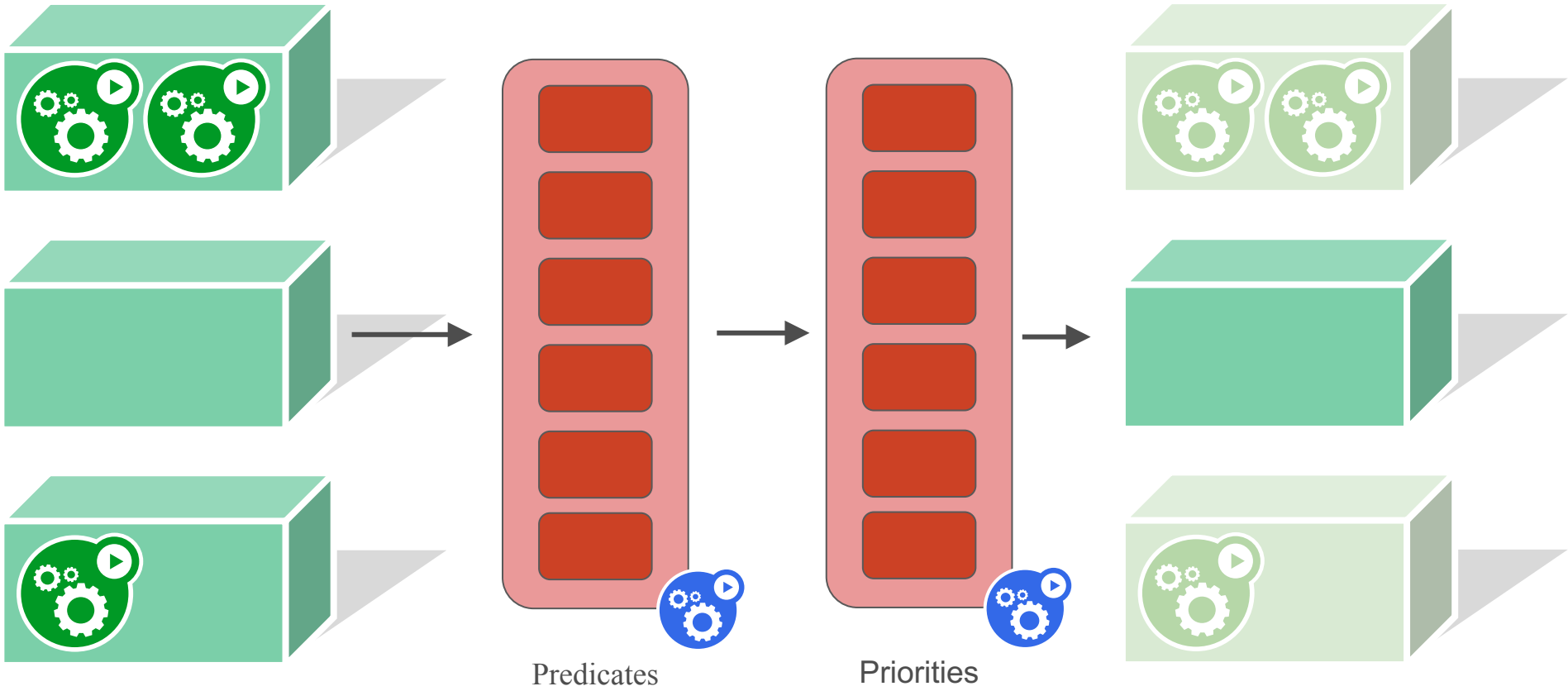
吉林大学硕士， 主修分布式系统, 网格计算。

现任职于IBM系统中心， 在资源管理, 资源调度及分布式计算方面有10多年的经验。

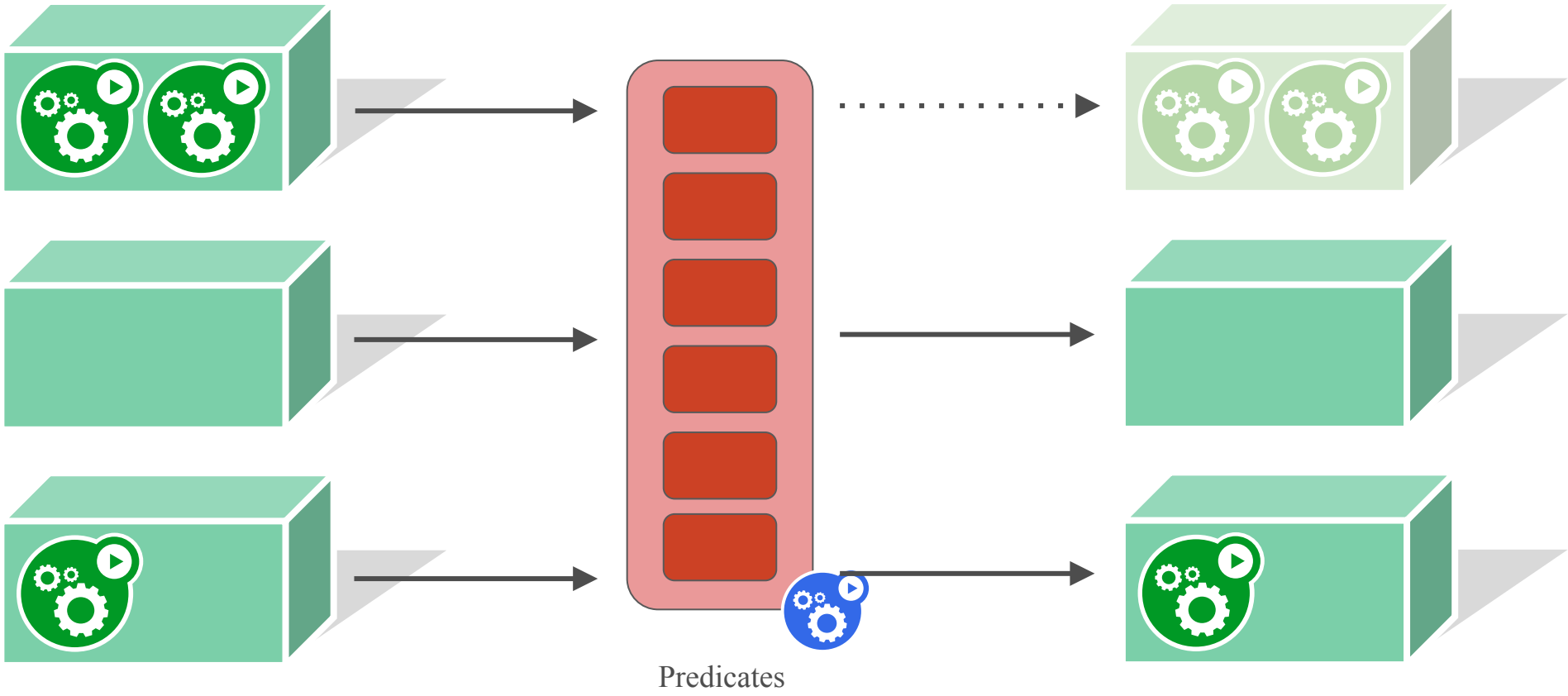
Scheduling



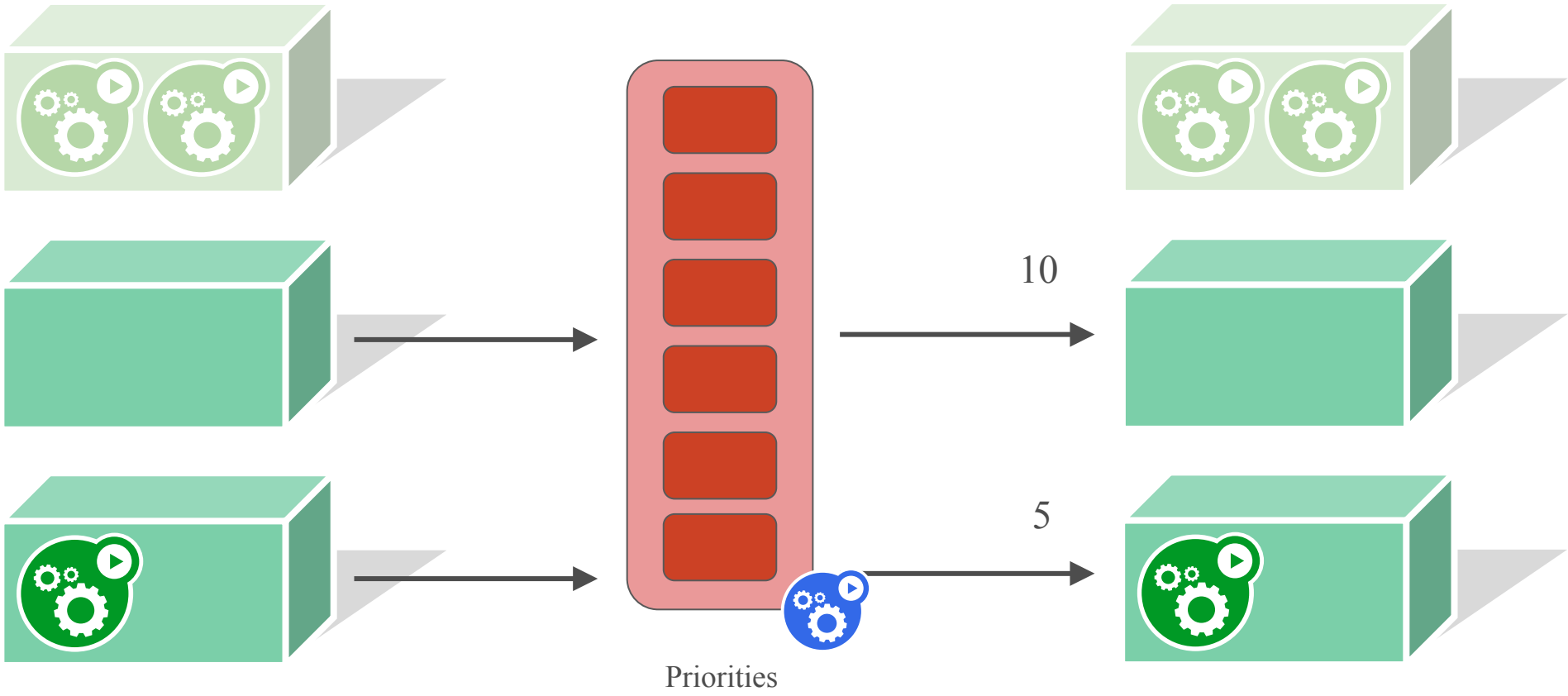
Scheduling



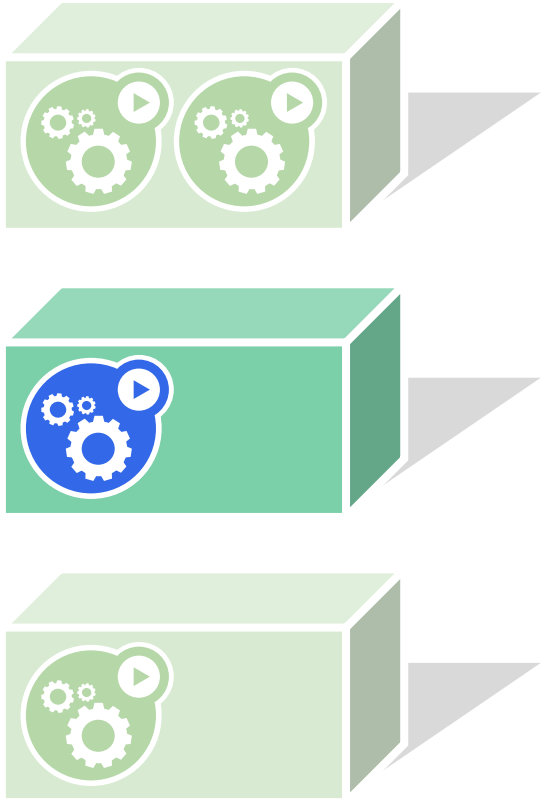
Scheduling



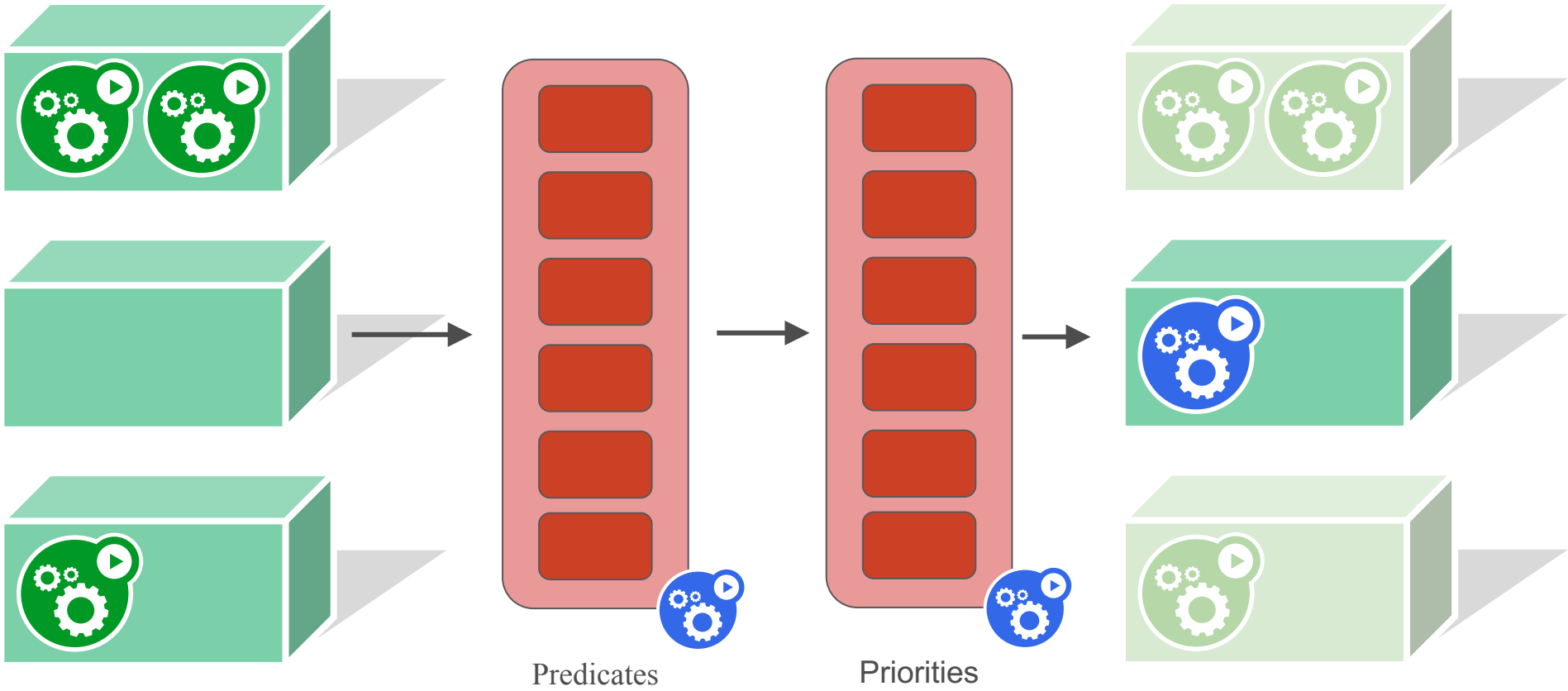
Scheduling



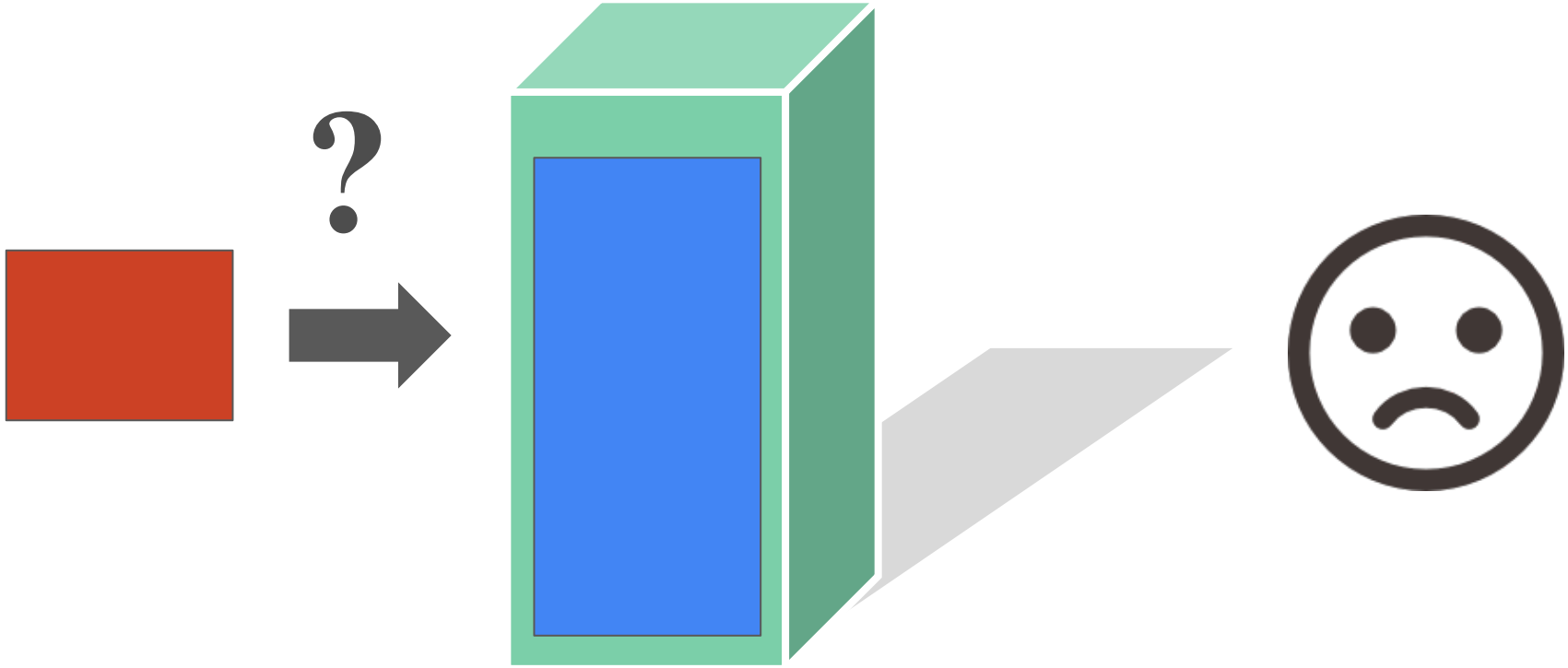
Scheduling



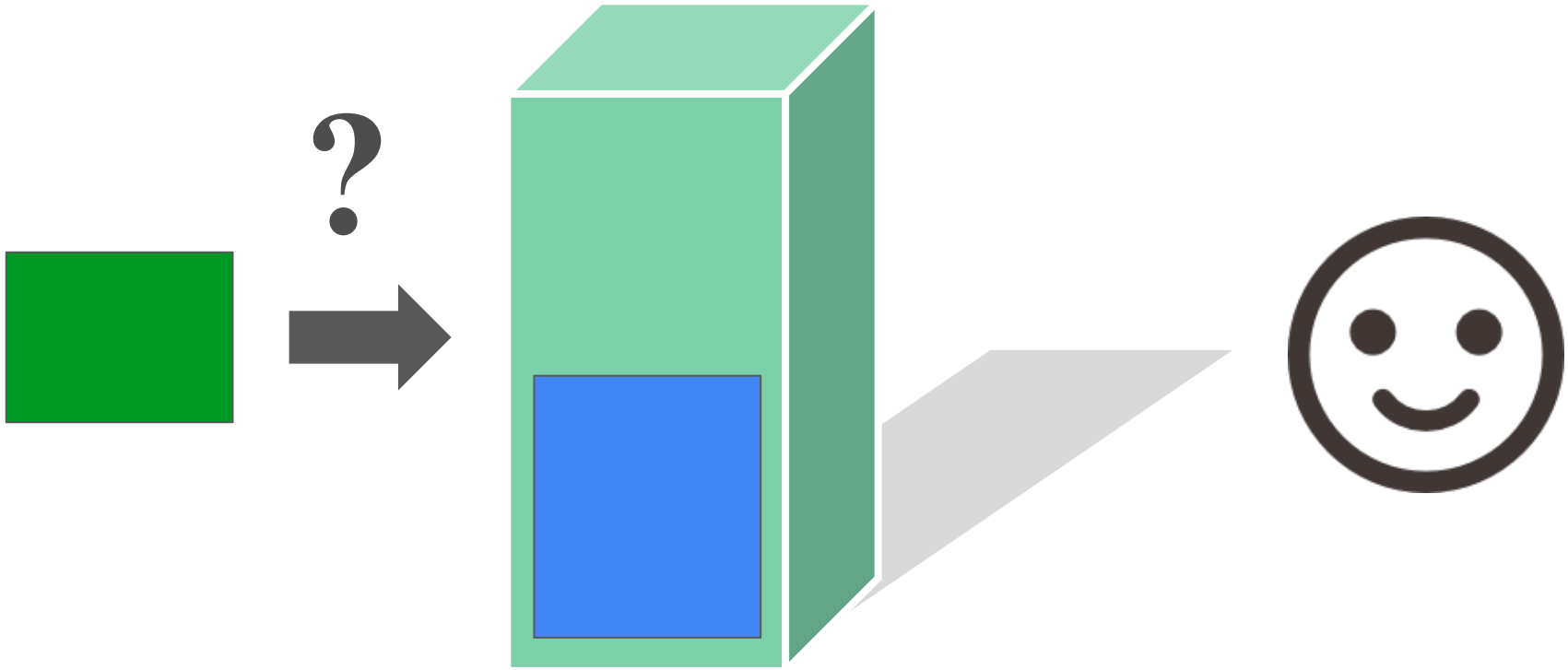
Scheduling



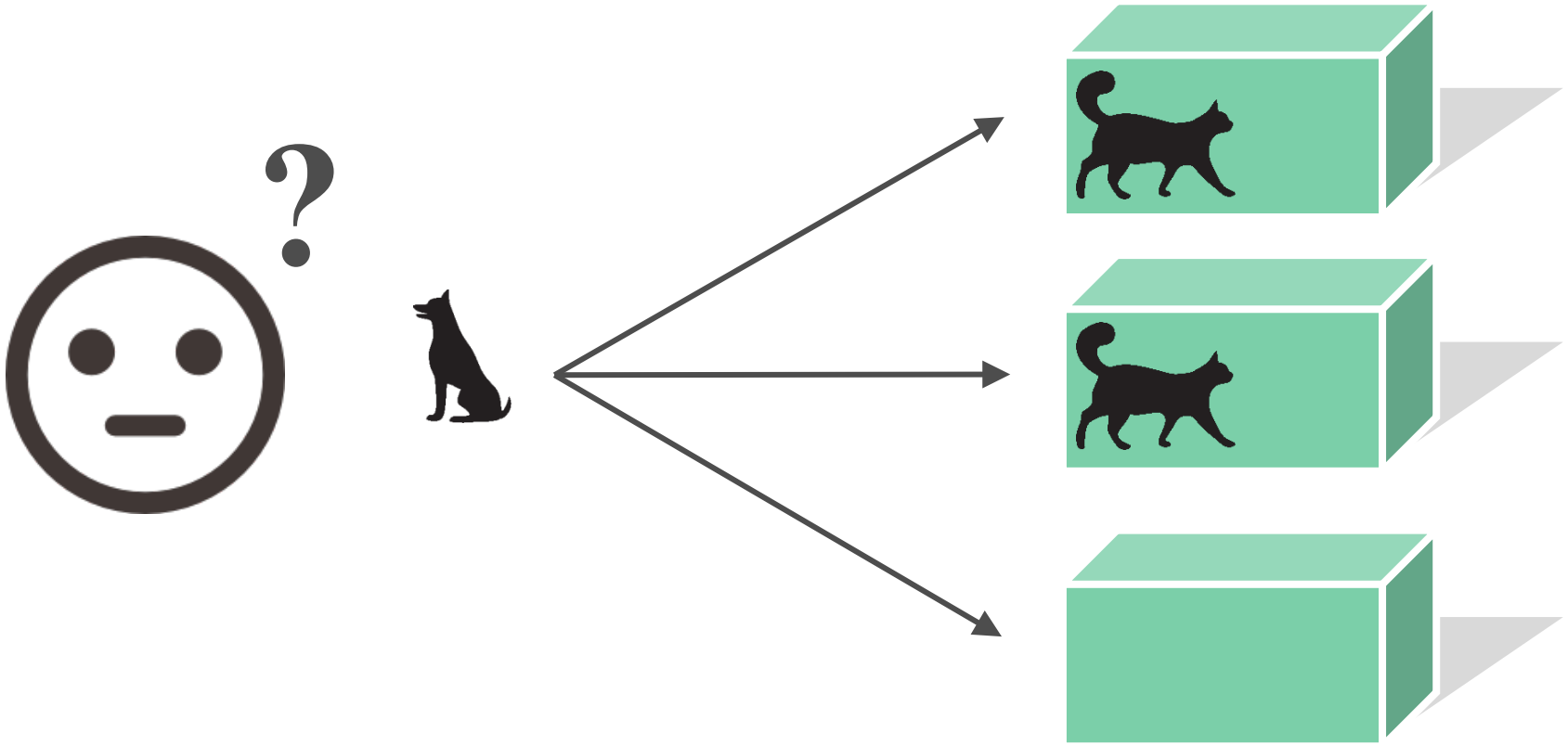
Prevent overcommit



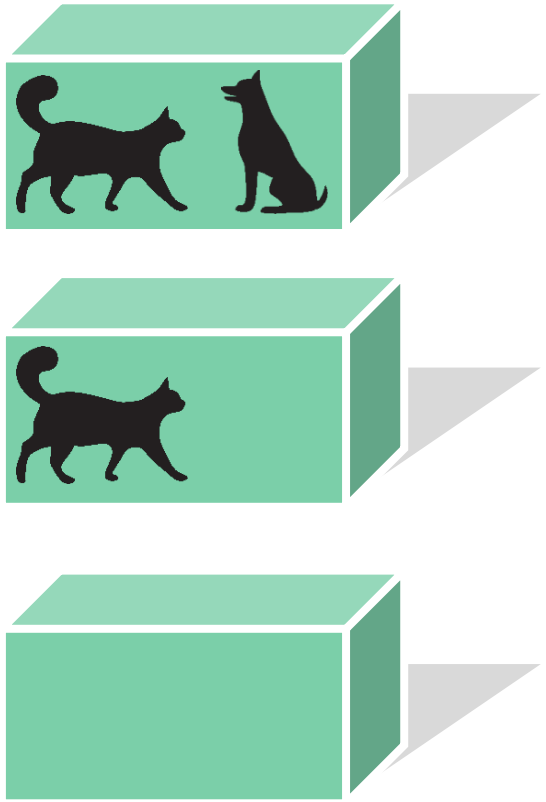
Prevent overcommit



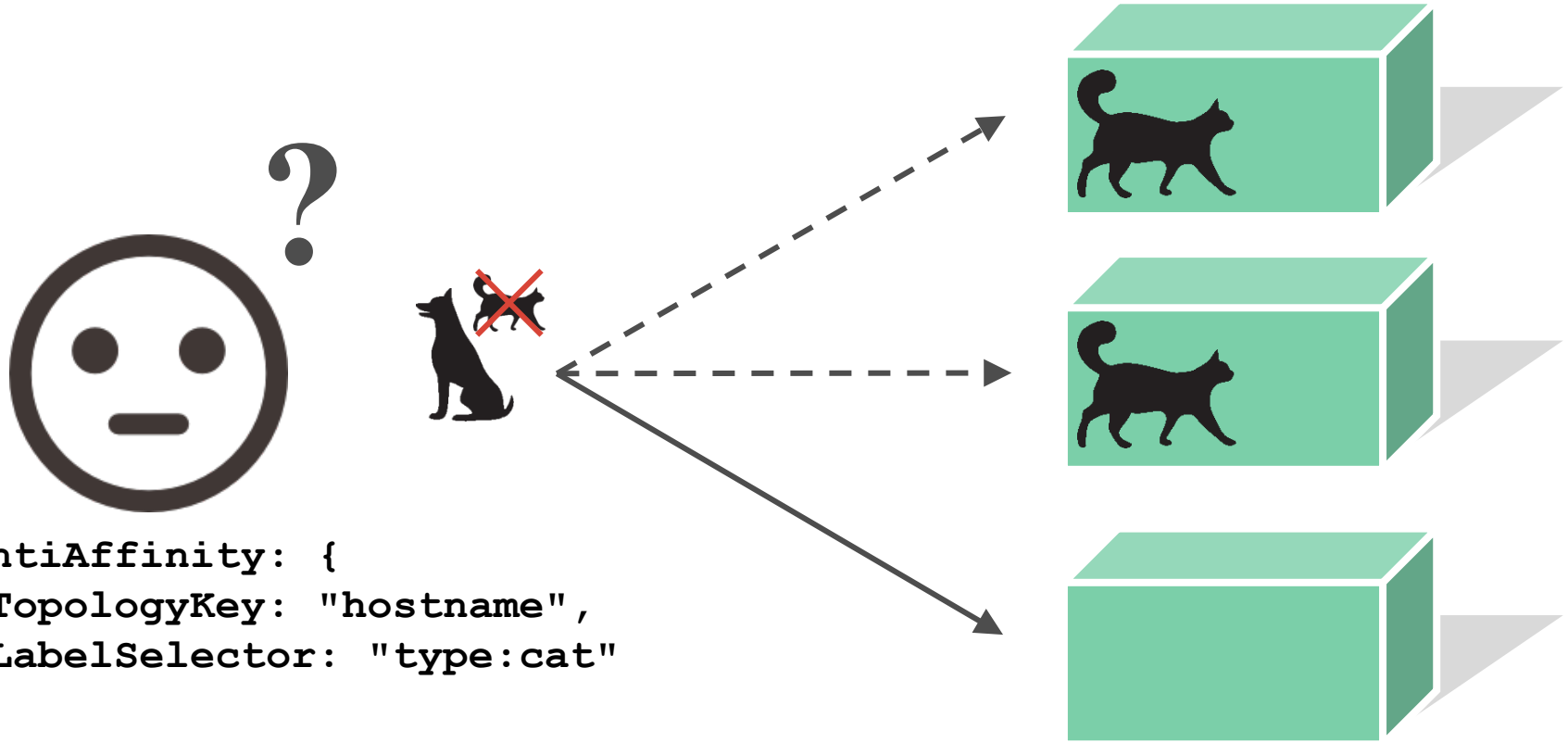
Prevent co-scheduling (pod anti-affinity)



Prevent co-scheduling (pod anti-affinity)

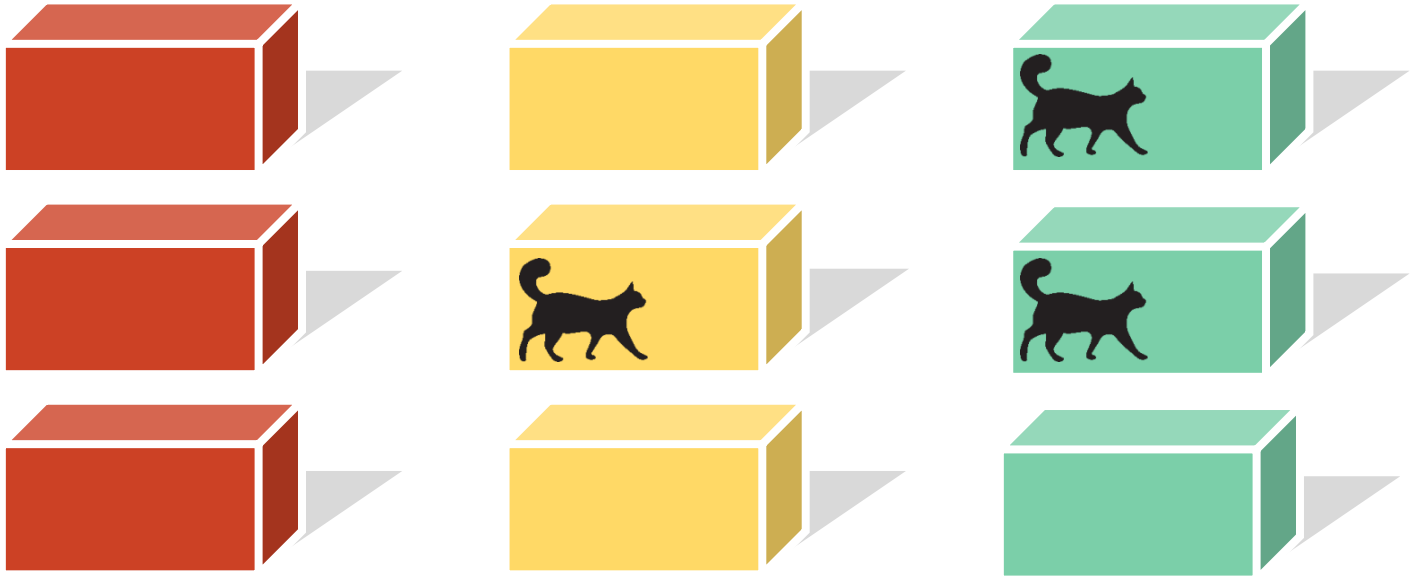


Prevent co-scheduling (pod anti-affinity)

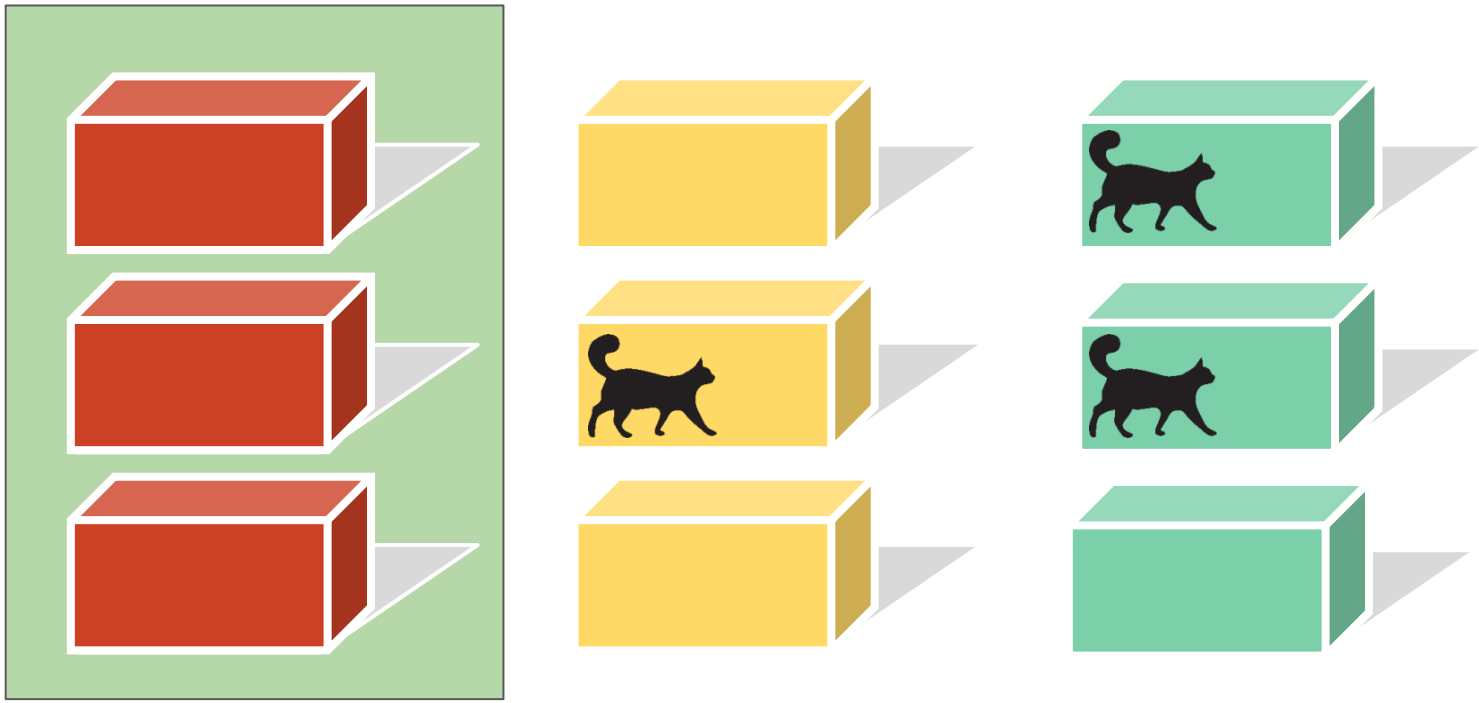


```
PodAntiAffinity: {
    TopologyKey: "hostname",
    LabelSelector: "type:cat"
}
```

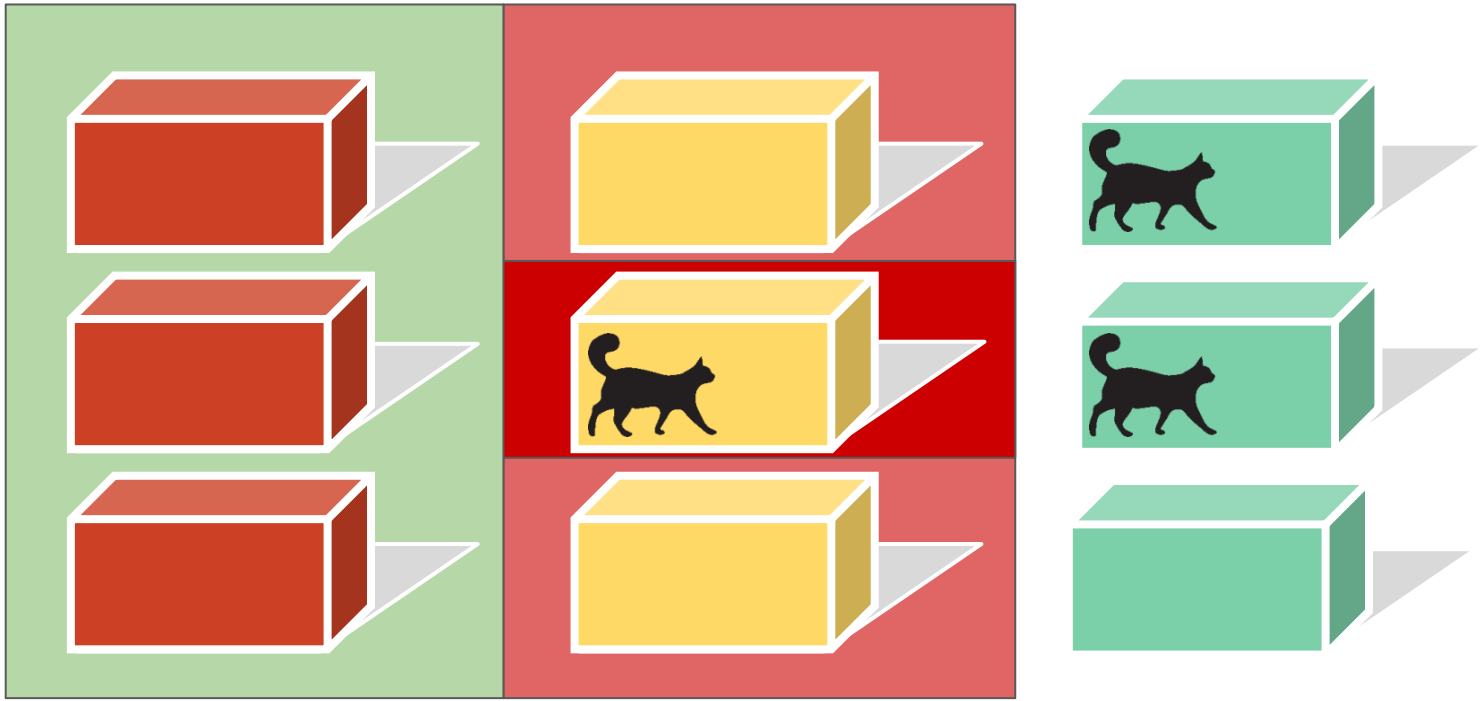
Prevent co-scheduling (pod anti-affinity)



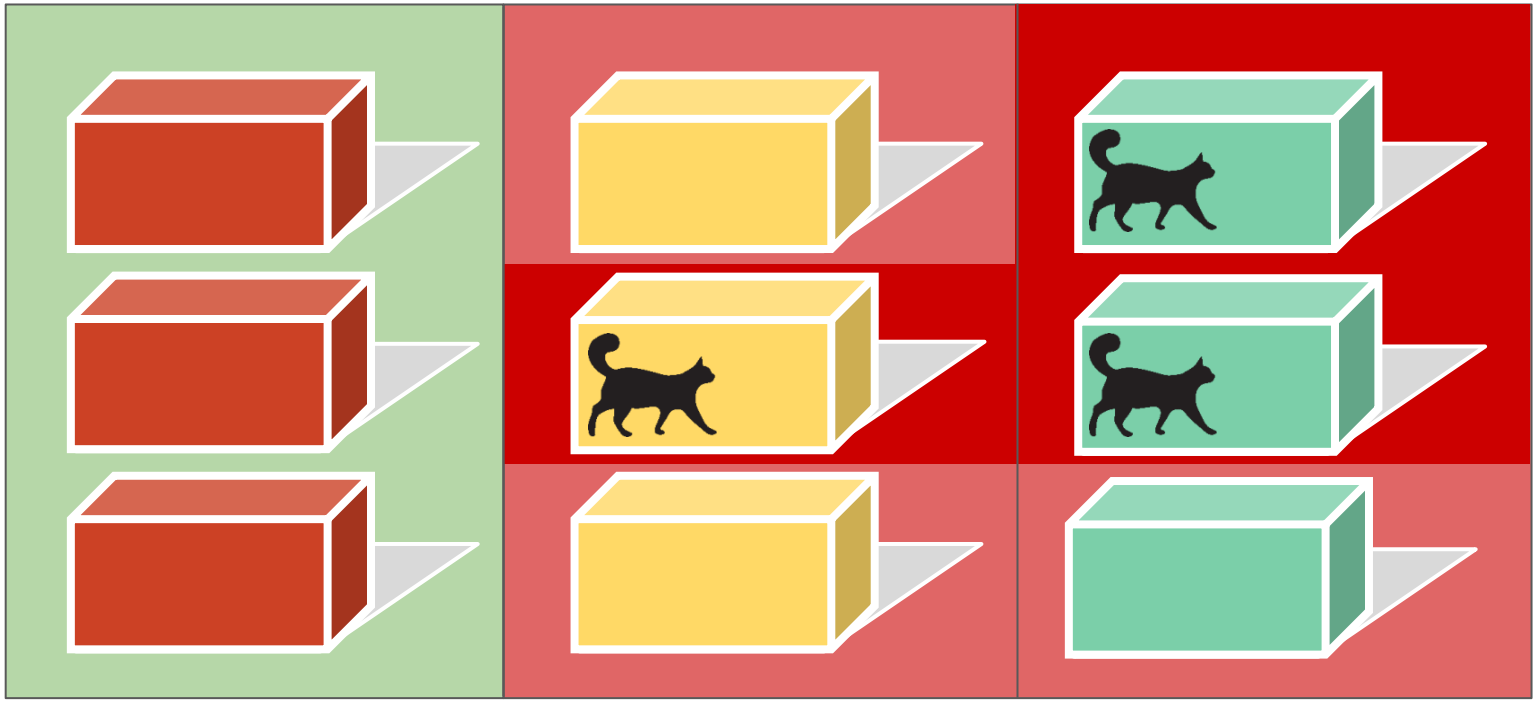
Prevent co-scheduling (pod anti-affinity)



Prevent co-scheduling (pod anti-affinity)

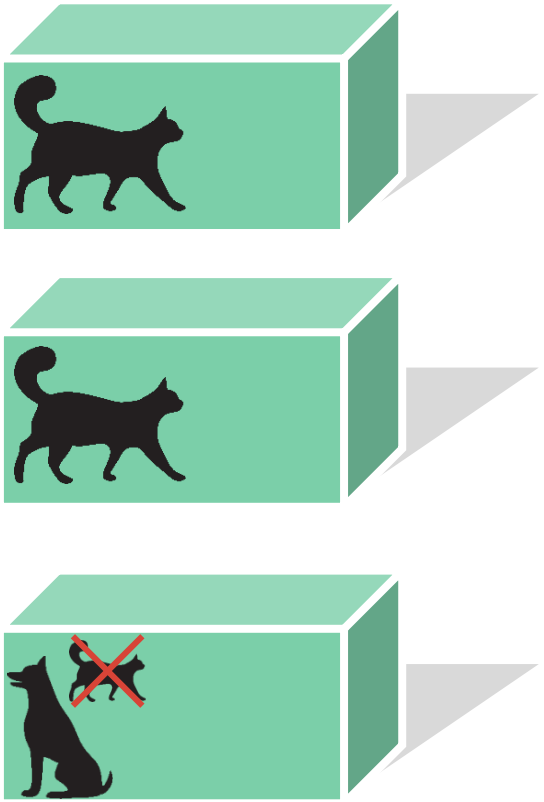


Prevent co-scheduling (pod anti-affinity)

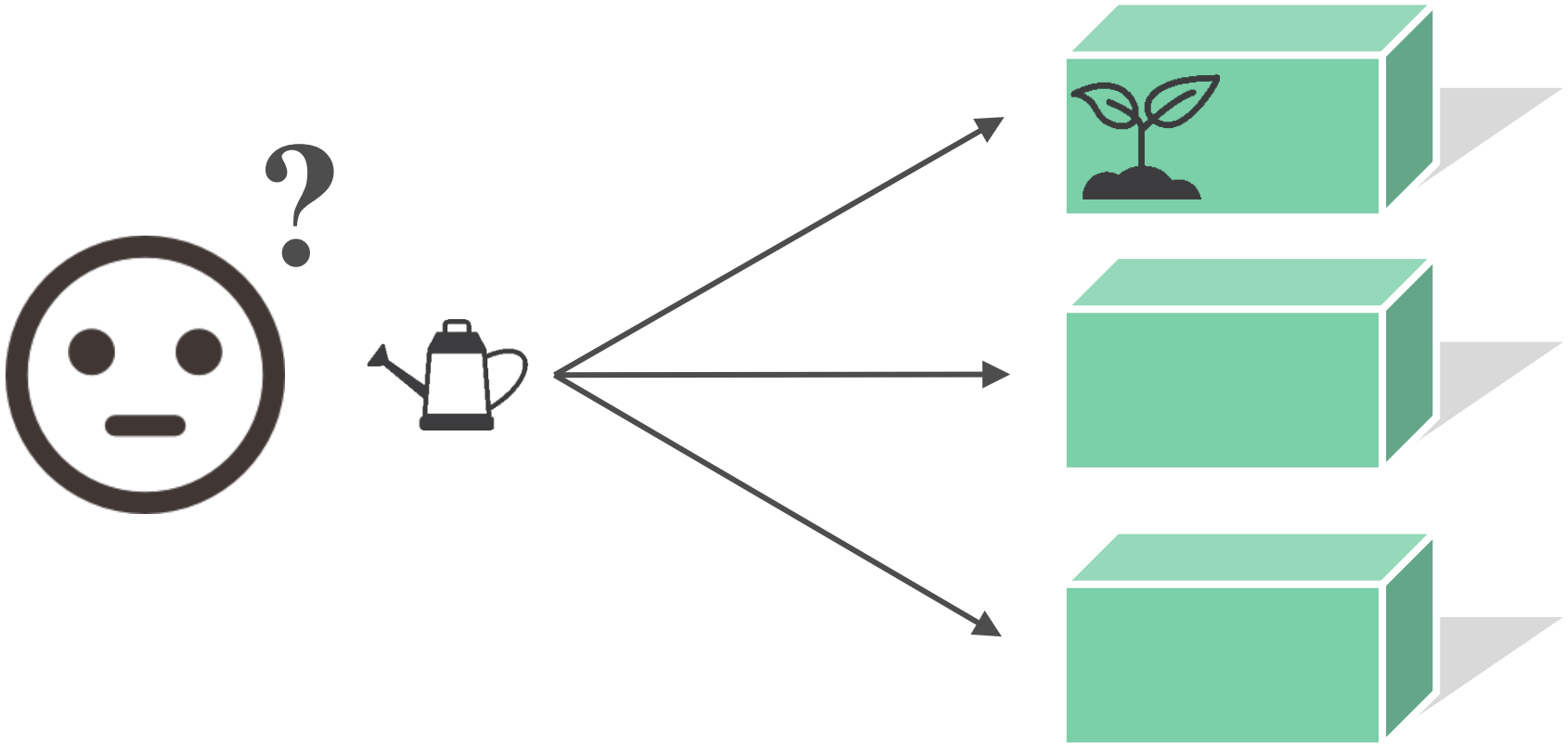


Prevent co-scheduling (pod anti-affinity)

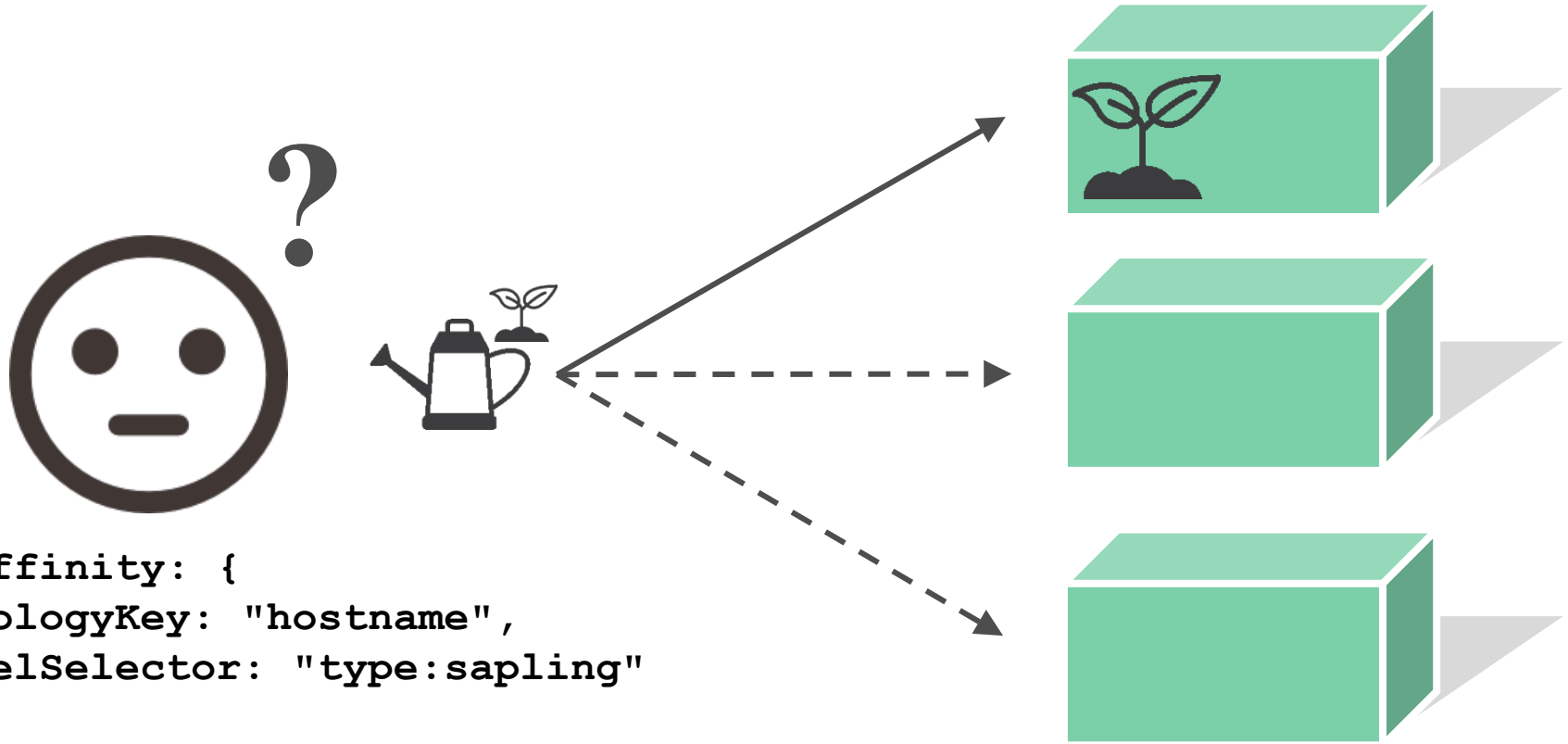
```
PodAntiAffinity: {
    TopologyKey: "hostname",
    LabelSelector: "type:cat"
}
```



Force co-scheduling (pod affinity)

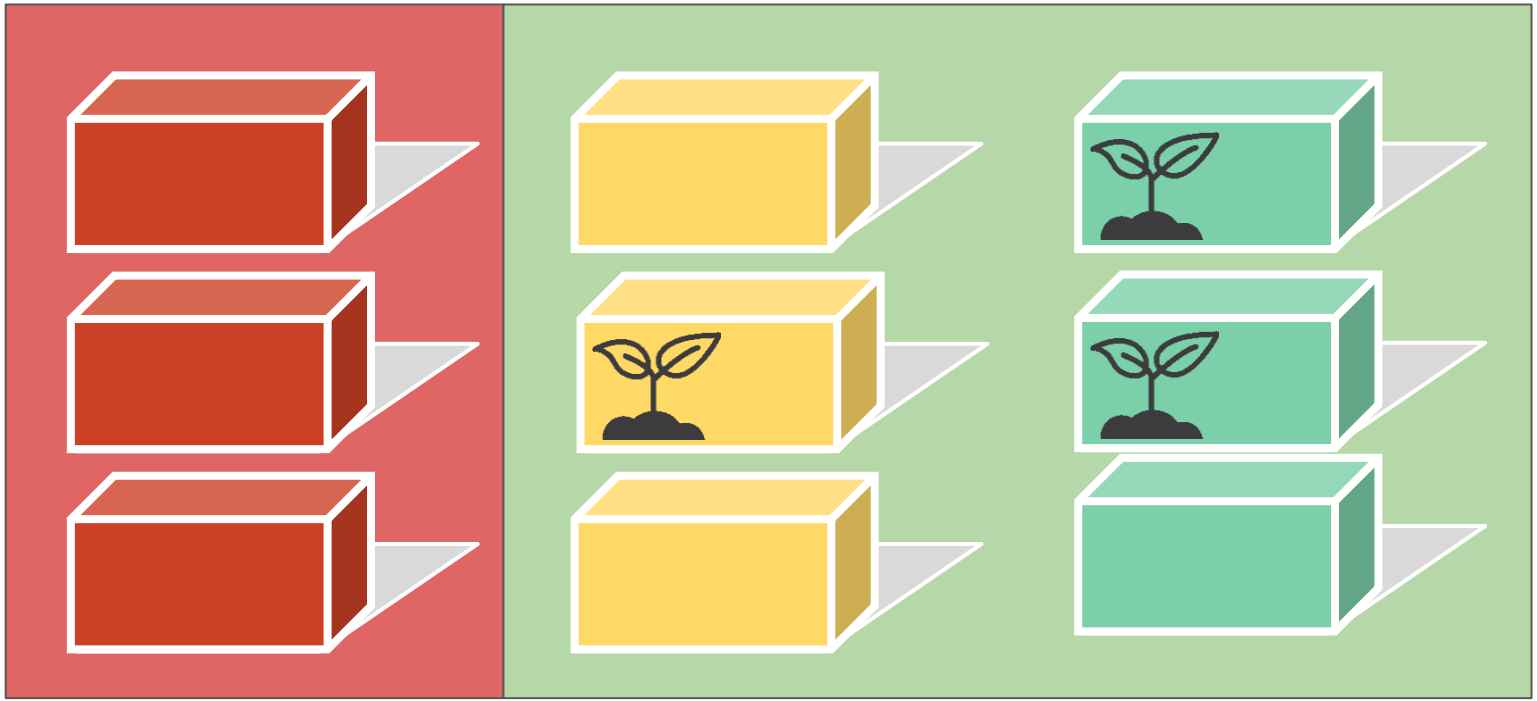


Force co-scheduling (pod affinity)

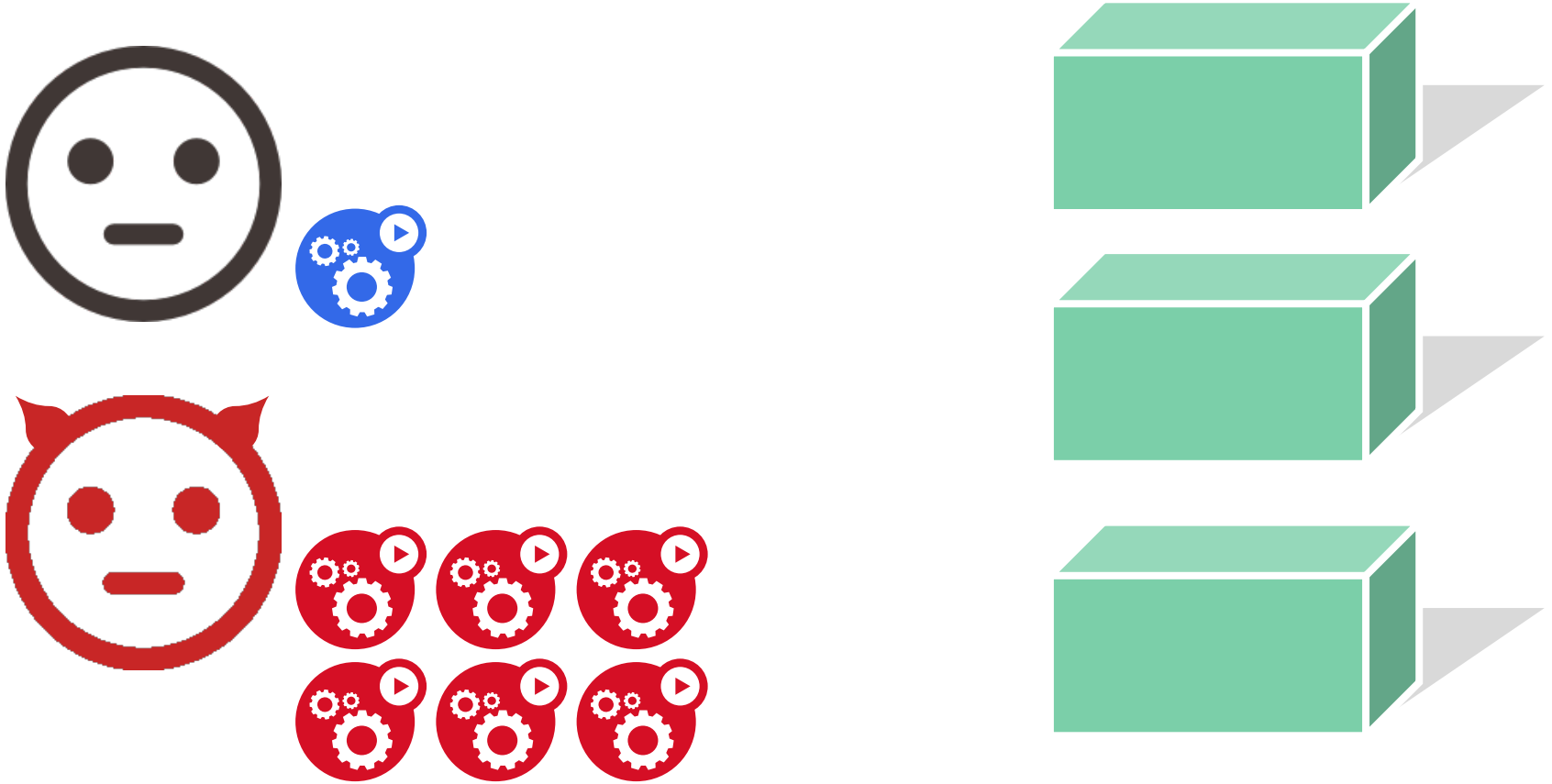


```
PodAffinity: {
 TopologyKey: "hostname",
 LabelSelector: "type:sapling"
}
```

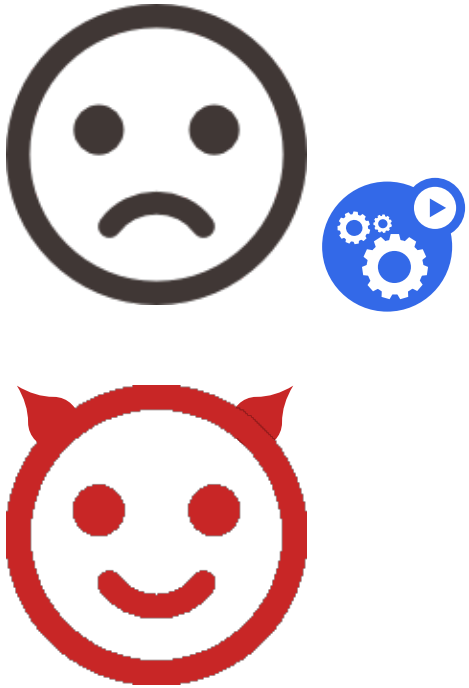
Force co-scheduling (pod affinity)

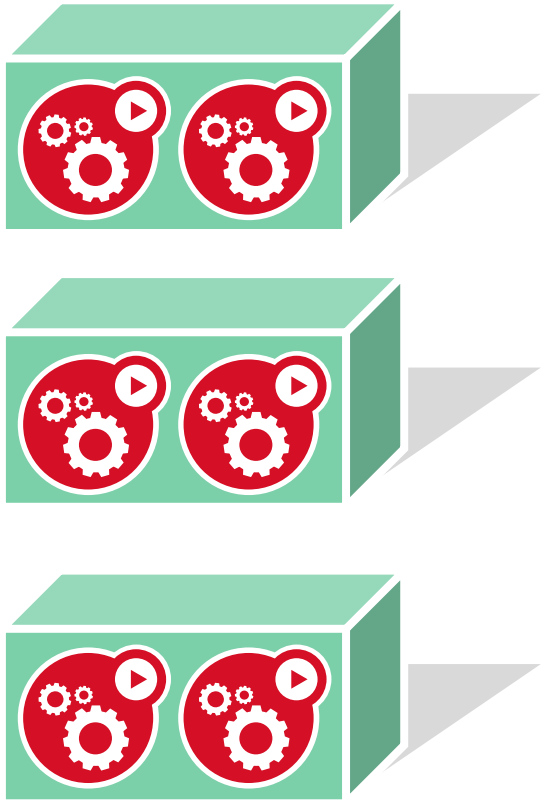


Dedicated machines (Taints)



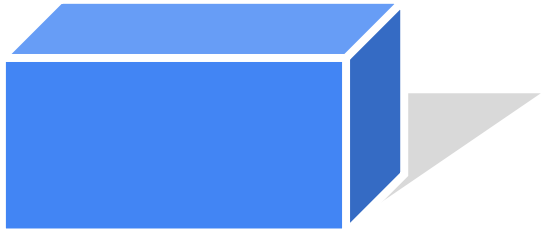
Dedicated machines (Taints)





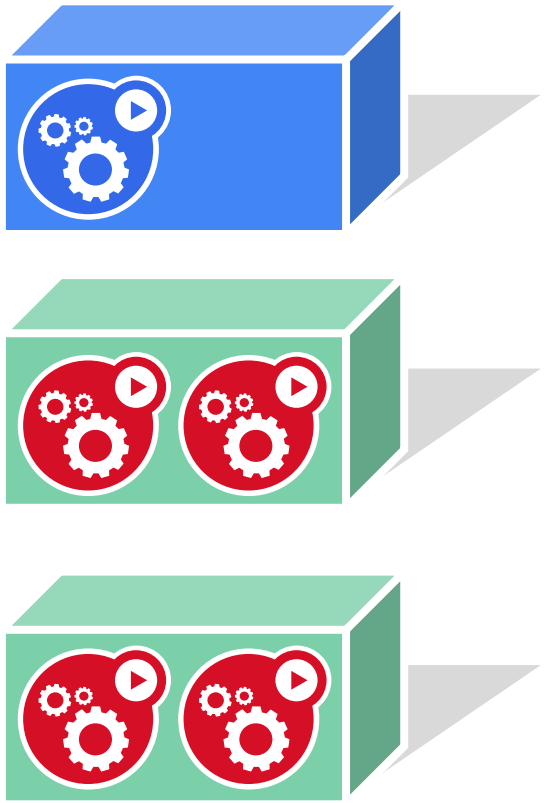
Dedicated machines (Taints)

```
Taint: {
 TaintEffect: "NoSchedule",
 Key: "color",
 Value: "blue"
}
```



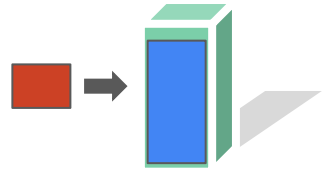
```
Toleration: {
 Key: "color",
 Value: "blue",
 Operator: "Equal",
 TaintEffect: "NoSchedule"
}
```

Dedicated machines (Taints)

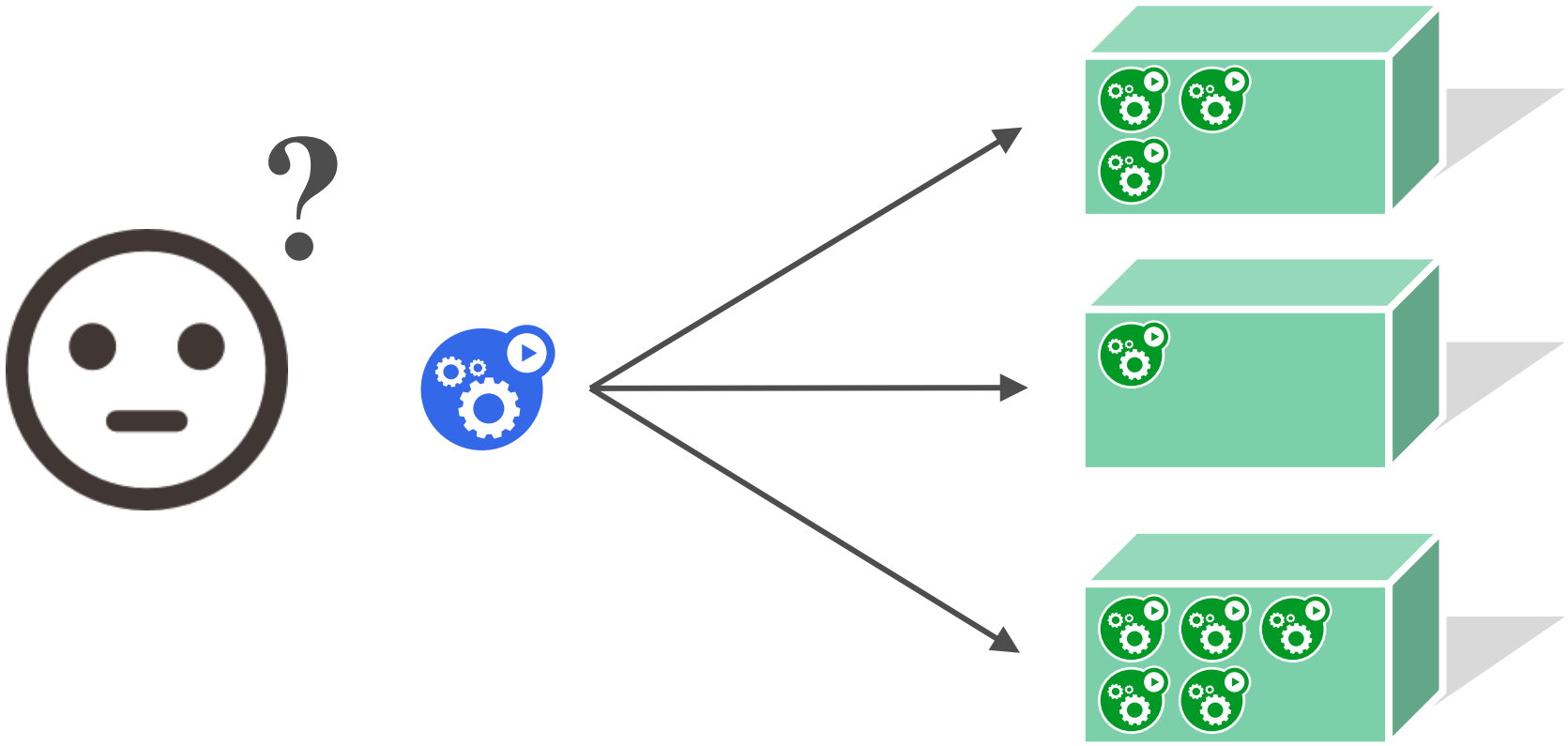


Predicate summary

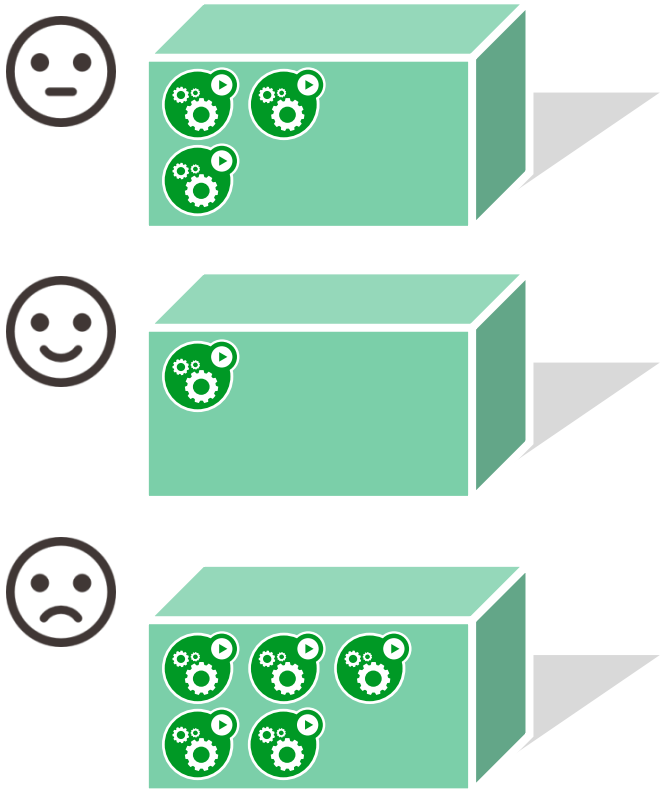
- Prevent overcommit
- Prevent co-scheduling
- Force co-scheduling
- Dedicated Nodes



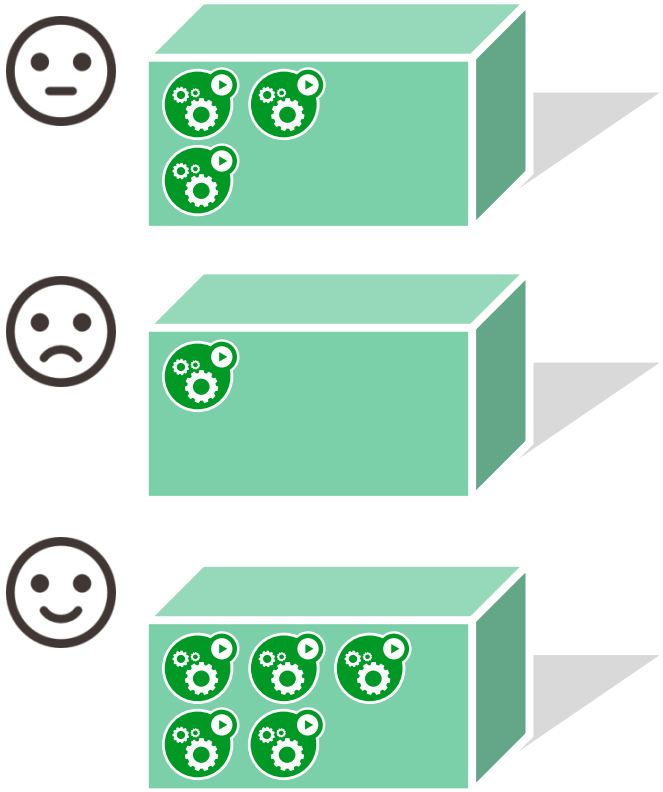
Best fit vs worst fit



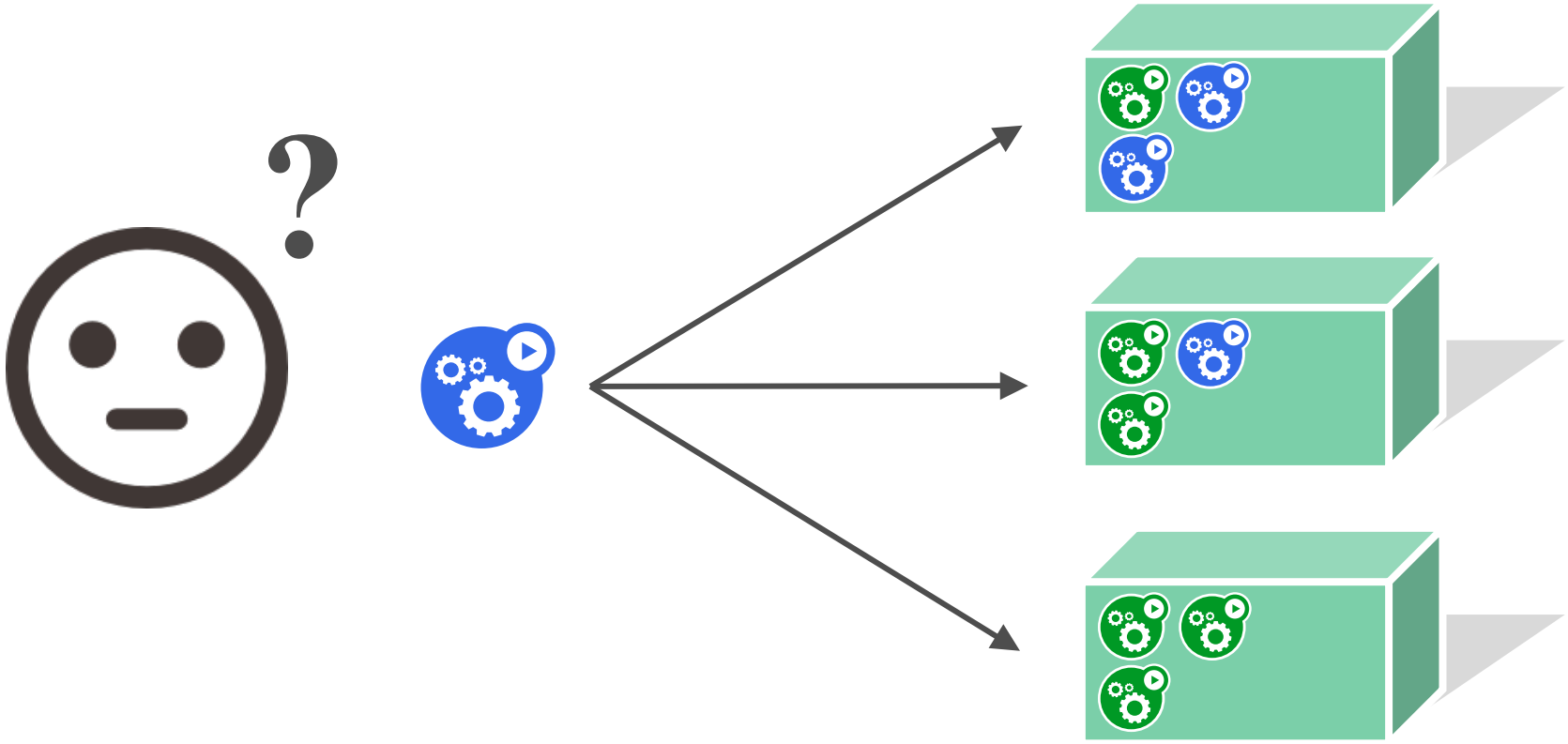
Best fit vs worst fit



Best fit vs worst fit

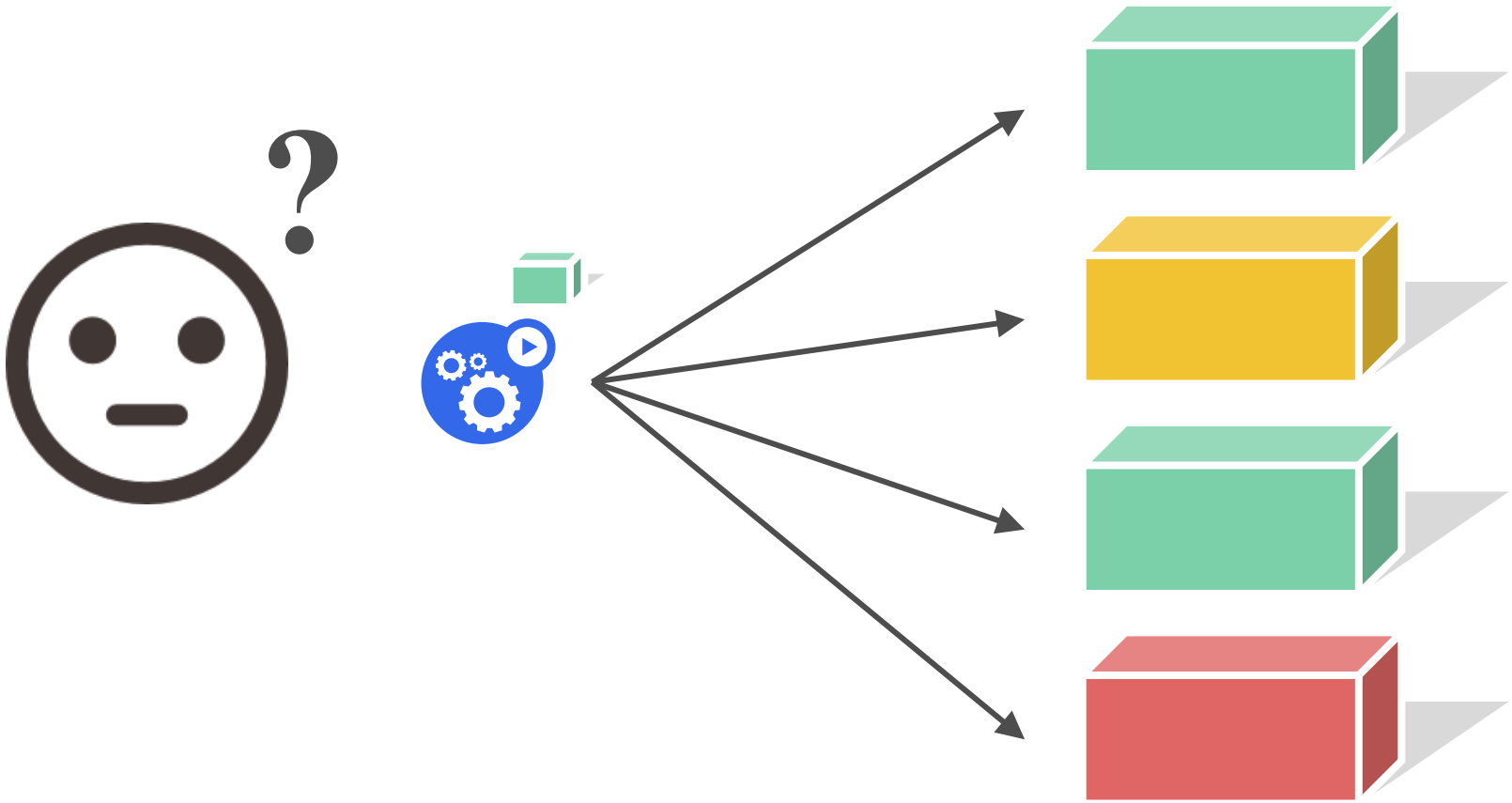


Selector Spreading

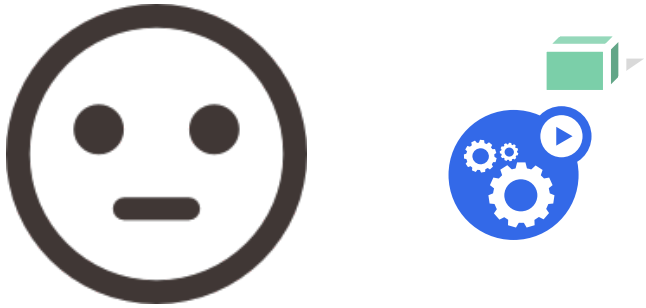


Selector Spreading

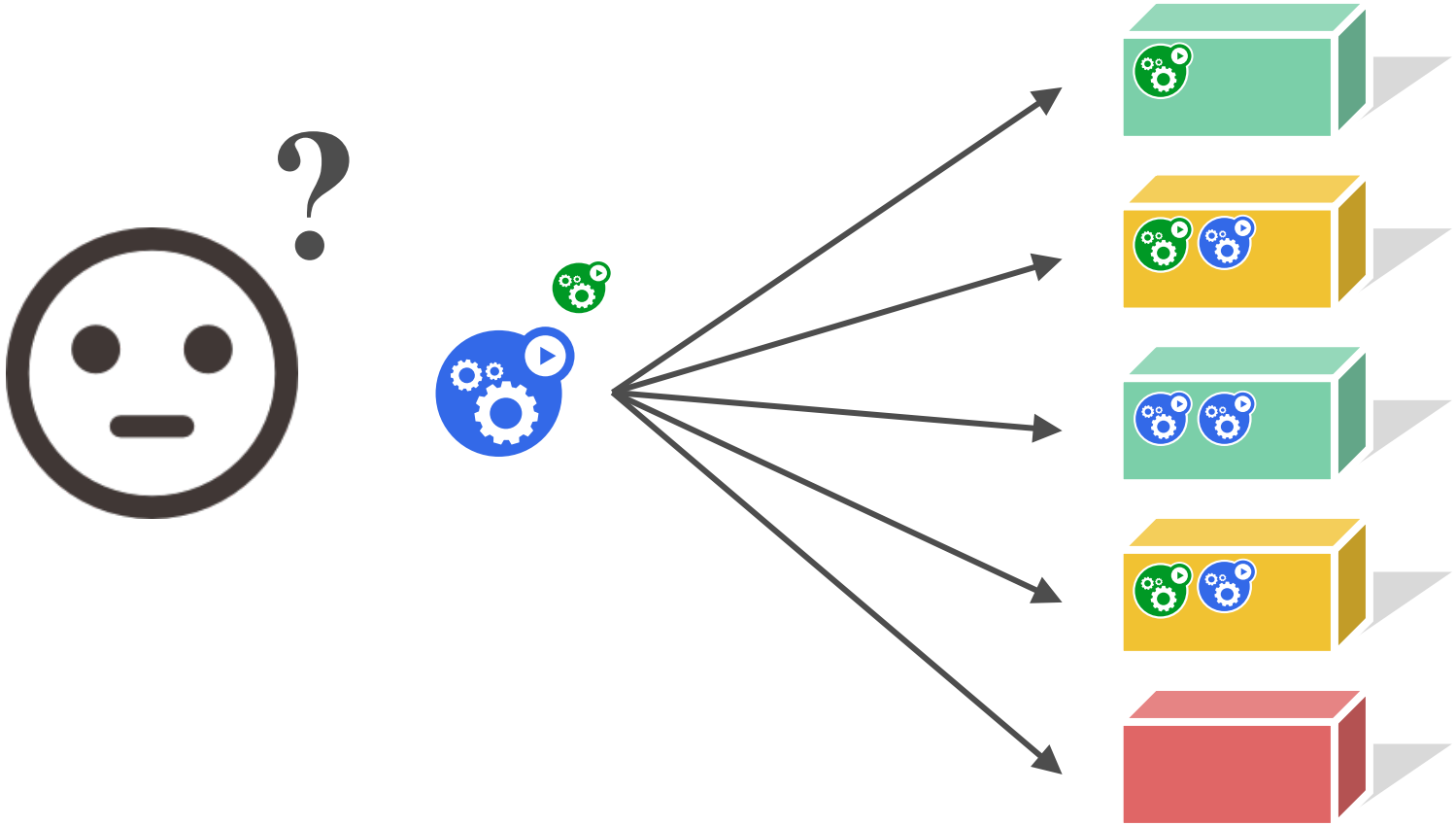
Node Affinity



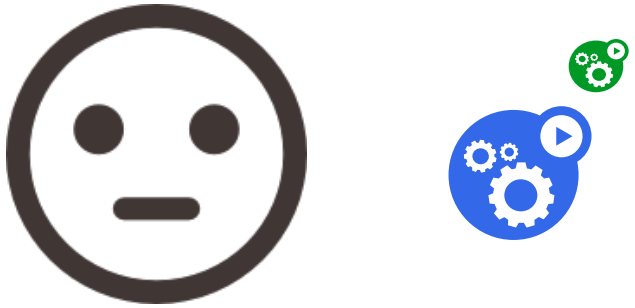
Node Affinity

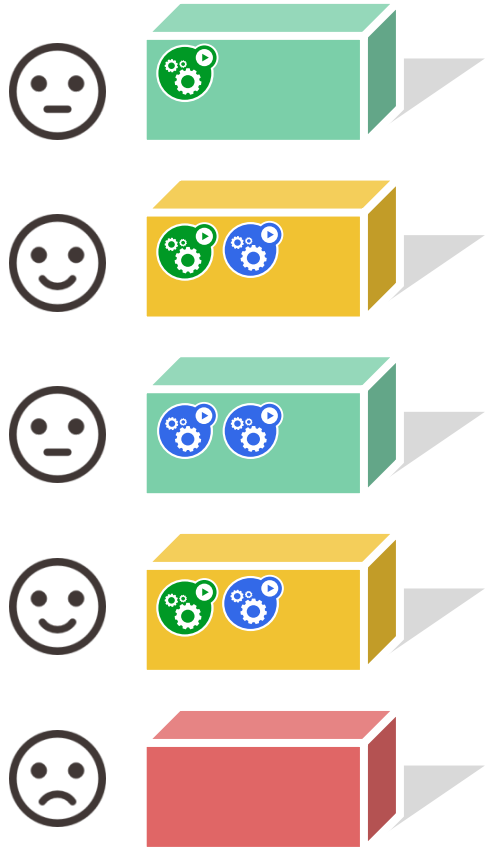


Pod Affinity

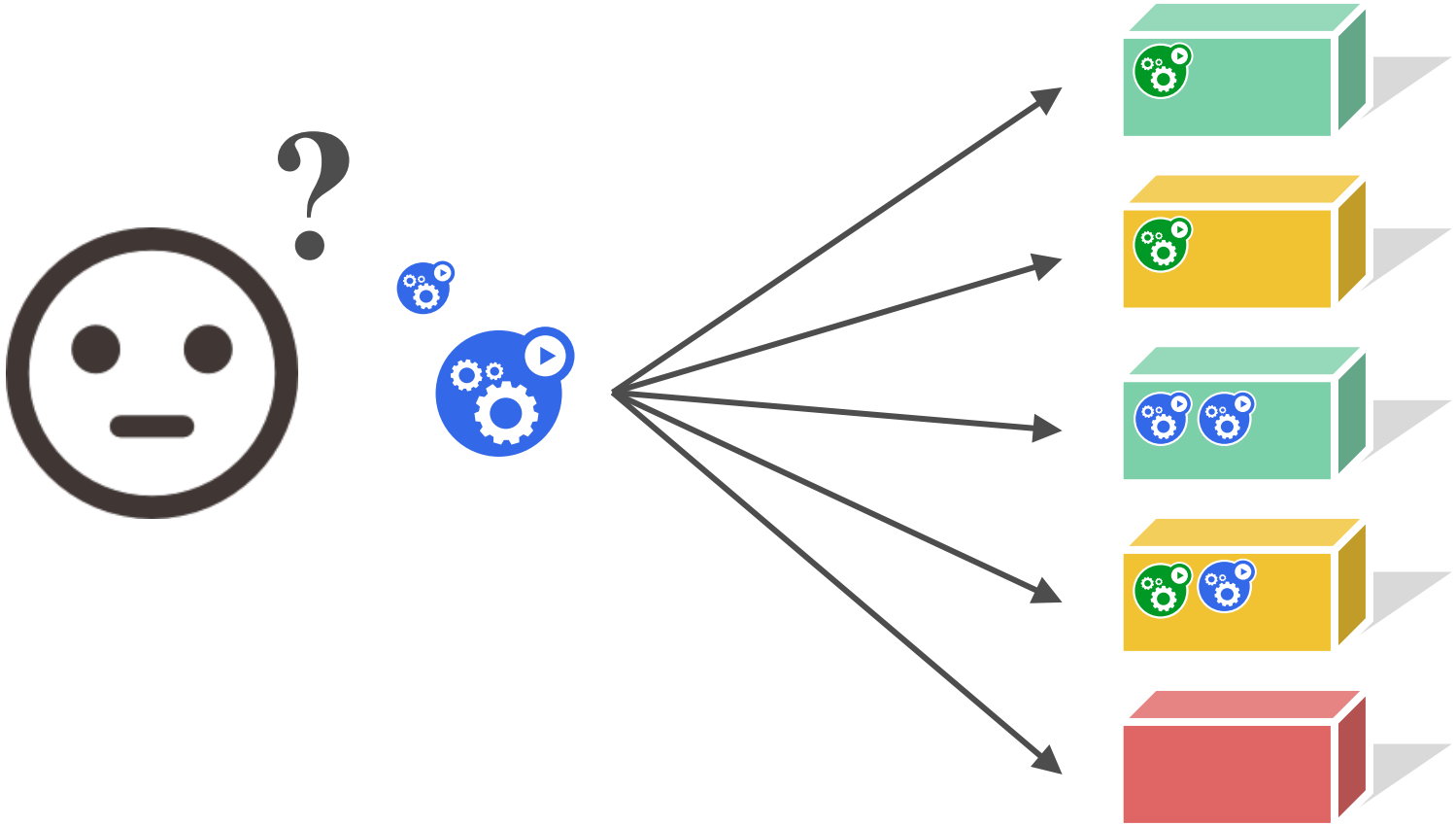


Pod Affinity

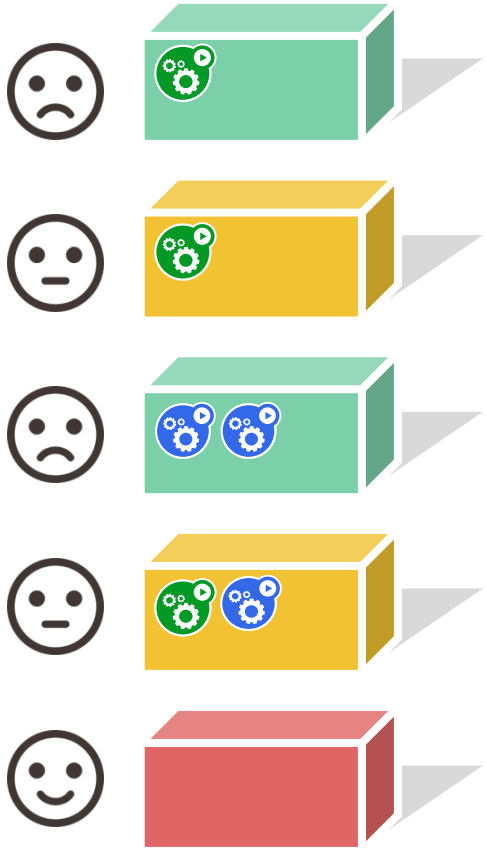




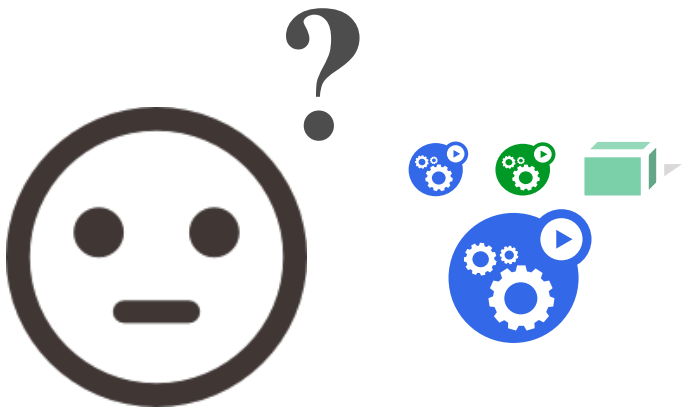
Pod Anti-Affinity

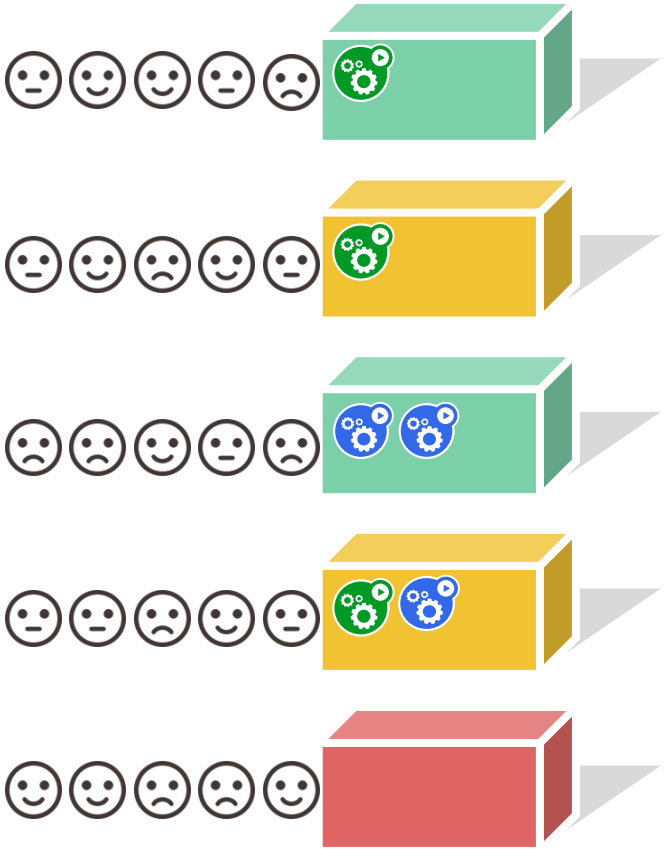


Pod Anti-Affinity

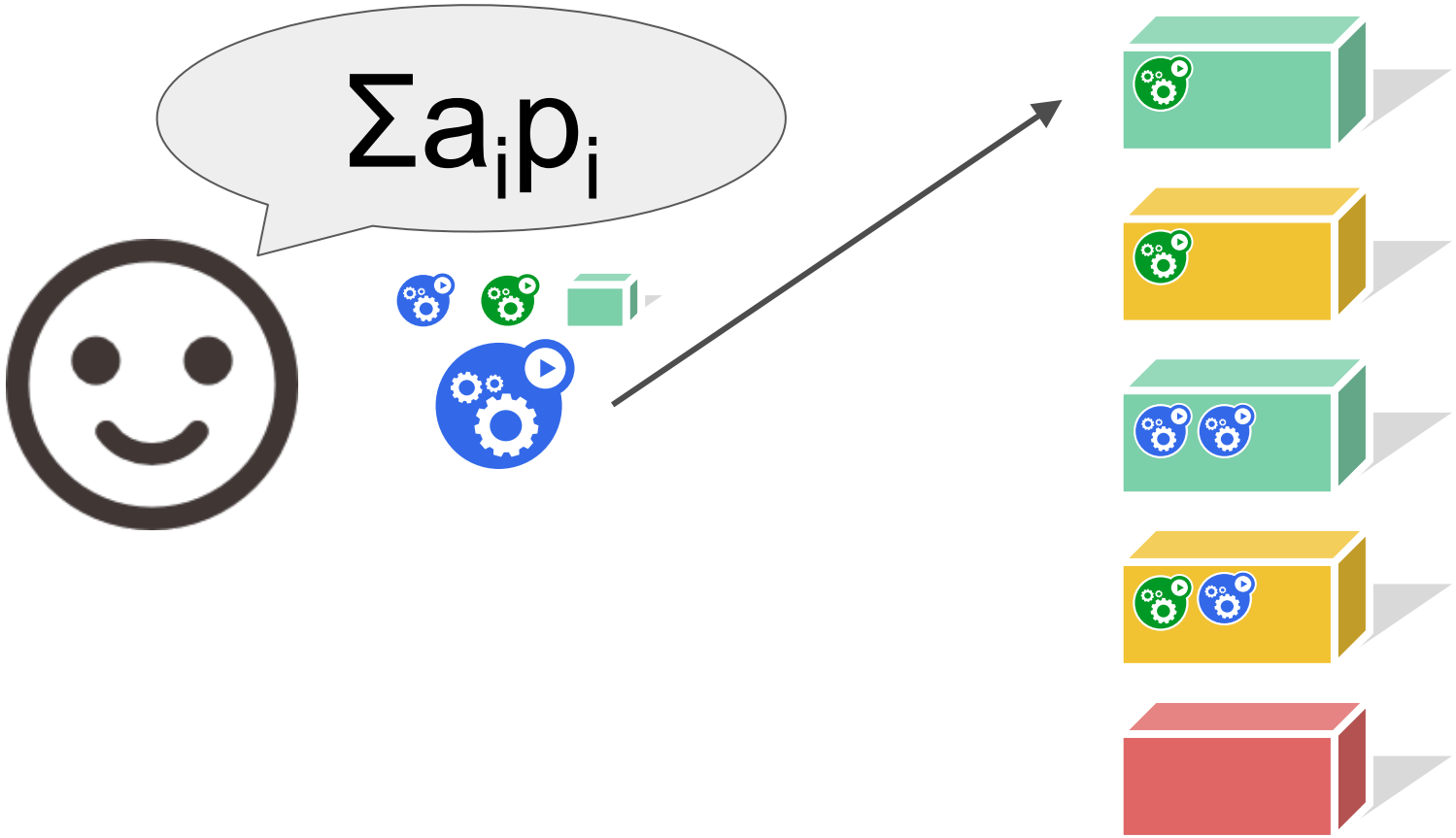


How to combine scores?



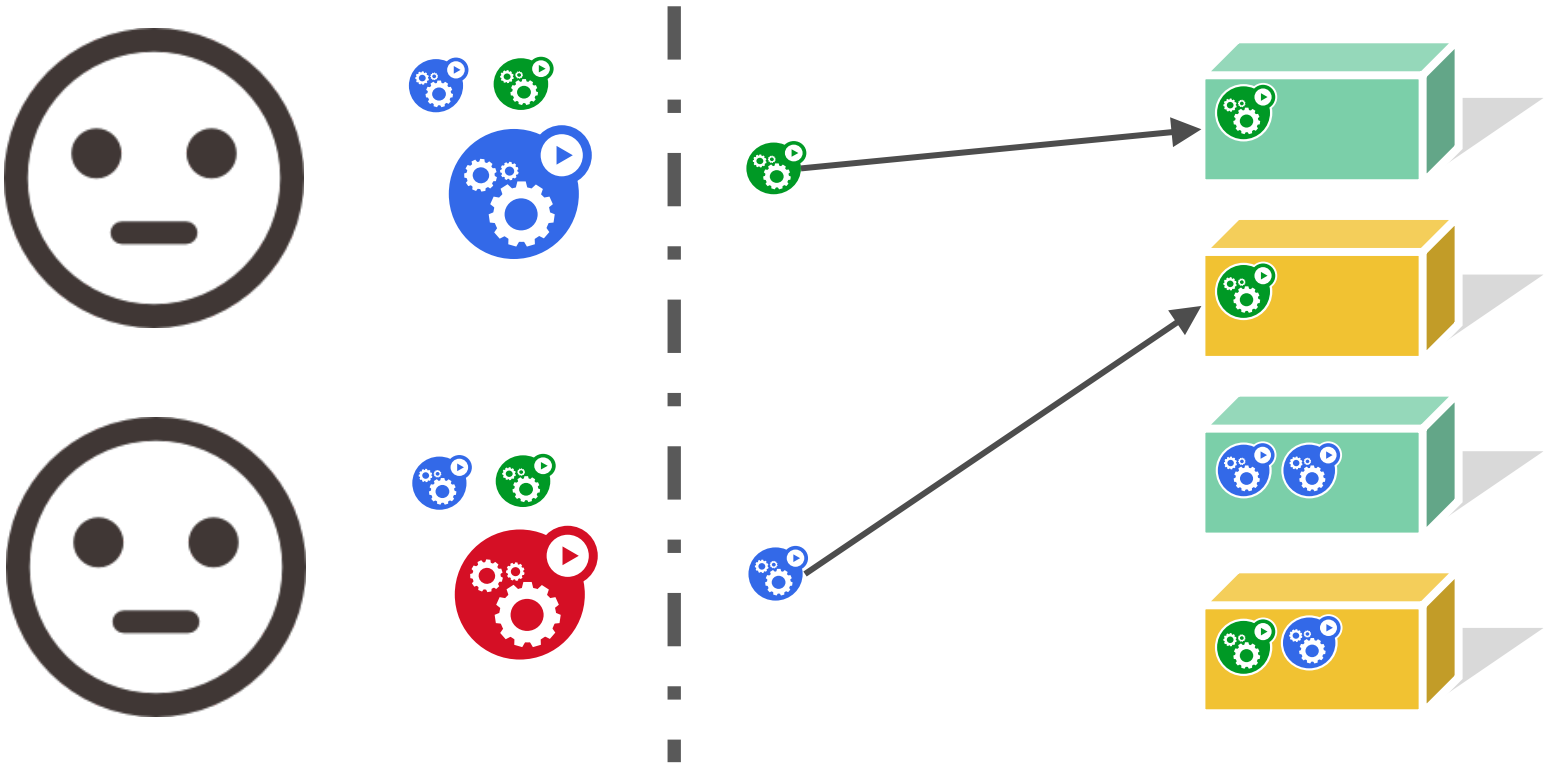


Linear combination



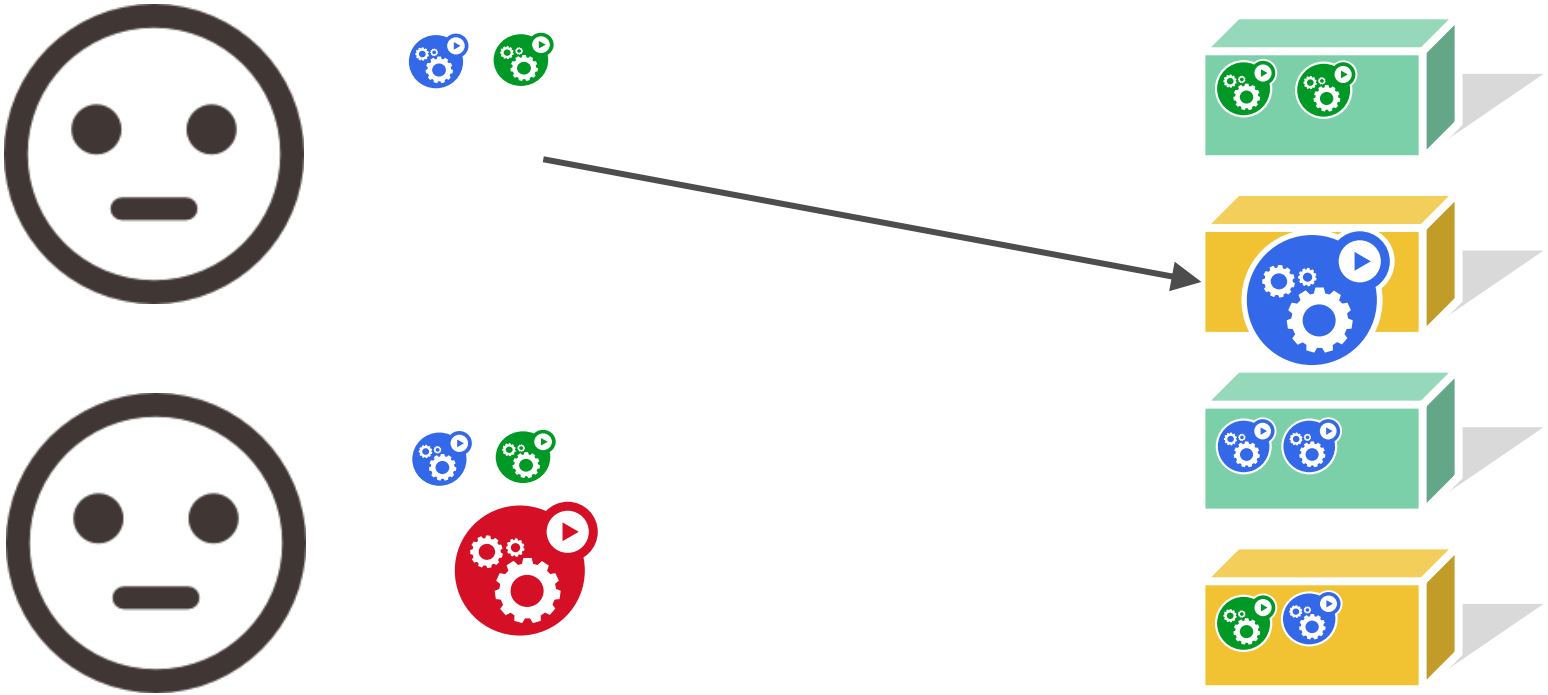
Resource Management

ResourceQuota



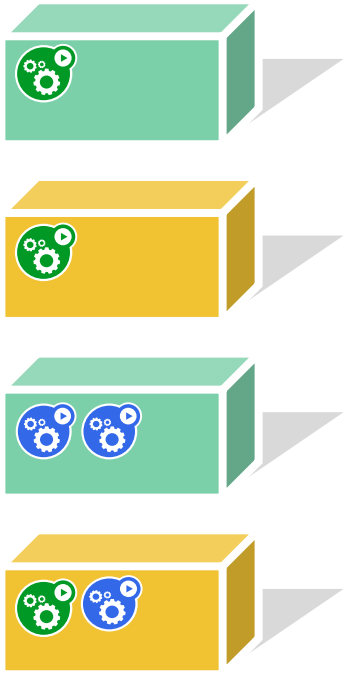
Rescheduler

Rescheduler

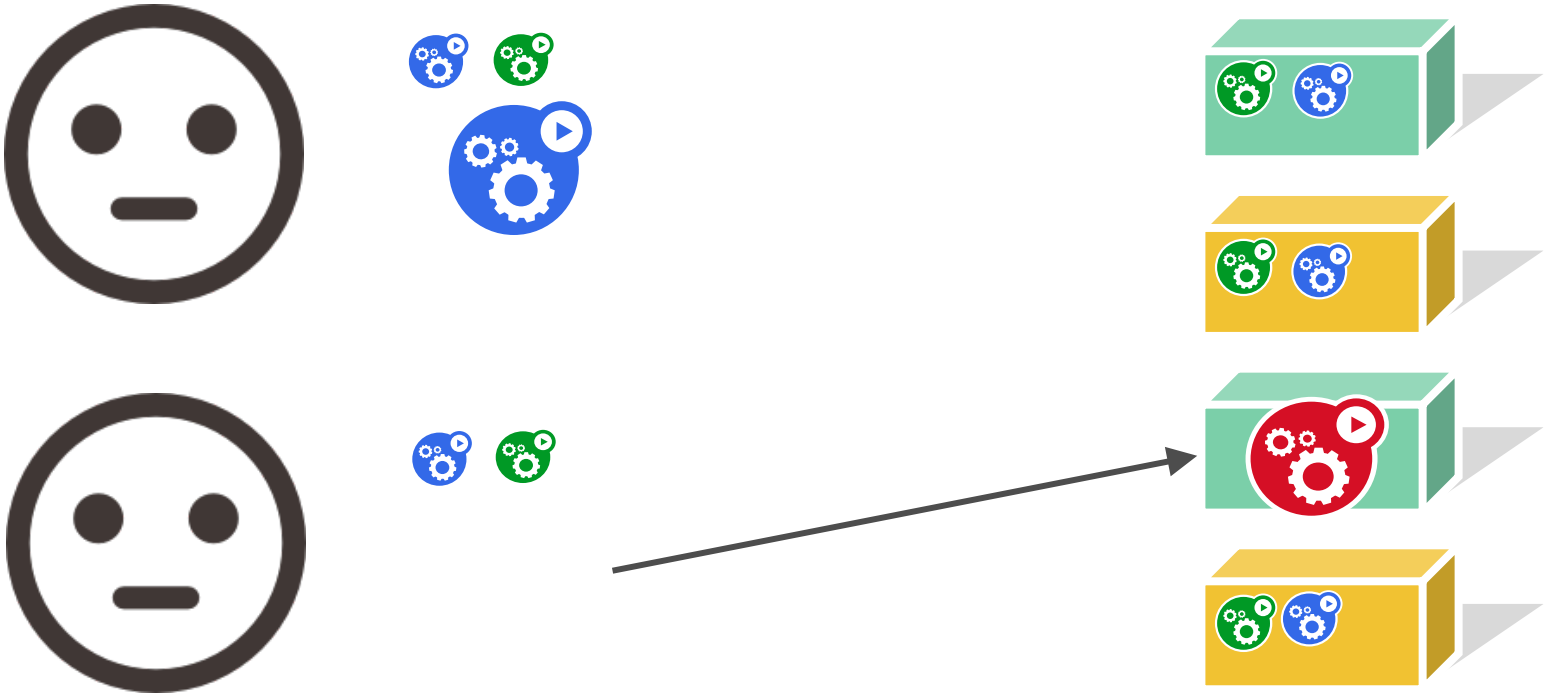


Priority/Preemption

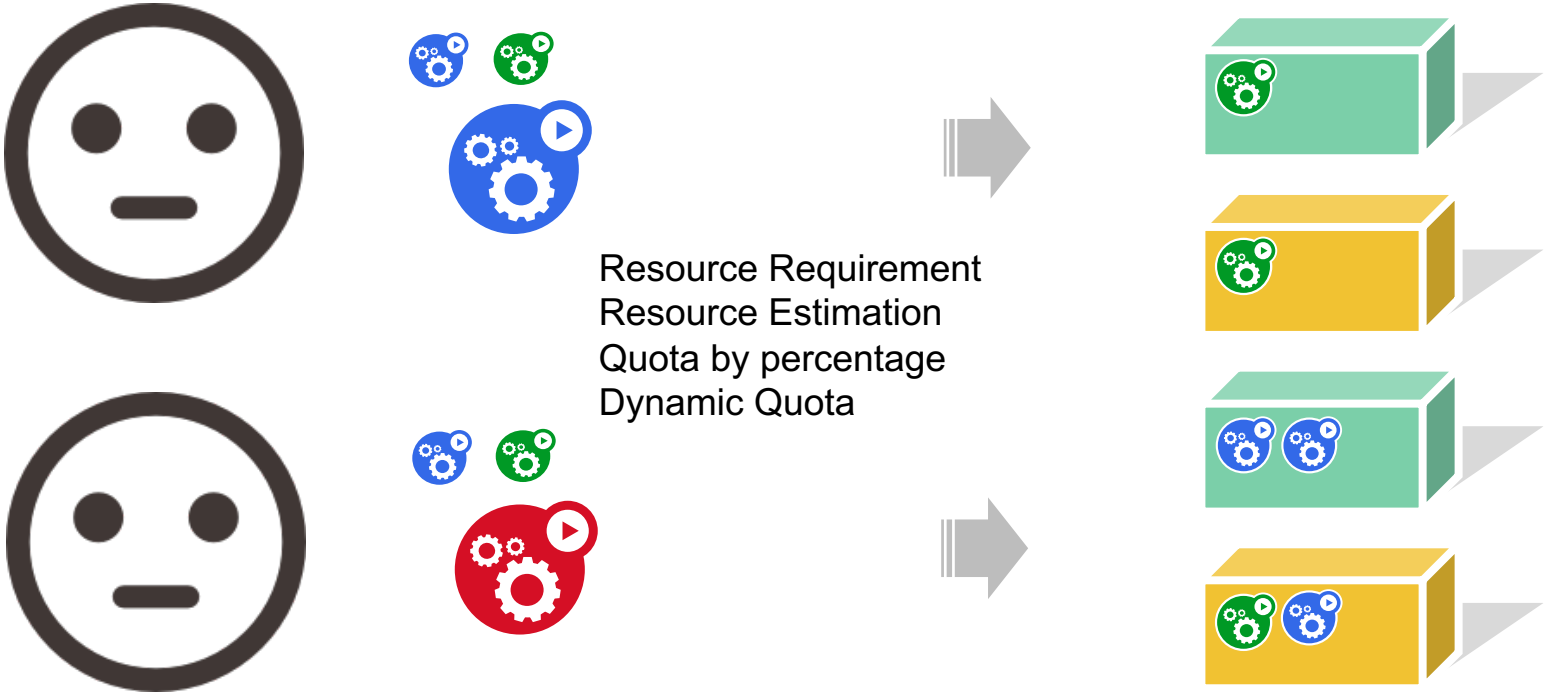
?



Priority/Preemption



Flexible resource allocation



kubernetes-incubator/kube-arbitrator

- **Sponsor**: Joe Beda (@jbeda)
- **Champion**: Timothy St. Clair (@timothysc)
- **SIG**: sig-scheduling, sig-bigdata
- **Github**: kubernetes-incubator/kube-arbitrator
- **Status**: startup and need more contributor ☺

Live Demo

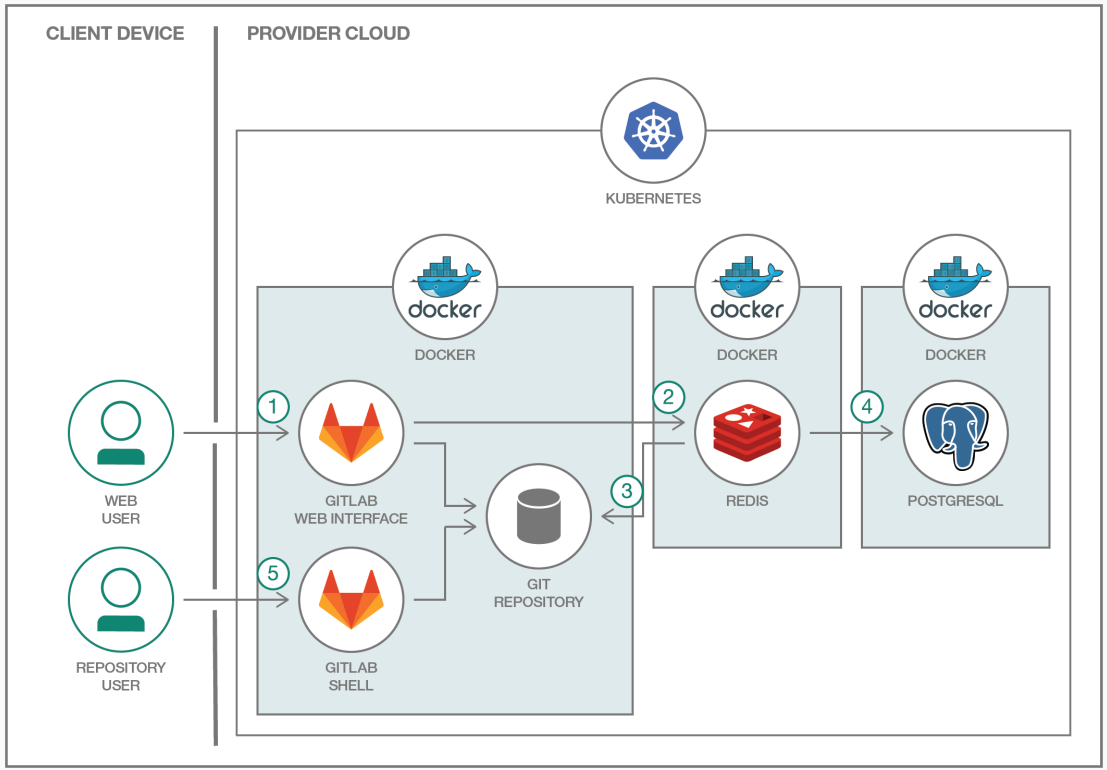
Join community

SIG Scheduling

- Meeting every other Monday 2pm PST
- #sig-scheduling Slack channel
- kubernetes-sig-scheduling@ mailing group

IBM Code: Deploy a distributed GitLab leveraging Kubernetes and Docker

https://developer.ibm.com/code/patterns/run-gitlab-kubernetes/



1. The user interacts with GitLab via the web interface or by pushing code to a GitHub repository. The GitLab container runs the main Ruby on Rails application behind NGINX and gitlab-workhorse, which is a reverse proxy for large HTTP requests like file downloads and Git push/pull. While serving repositories over HTTP/HTTPS, GitLab utilizes the GitLab API to resolve authorization and access and serves Git objects.
2. After authentication and authorization, the GitLab Rails application puts the incoming jobs, job information, and metadata on the Redis job queue that acts as a non-persistent database.
3. Repositories are created in a local file system.
4. The user creates users, roles, merge requests, groups, and more—all are then stored in PostgreSQL.
5. The user accesses the repository by going through the Git shell.

下载IBM Private Cloud，开始您的Kubernetes之旅：

https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W1559b1be149d_43b0_881e_9783f38faaff

Q&A

添加ibmopentech
加入讨论群与讲师互动

更多信息，请访问：http://ibm.biz/opentech-m