1. 定时任务？周期设为多少？

二．Service打分

1. 分数越接近0，则Service越好
2. 产生调度的场景
   1. Service需要的pod数量大于实际的pod数量
   2. Service需要的pod数量小于实际的pod数量
   3. 响应时间大于预设值
3. 其他需要考虑的因素
   1. Pod所在node相同，减分（但是这不足以进行调度）
   2. 某个pod的资源使用率比较高，而其他pod的资源使用率很低，减分（这种情况需要怎么做呢？？？）
4. 打分规则
   1. Service需要的pod数量a大于实际的pod数量b，打（a-b）分
   2. Service需要的pod数量a小于实际的pod数量b，打（a-b）分
   3. Service响应时间大于预设值（预设值可配置），打-100分
5. 规则执行
   1. Service响应时间优先于pod数量的比较，所以先检测Service的响应时间，如果响应时间大于预设值，则直接打-100分。如果响应时间小于预设值，则打（pod数量的期望值- pod数量的实际值）分
6. Servicce打分的时效性

维护一个表格，表格内容如下（这里的预期pod数不是从应用画像获取的，二是场景分析得出的结果）

|  |  |  |
| --- | --- | --- |
| ServiceName | 实际pod数 | 预期pod数 |
| ServiceA | 3 | 4 |
| ServiceB | 5 | 5 |
| …… | …… | …… |

当打分系统打出一个分数时，一方面将分数发给场景分析，另一方面去master中修改表格的实际pod数。场景分析看到分数是好的，则直接丢弃，再看下一条分数，如果场景分析判断分数不好，再去master中获取service的实际pod数和预期pod数，如果实际pod数与预期pod数不相同，则打分无效。只有当pod的实际个数与预期相同时，打分才有效，则分析场景一方面告诉调度算法要做什么调度，另一方面去master中修改表格的预期pod数。调度算法产生一条调度规则，调度算法通知规则执行器执行规则。当规则执行器执行完规则，k8s集群中Service的pod的数量是预期值后，打分系统对从prometheus获取的数据中service的实际pod数就等于预期的pod数了。

三．Node打分

1. Node分数越接近于0越好

1. 产生调度的场景（硬性要求）
   1. Node上的cpu超过80%
   2. Node上的mem超过80%
   3. Node上的磁盘超过90%
   4. Node上的cpu和内存的平均值超过70%
   5. Node上的cpu低于10%且内存低于10%
2. 其他考虑因素（附加要求）
   1. 同一类型的不同服务的pod数量（不足以触发调度）
3. 打分规则
   1. 对Node上的cpu利用率a，打X~N(50%,a)分
   2. 对Node上的mem利用率b，打X~N(50%,b)分
   3. 对Node上的磁盘利用率c，如果超过90%，打1分，否则打0分
4. 执行规则
   1. 初步想法是权重相同；另一个想法是直接把这三个分发给场景分析。
5. Node打分的时效性
   1. 和Service一样，维护一个额外的表格（当然这两个表格可以合并为1个）