



中国科学院软件研究所
Institute of Software Chinese Academy of Sciences

基于Kubernetes的RISC-V异构集群云任务调度系统

报告人：蒋筱斌

中国科学院软件研究所

2022年6月

目录

1

研究背景

2

系统设计

3

实验评估

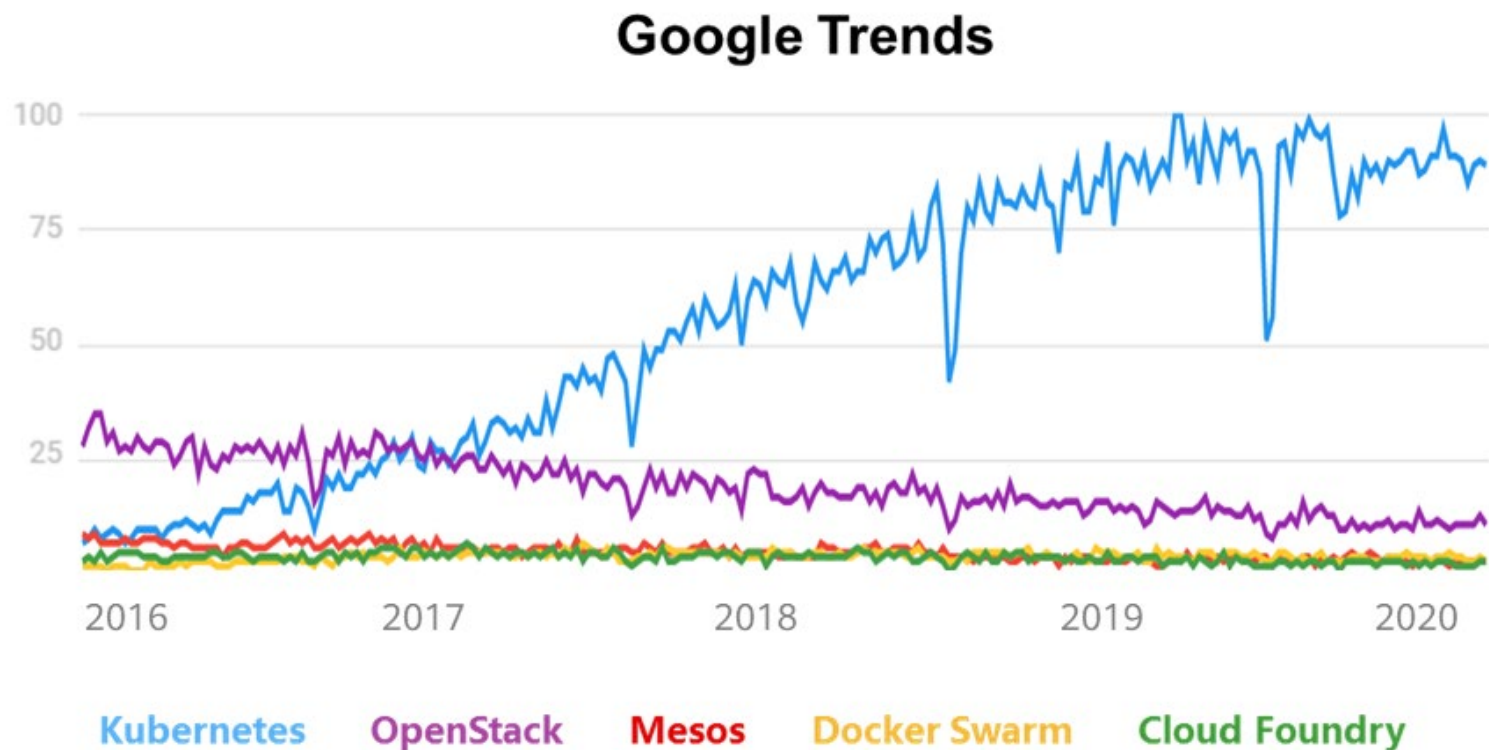
4

总结与展望

Kubernetes

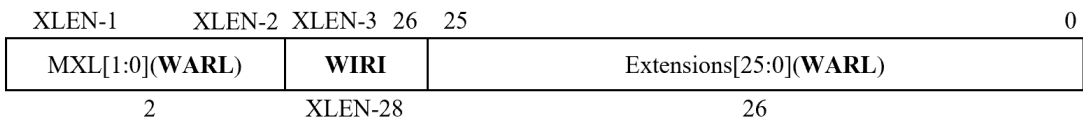
Kubernetes热度

CNCF (2020年度)最新的调查显示, Kubernetes 已经成为约定俗成的容器编排器, 有82% 的单位在生产中使用了 Kubernetes。



精简化、模块化、开源化

基础指令集和扩展指令集：芯片通过misa CSR（Control and Status Register，控制状态寄存器）管理扩展指令集。

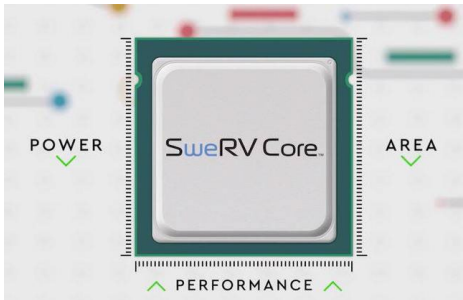


- V扩展：增强处理器的SIMD操作，将向量寄存器的宽度与指令集分离开，解决了指令集冗余和上层软件适配的问题；
- H扩展：加入了管理程序模式和基于内存页的二级地址翻译机制，提高在同一台计算机上运行多个操作系统的效率；
- K扩展：提供了一系列密码学相关指令，提升密码学算法速度，并降低应用程序大小；

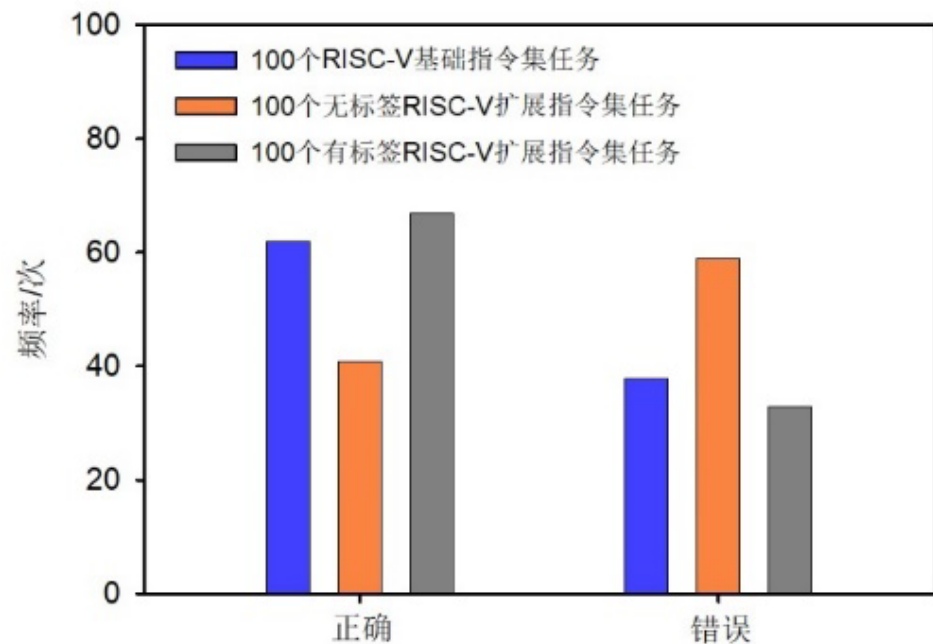
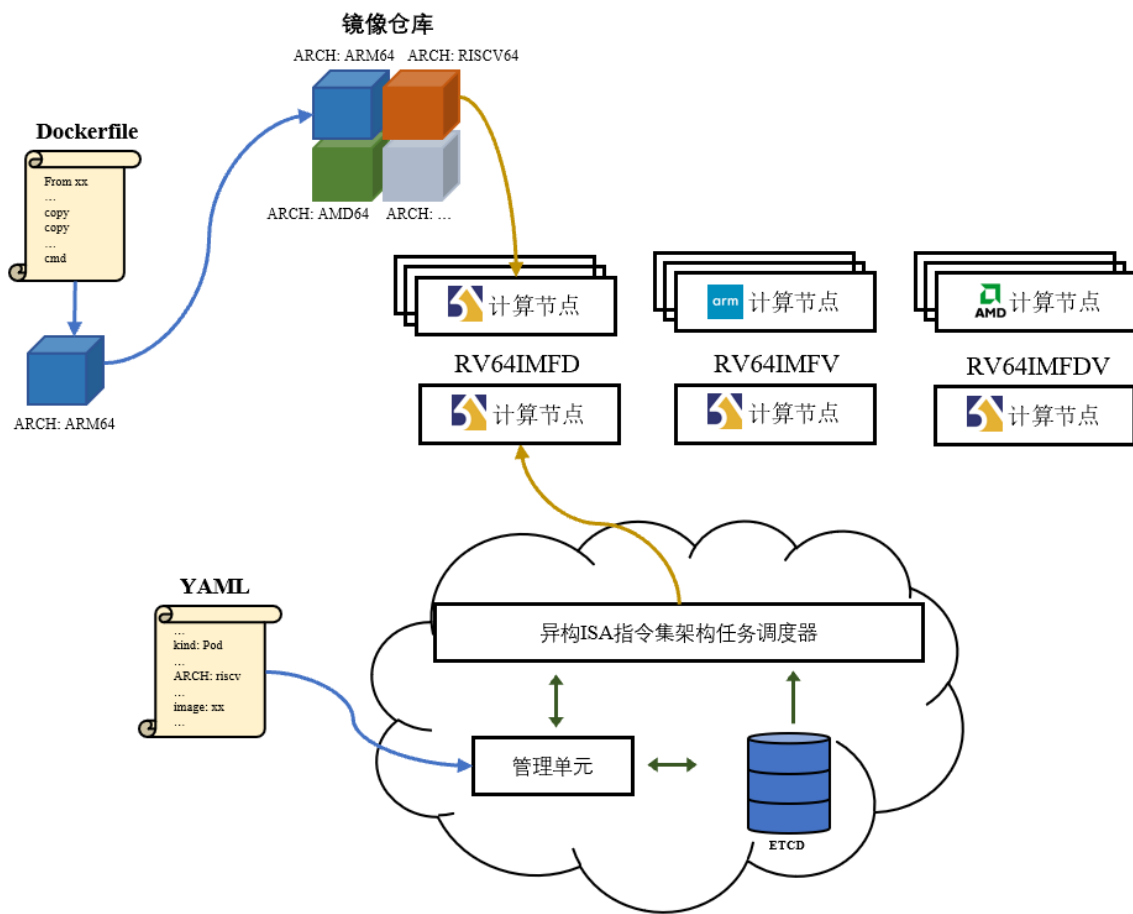
比特位	字母	描述
0	A	原子扩展
1	B	位操作扩展
2	C	压缩指令扩展
3	D	双精度（64bit）浮点扩展
4	E	RV32I 的子集，只支持 16 个整数寄存器
5	F	单精度（32bit）浮点扩展
6	G	存在其他标准扩展
7	H	支持 Hypervisor 管理程序扩展
8	I	RV32I/64I/128I base ISA
9	J	动态翻译语言扩展
10	K	密码学扩展
11	L	十进制浮点扩展
12	M	整数乘法与除法指令
13	N	用户态中断
14	O	保留
15	P	压缩单指令多数据流扩展
16	Q	四倍精度浮点指令扩展
17	R	保留
18	S	特权态
19	T	事务内存扩展
20	U	用户态
21	V	向量扩展
22	W	保留
23	X	存在非标准扩展
24	Y	保留
25	Z	保留



香山：开源高性能RISC-V处理器



Kubernetes+RISC-V



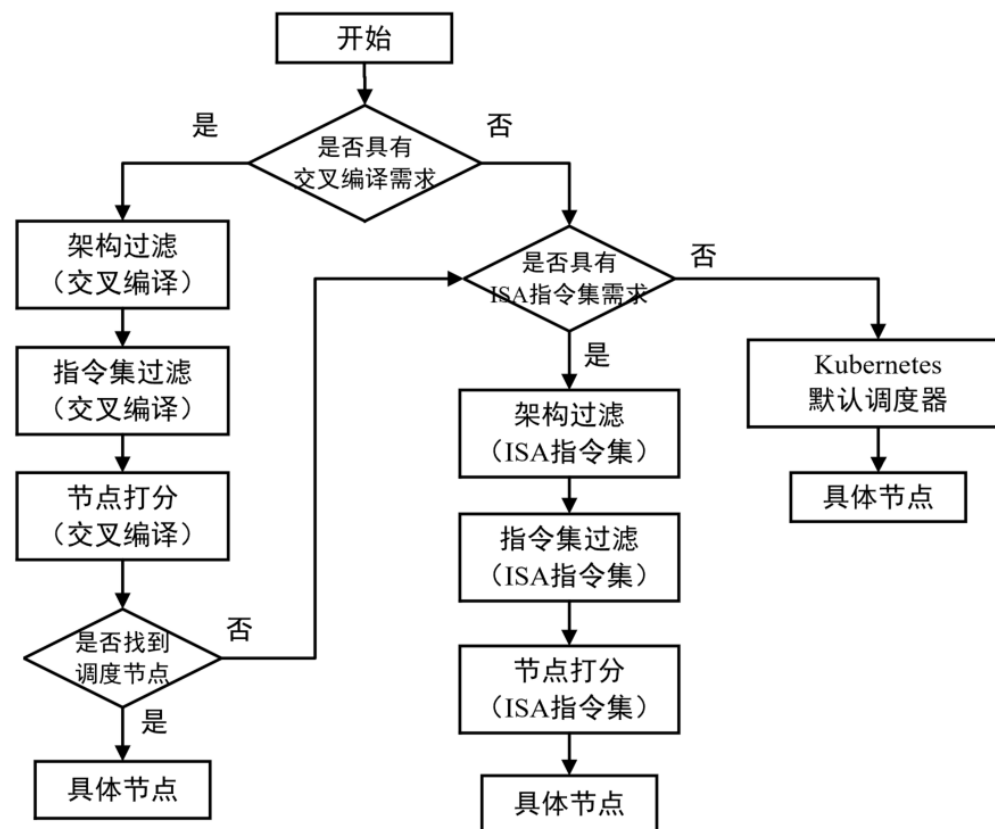
动机

任务创建时，现有调度器应对指令集需求任务的调度正确率为62%（调度RISC-V基础指令集任务）、41%（调度RISC-V扩展指令集任务）、67%（调度RISC-V扩展指令集任务且有“RISC-V”节点匹配标签），无法支持混合多种指令集架构的任务调度场景；

系统设计

ISAMatch模型流程

1. **交叉编译**，判断当前任务是否具有交叉编译需求；
2. **ISA指令集**，判断当前任务是否有具体ISA指令集架构需求；
3. **架构过滤**，粗粒度的过滤不匹配的顶层指令集架构节点；
4. **指令集过滤**，细粒度的过滤RISC-V特有的扩展指令集架构；
5. **节点打分**，综合指令集亲和性、同种指令集架构节点数量、资源利用率，选取得分最高的节点作为候选绑定节点。



基于Kubernetes的ISA架构匹配策略的设计与实现

集群预处理

两种编译模式

1. 混合编译模式QEMU

- 节点QEMU代表节点具有的交叉编译指令集架构属性;
- 任务QEMU标签代表任务所请求的指令集架构属性;

2. 节点指令集架构属性的同构编译ARCH

- 节点ARCH标签代表节点具有的指令集架构属性;
- 任务ARCH标签代表任务所请求的指令集架构属性;

```
01. # Pod YAML Label
02. metadata
03.   name: test
04.   labels:
05.     ARCH: AMD64
06. # Node YAML Label
07. Name: Node1
08. Roles: Master
09. Labels: ARCH=AMD64
10. QEMU=RV64IMFDV
```

```
01. # 获取所有节点 ip
02. string=`kubectl get nodes`
03. echo $string
04. i=0
05. for ip in $string
06. do
07. if [[ $ip =~ ^([0-9]{1,2}|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.([0-9]{1,2}|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.([0-9]{1,2}|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.([0-9]{1,2}|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$ ]]
08. then
09.   # echo $ip
10.   nodes[i]=$ip
11.   let i+=1
12. fi
13. done
14.
15. for node in ${nodes[*]}
16. do
17.   ARCH=`ssh root@$node "uname -m"`
18.   case $ARCH in
19.     "riscv64"|"riscv32")
20.     ISA=`ssh root@$node "gcc -c -Q --help=target | grep march"`
21.     kubectl label node $node ARCH=${ISA#*:}
22.     ;;
23.   *)
24.     kubectl label node $node ARCH=$ARCH
25.     ;;
26.   esac
27. done
```

基于Kubernetes的ISA架构匹配策略的设计与实现

过滤策略

- 1. 架构过滤：粗粒度筛选同族ISA指令集架构节点
- 2. 指令集过滤：细粒度筛选具体的RISC-V扩展指令集架构节点，满足：

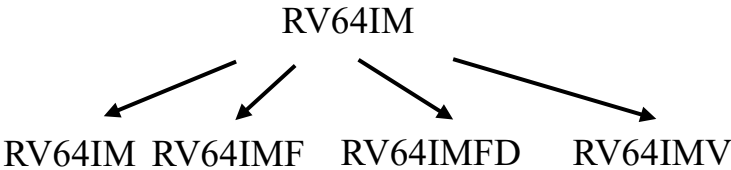
$$instruction_{Pod} \subseteq instruction_{Node}$$

打分策略

- 指令集亲和性：用于描述任务请求指令集模块和节点支持指令集模块的相似程度：

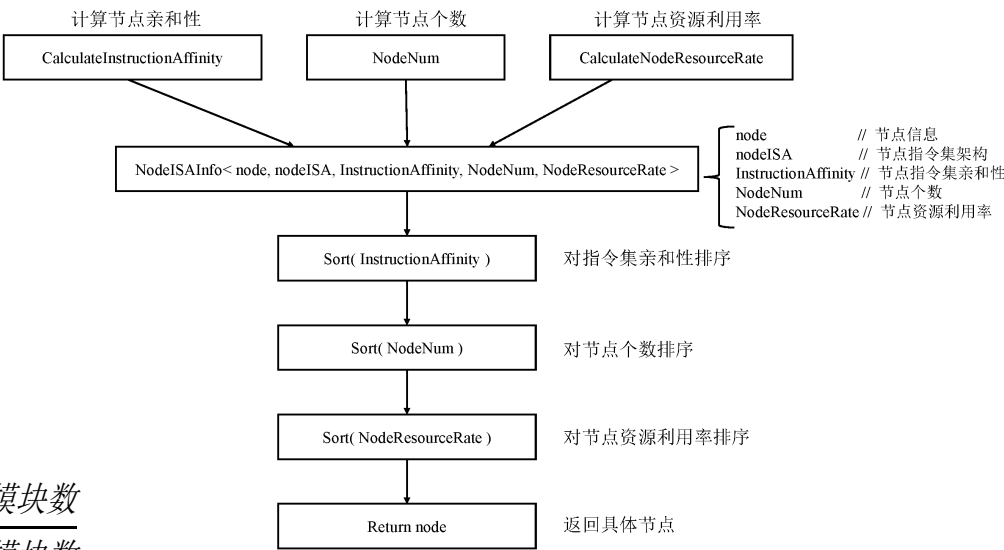
$$instructionAffinity = \frac{Pod模块数}{Node模块数}$$

- 节点个数：具有同种指令集架构的节点数量
- 节点资源利用率



指令集亲和性 = $\frac{任务模块数}{节点模块数}$

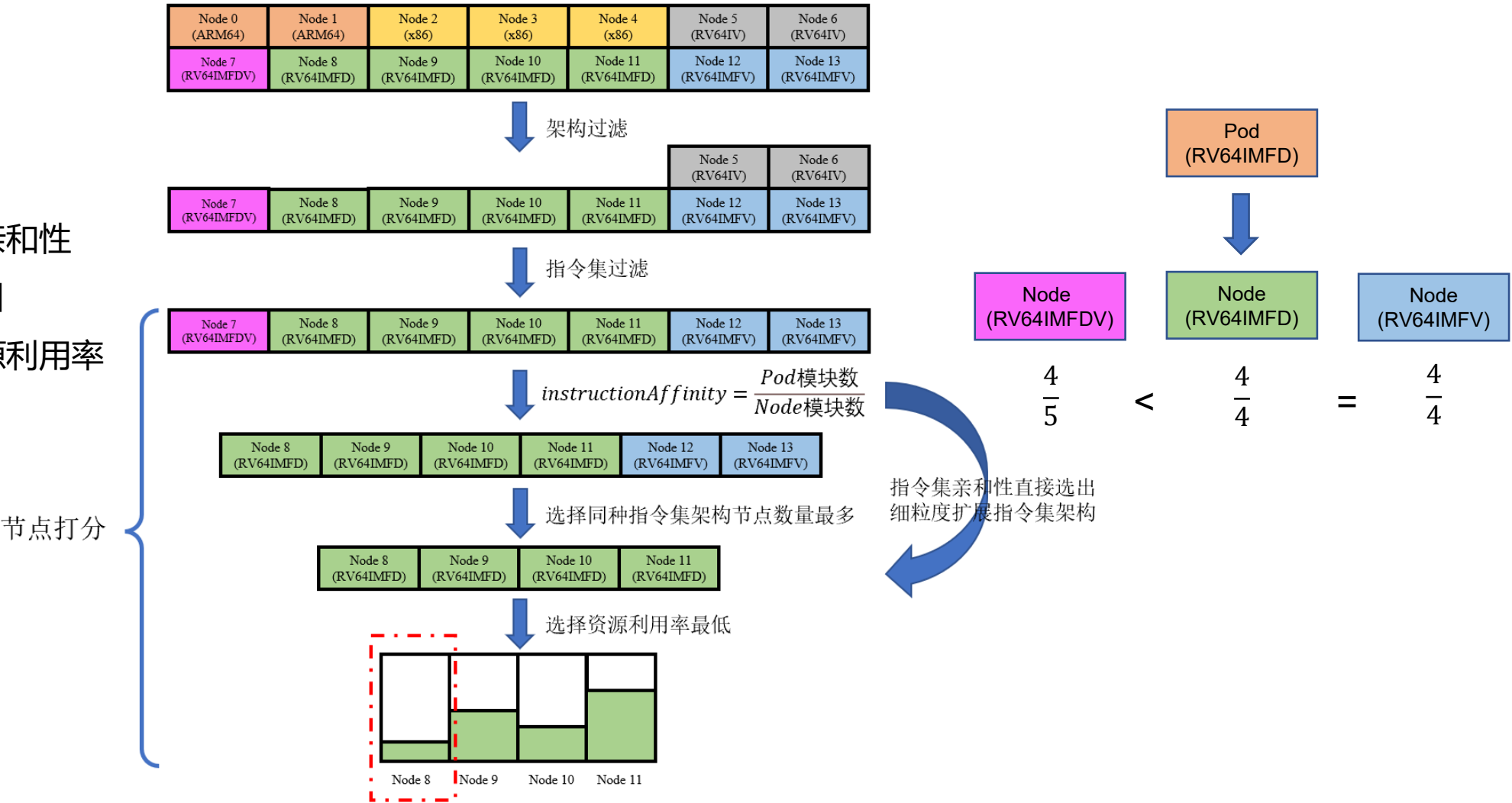
```
candidateNodes={}
if Contains( PodARCH , "RV" ) then
  for node in nodeList do
    if node->nodeARCH[:2] == "RV" && ContainIns(
      node->nodeARCH[4:] , PodARCH[4:] ) then
      candidateNodes = append( candidateNodes , node )
    end
  end
end
else
  for node in nodeList do
    if Contains( node->nodeARCH , PodARCH ) then
      candidateNodes = append( candidateNodes , node )
    end
  end
end
return candidateNodes
```



ISAMatch模型示例

基本流程

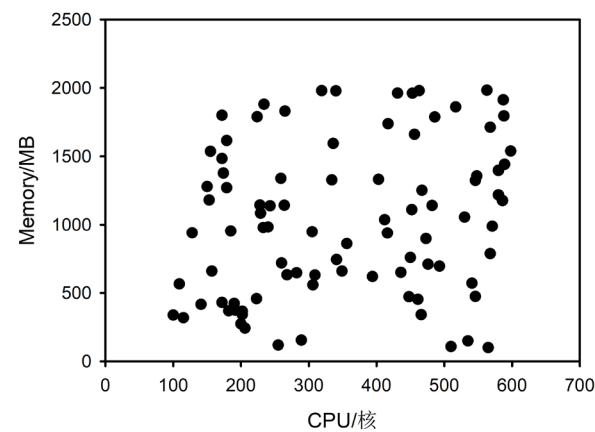
- 架构过滤
- 指令集过滤
- 节点打分
 - ✓ 指令集亲和性
 - ✓ 节点数目
 - ✓ 节点资源利用率



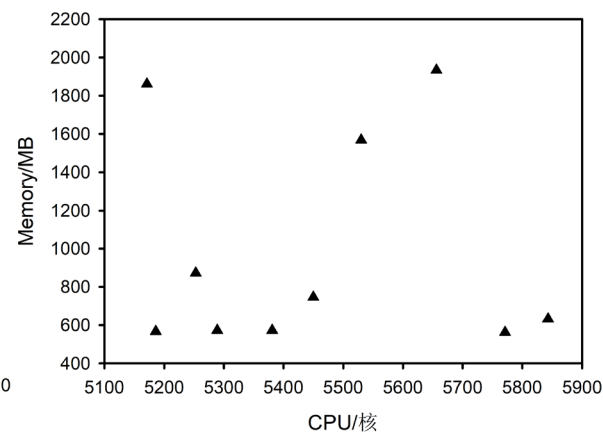
实验评估

实验数据

实验数据由随机函数生成了90个服务型任务和10个计算型任务，其中计算型任务为资源密集型任务，包括计算、编译、测试等，计算型任务通常会在固定时间内完成并退出集群，一般以Job资源对象请求。服务型任务资源占用较少，常驻于集群节点中，通过Service接口向外暴露服务IP，为用户提供应用服务，这类任务通常以Deployment资源对象请求。



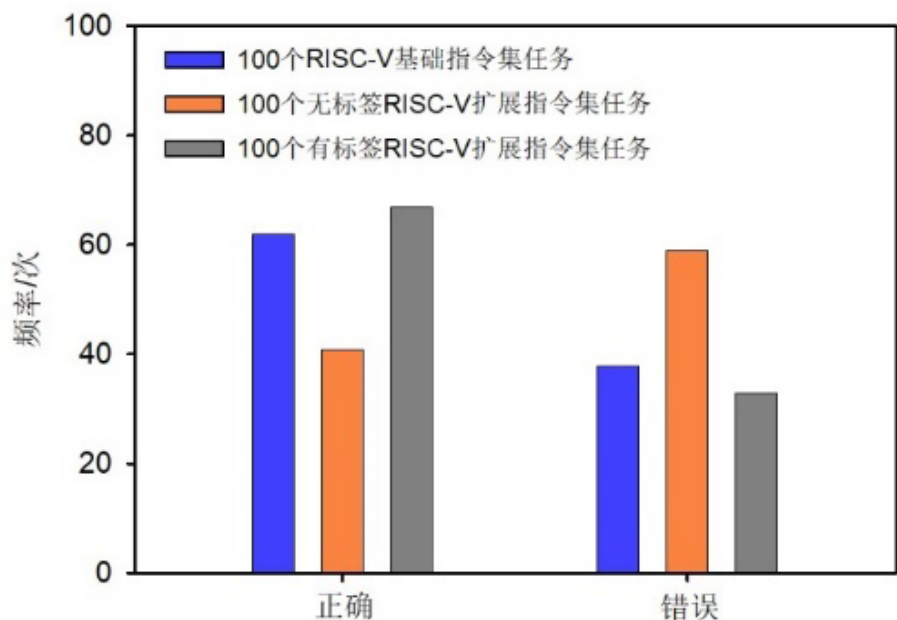
(a) 90个服务型任务资源分布



(b) 10个计算型任务资源分布

主机名	角色	指令集架构	扩展指令集	CPU	内存	QEMU
节点 1	Master	X86	-	8	32G	IMFDV
节点 2	Worker	ARM64	-	8	32G	-
节点 3	Worker	RISCV64	IMFD	8	32G	-
节点 4	Worker	RISCV64	IMFV	8	32G	-
节点 5	Worker	RISCV64	IMFDV	8	32G	-

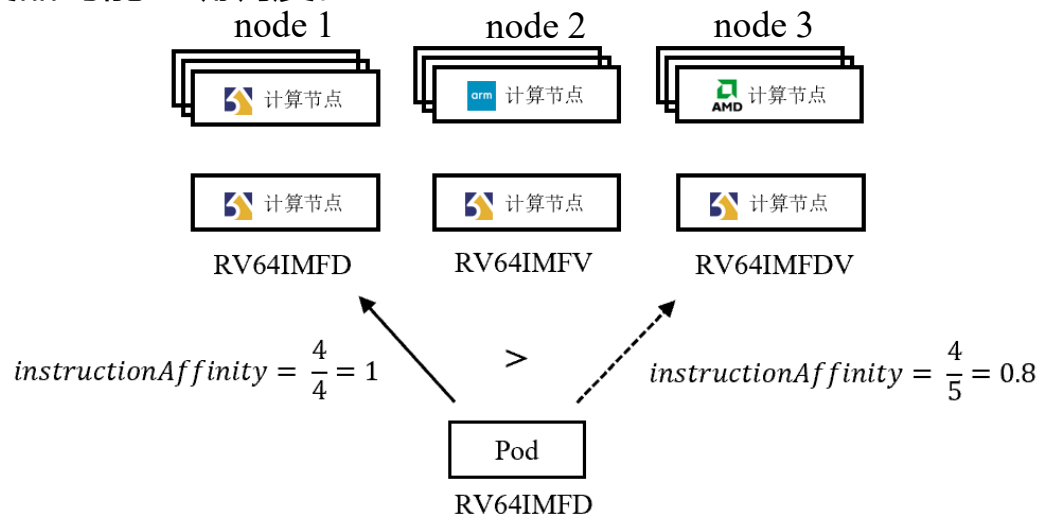
指令集架构调度正确性验证



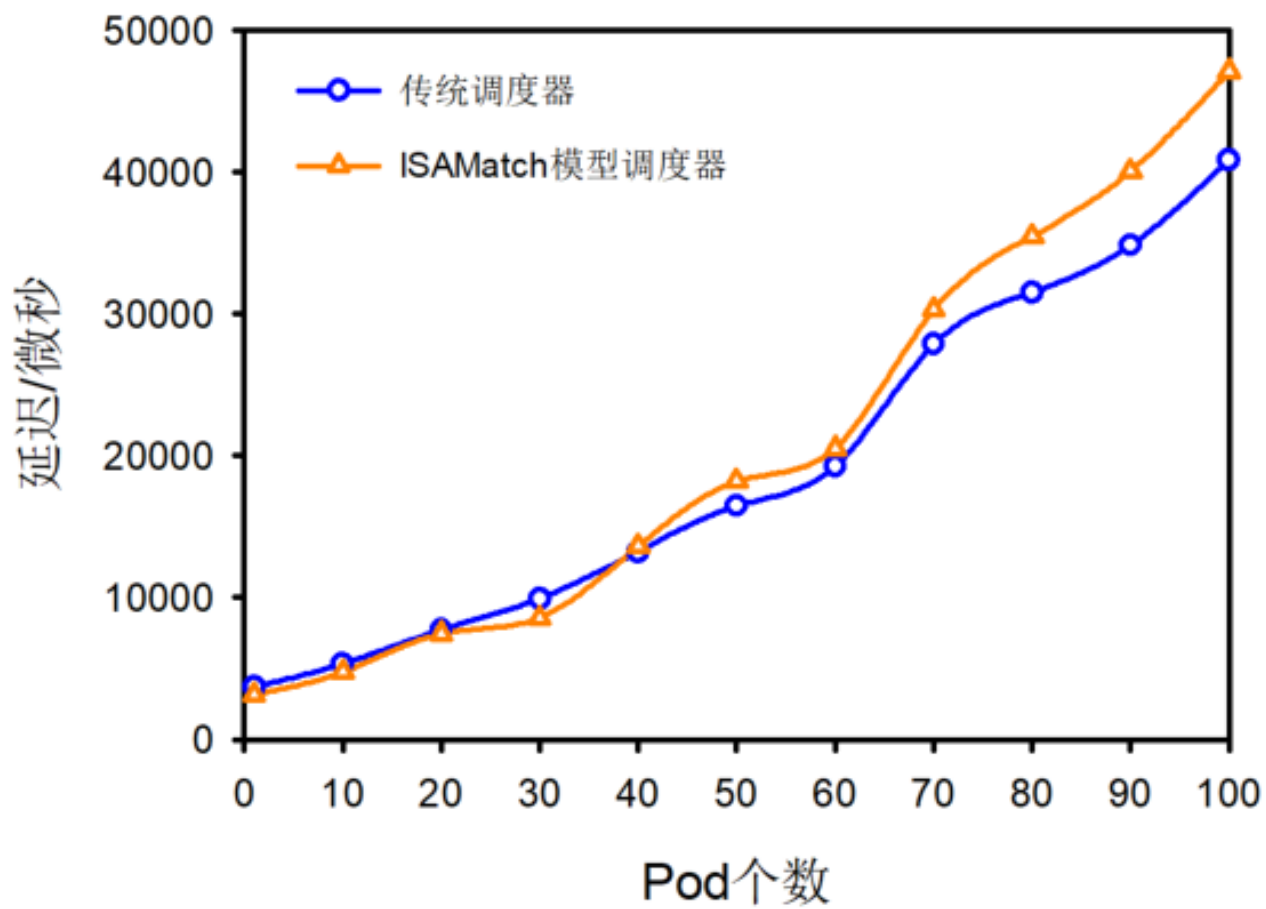
ISAMatch识别RISC-V扩展指令集架构

ISAMatch模型细粒度的通过指令集亲和性找到最佳匹配节点，如图x所示，RV64IMFD的任务可以成功运行在节点1和节点3上。根据指令集亲和性，ISAMatch模型会选取更为匹配的节点1作为最佳匹配节点，只有当节点1无法满足调度需求（如资源不足）时才会选用节点3进行调度。

- RISC-V基础指令集任务，即只需要部署到RISC-V节点上就能正常运行，调度正确率为62%；
- 未带有指令集架构相关标签的RISC-V扩展指令集任务，即需要精确到具体的扩展指令架构，调度正确率为41%；
- 带有“RISC-V”标签的RISC-V扩展指令集任务，即可以排除了ARM和X86两种架构节点，但部分调度未能精确识别扩展指令架构的节点，调度正确率为67%。
- RISC-V基础指令集任务和多种扩展指令集任务，带有ISAMatch模型调度器均能正确调度。



单任务及多任务调度延迟



- 部署40个以内的Pod时，传统调度器和ISAMatch模型调度延迟接近，大约1毫秒左右；
- 部署40个以上Pod时，ISAMatch模型调度延迟略大于传统调度器，且逐步增大；
- 部署单个Pod进行ISA指令集架构匹配的平均耗时为60.57微秒；

总结与展望

总结

1.系统设计并实现了ISAMatch模型，使之可以支持支持混合多种指令集架构的任务调度场景，包括多种RISC-V扩展指令集和交叉编译场景。

展望

1. 异构ISA指令集架构芯片存在性能差异，即不同指令集架构的单核计算性能不同，ISAMatch模型在调度过程中未能充分区分计算核的性能差异，应对计算密集型作业无法实现计算资源负载均衡。因此，在后续的工作中，可以先通过实验论证不同指令集架构芯片单核的性能差异，并以此指导调度过程，实现更为全面的资源调度策略。
2. ISAMatch模型只面向于调度Kubernetes中的节点资源，无法应对多种第三方设备资源，如GPU、物联网设备等。后续工作可以在Kubernetes中继续引入边缘计算调度策略，丰富ISAMatch模型的调度范围。
3. 对于指令集的分配没有对应的Benchmark，未来可在此论文的基础上形成标准的度量Benchmark，作为成果在行业内发布。



THANKS