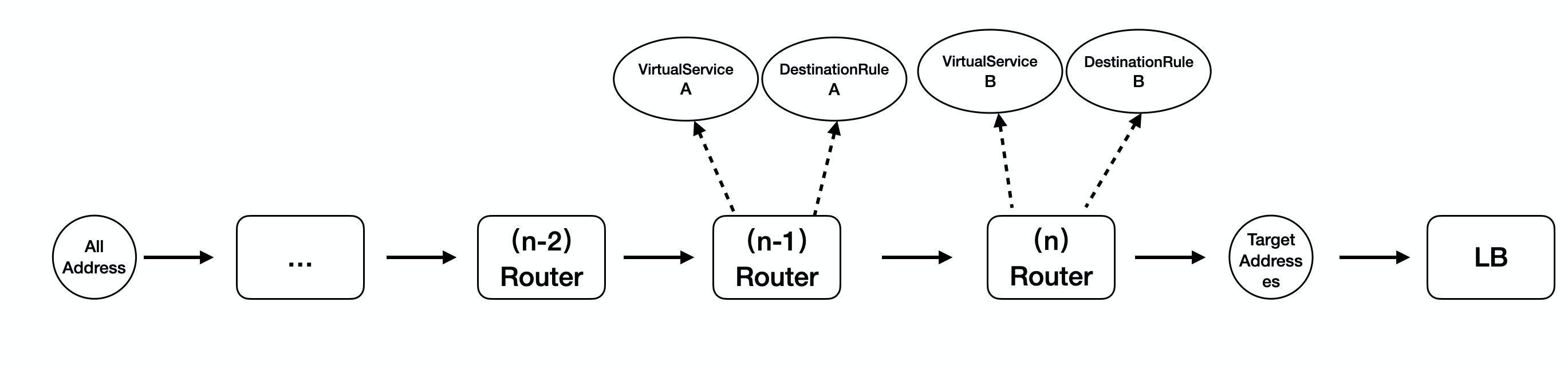
# **https://dubbo.apache.org/zh/docs/references/routers/**

# **路由规则**

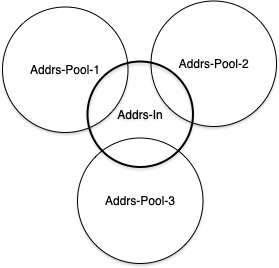
Dubbo支持的路由类型及配合方式

### 基本思想

基于路由链，采用Pipeline的处理方式，如下图所示：



可以把路由链的逻辑简单的理解为 target = rn(…r3(r2(r1(src))))。对于每一个 router 内部的逻辑，可以抽象为输入地址 addrs-in 与 router 中按全量地址 addrs-all 实现切分好的 n 个互不相交的地址池 addrs-pool-1 … addrs-pool-n 按实现定义好的规则取交集作为输出 addrs-out。以此类推，完成整个路由链的计算。



另外一方面，如果 router(n) 需要执行 fallback 逻辑的时候，那么需要经过 router(n) 就应该决定好 fallback 逻辑

### fallback 处理原则

由于多个 router 之间多个条件组件之后，很容易出现地址被筛选为空的情况，那么我们需要针对这情况进行 fallback 处理，保证业务在正确性的前提下，能够顺利找到有效地址。

首先我们看一下以下规则

apiVersion: service.dubbo.apache.org/v1alpha1kind: VirtualServicemetadata:

name: demo-routespec:

hosts:

- demo // 统一定义为应用名

dubbo:

- service:

- exact: com.taobao.hsf.demoService:1.0.0

- exact: com.taobao.hsf.demoService:2.0.0

routedetail:

- name: sayHello-String-method-route

match:

- method:

name\_match:

exact: "sayHello"

.....

argp:

- string

route:

- destination:

host: demo

subset: v1

fallback:

destination:

host: demo

subset: v2

fallback:

destination:

host: demo

subset: v3

- name: sayHello-method-route

match:

- method:

name\_match:

exact: "s-method"

route:

- destination:

host: demo

subset: v2

fallback:

destination:

host: demo

subset: v3

- name: interface-route

route:

- destination:

host: demo

subset: v3

- service:

....

---apiVersion: service.dubbo.apache.org/v1alpha1kind: DestinationRulemetadata:

name: demo-routespec:

host: demo

subsets:

- name: v1

labels:

sigma.ali/mg: v1-host

- name: v2

labels:

sigma.ali/mg: v2-host

- name: v3

labels:

sigma.ali/mg: v3-host

我们以脚本路由为例，这个脚本路由的匹配条件是遵循一个原则的，就是匹配的范围是从精确到广泛的一个过程，在这个示例来说，就是 sayHello(string)参数 -> sayHello 方法 -> 接口级路由 的一个匹配查找过程。

那么如果我们已经满足某个条件，但是选到的 subset 地址为空，我们将如何进行 fallback 处理呢？

以匹配 sayHello(string)参数 条件为例，我们选择到的是 v1 subset,如果是空，我们可以向上一级是寻找地址，也就是方法级去寻找地址，具体的配置为下

- name: sayHello-String-method-route

match:

- method:

name\_match:

exact: "sayHello"

.....

argp:

- string

route:

- destination:

host: demo

subset: v1

fallback:

destination:

host: demo

subset: v2

fallback:

destination:

host: demo

subset: v3

此时我们选到的地址是 v2 方法级地址，如果 v2 还是没有地址，根据规则的定义，我们是可以 fallback 到 v3 接口级。

假设我们有一个方法匹配时，如果没有地址，需要不进行 fallback，直接报错，我们可以这样配置

apiVersion: service.dubbo.apache.org/v1alpha1kind: VirtualServicemetadata:

name: demo-routespec:

hosts:

- demo // 统一定义为应用名

dubbo:

- service:

- exact: com.taobao.hsf.demoService:1.0.0

- exact: com.taobao.hsf.demoService:2.0.0

routedetail:

- name: sayHello-String-method-route

match:

- method:

name\_match:

exact: "sayHello"

.....

argp:

- string

route:

- destination:

host: demo

subset: v1

fallback:

destination:

host: demo

subset: v2

fallback:

destination:

host: demo

subset: v3

- name: sayHello-method-route

match:

- method:

name\_match:

exact: "s-method"

route:

- destination:

host: demo

subset: v2

fallback:

destination:

host: demo

subset: v3

- name: some-method-route

match:

- method:

name\_match:

exact: "some-method"

route:

- destination:

host: demo

subset: v4

- name: interface-route

route:

- destination:

host: demo

subset: v3

- service:

....

---apiVersion: service.dubbo.apache.org/v1alpha1kind: DestinationRulemetadata:

name: demo-routespec:

host: demo

subsets:

- name: v1

labels:

sigma.ali/mg: v1-host

- name: v2

labels:

sigma.ali/mg: v2-host

- name: v3

labels:

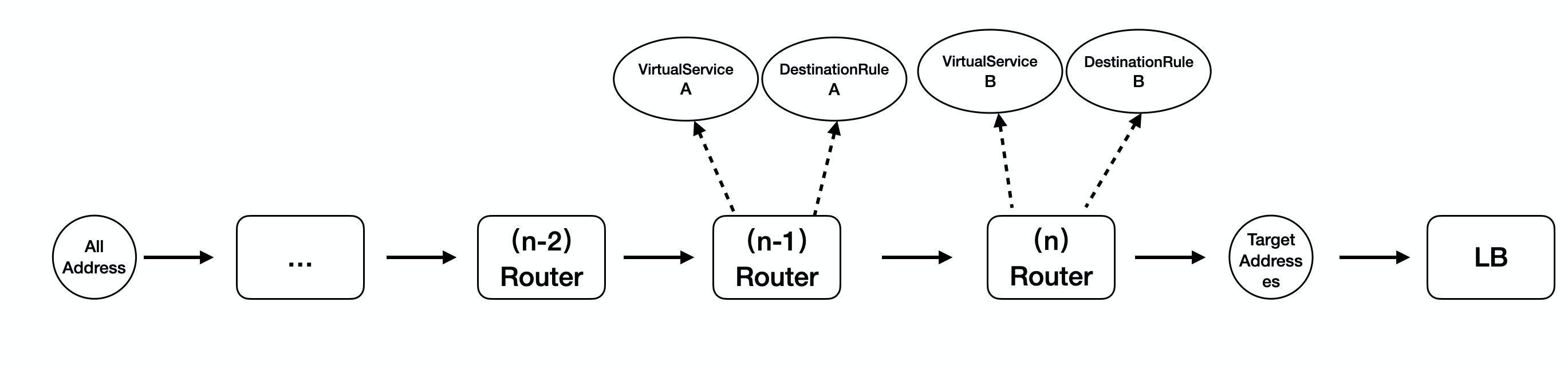
sigma.ali/mg: v3-host

从这个规则我们看出来匹配到 some-method 条件时对应的是 v4 subset，那么 v4 为空时，因为没有配置 fallback ，此时会直接报错

#### fallback 处理原则总结

* 我们应该在 VirtualService route 中配置好 Destination 的 fallback 处理逻辑
* 在 fallback subset 时，如果对应的 subset 也配置有 fallback subset 时，也应递归处理；fallback subset 之间的关系也应该是从具体到广泛
* 我们在编写匹配条件时，应该遵循从 具体条件到广泛条件 的原则

### RouteChain 的组装模式 (目前未实现)



我们看到上面的图，在路由的过程当中，我们是 Pipeline 的处理方式，Pipeline 的 Router 节点存在顺序，并且每个 Router 都有一个唯一对应的 VirtualService 和 ****多个**** 相应的 DestinationRule 进行描述。

以 Nacos 上存着的路由规则配置为例，配置的格式如下：

DataId: Demo.rule.yamlGROUP: HSF

content:

VirtualService A

---

DestinationRule A1

---

DestinationRule A2

---

VirtualService B

---

DestinationRule B

---

VirtualService C

---

DestinationRule C

---

...

VirtualService A 与 DestinationRule A1 、DestinationRule A2 组成一个 Router A，VirtualService B 与 DestinationRule B 组成 Router B,以此类推，完成整个 router 链的组装。

##### **[VirtualService](https://dubbo.apache.org/zh/docs/v3.0/references/routers/virtualservice/)**

入站流量的规则

##### **[DestinationRule](https://dubbo.apache.org/zh/docs/v3.0/references/routers/destination-rule/)**

目标地址规则