FlowBench：A flow table benchmark

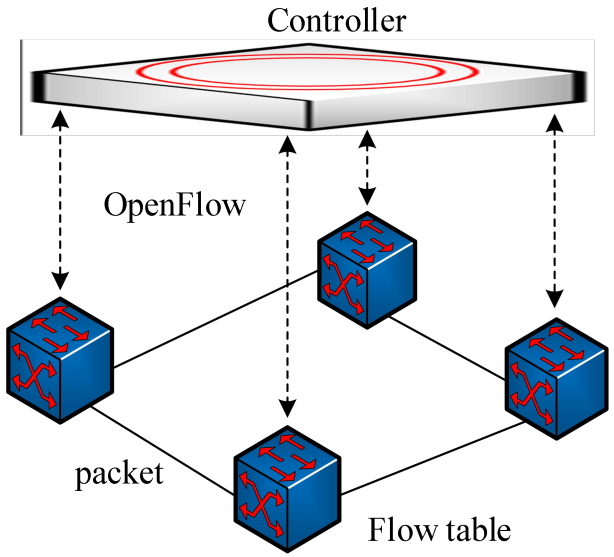
outline

- 问题背景
- 研究现状
- 研究目标
- 技术路线
  - ✓ 规则构造
  - ✓ 流量生成
- 性能评价

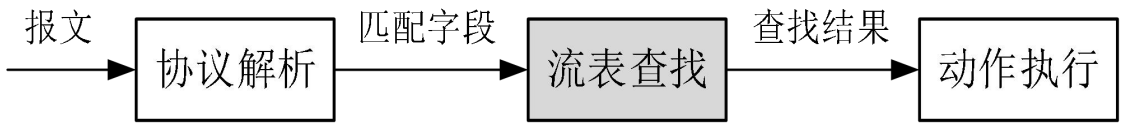
问题背景-Flow table

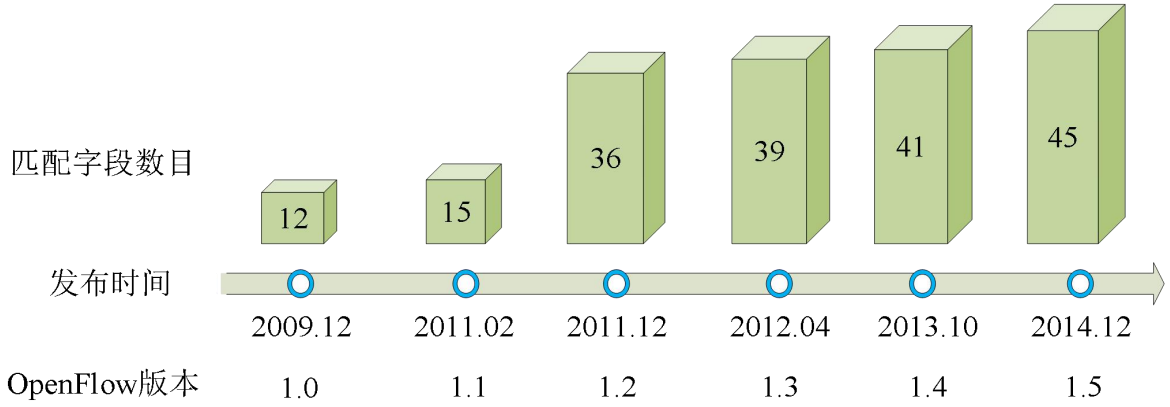
关键词：

1. 流量
2. 规则
3. 最佳匹配



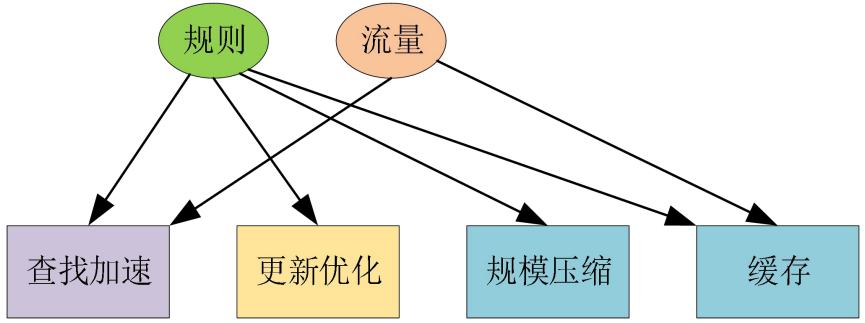
匹配域变多！

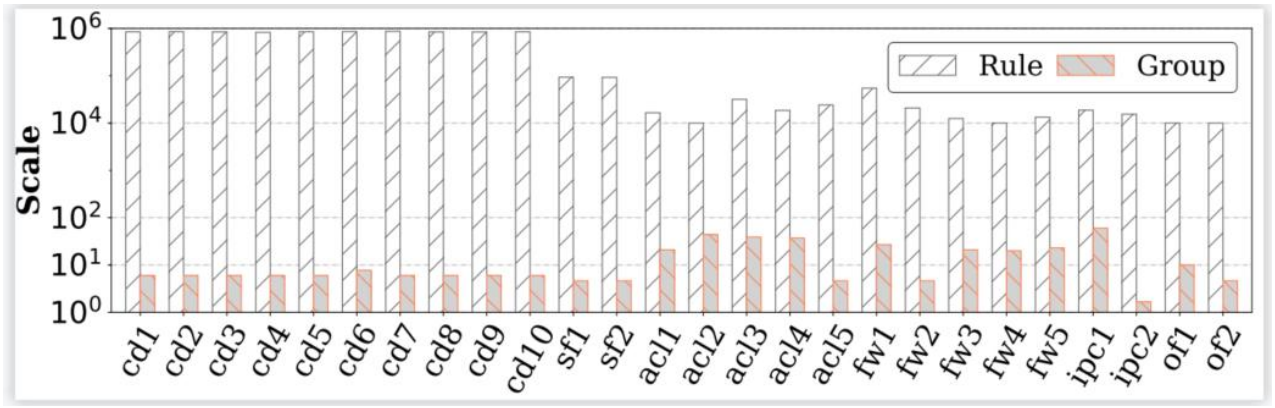


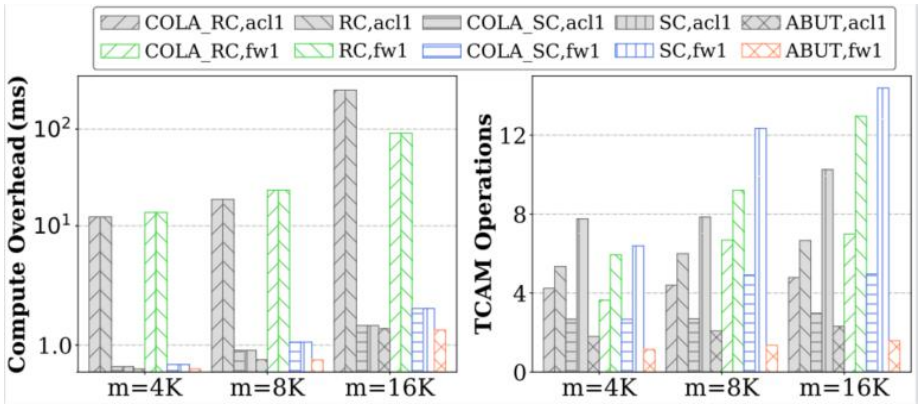


问题背景——Flow table的重要性

- 很多研究领域对于规则/流量
  - 需要，才能进行测试
  - 依赖，性能直接影响
    - ✓ 规则: 数量, 匹配域数量, 深度, 依赖度
    - ✓ 流量: 数量, 局部性, 速率

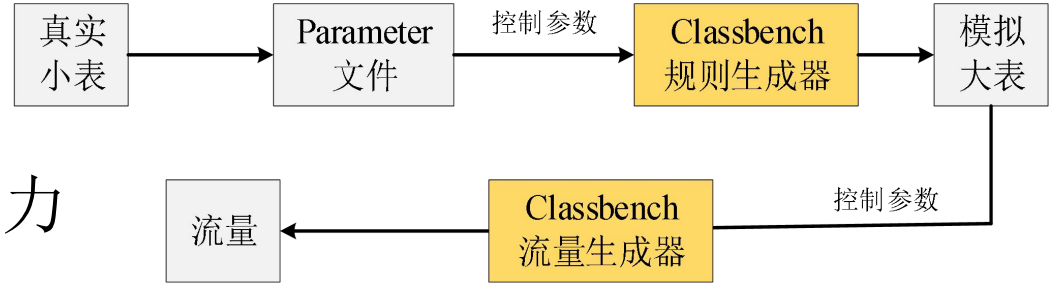






研究现状——获得Flow Table时遇到问题

- **真实**
  - 无法获得——隐私安全原因
  - 有限获得[1,2,3]——代表性存疑，算法泛化能力
- **虚拟**
  - 自己构造[4,5,6]——别人无法使用，说服力差
  - 工具生成[7,8,9]——Classbench，事实标准
    - ✓ 灵活性低：规则有且仅有5个固定匹配域，手动转成LPM
    - ✓ 控制力低：
      - ✓ 规则部分控制参数无效
      - ✓ overlap degree无法控制，规则之间毫无关系
      - ✓ 流量匹配无法保证



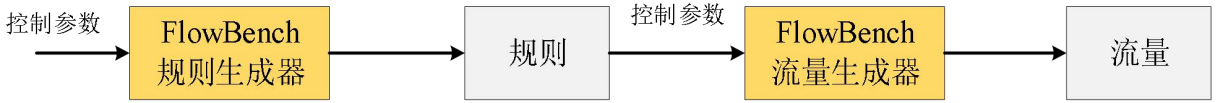
[1] CacheFlow—SOSR'16, HotSDN'14
[2] TCAM razor—INFOCOM'10
[3] CutSplit—INFOCOM'18
[4] PipeCache—Electronics'20
[5] MixedCache—CISAI'21
[6] FastRule—JSAC'19
[7] TupleMerge—TON'19
[8] MultilayerTuple—IFIP Networking'21
[9] FastRule—JSAC'19

研究目标——FlowBench

- **灵活性**
  - 参数可精确指定：n, m, k
  - 参数可模糊指定：
    - ✓ 规则：p, h_G, α_G, μ_R
    - ✓ 流量：L_F^s, L_F^t, γ_F
- **灵敏性**
  - 参数指定时，能起到效果
- **一致性**
  - 参数制定后，生成大致相似/相同

$$\begin{cases} R = (r_1, r_2, \ldots, r_n) \\ r = (f_1, f_2, \ldots, f_k) \\ G = \{(r_i \to r_j) | i, j < n\} \end{cases}$$

$$\begin{cases} T = (p_1, p_2, \ldots, p_m) \\ p = (h_1, h_2, \ldots, h_k) \\ \mathrm{F} = \mathrm{Set(T)} \end{cases}$$



h_G—规则图高度, α_G—规则图边数, μ_R—规则精确度

L_F^s—流量空间局部性, L_F^t—流量时间局部性, γ_F—流量完整性

规则生成器——方案一

- **随机DAG法—rDAG**
  - 好处：易于生成指定特征的随机DAG
  - 坏处：
    - 无法考虑规则属性
      - 很难，DAG和规则表可能性太多
      - 有时甚至无法根据DAG反推具体化规则
    - 随机性过大，一致性弱
    - 若需要n条k个LPM匹配域(宽度为$L_{1\sim k}$)的流表:

$$\begin{cases} n_r = \prod_{i=1}^{k}\left(\sum_{j=0}^{L_i} 2^j\right) = \prod_{i=1}^{k}(2^{L_i}-2) \\ n_R = C_{n_r}^{n} \\ n_G = \prod_{i=1}^{n}(2^{i-1}) = 2^{\sum_{i=1}^{n}(i-1)} = 2^{\frac{n(n-1)}{2}} \end{cases}$$

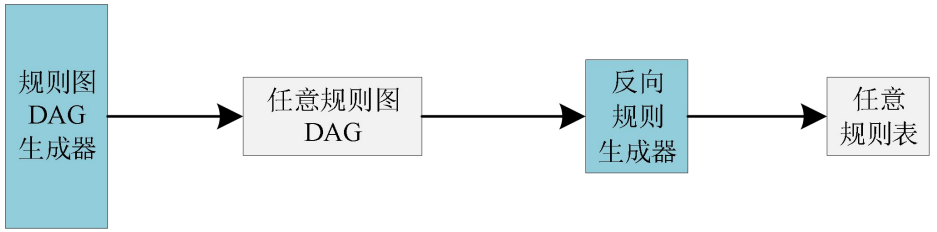


图2 某些边的缺失（A 优先级最低，B 优先级更高，D 优先级最高）

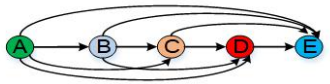


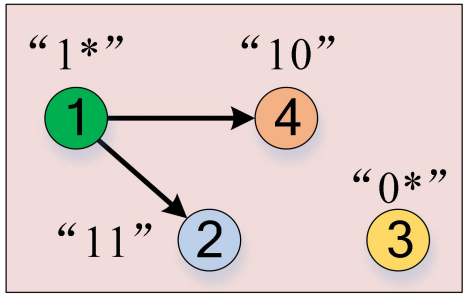
图3 过长的指向链

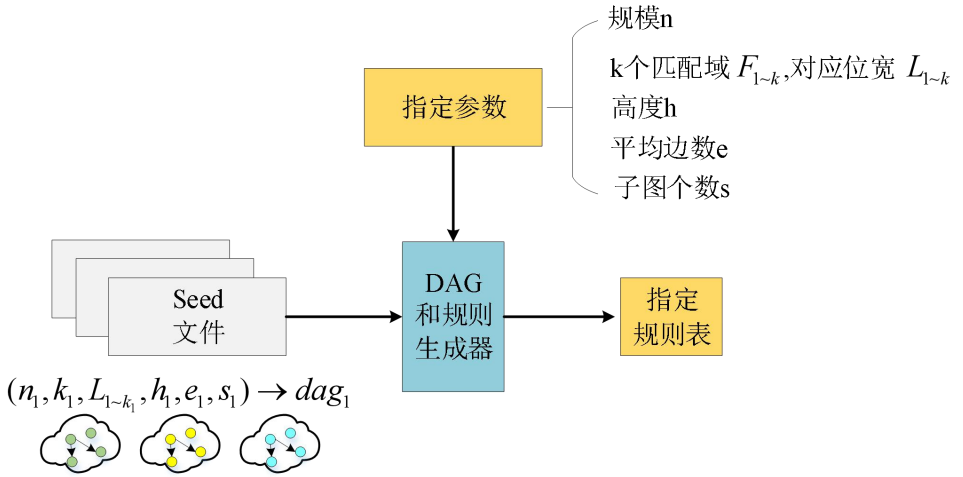
规则生成器——方案二

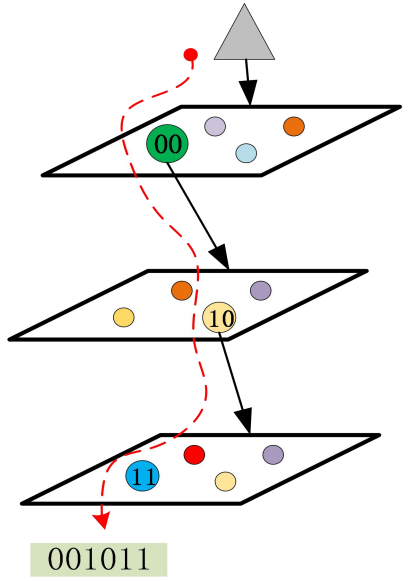
◘ 分层DAG法—L-DAG

- seed中存储所有可能子图(subdag)
  - ✓ 属性不同——一致性/灵活性/灵敏性

- 深度优先搜索算法DFSearch
  - ✓ 向下递归——规则具体化



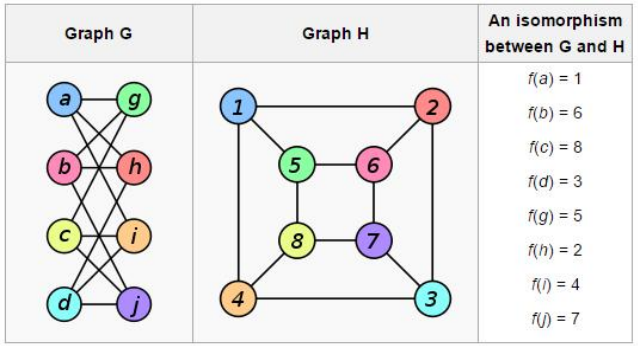


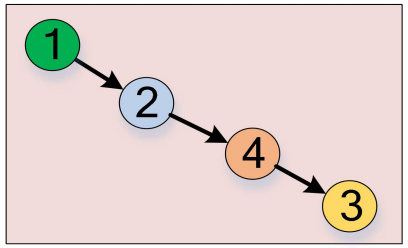


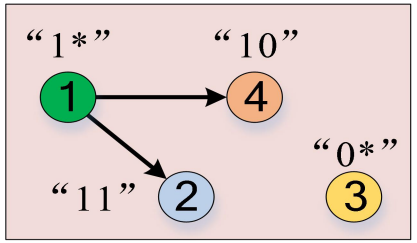
规则生成器——subdag的规模

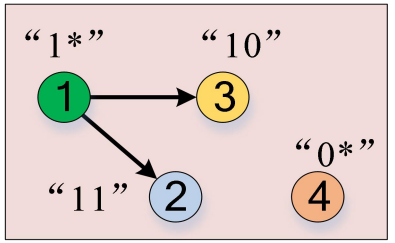
- subdag的数量和规模呈指数关系
  - 存储空间
    - n=4时，n_G=64
    - n=6时，n_G=32768
    - 同构消除
  - 搜索时间
    - 随机选择
    - 偏好选择

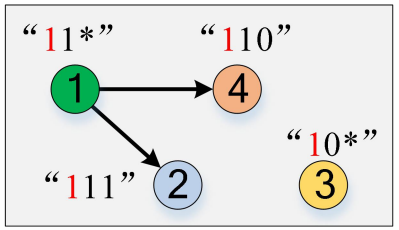
$$n_G = \prod_{i=1}^{n} (2^{i-1}) = 2^{\sum_{i=1}^{n}(i-1)} = 2^{\frac{n(n-1)}{2}}$$







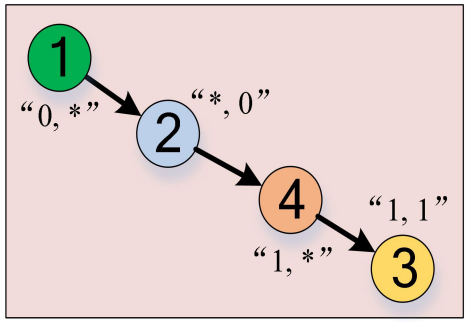


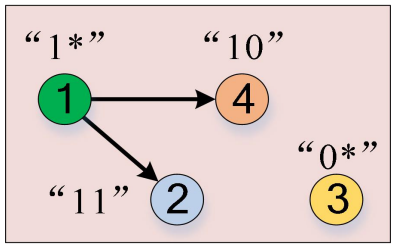


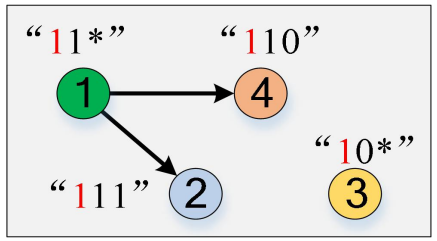
图同构问题，未解难题

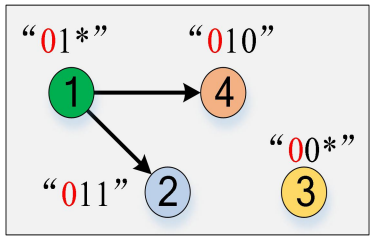
规则生成器——subdag具体化

- 每个subdag都对应无数种具体化方式S
  - 若n_field=a, n_width =b时, 设定 n_R 为 c
    - ✓ n_field=a, n_width =b+1 $\Longrightarrow n\_R \geq 2 * c$
    - ✓ n_field=a+1, n_width =b $\Longrightarrow n\_R \geq a * c$
- 存储形式： $Entry = \{< subdag, specializations, attribute >\}$
  - subdag 作为“主键” $\Longrightarrow$ 唯一标识符
  - specializations $=<$ string, n_field, n_width $>$ $\Longrightarrow$ DFS搜索
  - attribute=<n_edges, n_depth> $\Longrightarrow$ 偏好选择





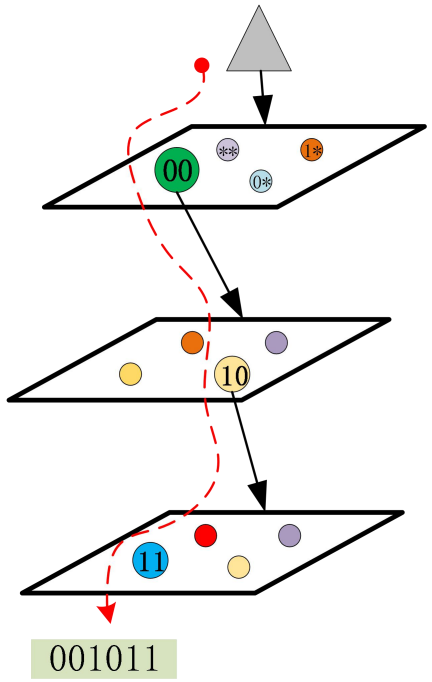


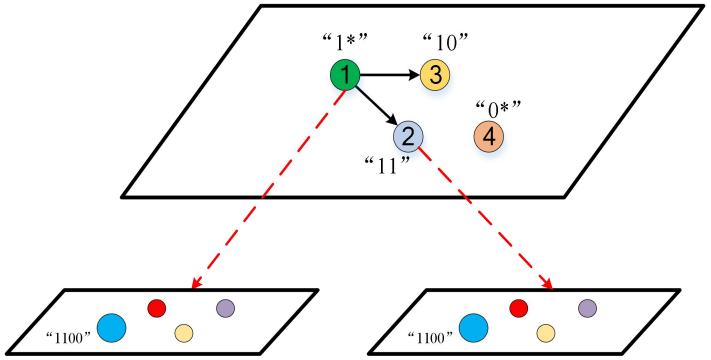


n_width变多， n_R爆炸

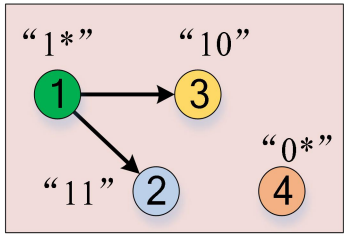
规则生成器——规则重复性

- DFSearch搜索时，生成重复规则
  - subdag内不同节点，向下搜索
    - ✓ 例如，1*和 11后续搜到相同的前缀
- 方案：
  - *视为具体化终结标识符，“遇*辄止”
- 坏处：
  - 生成规则数量过少





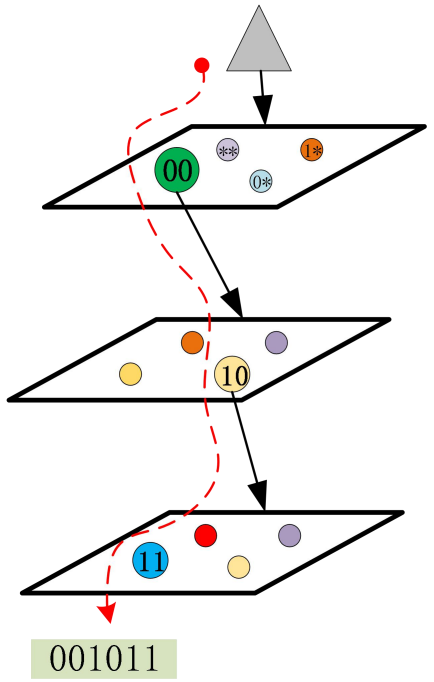
规则重复

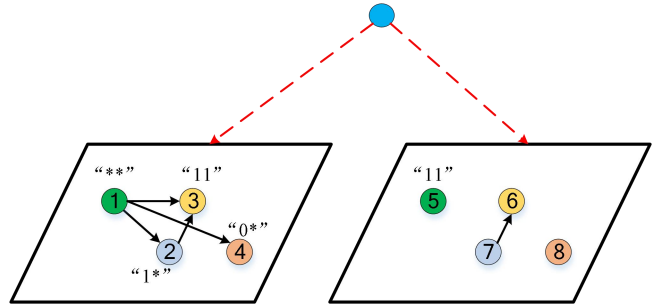


遇*辄止

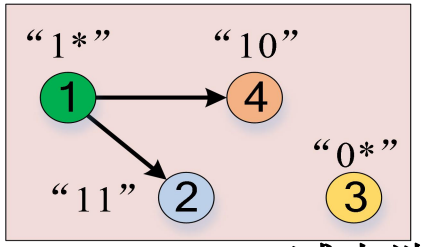
规则生成器——规则有限性

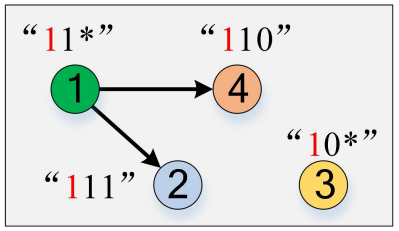
- DFSearch搜索时，生成规则数量有限
  - 纵向，深度有限
    - ✓ 规则位宽有限
  - 横向，广度有限
    - ✓ 遇*中断，防止重复
- 扩大搜索范围
  - 纵向
    - ✓ 尽可能少地消耗位宽，提高递归深度
  - 横向
    - ✓ 尝试不同subdag，但会造成重复
    - ✓ isolate-prefix：<specification, IPs>
      - 单匹配域内，IPs内必然精确值
      - 两个匹配域，暴力搜索



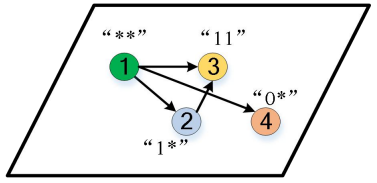


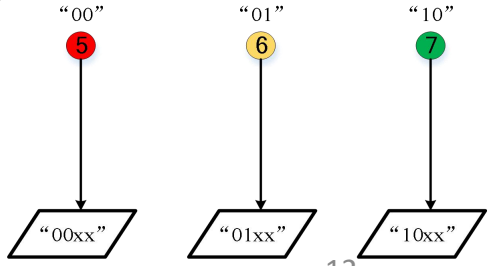
subdag重叠





减少消耗位宽

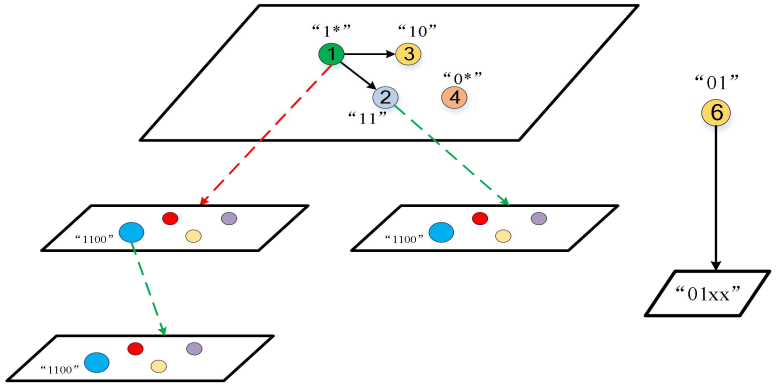




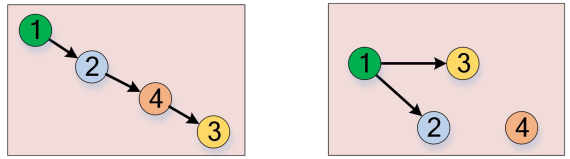
孤立化旁路

规则生成器——DAG偏好性

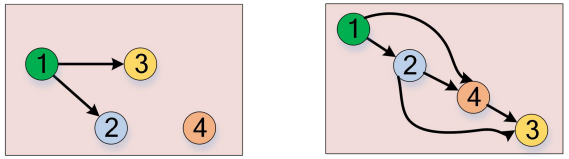
- 高度h
  - 横向：subdag的选取
  - 纵向：递归深度
  - 混合：DFS vs BFS
- 边数e
  - 横向：subdag的选取
  - 纵向：递归深度
  - 混合：DFS vs BFS
- 精确度
  - 混合：DFS vs BFS



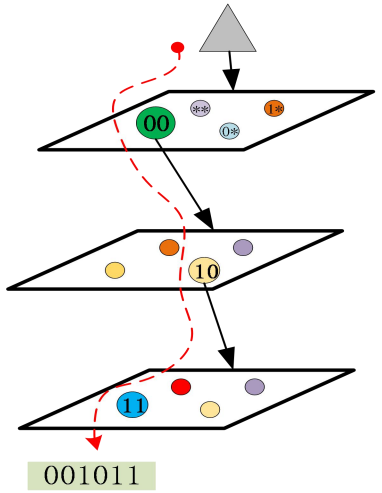
DFS vs BFS



边数一样，高度不同

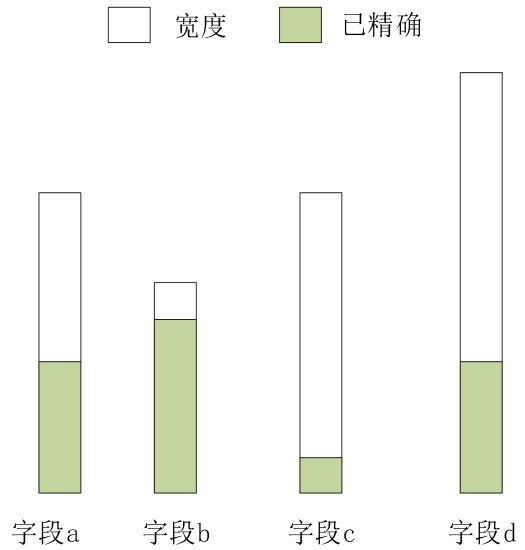


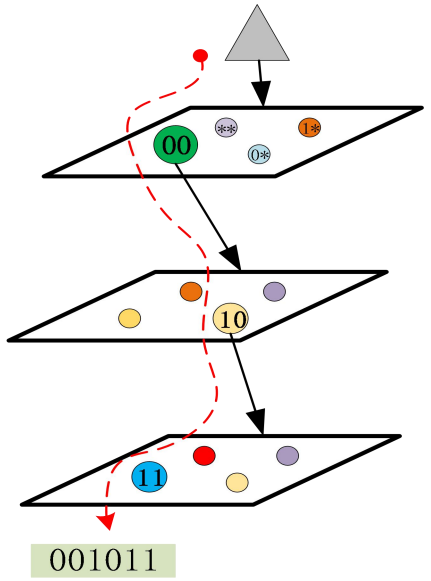
高度一样，边数不同



规则生成器——规则偏好性

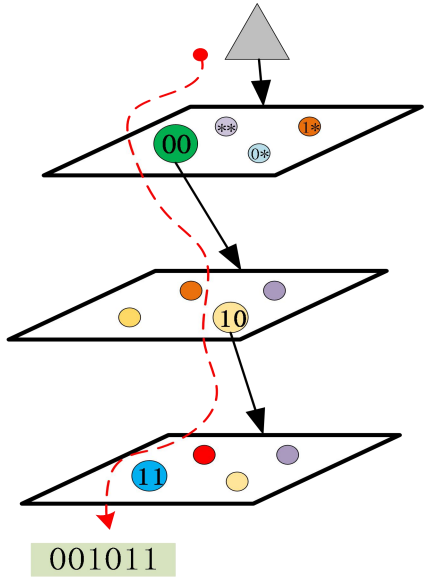
- 匹配域
  - 个数
  - 宽度
  - 精确度
    - 均匀
    - 概率分布

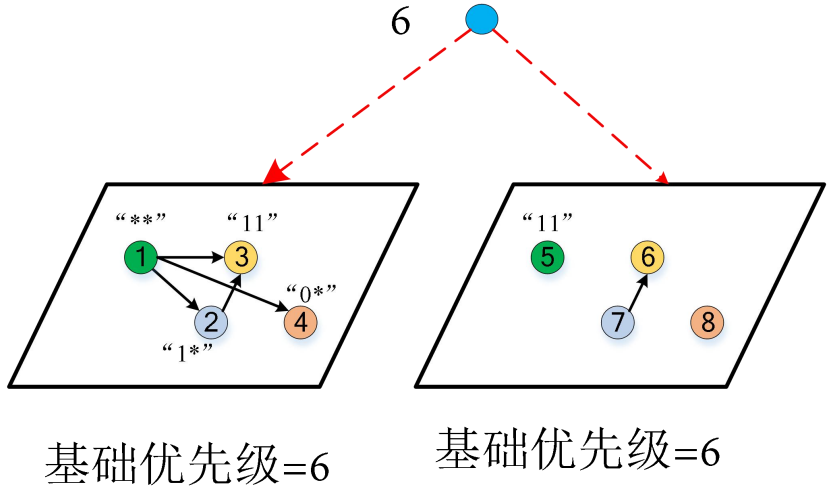




规则生成器——规则优先级

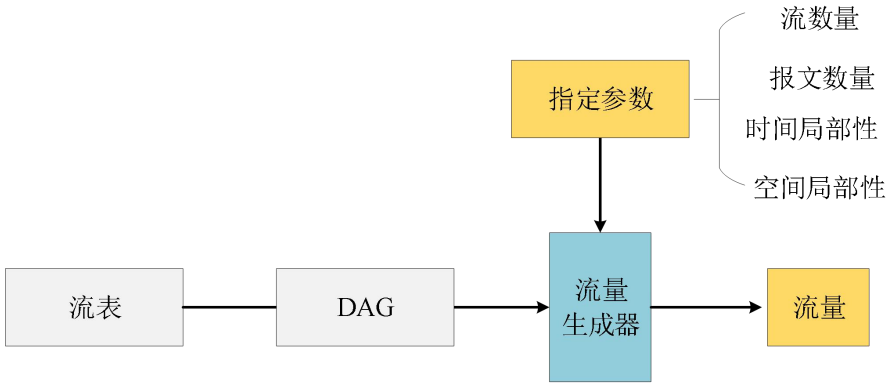
- 最少数目
  - 基础优先级+内部优先级
- 最多数目
  - 全局计数器

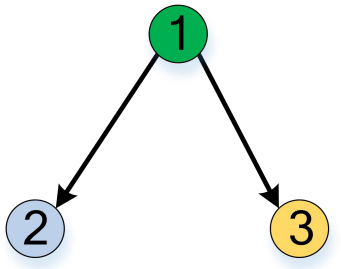


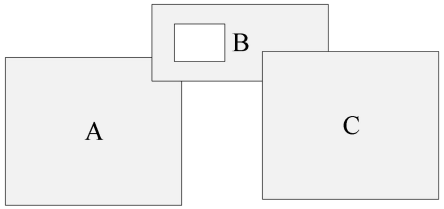


FlowBench-流量生成器

- 生成DAG
- 每个规则至少生成一个最佳匹配流
  - Isolate space
- 流的时间局部性
  - 每个流的报文数量
- 流的空间局部性
  - 平均每个规则最佳匹配的流的数目

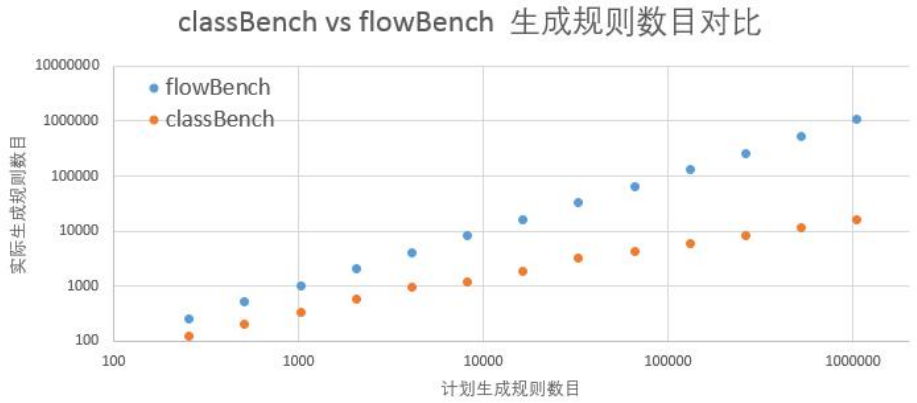


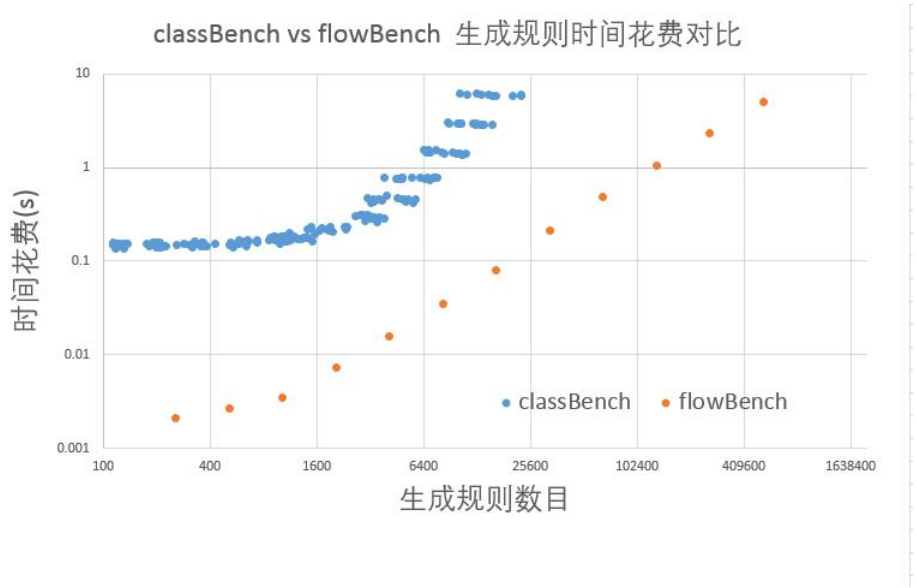




实验结果

- 灵活性
  - 任意匹配域个数
  - 任意匹配宽度
- 灵敏性
  - 实际生成数目
- 高速度





总结计划-FlowBench

- 设计了一个流表和流量生成器FlowBench
- FlowBench具有灵活性
  - 流表
    - 匹配域个数
    - 匹配域宽度
    - 规则数目
    - 规则依赖程度
  - 流量
    - 流和报文数目
    - 时间局部性
    - 空间局部性
    - 完备性
- FlowBench具有灵敏性
  - 按照参数进行控制
    - 显而易见的控制效果
- FlowBench具有高速度
  - 和classbench比较

- 计划投稿ICDCS 2022，1月中旬左右
  - 优化规则生成器（3个星期左右）
    - 偏好程度等
    - 增加精确匹配类型
  - 完成流量生成器（一个月左右）
    - 时间局部性
    - 空间局部性
    - 完备性
  - 写论文、画图等