



## **R2-D2: Filter Rule set Decomposition and Distribution in Software Defined Networks**

Ahmad Abboud, Rémi Garcia, Abdelkader Lahmadi, Michaël Rusinowitch,  
Adel Bouhoula

### ► **To cite this version:**

Ahmad Abboud, Rémi Garcia, Abdelkader Lahmadi, Michaël Rusinowitch, Adel Bouhoula. R2-D2: Filter Rule set Decomposition and Distribution in Software Defined Networks. CNSM 2020 - 16th International Conference on Network and Service Management, Nov 2020, Izmir/Virtual, Turkey. hal-03036292

**HAL Id: hal-03036292**

**<https://hal.inria.fr/hal-03036292>**

Submitted on 2 Dec 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

## Background and Motivation

### Context

- Large number of filtering rules due to the increasing number of hosts and applications.
- Increase in number of attacks that affect entries in access-control lists (ACL).
- Expensive and power-hungry ternary content-addressable memory (TCAM).

### Research question

How to decompose and distribute filtering rules on a set of limited size switch tables?

### Overview of Longest prefix matching (LPM) representation

Rule	Address field	Action
1	0 0 * *	$A_1$
2	0 0 0 *	$A_2$

Table: Example of a rule set in a switch table.

If a switch receives a packet with 0001 as address

**Prioritized list strategy** : Rule 1 is first,  $A_1$  is applied.

**LPM strategy** : Rule 2 is most specific,  $A_2$  is applied.

## Rule Representation

- Single field filtering.
- Sufficient for blacklists.
- Rules represented in a binary tree.
- One rule at most on each tree node.

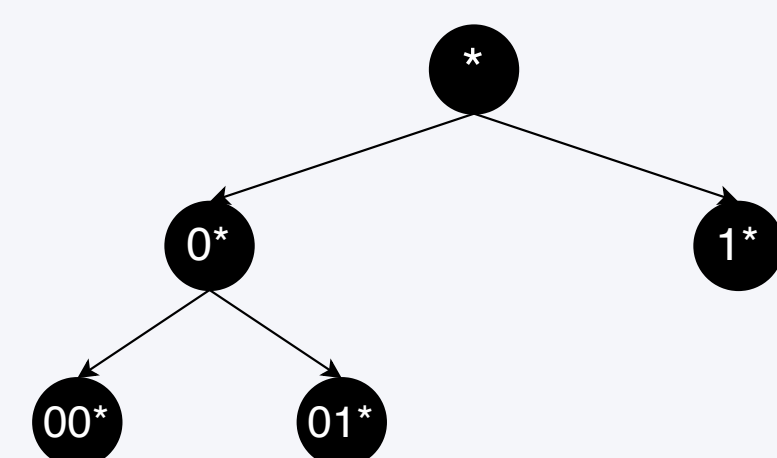
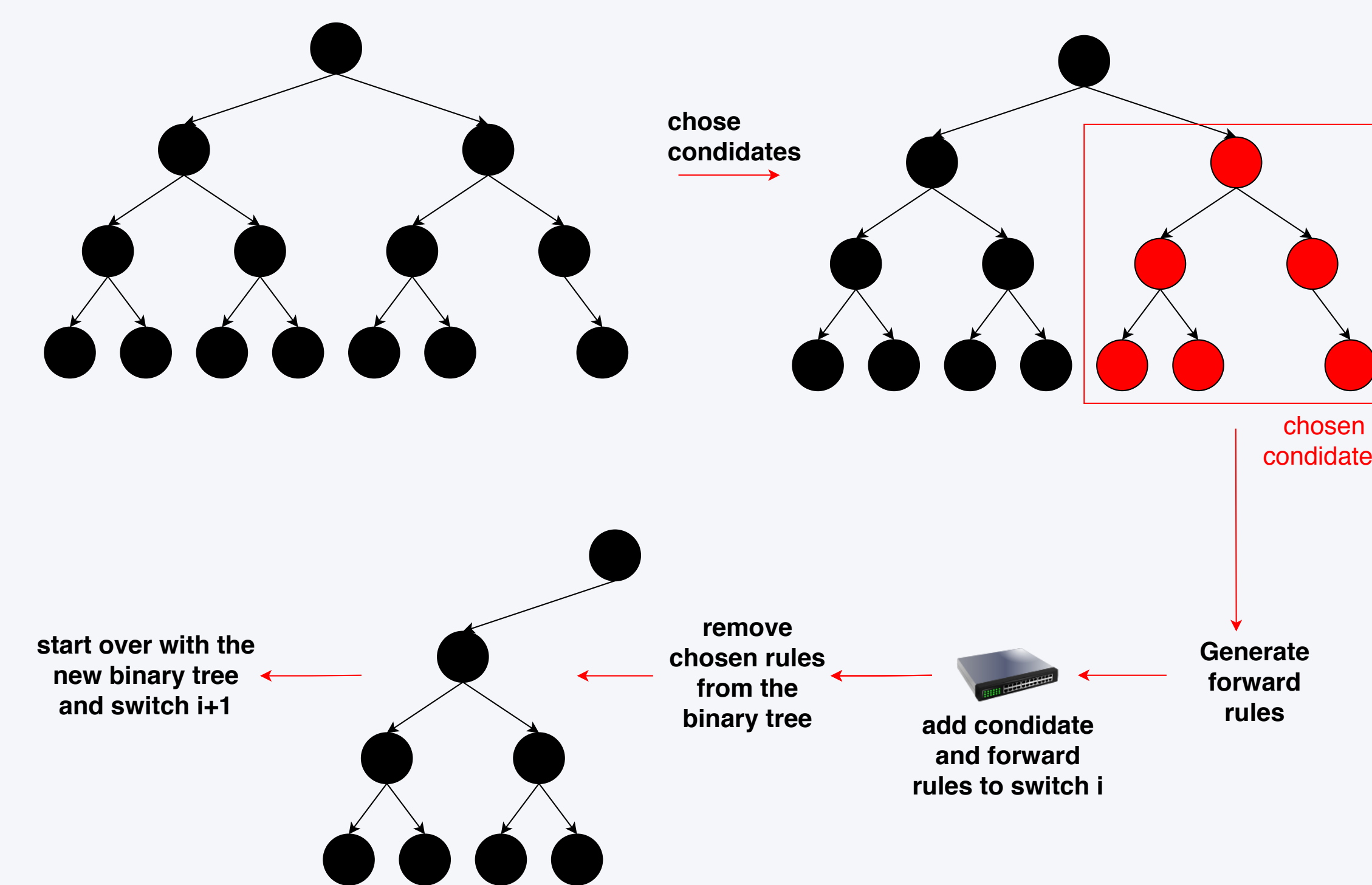


Figure: Compact representation of rules prefixes (00\*,01\*,0\*,1\*,0\*) in a binary tree.

## Decomposition Algorithm

- Input a set of rules and switches.
- Search the binary tree in order to find the best candidates.
- All rules present in the chosen node and all nodes below it, will be added to the switch.
- Minimize the number of generated rules by merging old forward rules with the ones from the best candidates.
- No rule duplication.



## Forward Rule Generation

- Forward rules avoid processing packets filtered by previous switches multiple times.
- Rules with deny action does not require a forward rule to be generated.

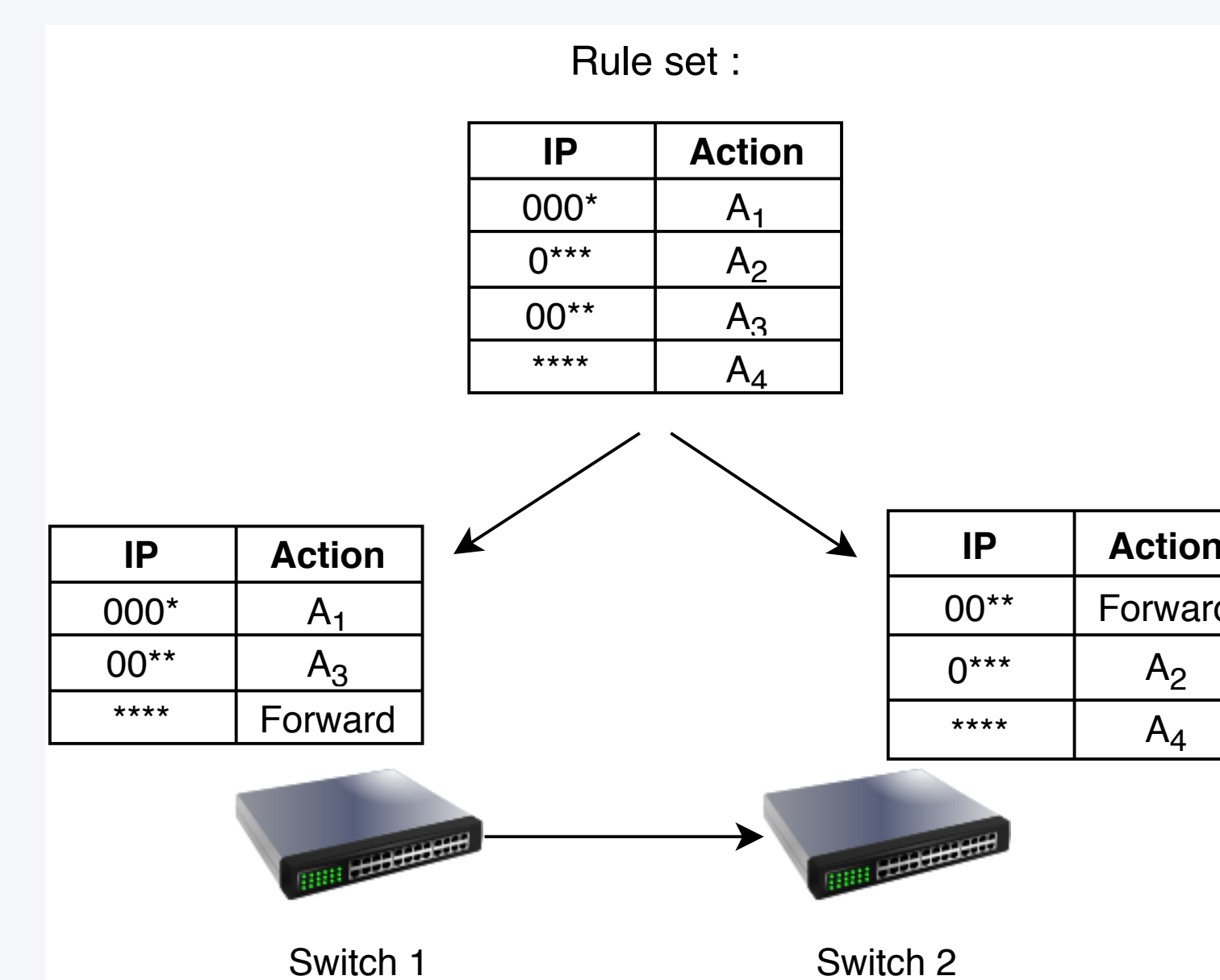


Figure: Illustration of forward rule generation with a rule set and two successive switches.

## Decomposition over a graph

- Series-parallel graphs.
- Build the binary tree from a parallel and series composition.
- Simplify the binary tree using S-components.
- Packets with different sources will be processed by the same rule table of an intersection switch.



Figure: S-component.

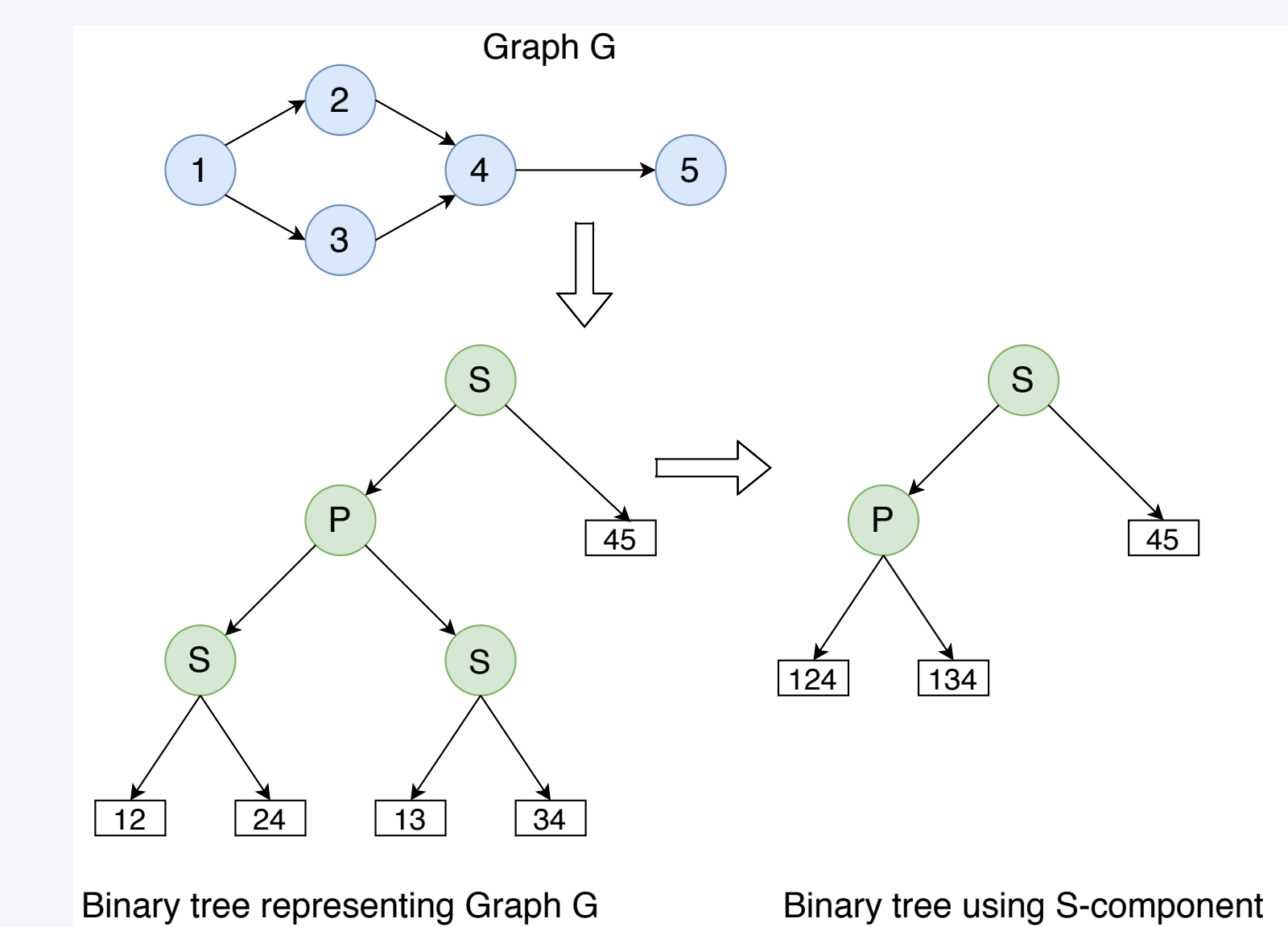
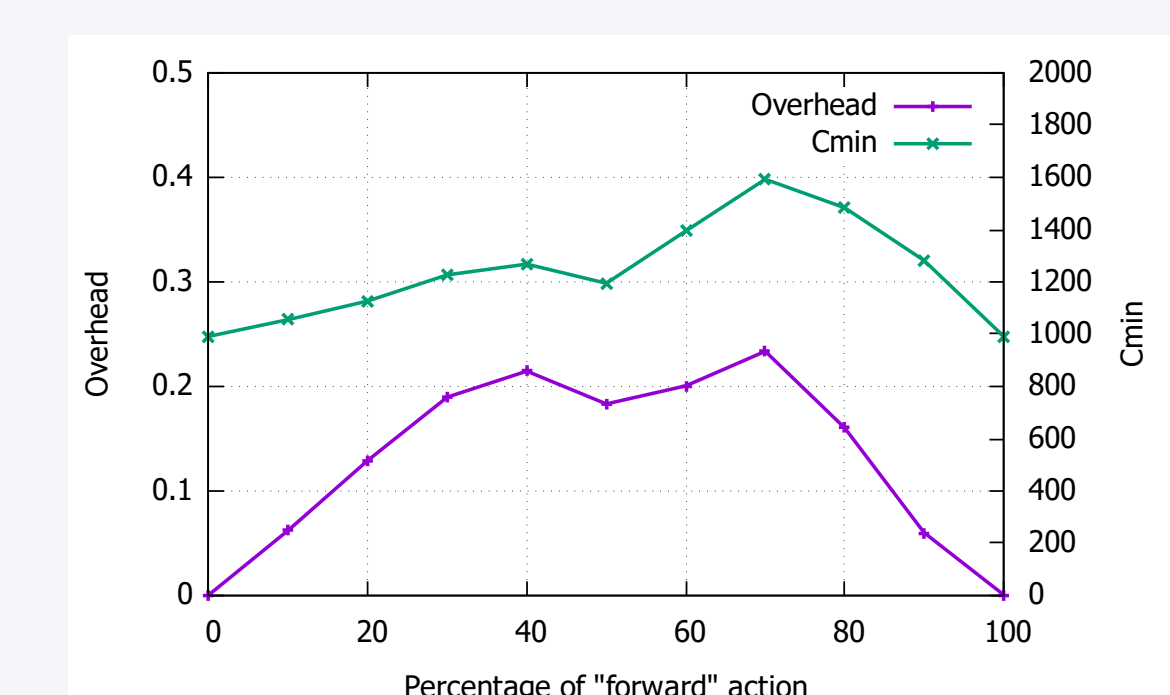


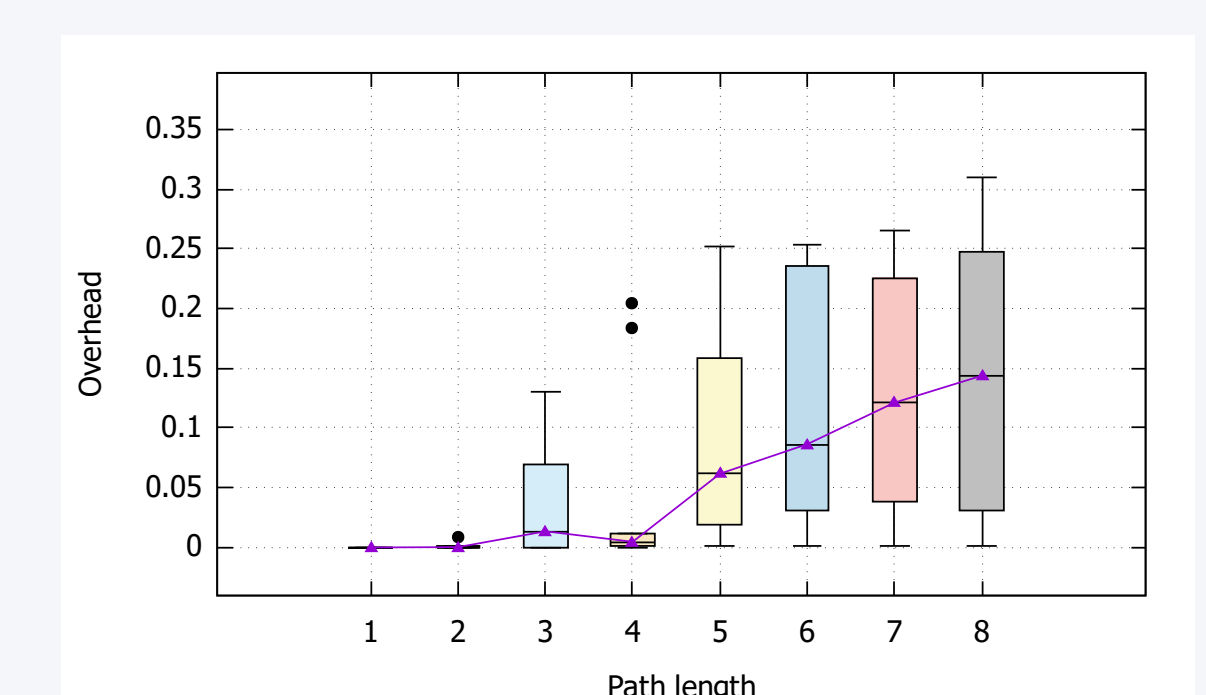
Figure: Tree representation of a series-parallel graph.

## Evaluation

- 12 sets of data generated using ClassBench.
- Percentage of forward action field between 0 and 100%.
- Rules with same action type have zero overhead.
- Around 15% overhead on 8 switches path length.



(a) Effect of action field on overhead  $OH$  and  $C_{min}$  using acl1 rule set.



(b) Rule space overhead while distributing a rule set with a 50% of rules having Forward actions.

### Acknowledgement

This work is supported by a CIFRE convention between the ANRT (National Association of Research and Technology) and the company NUMERYX Technologies.