# D-DIVE
# Design Diversity for Private Network model

Sumit Ahuja
Mehmet Guney
Marina Sahakyan
Deepaknath Tandur K.

# OUTLINE

- PROBLEM STATEMENT
- INTRODUCTION TO DESIGN DIVERSITY
- DESIGN METHODS
- CONSTRUCTION AND SHAPING METHODS
- CONCLUSIONS

# PROBLEM STATEMENT

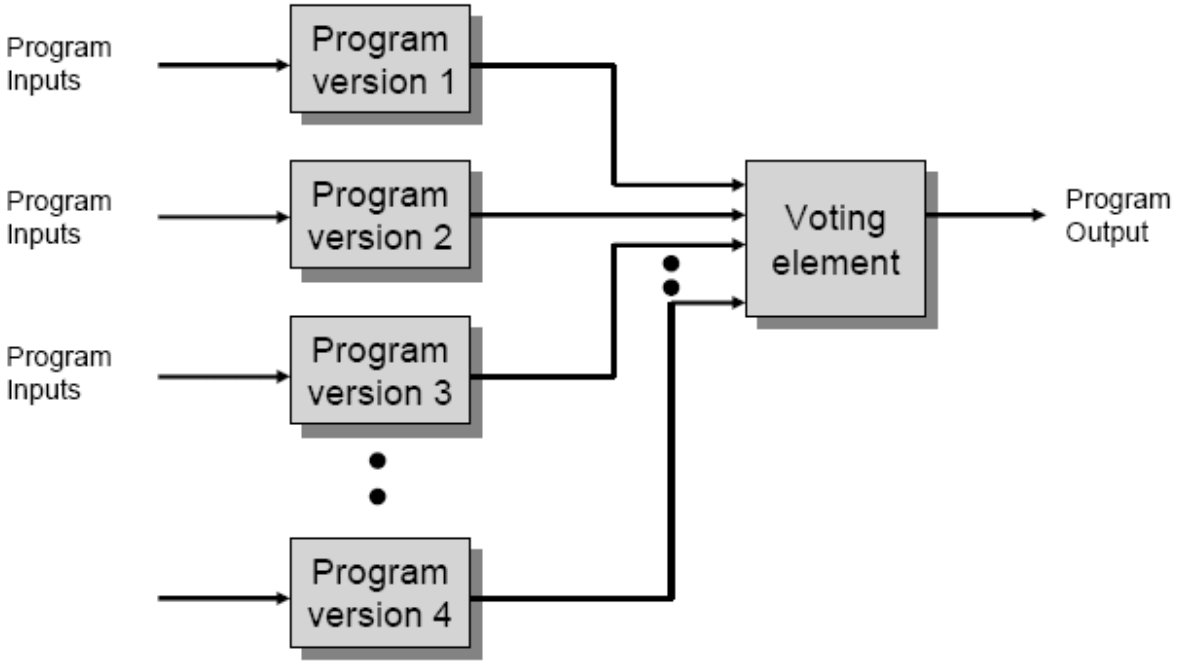
- Study Design Diversity methods and develope taxonomy for Secure communication.
- Develope and compare various design methods.
- Find out possible advantage of diversity methods.

# INTRODUCTION TO DESIGN DIVERSITY

- What is Design Diversity?
  - Design Diversity is used to tolerate faults in Software and Hardware.
- Techniques used for Design Diversity.
  - N version Programming.
  - Back to back testing.
  - Recovery Blocks.

# INTRODUCTION TO DESIGN DIVERSITY

- N-version Programming

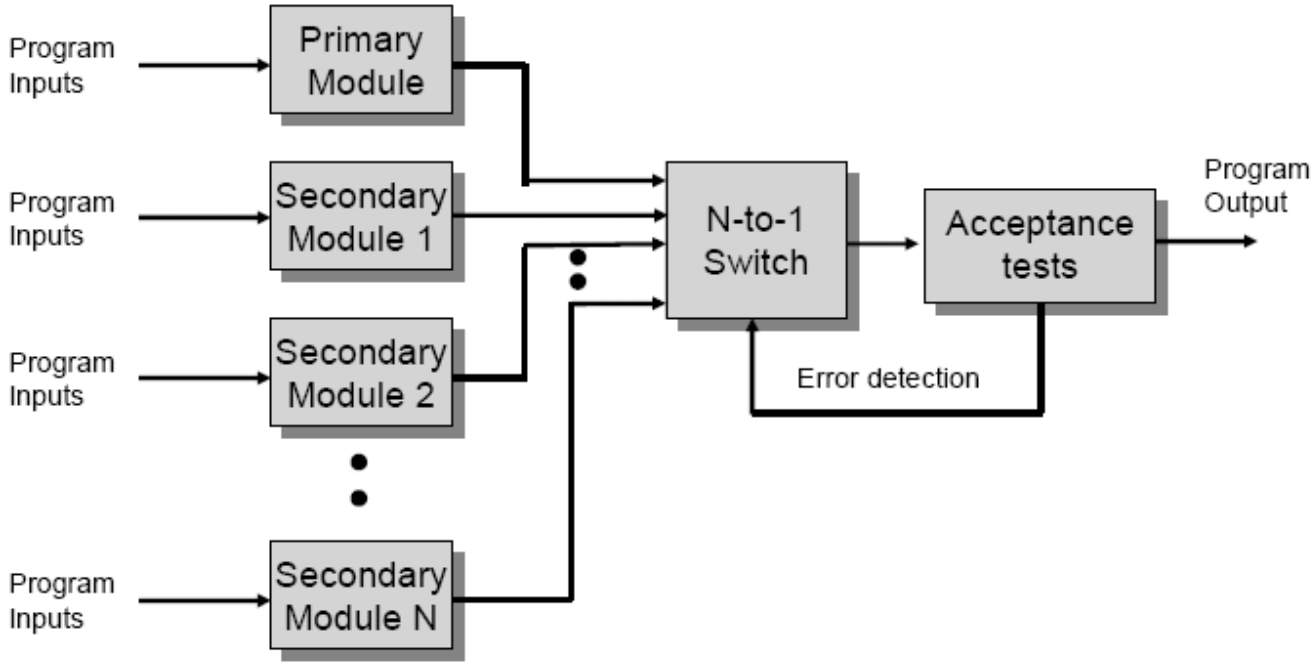


# INTRODUCTION TO DESIGN DIVERSITY

- Back to Back testing.
  - Complement to N-version Programming.
  - Error is calculated for every possible version.
  - Correction is applied if necessary.

# INTRODUCTION TO DESIGN DIVERSITY

- Recovery Blocks.



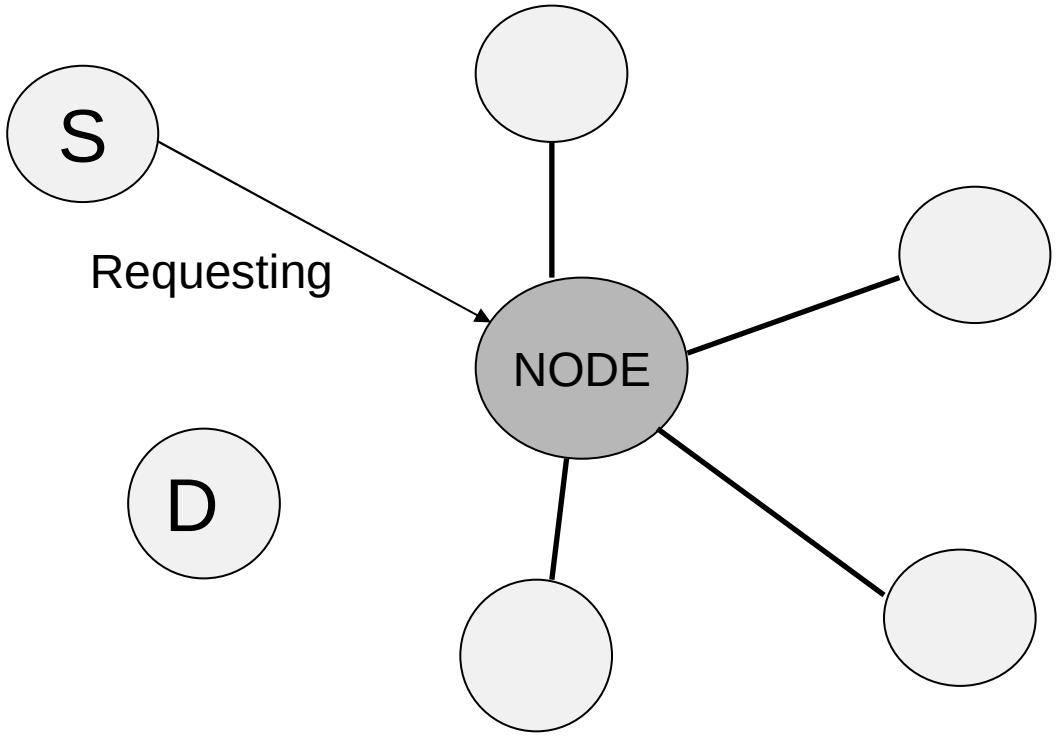
# Application – A simple communication network

**Specification:**

- Centre NODE is looking for new request of calls. When a new call request arrives. It confirms the ID of the source which should lie within a specific range.
- If the destination is available call can be established for insecure communication.
- For secure communication, the source public key is requested.
- All pre-existing communication routes via the NODE are not interrupted.
- Decision parameters to be included in every design:

  Node, Source, Destination, Communication Secure / Insecure, Key.

# A Simple Communication Network

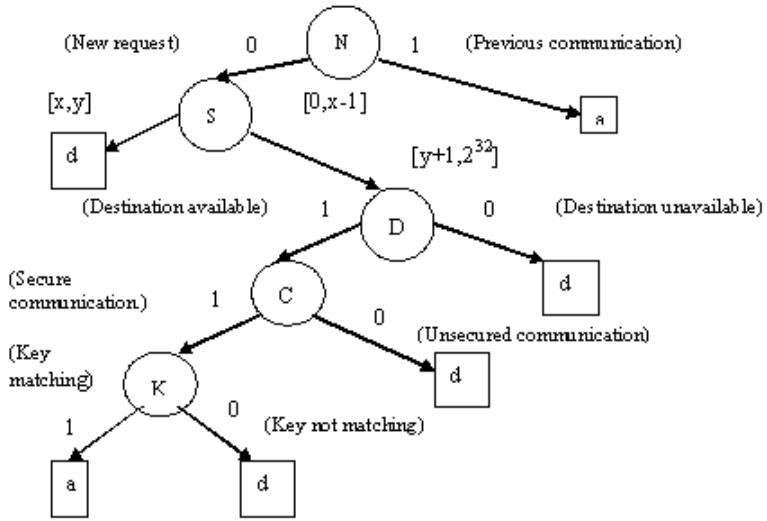


# DESIGN METHODS

- Rule based methods.
- Diagram based Methods.
- Flow chart based Methods.
- Many more……..

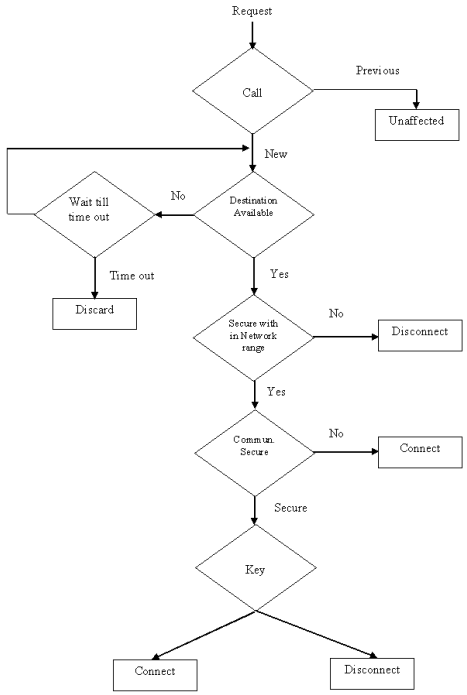
# DESIGN METHODS

- Team A
  - DIAGRAM BASED METHODS
    - Decision diagram is rooted, directed and acyclic graph.
    - Root of D.D has exactly one node with no incoming edge.
    - The node having no outgoing edge are terminal nodes and others are non-terminal node.
    - Example is shown in the coming slides.



# DESIGN METHODS

- Team B
  - Flow Chart based method
    - Example can be shown as.



# DESIGN METHODS

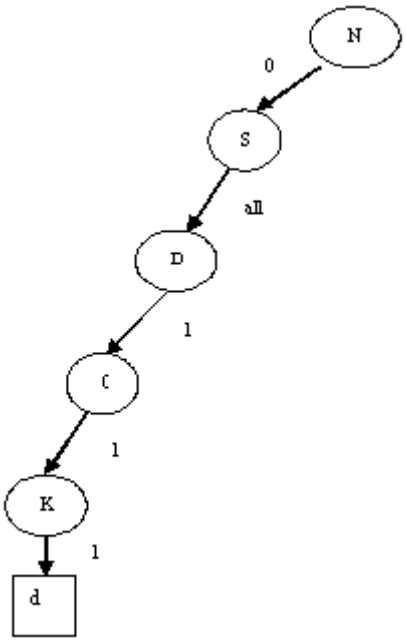
- Team C
  - RULE BASED METHODS
    - Rule represented as predicate ⟶ decision.
    - An example of rule is:
  - (N ∈ {0}) ^ (S ∈ {all}) ^ (D ∈ {1} ) ^ (C ∈ {1}) ^ (K ∈ {1}) -> a
  - (N ∈ {0}) ^ (S ∈ {all}) ^ (D ∈ {1} ) ^ (C ∈ {0}) ^ (K ∈ {1}) ->a
  - (N ∈ {0}) ^ (S ∈ {x,y}) ^ (D ∈ {all} ) ^ (C ∈ {all}) ^ (K ∈ {all}) ->d
  - (N ∈ {all}) ^ (S ∈ {all}) ^ (D ∈ {all} ) ^ (C ∈ {all}) ^ (K ∈ {all}) ->a

# Construction - 1 (Rule Based Model)

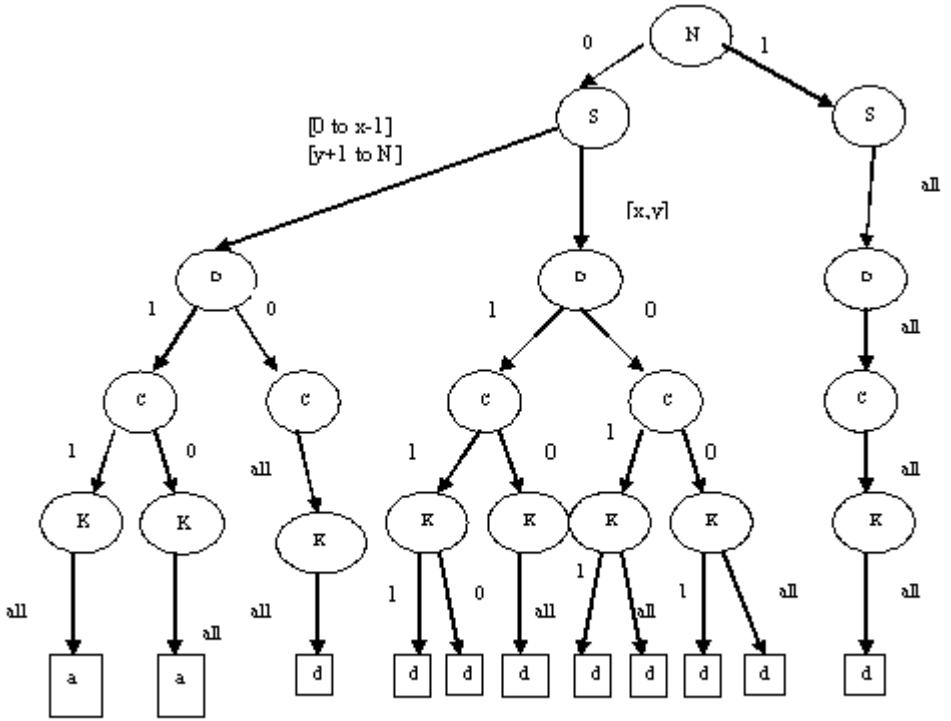
- Take one rule at a time and construct a partial Design Diagram (DD).
- To add a new rule, check:
  - S1 - ( $I(e_1)$ U $I(e_2)$ U ….U $I(e_k)$ ) ≠ 0, if not, then a new outgoing edge has to be added.
  - Pick a new rule:
    - S1 ∩ $I(e_j)$ = 0, skip the new edge $e_j$.
    - S1 ∩ $I(e_j)$ = $I(e_j)$: Add the new sub-graph rooted at this node.
    - S1 ∩ $I(e_j)$ ≠ $I(e_j)$ and S1 ∩ I ($e_j$) ≠ 0: Split edge e into e' with label $I(e_j)$ –S1 and e'' with label I(ej) ∩ S1.
    - **The resultant model is ordered.**

# Construction – 2 (Rule Based Model)

- Before Appending




- After Appending

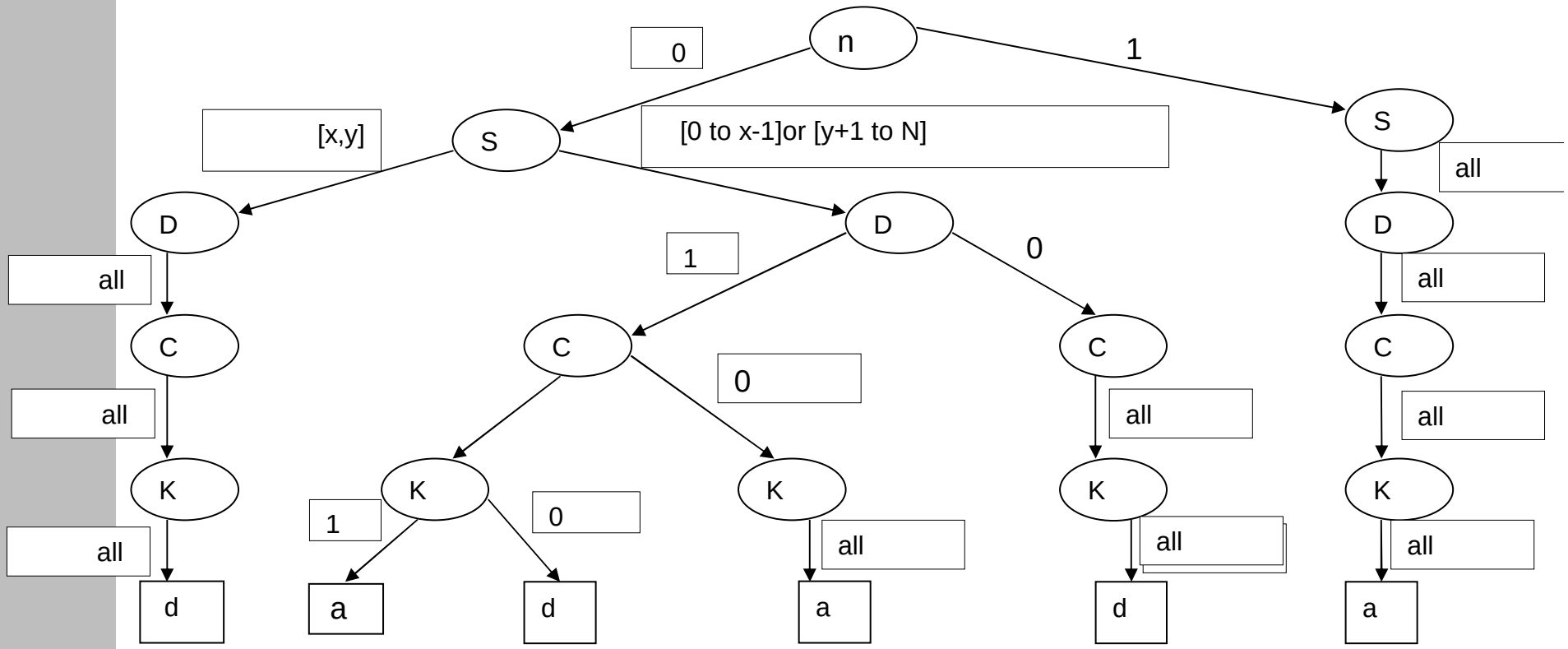


# Construction – 3 (Diagram based Model)

For unordered decision path not labeled with F, where F Є {F1,F2….Fd}:

- Insert a new node with label F.
- Connect it with the incoming edge at the place of insertion.
- The outgoing edge of the node  is labeled the domain of F.

# Construction - 4 (Diagram based Model)



# Shaping - 1

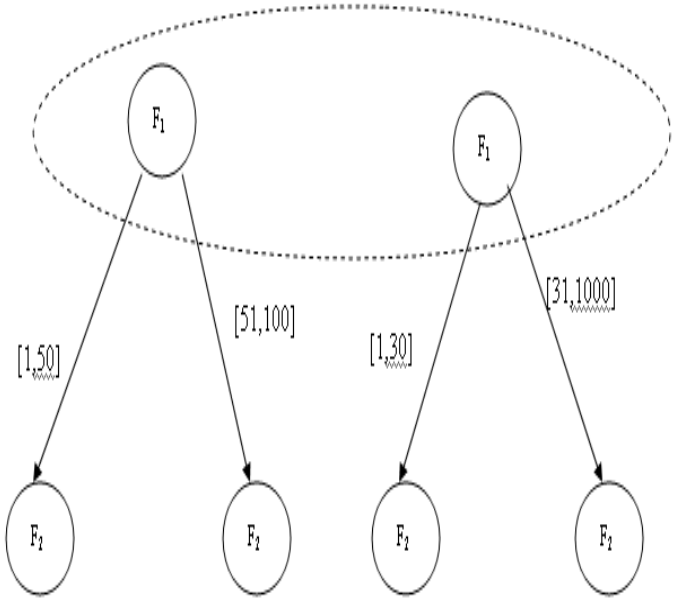
- Transforming of two ordered DD $f_a$ and $f_b$ to a two semi-isomorphic DDs $f_a$' and $f_b$'.
- Semi-Isomorphic DDs: Two DDs whose label of non terminal nodes and corresponding edges match. Only the labels of their terminal nodes may not match.

**Transformations:**

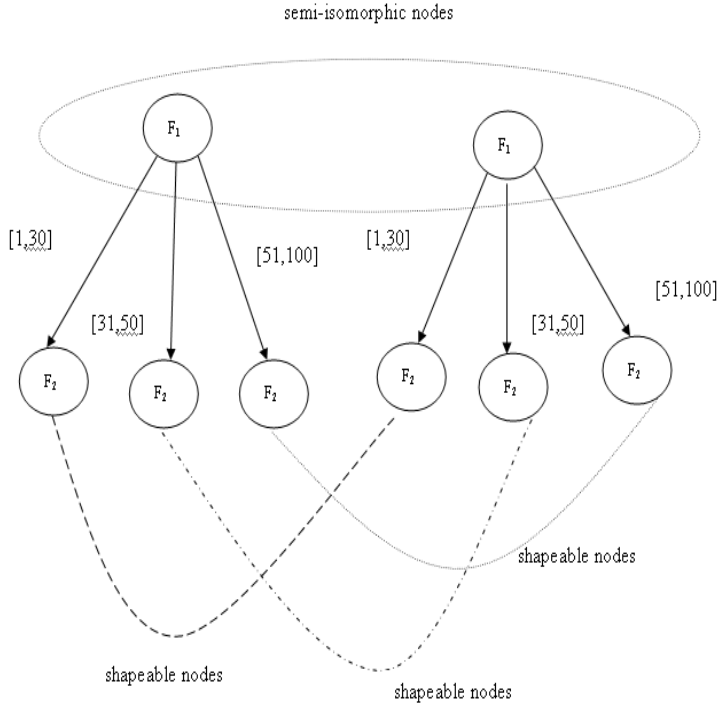
Node Deletion , Node Insertion, Edge Merging, Edge Splitting, Subgraph replication

# Shaping - 2

- Before node shaping Algorithm

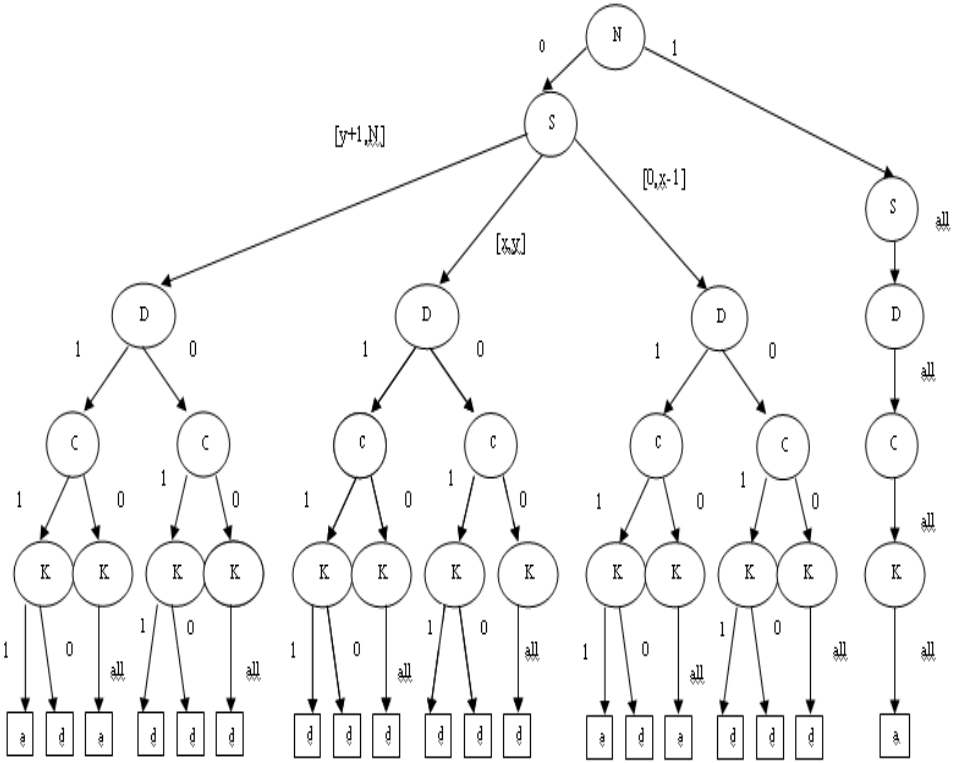


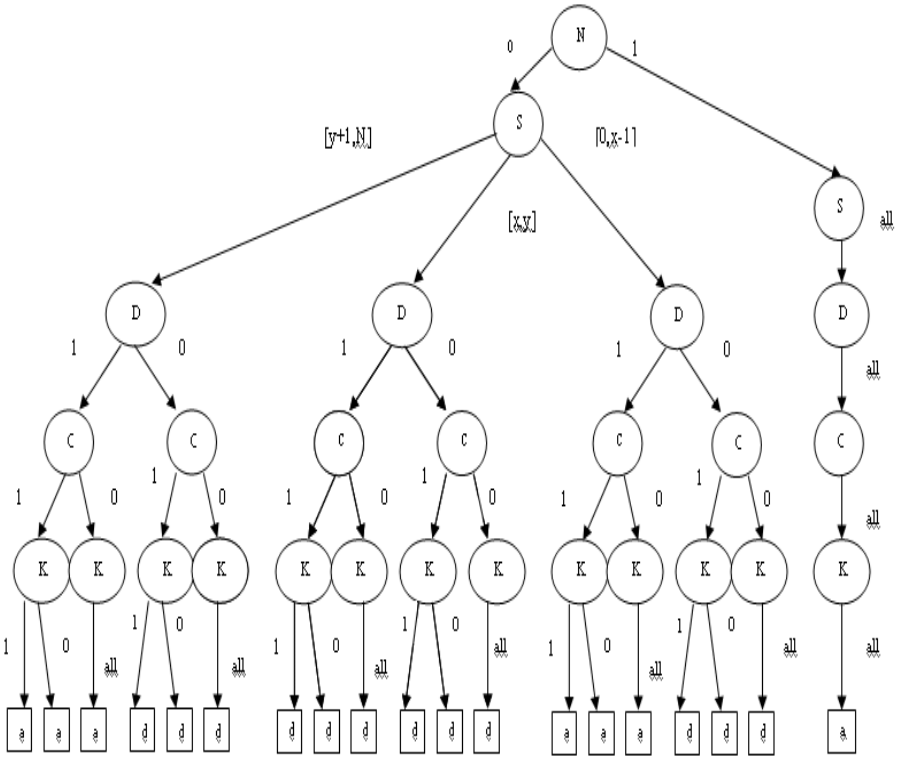

- Semi-Isomorphic nodes after node shaping algorithm



# Comparison

- Diagram Based Design
- Rule Based Design



# Conclusion

- Naïve solution uses brute force exhaustive input to test all the discrepancy. This is too slow.
- The discrepancy has been detected in one of the model. The designers of the model must correct it.
- This design diversity methodology can also be used in designing other static packet classifiers.