



Routes: consist of: **Flow** + ordered list of **Network Interfaces**

r1: (f1) S1_I1, SW1_I1, SW1_I2, SW2_I1, SW2_I2, SW4_I1, SW4_I5, S2_I1 (via SW2 to S2)
 r2: (f1) S1_I1, SW1_I1, SW1_I3, SW3_I2, SW3_I1, SW4_I2, SW4_I5, S2_I1 (via SW3 to S2)

r3: (f2) S1_I1, SW1_I1, SW1_I2, SW2_I1, SW2_I2, SW4_I1, SW4_I4, S3_I1 (via SW2 to S3)
 r4: (f2) S1_I1, SW1_I1, SW1_I3, SW3_I2, SW3_I1, SW4_I2, SW4_I4, S3_I1 (via SW2 to S3)

r5: f1 via SW2 to S3?
 r6: f1 via SW3 to S3?

Problem: Load-balance flow f1 between App2 and App3
 Ignore that flow f1 is only App1 → App2 ?

Idea 1:
 SDN RULE proposition:
 ru1: flow, interval function, +route, +probability??
 What is the influence of that on the transformations???

SDN Switching algorithm (in DNI):

- do you have a rule for the **flow**?
- if yes: forward to the port specified in **route** using {hw, sw} table (use **interval function** to calculate which one). Apply sw or hw **SDN forwarding performance** respectively.
- if no: forward to the controller, delay the flow based on **SDNControllerPerformance**, and forward to the next node based on appropriate **route** (assume, that the have just controller installed the rule only for you). If no **route** exists, do not forward flow - drop.

SDN Switching algorithm (in reality):

- do you have a rule for the **flow**?
- if yes: forward to the port specified in the rule
- if no: forward to the controller, wait until it adds a new rule. If no controller, use default action (e.g., drop traffic).

