Towards Smart Intents with Robust and Flexible Routing

Davide Sanvito, Mattia Gulli, Ilario Filippini, Antonio Capone
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Italy

ONOS Intent Framework allows programmers to specify high-level policies which are then compiled to low-level configurations by the controller. Intent gets re-compiled as a consequence of environment changes (e.g., link failures) to meet the objective. Without further constraints, intents are individually compiled to one of the shortest paths.

Can we consider jointly multiple intents in the compilation? Can we reactively take into account flow-level statistics events to optimize a global network objective, e.g., minimizing Maximum Link Utilization (MLU)? In this demo we extended the ONOS SDN-IP [1] application and evaluate the benefits in terms of average MLU with real traffic traces. Future plans include the definition of a new Smart Intent whose compiler monitors corresponding flows and periodically re-optimizes the paths according to their statistics. This would allow other applications to transparently take benefit from the new re-compilation logic and the exposed parameters (such as minimum time between reconfiguration, level of robustness to variations, etc.).

ONOS Intent Framework

SDN-IP application

SDN Network

BGP

ONOS intents

OpenFlow entries

External Network

Connects a SDN network to legacy external networks via BGP

BGP routes are received by internal BGP speakers, relayed to the ONOS app, translated to MultiPointToSinglePointIntents and then compiled to low-level OF messages

Extended SDN-IP application

TRAINING PERIOD

• Traffic is forwarded as in standard SDN-IP app
• AS-to-AS Traffic Matrices (TMs) are collected
  o TM endpoints inferred from BGP announcements
• Pairs of BGP routes are translated to PointToPointIntents

AT THE END OF THE TRAINING PERIOD

• Exploiting the quasi-periodicity of traffic, a new routing configuration is applied for the following period
  o Computed by solving [2] an optimization model taking into account flow statistics and minimizing the average MLU
  o Traffic deviations with respect to expected scenarios are coped with routing configurations robust over the TM space
• Optimization model and routing activation scheduling run by an off-platform app
  o RETrieve TM samples
  o Load a set of routing configurations
  o Apply a selected routing configuration

Demo description

• SDN-IP tutorial network fed with 2 days traffic from Abilene [3] replicated between pairs of Mininet hosts with iperf3
• MLU is monitored over the 2 days

SDN-IP

• traffic forwarding with standard intents for both days
  ext SDN-IP
  o traffic forwarding with standard intents during the 1st day
  o TMs collected during 1st day are used by the optimization model to generate robust routing configuration(s) for the 2nd day

A further step: Clustered Robust Routing (CRR)

• Set of robust routing configurations over the TM space
• Trade-off: number of reconfigurations vs robustness of routing
  o ONOS does not support consistent updates mechanism* during network update operations: a completely reactive approach can impair network performances!
• two optimization models:
  o Computation of a set of robust routing configurations
  o TM clustering in time, space and routing domains to compute the proper routing activation times
  o By-design guarantees on number of re-configurations and the minimum duration for a network configuration.

Issues and future works

• Splittable routing in ONOS?
  o Faster model resolution (LP vs ILP) and better solution (lower OF)
  o OpenFlow’s Group Tables? Advanced SDN data plane ([4],[5])?
• Connection disruption during network updates:
  o “Non-disruptive Intent Reallocation” from FBK CREATE-NET
• REST API:
  o gRPC more efficient with larger TMs and topologies?
• Transparent failure recovery by Intent Framework:
  o Paths enforced via LinkCollectionIntent not resilient
• Design data structures with ONOS distributed primitives
  o Current testbed runs a single ONOS instance
• Move the statistics-based recompilation logic from an off-platform app to the ONOS Intent Framework:
  o any application can transparently benefit
  o which parameters should we expose at Intent level?
  o heuristics as an alternative to the integration of optimization tools?

References

[1] https://wiki.onosproject.org/display/ONOS/SDN-IP

POLITECNICO MILANO 1863