Intent-Based Routing Policy Optimization in SD-WAN

Pham Tran Anh Quang[1], Sebastien Martin[1], Jérémie Leguay[1], Xu Gong[2], Xu Huiying[2]
Huawei Technologies Ltd., [1]Paris Research Center, France., [2] Dongguan Research Center, China.

***Abstract*—To optimize bandwidth utilization in wide area networks, a centralized controller typically maintains routing policies at edge routers. In this context, we propose a versatile intent-based policy optimization model that carefully selects the set of overlay links which are allowed for applications based on their requirements and the overall intents of the operator. The optimization model embeds QoS and traffic predictions to anticipate the impact of routing decisions. To address large scale scenarios where the behavior of the network and devices is not known exactly, we integrate data-driven predictions into a local search algorithm to optimize routing policies. The algorithm supports several intents such as the minimization of the congestion or the maximization of the network quality. Thanks to packet-level simulations on an SD-WAN scenario, we show that our intent-based policy optimization system improves significantly performances. For instance, the latency is improved by 40% when the high-quality intent is selected. In addition, the percentage of time SLAs are met is improved by 10% compared to legacy load balancing mechanisms.**

I. Introduction

Quality of Service (QoS) and routing policies are key to control how bandwidth is shared among different applications in computer networks. They can be tuned to satisfy Service Level Agreements (SLA) of applications and/or meet global objectives in terms of resource utilization. These later network-level objectives are often referred to as "intents" [1] and they can relate to the minimization of financial expenses, the congestion or the maximization of the experienced network quality. In typical Software-Defined Wide Area Networks (SD-WAN) [2] architectures, a centralized controller maintains a set of policies deployed at edge routers that interconnect multiple sites (e.g., enterprise branches, data centers). Each edge router is configured to send traffic to its peers over several transport networks (e.g., private lines based on MPLS, cheaper broadband Internet connections). Typically, these routers are responsible for the load balancing of flows so as to meet SLA requirements in terms of QoS, security, etc. The centralized controller can select high-level routing policies to improve SLA satisfaction and optimize global intents.

Conflicting objectives may arise at the controller when configuring routing policies. Indeed, specific SLA requirements must be satisfied at individual application level, while several intents expressed by the network owner also need to be optimized at the global network level. To take a concrete example, minimizing the overall financial cost may lead to the selection of low quality paths in terms of QoS, inducing a risk to violate application SLAs. Therefore, optimizing intents and satisfying SLAs at the same time in a dynamic environment where the traffic and the quality of transport networks vary over time is a real challenge. As we will see later in our work, the use of predictions to anticipate the impact of routing decisions greatly helps to resolve these conflicts.

Routing policy optimization algorithms have already been proposed for several purposes. In [3], the authors minimized financial expenses for the total traffic volume and a 95^{th} percentile charging rule. Google's B4 [4] fairly shares the available bandwidth of elastic applications. Centralized and distributed solutions have been proposed to maximize network utility [5], [6]. Several advanced variants of ECMP [7], [8], [9], mainly for data center networks, aim at minimizing congestion and latency. While these solutions solve a load balancing or path selection problem for different global objectives (e.g., cost, fairness, congestion), they do not aim at guaranteeing at the same time the QoS of individual applications. To optimize latency and other QoS parameters for a specific set of flows, closed-form performance models have been embedded into routing optimization algorithms. For instance, [10] considered the Kleinrock function [11] to minimize latency. Another line of work has applied Deep Reinforcement Learning (DRL) [12], under the umbrella of *experience-driven networking* [12]. However, these solutions are mostly centralized and face scalability issues, as the controller takes decisions for every flow in real-time.

In this paper, we present a semi-distributed intent-based system for SD-WAN that can optimize routing policies for multiple intents, defined by the network operator, while satisfying individual requirements of applications. It uses SLA and traffic predictions at the controller to optimize the Smart Policy Routing (SPR) [13] configurations that are deployed at edge devices. In its simplest form, an SPR policy defines for an application the set of transport networks, i.e. overlay links, that is allowed. Based on this, edge routers load balance traffic over the subset of allowed overlay links according to real-time measurements. To optimize policies in this context, we propose to use predictions to anticipate the impact of routing decisions and mitigate conflicts between individual and global objectives. We first formulate a policy optimization model that can support various intents (e.g., best quality, minimum congestion, minimum cost) while meeting QoS requirements of applications. As the model is nonlinear when a non-preemptive queuing model [14] is used to estimate the delay, we solve it using SCIP [15]. In order to optimize policies in large scale scenarios where only incremental policy changes are desirable and the behavior of devices (e.g., the scheduling architecture) is not exactly known, we introduce a local search algorithm that can embed data-driven prediction models for any type of traffic scheduler (i.e., beyond priority queues). Through packet-level simulations on a practical SD-WAN scenario, we demonstrate that the proposed solution based on predictions

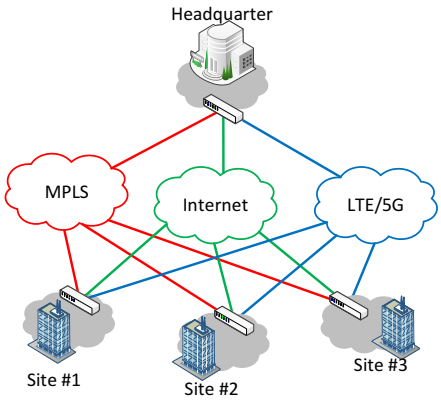


Fig. 1: SD-WAN network with an headquarter and 3 branches.

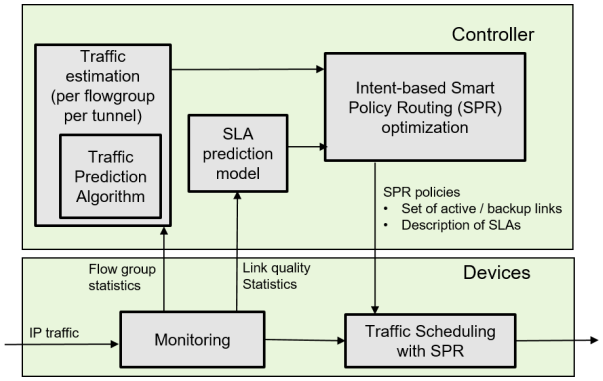


Fig. 2: Overall system architecture.

not only optimizes global intents (e.g., low congestion or best quality) but also SLA violations. The end-to-end delay and the portion of time where SLA are met are improved by 40% and 10%, respectively. We also show that the optimal policy calculated with the nonlinear solver SCIP provides similar performances compared to our local search procedure.

The rest of this paper is structured as follows. The system architecture and the problem formulation are introduced in Sec. II and III. Sec. IV presents the prediction models for traffic and SLAs. Sec. V introduces the local search algorithm to incrementally optimize policies at scale. Sec. VI describes our numerical results and Sec. VII concludes this paper.

II. System Architecture

We consider a semi-distributed architecture where edge devices are controlling traffic based on real-time measurements and policies managed by a centralized controller.

A. SD-WAN use case

Fig. 1 presents a typical SD-WAN use case where the headquarter and 3 branches (sites) of an enterprise are interconnected by several networks (e.g. MPLS, Internet, LTE) controlled by third-party operators. A controller is placed at the headquarter site and branches are equipped with Access Routers (ARs). Flows from applications are aggregated by *flow groups* that correspond to traffic classes with different SLA requirements. A typical traffic scenario includes Real-time, Business and Bulk flow groups, that respectively correspond to multimedia, business critical and non-critical applications.

The system architecture is split into two control entities operating at two different time scales. In a slow control loop, the global controller (at headquarter site) updates policies and communicates them to edge devices. In a fast control loop, devices take tactical decisions to follow the evolution of traffic and network conditions. For a flow group, the policy sent by the controller defines the routing logic (e.g., how to select outgoing paths), the QoS parameters (e.g., priorities, rate allocations) and security measures to be enforced. Without loss of generality, we will consider that the policy for each flow group consists in the set of overlay links that is allowed and its priority level. Priorities are encoded using the DSCP field of IP packets and a non-preemptive priority scheduler is used at each outgoing port. Each AR router load balances traffic over the set of allowed overlay links according to actual network conditions (e.g. average delay, loss, jitter of overlay links).

Fig. 2 depicts the overall system architecture. As we can see, the traffic is scheduled by the device according to the Smart Policy Routing (SPR) mechanism (see Sec. II-B) and policies are optimized globally at the controller. In this work, we propose to optimize SPR policies to satisfy global intents and SLA requirements. In order to take non-myopic decisions, this process takes as input traffic and SLA predictions.

B. Smart Policy Routing

We briefly present how the Smart Policy Routing (SPR) mechanism works inside Huawei devices (see [13] for details). Access routers are configured with a policy for each flow group that contains the set of allowed overlay links and the SLA requirements that must be satisfied (i.e, acceptable latency, jitter and loss). The quality of overlay links is continuously monitored and evaluated. The set of overlay links that remain eligible with regards to the SLA requirements of each flow group is used to load balance traffic. Traffic is distributed proportionally to the nominal capacity of overlay links.

C. Configuration of policies

In order to avoid congestion and satisfy SLA requirements, policies have to be carefully decided by the controller to mitigate interferences between flow groups. The goal of the controller is twofold: 1) help devices to satisfy SLAs and 2) optimize global intents. The policy optimization process takes as input measurements collected by the monitoring system, e.g., average throughput of flow groups per min/h/day/month, average throughput and QoS of overlay links, and average SLA measurements of each flow group over each overlay link. The controller also considers available information such as the nominal capacity of overlay links, the global intents to optimize and the SLA requirements of each flow group. It can also use parameters related to the cost model, the relative priority of flow groups, their preferences for each transport network, etc.

III. Intent-based Policy Optimization Model

In this section we present a mixed nonlinear mathematical model to determine the best routing policy according to several global intents and SLA requirements of applications.

We consider an SD-WAN network composed by a set of overlay links E with capacity C_e, $\forall e \in E$, and a set of flow groups K. For each flow group $k \in K$, the estimated traffic demand is denoted b^k and the requirement in terms of maximum delay over all overlay links is $\bar{D}_k$.

Let's consider the following variables:

- $x_e^k \in [0,1]$ the split ratio of each flow group $k \in K$ over each overlay link $e \in E$;
- $u_k \in \{0,1\}$ equals to 1 if the SLA of flow group $k \in K$ is respected and 0 otherwise;
- $LU_e \in [0,1]$ the utilization in percentage for each overlay link $e \in E$;
- $D_k \in \mathbb{R}$ the maximum delay estimated by flow group k over all overlay links.

The policy optimization problem can be formulated by the following model with 3 different intents (objectives):

$$\min \quad Obj_1 \sum_{k \in K} u_k + Obj_2 \sum_{e \in E} LU_e + Obj_3 \sum_{k \in K} D_k \qquad (1)$$

$$LU_e = \sum_{k \in K} b_k x_e^k \leqslant C_e \quad \forall e \in E, \qquad (2)$$

$$f_e^k(x) \leqslant D_k \quad \forall k \in K, \forall e \in E, \qquad (3)$$

$$D_k - \bar{D}_k \leqslant M u_k \quad \forall k \in K, \forall e \in E, \qquad (4)$$

$$\sum_{e \in E} x_e^k = 1 \quad \forall k \in K \qquad (5)$$

where $f_e^k(x)$ is the delay function that provides the delay of flow group k over overlay link e by considering the assignment given by x and M is a big value. The delay function can be derived from a closed-form expression, obtained through measurements or estimated using a data-driven model (see Sec. IV-B). Constraints (2) ensure that the capacity of each overlay link is satisfied. Inequalities (3) compute the delay of each flow group. Constraints (4) verify the satisfaction of SLAs. Constraints (5) ensure that all the traffic demand is routed.

In this model (1)-(5), the first intent (i.e., Obj_1) maximizes the SLA satisfaction of flow groups (*Safety* intent). The second intent (i.e., Obj_2) minimizes the MLU (Maximum Link Utilization) - (*Low congestion* intent). The third one (i.e., Obj_3), also called *High quality* intent, minimizes the average delay, decreasing the delay beyond SLA requirements. By defining a specific weight for each intent (Obj_1, Obj_2, Obj_3) we can tune the routing strategy and define a particular priority order for the different intents.

Other intents can be considered like the minimization of a financial cost (*minimum cost* intent) or the minimization of policy modifications over time (*stability* intent). To consider a financial cost, we can introduce a fourth objective $Obj_4 \sum_{e \in E} W_e LU_e$ where W_e is the cost associated with the overlay link $e \in E$. More complicated models such as the 95^{th} quantile charging rule [3] can be easily integrated. For the stability intent, stickiness constraints can be considered to limit the number of split ratio modifications between two time steps. To do so, we need to add a time slot index on each variable ($x_e^k(t)$, $D_k(t)$, $LU_e(t)$, $u_k(t)$) to consider the values of each variables at time step t. Furthermore, we need to consider additional variables $z_e^k(t)$, for each overlay link $e \in E$, each flow group $k \in K$ and each time slot $t > 0$. The stickiness constraints are then given by the following inequalities: $|x_e^k(t-1) - x_e^k(t)| \leqslant z_e^k(t), \forall t > 0$, where $x_e^k(0)$ represents the current split ratio. We also need the following term in the objective function: $Obj_5 \sum_{k \in K} \sum_{t>0} z_e^k(t)$. By considering time steps, traffic predictions for each flow group can be used with Model Predictive Control (MPC) [16]. In this case, we simply replace b^k by traffic predictions $b^k(t)$.

In our implementation, the SLA prediction f relies either on a queuing model or a machine learning model (see Sec. IV-B). The full optimization model supports a proper estimation of UCMP weights proportionally to the capacity of overlay links. In this case, x variables are binary and defines the SPR policy (the set of allowed links).

IV. Prediction models

To mitigate conflicts between the optimization of global intents and the satisfaction of individual SLAs, we propose to integrate traffic and SLA predictions into the intent-based policy optimization model. The prediction of SLA violations can be realized using a closed-form model from queuing theory or network calculus. While these analytical models can be complex for tight latency estimations or sophisticated scheduling architectures, tractable expressions can be used in some cases and integrated in the overall policy optimization model. For a more general approach, the use of data-driven or machine models is a promising alternative. Indeed, they can handle situations where the behavior of devices and the underlying network is not known a priori (e.g., scheduling architecture, overlay with background traffic). They require a sufficient dataset to train an appropriate model. However, in the case of SD-WAN, the controller is already responsible of giving visibility to users about the performance of applications based network statistics collected by devices. Therefore, it is safe to assume that the controller has abundant data to train models for SLAs and traffic predictions. Input data is the traffic and the corresponding delay for each flow group and each overlay link, collected every τ seconds (e.g. $\tau = 5s$). The controller can make predictions for each flow groups with individual models or jointly for all of them with a single model. Several solutions are possible.

In the rest of this section, we present two approaches to predict the delay of overlay links for our SD-WAN architecture (i.e., $f_e^k(x)$): (i) a closed-form model from queuing theory that can be used as a benchmark when a non-preemptive priority scheduler is used and (ii) a more general data-driven model.

A. Queuing model

The delay of overlay links can be theoretically estimated under some assumptions about the network. For instance, we can consider that the network is homogeneous and that every device has the same queuing mechanism at outgoing ports, a non-preemptive priority scheduler with one queue for each flow group. Note that priority queues are widely available in commercial off-the-shelf devices [17]. From [14], the mean waiting time for each priority can be modelled as follows:

$$\bar{W}_k = \frac{\bar{R}}{(1-\rho_1-...-\rho_{k-1})(1-\rho_1-...-\rho_k)} \quad (6)$$

where $\bar{R} = \frac{\sum_k \lambda_k \overline{S_k^2}}{2}$ is the mean residual time in the server when a packet arrives and it is computed based on the arrival rate of flow group k, i.e. λ_k, and the service time of flow group k, i.e. $\overline{S_k}$, that depends on the overlay link capacity and the average packet size. To finally derive the overall delay for a flow group, it is necessary to add the propagation delay and packet transmission time.

B. Data-driven model

As the controller already monitors the network quality for overlay links and applications, it has sufficient data to derive prediction models. The traffic data for each flow group on each overlay link can be exploited to infer the delay of the overlay links based on the estimation of their future load. To solve the underlying regression problem, we use two models: (i) a polynomial regression (Polyfit) [18] and (ii) a neural network (ML). We divided the data set into a training set and a test set. The regression performance is evaluated with the R^2 score defined as follows:

$$R^2 = 1 - \frac{\sum (y - f(x))^2}{\sum (y - \bar{y})^2}, \quad (7)$$

with $f(x)$ the regression model, y the ground truth, and $\bar{y}$ the mean of y. This score measures how good the regression model is compared to the constant model, i.e. a naive predictor returning the expected value. The main benefit of this approach is that it comes with no assumption about the networks, and that therefore, it can be adopted in a wide range of scenarios.

In the rest of the paper, we will consider that the delay on each overlay link for each flow group is predicted by a separate model. More complicated approaches could be used, however the goal of our work is to assess the gains of embedding such a prediction into the optimization of routing policies. Each prediction model takes as input the traffic demand of all flow groups over a particular overlay link and returns the predicted delay of a specific flow group. These models are embedded into the evaluation function of the local search algorithm presented in the next section.

V. Local Search Algorithm

In this section, we propose a local search algorithm to calculate a good routing policy update at each step (each time algorithm is called). As we cannot drastically change all policies and we need to incrementally modify policies at every time step, the local search method turns to be a natural approach to solve the routing policy optimization problem. The goal of this algorithm is to evaluate all possible modifications of overlay link assignments to flow groups (i.e., SPR policies) and select the best one in an iterative manner. The algorithm continues until the solution cannot be improved.

Considering the current network status and predictions the algorithm assigns a set of overlay links to each flow group k. Predictions models are used to predict the traffic demand for each flow group and to estimate SLA violations. UCMP is used to quantity the traffic of that flow group sent over overlay links once the assignment is known.

The current solution is represented by a vector of Boolean that provides the assignment for each flow group to the associated overlay links. The algorithm tests all possible modifications to keep the best one and continues until convergence (no possible improvement). The objective function of model (1)-(5) is used in *eval()*, the function that evaluates candidate solutions. For instance, if a flow group is assigned to two overlay links $\{a, b\}$ and a third link c is available for this flow group, then the local search algorithm evaluates the following assignment $\{a\}$, $\{b\}$,$\{a, b, c\}$ where the assignments of other flow groups do not change. The algorithm selects the best solution over all flow groups. If the best one is worse than the current solution the local search algorithm then stops.

Algorithm 1: Local search algorithm

```
Result: A heuristic solution
CurrentSol is the assignment of flow groups to overlay links;
eval() uses the objective function of model (1)-(5);
Improvement ← true;
while Improvement do
    Improvement← false;
    BestSol ← CurrentSol;
    EvalBestSol ← eval(BestSol);
    for each couple overlay link e and flow group k do
        CurrentLocalSol ← CurrentSol;
        CurrentLocalSol[e, k] ← no CurrentLocalSol[e, k];
        if CurrentLocalSol is not valid then
            continue;
        end
        EvalCurrentLocalSol ← eval(CurrentLocalSol);
        if EvalBestSol>EvalCurrentLocalSol then
            Improvement ← true;
            BestSol ← CurrentLocalSol;
            EvalBestSol ← EvalCurrentLocalSol;
        end
    end
    if Improvement then
        CurrentSol ← BestSol;
    end
end
return CurrentSol;
```

VI. Simulation Results

We now evaluate our intent-based policy optimization system using NS3 [19] with OpenFlow 1.3 [20] to emulate the control plane logic and select routing paths. As depicted in

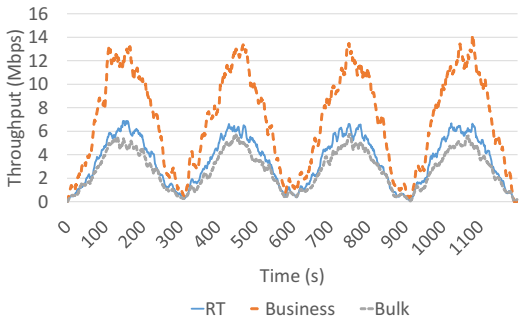


Fig. 3: Traffic of different applications (i.e., flow groups).

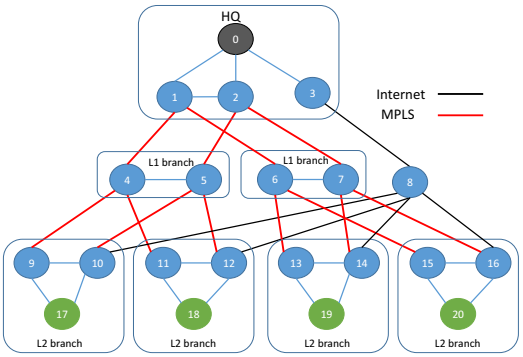


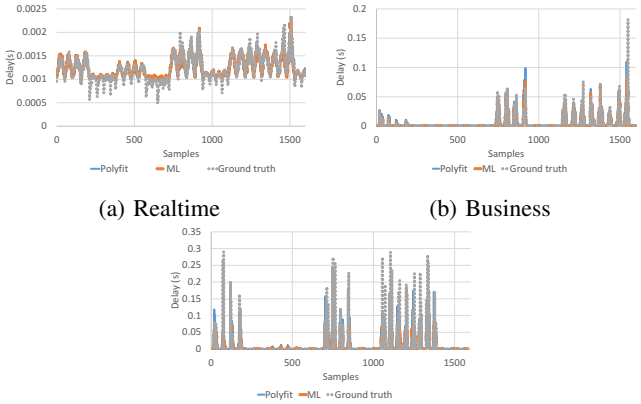
Fig. 4: SD-WAN scenario for the simulations.

Fig. 4, the simulation scenario is composed of an Headquarter (HQ) site connected to 2 Level-1 branches (L1) with MPLS lines with propagation delays of 1ms and capacities of 4 Mbps and 8 Mbps, respectively. Each L1-branch also connects to 2 Level-2 branches (L2) with MPLS with delays of 10ms and 15ms and capacities of 4 Mbps and 8 Mbps, respectively. The HQ is also connected to all L2-branches using a broadband Internet access, with a propagation delay of 45ms and a capacity of 12 Mbps. In the figure, red links represents MPLS and black links Internet. We considered three types of applications: Real-time (RT), Business, and Bulk. SLA requirements of these flow groups are a maximum delay of 40ms, 60ms, 200ms respectively. The transport layer is TCP. The microflow inter-arrival time varies to generate diurnal traffic patterns. Fig. 3 shows the overall amount of traffic per flow group. Business traffic occupies most of the bandwidth in high load periods, while RT and Bulk are similar in intensity. In this scenario, we only consider traffic between L2 and HQ. Traffic has also been down scaled compared to reality so that execution time remains acceptable. The simulation time is 1200s.

Traffic is prioritized using priority queues (one per flow group) and packets are marked with DSCP. Measurements are averaged over a time window of 5s and policies are updated by the controller every 5s. Measurements in the current time slot are exploited to predict the delay in the next time slot.

Flow group	RT	Business	Bulk
Polyfit	0.82	0.77	0.5
ML	0.8	0.72	0.25

TABLE I: R^2 score for delay predictions.



(a) Realtime (b) Business

(c) Bulk

Fig. 5: Predicted and ground truth delay for Polyfit and ML.

A. Accuracy of SLA prediction models

To evaluate Polyfit and ML approaches, we generated a data set by executing multiple simulations with ECMP. It contains 7200 samples in which 6000 samples are used for training and 1200 samples are used for testing. We also do a grid search to find the optimal hyper-parameters for both approaches. Table I presents the R^2 score obtained for different applications. In both models, the R^2 scores of high priority flows, i.e. RT and Business, are high. That indicates a high accuracy of the model for most critical flow groups. Fig. 5 shows that prediction errors compared to ground truth are small in practice. Polyfit plots a better performance than ML. As Bulk depends on higher priorities, the results confirm that it is harder to predict latency for this class. However, latency peaks (not their intensity) can well be predicted, which is enough for the optimization of routing policies. Although it may be possible to obtain better performance with more advanced techniques, e.g. graph convolutional network [21], we used Polyfit inside the local search algorithm.

B. Benchmark solutions

As mentioned in Sec. IV, we can use a closed-form queuing delay model as the scheduling architecture is known in our simulations. In this case, we solve the policy optimization model using the queuing model of Sec. IV-B with SCIP [15], a non linear solver, for a benchmark solution (referred to as *OM (Optimization Model)* later). We considered safety (Obj_1), low congestion (Obj_2) or high quality (Obj_3), and stability (Obj_5) intents in the objective function. Furthermore, we also compare our solution to the cases 1) where no SLA predictions are available, called *NM (No Model)*, or 2) where all available overlay links are allowed (policies are not optimized), called *AL (All Links)* later. To disable SLA predictions in the optimization model presented in Sec. III, the

Intent	High Quality			Low Congestion			
Algo.	OM	NM	LS	OM	NM	LS	AL
RT	99.3	94.6	100	97.45	95.63	99.53	93.0
Business	99.72	90.9	100	97.4	91.2	96.9	87.6
Bulk	98.14	93.51	98.53	98.78	90.5	97.78	91.1

TABLE II: Percentage of time flow groups meet their SLAs.

function $f_e^k(x)$ in Constraint (3) always returns the measured delay of the overlay link. Note that this measure is the delay of the current load balancing policy which can be quite different after the new policy is applied.

C. Performance of policy optimization algorithms

We consider two global intents: (i) high quality and (ii) low congestion. For both intents, guaranteeing all SLAs of flow groups is still the highest priority. For the high quality intent, the algorithm attempts to obtain the lowest average delay while for the low congestion intent it minimizes the MLU. Table II shows the portion of time flow groups meet their SLA requirements. Results are presented for the Local Search (LS) algorithm that embeds SLA predictions using the Polyfit model. It also shows results for benchmark solutions: OM, NM and AL In the OM approach, the execution time of SCIP solver is limited to 100s and the best feasible solution found is returned. We can observe that for both intents, OM and LS performs similarly reflecting the high accuracy of SLA predictions and the effective capability of local search to find a near-optimal solution. Without SLA prediction, i.e with NM, a degradation of 5% to 10% occurs in all flow groups, especially for Business which has a strict SLA requirement and a high demand. When the device takes decision without instructions from controller, i.e. AL, we even see a deeper degradation because routing policies are not coordinated. The average delay and the 95^{th} percentile are shown in Table III. For both intents, the delay of OM and LS is significant lower than for ON and AL. For Business, the average delay of LS is 80% lower than for NM. For Bulk, the gap is even greater and the average delay of LS is twice lower than for NM.

Table III also presents the MLU. For both intents, NM has a higher MLU compared to others because it places traffic on the lowest delay link without anticipating the impact. In contrast, LS and OM are able to predict the increase of the delay, thus actively avoiding congestion. As expected the MLU is smaller for the low congestion intent compared to high quality.

Intent	High Quality			Low Congestion			
Algo.	OM	NM	LS	OM	NM	LS	AL
	Average delay						
RT	15.88	17.71	16.57	16.41	17.64	15.22	17.18
Business	23.47	45.92	24.88	30.33	37.5	29.3	41.48
Bulk	39.95	78.87	34.34	46.43	49.55	52.6	45.3
	95th Percentile delay						
RT	30.15	33.3	24.6	29.34	33.9	17	36.74
Business	42.48	74.24	46.42	46.34	68.32	44.24	68.05
Bulk	126.7	287.7	120.5	118.7	213.3	139	93.8
	Average Maximum Link Utilization (MLU)						
MLU	67.5	75.34	63.03	54.43	63.53	56.3	56.96

TABLE III: End-to-end delay and MLU for all flow groups.

VII. Conclusion

We proposed a semi-distributed intent-based system for SD-WAN that splits traffic according to global intents and applications SLAs. To better optimize routing policies, it leverages traffic and SLA predictions. It uses light-weight data-driven models to evaluate actions inside a local search algorithm. We demonstrated through simulations that predictions can achieve a high accuracy for the delay, thus helping the policy algorithm to find near-optimal solutions. The proposed mechanism remarkably helps to guarantee SLA and optimize intents. Future work could focus on improving the accuracy of SLA predictions using advanced ML techniques and evaluating the system with other constraints on packet loss and jitter.

References

[1] L. Pang, C. Yang, D. Chen, Y. Song, and M. Guizani, "A survey on intent-driven networks," *IEEE Access*, vol. 8, pp. 22 862–22 873, 2020.

[2] Z. Yang, Y. Cui, B. Li, Y. Liu, and Y. Xu, "Software-defined wide area network (SD-WAN): Architecture, advances and opportunities," in *Proc. IEEE ICCCN*, 2019.

[3] Z. Duliński, R. Stankiewicz, G. Rzym, and P. Wydrych, "Dynamic traffic management for SD-WAN inter-cloud communication," *IEEE JSAC*, vol. 38, no. 7, pp. 1335–1351, 2020.

[4] Jain *et al.*, "B4: Experience with a globally-deployed software defined wan," *ACM SIGCOMM CCR*, vol. 43, no. 4, pp. 3–14, 2013.

[5] Z. Allybokus, K. Avrachenkov, J. Leguay, and L. Maggi, "Multi-path alpha-fair resource allocation at scale in distributed software-defined networks," *IEEE JSAC*, vol. 36, no. 12, pp. 2655–2666, 2018.

[6] Y. Magnouche, P. T. A. Quang, J. Leguay, X. Gong, and F. Zeng, "Distributed utility maximization from the edge in ip networks," in *2021 IFIP/IEEE IM*, 2021.

[7] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V. T. Lam, F. Matus, R. Pan, N. Yadav, and G. Varghese, "CONGA: Distributed Congestion-Aware Load Balancing for Datacenters," *ACM SIGCOMM CCR*, p. 503–514, Aug. 2014.

[8] A. Kabbani, B. Vamanan, J. Hasan, and F. Duchene, "Flowbender: Flow-level adaptive routing for improved latency and throughput in datacenter networks," in *Proc. ACM CoNext*, 2014.

[9] K. He, E. Rozner, K. Agarwal, W. Felter, J. Carter, and A. Akella, "Presto: Edge-based load balancing for fast datacenter networks," *ACM SIGCOMM CCR*, vol. 45, no. 4, pp. 465–478, 2015.

[10] W. Ben-Ameur and A. Ouorou, "Mathematical models of the delay constrained routing problem," *Algorithmic OR*, vol. 1, no. 2, 2006.

[11] L. Kleinrock, *Communication nets: Stochastic message flow and delay*. Courier Corporation, 2007.

[12] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *IEEE INFOCOM*, 2018.

[13] "Huawei Technologies, Smart Policy Routing," Tech. Rep., 2021.

[14] J. Sztrik, *Basic queueing theory*. Faculty of Informatics, University of Debrecen, 2012.

[15] G. G. et al., "The SCIP Optimization Suite 7.0," Optimization Online, Technical Report, March 2020.

[16] P. T. A. Quang, Y. Magnouche, J. Leguay, X. Gong, and F. Zeng, "Model predictive control for load balancing," in *ACM SIGCOMM (demo session)*, 2020.

[17] "Cisco IOS Quality of Service Solutions Configuration Guide," 2012.

[18] E. Ostertagová, "Modelling using polynomial regression," *Procedia Engineering*, vol. 48, pp. 500–506, 2012.

[19] G. F. Riley and T. R. Henderson, *The ns-3 Network Simulator*, 2010.

[20] L. J. Chaves, I. C. Garcia, and E. R. M. Madeira, "OFSwitch13: Enhancing Ns-3 with OpenFlow 1.3," in *Proc. of NS3 Workshop*, 2016.

[21] S. Lathuilière, P. Mesejo, X. Alameda-Pineda, and R. Horaud, "A comprehensive analysis of deep regression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 9, pp. 2065–2081, 2020.