R02 Network Architectures Essay 2

Alex Chau (ac2822), Churchill College

1 Introduction

In this essay I will be covering two papers; PCC Vivace – Online-Learning Congestion Control and DOTE. Both of these papers show how ML techniques can be applied to networks, the former being an application of a learning framework to congestion control. The latter applying ML to enable robust traffic engineering. Whilst both techniques overall aim to maintain network performance and prevent overload, congestion control is reactive and operates in small timeframes (milliseconds), whereas traffic engineering is proactive and operates in timeframes of minutes. A naive approach would be to treat ML/AI as a one-size fits all solution in networking, in reality the applicability of these techniques is highly dependent on the network constraints and how best you can tailor the ML technique to solve the problem.

2 PCC Vivace

PCC Vivace is a novel rate-control protocol which realizes the PCC framework from an online-learning perspective. The original realization of PCC, known as "PCC Allegro" takes a black box network-modeling approach to the problem. Allegro divides time into consecutive Monitor Intervals (MIs), where each interval is used to test the performance implications of sending at a specific rate. These metrics are collected and fed into a utility function, outputting a numerical value. It then adapts the sending rate in the direction that is empirically associated with higher utility. The paper's experiments indicate that this technique fails to achieve optimal low latency, has an overly long convergence time and are highly aggressive towards TCP. Some of the reasons for the far-from optimal performance is Allegro's simple utility function (essentially plotting rate against utility) and lack of latency-awareness. One of the reasons for the simple utility function is that adding more parameters such as latency makes reasoning about tradeoffs in utility all the more difficult, and adding more parameters to the utility function further complicates this issue. Allegro also has a naive aproach to rate control, as it uses a fixed step size which is bound to be too much or too little to converge or react appropriately to maximum utility. These choices prevent Allegro from attaining good performance under rapidly changing network conditions and fails to alleivate bufferbloat. Whilst Allegro produces a convergence rate/stability tradeoff that is better than TCP, it leaves much to be desired and causes high packet loss upon convergence.

PCC Vivace addresses the limitations with Allegro by using ideas from online convex optimisation, replacing the utility function framework with an online learning theory-informed framework for utility derivation, incorporating latency into the utility function. Latency-awareness is incorporated into the utility function via an RTT gradient, where this value represents the change in latency experienced within the MI. This decision guarantees multiple competing Vivace senders converge to a near-optimal, unique, and fair stable rate configuration. Vivace then employs a gradient-ascent based no-regret online optimisation to adjust the sending rates. Experimentally, Vivace significantly improved on the state of the art (at the time of 2018) achieving fast convergence, better performance, greater reactivity and being more TCP-friendly. Additionally Vivace is easily deployed, requiring sender-only changes.

The approach to Vivace is well-intentioned, you can imagine that with the interest around AI there would be attempts to overhaul the entire PCC framework with a deep neural net to leverage their inherent inference/predictive abilities. At the time-scale of MIs, you cannot afford the heavy inference provided by

DNNs and state changes too quickly for offline training to be useful. Instead, the authors subtly lean into machine learning techniques rather than just flatly applying predictive modeling via a neural network as would be done with a "hammer" approach. By appropriately borrowing ideas from convex optimisation and online-learning, the authors are able to provide (mathematically provable) performance guarantees about their utility function. On the flip-side, lets say you were to implement a DNN for congestion control (assuming it can be done fast enough). Whilst you can account for other metrics other than throughput, latency and loss over an MI, there are no performance guarantees about the outputs of the DNN. The solution would largely be a black-box, a DNN could probably discover some useful parameter configurations which trade off metrics even more optimally than Vivace does over MIs but would require substantial compute, which isn't feasible for typical end devices like mobile devices. Additionally how would a DNN react to network conditions that differ substantially from what they're trained on? Vivace can loosely be described as an ML/AI approach, framing the PCC problem from an online-learning persepctive and using a subset of optimisation techniques to overcome the identified issues seen in Allegro. Vivace operates on a "no-regret" learning model, the near-optimality guarantees about Vivace's rate selection are relative to the best fixed-rate in hindsight. This has some limitations, in reality the optimal rate for the environment most likely isn't fixed over a given time-period as networks are highly dynamic. The best fixed-rate in hindsight is a weak benchmark for the optimality of Vivace.

2.1 Bullet point notes

Some extra things worth mentioning:

- Vivace relies on empirical measurements of throughput, latency and loss over MIs. These measurements can be unreliable due to network conditions.
- There's contention between accurate measurements and reactivity. The confidence metric used in the paper sacrifices reactivity, but limiting confidence will degrade measurement accuracy/usefulness.
- In short lived flows/bursty networks, can Vivace meaningfully minimise regret? As the confidence metric requires some longevity in the flow to determine if its a good measurement or not.

3 DOTE

Direct Optimisation for Traffic Engineering (DOTE), applies a neural network approach to WAN TE. The two traditional approaches to TE are demand-prediction and brokerage/explicit demand signalling. The demand-prediction approach collects data from switches and passes this data to a demand prediction model to forecast future traffic demands. Given this prediction, they solve a TE optimisation problem for the network attempting to minimise maximum link utilisation (MLU), maximum throughput, and maximum concurrent-flow. The brokerage/explicit demand signalling technique requires the deployment of agents/shims into the OS of hosts generating traffic. Which then signal their expected demands to a central broker, this broker aggregates the demands and passes them to a centralised optimiser for TE. This method of TE is typically used for "scavenger-class traffic", which is non-latency sensitive or backup traffic. The authors argue there is an objective mismatch with these techniques and the overall aim of TE. Demand-prediction requires minimising mean-squared error between predicted demand and actual demand, there is an objective mismatch between the true goal of TE (MLU, throughput, and concurrent-flow) A predictor may perform well under the metric of MSE, but the resulting TE configuration isn't guaranteed to perform well under real network traffic conditions. For customer-class traffic, which is highly bursty and unpredictable, accurate demand-prediction is challenging and unreliable. In the case of explicit demand signalling, it is

infeasible to require end-hosts and applications to install shims, this approach can only realistically be used for scavenger-class traffic and not general customer-facing traffic.

DOTE overcomes the drawbacks of the afforementioned methods in several ways. Rather than trying to demand-predict future traffic loads, and then solve a TE optimisation problem based on predicted traffic, DOTE instead directly learns a mapping of recent traffic history to a TE configuration. This is achieved via a fully-connected deep neural network which, given the last H-long demand-matrix observations it outputs TE configurations for the next interval. This overcomes the objective mismatch seen in demand-prediction as DOTE directly optimises the TE objectives rather than an intermediate prediction error seen in demand-prediction rather than an intermediate prediction error seen in demand-prediction. The paper shows that for the standard TE objectives, the expected objective as a function of the TE configuration satisfies convexity, meaning the problem can be solved efficiently using stochastic gradient-based learning techniques and a guarantee of convergence. DOTE only requires passive monitoring of past traffic, overcoming the invonvenience of installing shims in the brokerage method, making it more applicable to consumer-class traffic. Overall, these improvements remove the burden of predicting future demand to directly learning how to set TE given past behaviour, aligning the learning objective with the network objective, making gains in robustness, performance and practicality.

DOTE applies ML/AI techniques appropriately following an analysis of the state of the art solutions in the problem space. TE is framed as a stochastic optimisation problem which lends itself to the use of a DNN, optimally mapping traffic histories to TE configurations. Leveraging the inference/pattern recognition capabilities of DNNs in high-dimensional data such as traffic histories allows DOTE to capture correlations in traffic unseen to conventional prediction. Learning directly in the space of TE decisions allows the DNN to generalise across complex traffic patterns as seen in customer-facing traffic. DOTE represents a careful, proportiionate use of AI. Where the DNN handles the high-dimensionality of traffic data, and the optimisation structure (stochastic) ensures performance oriented decisions. There is some discussion on the paper about leaning even further into ML/AI techniques, specifically why they chose not to use reinforcement learning (RL), this can be summed up as follows. Whilst RL techniques achieve the same replacement of explicit demand-prediction with end-to-end optimisation. RL suffers from higher data-sample complexity, suffers from noisy training and a complex optimisation process. The authors observe that WAN TE exhibits a desirable structure which allows for much simpler and robust optimisation, leaving RL as an ”overkill”.

4 General Principles for ML/AI in networking

Between Vivace and DOTE there is a clear underlying theme, effective application of ML/AI for networking is not about using the most expressive model, but the most appropriate one for the problem structure and time-scale. Congestion control operates at the millisecond granularity and under tight resource constraints. Making heavily predictive ML techniques unsuited for the task. Instead lightweight online optimisation and no-regret learning provide both responsiveness and theoretical guarantees, which is lost if you were to choose a more ML heavy method. Vivace demonstrates that borrowing targeted ideas from ML/AI, is sufficient and superior when decisions need to be fast, interpretable and robust to noise.

In contrast, WAN TE operates on intervals in the order of minutes, with more historical data and a slower decision cycle. This problem permits a more complex modelling in the form of a DNN because inference doesn't need to occur as rapidly as seen in Vivace. However, even with an environment more permissive to ML/AI techniques, DOTE restrains its usage of the techniques and uses the DNN only where it is truly needed. Avoiding the brittleness and unpredictability of treating neural networks as a black-box.

A broader principle emerges from these two papers: AI should augment, not replace domain insight. These

highly successful designs integrate learning with established optimisation structures, and exploit network objectives in a way that lends itself to these ML/AI techniques. The papers prioritise interpretability and performance guarantees over raw model complexity. A key takeaway is that ML/AI shouldn't be applied broadly to networking, but is a design choice that respects the dynamics and operational constraints of the system.

5 Further work

In 2020, the same research group followed up with PCC Proteus, applying the PCC Vivace framework to "Scavenger" traffic. In this paper they show that the same Vivace Utility function can be generalized to adopt priority??????