Network Performance Models

Link Rate – How many bits can be "pushed" onto a link per unit time (only relevant to transmission)

Throughput – How many bits can be "communicated" per unit time (encompasses e2e communication process, and can be defined over multiple links)

End-to-end delay – Processing + Queueing + transmission + propagation delay.

Store and Forward – entire packet must arrive at router before it can be transmitted

Little's Law

$\boldsymbol{\lambda}$ – average arrival rate

$\boldsymbol{L(t)}$ – Number of customers in the system at time t

$\boldsymbol{t_i}$ **and** $\boldsymbol{t_i^d}$ – arrival and departure time of customer i

$\boldsymbol{W_i}$ – sojourn time of customer i (= $t_i^d - t_i$)

Little's Law: $E[L] = \lambda E[W]$

Statistics Refresher

Consider a random experiment whose outcome cannot be determined in advance.

Sample Space S – the set of all outcomes

Event E – a subset of the sample space

- An event E occurred if outcome $s \in E$

Probability function $P(E)$

- $0 \le P(E) \le 1$
- $P(S) = 1$
- $P(\bigcup_{i=1}^{\infty} E_i) = \sum_{i=1}^{\infty} P(E_i)$ for any sequence of events $E_1, E_2, \ldots$ that are mutually exclusive

A **random variable** X is a function that assigns a real value to each outcome $s \in S$.

For any set of real numbers $A \subseteq \mathcal{R}$,

$$P\{X \in A\} \overset{def}{=} P\big(X^{-1}(A)\big)$$

The **distribution function** F of the r.v. X is defined on any real number x by

$$F(x) \overset{def}{=} P\{X \le x\} = P\big(X^{-1}((-\infty, x])\big)$$

A r.v. is continuous if there exists a **probability density function** $f(x)$ such that $f(x) \overset{def}{=} \frac{d}{dx}F(x)$, i.e. it's distribution function is differentiable

Two random variables are independent if the realization of one does not affect the probability distribution of the other.

$$f_{X,Y}(x, y) = f_X(x) f_Y(y)$$

The expectation or mean of a random variable X:

$$E[X] \overset{def}{=} \int_{-\infty}^{\infty} x f(x)\, dx \quad or \quad \sum_{-\infty}^{\infty} x P\{X = x\}$$

Exponential Distribution

A continuous r.v. T follows an exponential distribution with parameter $\lambda > 0$ if, for $x \ge 0$

- $F(x) = P\{T \le x\} = 1 - e^{-\lambda x}$ or
- $\bar{F}(x) = P\{T > x\} = e^{-\lambda x}$
- $f(x) = \frac{d}{dx}F(x) = \lambda e^{-\lambda x}$

$\boldsymbol{E[T]} = \frac{1}{\lambda}$, where λ is the average arrival rate

An exponential distribution has the **memoryless property**: $P\{T > s + t | T > s\} = P\{T > t\}$

If events occur according to a Poisson process (where the number of events in any fixed interval is Poisson distributed), then the time (or space) between successive events will be exponentially distributed.

Merging two Poisson processes with rate $\lambda_1 and\ \lambda_2$ creates a new Poisson process with $\lambda = \lambda_1 + \lambda_2$

- If X and Y are two i.i.d exponential variables, then $P\{X > Y\} = \int_0^{\infty} P\{X > Y | Y = y\} f_Y(y) dy = \frac{\lambda_y}{\lambda_x + \lambda_y}$

Network Queuing Models

Service time $\boldsymbol{S_i}$

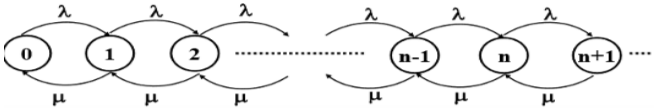
- processing time of packet i under a fixed link rate
- follows i.i.d. r.v. S with mean $E[S] = 1/\mu$
- μ – service rate,

M/M/1 model

- Markovian arrival time/ Markovian service time/ number of service counters
- ρ – utilization rate or percentage time that server is busy, equals to $\frac{\lambda}{\mu}$;
  - requires condition of $\lambda < \mu$ for system stability

Deriving $\boldsymbol{\rho}$ **using Little's law**

Birth-Death Process



- State transition diagram for M/M/1 system
- Define system state L as the (random) number of users in the system
- $\boldsymbol{\pi_i} = P\{L = i\} = \rho^i(1 - \rho)$
  - probability that exactly i packets or customers in the system (server + queue)
- L follows a Geometric distribution
- $E[L] = \frac{\rho}{1-\rho} \rightarrow \rho = \frac{E[L]}{1+E[L]}$
- $E[W] = \frac{1}{\mu - \lambda}$ (derived from applying Little's Law)
- $E[Q]$ – average number of packets in queue $E[Q] = E[L] - \rho = \frac{\rho^2}{1-\rho}$
- $E[D]$ – Average queuing delay of packets $E[D] = E[W] - E[S] = \frac{\rho}{\mu - \lambda}$

Burke's Theorem – If an M/M/1 system with an arrival rate of λ starts in a steady state,

- the departure process is Poisson with rate λ
- the number of customers in the system at any time t is independent of the sequence of departure times prior to t

Effective arrival rate into a stable subsystem i: $\lambda_i = r_i + \sum_{j=1}^{n} \lambda_j P_{ji}$

For the entire system with n subsystems, we can represent this in a matrix form:

$$\boldsymbol{\lambda = r + P\lambda} \quad \Rightarrow \quad \boldsymbol{\lambda = r(I - P)^{-1}}$$

Where $\boldsymbol{\lambda}, \boldsymbol{r}$ are both $n \times 1$ vector, $\boldsymbol{P}$ is a $n \times n$ matrix

In a stable M/M/1 system, throughput is positively correlated with queuing delay.

Resource Allocation

- **Internet** provides best-effort service
  - No guarantee for packet delivery
  - Suitable for elastic traffic but not suitable for real-time applications
  - Solution: allocate different amount of resource to different application flows
  - From a **utility** perspective: User's happiness U_i as a function of performance metrics, e.g. delay or throughput

Max-min fairness

- A feasible allocation is max-min fair if and only if an increase of any rate within the feasible domain must be at the cost of a decrease of an already smaller or equal rate.
- $\boldsymbol{x}$ is max-min fair if for any feasible $\boldsymbol{y}$, if $y_i > x_i$, then $\exists j$ such that $y_j < x_j \le x_i$

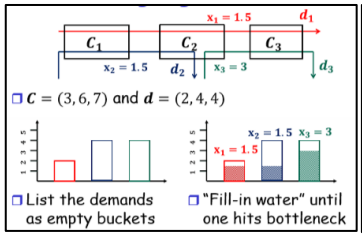
Bottleneck Resource

A resource r is a bottleneck resource for flow i if and only if

- Resource r is saturated and
- Flow i has the maximum rate among all flows using resource r
- Flow i cannot get more resource r if allocation is fair; Else, it hurts flows with lower rates

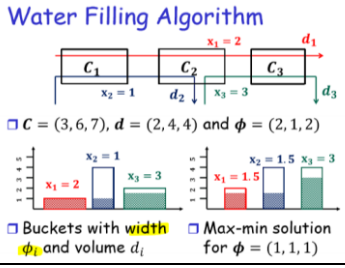
Theorem: When each flow has an infinity demand under a network system, a flow allocation is max-min fair if and only if every flow has a bottleneck resource.

Water filling algorithm



Weighted Max-Min Fair

Given a weight vector $\boldsymbol{\phi} = (\phi_1, \phi_2, \ldots)$



Summation of Geometric series

Infinite geometric sum

$s = a + ar^1 + ar^2 + \cdots = \sum_{k=0}^{\infty} ar^k = \frac{a}{1-r}$

Finite Geometric sum:

$s = a + ar + \cdots + ar^{n-1}$

$= \sum_{k=0}^{n-1} ar^k = a\frac{1-r^n}{1-r}$

Derivative/Integral of Exponential

$\int e^{ax+b}\, dx = \frac{1}{a}e^{ax+b} + c$

$\frac{d}{dx}e^{f(x)} = f'(x)e^{f(x)}$

L4 – SDN Design Principles

- SDN is a new approach to networking, particularly for implementing and managing networks.
- It is based on new fundamental principles and significantly impacts the pace of innovation.
- It differs from traditional networking and addresses existing issues with the current internet infrastructure.

The Internet's Development and Challenges:

- The internet has transitioned from a research experiment to a global infrastructure.
- Brilliant at under specifying
  - E.g., IP Layer does not specify much other than best effort delivery
- Innovations are easy at the edge (e.g., web, P2P, VoIP, social networks) but **challenging within the network**
- Within the network, issues include **closed equipment**, **slow protocol standardization**, and limited innovation opportunities.

Difficulties in Network Management:

- Operating a network is expensive and complex.
- Software in network hardware equipment is often buggy and prone to vulnerabilities
- Networks, especially in data centers and home setups, are hard to manage and evolve.
  - Networks are harder to manage compared to virtualized systems like computation and storage.
  - Network tech tends to evolve slowly, with outdated routing algos & primitive network management.

Networking as a Discipline:

- In contrast to other system fields (OS, DB, DS), networking teaches a broad range of protocols, is difficult to manage, and evolves slowly.

Principles of SDN:

Separation of Control Plane and Data Plane / Disaggregation: This principle suggests that the control (routing) and forwarding functions of network devices should be separated.

- **Data Plane:** Process/deliver packets based on state in routers and endpoints
- **Control plane**: Establish router state
  - Determine how and where packets are forwarded
  - Routing, traffic engineering, firewall
- In **Disaggregation**, the control plane communicates with the data plane via a southbound API
  - Network operators are able to purchase control & data planes from separate vendors
  - Data plane now consists of cheaper commodity forward devices
  - In OpenFlow switches, it uses FlowRules, a general purpose way for the control plane to tell data plane to forward packets in a particular way (i.e. a forwarding abstraction)

Logically Centralized Control: The control plane should be logically centralized, allowing for easier and more effective management of network resources.

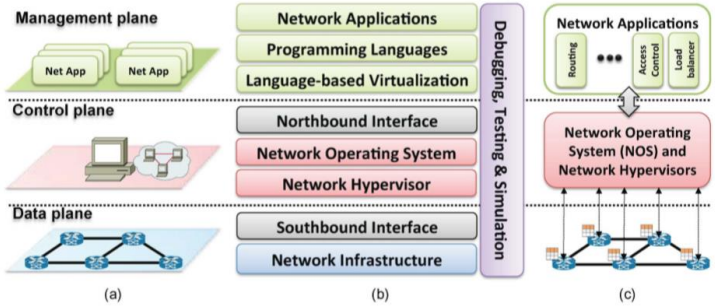
- In contrast to each individual switch having their own controller in the control plane

Programmability: Setting up a Northbound API between a control application & centralized control plane

- **Control**: Making real-time decisions about how to manage traffic, e.g. respond to link/switch failures etc.
  - Control plane needs to learn about failure and provide remedy within milliseconds, no human intervention involved.
- **Configuration**: Setting up & modification of network behaviours and policies, human intervention present

Benefits of SDN:

- SDN allows for easier network management, rapid innovation, and a shift in control from vendors to operators and ultimately to users.
- It simplifies network programming and diagnostics.



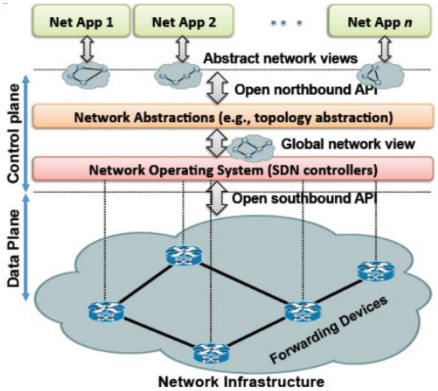
L5 – SDN Examples

ONOS Model : Open northbound API, Specification Abstraction

ONOS System: Distribution abstraction

OpenFlow Protocol: Open southbound API

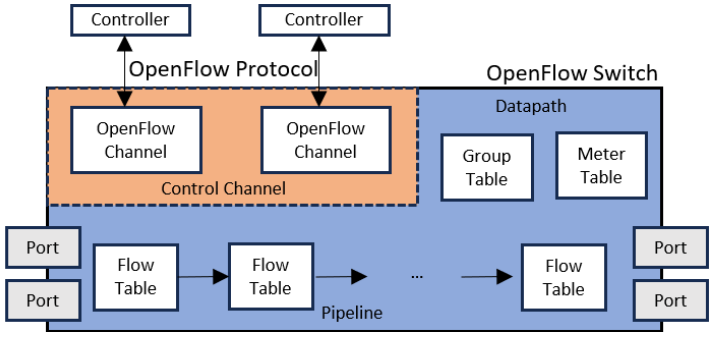
OpenFlow Switch: Forwarding abstraction

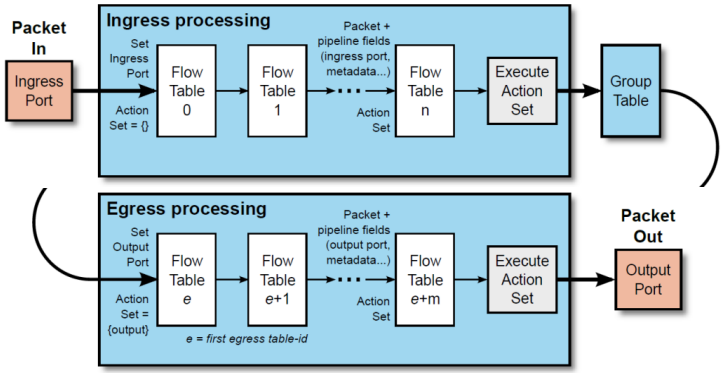


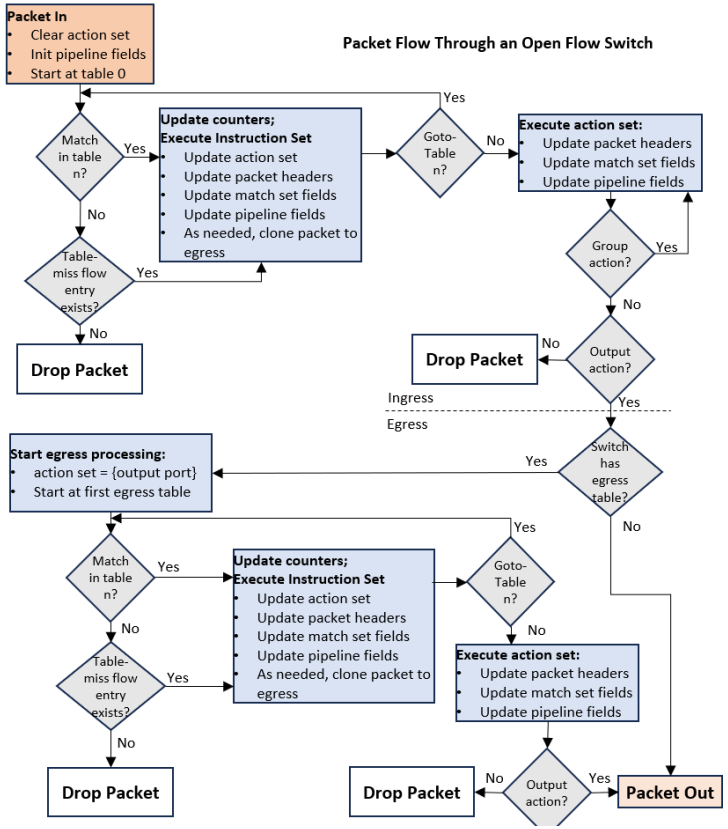
L7 - BGP Subprefix Hijacking

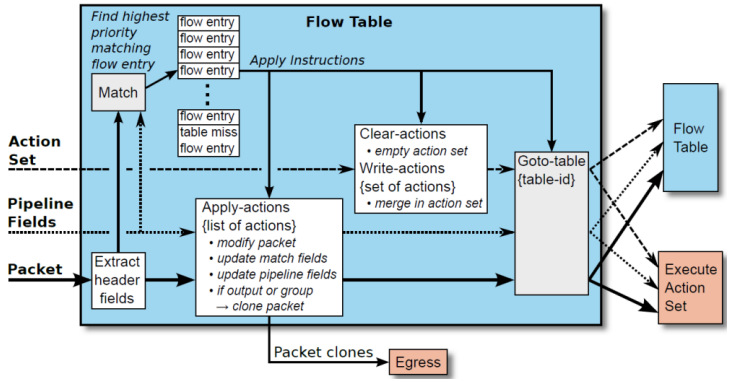
Rogue AS advertises prefix it does not own, allowing for blackholing, snooping & impersonation of data traffic

Rogue AS can also advertise a more specific prefix to redirected all traffic to it due to longest prefix matching.









For each packet from a packet Flow
- Header & Pipeline (vals attached to the pkt during pipeline processing, e.g. ingress port & metadata) fields
- Action: An operation that acts on a packet (e.g. drop, forward to port, modify/decrease TTL)
- Action set: Accumulated while processed by flow tables, executed at the end of the pipeline processing

OpenFlow flow entry
- A flow entry in a flow table looks like

Match Fields	Priority	Counters	Intstructions	Timeouts

  - **Match Fields**: Packets are matched against header fields and pipeline fields
  - **Priority**: used to choose from multiple matches
  - **Counters**: counter how many packets got matched by this flow entry
  - **Instructions**: contains a list of actions to apply immediately, contains a set of actions to add to the action set, modifies pipeline processing (e.g. go to another flow table)
  - **Timeout**: How long this flow entry should stay in the switch
- **Default flow entry:** Table-miss flow entry – for when none of the flow entries match
  - Could either drop the packet or forward as per normal to the next flow table

Architecture of ONOS:

Northbound Interfaces (NBI): Allows applications to stay informed about the network state (e.g., topology, packet interception) and control the network data plane.

Distributed Core: Responsible for managing the network state and notifying apps about relevant changes.
- Includes a scalable key/value store (Atomix) for internal state management

Southbound Interface: Constructed from a set of plugins, shared protocol libraries, and device-specific drivers.
- Facilitates communication between the ONOS core and various network devices, translating device-level information into higher-level abstractions.

ONOS Abstractions:

Intent: High-level, network-wide programming constructs that are topology-independent.
- Enables the specification of desired nw behaviors without worrying about underlying nw configurations.

Flow Objective: Finer-grained, device-centric programming constructs that are pipeline-independent.
- Allows for more granular control of network devices, acting as an intermediary between high-level intents and low-level flow rules.

Flow Rule: Used to control various pipelines, both fixed-function and programmable.

L6 – Interconnections

Autonomous Systems (AS):

An AS is a network of interconnected routers. Each AS is identified by a unique AS Number (ASN) and is controlled by a single administrative domain. They are able to participate in the global exchange of information and traffic on behalf of other networks. ASes use a common routing protocol and policy.
- ASNs assigned by the Internet Assigned Numbers Authority (IANA), and has 5 regional internet registries

Internet Peering: Peering is the voluntary interconnection of administratively separate Internet networks for exchanging traffic. Peering agreements can be bilateral (between neighbor ASes) or multilateral and depend on business relationships such as customer-provider or peer-to-peer. Often a settlement-free relationship

Valid AS Paths
- Providers provides transit services for customers. Peers agree to provide transit services between their respective customers, but between peers.
- **Valley-free property:** Single peak (uphill + downhill) or single flat top (uphill + 1 peering + downhill), or any subpath of the above
- **Invalid**: provider→cust→peering / provider→cust→provider / peering→peering / peering→provider

Internet Exchange Point (IXP)

An open & neutral location where ASes freely interconnect to exchange traffic. Enables cheap, bilateral peering. It saves upstream transit costs, keeps traffic local, results in better network performance and QoS, and scalability. Any network that has its own address space, AS number and transit agreements can join.
- **Location**: The IXP must be located at a neutral, secure, accessible, safe, expandable location.
- **Operation**: Requires neutral management body, typically a consortium of all participants in the IXP
- **Funding**: Costs agreed and covered equally by participants. Landlord contributes as IXPs bring business

L7 – BGP

Routing Algorithms:

Link state routing: All routers have complete topology and link cost info, using algos like Dijkstra's (OSPF).
- Topology information must be flooded on network for all routers to have complete view of network
- Entire path is computed locally per node, resulting in high processing overheads
- **Cons:** minimizes some notion of `distance` only if all nodes agree on the notion

Distance vector routing: Routers know neighbor link costs, using decentralized algos like Bellman-Ford (RIP).
- **Pros:** hides details of the network topology; only next hop is determined per node
- **Cons:** Slow convergence; minimizes some notion of `distance` only if all nodes agree on the notion

Path Vector routing: Extension of DV routing; Advertises the entire path
- DV: send distance metric per destination d
- PV: send entire path for each destination d, allows each node to compute their own notion of cost.
- **Pros:** Faster Loop detection – node can check for loops

Border Gateway Protocol

BGP Session: Two BGP routers (aka peers/speakers) advertises paths to different destination network prefixes
- TCP port 179, Initially exchange all active routes, subsequently only exchange incremental updates

eBGP: Exchange reachability info from neighbouring ASes, peers need to be directly connected
- When AS3 advertises and IP prefix to AS1, it promises to forward packets towards that prefix.

iBGP: Propagates reachability info across backbone; carries ISP's own customer prefixes
- Peers doesn't need to be directly connected since it runs on top of IGP(either RIP/OSPF)
- iBGP peers are fully meshed → any iBGP speaker can broadcast to all other iBGP peers in the network
  - An iBGP peer can pass on prefixes learned from outside AS to other peers
  - Does not pass on prefixes learned from other iBGP speakers.

BGP Messages:

Marker (16 bytes) - used for synchronization and authentication	
Length (2) - len of message in bytes	**Type(1) –** OPEN/UPDATE/KEEPALIVE/NOTIFICATION
Withdrawn Routes Length	**Withdrawn routes –** IP prefixes for the routes withdrawn - Can withdraw multiple routes in an UPDATE message
Path Attribute length	**Path Attributes**
Network Layer Reachability Info (variable**) –** IP prefixes that can be reached from the advertised route - Can only advertise one feasible route	

- **OPEN**: Opens TCP connection to peer and authenticates sender.
- **UPDATE**: Advertises new paths or withdraws old paths.
- **KEEPALIVE**: Keeps connection alive and acknowledges OPEN requests.
- **NOTIFICATION**: Reports errors and closes connections.

Withdrawn Routes – Do not expire; actively withdrawn by original advertiser or replaced by existing routes; All routes from a peer become invalid when peer goes down (using the keepalive message)

Well Known Mandatory Path Attributes
- **mandatory** - must be included in UPDATE msgs containing NRLI
- **well known** – once a well-known attribute is updated by a BGP peer, it must pass them to its peers

ORIGIN – conveys the origin of the prefixes, learned from (i (IGP), ? (INCOMPLETE), e (EGP))

AS-PATH – contains ASes through which NRLI has passed, expressed as a sequence (e.g. AS79, AS11, …)

NEXT-HOP – indicates IP address of the router intrface that begins the AS PATH

LOCAL_PREF: 4 byte unsigned int (default 100); for BGP speaker to inform internal peers of its degree of preference for a route

MULTI_EXIT_DISC (MED): 4byte u_int (default 0); for BGP speaker to discriminate among multiple entry points to a neighbouring AS to control inbound traffic. If received over eBGP, may be propagated over iBGP but not to further neighbouring ASes.

COMMUNITY: Used to group destinations; useful for applying import/export policies based on this attribute

Getting Entries in Forwarding Table

1. **Router becomes aware of prefix** via BGP route advertisements from other routers
2. **Determine router output port for prefix**
   - Chooses best inter-AS route from BGP
   - Use OSPF to find best intra-AS route leading to the border router of the chosen inter-AS route
   - Identify port number for best route
3. **Enter prefix-port entry in forwarding table**

Best Route Selection

1. Calculation degree of **preference** using LOCAL_PREF or preconfigured policy
2. Best Route Selection: Highest LOCAL_PREF → Smallest number of ASNs in AS_PATH / Lowest origin number in ORIGIN / Most preferred (typically lowest) MED / lowest interior cost based on NEXT_HOP