Minimizing User Wait Time During Resource Crunch

Weekly Meetings

Rafael Lourenço

March 23, 2018

UC DAVIS
UNIVERSITY OF CALIFORNIA

Outline

- General Motivation
- Previous Work: “Running the Network Harder: Connection Provisioning under Resource Crunch”
- Minimizing User Wait Time under Resource Crunch

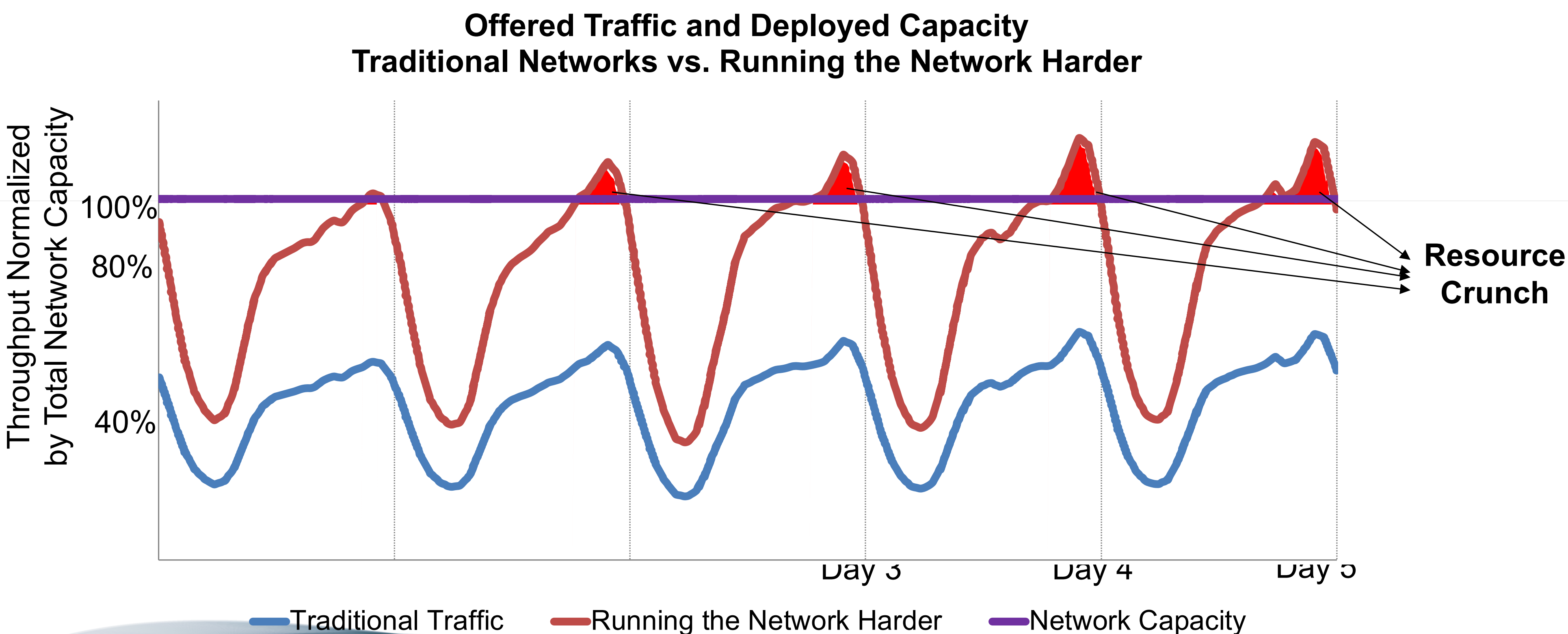
Resource Crunch

- Traditional Networks deploy excess capacity that:
  - Provides redundancy
  - Accommodates traffic fluctuations
  - Absorbs traffic growth
- **Flexibility introduced by new technologies (e.g. SDN) are allowing higher capacity utilization**
  - Microsoft, Google, and others report more than 60% average link utilization

- F. Dikbiyik, L. Sahasrabuddhe, M. Tornatore, and B. Mukherjee, “Exploiting excess capacity to improve robustness of WDM mesh networks,” *IEEE/ACM Trans. Netw.*, 2012.
- C.-Y. Hong *et al.*, “Achieving high utilization with software-driven WAN,” in *Proc. ACM SIGCOMM*, 2013
- S. Jain *et al.*, “B4: Experience with a globally-deployed software defined WAN,” *Proc. ACM SIGCOMM*, 2013.

UCDAVIS

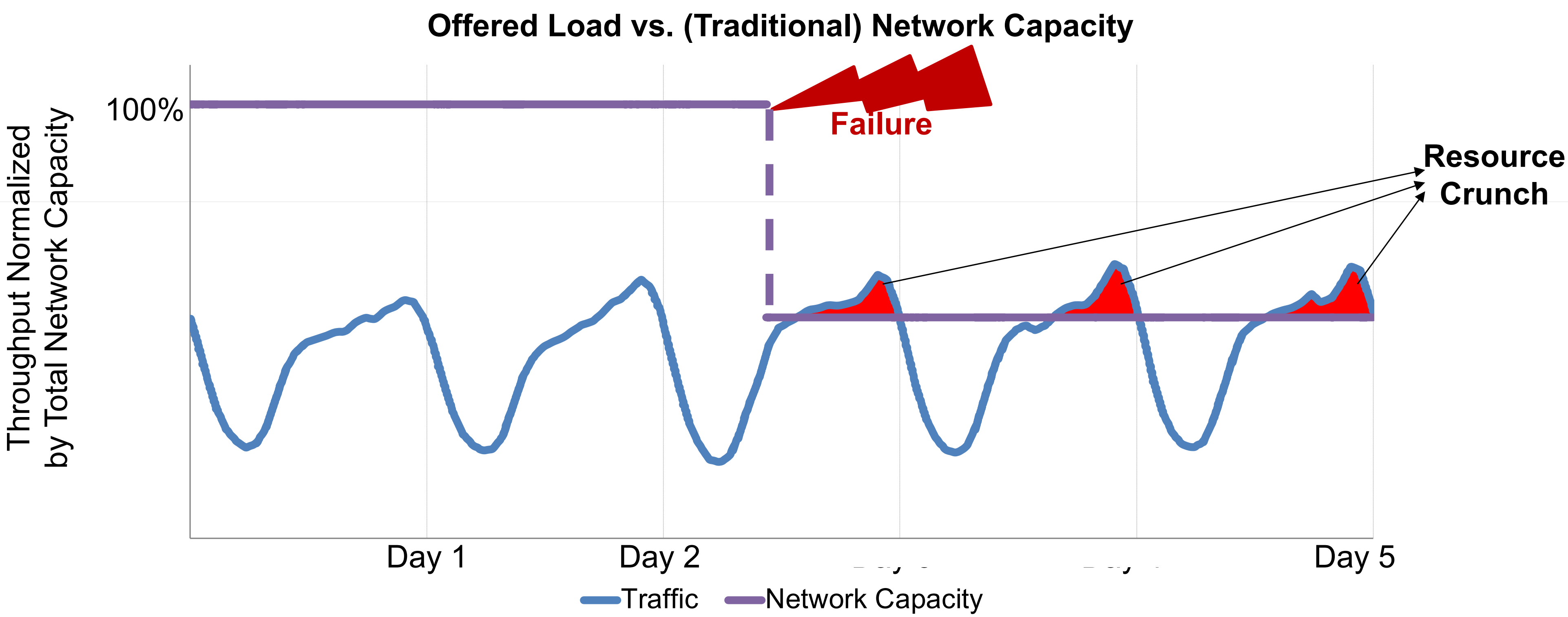
Resource Crunch: Traffic Fluctuations



- Based on the traffic report of the LINX: London Internet Exchange, from April 7 to 12, 2017.
- C. Labovitz *et al.*, "Internet inter-domain traffic," in *Proc. ACM SIGCOMM*, 2010.
- I. Ari, B. Hong, *et al.*, "Managing flash crowds on the Internet," in *Proc. IEEE/ACM MASCOTS*, 2003.

UCDAVIS

Resource Crunch: Disasters



- Based on the traffic report of the LINX: London Internet Exchange, from April 7 to 12, 2017.

Running the Network Harder: Connection Provisioning under Resource Crunch

- ***Resource Crunch:*** situations where offered traffic cannot possibly be carried by the network

- During Resource Crunch, if a connectivity demand arrives, the network probably won't be able to provision it using its normal allocation procedures
  - We call such demands a ***crunched demand***

- To deal with Resource Crunch, flexibility must be introduced in the system:
  - We consider that demands have flexible bandwidth requirements (i.e., required bandwidth and minimum acceptable bandwidth)
  - Connections can undergo ***service degradations***
  - ***Crunched demands*** are initially allocated their minimum required bandwidth, if possible (and are upgraded as soon as possible)

UCDAVIS

Service Classes

Examples of service classes:

- **Interactive**: directly impact end user experience (e.g., serving a user query), and cannot suffer degradation. These services have highest impact on revenue
- **Elastic**: more flexible than Interactive, end users either have more flexibility (making a video call, or sending an e-mail), or are not directly impacted by them (replicating data update between Data Centers). These services can be degraded and have less impact on revenue than Interactive
- **Background**: relate to maintenance activities that are not directly accessible to end users (backup migration, synchronization, configuration, etc). Can be significantly degraded (more than Elastic) and have the smallest impact on revenue

- C.-Y. Hong *et al.*, "Achieving high utilization with software-driven WAN," in *Proc. ACM SIGCOMM*, 2013.
- Y. Chen *et al.*, "A first look at inter-data center traffic characteristics via Yahoo! datasets," in *Proc. IEEE INFOCOM*, 2011.

Revenue and Service Classes

- Revenues are proportional to geographical distances of source and destination of a demand (not considering distances (or hops) is detrimental to long paths)
- Blocking a demand might negatively impact revenue (blocking cost)
- Microsoft Azure: $0.09 USD per GB; Average US-wide source-to-destination distance: 1500km

$$\frac{\$0.0000075}{\text{Gbit} \cdot \text{km}} \times 1500 \text{ km} \times 1 \text{ Gbps} \times 8 \text{ s} = \$0.09$$

Service Class	% of Total Traffic	Requested Bandwidth per Demand	Minimum Required	Revenue Increase ($/(Gbit · km))	Cost of Blocking ($/(Gbit · km))
Interactive	15%	2 Gbps	2 Gbps	0.0000075	0.00000375
Elastic	25%	3 Gbps	2 Gbps	0.000006	0.000003
Background	60%	5 Gbps	2 Gbps	0.000000375	0.0

- C.-Y. Hong *et al.*, “Achieving high utilization with software-driven WAN,” in *Proc. ACM SIGCOMM*, 2013.
- Y. Chen *et al.*, “A first look at inter-data center traffic characteristics via Yahoo! datasets,” in *Proc. IEEE INFOCOM*, 2011.
- Microsoft.(2017)Microsoft Azure: Bandwidth Pricing Details. [Online]. Available: https://azure.microsoft.com/en-us/pricing/details/bandwidth/

Previous Problem Statement

Given

- Network to
- A set of all s;
- A **crunche** ng
  cost.

Output

A decision f
other dema
place the c

Goal

Maximize profits, measured by revenue generated from served connections after
subtracting

Constraints

Link capac
bandwidth,

In other words…

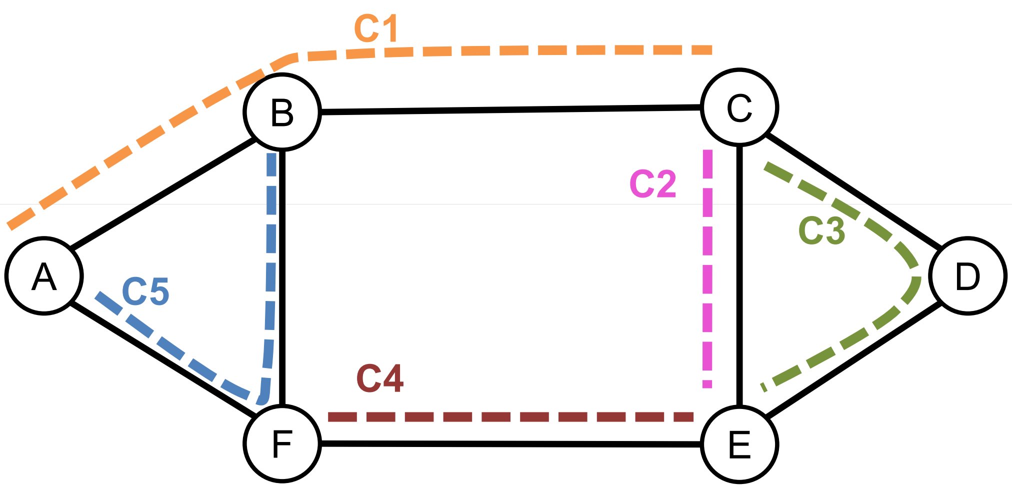
To provision a **crunched demand**, we propose degrading the bandwidth of allocated connections to make room for the crunched demand.

Considering our goal: which other connections should be degraded for that?

Optimum Results: find the set of allocated connections whose degradation will generate the smallest possible decrease in revenue.

UCDAVIS

Illustrative Example



All links have 20 Gbps capacities.
Network is in Resource Crunch state.

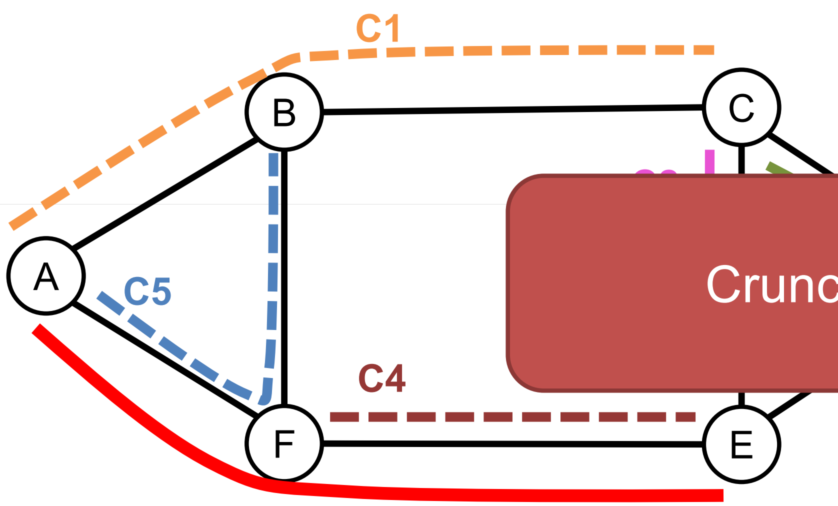
Connection	Requested Bandwidth	Minimum Bandwidth	Revenue Per Gbit
C1	20 Gbps	10 Gbps	\$3
C2	20 Gbps	10 Gbps	\$3
C3	20 Gbps	10 Gbps	\$2
C4	20 Gbps	10 Gbps	\$9
C5	20 Gbps	10 Gbps	\$1

A demand from A to E arrives. It requests minimum 10 Gbps, offers \$6 per Gbit, has blocking cost \$30.

It is crunched.

UCDAVIS

Illustrative Example: Shortest Path



Connection	Requested Bandwidth	Minimum Bandwidth	Revenue Per Gbit
[illegible]	[illegible]	10 Gbps	$3
[illegible]	[illegible]	10 Gbps	$3
[illegible]	[illegible]	10 Gbps	$2
[illegible]	[illegible]	10 Gbps	$9
C5	20 Gbps	10 Gbps	$1

Crunched demand is blocked

Shortest path from A to E => Degrade C4 and C5

Revenue lost

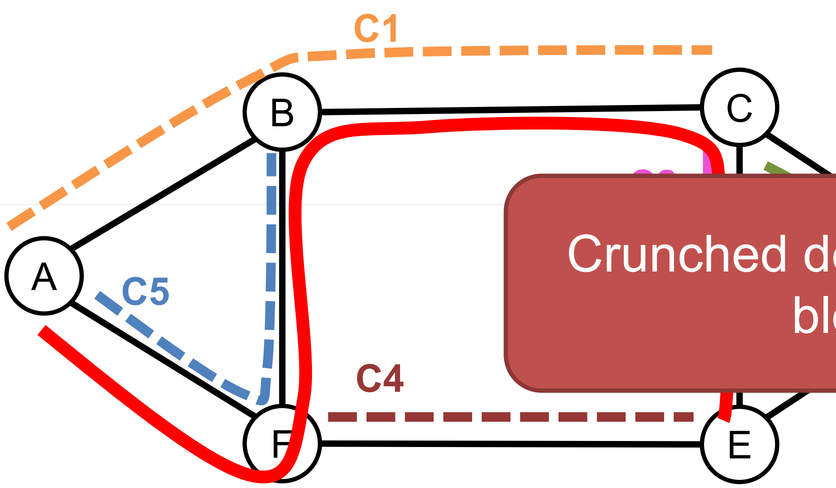
$1x10 + $9x10 = $100

Revenue increase + Blocking cost

$60 + $30 = $90

- S. Savas, C. Ma, M. Tornatore, and B. Mukherjee, “Backup reprovisioning with partial protection for disaster-survivable software-defined optical networks,” *Photonic Network Comm.*, 2016.
- A. Roy, M. F. Habib, and B. Mukherjee, “Network adaptability under resource crunch,” in Proc. IEEE ANTS, 2014.
- Z. Zhong, J. Li, N. Hua, G. B. Figueiredo, Y. Li, X. Zheng, and B. Mukherjee, “On QoS-assured degraded provisioning in service- differentiated multi-layer elastic optical networks,” in *IEEE GLOBECOM*, Dec 2016.

Illustrative Example: Shortest Path (Cost)



Connection	Requested Bandwidth	Minimum Bandwidth	Revenue Per Gbit
C1	20 Gbps	10 Gbps	$3
[illegible]	[illegible]	10 Gbps	$3
[illegible]	[illegible]	10 Gbps	$2
[illegible]	[illegible]	10 Gbps	$9
C5	20 Gbps	10 Gbps	$1

Crunched demand would be blocked if its blocking cost was $0.

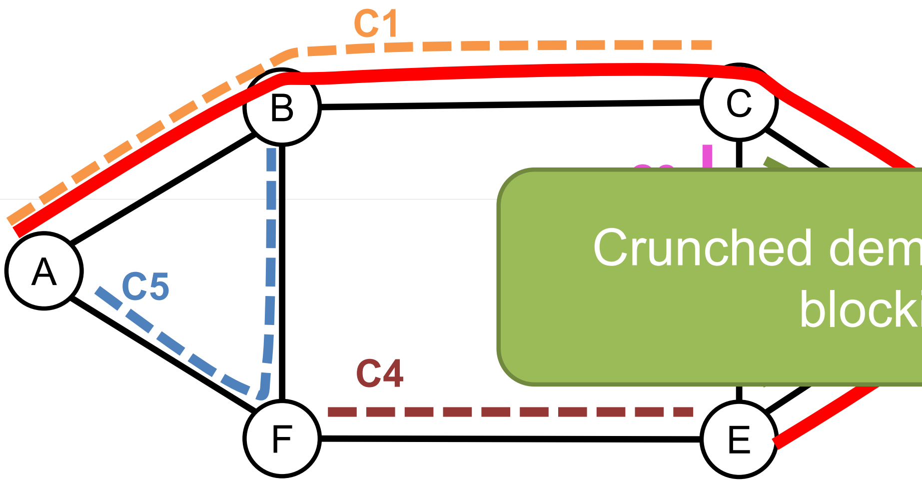
Shortest path from A to E using as weights degradation costs => Degrade C1, C2, C5

Revenue lost
$1x10 + $3x10 + $3x10 = $70

Revenue increase + Blocking cost
$60 + $30 = $90
If no block cost: $60 + $0 = $60

UCDAVIS

Illustrative Example: Optimum Solution



Connection	Requested Bandwidth	Minimum Bandwidth	Revenue Per Gbit
[illegible]	[illegible]	10 Gbps	$3
[illegible]	[illegible]	10 Gbps	$3
[illegible]	[illegible]	10 Gbps	$2
[illegible]	[illegible]	10 Gbps	$9
C5	20 Gbps	10 Gbps	$1

Crunched demand is served even if its blocking cost was 0.

Optimum Solution => Degrade C1 and C3

Revenue lost

$3x10 + $2x10 = $50

Revenue increase + Blocking cost

$60 + $30 = $90

If no block cost: $60 + $0 = $60

Connection Adjacency Graph - CAG

1. For each connection that has some degradable capacity, create a vertex (called a c-vertex) and annotate in it all physical nodes that connection touches

2. If there are … that capacity by adding a … ces

3. Add edge … ated in both of them. Add …

4. Associate … e bandwidth unit the co… inting to l-vertices

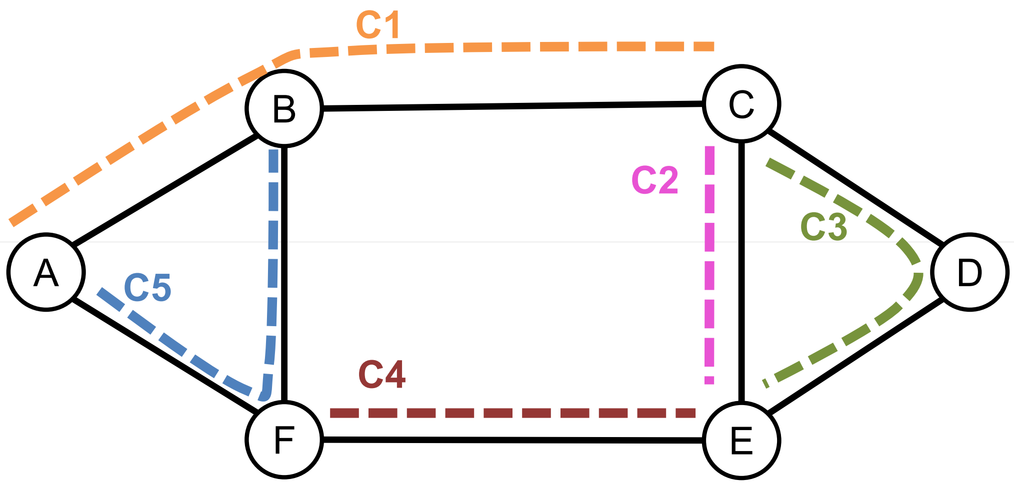
5. When a demand is crunched, add a dummy source vertex containing only that demand's source. Connect it to all other vertices that contain that node. Do the same for the destination node. After the demand is served or blocked, remove the dummy vertices

The CAG enables the decoupling of the problem:

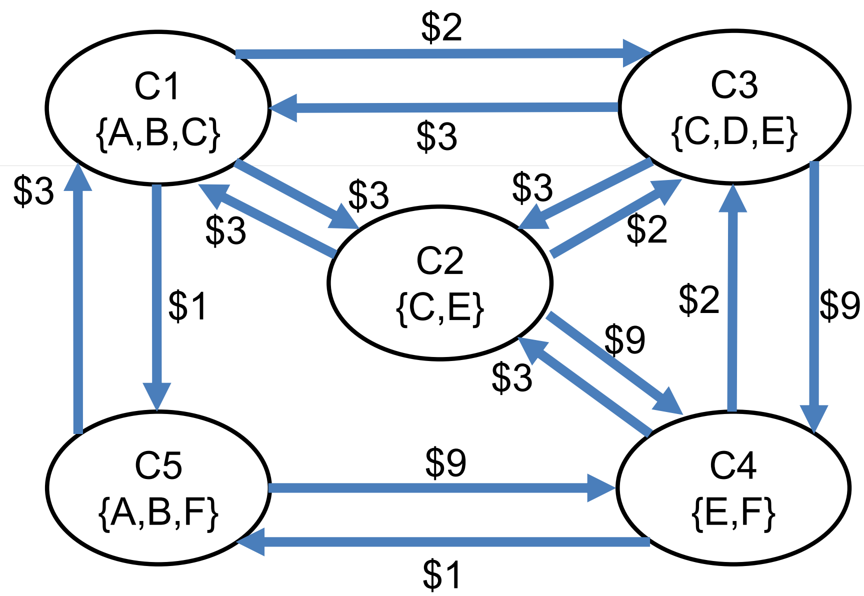
- ***First, find a cheap degradation***
  - ***Then, find a physical path***

UC DAVIS

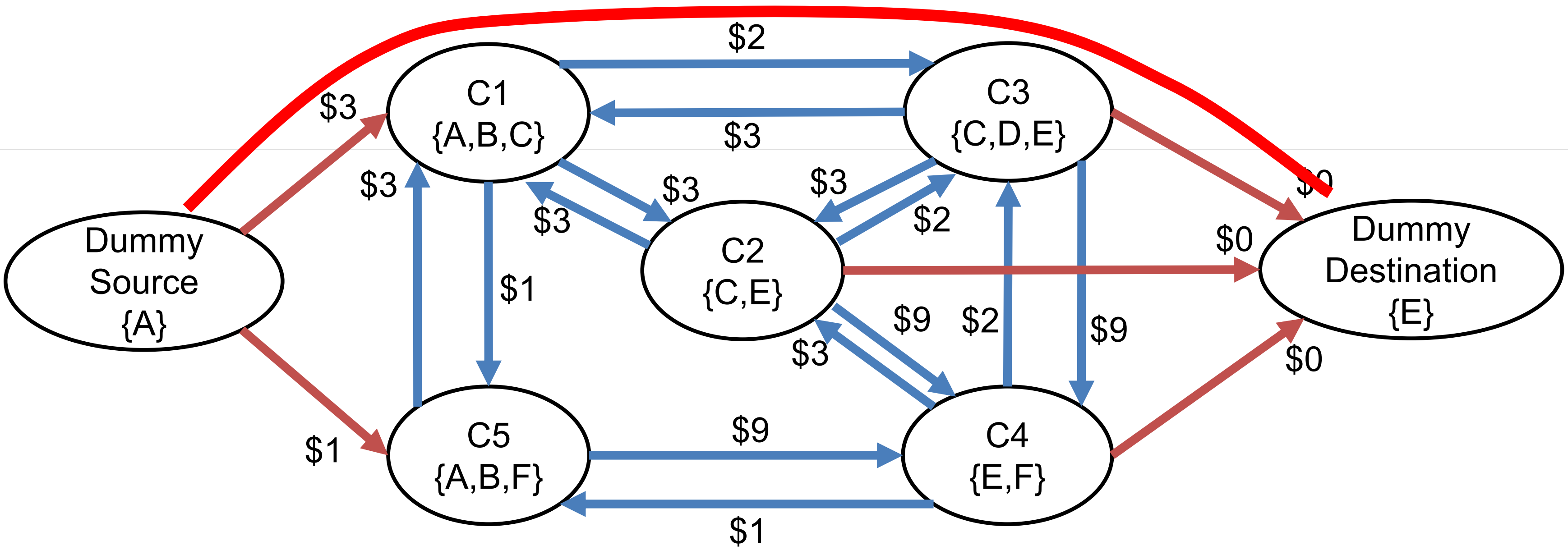
Illustrative Example - CAG



Connection	Revenue Per Gbit
C1	$3
C2	$3
C3	$2
C4	$9
C5	$1



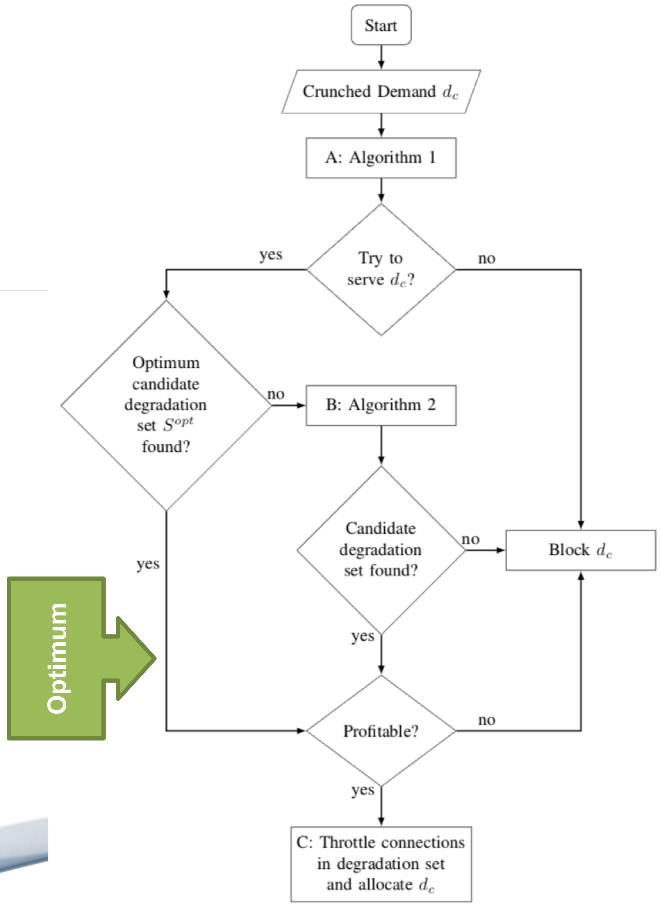
Illustrative Example – CAG
Crunched Demand from **A to E**



Find the cheapest path on the CAG.

UCDAVIS

The PROVISIONER Algorithm:



Algorithm 1 Find Cheapest Capacitated CAG Path

Input: Crunched demand d_c and CAG
Output: Block d_c, or Optimum set S^{opt}, or Candidate set S^{good}, or null

Add dummy source and destination vertices to CAG
Find cheapest CAG path P from dummy source to destination (Dijkstra)
if P not found **then**
  Block d_c
end if
$L(S^{opt}) =$ cost of path P
Find cheapest CAG path P^c with capacity at least $B_{d_c}^{min}$
if P^c not found **then**
  return null
end if
Create Set $S^{candidate}$ with all connections in P^c
if $L(S^{candidate}) = L(S^{opt})$ **then**
  return Optimum Degradation Set $S^{opt} = S^{candidate}$
else
  return Degradation Set $S^{good} = S^{candidate}$
end if

Algorithm 2 Cheapest Degradation Among k-Shortest Paths

Input: crunched demand d_c, k-shortest paths P^S, Degradation set S^{good}
Output: Cheapest degradation set found S^{final} or null

$X^* = \infty$
$S^{LP} =$ Place-holder for best degradation found by LP
for all $P \in P^S$ **do**
  Construct LP input set C based on path P
  Construct LP model M_P
  $X =$ Solve M_P
  if $X < X^*$ **then**
    $X^* = X$
    $S^{LP} =$ Get degradation set from LP
  end if
end for
if S^{LP} is empty and S^{good} is empty **then**
  return null
end if
if $L(S^{LP}) < L(S^{good})$ **then**
  return $S^{final} = S^{LP}$
else
  return $S^{final} = S^{good}$
end if

$$\text{minimize} \quad \left(X = \sum_{c_i \in C} r_{c_i}(B_{c_i}) - \sum_{c_i \in C} r_{c_i}(y_{c_i}) \right)$$

subject to

$$B_{c_i}^{min} \le y_{c_i} \le B_{c_i}, \qquad \forall c_i \in C$$

$$\sum_{c_i \in C} y_{c_i} \cdot v_{c_i}^{l_j} \le z_j - B_{d_c}^{min}, \qquad \forall l_j \in P$$

$$X \le X^*$$

What if…

1. We are not allowed to degrade (throttle) connections
2. We can, however, schedule crunched demands for a future time

Then, we can ask this question:

"How can I schedule the crunched demand for a future time, such that it has to wait the least possible amount?"

UCDAVIS

Minimizing User Wait Time under Resource Crunch Problem Statement

Given
- Network
- A set of
- A **crunc**

Output

The am through the dem

Goal

Minimize

Constraint

Link capac ned demand bandwidth, all connections are non-splittable.

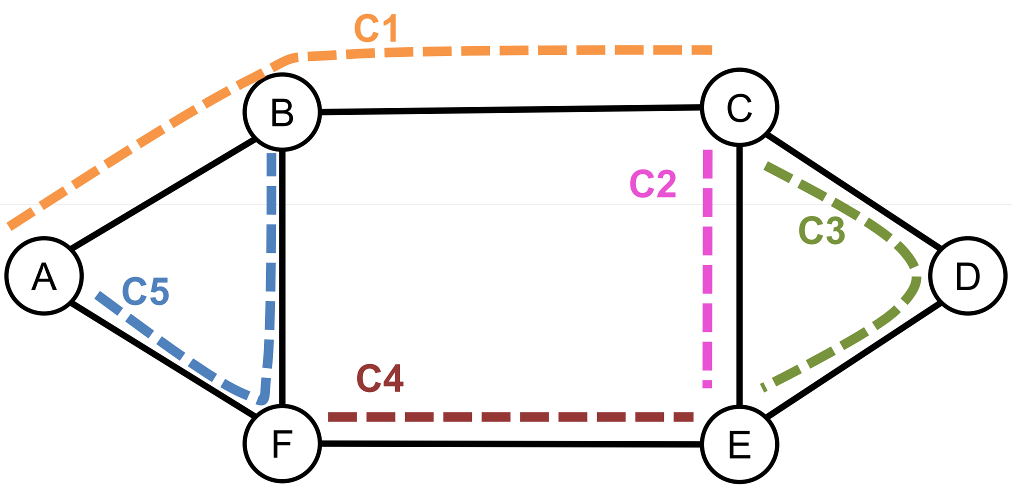
In other words…

To provision a **crunched demand**, we propose finding the minimum amount of time that this demand must wait before being served.

Good news: We can solve this optimally.

UCDAVIS

Illustrative Example 2



All links have 20 Gbps capacities.
Network is in Resource Crunch state.

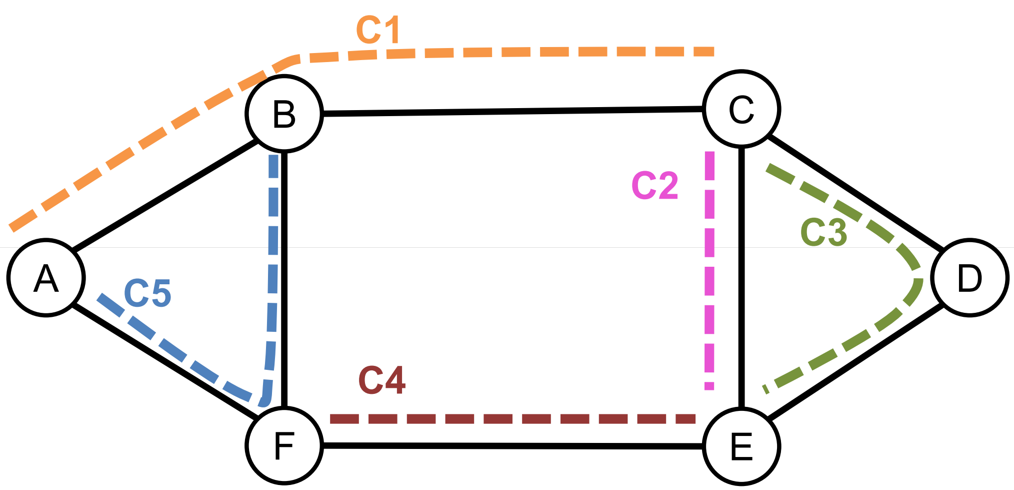
Connection	Used Bandwidth	Remaining Holding Time
C1	20 Gbps	3 s
C2	20 Gbps	4 s
C3	20 Gbps	5 s
C4	20 Gbps	6 s
C5	20 Gbps	7 s

A demand from A to E arrives. It requests 10 Gbps and needs to be served as soon as possible.

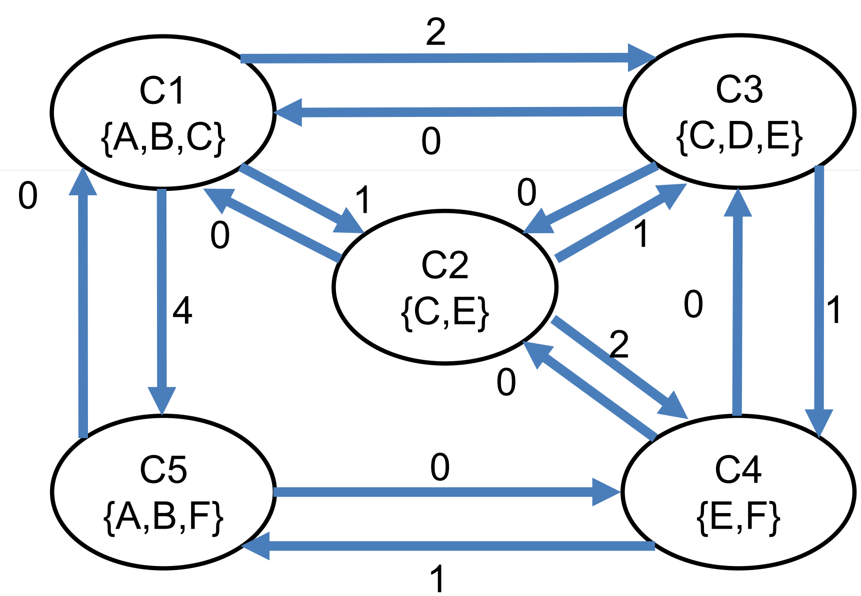
It is crunched.

UCDAVIS

Connection's Times-to-Complete Graph

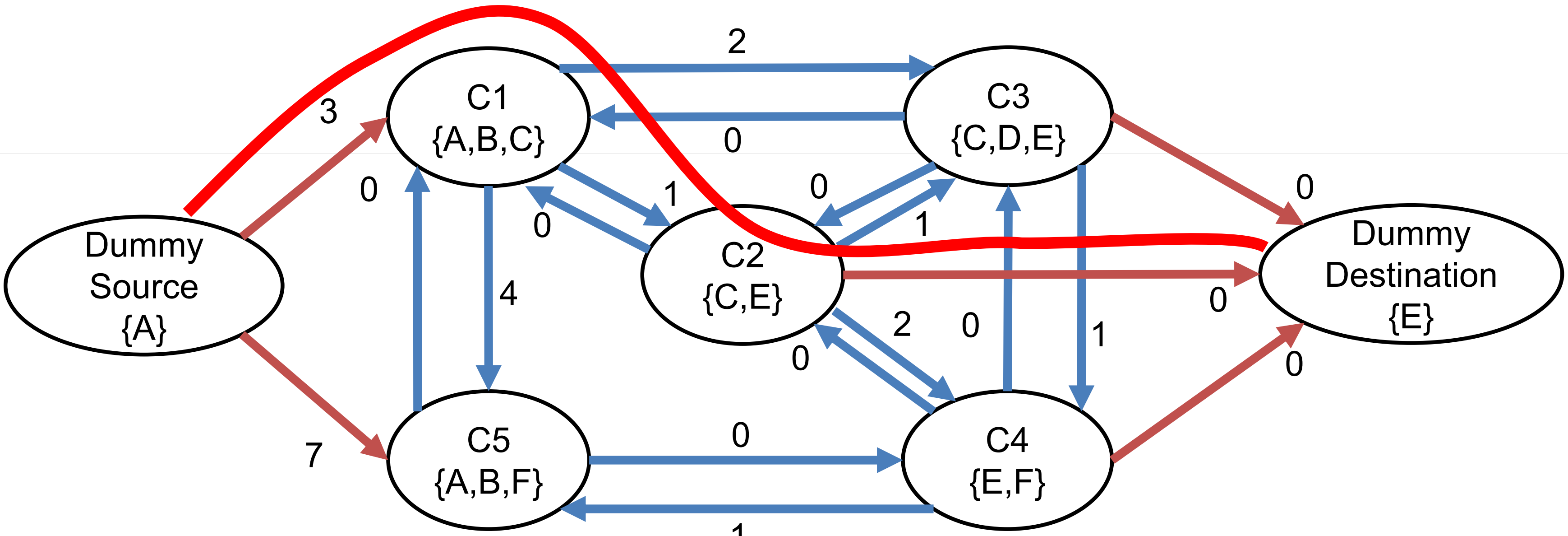


Connection	Remaining Holding Time
C1	3 s
C2	4 s
C3	5 s
C4	6 s
C5	7 s



UCDAVIS

Illustrative Example 2 – CTCG Crunched Demand from **A to E**



Find the cheapest path on the CTCG.
The shortest wait for the crunched demand is 4 seconds.

CTCG Pros and Challenges

Pros	Challenges
Find optimum solution for one crunched demand	As different crunched demands arrive, they might require re-ordering
Finds both the amount of time it must wait, and the route that through which it will be served	If allocated connections are upgraded when others depart, the CTCG become much larger (O(C^2))
Very fast	The re-ordering procedure can be quite convoluted
	If the crunched demand asks for certain amount of **data** to be transmitted (instead of bandwidth), it gets even more convoluted

UCDAVIS

CTCG: Lowering the Minimum Wait Time

Different question: Can we (and how) make the minimum Wait Time (found before) even lower?

A possible approach is:

1. Find the minimum Wait Time;
2. Among the connections that are congesting that path, find the *N* biggest offenders (longest times)
   - ➢ With each of these *N* connections, check if we can make them finish faster, by:
     1. Provisioning an extra path for that connection;
     2. Throttling other flows to free up more bandwidth for that connection
3. With that, we might be able to make other connections finish faster and, thus, lower the minimum wait time for the current crunched demand

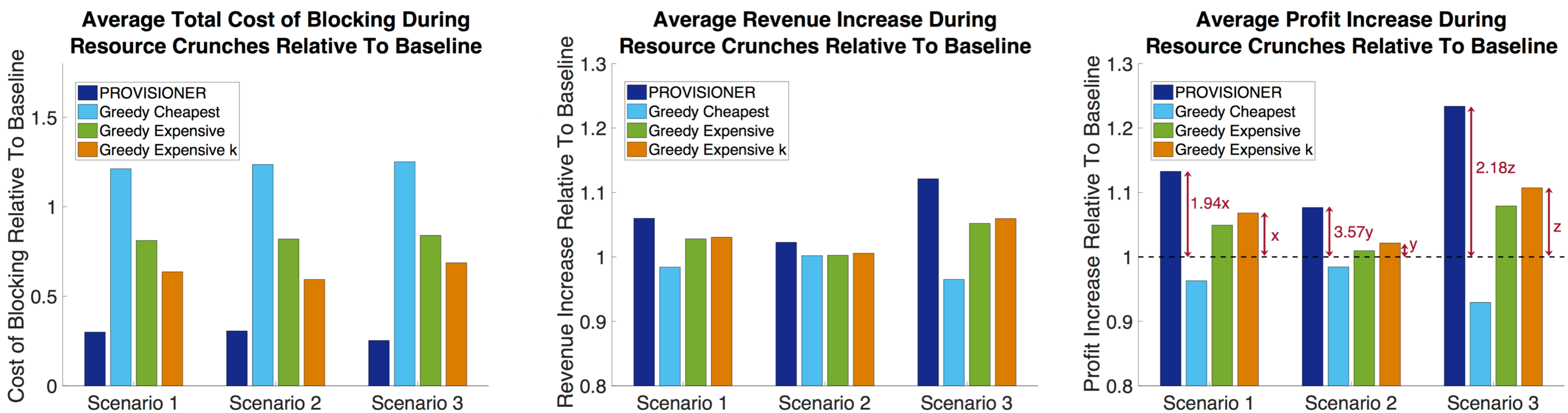
➢ Drawback: it might be better to lower some wait time other than the minimum

UCDAVIS

Next steps

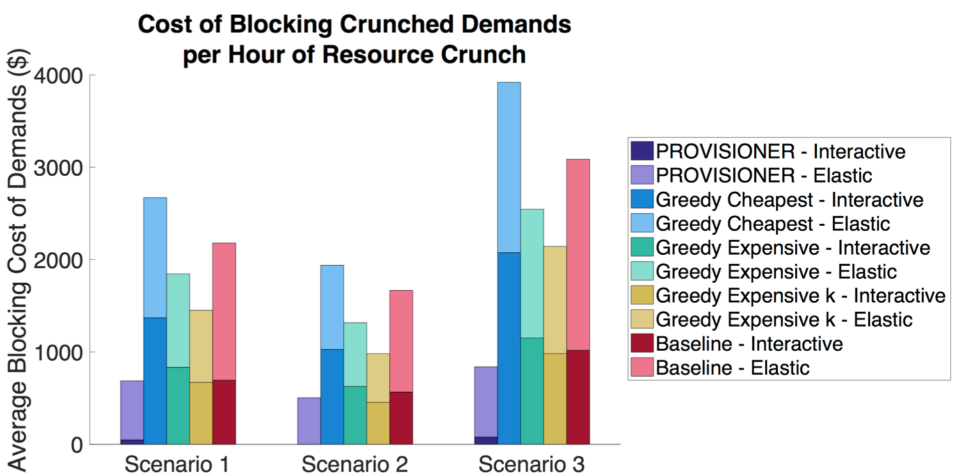
- Decide between: General network (WAN, possibly) Vs. Intra-DC OCS switched network
- General Scheduling Problem Vs. Scheduling under Resource Crunch
- Investigate Advance Reservation schemes
- Implementation and results

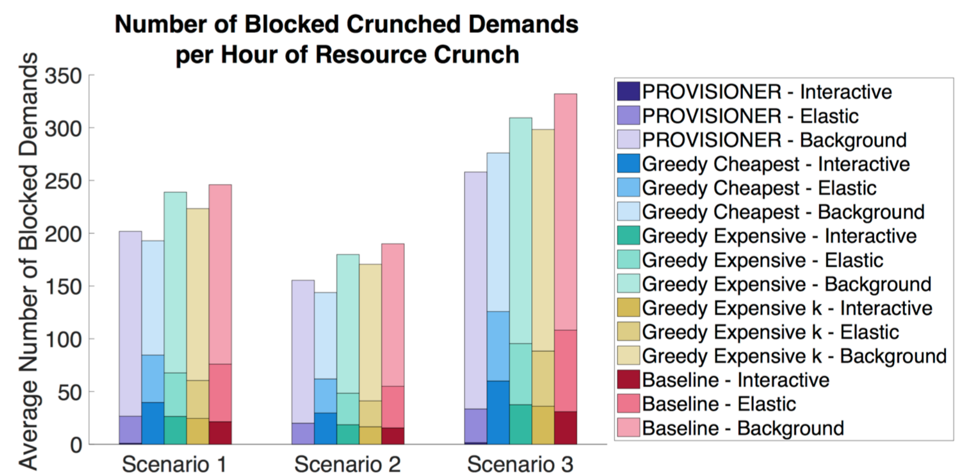
Blocking Costs, Revenues, Profits



UCDAVIS

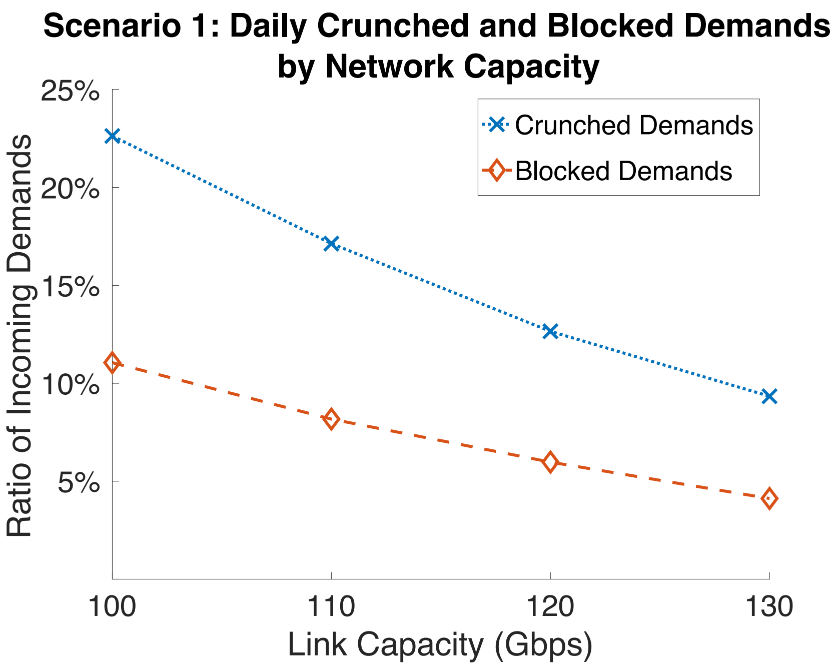
Crunched Demands not Served





UCDAVIS

Scenario 1: Crunching and Blocking Ratio



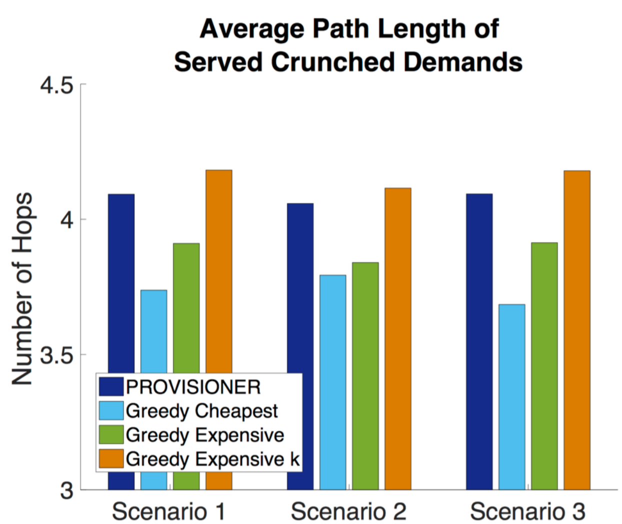
Vary link capacities from 100 to 130 Gbps:

- 100 Gbps: Resource Crunch ~ 8:00 hours/day
- 110 Gbps: Resource Crunch ~ 3:30 hours/day
- 120 Gbps: Resource Crunch ~ 1:30 hours/day
- 130 Gbps: Resource Crunch ~ 30 min/day

As Resource Crunch gets smaller, PROVISIONER is used less, leading to lower impact in the blocking ratio. For very small Resource Crunch, the orange curve converges to the regular blocking ratio of the network.

UCDAVIS

Average Path Lengths



PROVISIONER serves demands through longer paths than *Greedy Cheapest* and *Greedy Expensive*. This higher occupation of the network hinders the search of cheap degradations for future crunched demands. Thus, the shorter the Resource Crunch, better the PROVISIONER approach performs.

Note: shortest path routing is not necessarily more revenue-efficient

UCDAVIS

Future Work

- Data Evacuation through Aerial Platforms:
  - Investigate the use of flexible trajectory aerial platforms (e.g., drones) to assist in evacuating data and reestablishing continuous communications

- Running the Network Harder:
  - Investigate how traffic growth can be handled by networks being driven harder: when to execute network upgrades, where, and how
  - Investigate splittable connections, rerouting of allocated connections, deadline-driven demands
  - Investigate different CAG weighting schemes

UCDAVIS