Friedrich-Alexander-Universität
Technische Fakultät

Friedrich-Alexander-Universität, Erlangen-Nürnberg

Multiple DCLC Routing Algorithms for Ultra-Reliable and Time-Sensitive Applications

Piyush Navade, Lisa Maile, Reinhard German

Informatik 7 (Rechnernetze und Kommunikationssysteme)

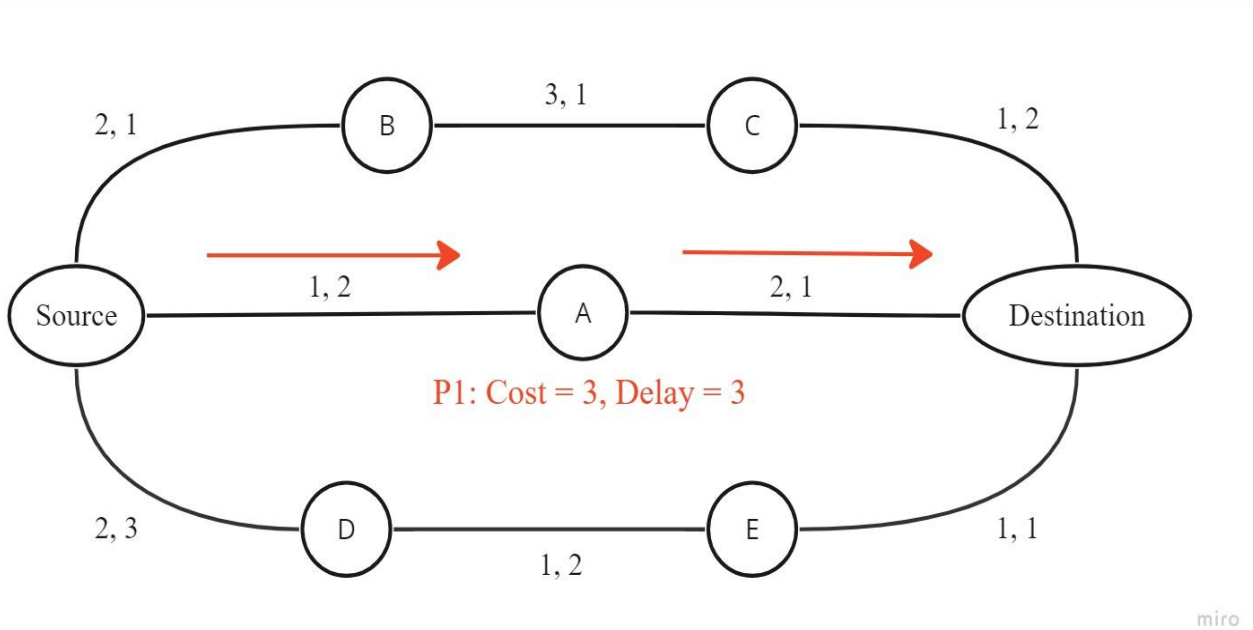
Outline

Problem Statement

- Time sensitive applications need highly **reliable** networks.
- In case of **faulty connection**, a new path needs to be configured to take over.
- All paths should have as **low cost** (weight) as possible.
- All paths should satisfy the **delay constraint**.
- **Objective**: Establish two or more disjoint shortest path from source to destination, considering both cost and delay requirements and simultaneously transmit data on all the paths to achieve ultra high reliability.

Example of Disjoint Paths

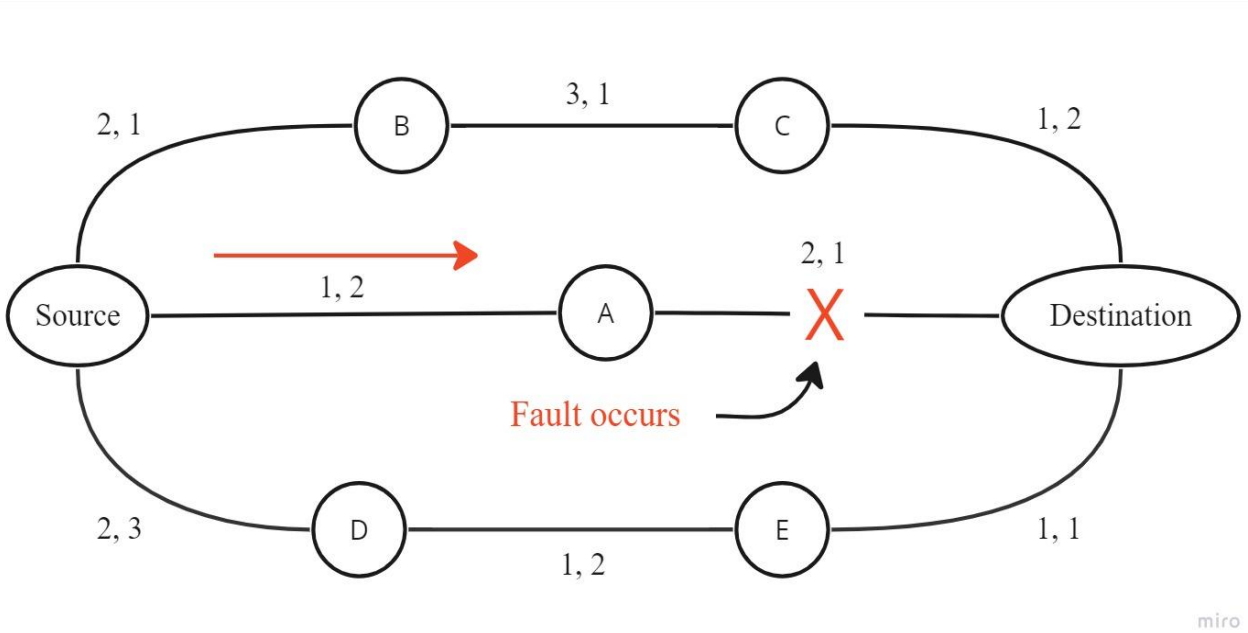
- Link attributes: **Delay and Cost (weight).**
- **Example:**
  - Generate two disjoint paths with least cost and max end-to-end delay of 5 units.



Path P1: Source - A - Destination (cost = 3, delay = 3 units).

Example of Disjoint Paths

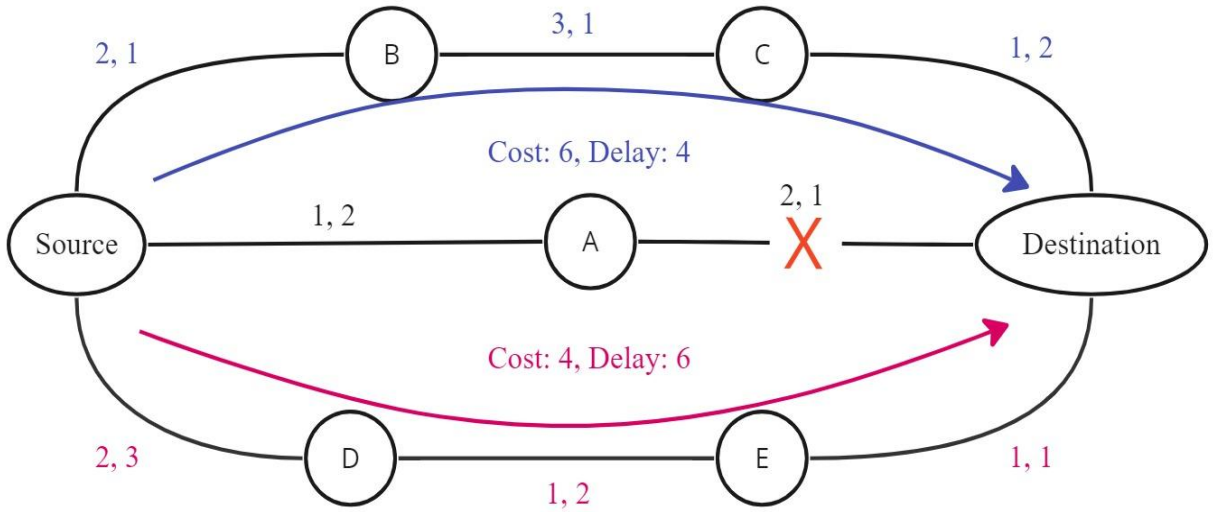
- Link attributes: **Delay and Cost (weight).**
- **Example:**
  - Generate two disjoint paths with least cost and max end-to-end delay of 5 units.



Path P1: Source - A - Destination (cost = 3, delay = 3 units).

Example of Disjoint Paths

- Link attributes: **Delay and Cost (weight).**
- **Example:**
  - Generate two disjoint paths with least cost and max end-to-end delay of 5 units.

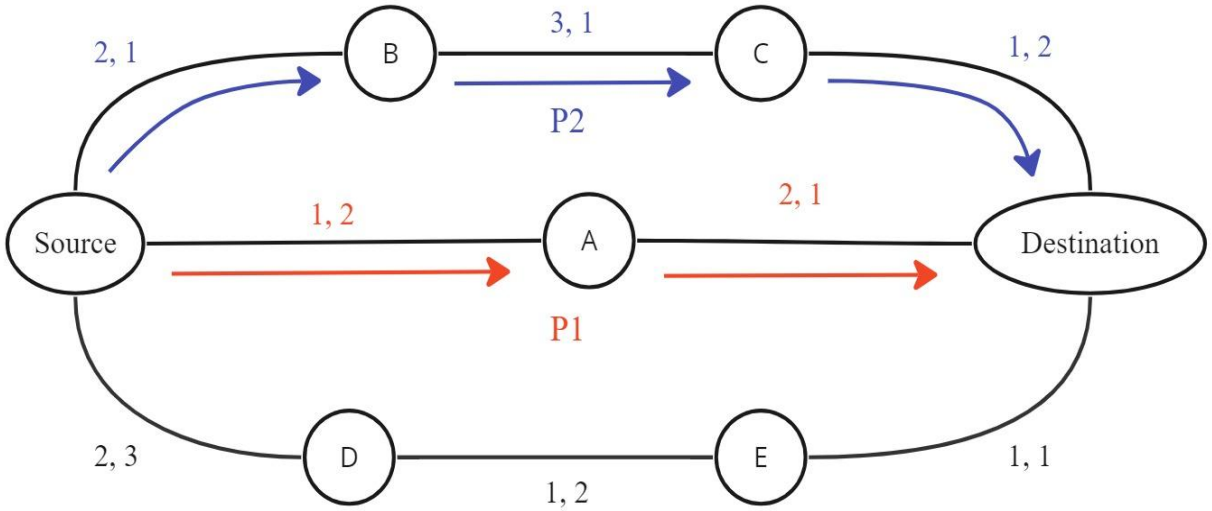


Path P2: Source - B - C - Destination (cost = 6, delay = 4 units).

Path P3: Source - D - E - Destination (cost = 5, delay = 6 units).

Example of Disjoint Paths

- Link attributes: **Delay and Cost (weight).**
- **Example:**
  - Generate two disjoint paths with least cost and max end-to-end delay of 5 units.



Path P1: Source - A - Destination (cost = 3, delay = 3 units).

Path P2: Source - B - C - Destination (cost = 6, delay = 4 units).

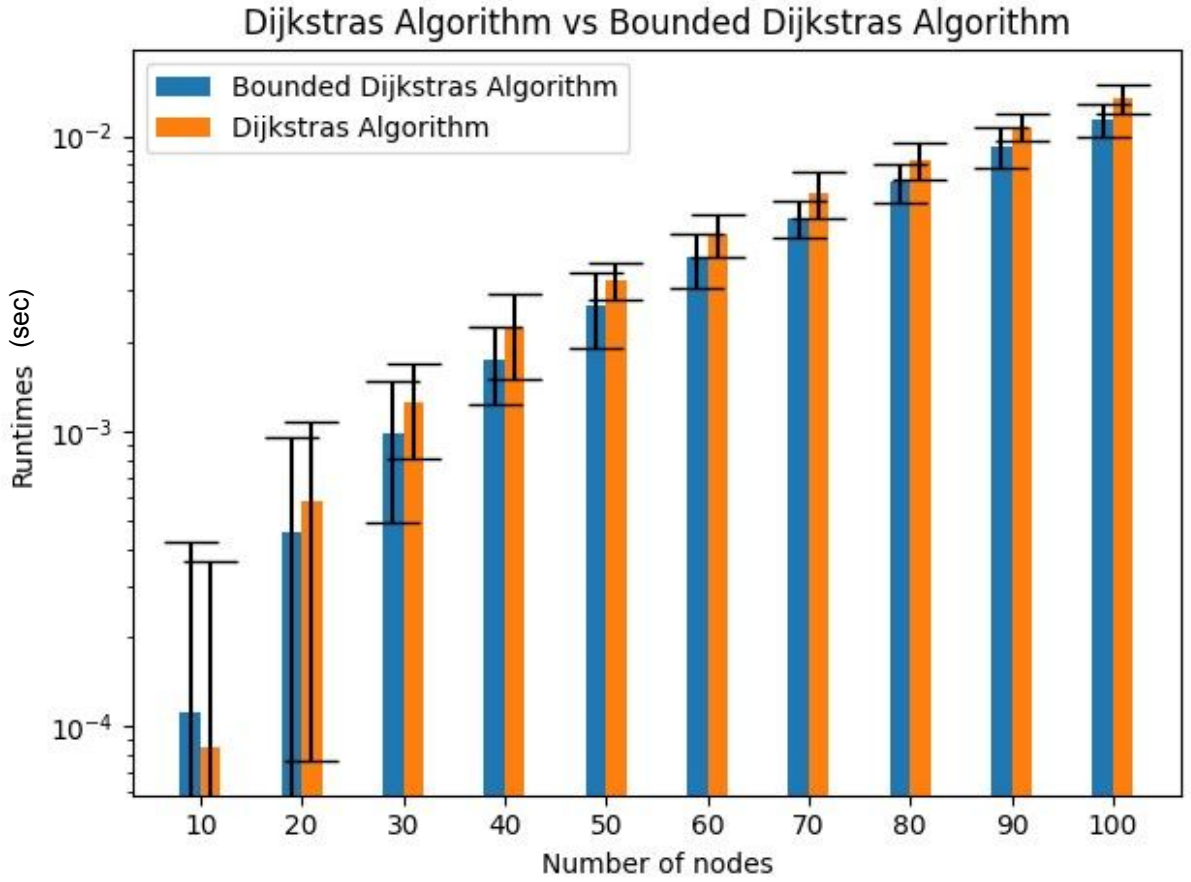
~~**Path P3: Source - D - E - Destination (cost = 5, delay = 6 units).**~~

Bounded Dijkstra's Algorithm[1]

1. Bemten, A.V., Guck, J.W., Machuca, C.M., and Kellerer, W. (2019). Bounded Dijkstra (BD): Search Space Reduction for Expediting Shortest Path Subroutines. ArXiv, abs/1903.00436.

Bounded Dijkstra's Algorithm

- Dijkstra's algorithm discovers paths in increasing order of cost.
- Delay values from source to other nodes are tracked in the same way as the weight is calculated in simple Dijkstra's.
- Additionally, a **delay threshold** is introduced.
- If this threshold exceeds, the corresponding cost is **not** updated.
- Thereby **reducing the search space and runtime**.



Bounded Dijkstra’s Algorithm

Drawbacks of Bounded Dijkstra’s:

- Running Dijkstra's algorithm multiple times makes them **computationally expensive**.
- Possibility of **not finding enough disjoint paths**.
- **Runtime increases** when more than two shortest paths are required.

Multiple Disjoint Path Algorithm (MDPAlg)[1]

1. D. Lopez-Pajares, E. Rojas, J. A. Carral, I. Martinez-Yelmo and J. Alvarez-Horcajo, "The Disjoint Multipath Challenge: Multiple Disjoint Paths Guaranteeing Scalability," in IEEE Access, vol. 9, pp. 74422-74436, 2021.

Multiple Disjoint Path Algorithm (MDPAlg)

Two step algorithm:

- Construction of a **cost matrix**:
  - Requires a single full graph search.
  - Stores **‘extra’** information compared to Dijkstra’s.

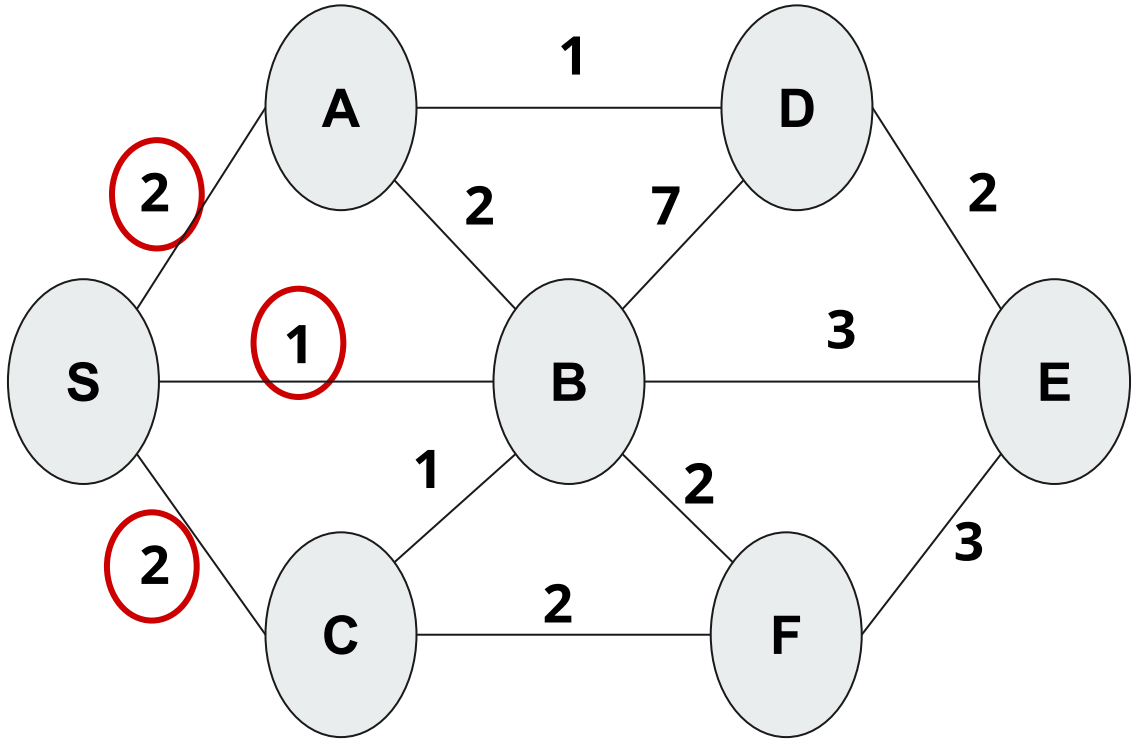
- Construction of disjoint paths:

Improvements, over Bounded Dijkstra’s, with MDPAlg:

- Does not make use of Dijkstra’s algorithm.

- Guaranteed to extract **maximum** number of disjoint paths possible.

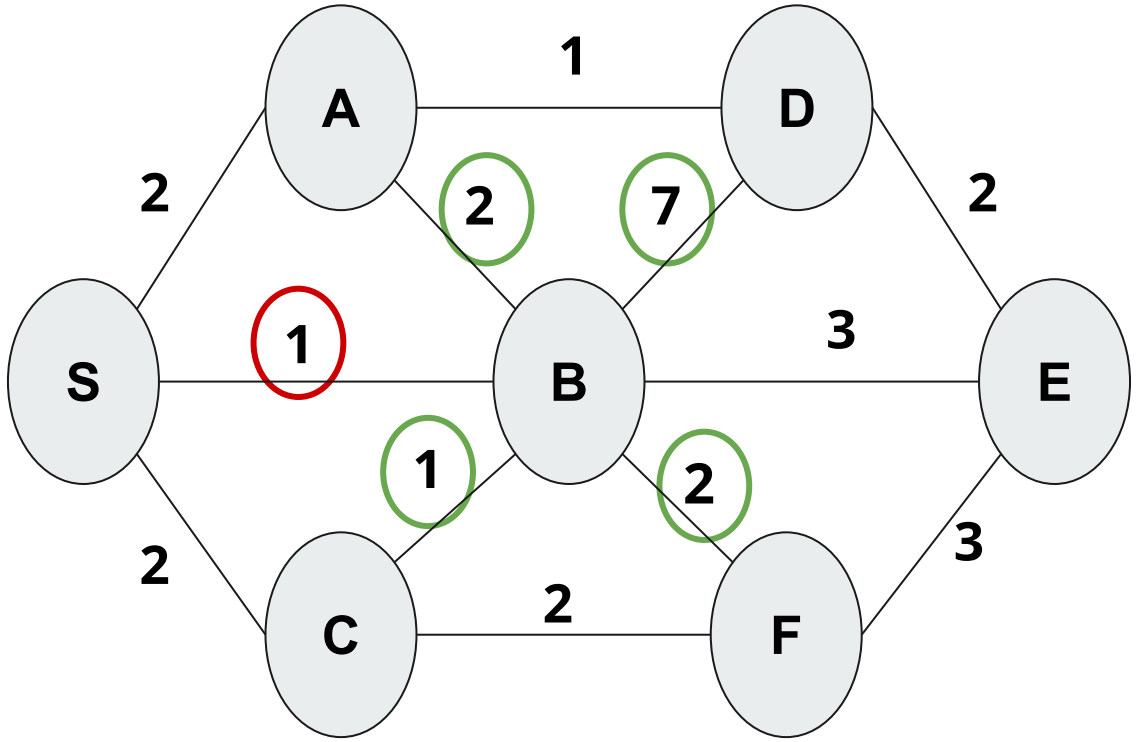
Step 1: Construction of Cost Matrix



	S	A	B	C	D	E	F
S		2	1	2			
A							
B							
C							
D							
E							
F							

COST MATRIX

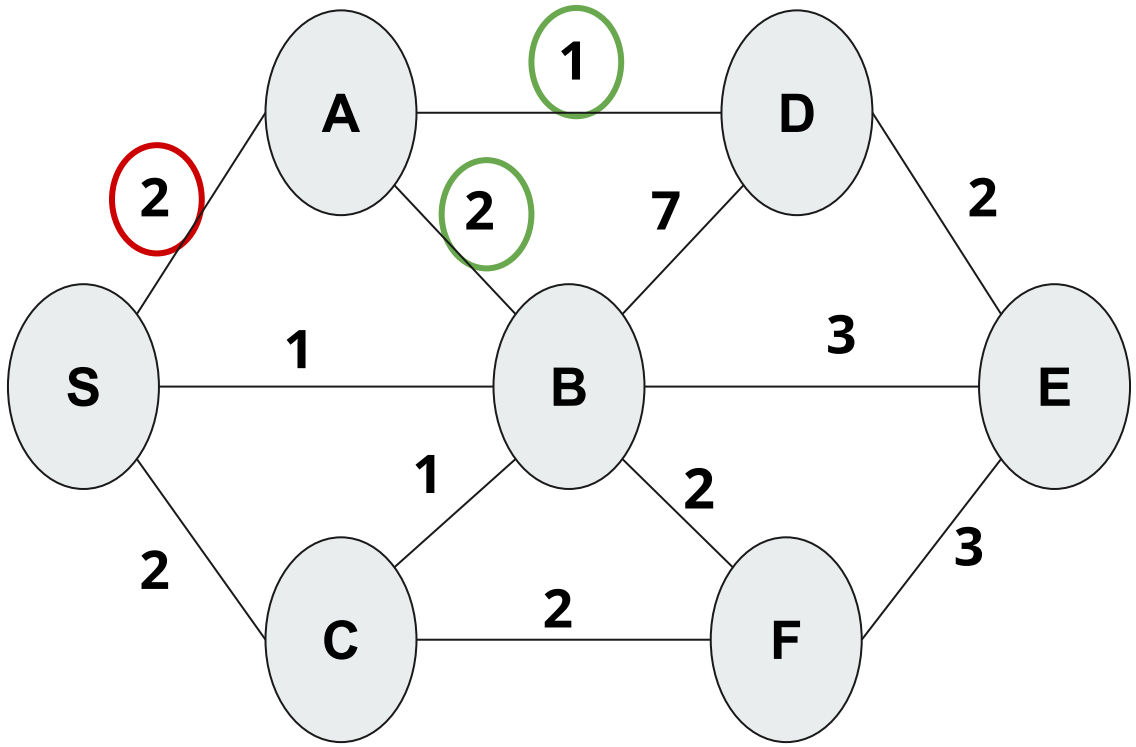
Step 1: Construction of Cost Matrix



	S	A	B	C	D	E	F
S		2	1	2			
A							
B		3		2	8	4	3
C							
D							
E							
F							

COST MATRIX

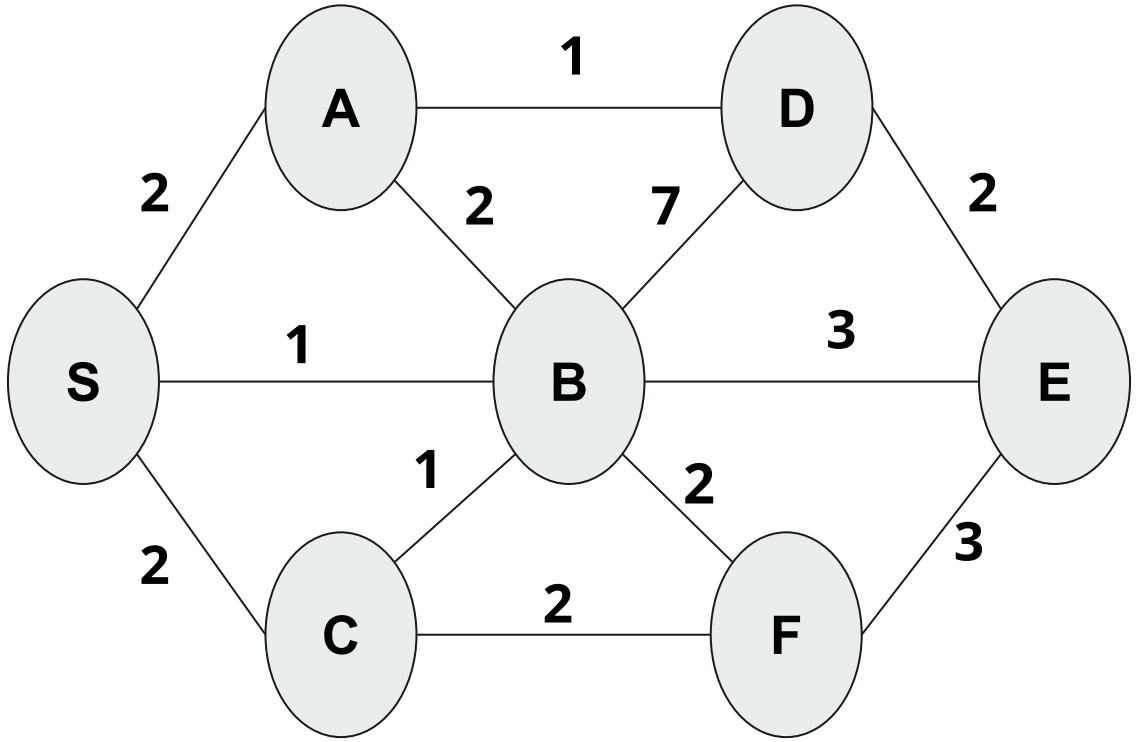
Step 1: Construction of Cost Matrix



	S	A	B	C	D	E	F
S		2	1	2			
A			4		3		
B		3		2	8	4	3
C							
D							
E							
F							

COST MATRIX

Step 1: Construction of Cost Matrix



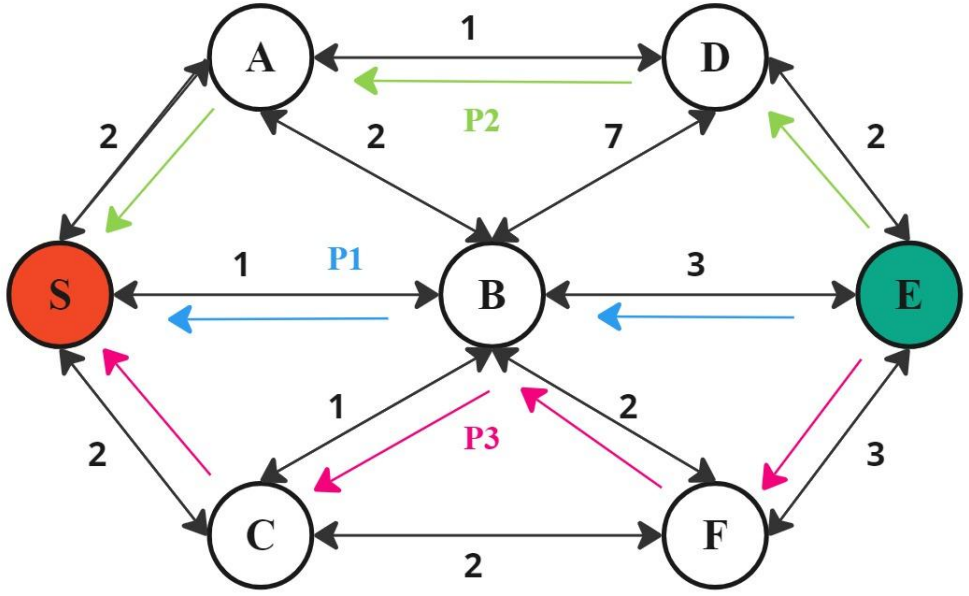
	S	A	B	C	D	E	F
S		2	1	2			
A			4		3		
B		3		2	8	4	3
C			3				4
D		4	10			5	
E			7		6		7
F			5	5		6	

COST MATRIX

Step 2: Construction of Disjoint Paths

From source: S ⟹ To destination: E

E ⇨ B ⇨ S

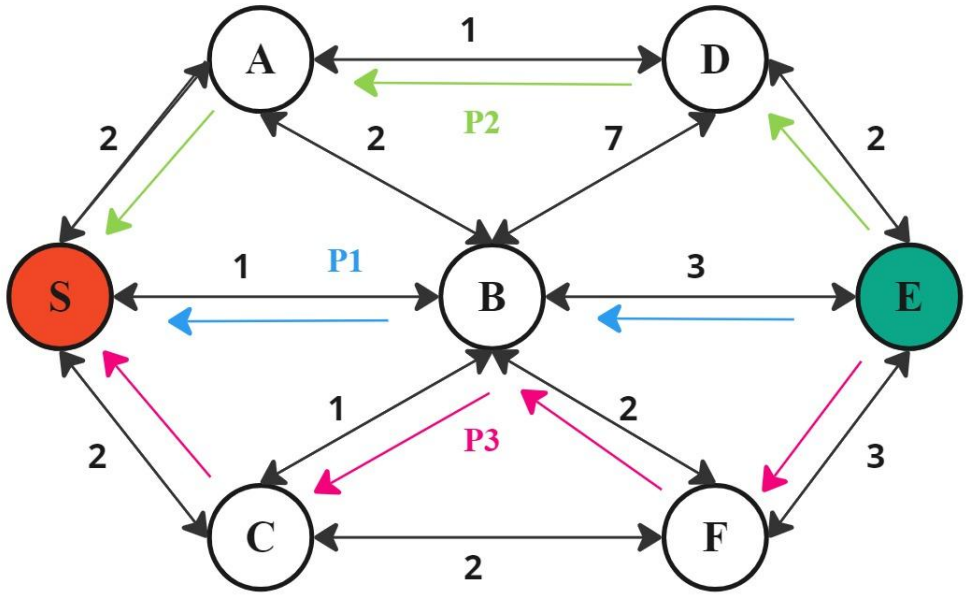


	S	A	B	C	D	E	F
S		2	1	2			
A			4		3		
B		3		2	8	inf	3
C			3				4
D		4	10			5	
E			inf		6		7
F			5	5		6	

Step 2: Construction of Disjoint Paths

From source: S ⟹ To destination: E

E ⟹ D ⟹ A ⟹ S



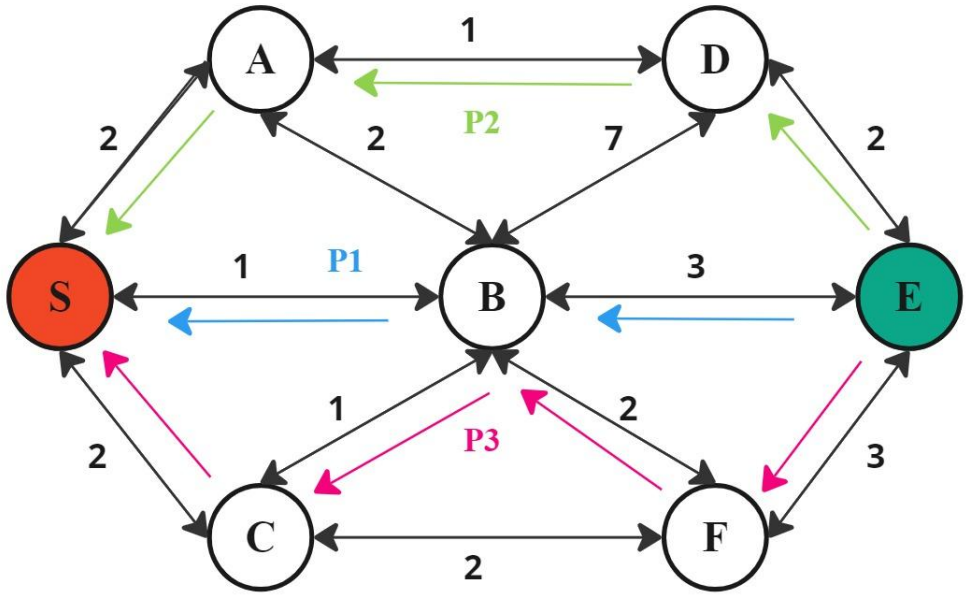
	S	A	B	C	D	E	F
S		2	inf	2			
A			4		inf		
B		3		2	8	inf	3
C			3				4
D		inf	10			inf	
E			inf		6		7
F			5	5		6	

miro

Step 2: Construction of Disjoint Paths

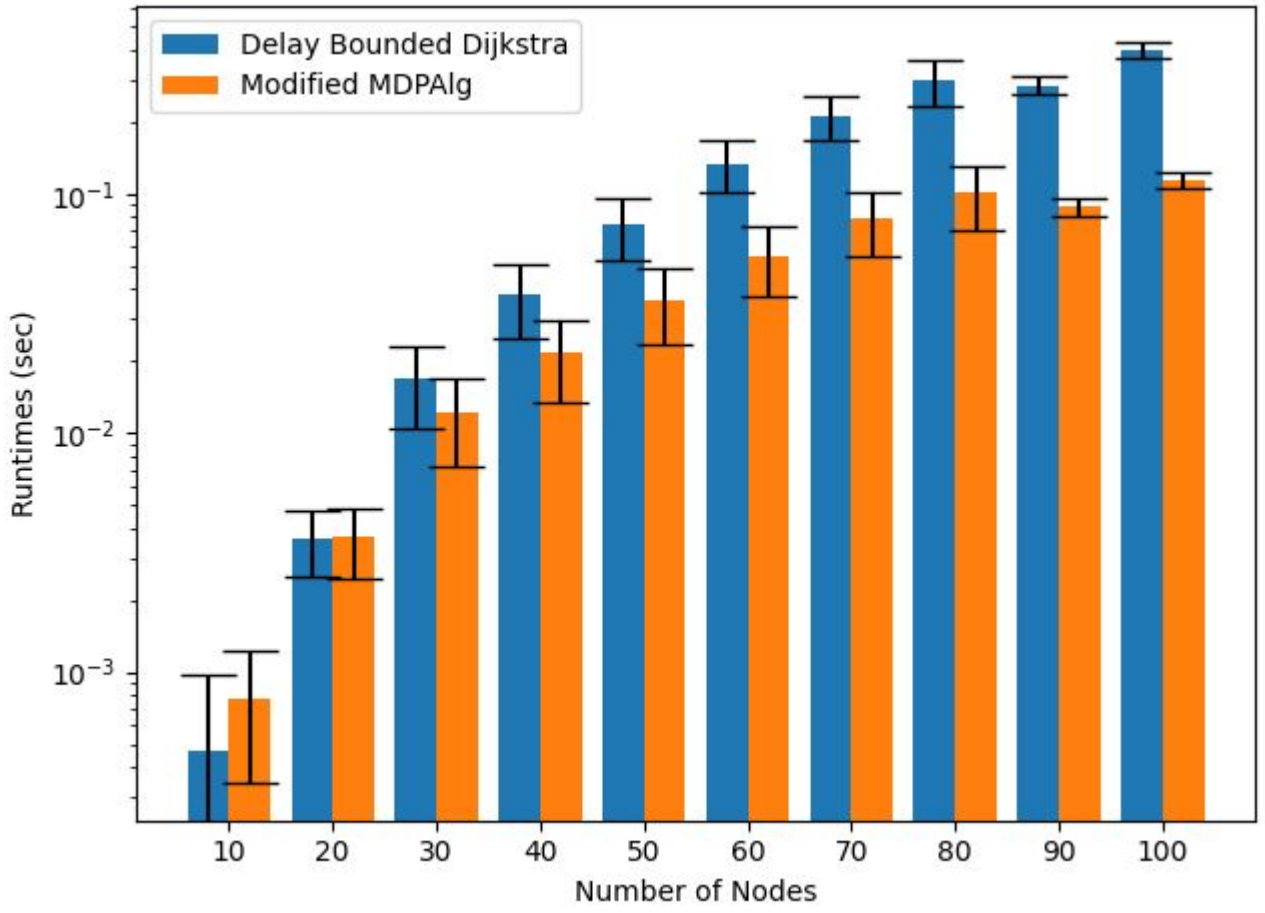
From source: S ⟹ To destination: E

E ⟹ F ⟹ B ⟹ C ⟹ S



	S	A	B	C	D	E	F
S		inf	inf	2			
A			4		inf		
B		3		2	8	inf	3
C			3				4
D		inf	10			inf	
E			inf		6		7
F			5	5		6	

Results



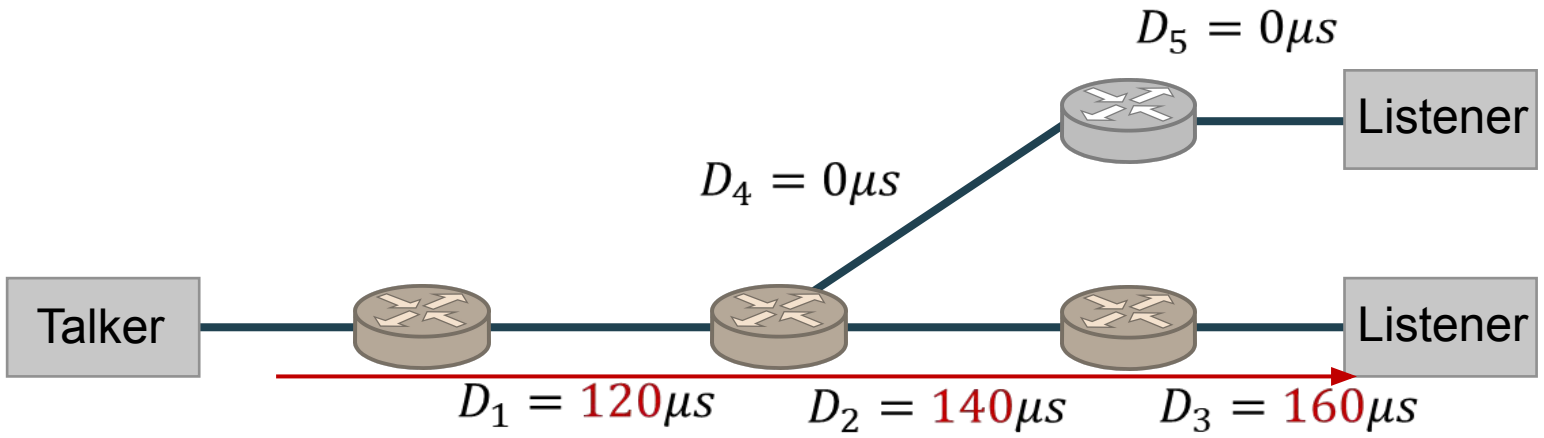
Test Case **Parameters**:

- Randomly generated 1000 Fully-connected graphs with 10, 20, 30, ….., 100 Nodes.
- Randomly generated link attributes (cost and delay).
- Randomly generated source-destination pair.

Delay Constraints for Real-Time Communications

Problem:

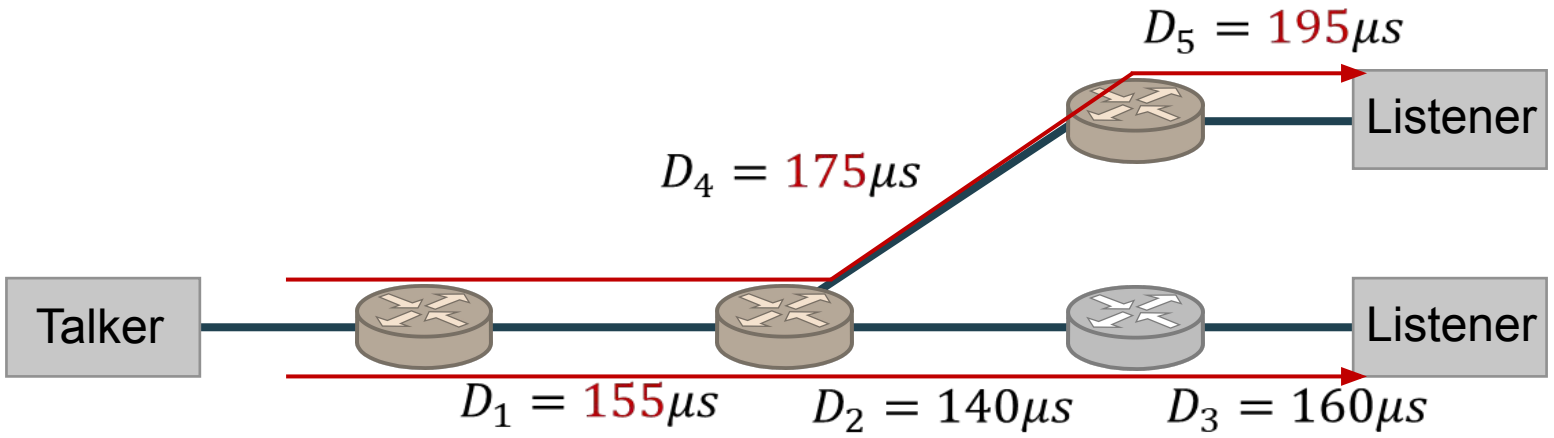
- Proposed routing algorithms run iteratively for each new flow in the network



Delay Constraints for Real-Time Communications

Problem:

- Proposed routing algorithms run iteratively for each new flow in the network
- Delays at each hop depend on the flows in the network

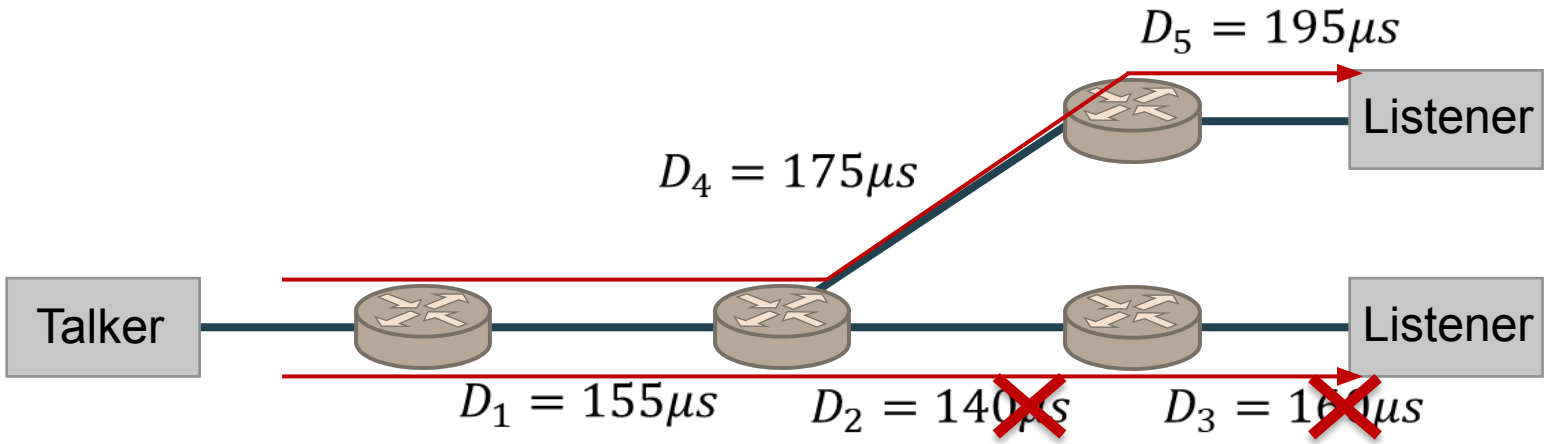


Delay Constraints for Real-Time Communications

Problem:

- Proposed routing algorithms run iteratively for each new flow in the network
- Delays at each hop depend on the flows in the network
- Delays for already reserved flows might no longer be valid after a new reservation

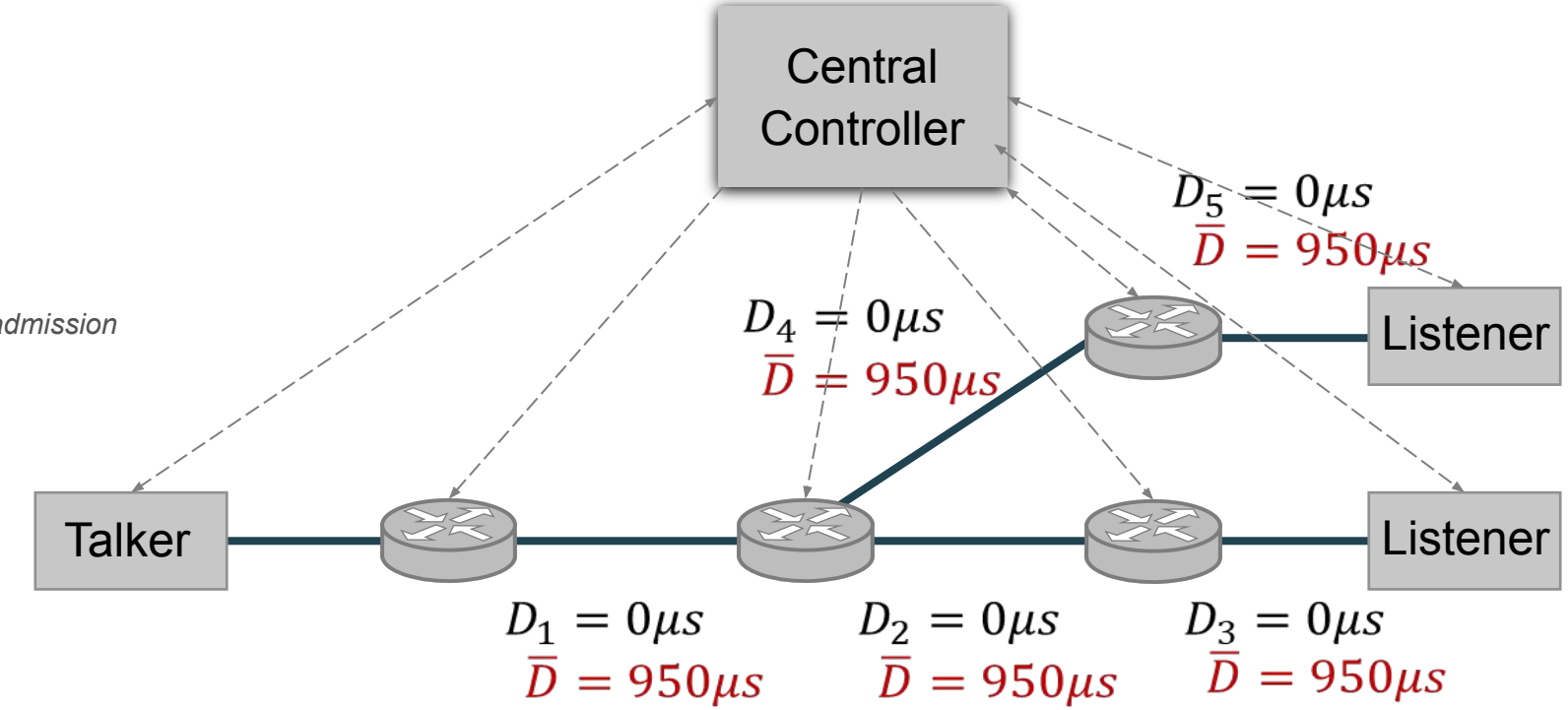
→ **re-run routing algorithm for of all flows?**



Delay Constraints for Real-Time Communications

Solution:

- Central network controller to determine reservation-independent delay bounds $\overline{D}$ [1]
- Used by the routing algorithms

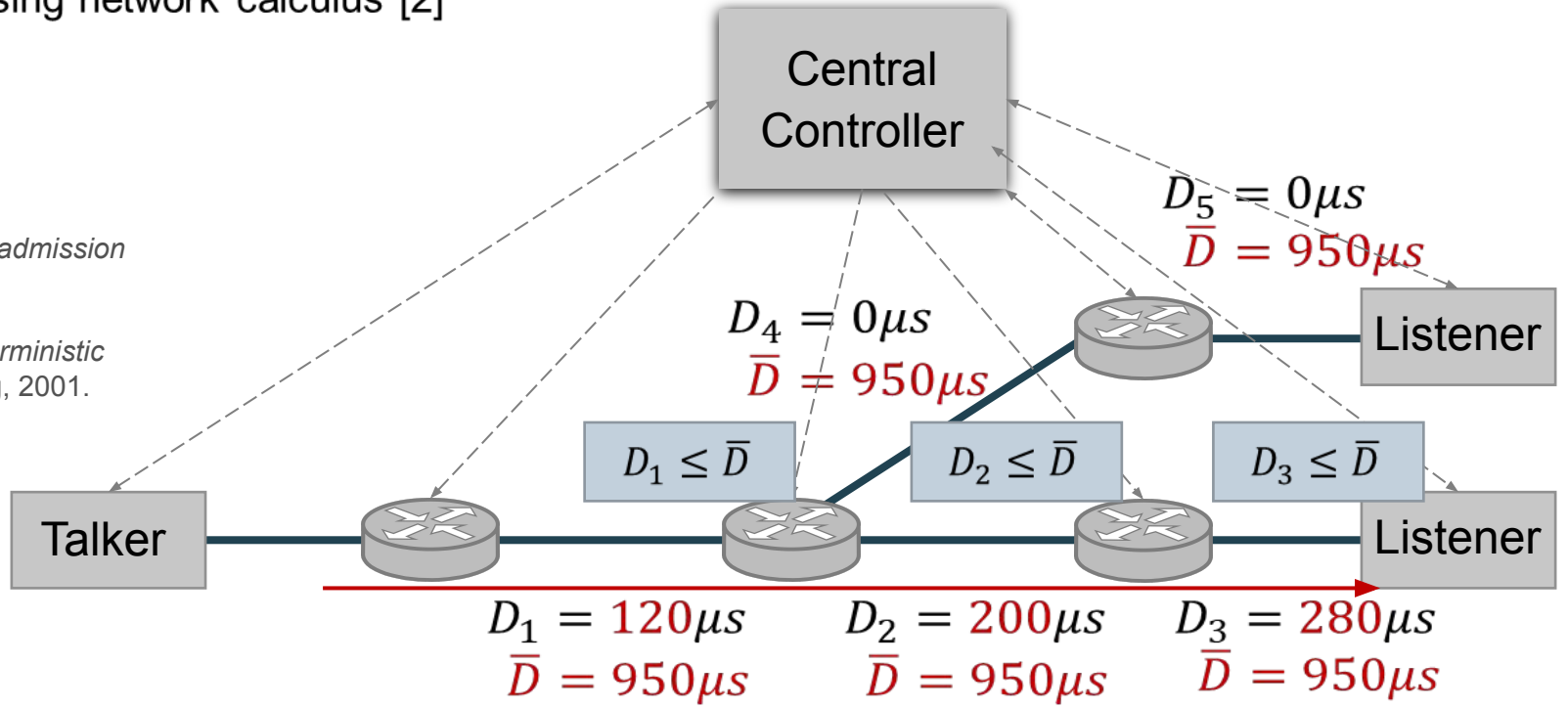


[1] L. Maile, K.-S. J. Hielscher, and R. German, "*Delay-guaranteeing admission control for time-sensitive networking using the credit-based shaper*," IEEE Open Journal of the Communications Society, vol. 3, 2022.

Delay Constraints for Real-Time Communications

Solution:

- Central network controller to determine reservation-independent delay bounds $\overline{D}$ [1]
- Used by the routing algorithms
- Validated before each new reservation using network calculus [2]



[1] L. Maile, K.-S. J. Hielscher, and R. German, "*Delay-guaranteeing admission control for time-sensitive networking using the credit-based shaper*," IEEE Open Journal of the Communications Society, vol. 3, 2022.
[2] J.-Y. Le Boudec and P. Thiran, "*Network calculus: A theory of deterministic queuing systems for the Internet*". Berlin, Heidelberg: Springer-Verlag, 2001.

Conclusion

- Most of the disjoint routing algorithms are constrained to finding **only one pair** of shortest paths.
- **Computational complexity** of Dijkstra's Algorithm explodes in huge networks. Therefore not suitable for disjoint routing problems.
- Multiple Disjoint Path Algorithm proposes an innovative way of finding disjoint paths in a network through **single full graph search**.
- MDPAlg can be easily modified to make it useful for multicast transmission.

Thank You

Contact:
piyush.navade@fau.de
lisa.maile@fau.de