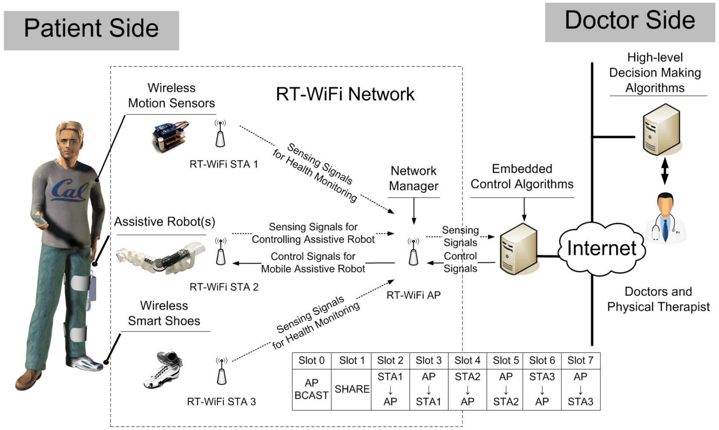
FD-PaS: A Fully Distributed Packet Scheduling Framework for Handling Disturbances in Real-Time Wireless Networks

Tianyu Zhang[1,3], Tao Gong[2], Zelin Yun[2], Song Han[2],
Qingxu Deng[1], X. Sharon Hu[3]

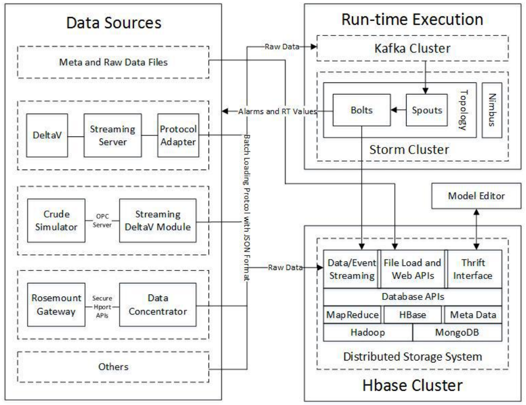
[1]Northeastern University, China
[2]University of Connecticut, USA
[3]University of Notre Dame, USA

Real-Time Wireless Networks (RTWNs)

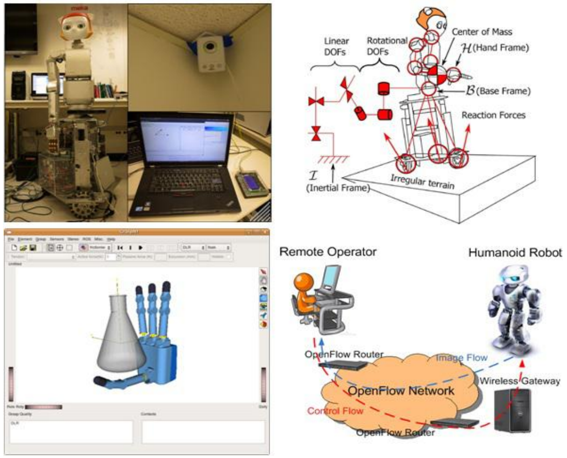


Slot 0	Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7
AP BCAST	SHARE	STA1 ↓ AP	AP ↓ STA1	STA2 ↓ AP	AP ↓ STA2	STA3 ↓ AP	AP ↓ STA3

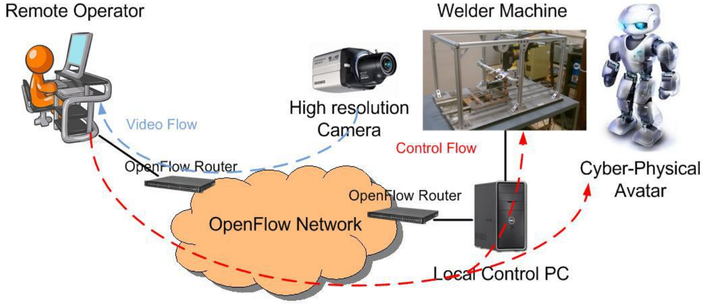
Network-based Rehabilitation System



Real-Time Analytics Platform for Process Control

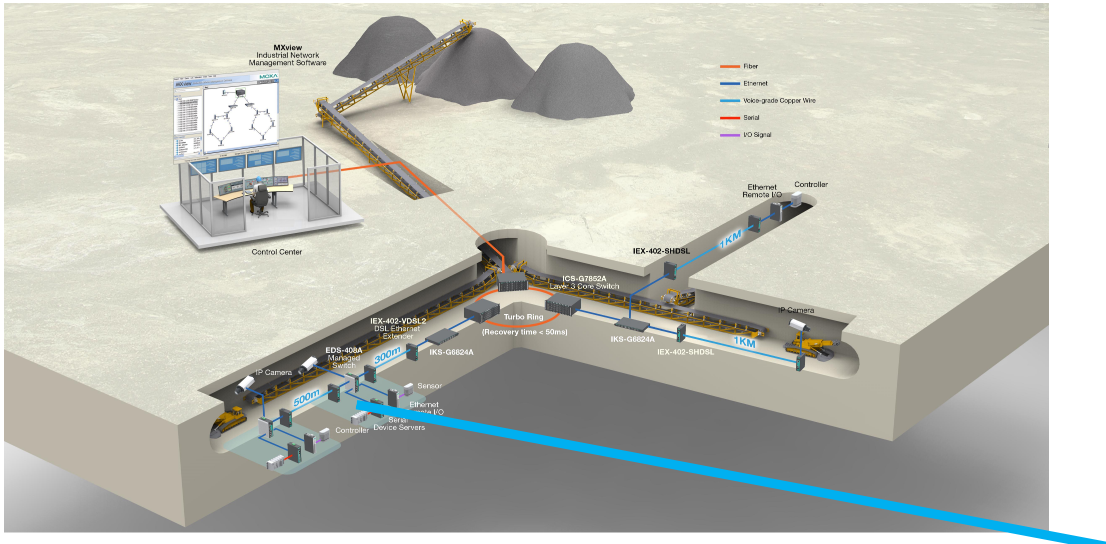


Cyber-physical Avatar



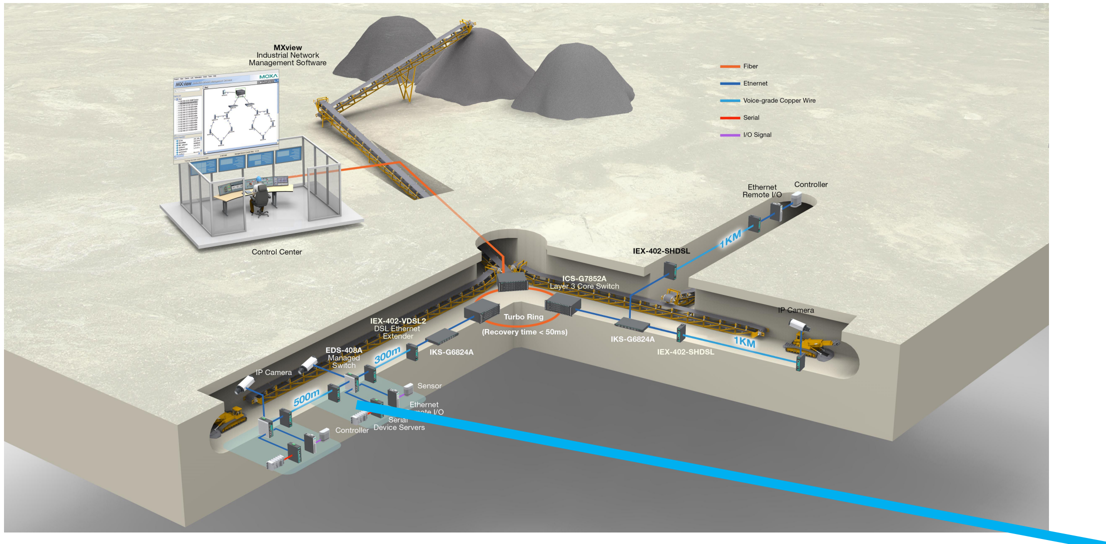
Remote and Real-time Welding System

An Example: Mining Monitoring System



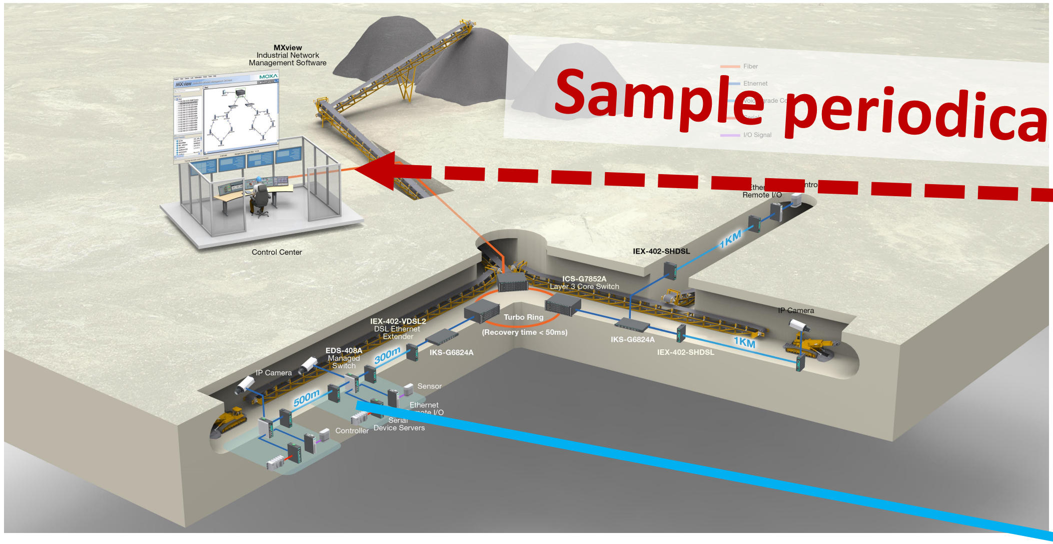
- Sensors monitor environment in real time
- Tasks are usually sampled and processed periodically with **fixed periods**

An Example: Mining Monitoring System



- Sensors monitor environment in real time
- Tasks are usually sampled and processed periodically with **fixed periods**

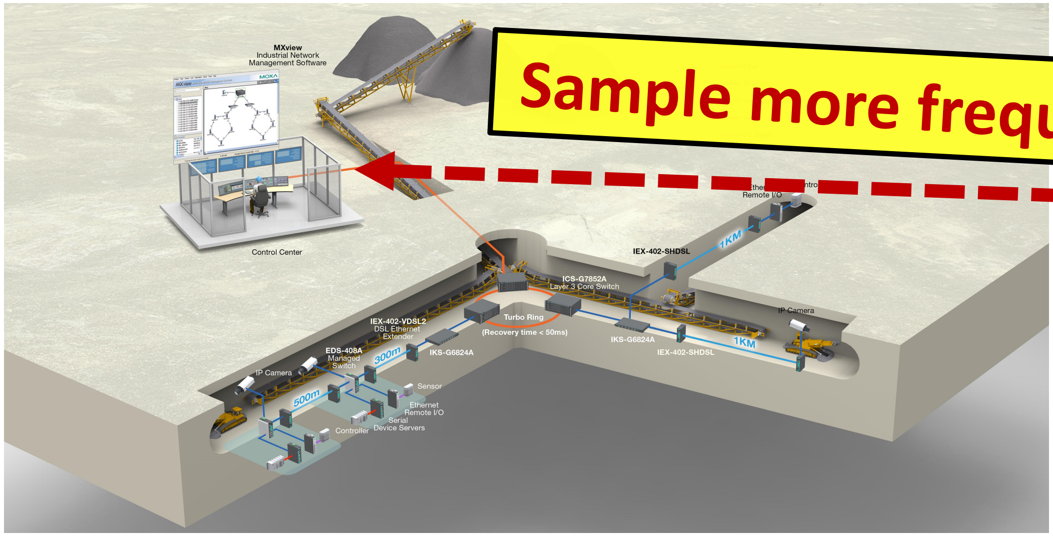
An Example: Mining Monitoring System



Sample periodically

- Sensors monitor environment in real time
- Tasks are usually sampled and processed periodically with **fixed periods**

An Example: Mining Monitoring System



Sample more frequently

- **External disturbance**: unexpected changes in temperature/pressure, etc.
- Require more frequent monitoring/response

What We Want to Achieve?

Requirements of a RTWN

- Data must be collected timely

What We Want to Achieve?

Requirements of a RTWN

- Data must be collected timely
- Deployed over large area

What We Want to Achieve?

Requirements of a RTWN

- Data must be collected timely
- Deployed over large area
- Fast response to disturbance

What We Want to Achieve?

Requirements of a RTWN

- Data must be collected timely
- Deployed over large area
- Fast response to disturbance
- High QoS (how well it satisfies real-time deadlines)

What We Want to Achieve?

Requirements of a RTWN

- Data must be collected timely
- Deployed over large area
- Fast response to disturbance
- High QoS (how well it satisfies real-time deadlines)

Our Design

- ✓ On-line scheduling
- ✓ Fully distributed
- ✓ Guaranteed fast response
- ✓ Fewest dropped packets

Outline

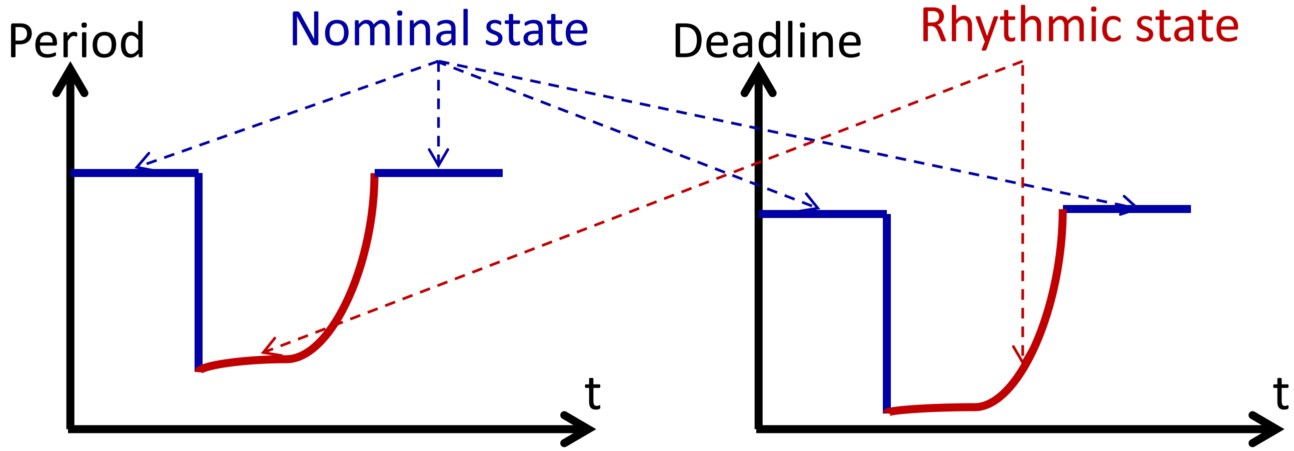
- **System model & related work**
- Fully distrubuted packet scheduling framework (FD-PaS)
- Experimental evaluation

Model Disturbance

- When nothing happens
  - ❑ All tasks follow regular periods
- When disturbance occurs
  - ❑ The corresponding task follows a specific release pattern

Rhythmic Model

- When nothing happens
  - All tasks follow regular periods
- When disturbance occurs
  - The corresponding task follows a specific release pattern

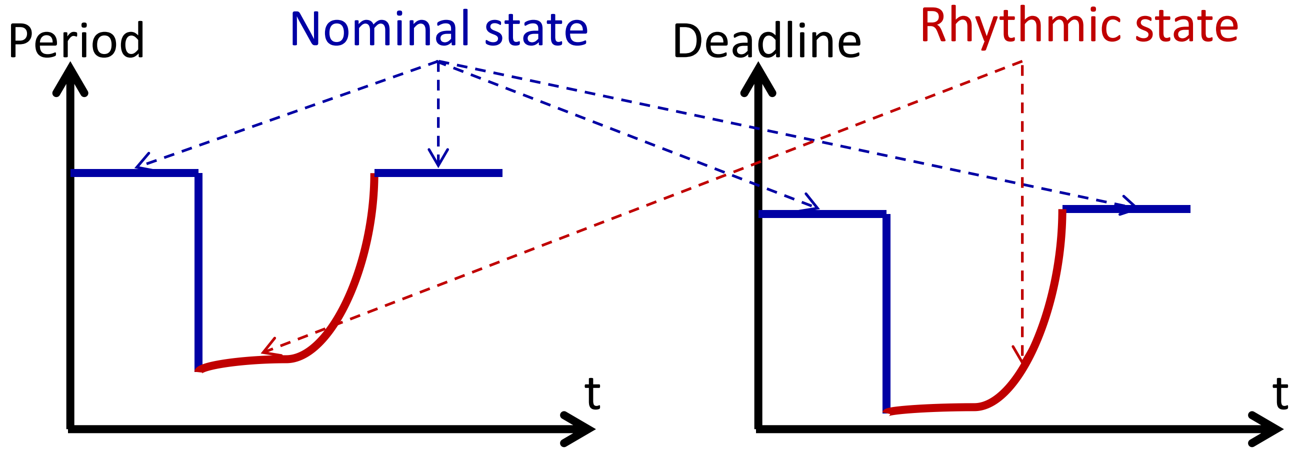


J. Kim, K. Lakshmanan and R. Rajkumar, *ICCPS*, 2012

Rhythmic Model

- When nothing happens
  - ❑ All tasks follow regular periods
- When disturbance occurs
  - ❑ The corresponding task follows a specific release pattern

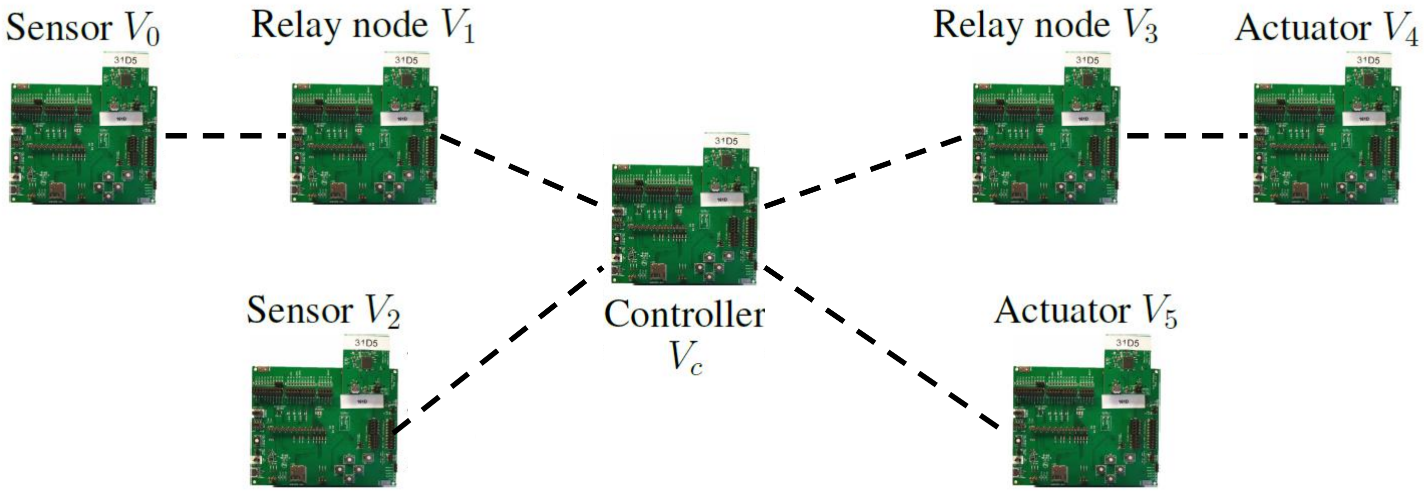
Works for other models



J. Kim, K. Lakshmanan and R. Rajkumar, *ICCPS*, 2012

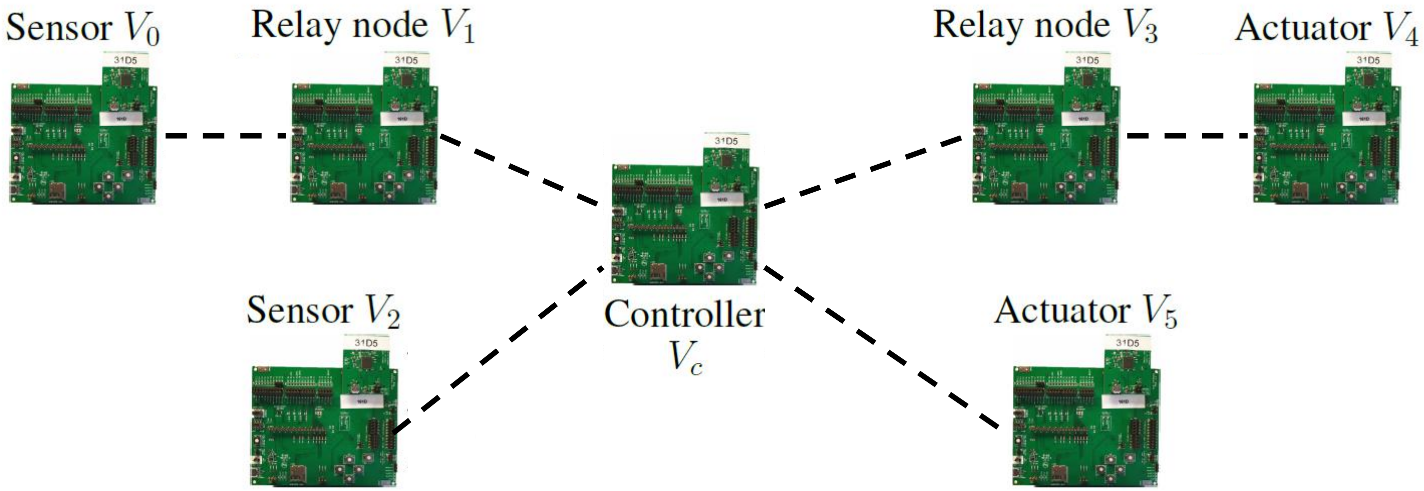
System Model

- RTWN infrastructure
  - A controller, sensors, relay nodes and actuators sharing a channel
  - Nodes have computing capability



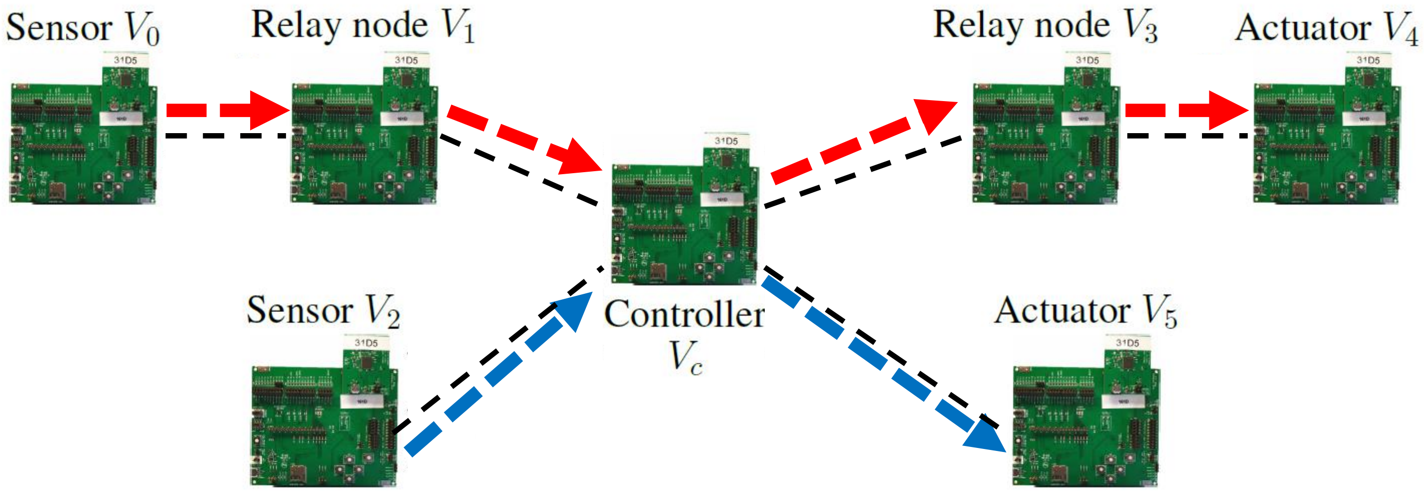
System Model

- RTWN infrastructure
  - A controller, sensors, relay nodes and actuators sharing a channel
  - Nodes have computing capability
- Task model
  - Unicast tasks (periodic and rhythmic) release infinite packets
  - One disturbance in the system at a given time

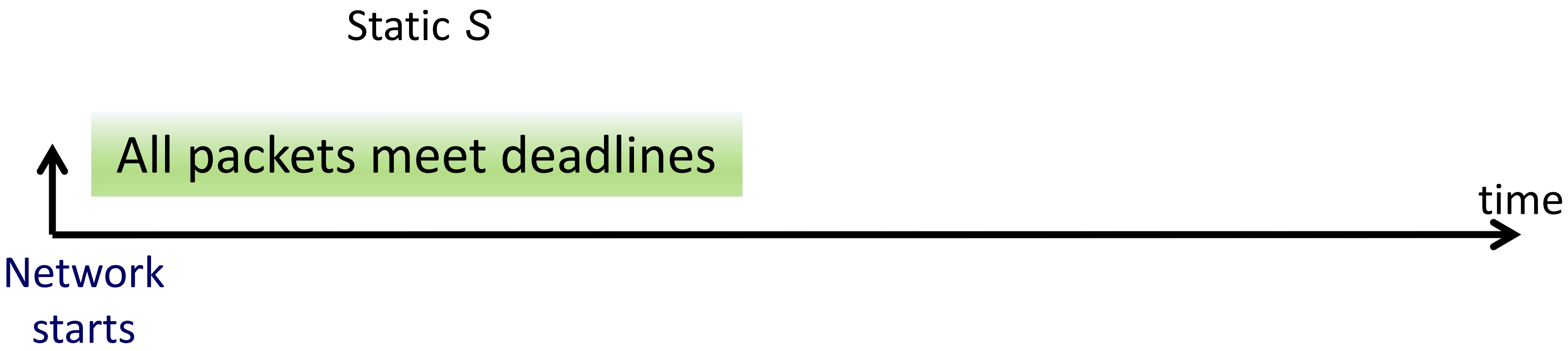


System Model

- RTWN infrastructure
  - ❑ A controller, sensors, relay nodes and actuators sharing a channel
  - ❑ Nodes have computing capability
- Task model
  - ❑ Unicast tasks (periodic and rhythmic) release infinite packets
  - ❑ One disturbance in the system at a given time
  - ❑ Routing path: every task passes through the controller

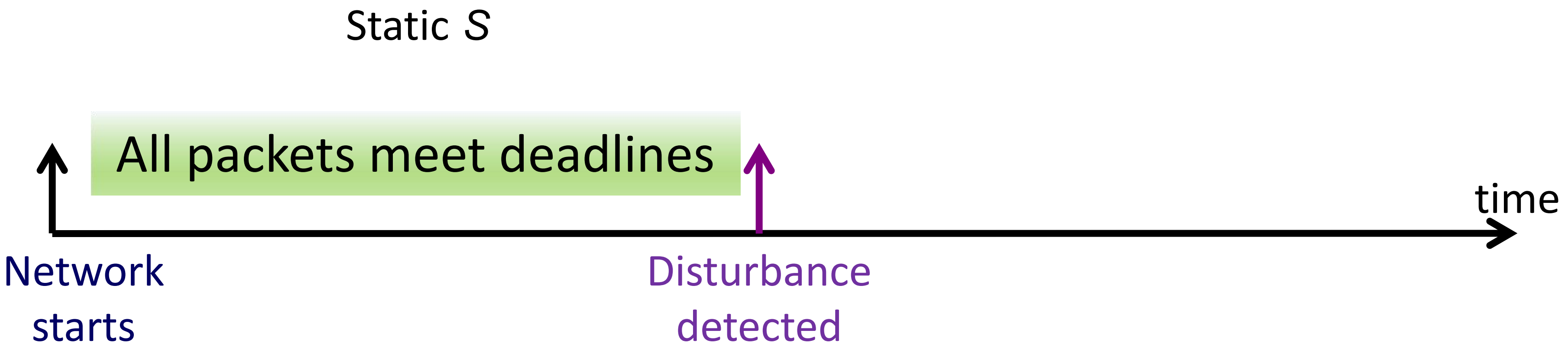


Problem Overview



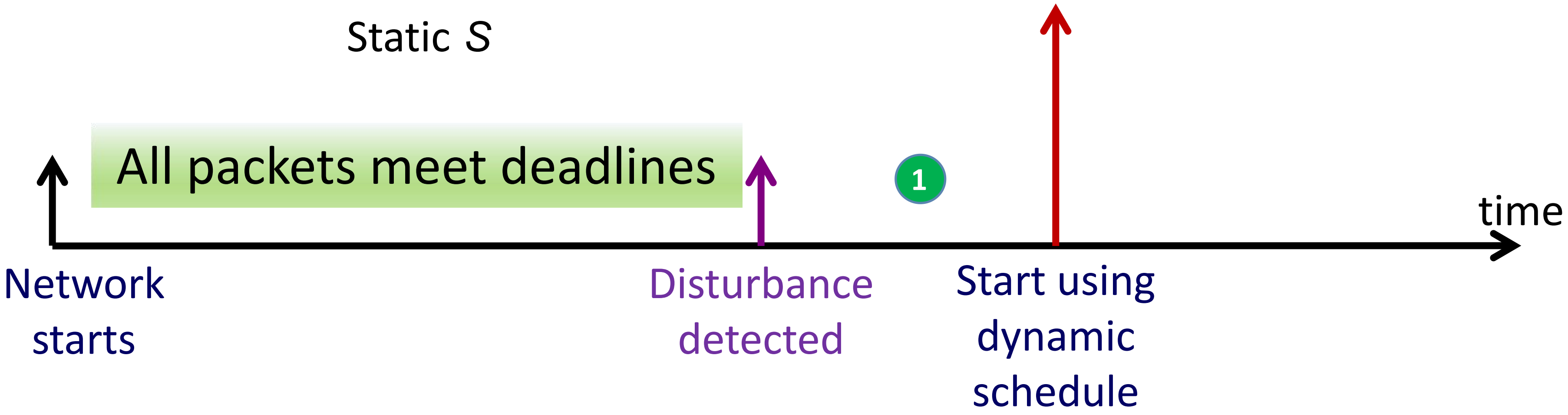
- No disturbance
  - ❑ Use a feasible static schedule

Problem Overview



- No disturbance
  - Use a feasible static schedule
- Upon detection of a disturbance, determine a dynamic schedule

Problem Overview



Static *S*

All packets meet deadlines

1

time

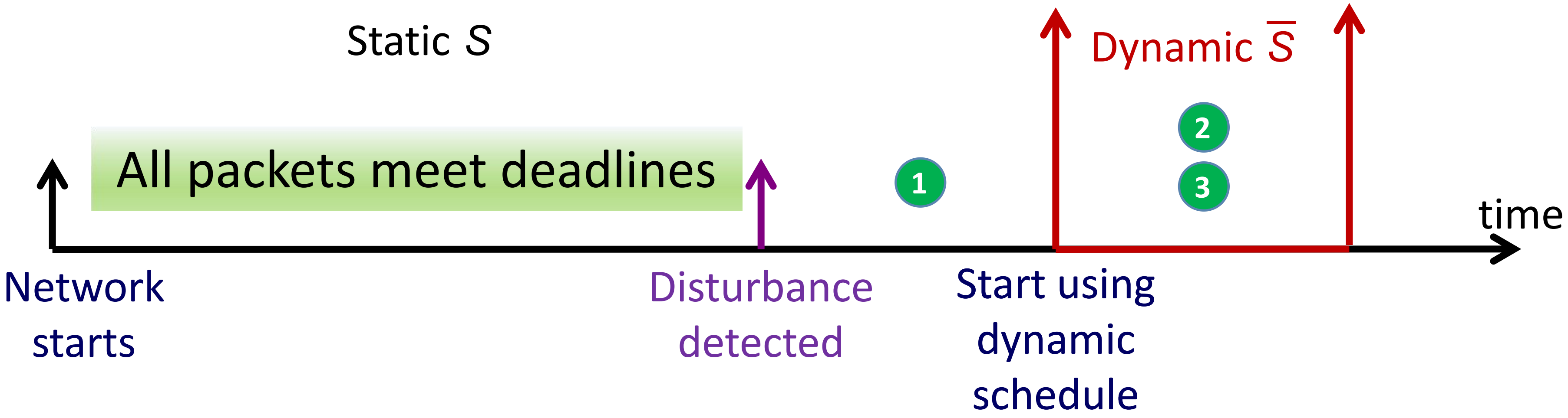
Network starts

Disturbance detected

Start using dynamic schedule

- ➢ No disturbance
  - ❑ Use a feasible static schedule
- ➢ Upon detection of a disturbance, determine a dynamic schedule
  - 1 Guaranteed fast response to the disturbance

Problem Overview



Static S

Dynamic $\overline{S}$

All packets meet deadlines

1

2

3

time

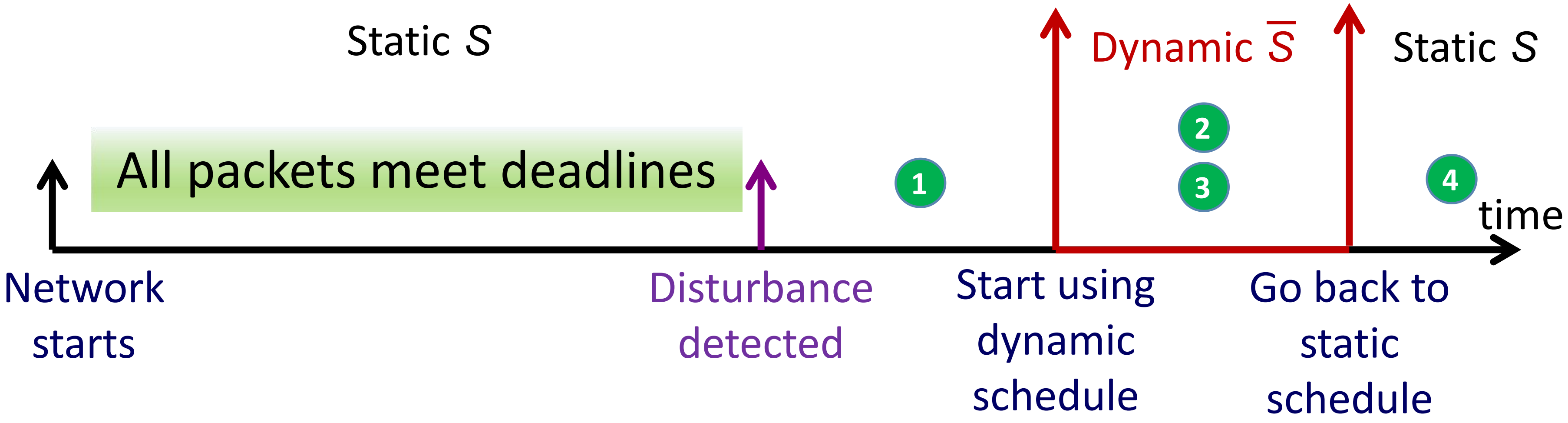
Network starts

Disturbance detected

Start using dynamic schedule

- No disturbance
  - Use a feasible static schedule
- Upon detection of a disturbance, determine a dynamic schedule
  1. Guaranteed fast response to the disturbance
  2. All rhythmic packets meet their deadlines
  3. Fewest periodic packets are dropped

Problem Overview



Static S

All packets meet deadlines

1

Dynamic $\overline{S}$

2

3

Static S

4

time

Network starts

Disturbance detected

Start using dynamic schedule

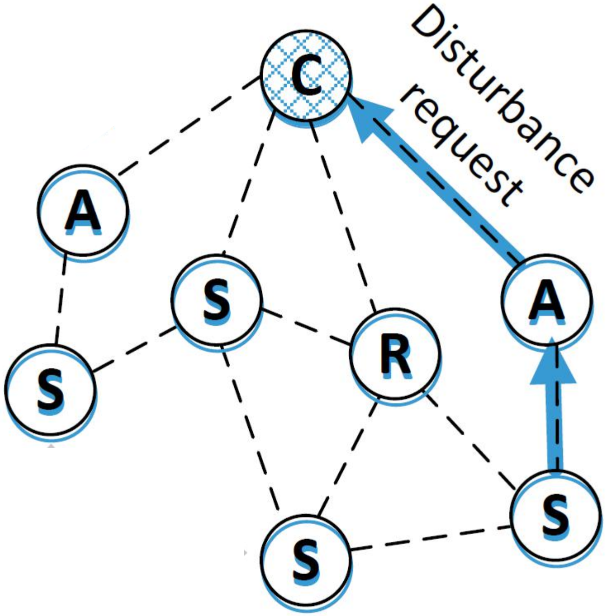
Go back to static schedule

- No disturbance
  - Use a feasible static schedule
- Upon detection of a disturbance, determine a dynamic schedule
  1. Guaranteed fast response to the disturbance
  2. All rhythmic packets meet their deadlines
  3. Fewest periodic packets are dropped
  4. System can safely return to the nominal mode

Centralized Approach

OLS

- Sensor sends a rhythmic event request to the controller/gateway

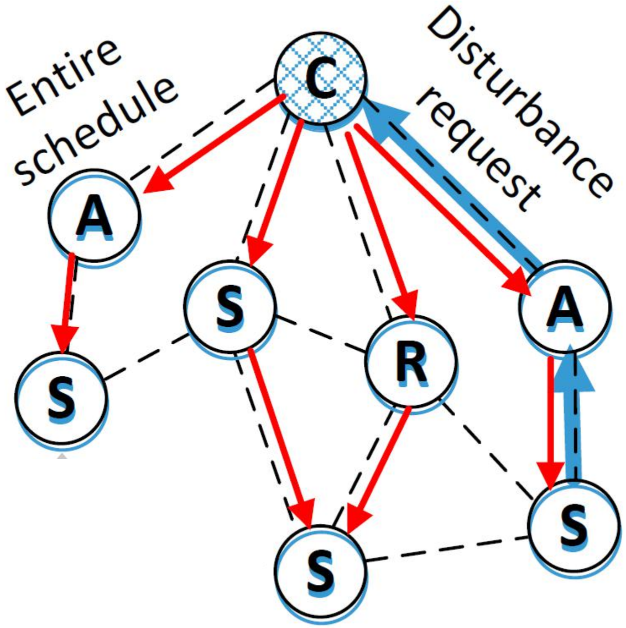


S. Hong, X. Hu, T. Gong and S. Han, *ECRTS* 2015

Centralized Approach

OLS

- Sensor sends a rhythmic event request to the controller/gateway
- Gateway generates and broadcasts a dynamic schedule

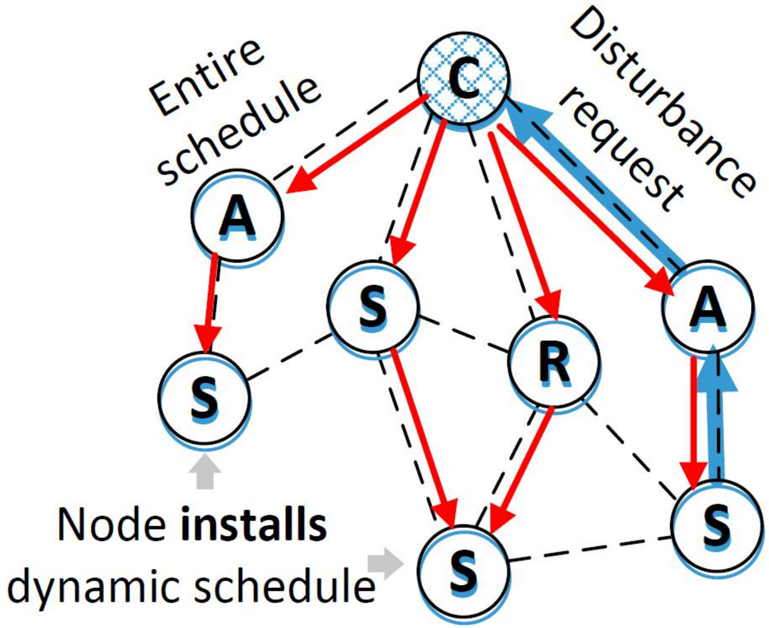


S. Hong, X. Hu, T. Gong and S. Han, *ECRTS* 2015

Centralized Approach

OLS

- Sensor sends a rhythmic event request to the controller/gateway
- Gateway generates and broadcasts a dynamic schedule
- Nodes install and follow a dynamic schedule

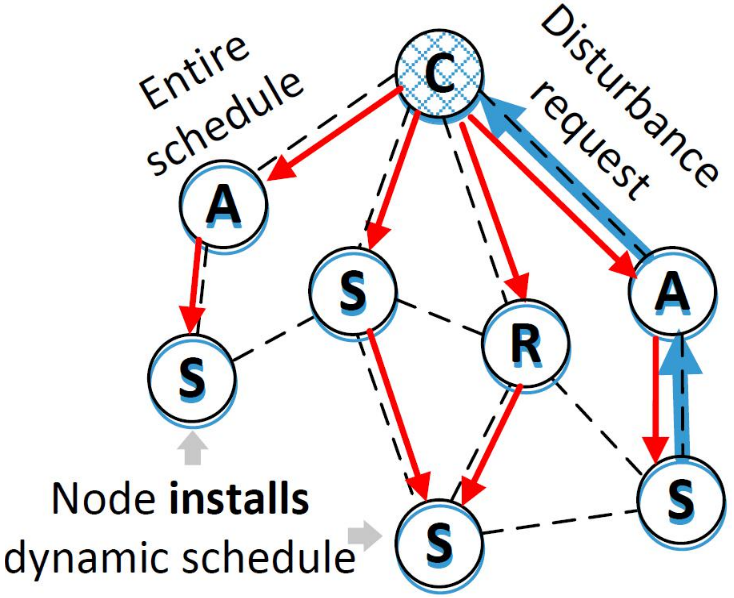


S. Hong, X. Hu, T. Gong and S. Han, *ECRTS* 2015

Centralized Approach

OLS

- Sensor sends a rhythmic event request to the controller/gateway
- Gateway generates and broadcasts a dynamic schedule
- Nodes install and follow a dynamic schedule



👎 **Dynamic programming**

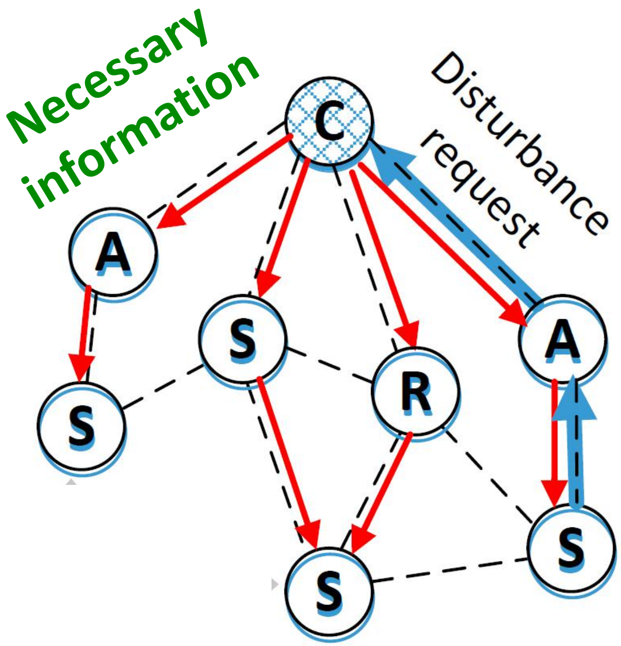
👎 **Drop more packets than necessary**

S. Hong, X. Hu, T. Gong and S. Han, *ECRTS* 2015

Hybrid Approach

D^2-PaS

- Sensor sends a rhythmic event request to the controller/gateway
- Gateway generates and broadcasts **only necessary information**

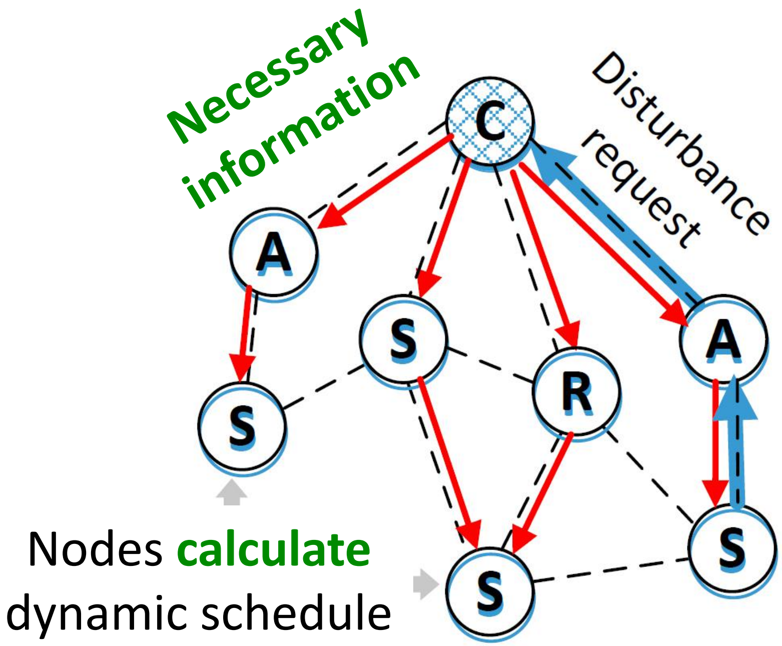


T. Zhang, T. Gong, C. Gu, S. Han, Q. Deng and X. Hu, *RTAS* 2017

Hybrid Approach

D^2-PaS

- Sensor sends a rhythmic event request to the controller/gateway
- Gateway generates and broadcasts **only necessary information**
- Nodes **calculate** a dynamic schedule

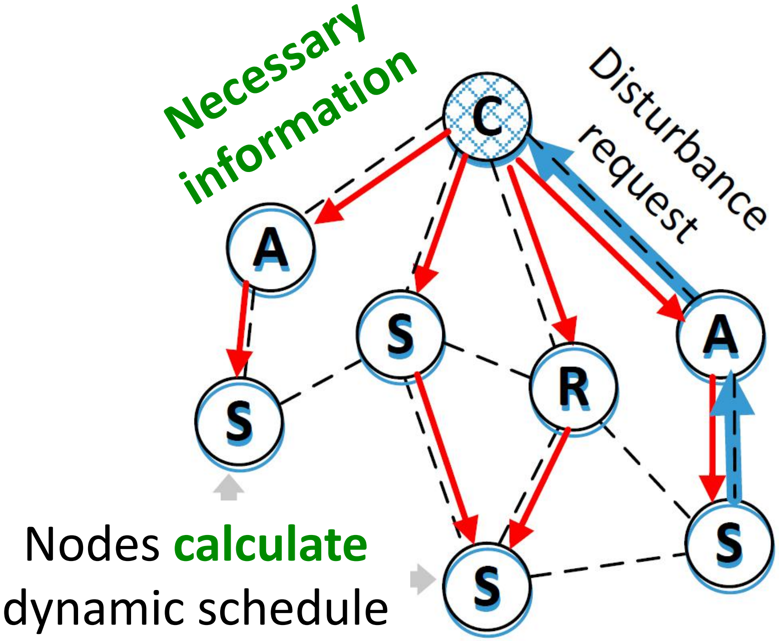


T. Zhang, T. Gong, C. Gu, S. Han, Q. Deng and X. Hu, *RTAS* 2017

Hybrid Approach

D^2-PaS

- Sensor sends a rhythmic event request to the controller/gateway
- Gateway generates and broadcasts **only necessary information**
- Nodes **calculate** a dynamic schedule



Necessary information

Disturbance request

Nodes **calculate** dynamic schedule

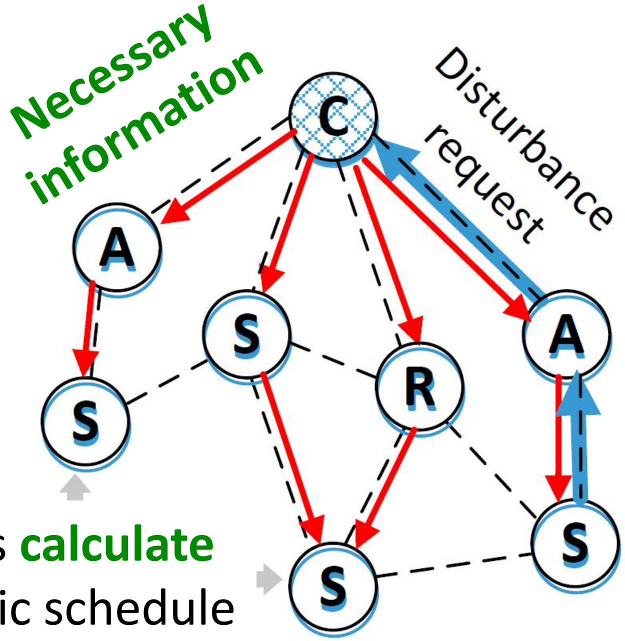
Drop fewer periodic packets

T. Zhang, T. Gong, C. Gu, S. Han, Q. Deng and X. Hu, *RTAS* 2017

Hybrid Approach

D^2-PaS

- Sensor sends a rhythmic event request to the controller/gateway
- Gateway generates and broadcasts **only necessary information**
- Nodes **calculate** a dynamic schedule



Nodes **calculate** dynamic schedule

👍 **Drop fewer periodic packets**

👎 **Long response time to disturbance**

👎 **Rely on a single point (gateway)**

T. Zhang, T. Gong, C. Gu, S. Han, Q. Deng and X. Hu, *RTAS* 2017

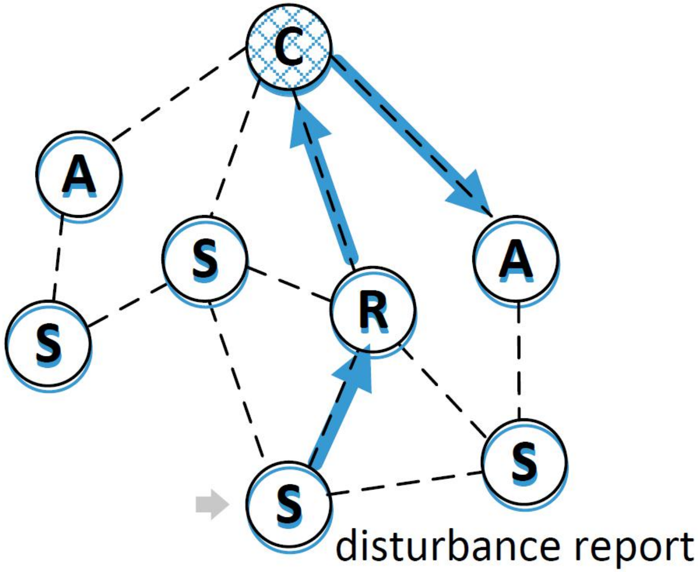
Outline

- System model & related work
- **Fully distrubuted packet scheduling framework (FD-Pas)**
  - □ **Overview**
  - □ **MP-MAC**
  - □ **Dynamic schedule generation**
- Experimental evaluation

Fully Distributed Approach

FD-PaS

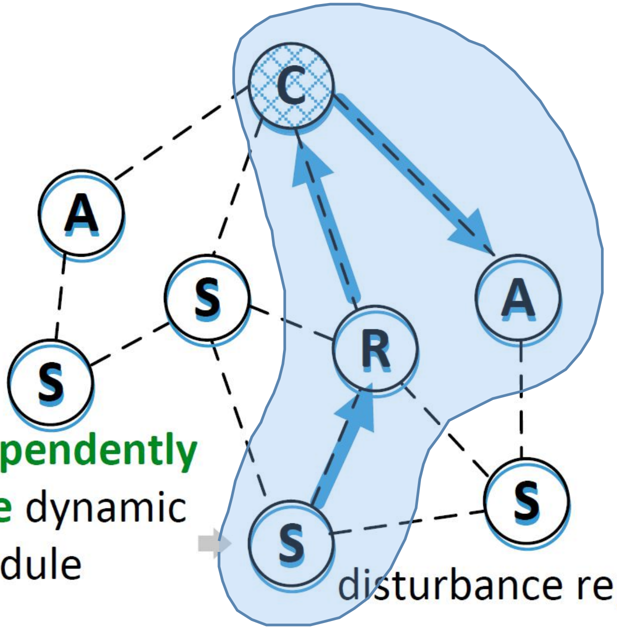
- Sensor sends a rhythmic event report only to **necessary nodes**



Fully Distributed Approach

FD-PaS

- Sensor sends a rhythmic event report only to **necessary nodes**
- Nodes **independently determine** dynamic schedule locally



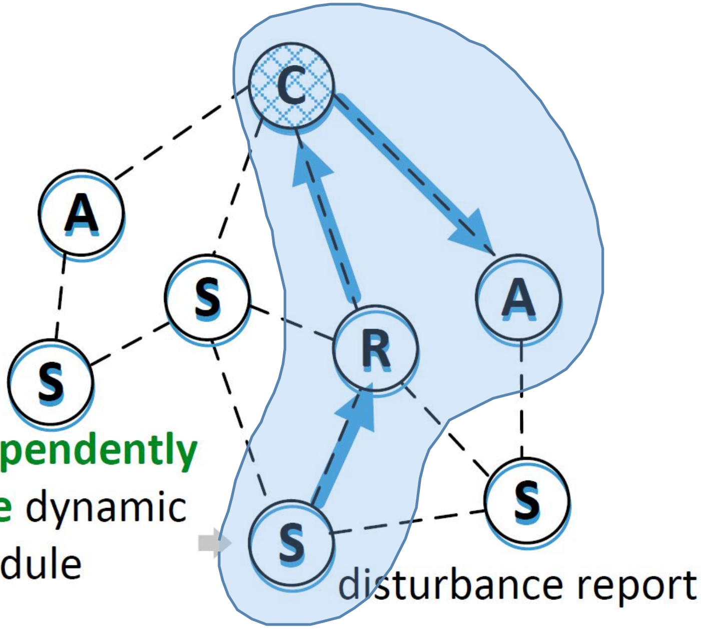
Node **independently determine** dynamic schedule

disturbance report

Fully Distributed Approach

FD-PaS

- Sensor sends a rhythmic event report only to **necessary nodes**
- Nodes **independently determine** dynamic schedule locally



Node **independently determine** dynamic schedule

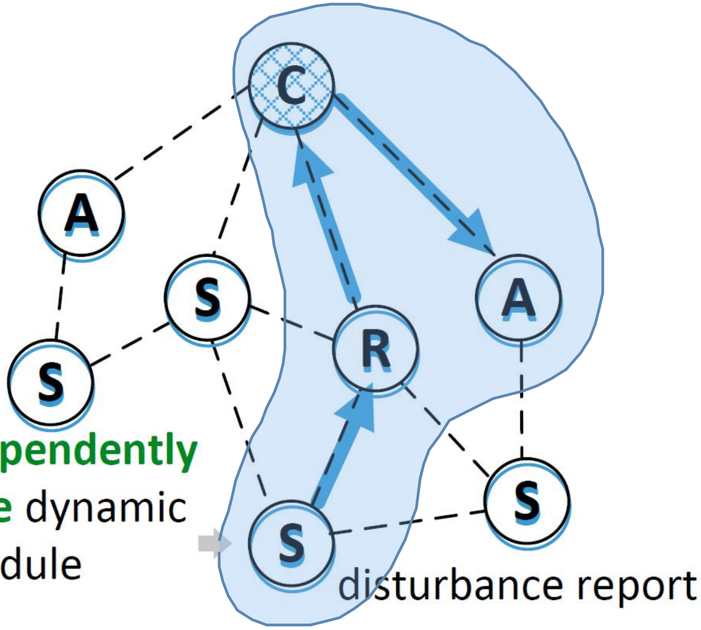
👍 **Fast response to disturbance**

👍 **Don't rely on any single point**

Fully Distributed Approach

FD-PaS

- Sensor sends a rhythmic event report only to **necessary nodes**
- Nodes **independently determine** dynamic schedule locally

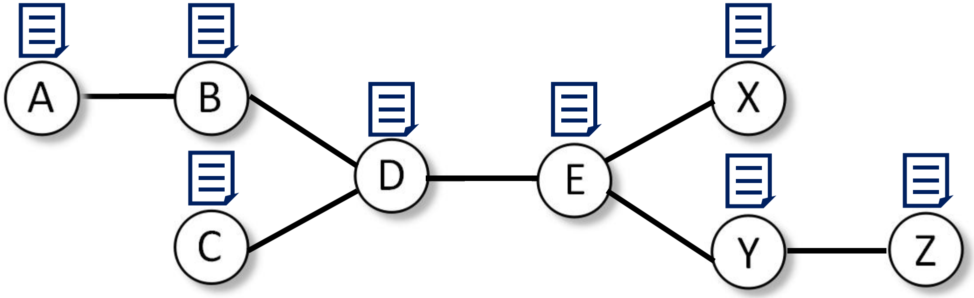


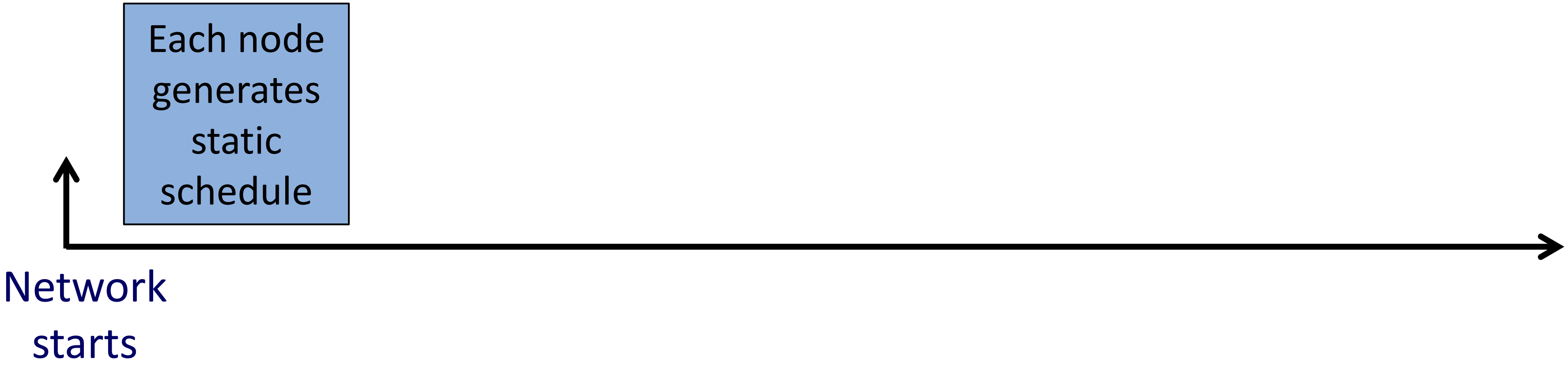
Node **independently determine** dynamic schedule

Q1. **Which** nodes need to know the disturbance?

Q2. **How** these nodes know the disturbance?

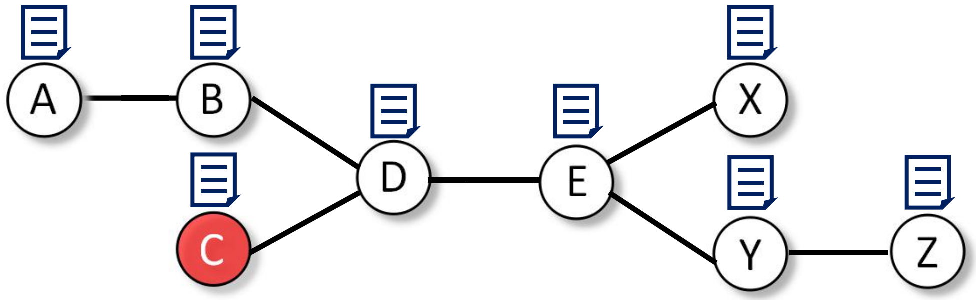
FD-PaS Framework

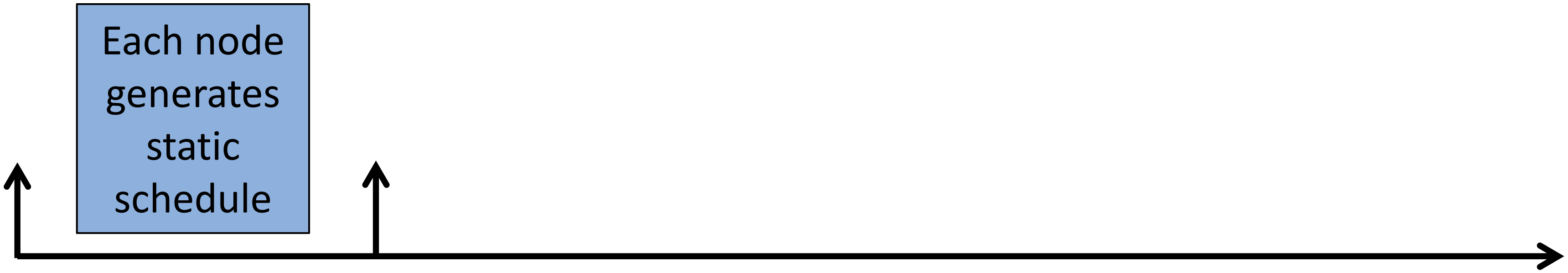



Each node generates static schedule

Network starts

FD-PaS Framework

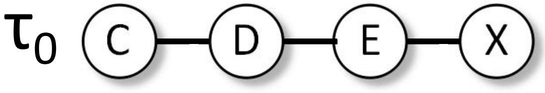




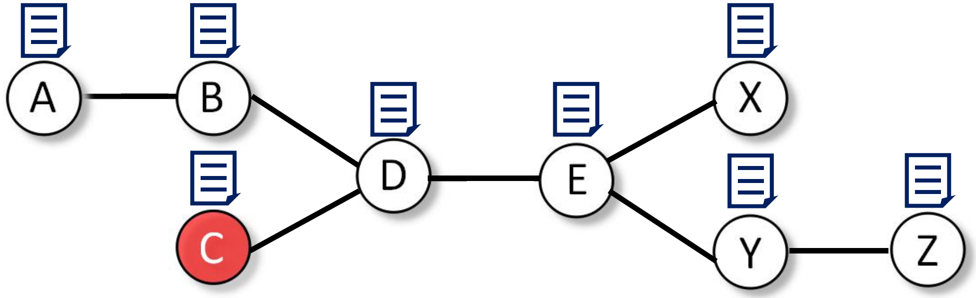
Each node generates static schedule

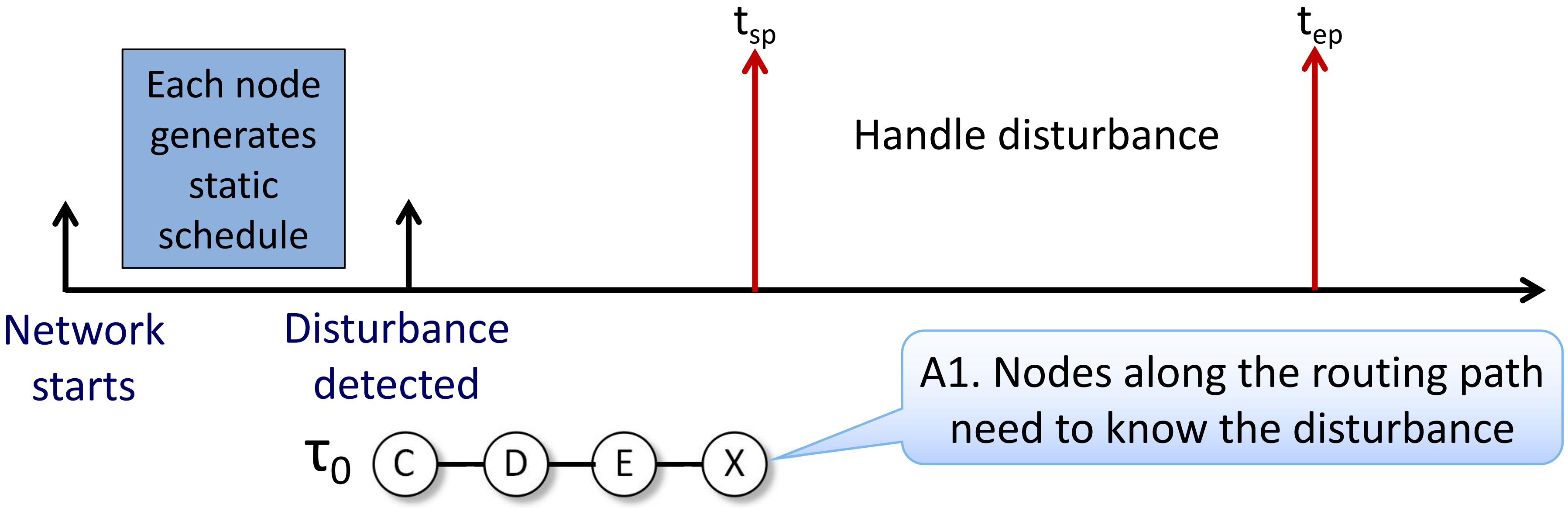
Network starts

Disturbance detected



FD-PaS Framework





Each node generates static schedule

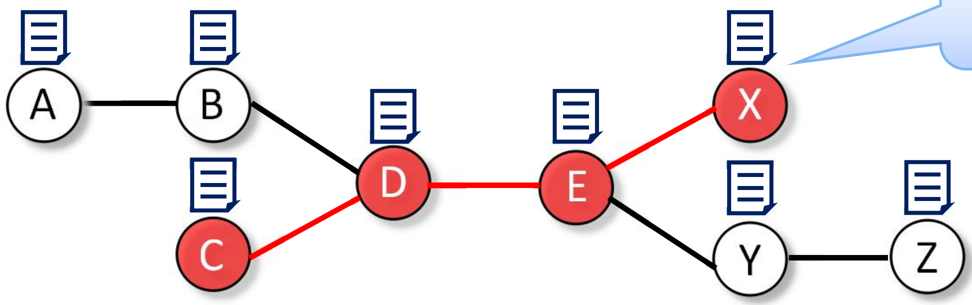
Handle disturbance

Network starts

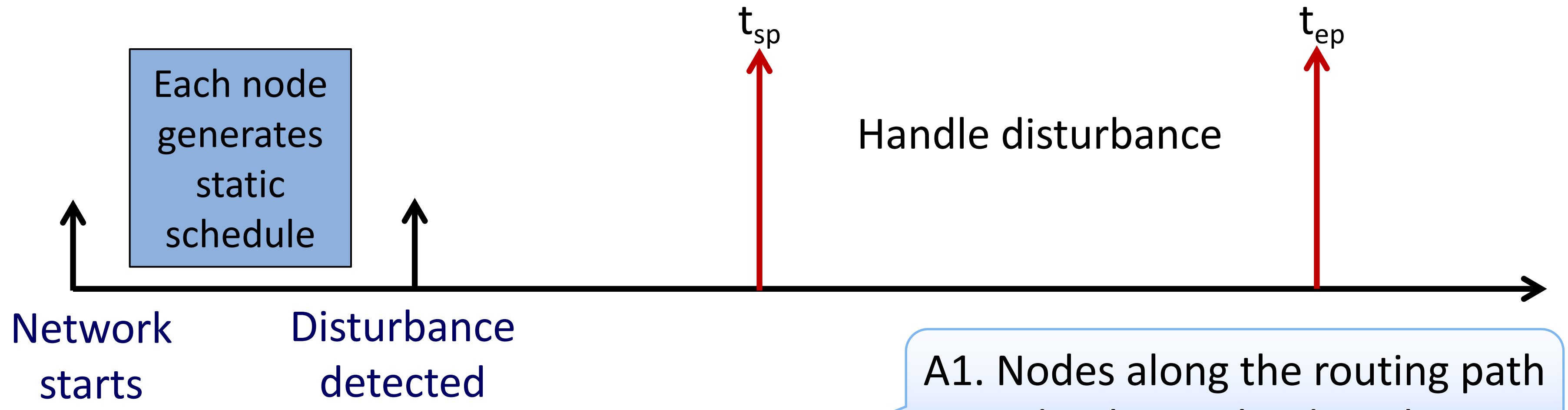
Disturbance detected

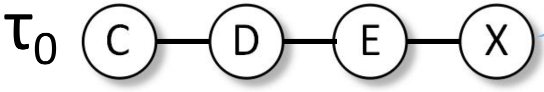
A1. Nodes along the routing path need to know the disturbance

FD-PaS Framework



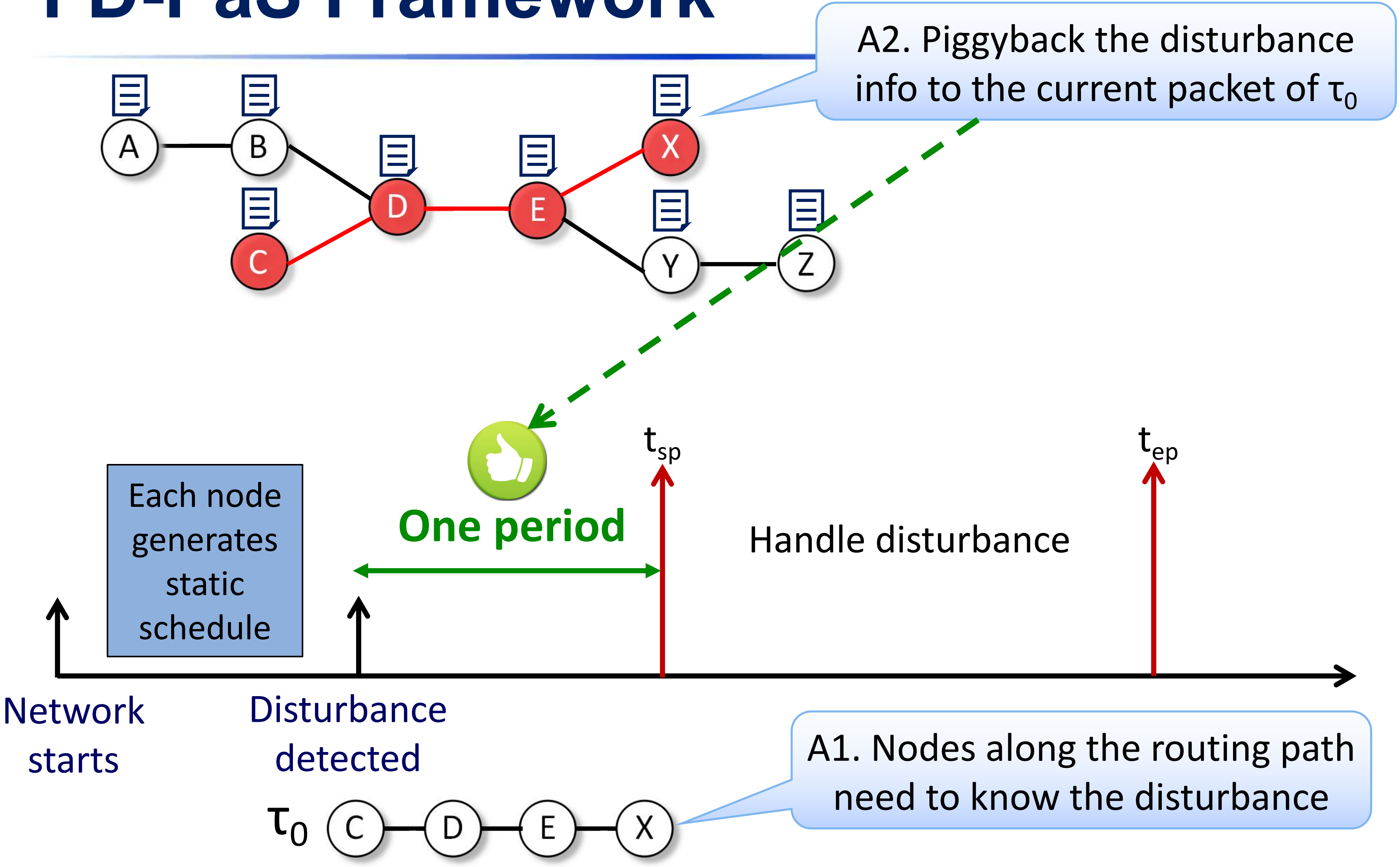
A2. Piggyback the disturbance info to the current packet of τ_0





A1. Nodes along the routing path need to know the disturbance

FD-PaS Framework



A2. Piggyback the disturbance info to the current packet of τ_0

Each node generates static schedule

One period

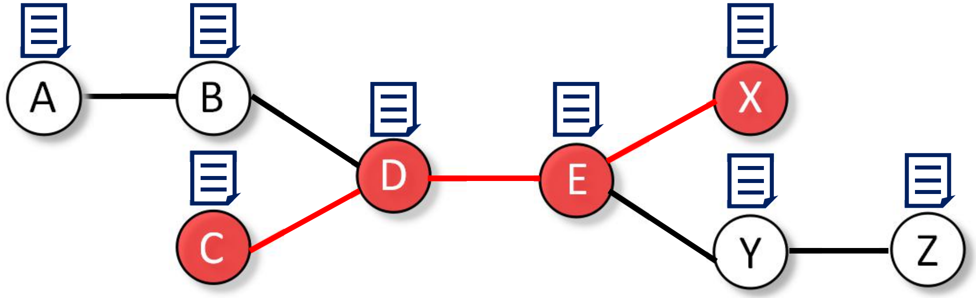
Handle disturbance

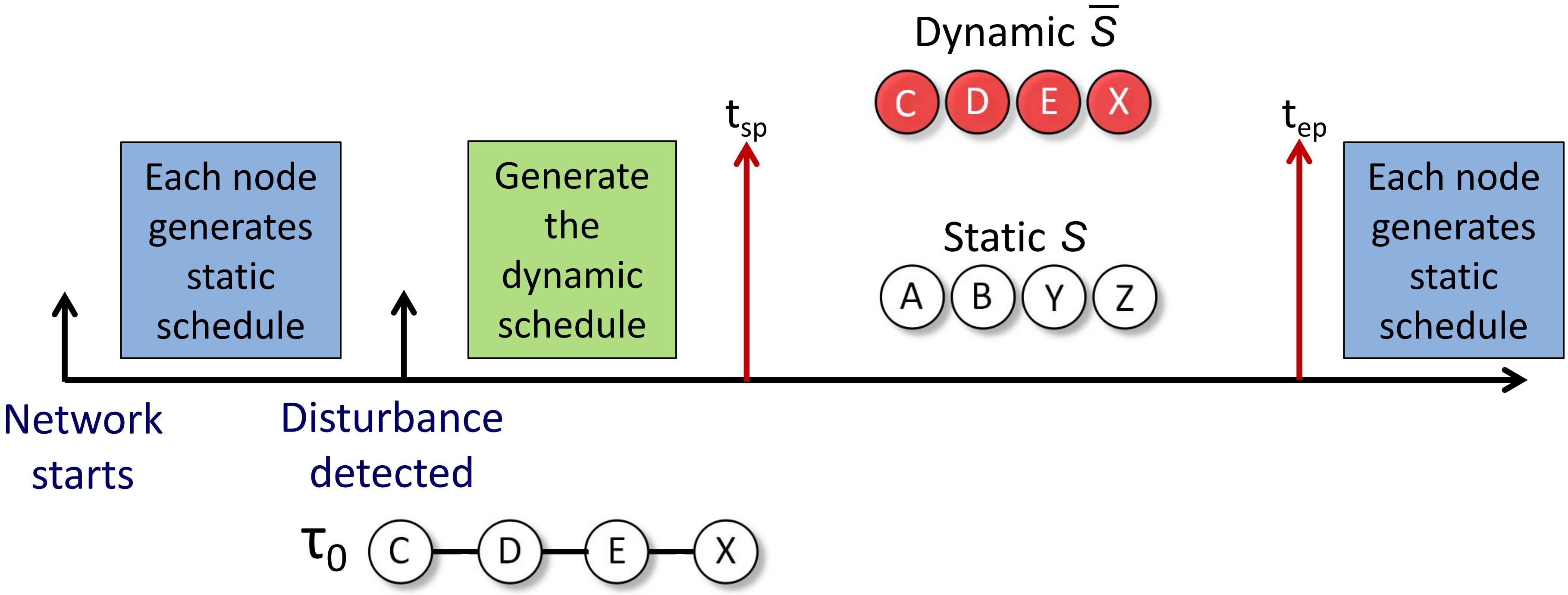
Network starts

Disturbance detected

A1. Nodes along the routing path need to know the disturbance

FD-PaS Framework




Dynamic $\overline{S}$

t_{sp}

t_{ep}

Each node generates static schedule

Generate the dynamic schedule

Static S

Each node generates static schedule

Network starts

Disturbance detected

τ_0

Challenges

- Transmission collisions among different nodes with inconsistent schedule would occur

- An efficient method is needed at each node to determine a dynamic schedule

Challenges

- Transmission collisions among different nodes with inconsistent schedule would occur

MP-MAC (Multi-priority wireless packet preemption)

- An efficient method is needed at each node to determine a dynamic schedule

Challenges

- Transmission collisions among different nodes with inconsistent schedule would occur

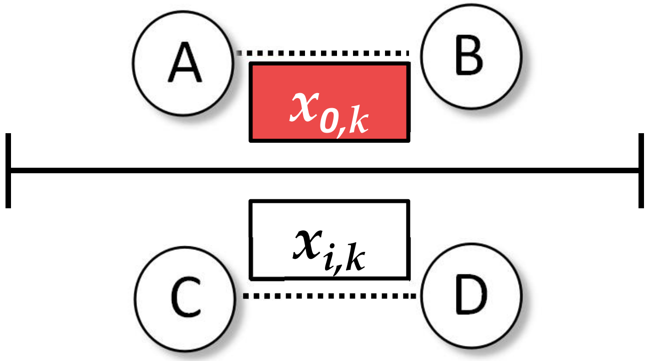
MP-MAC (Multi-priority wireless packet preemption)

- An efficient method is needed at each node to determine a dynamic schedule

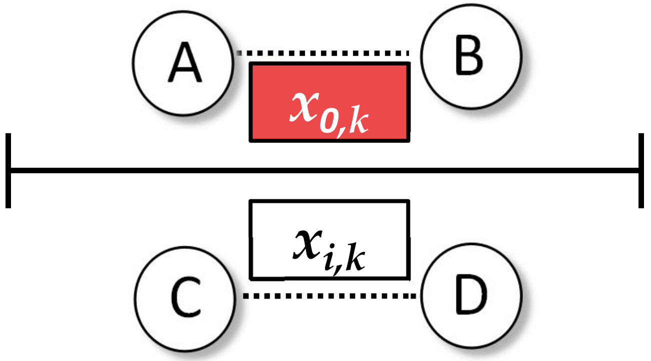
Formulate the packet dropping problem

Introduce an efficient heuristic

Avoid Transmission Collisions

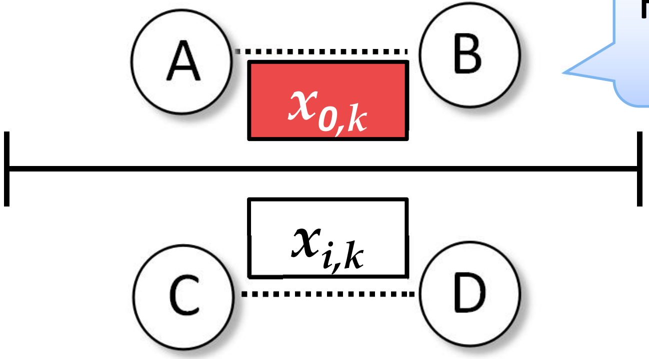


Avoid Transmission Collisions



- Currently
  - Most TDMA-based RTWN protocols employ the Clear Channel Assessment (CCA)
  - CCA cannot prioritize packet transmission

Avoid Transmission Collisions

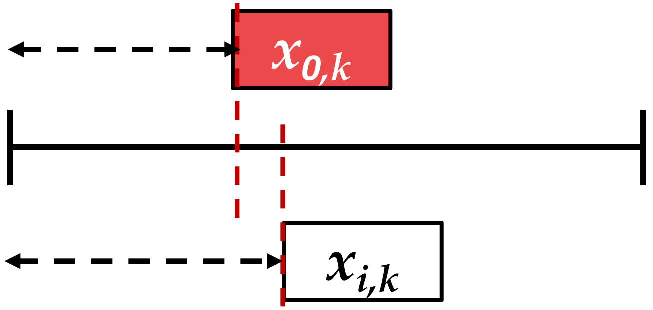


Rhythmic transmissions may be preempted

- Currently
  - Most TDMA-based RTWN protocols employ the Clear Channel Assessment (CCA)
  - CCA cannot prioritize packet transmission
  - No guarantee on which packet is granted the channel access

Multi-Priority MAC (MP-MAC)

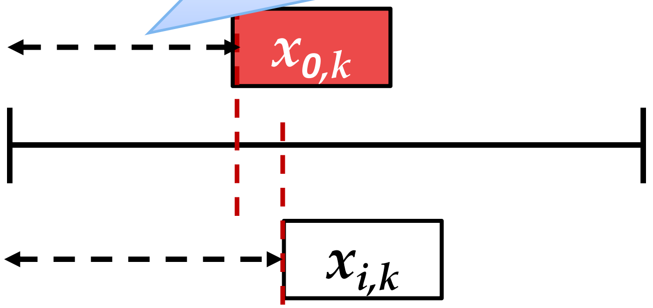
- Give higher priority to rhythmic packets
  - Adjusting the Start-Of-Frame (SOF) time offset to indicate transmission priority



Multi-Priority MAC (MP-MAC)

- Give higher priority to rhythmic packets
  - Adjusting the Start-Of-Frame (SOF) time offset to indicate transmission priority

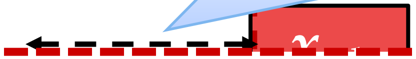
A packet with higher priority is associated with a **shorter Offset** to start the transmission earlier



Multi-Priority MAC (MP-MAC)

- Give higher priority to rhythmic packets
  - Adjusting the Start-Of-Frame (SOF) time offset to indicate transmission priority

A packet with higher priority is associated with a **shorter Offset** to start the transmission earlier



MP-MAC guarantees the rhythmic transmissions in the dynamic schedule are always successful

Challenges

- Transmission collisions among different nodes with inconsistent schedule would occur

Multi-priority wireless packet preemption mechanism

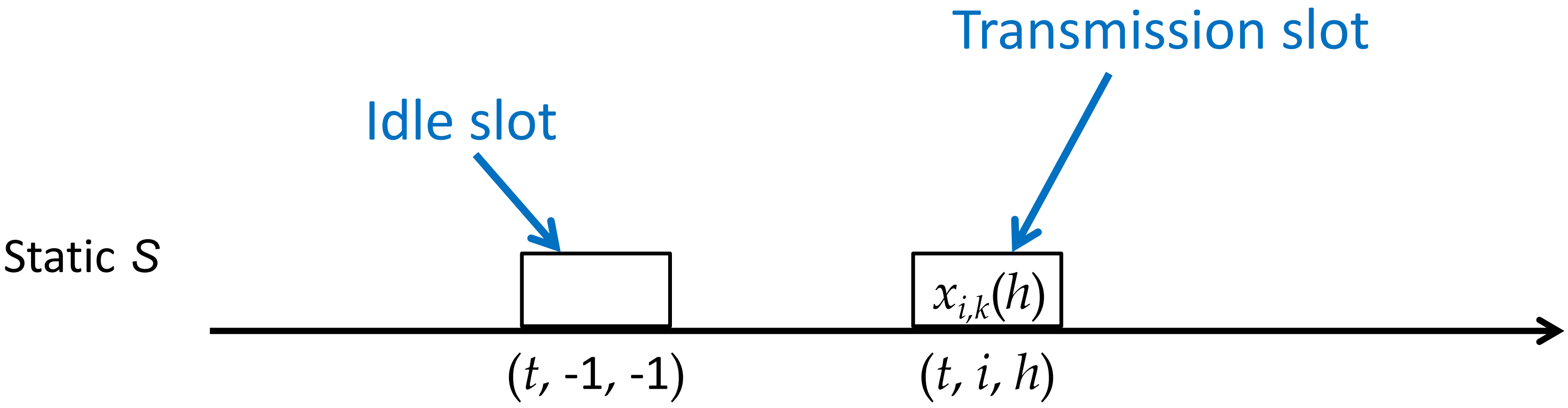
- An efficient method is needed at each node to determine a dynamic schedule

Formulate the packet dropping problem

Introduce an efficient heuristic

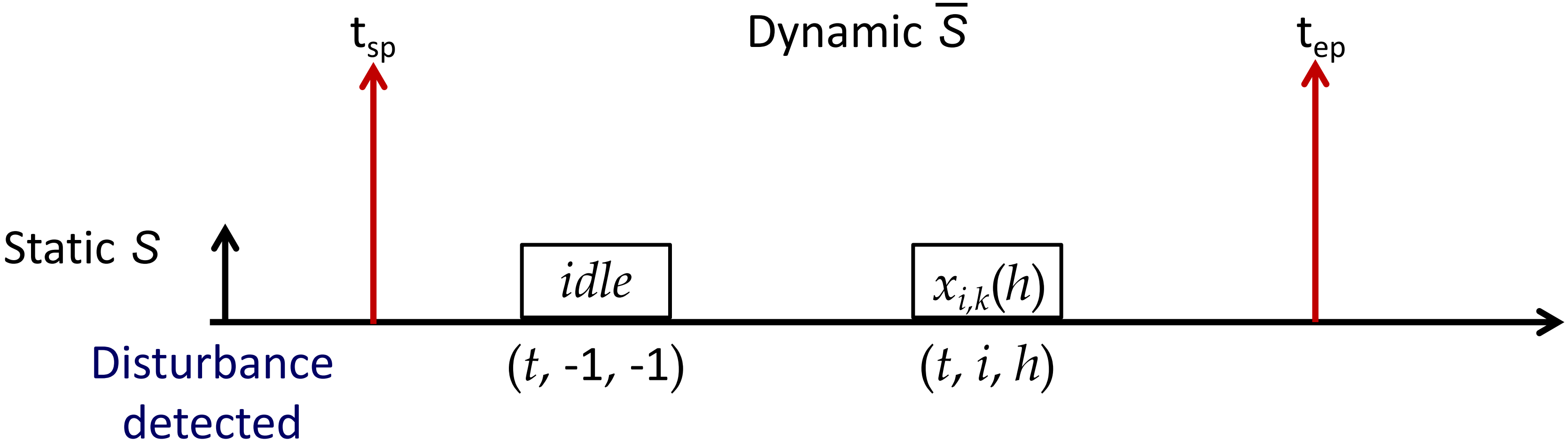
Packet Dropping Problem

- No disturbance: all nodes follow the static schedule



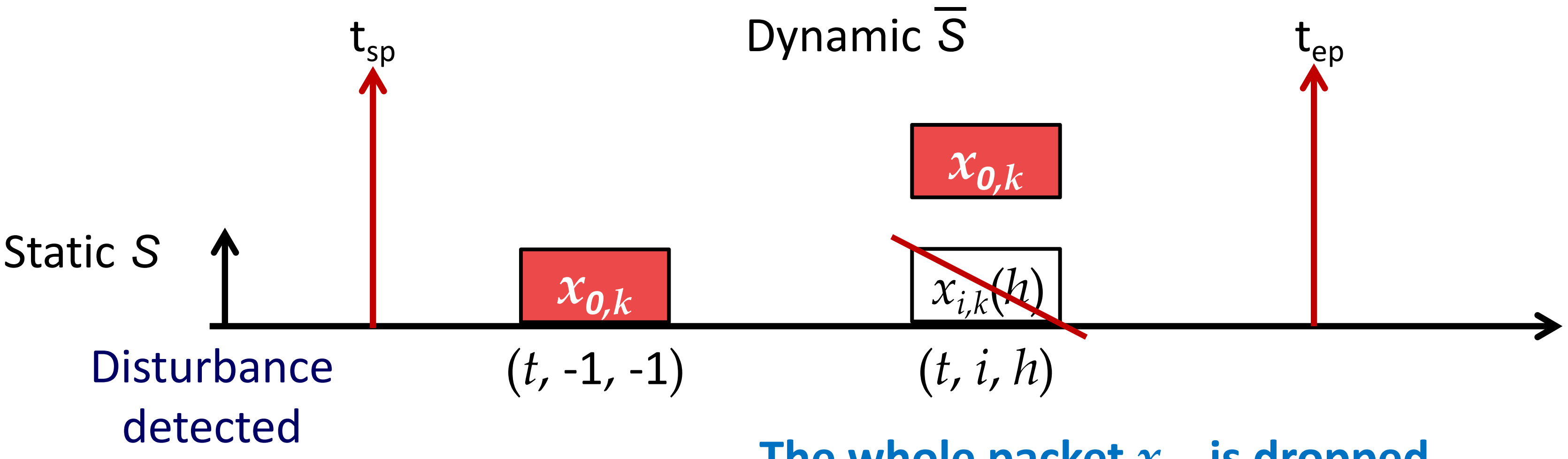
Packet Dropping Problem

- No disturbance: all nodes follow the static schedule
- Disturbance detected: a dynamic schedule is needed to accommodate the increased rhythmic workload $\boxed{x_{0,k}}$



Packet Dropping Problem

- No disturbance: all nodes follow the static schedule
- Disturbance detected: a dynamic schedule is needed to accommodate the increased rhythmic workload
  - □ Use idle slots
  - □ Drop some periodic transmissions

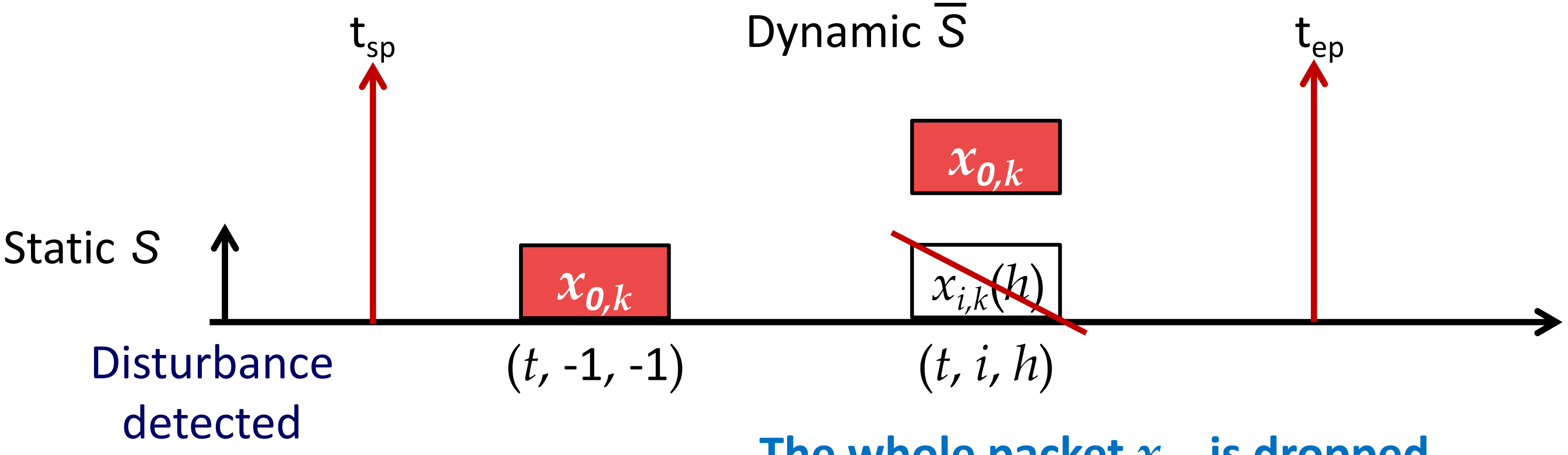


The whole packet $x_{i,k}$ is dropped

Packet Dropping Problem

- No disturbance: all nodes follow the static schedule
- Disturbance detected: a dynamic schedule is needed to accommodate the increased rhythmic workload
  - □ Use idle slots
  - □ Drop some periodic transmissions

How to choose periodic transmissions to be replaced?



The whole packet $x_{i,k}$ is dropped

Packet Dropping Problem Formulation

- Given [t_{sp}, t_{ep}), rhythmic packet set and static schedule *S*, determine the **dynamic schedule $\overline{S}$** in which the fewest periodic packets are dropped and
  - ❑ All rhythmic packets meet their deadlines
  - ❑ Any periodic transmission can only either be replaced or kept unchanged

Strongly NP-Hard!

- **Heuristic**: drop the periodic packet that can give up the most slots to all rhythmic packets

Outline

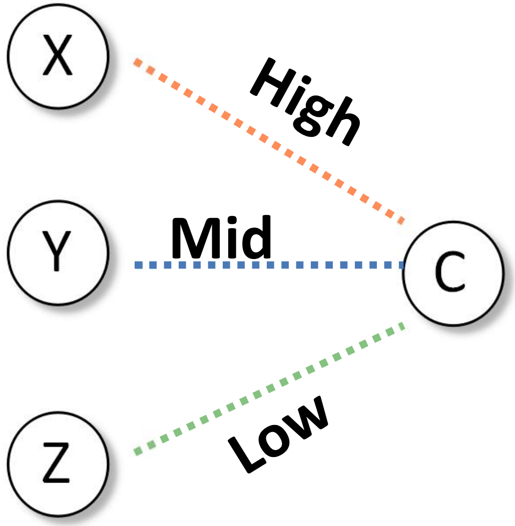
- System model & related work
- Fully distrubuted packet scheduling framework (FD-Pas)
- **Experimental evaluation**
  - ❑ **Testbed**
  - ❑ **Simulation**

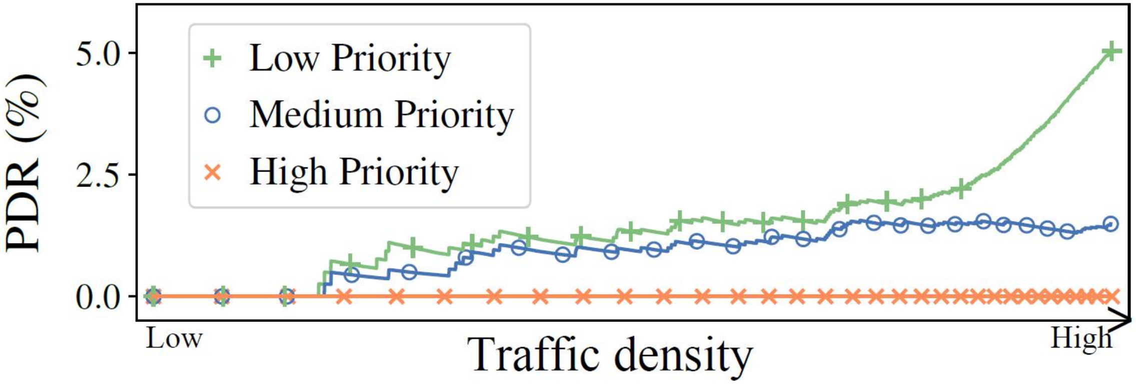
Testbed

- FD-PaS on a 6TiSCH testbed (a real-time IoT protocol)
- MP-MAC through enhancing the slot timing in the data link layer
- Dynamic schedule generation in the application layer

MP-MAC Validation

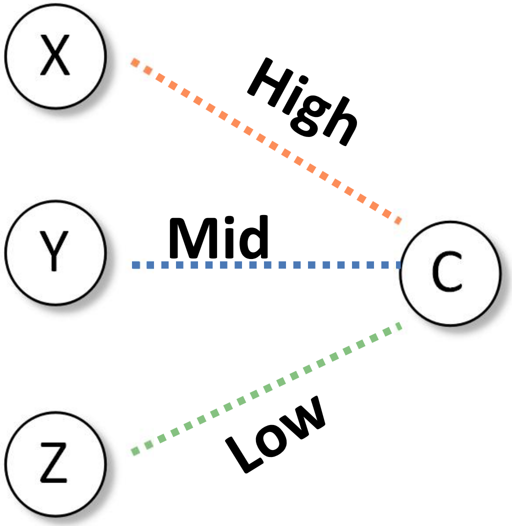
- Functional correctness
  - Higher priority packets can preempt lower ones

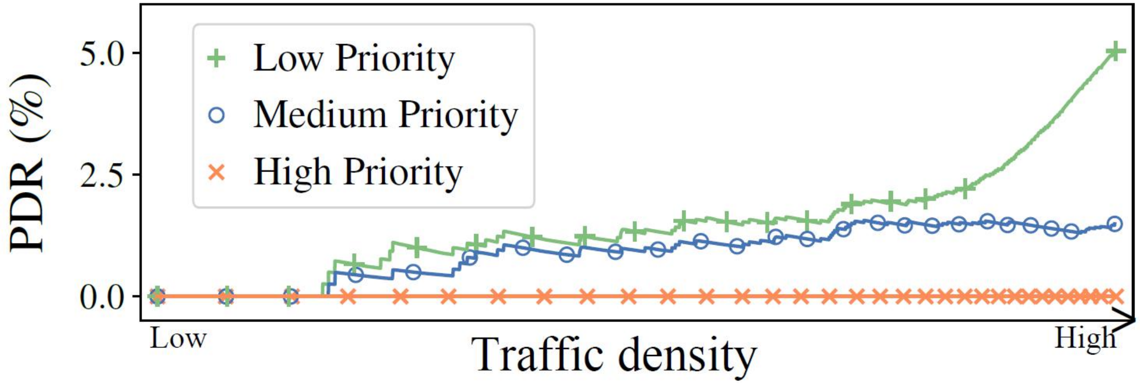




MP-MAC Validation

- Functional correctness
  - Higher priority packets can preempt lower ones

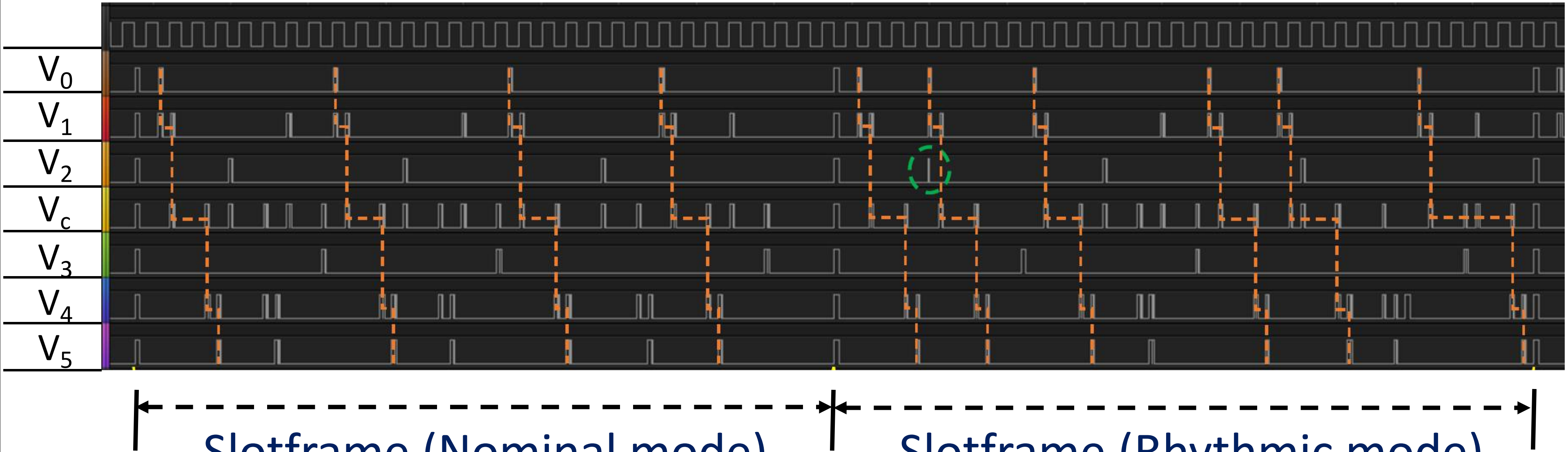




More experimental results in the paper and join us at our demo

FD-PaS Validation

- Functional validation in a multi-task multi-hop RTWN
  - Use a logic analyzer to capture the radio activities from a pin of each device



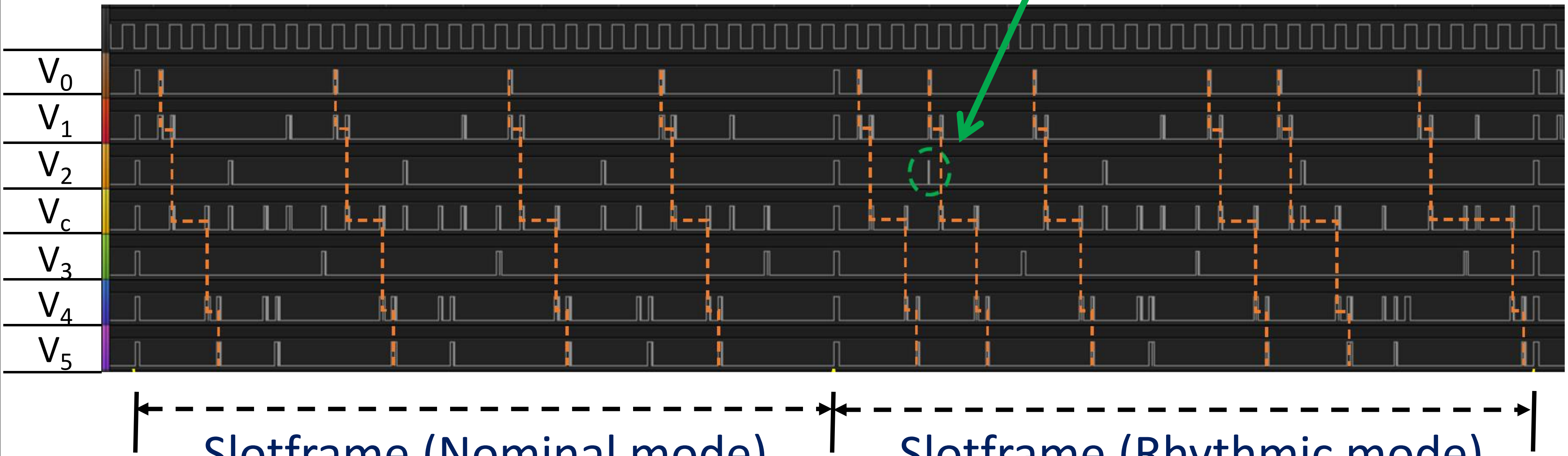
Slotframe (Nominal mode)

Slotframe (Rhythmic mode)

FD-PaS Validation

- Functional validation in a multi-task multi-hop RTWN
  - ❑ Use a logic analyzer to capture the radio activities from a pin of each device

Rhythmic transmission preempt a periodic transmission

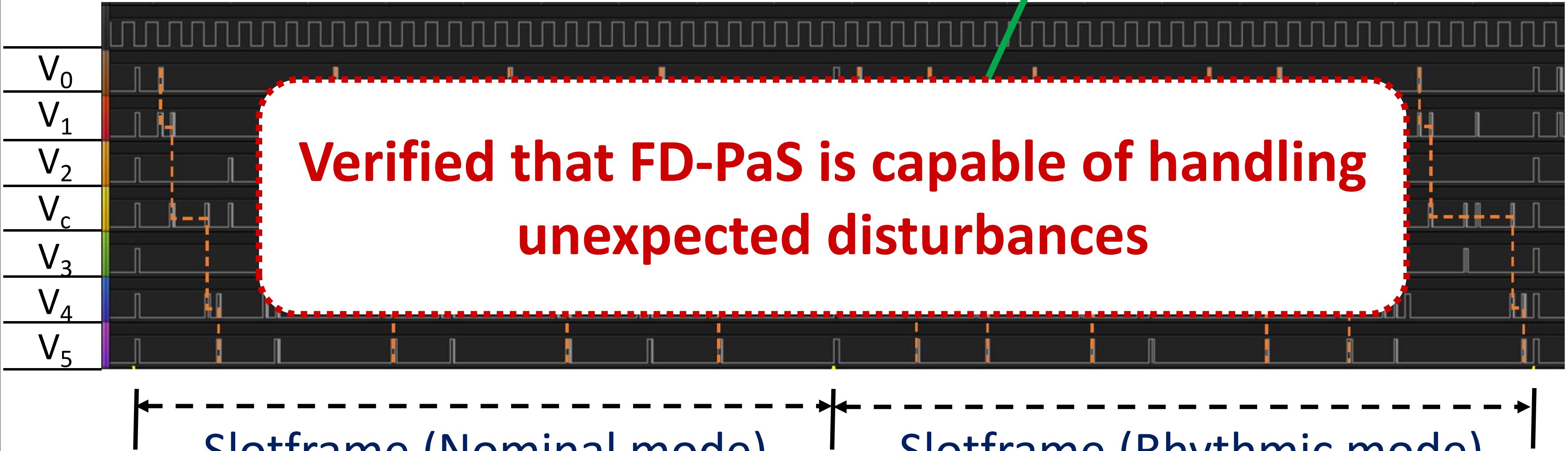


Slotframe (Nominal mode)

Slotframe (Rhythmic mode)

FD-PaS Validation

- Functional validation in a multi-task multi-hop RTWN
  - ❑ Use a logic analyzer to capture the radio activities from a pin of each device



Rhythmic transmission preempt a periodic transmission

Verified that FD-PaS is capable of handling unexpected disturbances

Simulation

Setup

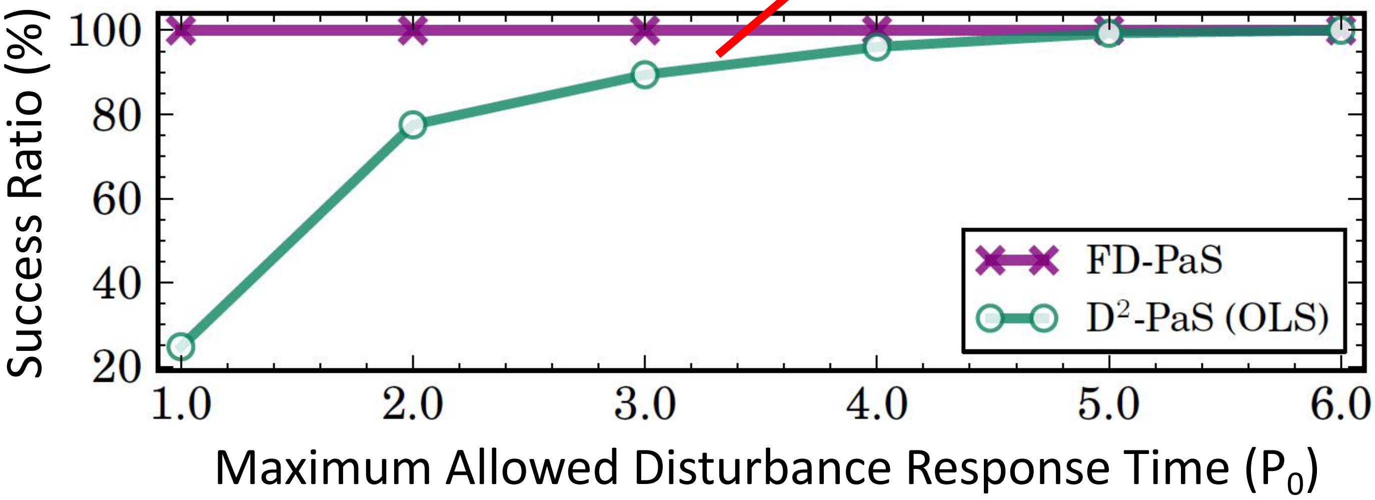
- Randomly generated task sets (based on realistic RTWN applications)
- Compare with OLS and D^2-PaS

Evaluation metrics

- **How fast is FD-PaS in responsing a disturbance?**
  - Success ratio (SR) = Feasible task sets / All the generated task sets.
- **How effective is FD-PaS in reducing dropped packets?**
  - Drop rate (DR) = Number of dropped packets / Total number of generated packets.

Simulation Results

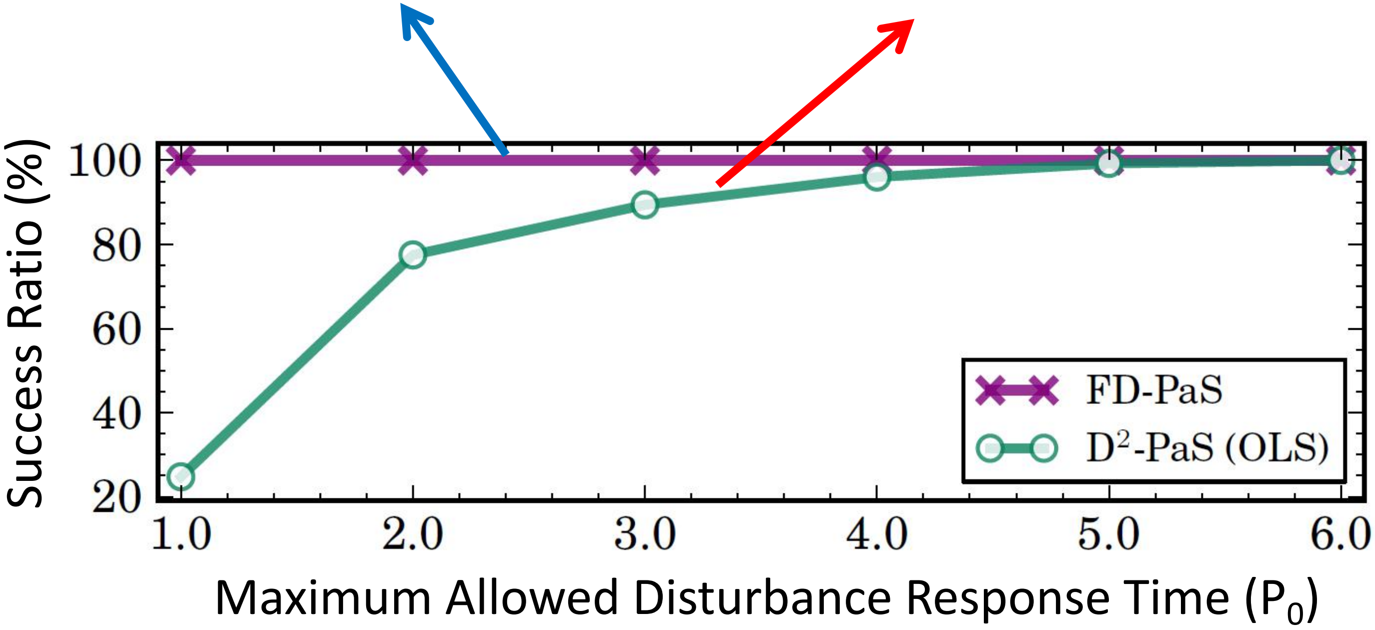
OLS and D²-PaS only feasible if DRT ≥ 6 periods



Simulation Results

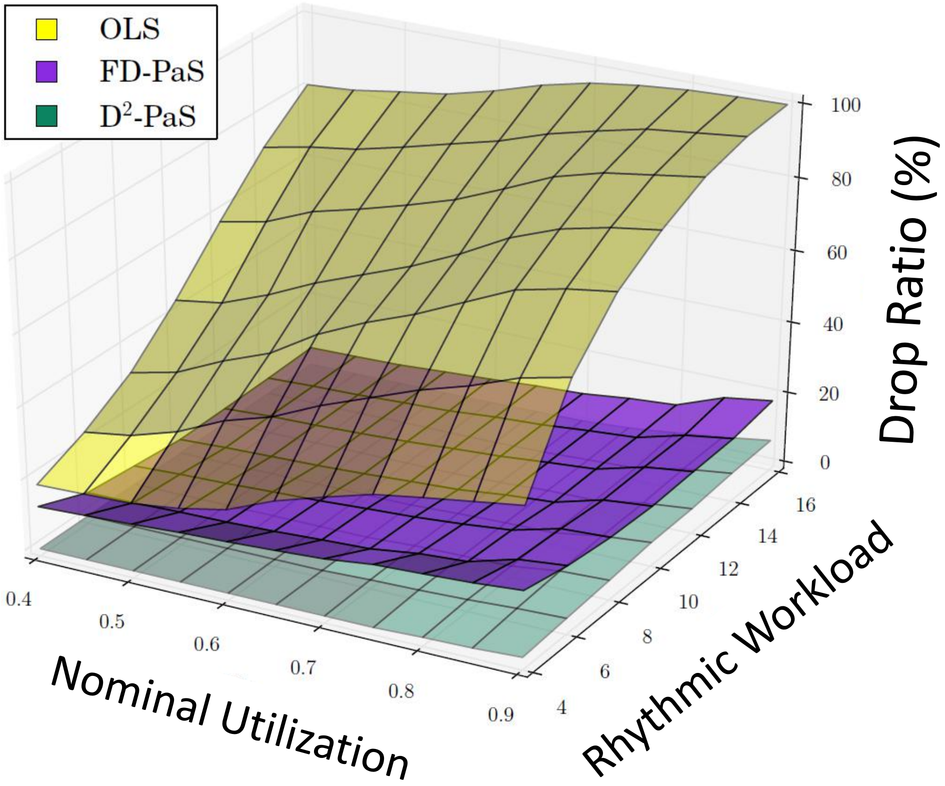
FD-PaS can always achieve 100% SR

OLS and D²-PaS only feasible if DRT ≥ 6 periods



Simulation Results

- FD-PaS has significantly lower DR over OLS (82% max and 53% on avg.)
- Compared to D²-PaS, FD-PaS drops around 12% more packets on average



Summary and Future Work

- Summary
  - Proposed the **first fully distributed** dynamic **fast response** framework for handling disturbances in RTWNs
    - Colission avoidance
    - Packet dropping

Summary and Future Work

- Summary
  - Proposed the **first fully distributed** dynamic **fast response** framework for handling disturbances in RTWNs
    - Colission avoidance
    - Packet dropping
  - Implemented the proposed framework on a testbed
  - Validated the correctness of the framework on the testbed

Summary and Future Work

- Summary
  - Proposed the **first fully distributed** dynamic **fast response** framework for handling disturbances in RTWNs
    - Colission avoidance
    - Packet dropping
  - Implemented the proposed framework on a testbed
  - Validated the correctness of the framework on the testbed
  - Evaluated the effectiveness of the framework

Summary and Future Work

- Summary
  - Proposed the **first fully distributed** dynamic **fast response** framework for handling disturbances in RTWNs
    - Colission avoidance
    - Packet dropping
  - Implemented the proposed framework on a testbed
  - Validated the correctness of the framework on the testbed
  - Evaluated the effectiveness of the framework
- Future work
  - Handle concurrent disturbances
  - Consider unreliable networks
  - Support multiple communication channels

Thank you!

Questions?

东北大学 NORTHEASTERN UNIVERSITY 1923

UCONN
UNIVERSITY OF CONNECTICUT

VITA DULCEDO SPES

UNIVERSITY OF
NOTRE DAME