IOWA

Topics in Computer Science II: Real-Time Cyber-Physical Systems | Fall 2025

DistributedHART: A Distributed Real-Time Scheduling System for WirelessHART Networks

Matt Krueger

Oct. 15, 2025

1

Introduction

Background

- 2019 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)
- Wayne State University, Detroit, MI, USA
- Missouri University of Science and Technology, Rolla, MO, USA

DistributedHART: A Distributed Real-Time Scheduling System for WirelessHART Networks

Venkata P. Modekurthy[†*], Abusayeed Saifullah[†*] and Sanjay Madria[‡]
[†]Department of Computer Science, Wayne State University, Detroit, MI, USA
[‡]Department of Computer Science, Missouri University of Science & Technology, Rolla, MO, USA

***Abstract*—Industry 4.0 is a new industry trend which relies on data driven business model to set the productivity requirements of the cyber physical system. To meet this requirement, Industry 4.0 cyber physical systems need to be highly scalable, adaptive, real-time, and reliable. Recent successful industrial wireless standards such as WirelessHART appeared as a feasible approach for such cyber physical systems. For reliable and real-time communication in highly unreliable environments, they adopt a high degree of redundancy. While a high degree of redundancy is crucial to real-time control, it causes a huge waste of energy, bandwidth, and time under a centralized approach, and are therefore less suitable for scalability and handling network dynamics. To address these challenges, we propose DistributedHART - a distributed real-time scheduling system for WirelessHART networks. The essence of our approach is to adopt local (node-level) scheduling through a time window allocation among the nodes that allows each node to schedule its transmissions using a real-time scheduling policy locally and online. DistributedHART obviates the need of creating and disseminating a central global schedule in our approach, and thereby significantly reducing resource usage and enhancing the scalability. To our knowledge, it is the first distributed real-time multi-channel scheduler for WirelessHART. We have implemented DistributedHART and experimented on a 130-node testbed. Our testbed experiments as well as simulations show at least 85% less energy consumption in DistributedHART compared to existing centralized approach while ensuring similar schedulability.**

I. Introduction

Industry 4.0 is a new industry trend which relies on a data-driven business model to set the productivity requirements of the cyber-physical systems. To meet these requirements, cyber-physical systems need to be highly scalable, adaptive, real-time and reliable. Recent successful industrial wireless standards such as WirelessHART have shown their feasibility as a cost-efficient, real-time, and robust approach for such cyber-physical systems [7]. To make reliable and real-time communication in highly unreliable wireless environments, WirelessHART adopts a high degree of redundancy using a Time Division Multiple Access (TDMA) based Media Access Control (MAC) protocol. A time slot can be either *dedicated* (i.e., a time slot when at most one transmission is scheduled to a receiver) or *shared* (i.e., a time slot when multiple nodes may contend to send to a common receiver). To handle transmission failures, each node on a path from a sensor to an actuator is assigned two dedicated time slots and a third shared slot on a separate path for another retransmission [2]. A network manager creates the transmission schedule **centrally** and in advance for all nodes in the network and then disseminates them. A centralized WirelessHART scheduler with high redundancy raises several practical challenges in achieving scalability as described below.

High level of redundancy in centralized algorithms [27], [29] causes a huge waste of time and bandwidth, and hence is not scalable. For example, if the transmission of a packet along a particular link succeeds, all time slots (on the current link and redundant links) that were assigned to handle its failure remain unused. Similarly, if it fails along that particular link, all time slots that were assigned for its subsequent links to handle a successful transmission remain unused. Our experiments observed up to 70% unused time slots in WirelessHART networks (see Section IV). Furthermore, there can be events or emergencies that occur unpredictably or aperiodically. For example, a WirelessHART network in an oil-refinery may suddenly detect a safety valve displacement requiring immediate attention to avoid accidents. Existing solution handles emergencies by allocating time slots in the centrally created schedule and by stealing slots in the absence of emergencies [17]. However, this approach leaves most of the slots of the periodic server unstolen, and hence unused. Thus the network remains largely underutilized which affects the scalability of the system.

Schedules dissemination in centralized algorithm consumes bandwidth, energy, and time, even for a smaller network or a smaller workload. Typically, hyper-period and length of the schedule increase exponentially with the increase in the number of flows or their periods, which hinders the scalability of the network. Note that, in general, periods can be non-harmonic to ensure stability or control performance [28]. In Industry 4.0, the data-driven business model introduces frequent changes to sampling rates, which requires re-configuration and re-dissemination of schedules. In addition, network dynamics such as a change in channel, node, or link condition requires reconfiguration and re-dissemination of schedules [29]. Such frequent re-dissemination of the schedule consumes very high energy, time, and bandwidth. Thus, a fully centralized WirelessHART scheduling is less suitable for cyber-physical systems, especially those in the domain of Industry 4.0 [6]. Besides, it is typically suitable for deterministic traffic patterns such as periodic traffics or traffics with known arrival pattern.

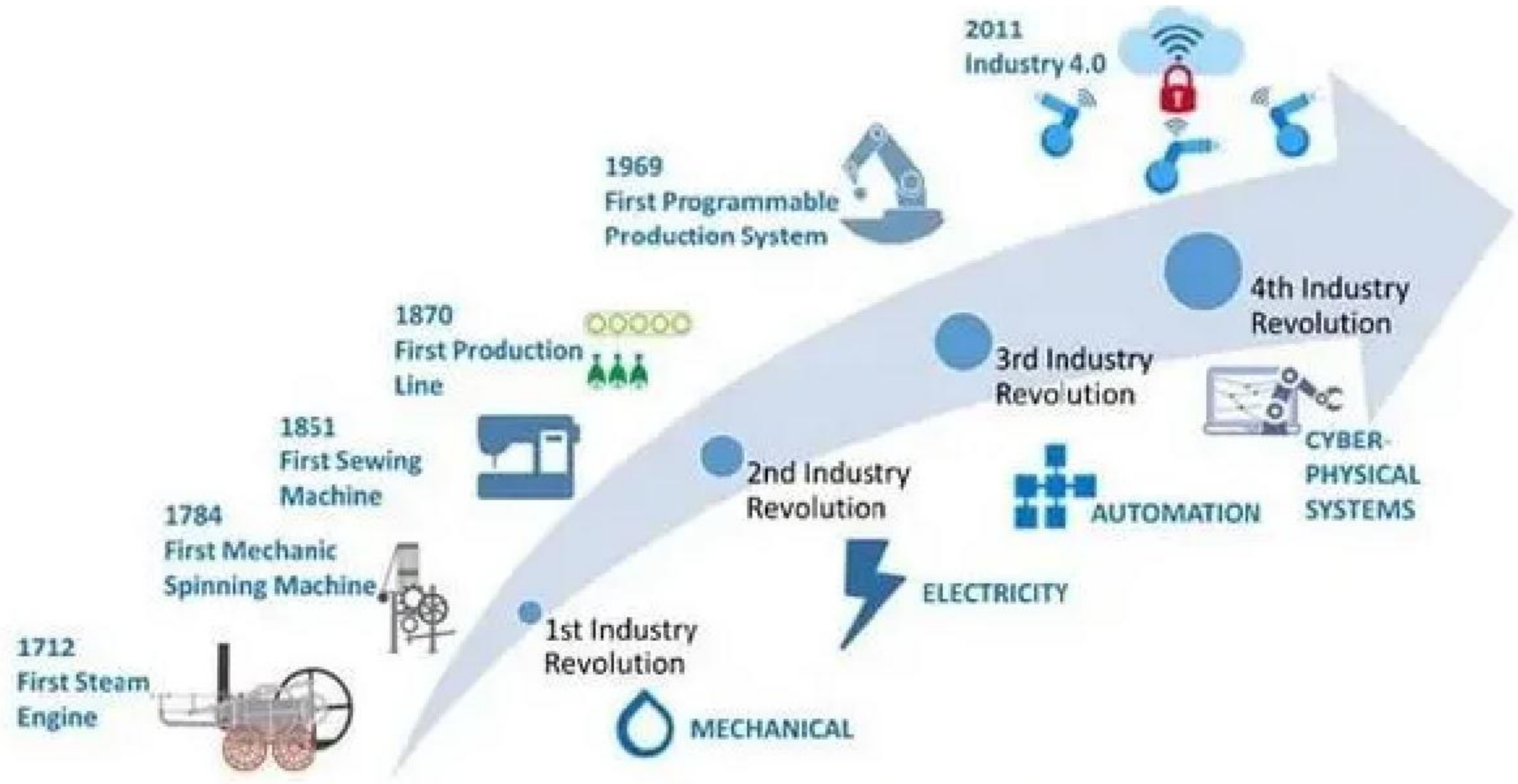
To address the above limitations, in this paper, we propose a distributed real-time scheduling system for WirelessHART networks. Designing a distributed TDMA protocol with scheduling

[*]co-first author

978-1-7281-0678-6/19/$31.00 ©2019 IEEE
DOI 10.1109/RTAS.2019.00026

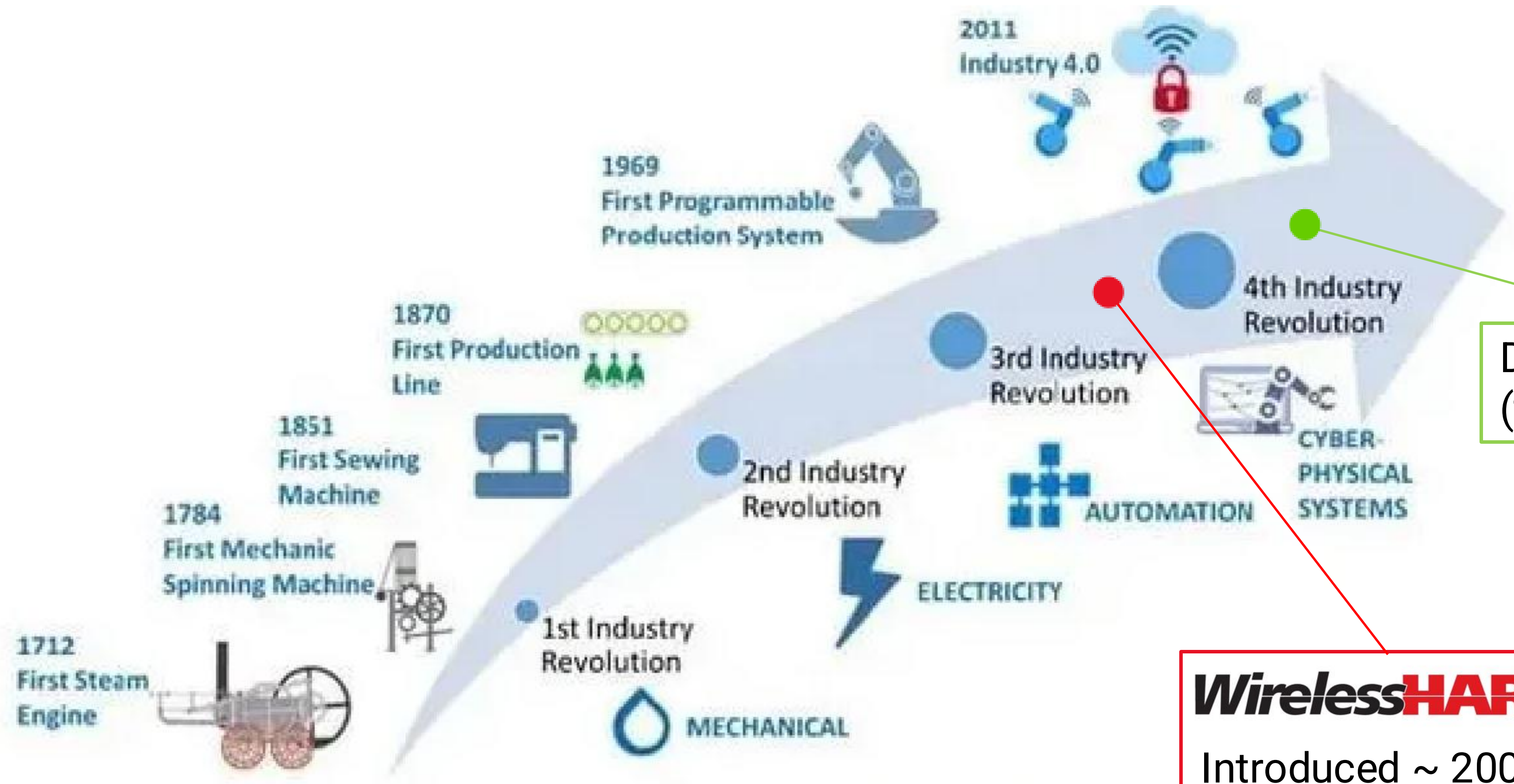
IOWA

Industry 4.0



2011
Industry 4.0
1969
First Programmable
Production System
1870
First Production
Line
1851
First Sewing
Machine
1784
First Mechanic
Spinning Machine
1712
First Steam
Engine
1st Industry
Revolution
2nd Industry
Revolution
3rd Industry
Revolution
4th Industry
Revolution
MECHANICAL
ELECTRICITY
AUTOMATION
CYBER-
PHYSICAL
SYSTEMS

Industry 4.0



DistributedHART
(this paper) 2019

WirelessHART
Introduced ~ 2007

Motivation of DistributedHART

Address WirelessHART's limitations:

- Industry 4.0 suitability
- Network utilization
- Energy intensive

Maintain Performance:

- Scalable
- Adaptive
- Reliable
- Real-time

Limitation 1: High Level of Redundancy

Multiple paths

- Exponential scalability
- Underutilization under periodic conditions

70% unused slots [27]

- 30 flows
- 69 nodes

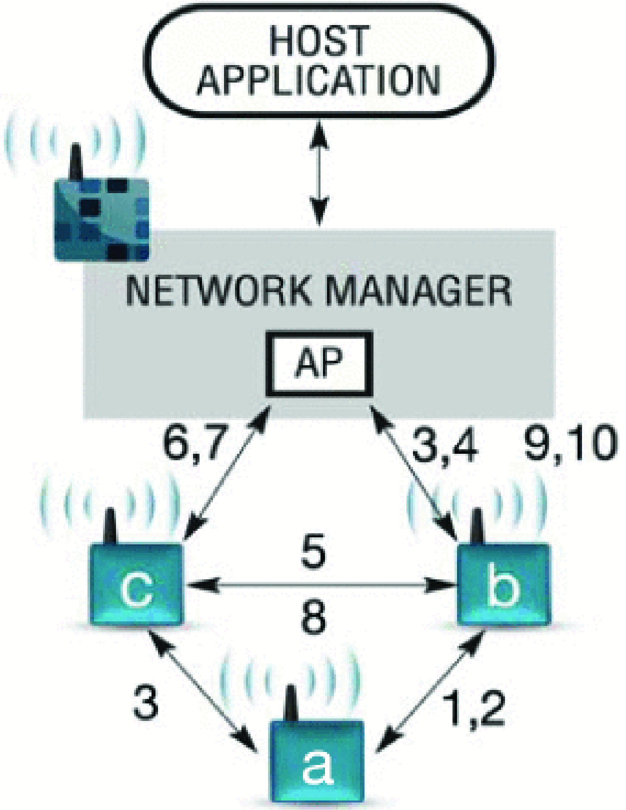


Figure 1: An example of scheduling in WirelessHART

Limitation 2: Global Dissemination

Exponential Growth

- Hyper-period
- Schedule length

Non-Adaptive

- Full re-dissemination
- High cost

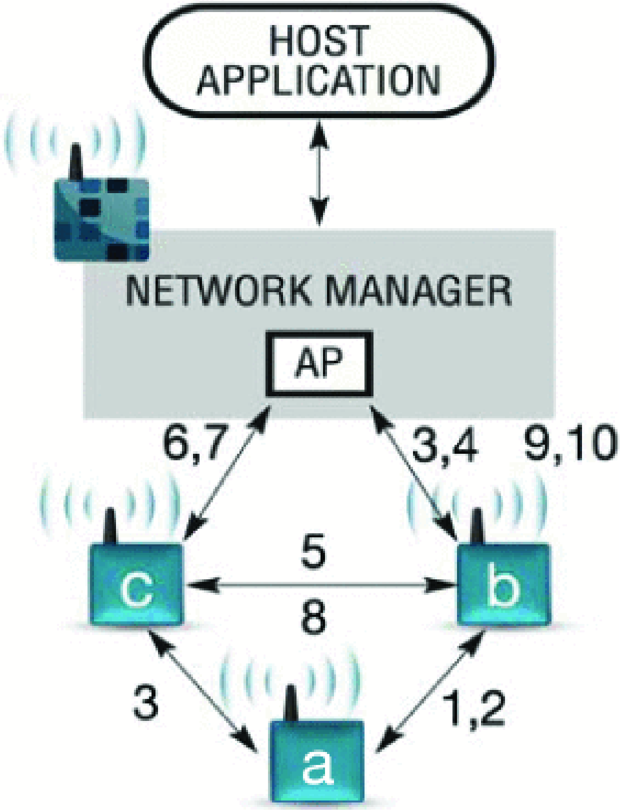


Figure 1: An example of scheduling in WirelessHART

Proposed Architecture

WirelessHART

- 802.15.4
- 16 channels
- TDMA MAC
- Multi-Hop
- Graph Routing

- Offline centralized scheduling

DistributedHART

- 802.15.4
- 16 channels
- TDMA MAC
- Multi-Hop
- Graph Routing

- Online decentralized scheduling

Outcomes of Paper

Addressed WirelessHART's limitations

– More suitable for Industry 4.0
– Higher network utilization
– 85% less energy intensive

Maintained Performance

– Highly scalable
– Improved adaptability
– Preserves reliability
– Questionable latency but improved long-term Real-Time performance
– Decreased Schedulability but competive

2

DistributedHART Design

Design of DistributedHART

Bulk of work done at node level

- Nodes schedule their transmissions locally online
- Each node performs channel/time allocations, and scheduling

Execution Order

1. Network Manager generates routes (global)
2. Channel Allocation (distributed, offline)
3. Time Window Allocation (distributed, online)
4. Real-Time Scheduling (local, online)

Channel Allocation

Signal-to-Noise plus Interference (SNIR)

- If exceeds threshold, then two nodes in conflict

Receiver conflict graph

- Each node maintains a physical interference model using SNIR
- Use vertex coloring to assign channels

Time Window Allocation

Transmitter conflict graph

- Graph over remaining conflicting nodes after channel allocation
- Two nodes have an edge if transmissions by both will lead to a collision at the recipients
- Use vertex coloring to assign time windows

Epoch

Epoch

- defines 1 full cycle of all assigned time windows, after which the sequence repeats.

W: time slots

γ: total number of unique windows

epoch = $\gamma \times w$

Goal: minimize γ → smaller epoch → lower end-to-end delay

Scheduling

Works on both fixed or dynamic policies

- Deadline Monotonic (DM)
- Earliest Deadline First (EDF)

Each node adopts policy locally

- Network manager does not assign priorities
- Each node knows priorities of packets at buffer
- Source nodes determine their sampling period/deadline

Nodes respond to aperiodic events locally **locally** without global dissemination

DistributedHART EDF Scheduling Example

Assumption: only one channel

5 Nodes

2 Flows

$F = \{F_1, F_2\}$

$T_1 = 6$

$T_2 = 12$

$D_1 < D_2$

Flow
Flow: F_i Period: T_i Deadline: D_i $D_i <= T_i$

DistributedHART EDF Scheduling Example

$F = \{F_1, F_2\}$
$T_1 = 6$
$T_2 = 12$

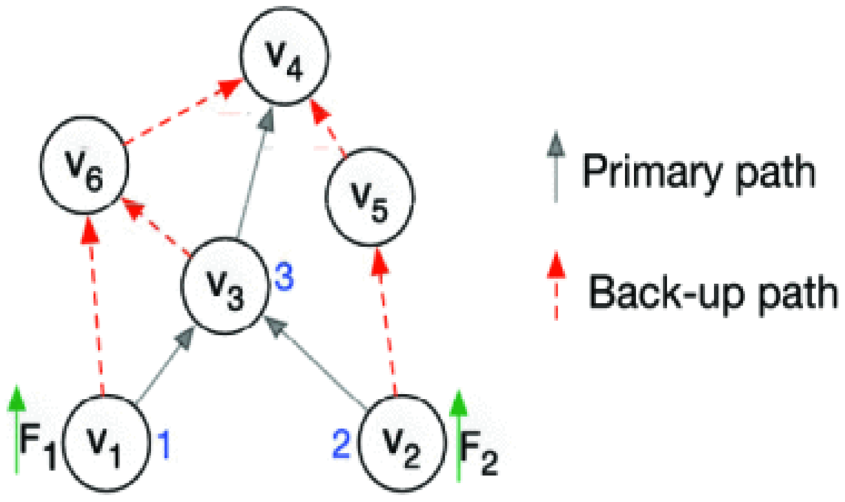


Figure 2a: Time window allocation example

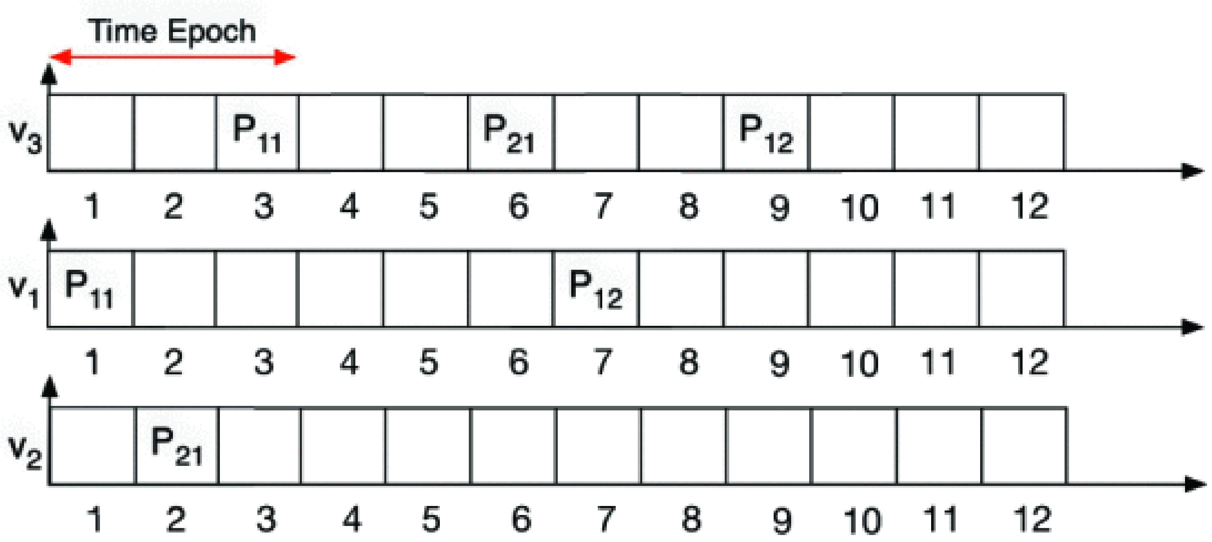


Figure 2b: local scheduling at nodes v1, v2, v3

DistributedHART EDF Scheduling Example

$F = \{F_1, F_2\}$
$T_1 = 6$
$T_2 = 12$

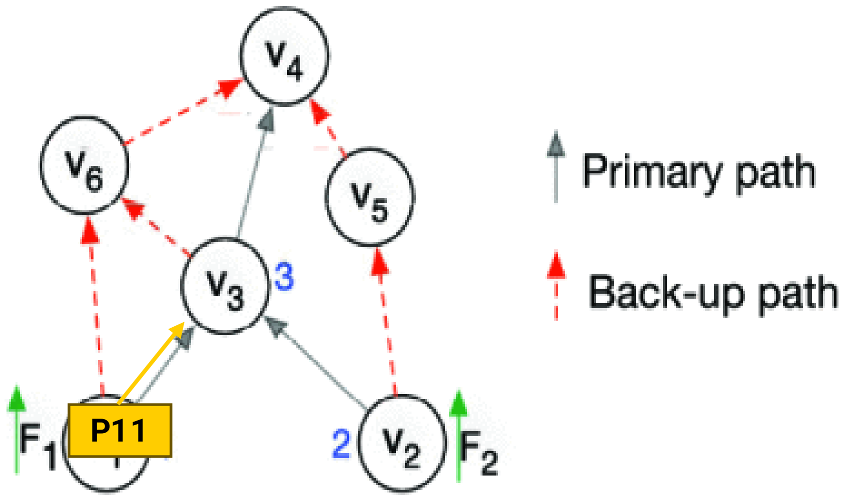


Figure 2a: Time window allocation example

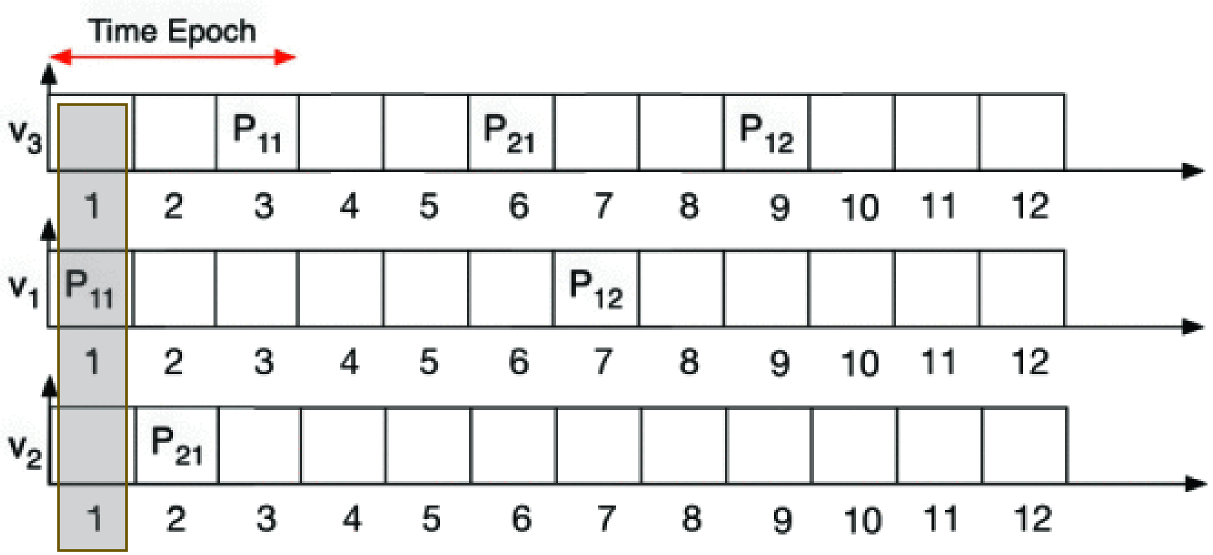


Figure 2b: local scheduling at nodes v1, v2, v3

DistributedHART EDF Scheduling Example

$F = \{F_1, F_2\}$
$T_1 = 6$
$T_2 = 12$

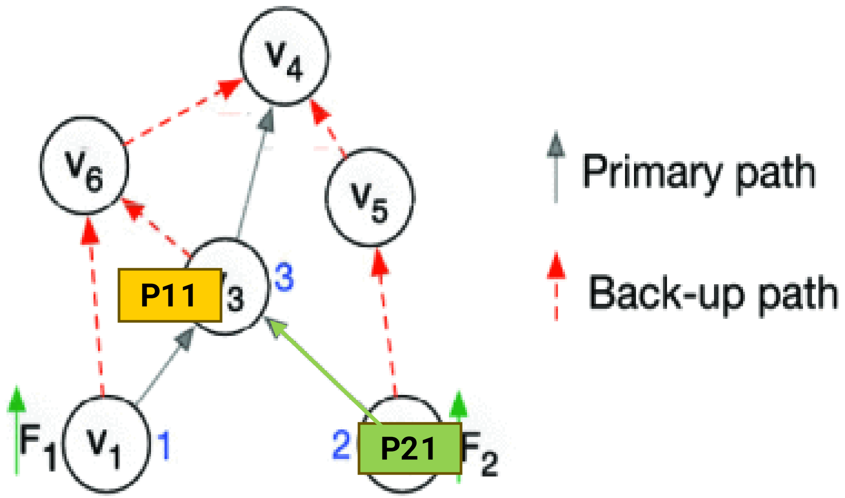


Figure 2a: Time window allocation example

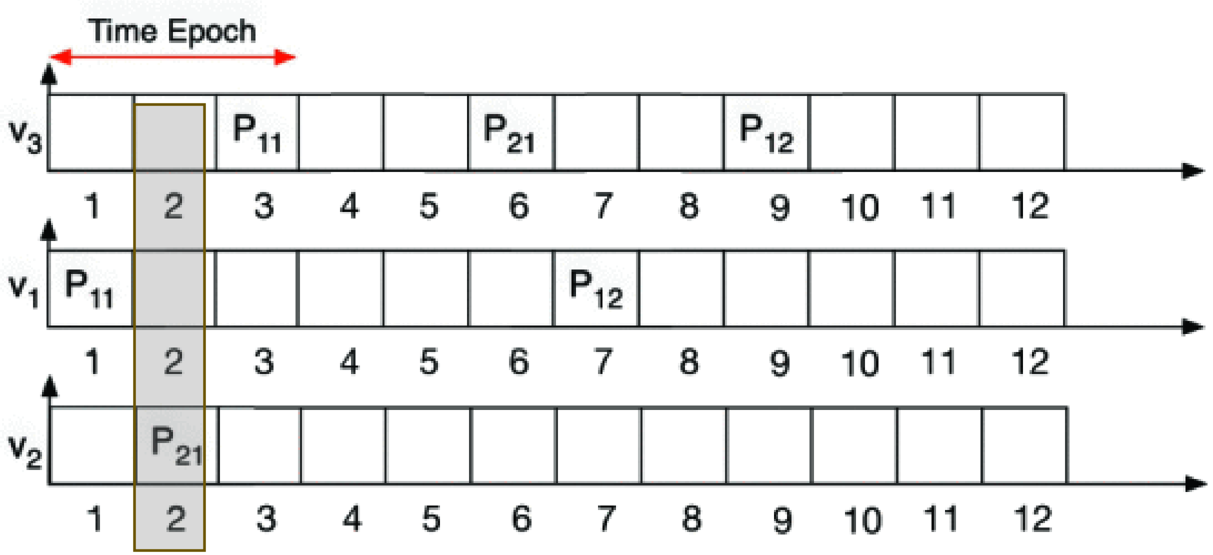


Figure 2b: local scheduling at nodes v1, v2, v3

DistributedHART EDF Scheduling Example

$F = \{F_1, F_2\}$
$T_1 = 6$
$T_2 = 12$

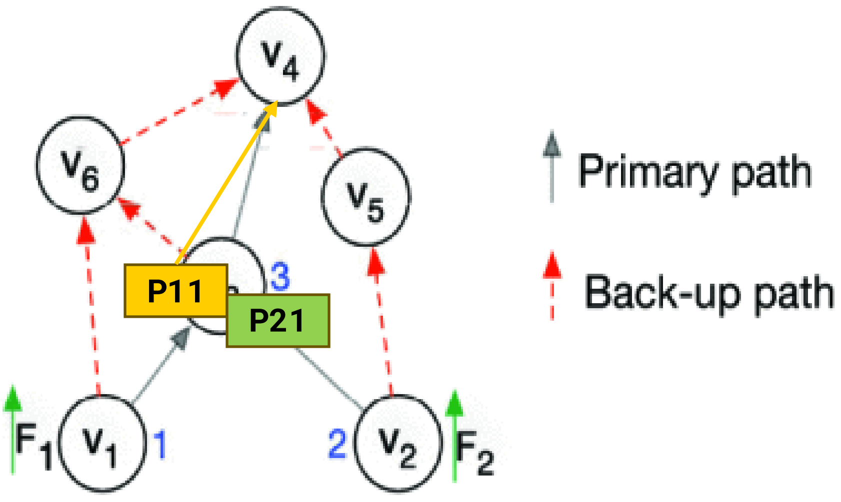


Figure 2a: Time window allocation example

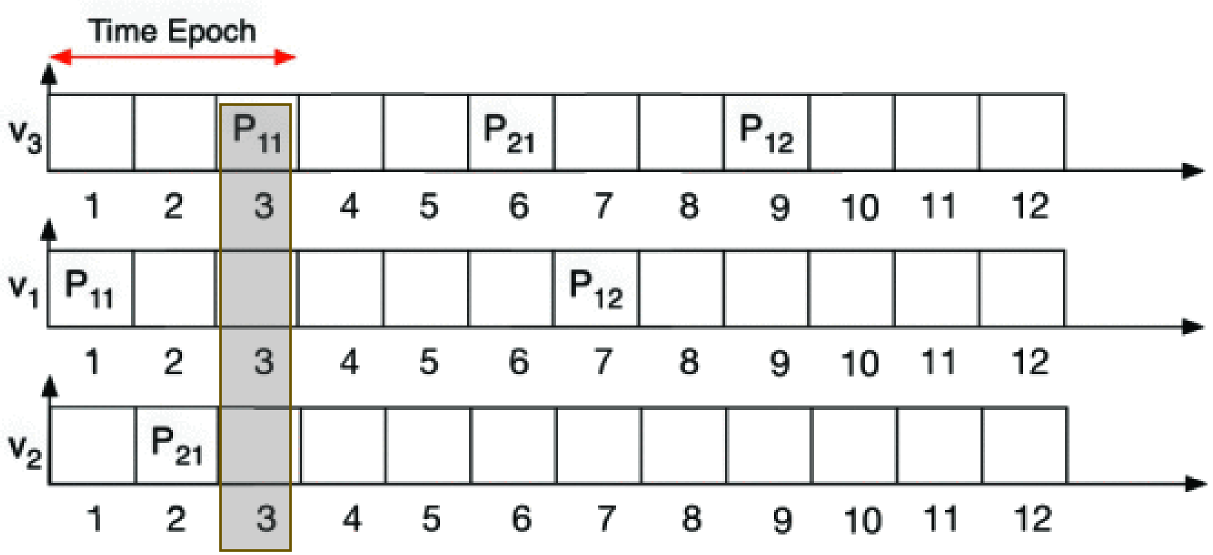


Figure 2b: local scheduling at nodes v1, v2, v3

DistributedHART EDF Scheduling Example

$F = \{F_1, F_2\}$
$T_1 = 6$
$T_2 = 12$

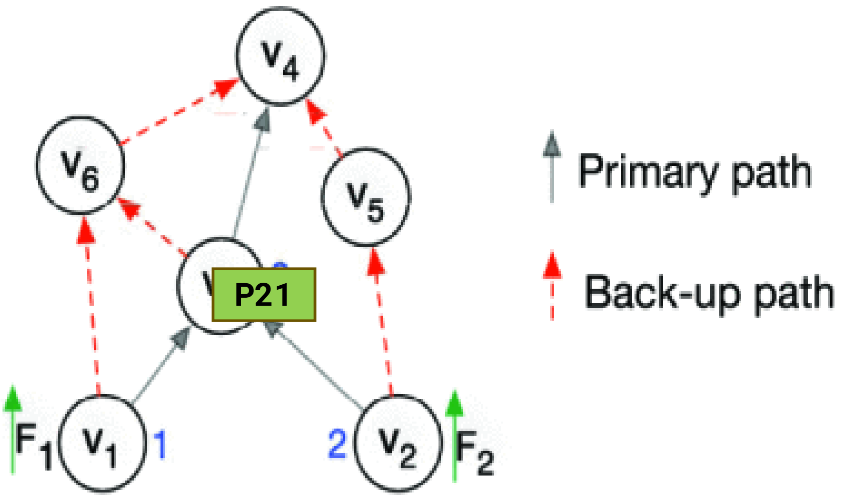


Figure 2a: Time window allocation example

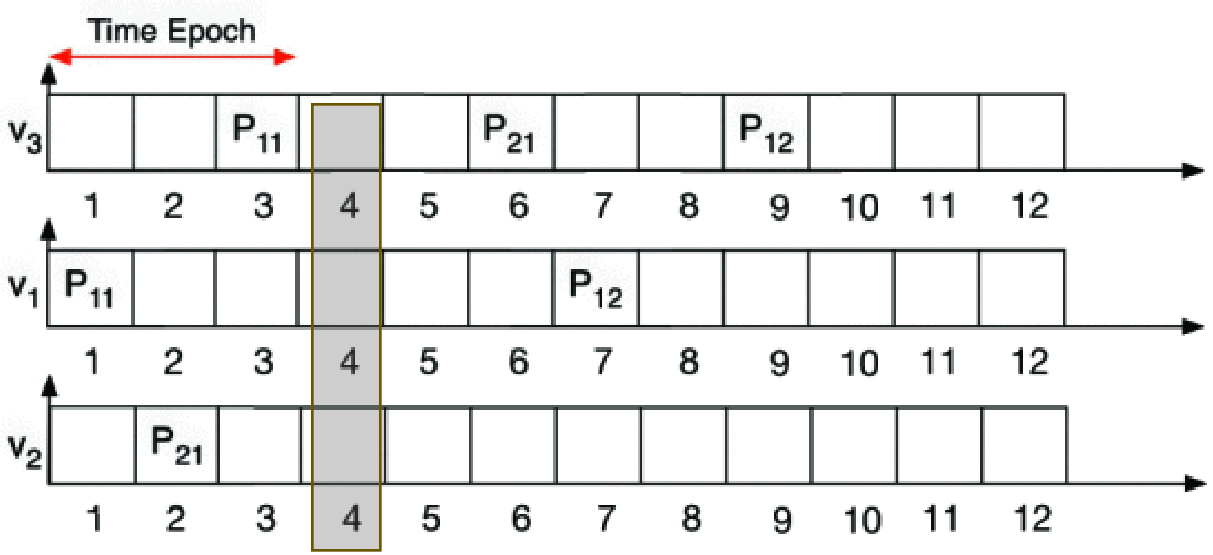


Figure 2b: local scheduling at nodes v1, v2, v3

DistributedHART EDF Scheduling Example

$F = \{F_1, F_2\}$
$T_1 = 6$
$T_2 = 12$

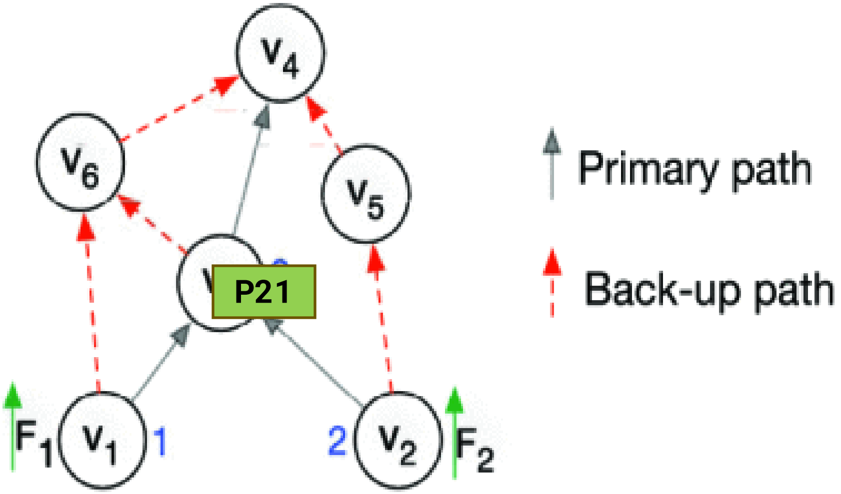


Figure 2a: Time window allocation example

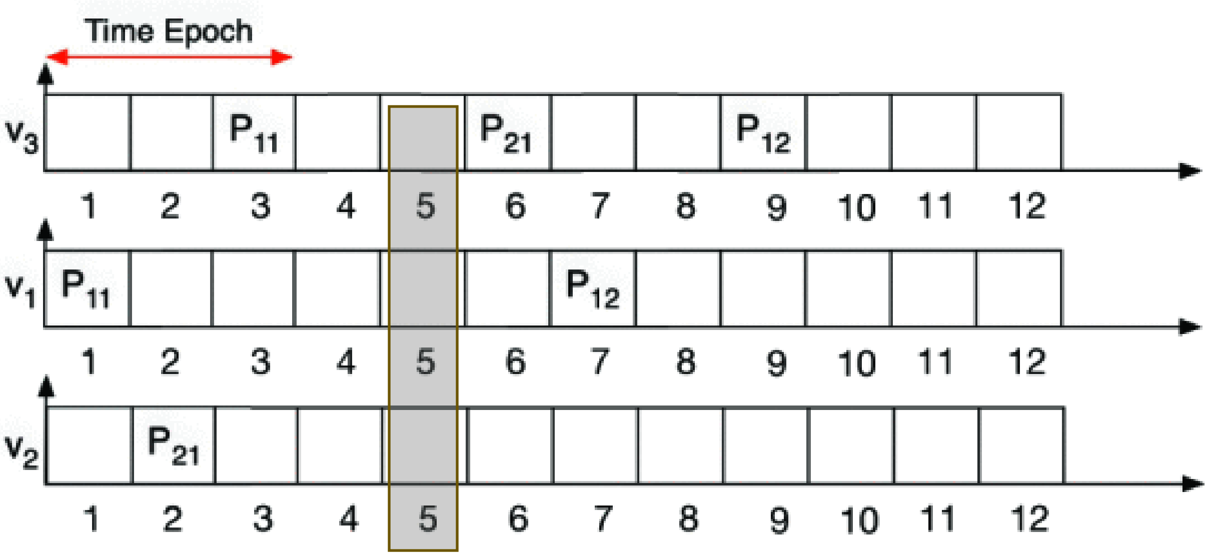


Figure 2b: local scheduling at nodes v1, v2, v3

DistributedHART EDF Scheduling Example

$F = \{F_1, F_2\}$
$T_1 = 6$
$T_2 = 12$

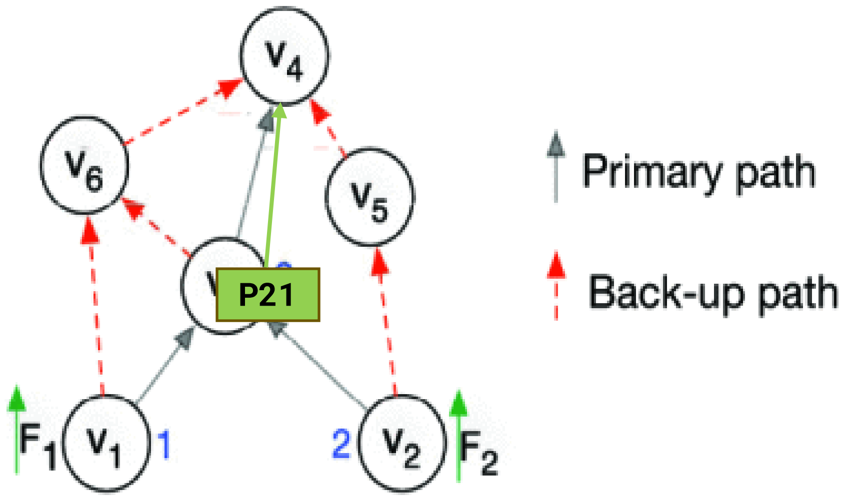


Figure 2a: Time window allocation example

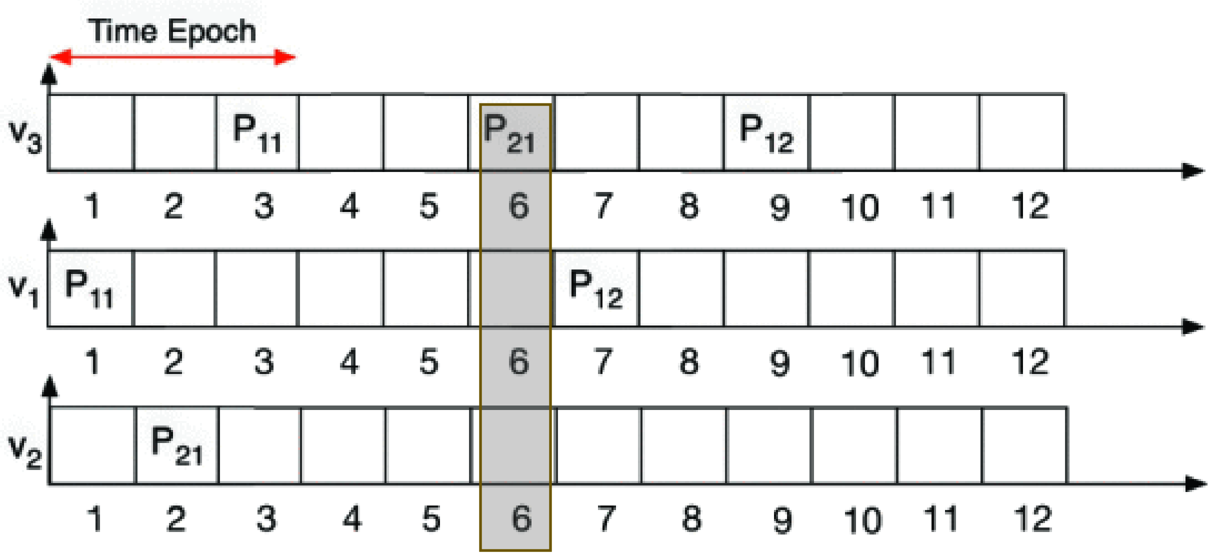


Figure 2b: local scheduling at nodes v1, v2, v3

Slot Sharing

Dedicated Slots

- time slots within a node's time window used immediately

Shared Slots

- time slots within or outside a node's time window, used opportunistically if idle.

If transmission failure is handled by nodes locally:

1. After failure, node waits for some Θ.
2. After Θ, If no transmissions sensed, take slot as "shared".
3. wait random back-off and transmit packet.

$\boldsymbol{\Theta}$: sensing delay

Slot Sharing Re-Transmission Example

$F = \{F_1, F_2\}$
$T_1 = 12$
$T_2 = 24$

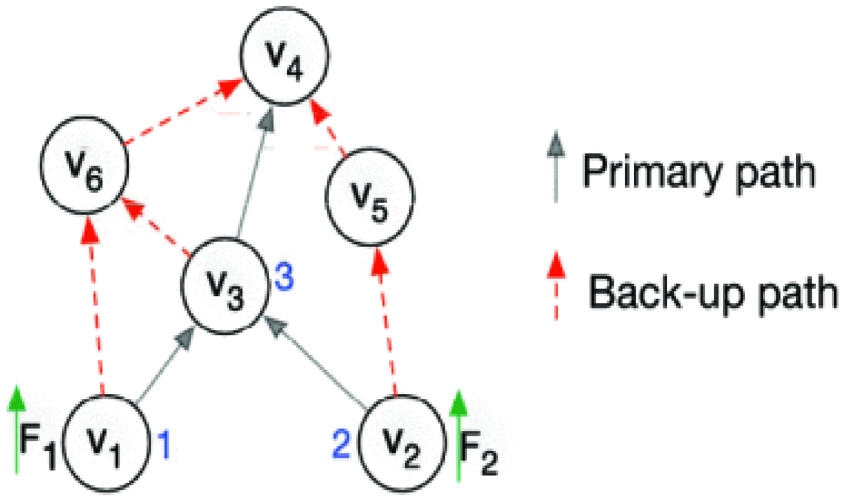


Figure 2a: Time window allocation example

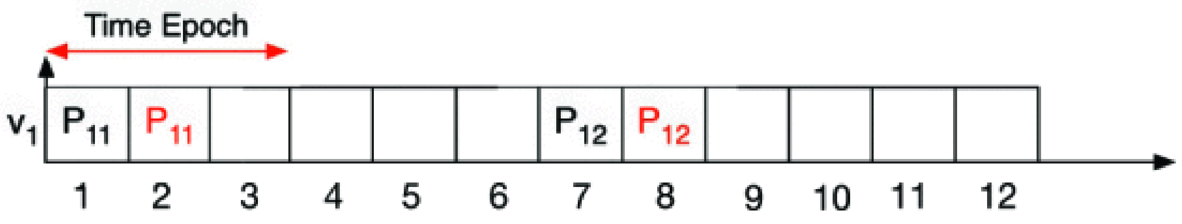


Figure 3: An example of scheduling as dedicated and shared slot

Hidden Terminals

Overlapping packet re-transmissions

- Two nodes cannot hear each other, and both transmit to a third node
- Failure in carrier sensing

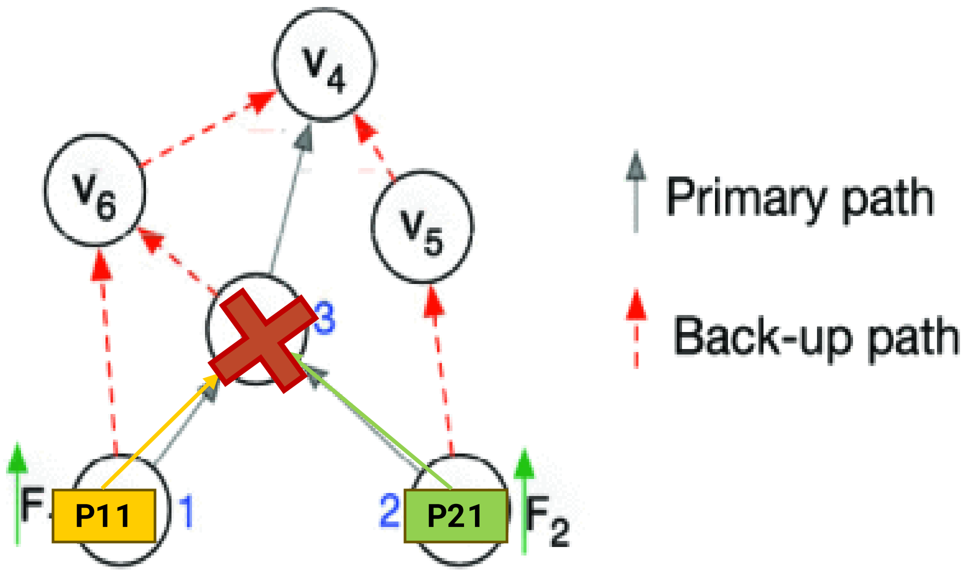


Figure 2a: Time window allocation example

Capture Effect

802.15.4 Radio Behavior

1. Detect and locks preamble with strongest Received Signal Strength (RSS)
2. Once locked, raise interrupt, and stop searching
3. Decode packet and ignore subsequent packets

Exploit

- If in dedicated slot, transmit at full power immediately
- If a re-transmission in shared slot, transmit at weaker power after Θ

Capture Effect Experiment

Goals

- Determine Θ before re-transmission
- Determine re-transmission Tx power
- Observe distance variation

Metric

- Packet Reception Rate (PPR)

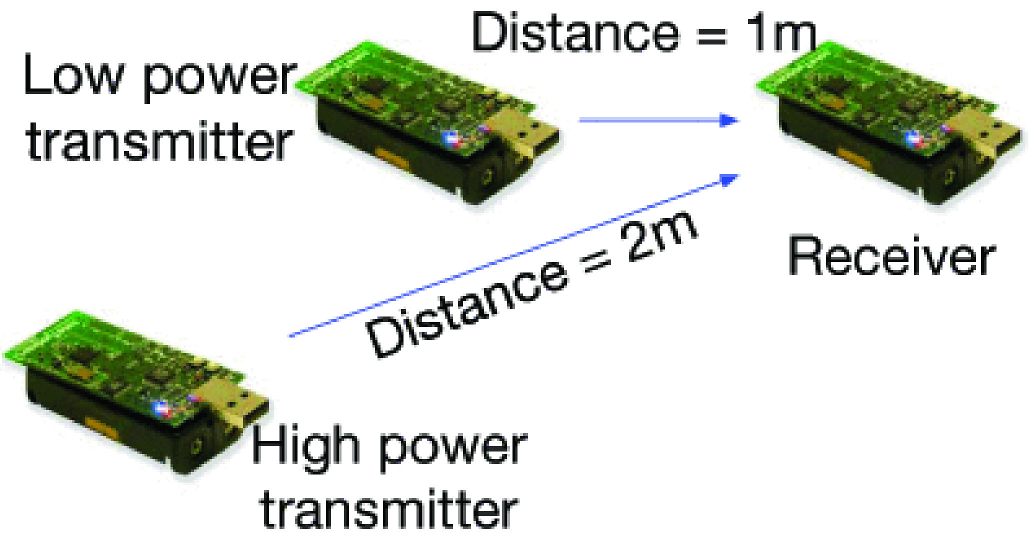


Figure 4: Capture effect experiment setup

Capture Effect Experiment

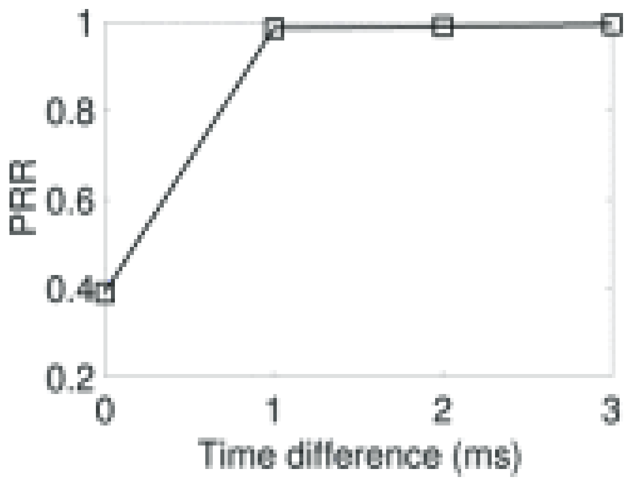


Figure 5a: Time difference

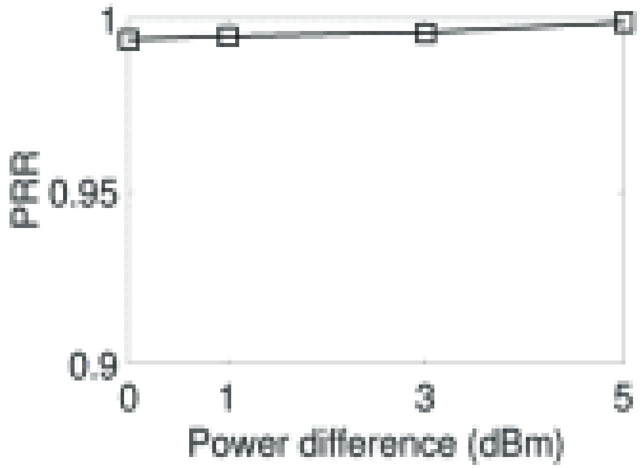


Figure 5b: Power difference

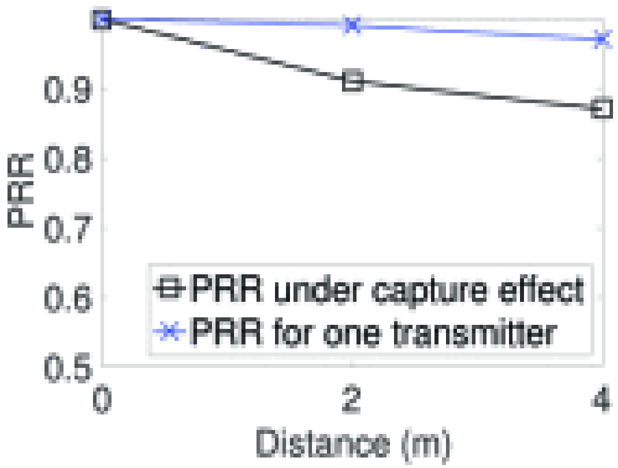


Figure 5c: Distance

5a Conditions
- Transmitters at 0dBm power
- Varied θ between transmissions

5b Conditions
- Fixed θ = 3 ms.
- One transmitter kept at 0dBm, other reduced.

5c Conditions
- Fixed θ = 3 ms
- Fixed 3 dB power difference.
- Varied distance between transmitters and receiver.

3

DistributedHART Analysis

Flow P(Success)

k: index representing a link

E_i : set of links making up primary route of flow F

p_k : probability successful transmission over link k

$$\mathbb{P}(F)_i = \prod_{k \in E_i} \left(1 - (1 - \rho_k)^2\right)$$

Equation 1: P(successful reception of packet through primary path of graph a route)

Flow Delay

$\mathbf{V_j}$ = set of nodes in primary path $\mathbf{F_j}$

$\mathbf{I_v (F_i) =}$ $\{ F_j \mid v \in V_j \}$ = set of flows passing through node v

$$\delta_v (F_i) = (\gamma - 1) \times w + 2 + \sum_{F_j \in I_v(F_i)} \max \left\{ 0,\ 1 + \left\lfloor \frac{\delta_v (F_i) - D_j}{T_j} \right\rfloor \right\} 2 \times \gamma$$

Equation 2: total delay experience by a flow F_i at node v

Flow Response Time

Flow Probability: $P(F_i)$

Flow Delay: $\delta\upsilon$

$$R_i = \sum_{\nu \in V_v} \delta_v (F_i)$$

Equation 3: response time experienced by a control loop DistributedHART

4

Comparison to WirelessHART

Testbed Setup

Setup

- **Platform:** TinyOS 2.2 running TelosB motes
- **Network Size:** 130 nodes
- **Transmit Power: -**28.7 dBm
- **Topology:** 4-hop
- **Routing Baseline:** Centralized graph routing
- **Manager Role:** Computed initial channel and time-window allocations centrally
- **Scheduling Policies:** EDF and DM (with spatial reuse)

Metrics

1. **Energy:** energy consumed per node (J)
2. **Memory:** memory consumed to store a schedule (KB)
3. **Convergence Time:** average time taken for all nodes to obtain a schedule (s)
4. **Schedulability Ratio:** fraction of test cases schedulable among all cases

Experiment Results

Figure 6: Experimental result under varying number of flows considering harmonic periods

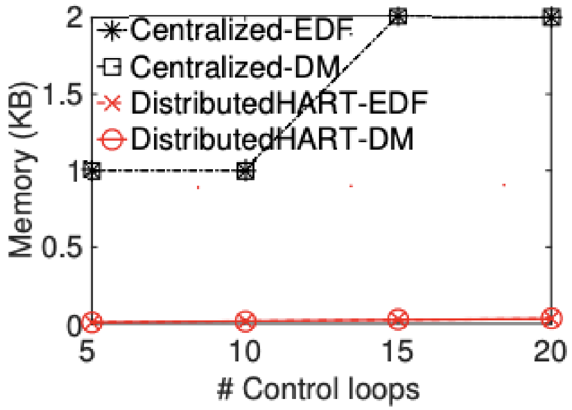


Figure 6a: Memory Consumption

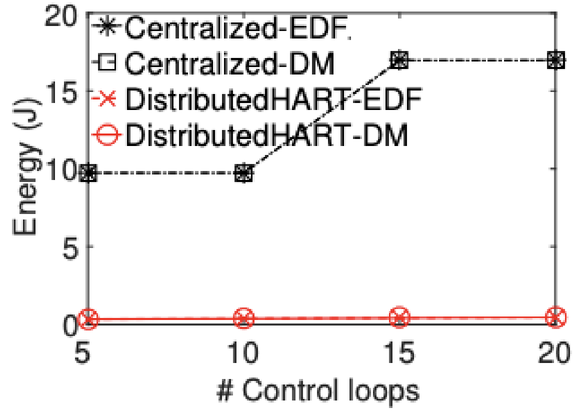


Figure 6b: Energy Consumption

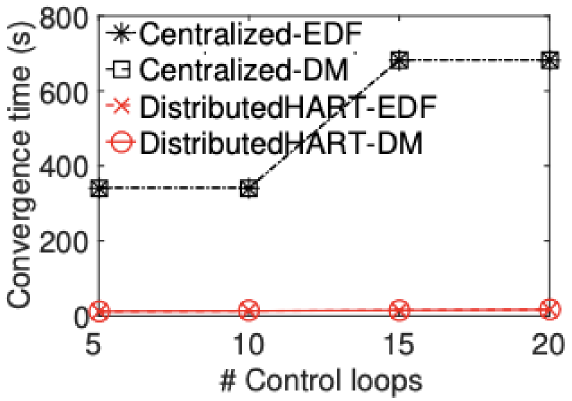


Figure 6c: Convergence Time

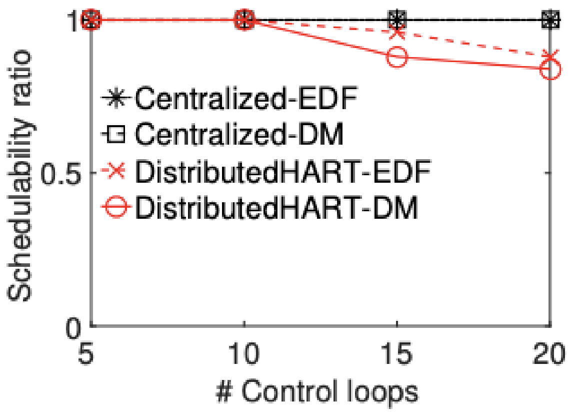


Figure 6d: Schedulability Ratio

Simulation Setup

Setup

- **Simulator:** TOSSIM (TinyOS Simulator)
- **Network Size:** 148 nodes
- **Transmit Power: -**28.7 dBm
- **Fully-Distributed:** channel and time window allocated by nodes (both initially and operationally)
- **Test-Cases:**
    - 50 randomly selected sensor/actuators
    - Sensor/Actuators assigned random harmonic periods in the range of 2^{11-13} time slots
    - Every 10 flows, double the range
- **Scheduling Policies:** WirelessHART/DistributedHART EDF and DM, Orchestra, DiGS (

Metrics

1. **Energy:** energy consumed per node (J)
2. **Memory:** memory consumed to store a schedule
3. **Convergence Time:** average time taken for all nodes to obtain a schedule
4. **Schedulability Ratio:** fraction of test cases schedulable among all cases

Simulation Results

Figure 7: Performance under varying number of control loops considering harmonic periods

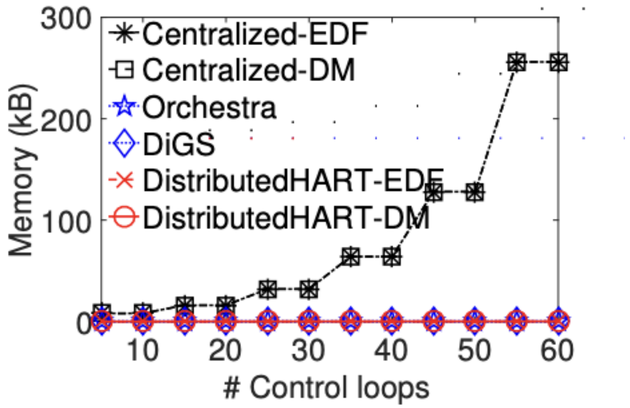


Figure 7a: Memory Consumption

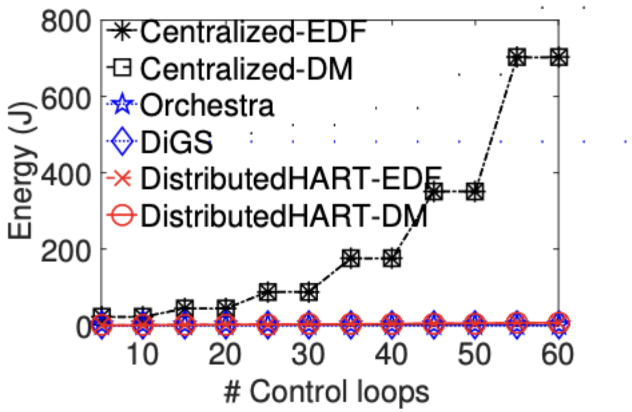


Figure 7b: Energy Consumption

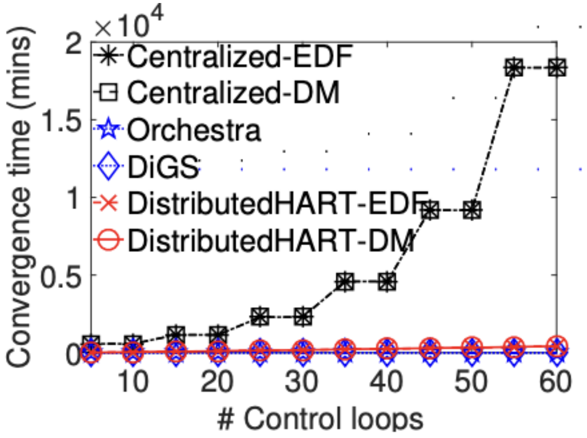


Figure 7c: Convergence Time

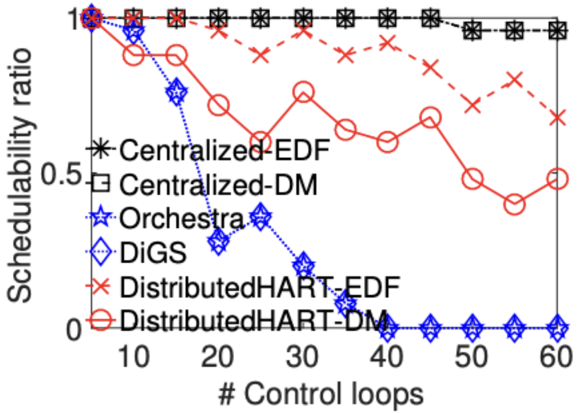


Figure 7d: Schedulability Ratio

Simulation Results

Figure 8: Performance under varying number of nodes

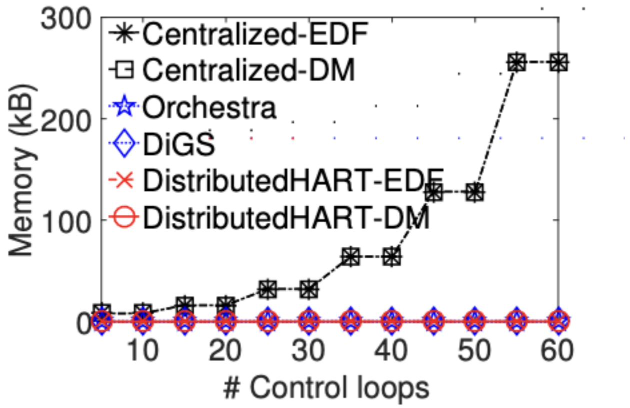


Figure 8a: Memory Consumption

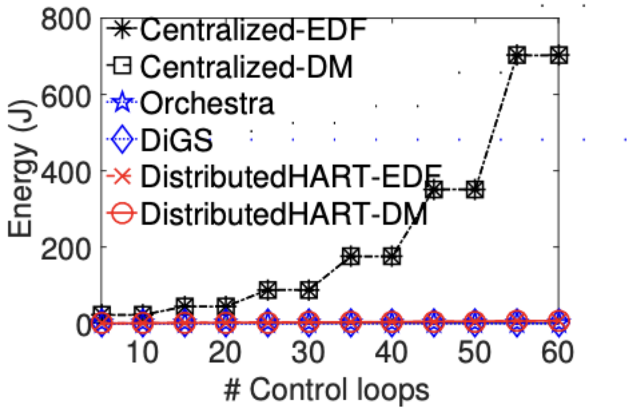


Figure 8b: Energy Consumption

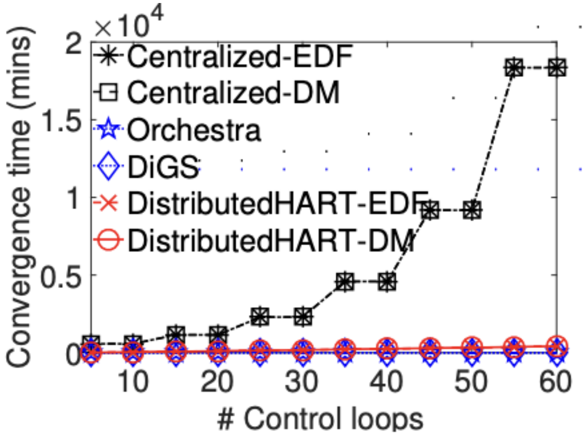


Figure 8c: Convergence Time

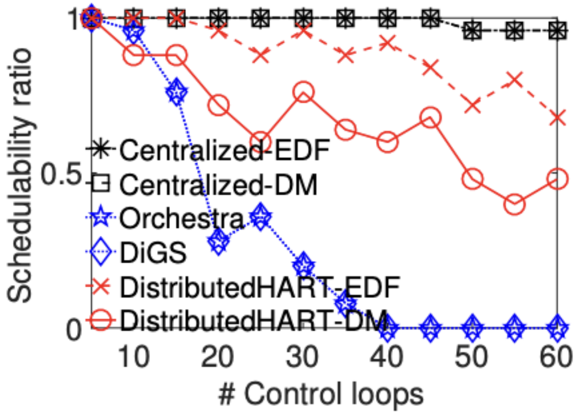


Figure 8d: Schedulability Ratio

Figure 9: Performance under varying workload dynamic

Simulation Results

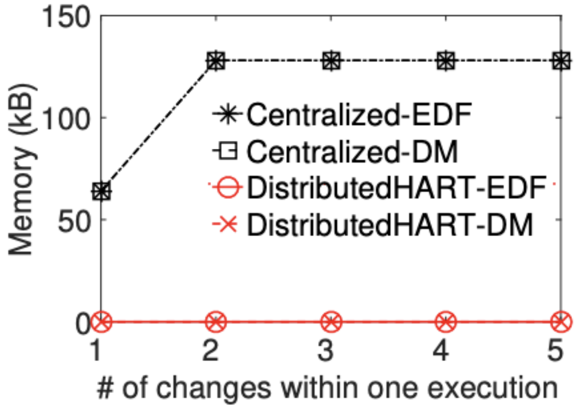


Figure 7a: Memory Consumption

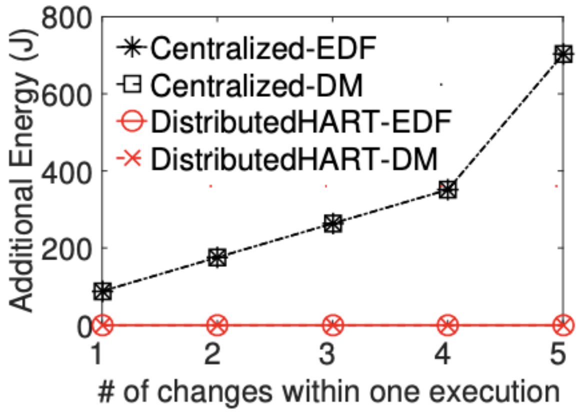


Figure 7b: Energy Consumption

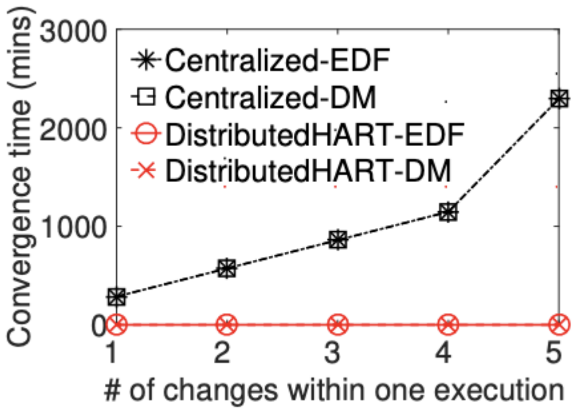


Figure 7c: Convergence Time

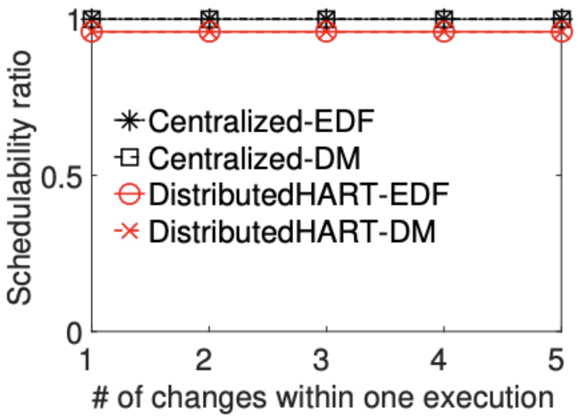


Figure 7d: Schedulability Ratio

Simulation Results

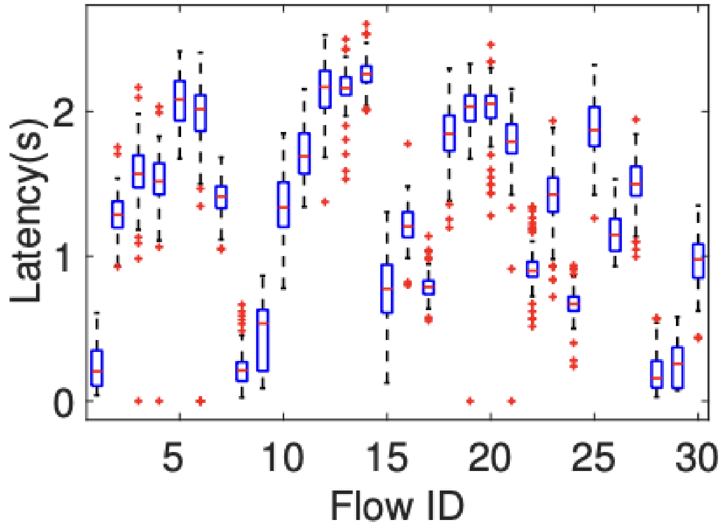


Figure 10: Latency under DistributedHART

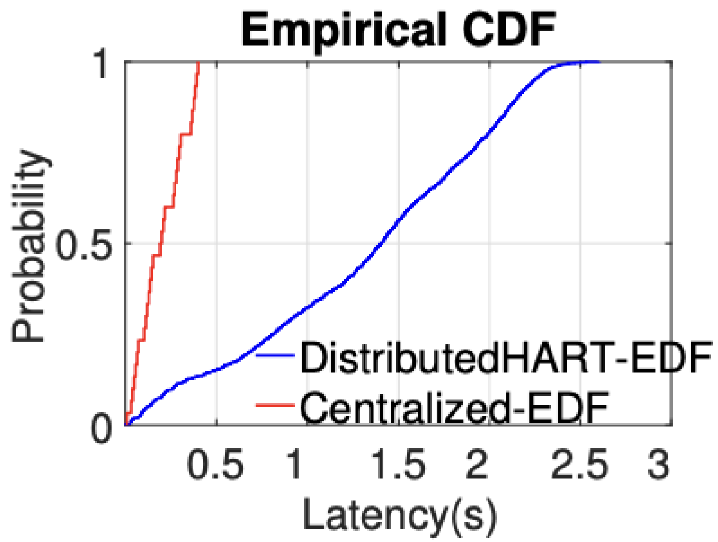


Figure 11: Latency comparison between Centralized-EDF and DistributedHART-EDF

Conclusion

Compelling argument for decentralized scheduling

- More suitable for Industry 4.0
- Higher network utilization
- 85% less energy intensive
- Highly scalable
- Improved adaptability
- Preserves reliability
- Questionable latency but improved long-term Real-Time performance

IOWA

Works Cited

This presentation was part of the University of Iowa's CS:4980:0007 Course

Presentation not affiliated with Wayne State University, Missouri University of Science and Technology, or the authors of "DistrbutedHART: A Distributed Real-Time Scheduling System for WirelessHART Networks".

- Paper: https://ieeexplore.ieee.org/document/8743267
- DOI: 10.1109/RTAS.2019.00026
- Figures [1,2,3,4,5,6,7, 8, 9, 10, 11]

Other Images:

- Industry 4.0 graph: https://www.texspacetoday.com/step-towards-sustainability-from-industry-4-0-to-industry-5-0/
- WirelessHART: https://www.emerson.com/en-us/automation/measurement-instrumentation/industrial-wireless-technology/wireless-gateways