App-Aware Scheduling on Networked Systems

DATE 2020
March 10th

Kacper Wardega
Boston University

About us

Dependable Computing Lab
Boston University

Kacper Wardega
ktw@bu.edu

Wenchao Li
wenchao@bu.edu

Industrial Wireless

- Cheaper to install
- Easier to maintain
- Flexible
- Boosts productivity

(WN, 2017)

Problem statement

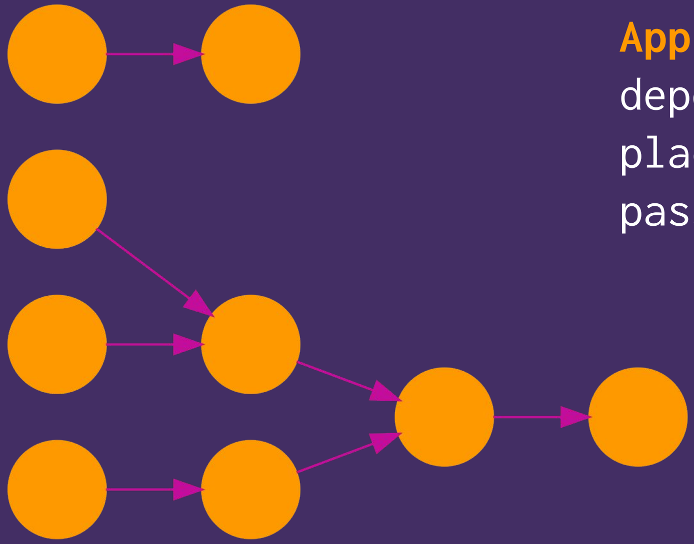
System designers aim to ensure the real-time properties of applications running over wireless networked systems in the face of communication uncertainties without sacrificing performance.

Scheduling problem

Take an application consisting of interdependent tasks and compute:

- Start/end times for each task,
- Assign messages to wireless communication rounds, and
- Determine the communication parameters for each message.

DAG scheduling on wireless systems

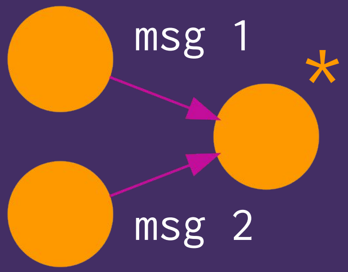


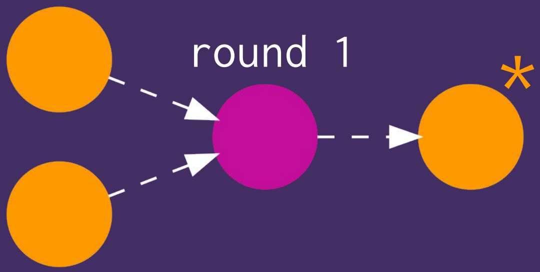
Applications consists of **tasks** that depend on one another. Task placement is known, but message passing is **unreliable**.

Scheduling objectives

- Communications & tasks occur in the correct order.
- Tasks meet respective deadlines.
- Each communication round lasts long enough for messages to propagate across the network (flooding).
- Tasks meet real-time guarantees.
- Minimize the makespan.

Scheduling objectives: real-time guarantees





***Soft**: Task * should succeed 80% of the time*

***Weakly-hard**: Task * should fail no more than 5 out of every 6 consecutive executions*

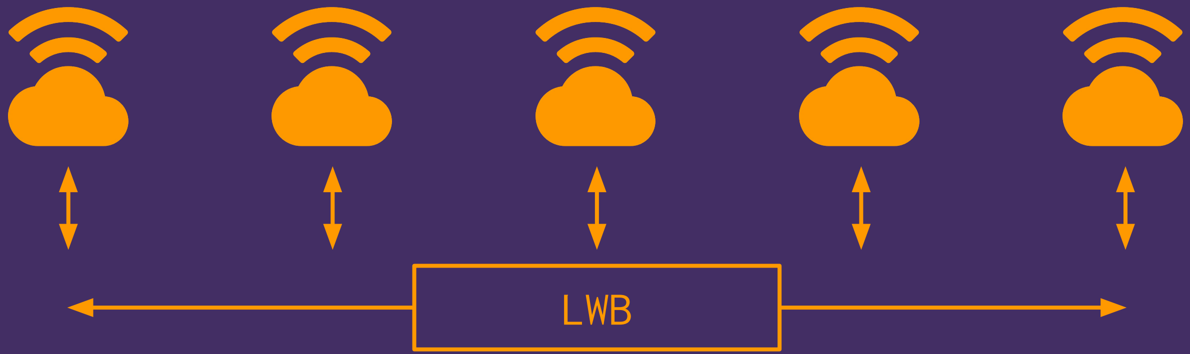
(TC, 2001)

Known quantities

The application is known and fixed, i.e. known task durations, message widths, task dependencies.

The network statistics are known, i.e. under given communication parameters, the scheduler knows the real-time behavior of the message-passing.

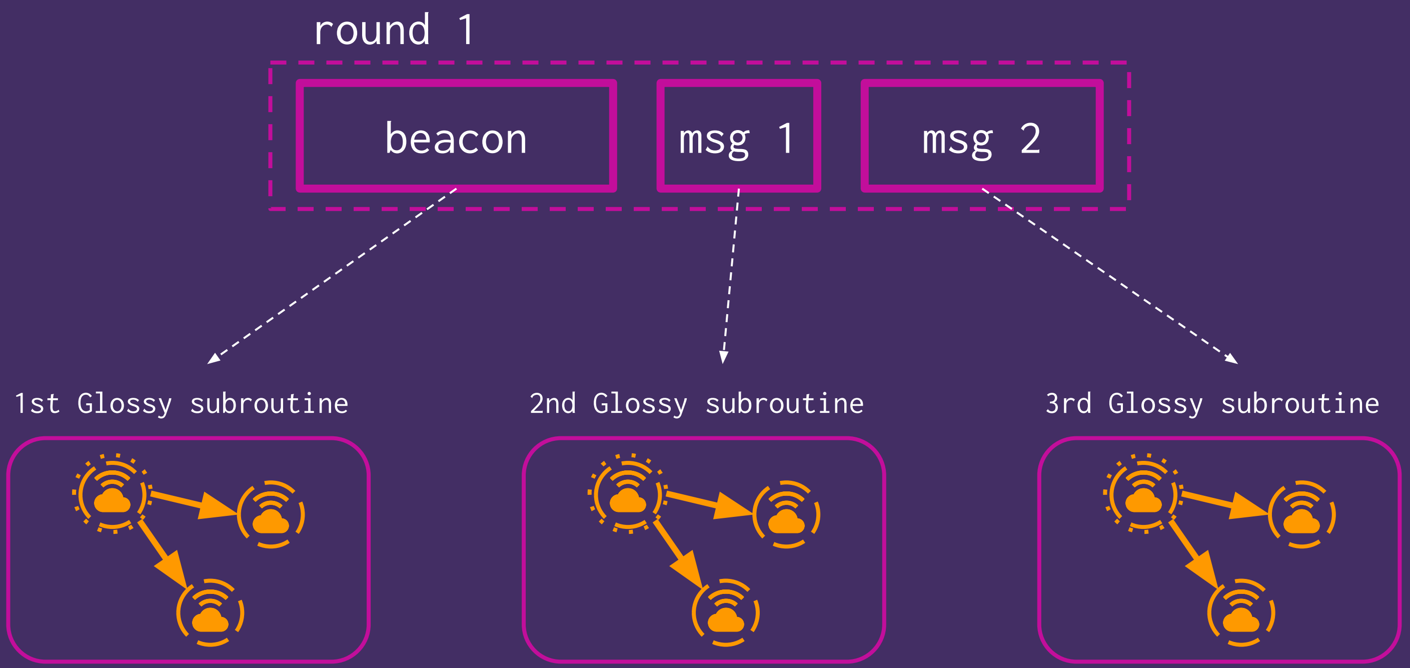
Enter: the low-power wireless bus



The LWB abstracts away the radio so that it's as if each node were wired to every other node.

(SenSys, 2012)

The LWB consists of Glossy floods

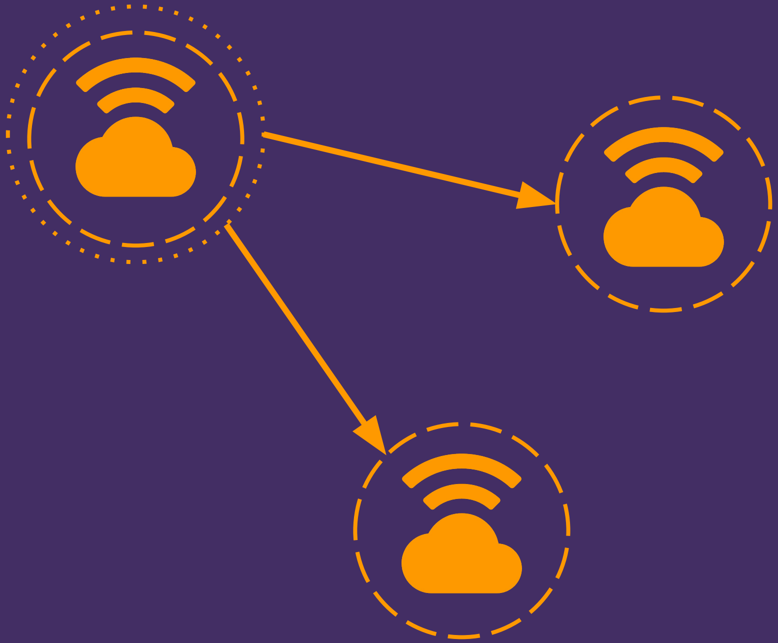


(IPSN, 2011)

Glossy floods are the backbone of the LWB

STATUS: compute

Glossy floods are the backbone of the LWB

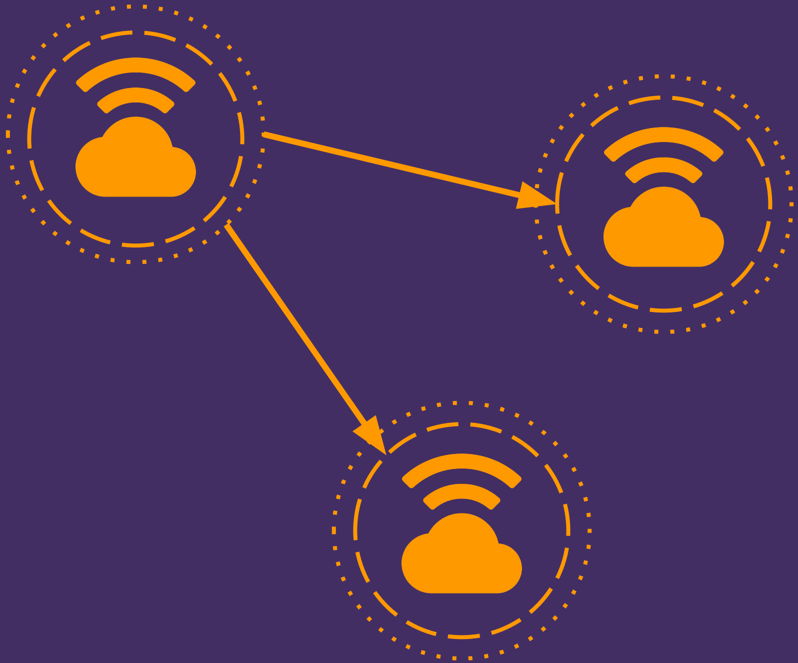


STATUS: Glossy

Glossy floods are the backbone of the LWB

STATUS: Glossy

Glossy floods are the backbone of the LWB



STATUS: Glossy

Glossy floods are the backbone of the LWB

STATUS: Glossy

Glossy floods are the backbone of the LWB

STATUS: complete

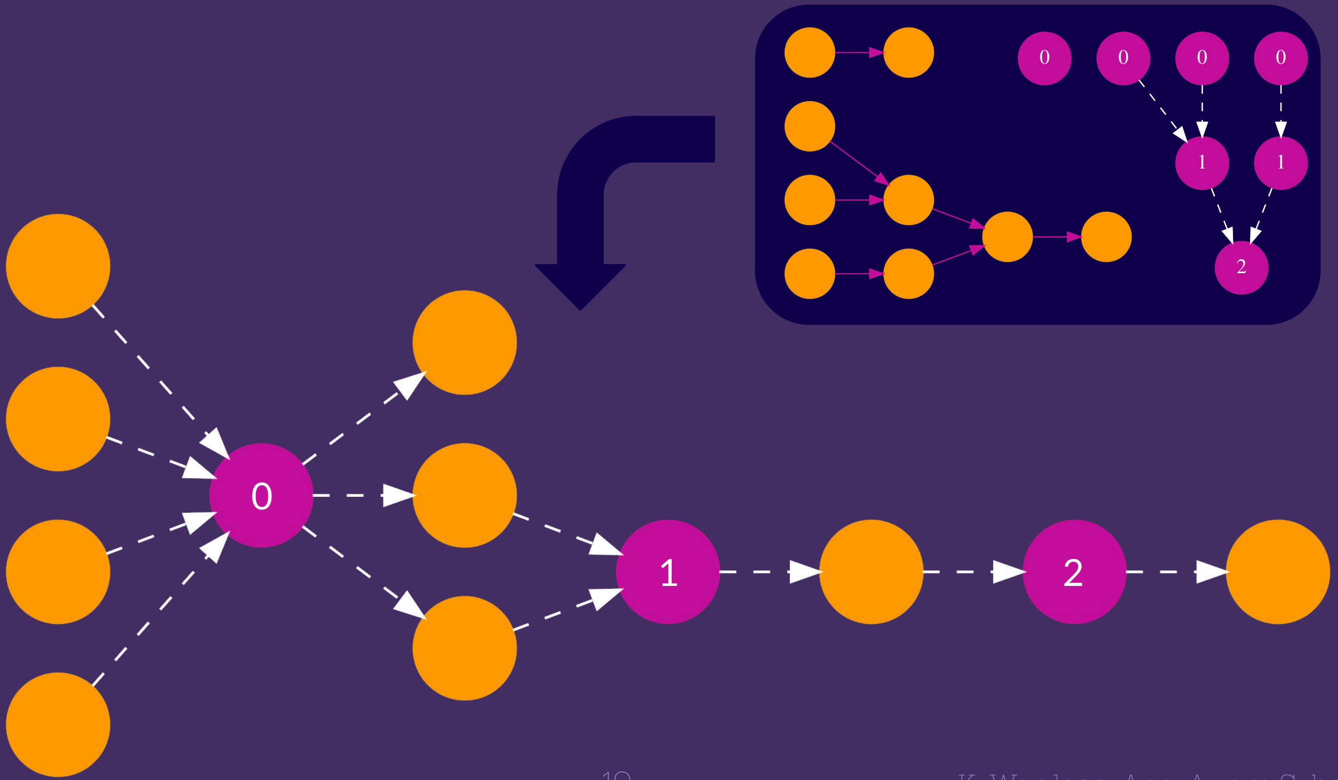
Flood successful

LWB & glossy

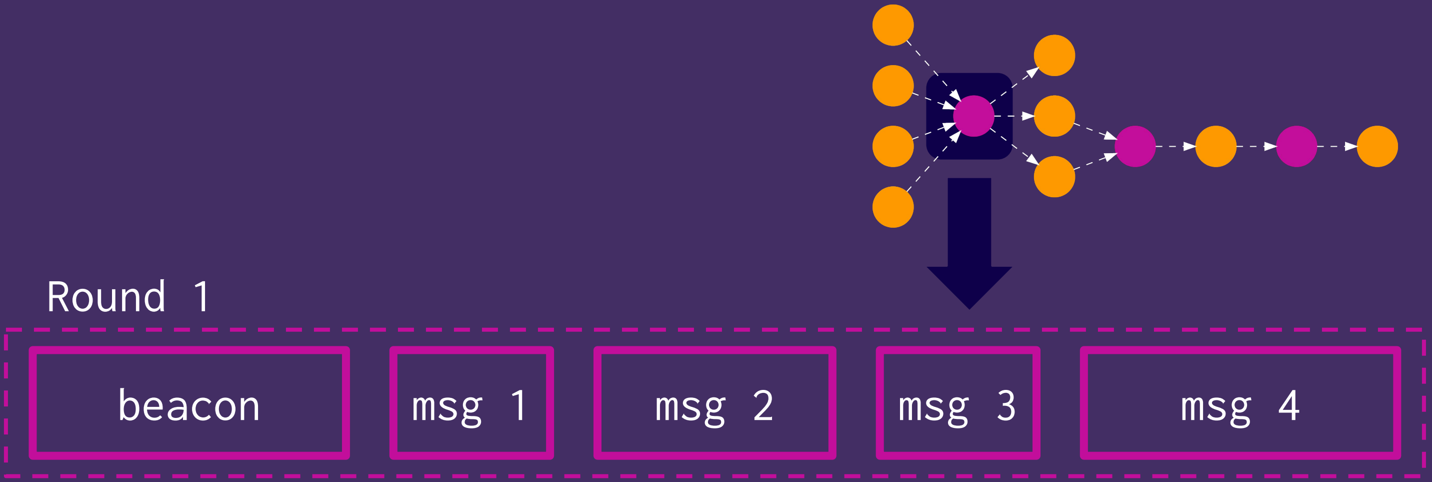
1. Glossy is event-triggered, but the LWB is time-triggered.
2. There is a fundamental tradeoff between reliability and time/energy controlled by the retransmission parameter.
3. Wireless control has been demonstrated over the LWB.

(ICCPS, 2019)

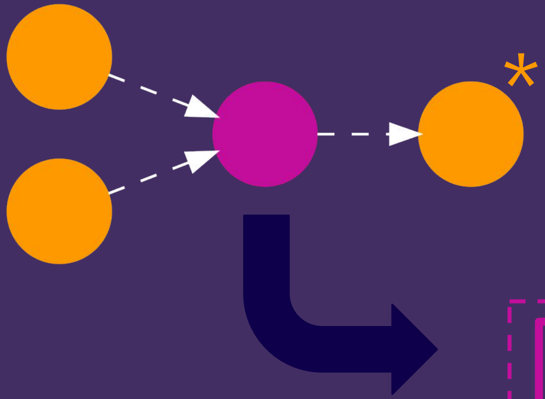
Communication-adjusted task graph



Communication round dissected



Rounds consist of several glossy floods. Flood duration depends on message width and the retransmission parameter.



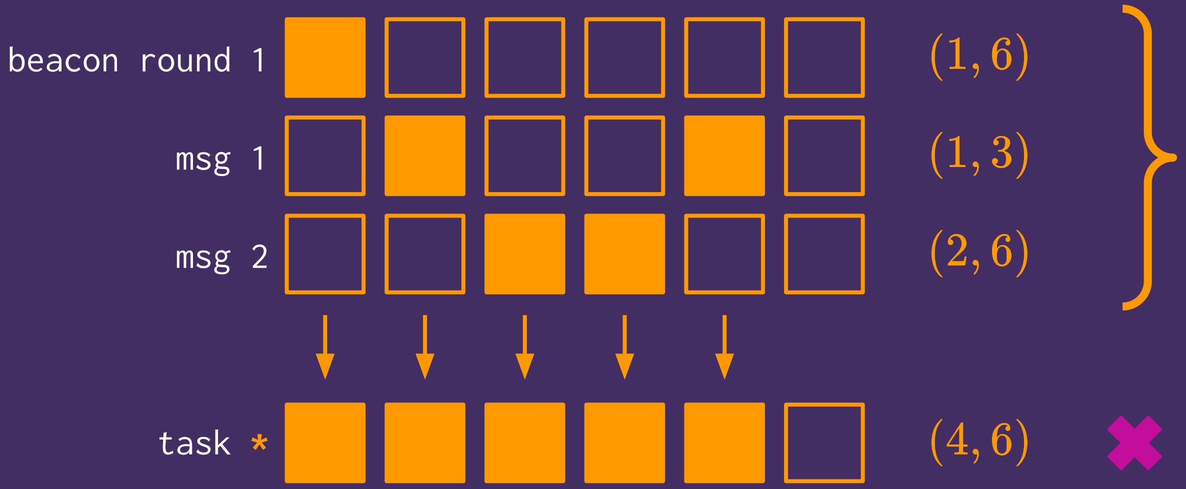
*Task * should succeed 80% of the time* = *The product of the success rates of the Glossy floods carrying the beacon, msg 1 & 2 must be at least 80%*

Round 1

beacon	msg 1	msg 2

e.g. 95% x 90% x 94% = 80.37%

Optimal soft real-time schedules are obtained via **MILP** or **SMT**. But what about weakly-hard real-time?



Allowing any failure pattern allowed by these constraints…

Will the task depending on those messages always obey this constraint?

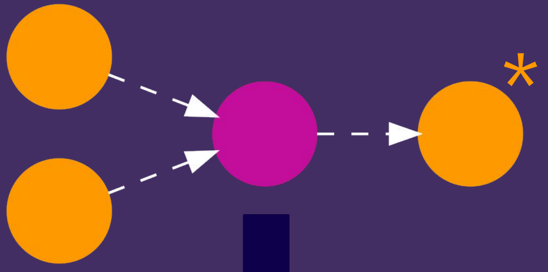
Communication failure patterns for preceding messages may violate the task’s (m,K).

Abstraction for layered weakly-hard constraints

- Checking satisfaction of w-h real-time constraints requires universal quantifiers.
- We prove a min-plus abstraction for layered w-h constraints.

$$(\alpha, \gamma) \oplus (\beta, \delta) = (\min\{\alpha + \beta, \gamma, \delta\}, \min\{\gamma, \delta\})$$

To compose two w-h constraints we leverage that in the worst case, as many misses as possible occur within the smaller window



*Task * should fail no more than 5 out of every 6 consecutive executions*

=

The min-plus sum of the failure characteristic of the glossy floods carrying the beacon, msg 1 & 2 must be at least (5,6)

Round 1

beacon	msg 1	msg 2

e.g. $(1,8) \oplus (2,9) \oplus (1,7) = (4,7) \leq (5,6)$

Using the min-plus abstraction, we can encode the problem to **SMT** to obtain optimal weakly-hard real-time schedules.

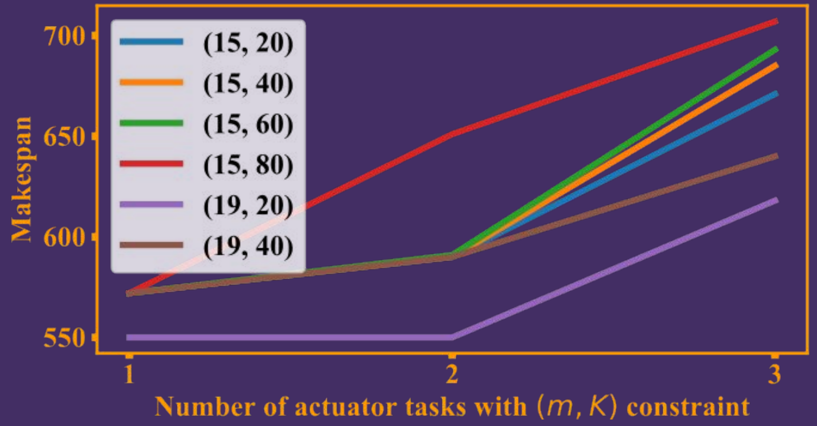
Validation & experiments

We validate scheduler correctness on synthetic and industry-related applications. Furthermore, we show how a real-time scheduler enables design automation…

Applications

MIMO/switched control

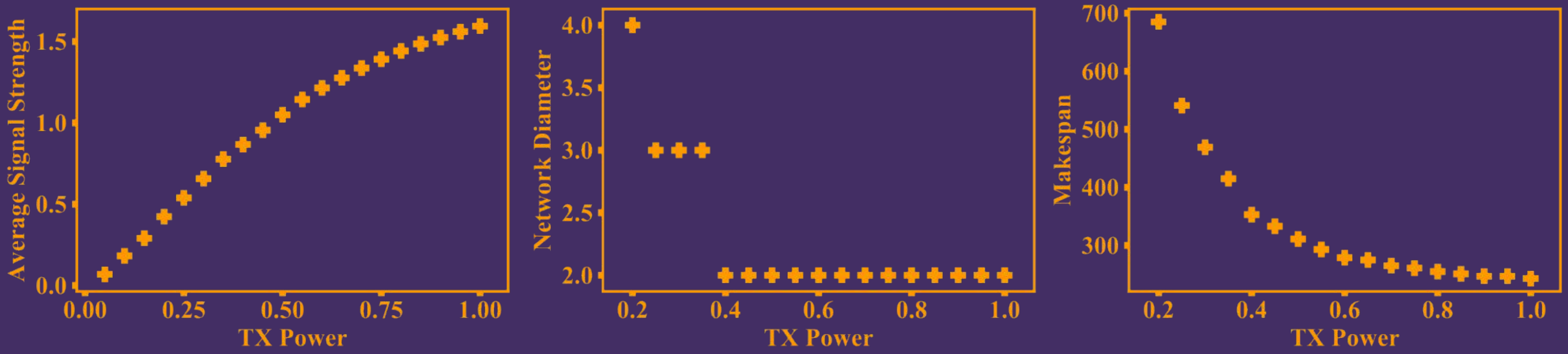
- Multiple sensors as inputs to controllers for multiple actuators
- Designer specifies worst-case bounded failures permitted



Applications

Design space exploration

- Mobile robots in a closed environment
- Designer specifies the application success rate and aims to minimize power usage



https://github.com/netdag/netdag

Our scheduler implementation is open-source

References

X. Li, D. Li, J. Wan et. al. “A review of industrial wireless networks in the context of Industry 4.0”. Wireless Networks 23, pp. 23-41, 2017.

G. Bernat, A. Burns, and S. Member. “Weakly hard real-time systems”. IEEE Transactions on Computers, vol. 50, no. 4, pp. 308-321, 2001.

F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. “Efficient network flooding and time synchronization with glossy”. Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), pp. 73-84, 2011.

F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. “Low-power wireless bus”. Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems (SenSys), p. 1, 2012.

F. Mager, et al. “Feedback control goes wireless: guaranteed stability over low-power multi-hop networks”. Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS), 2019.

Thanks!