

# Energy management in Wireless Sensor Networks: Applications, issues and opportunities

Kameshwar Poolla

University of California, Berkeley

*July 08, 2010*

*Symposium on Industrial Embedded Systems*

*Trento*

# This Talk

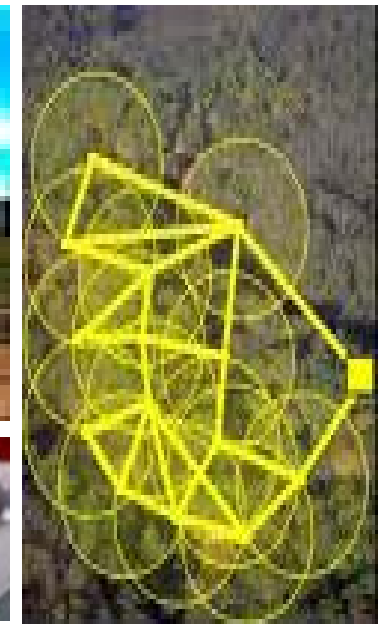
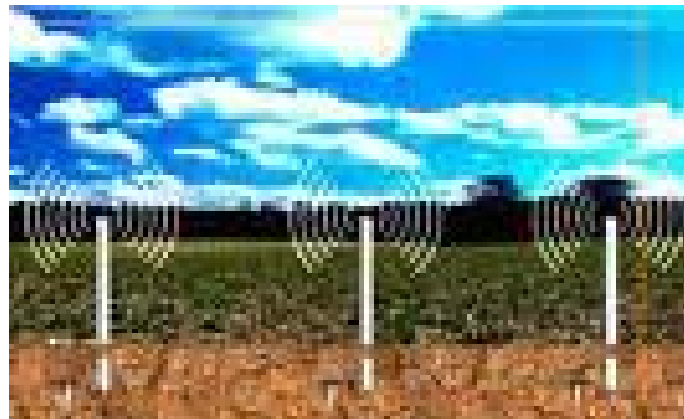
---

- I. When will we *really* see WSN industrial applications?  
[Provocative opinions]
- I. Our Embedded Systems Applications
  - Devices for Assisted Living
  - Precision Agriculture
  - Bacteria as Embedded Systems
- II. System level Energy management in WSN

# Industrial Applications of WSN

---

- Hasn't **really** happened yet
- Many companies – Dust Networks, CrossBow, ArchRock, Ember, Moteiv, MeshNetics, Invensys, ...
- Many deployments
  - perimeter security
  - building monitoring
  - networked control



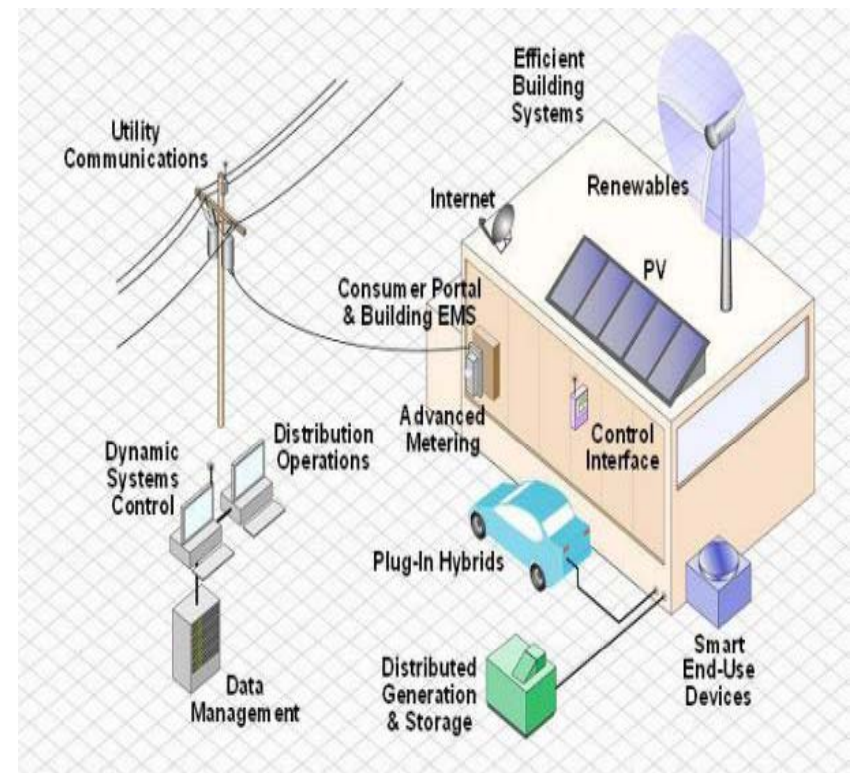
# But No Home Run !!

---

- US \$ 5.3 Billion market by 2010 predicted
- < \$50 Million realized in 2009
- **Why no home run?**
  - Cost            will be addressed when consumer market materializes
  - Standards    being addressed now
  - Ownership    this limits the market in many applications

# The Driving Application for WSN?

- Home appliances for demand response
  - Refrigerators, air-conditioners, lighting
  - Turning on/off appliances using price signals
  - Smart charging of PHEVs
  - Smart Grid comes home !!
- It is a **consumer** market
- Very innovative selling strategies
- Aggregator subsidies



# The DALi Project

---

- [Consumer] Devices for Assisted Living

[Work in progress]

- Luigi Palopoli + Roberto Passerone

- Inspiration:

Enormously successful, sophisticated embedded systems

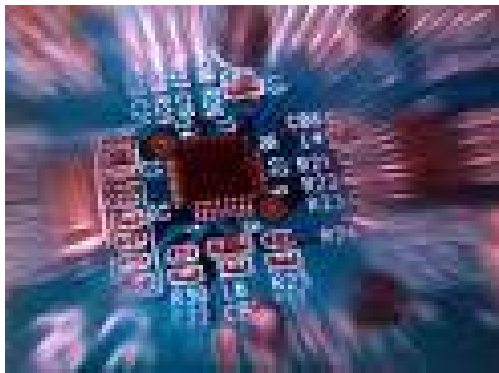
Integrated sensing, haptic interfaces, etc



# Objectives

---

- New generation of consumer [ $< 1000$  Euro] devices
- **Assist** elderly, deaf, impaired
- For decision making in **uncertain human environments**
- **Smart Walkers, Headphones, Navigators**



# Problems

---

- Understanding user needs
- Sensing the environment inexpensively
- Representing the environment to the user  
acoustic, haptic, visual  
what are language constructs
- Kinematic Modeling of the environment [obstacles, pedestrians]
- Universal design abstractions

# Precision Agriculture

---

- Lew Feldman + Andy Packard
- Greenhouses
  - Inexpensive
  - Large scale
  - Fully Instrumented
  - Distributed actuation
- China has 5 million acres!



# Problems

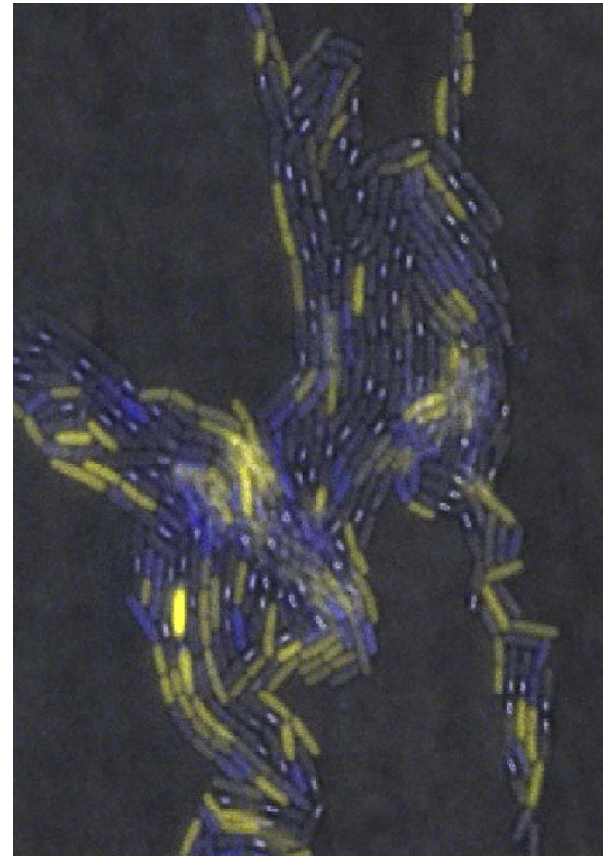
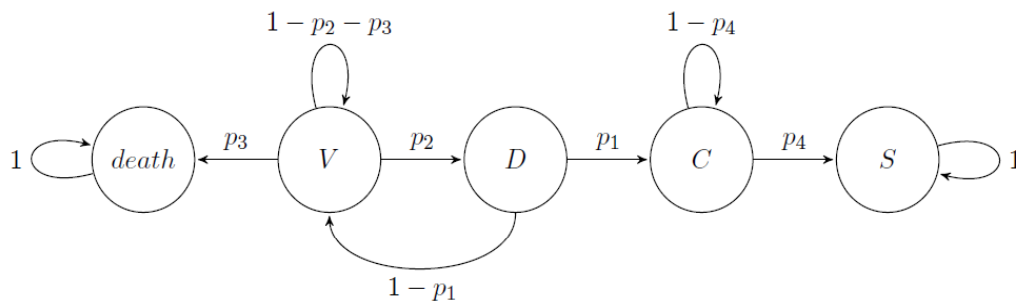
---

- Need to sense **plant stress state**
- Idea: use a guinea-pig plant as a sensor
  - genetically modified *Arabidopsis*
  - green-fluorescent-protein constructs
  - leaves fluoresce with different signatures
  - can detect salinity, thermal, water stress
- Possible good WSN application!
- But not a home run!

# Bacteria as Embedded Systems

- Adam Arkin + Mike Morimoto
- *Bacillus subtilis* is a widely studied bacterium
- States

$V$  vegetatively growing  
 $D$  deciding to sporulate or divide  
 $C$  committed to sporulation  
 $S$  mature spore



# Fundamental Questions

---

- How do bacteria encode conditional logic?

If                    [enough food & temp is good]

Then                [grow and divide]

Else                [sporulate]

End

- We believe it is a **Phospor-relay** mechanism ...
- Are bacteria optimists or pessimists?  
from evolutionary objectives, we believe they are **pessimists** ...

# Energy Aware Sensor Scheduling in Wireless Sensor Networks

Eilyan Bitar, Enrique Baeyens, Kameshwar Poolla

University of California, Berkeley

July 7, 2010

# Outline

Introduction and Problem Motivation

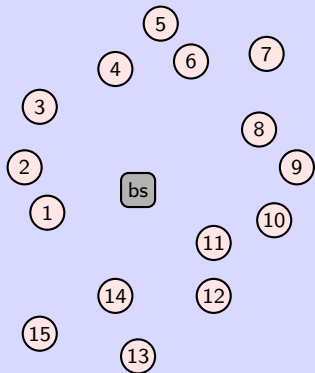
Results I

Results II

Conclusions

Future Work

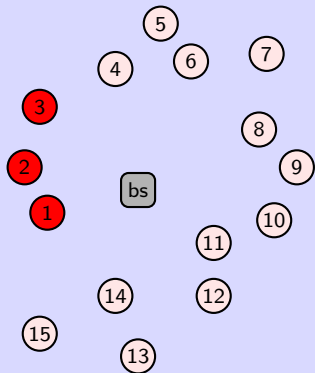
# Motivation



## What is the job of a sensor network?

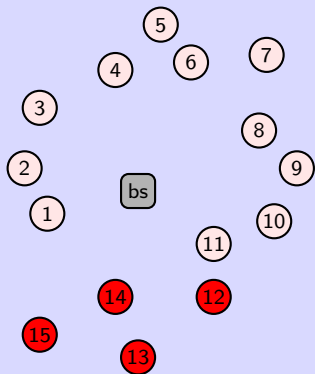
To gather a *sufficient* amount of information about the immediate environment to enable execution of tasks or decisions at base-station.

# Motivation



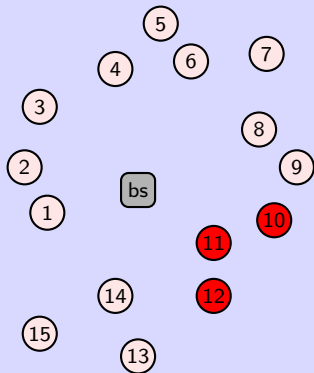
**Sufficient amount of information:**  
measurements from certain *subsets* of  
sensors

# Motivation



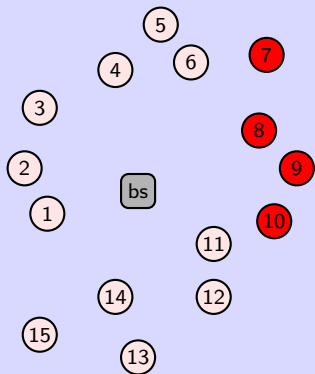
A **sufficient amount of information** is commonly comprised of measurements belonging to a subset of the sensors in the networks.

# Motivation



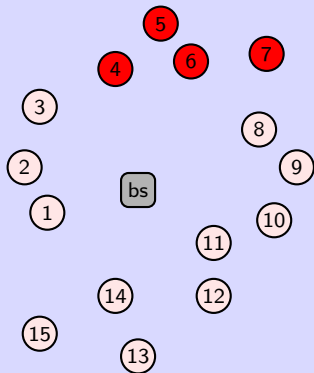
A **sufficient amount of information** is commonly comprised of measurements belonging to a subset of the sensors in the networks.

# Motivation



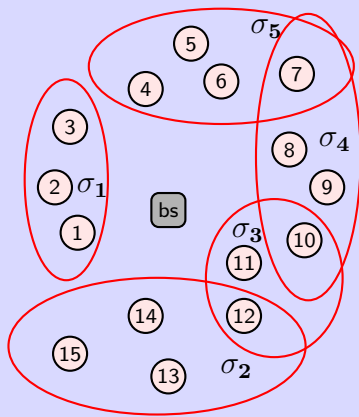
A **sufficient amount of information** is commonly comprised of measurements belonging to a subset of the sensors in the networks.

# Motivation



A **sufficient amount of information** is commonly comprised of measurements belonging to a subset of the sensors in the networks.

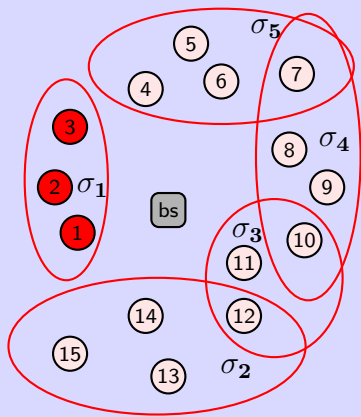
# Motivation



**Redundancy** built into the network implies these *sufficient* sensor sets are not unique.

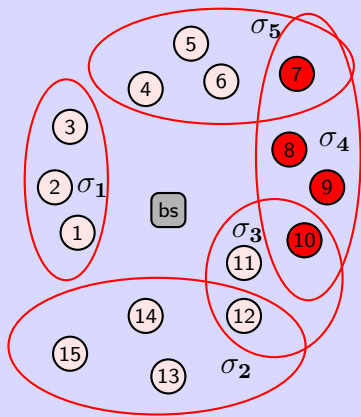
So base-station has a **choice** of interrogating suitable sensor sets in order to accomplish its task.

# Motivation



If the base-station must conduct its task repeatedly at times  $t_1, t_2, t_3, \dots$ ,

# Motivation



If the base-station must conduct its task repeatedly at times  $t_1, t_2, t_3, \dots$ ,

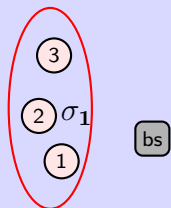
# Definitions: Network Composition

## Definition

**Task:** job that must be conducted repeatedly at the base-station. Requires data from a pod at each time  $t$ .

bs

# Definitions: Network Composition



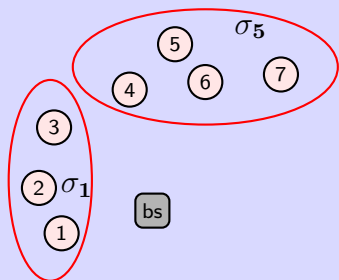
## Definition

**Task:** job that must be conducted repeatedly at the base-station. Requires data from a pod at each time  $t$ .

## Definition

**Pod:** is a set  $\sigma$  of sensors whose measurements at time  $t$  are sufficient for the base-station to accomplish its task at time  $t$ .

# Definitions: Network Composition



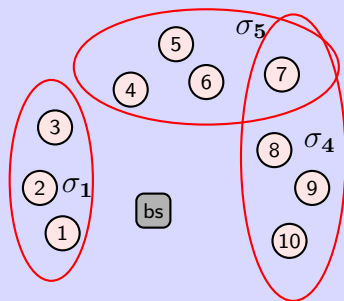
## Definition

**Task:** job that must be conducted repeatedly at the base-station. Requires data from a pod at each time  $t$ .

## Definition

**Pod:** is a set  $\sigma$  of sensors whose measurements at time  $t$  are sufficient for the base-station to accomplish its task at time  $t$ .

# Definitions: Network Composition



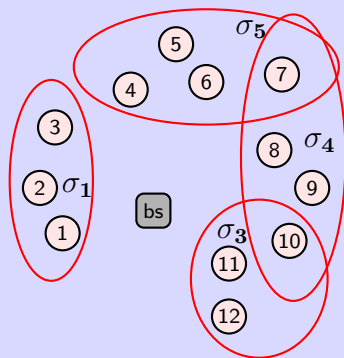
## Definition

**Task:** job that must be conducted repeatedly at the base-station. Requires data from a pod at each time  $t$ .

## Definition

**Pod:** is a set  $\sigma$  of sensors whose measurements at time  $t$  are sufficient for the base-station to accomplish its task at time  $t$ .

# Definitions: Network Composition



## Definition

**Task:** job that must be conducted repeatedly at the base-station. Requires data from a pod at each time  $t$ .

## Definition

**Pod:** is a set  $\sigma$  of sensors whose measurements at time  $t$  are sufficient for the base-station to accomplish its task at time  $t$ .

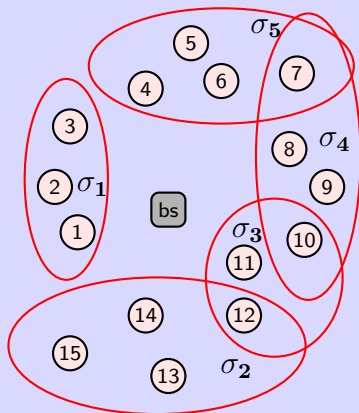
# Definitions: Network Composition

## Definition

**Task:** job that must be conducted repeatedly at the base-station. Requires data from a pod at each time  $t$ .

## Definition

**Pod:** is a set  $\sigma$  of sensors whose measurements at time  $t$  are sufficient for the base-station to accomplish its task at time  $t$ .



# Definitions: Network Composition

## Definition

**Task:** job that must be conducted repeatedly at the base-station. Requires data from a pod at each time  $t$ .

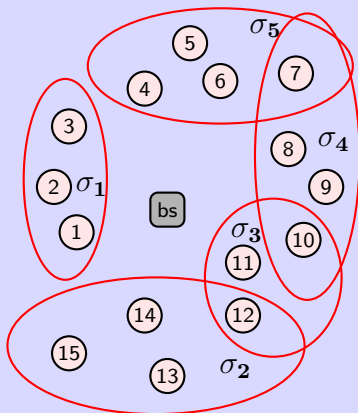
## Definition

**Pod:** is a set  $\sigma$  of sensors whose measurements at time  $t$  are sufficient for the base-station to accomplish its task at time  $t$ .

## Definition

$n_S \doteq$  **number of sensors.**

$n_{P(t)} \doteq$  **number of pods** at time  $t$ .



# Definitions: Battery Model

## Definition

$e_i(t)$  denotes the **energy state** of sensor  $i$  at time  $t$ .

- If sensor  $i$  is interrogated by the base station in  $[t, t + \tau]$ , its energy decrements by  $\beta_i = 1$  after interrogation.
- Model ignores energy loss due to **parasitic discharge, sensing, receiving**.
- Transmission is dominant energy depletion mode in many applications.

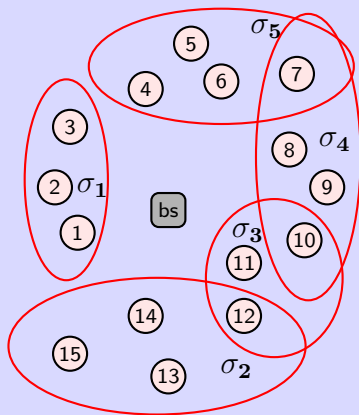
# Definitions: Network Lifetime

## Definition

Network **lifetime**  $L$  is the number of times the base-station can successfully accomplish its task.

$L$  depends on policy and is a function of the energy states of the sensors.

# Definitions: Network Composition



## Definition

$$\mathbb{S}(t) = \{\sigma_1, \dots, \sigma_{n_P(t)}\}$$

are the **acceptable pods** at time  $t$ .

Note that

- The composition of  $\mathbb{S}(t)$  is application dependent.
- $\mathbb{S}(t)$  may be time varying.

# Separation Between Application and Scheduling Layers

- **Application specific tasks** conducted at the base-station are abstracted as **pods** [sufficient sensor sets].
- Our scheduling policies will depend solely on  $\mathbb{S}(t)$  and the sensor energy state  $(e_1, \dots, e_{n_S})$ .
- $\implies$  Isolation of application specific performance objective from the network performance objective.

# Some Applications

Some possible applications that fit naturally in our framework are  
*State estimation, Target tracking, Event detection*

**Dynamical System:**

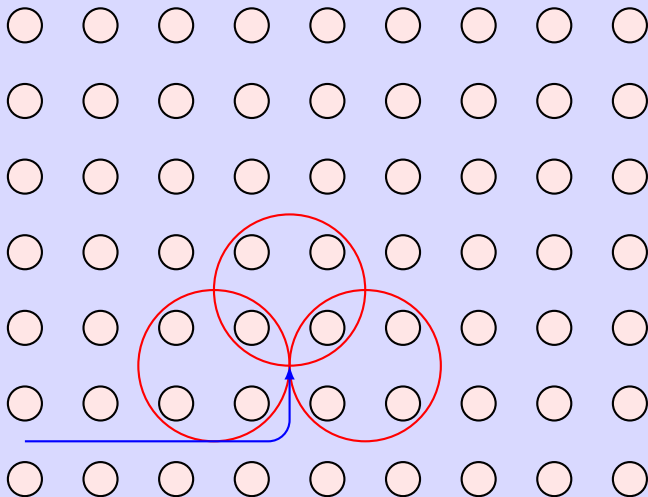
$$\begin{aligned}x(t+1) &= Ax(t) + Gw(t) \\ y_i(t) &= C_i x(t) + v_i(t) \quad i = 1, \dots, n_S\end{aligned}$$

**Computation of Pods  $\mathbb{S}(t)$ :**

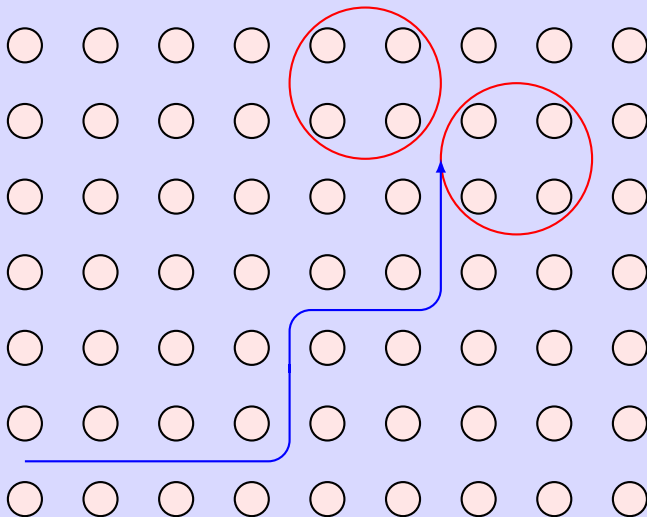
$$\mathbb{S}(t) = \{\sigma \subseteq \{1, \dots, n_S\} : \|P_\sigma(t)\| \leq \gamma\}$$

where  $P_\sigma(t) :=$  estimation error covariance at time  $t$   
 $\gamma :=$  upper bound on norm of  $P_\sigma(t)$

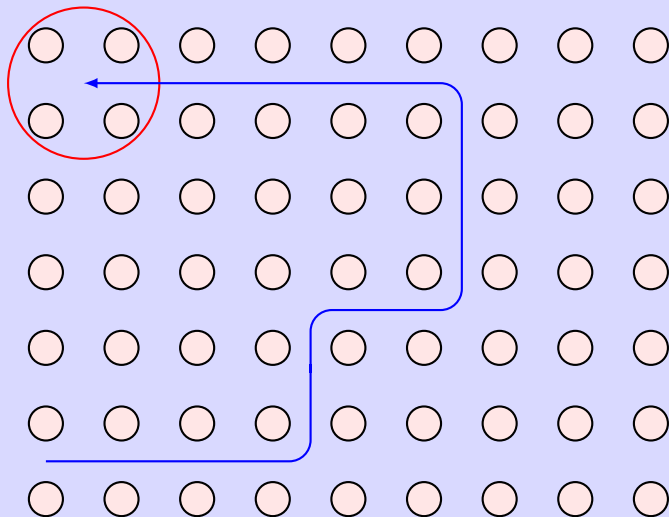
# Target Tracking Example



# Target Tracking Example



# Target Tracking Example

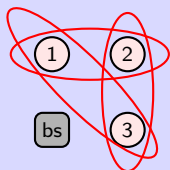


# Example 1: policy = $\{\sigma_3, \sigma_3, \sigma_2\}$

*time:*  $t = 0$

*lifetime:*  $L = 0$

*policy:* **Use pod -**



$$e_1(0) = 2 \quad \circ \circ$$

$$e_2(0) = 4 \quad \circ \circ \circ \circ$$

$$e_3(0) = 3 \quad \circ \circ \circ$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

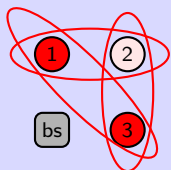
$$\sigma_3 = \{3, 1\}$$

# Example 1: policy = $\{\sigma_3, \sigma_3, \sigma_2\}$

*time:*  $t = 1$

*lifetime:*  $L = 1$

*policy:* **Use pod 3**



$$e_1(1) = 1 \quad \bigcirc$$

$$e_2(1) = 4 \quad \bigcirc \bigcirc \bigcirc \bigcirc$$

$$e_3(1) = 2 \quad \bigcirc \bigcirc$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

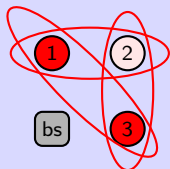
$$\sigma_3 = \{3, 1\}$$

# Example 1: policy = $\{\sigma_3, \sigma_3, \sigma_2\}$

*time:*  $t = 2$

*lifetime:*  $L = 2$

*policy:* **Use pod 3**



$$e_1(2) = 0$$

$$e_2(2) = 4 \quad \bigcirc \bigcirc \bigcirc \bigcirc$$

$$e_3(2) = 1 \quad \bigcirc$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

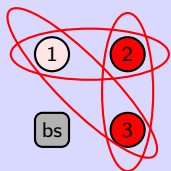
$$\sigma_3 = \{3, 1\}$$

# Example 1: policy = $\{\sigma_3, \sigma_3, \sigma_2\}$

*time:*  $t = 3$

*lifetime:*  $L = 3$

*policy:* **Use pod 2**



$$e_1(3) = 0$$

$$e_2(3) = 3 \quad \bigcirc \bigcirc \bigcirc$$

$$e_3(3) = 0$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

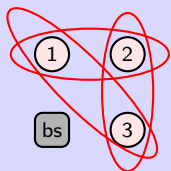
$$\sigma_3 = \{3, 1\}$$

## Example 2: policy = $\{\sigma_2, \sigma_2, \sigma_2 \sigma_1\}$

*time:*  $t = 0$

*lifetime:*  $L = 0$

*policy:* **Use pod -**



$$e_1(0) = 2 \quad \circ \circ$$

$$e_2(0) = 4 \quad \circ \circ \circ \circ$$

$$e_3(0) = 3 \quad \circ \circ \circ$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

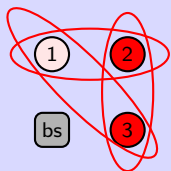
$$\sigma_3 = \{3, 1\}$$

## Example 2: policy = $\{\sigma_2, \sigma_2, \sigma_2 \sigma_1\}$

*time:*  $t = 1$

*lifetime:*  $L = 1$

*policy:* **Use pod 2**



$$e_1(1) = 2 \quad \circ \circ$$

$$e_2(1) = 3 \quad \circ \circ \circ$$

$$e_3(1) = 2 \quad \circ \circ$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

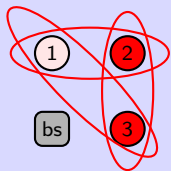
$$\sigma_3 = \{3, 1\}$$

## Example 2: policy = $\{\sigma_2, \sigma_2, \sigma_2 \sigma_1\}$

*time:*  $t = 2$

*lifetime:*  $L = 2$

*policy:* **Use pod 2**



$$e_1(2) = 2 \quad \bigcirc \bigcirc$$

$$e_2(2) = 2 \quad \bigcirc \bigcirc$$

$$e_3(2) = 1 \quad \bigcirc$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

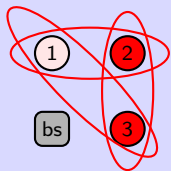
$$\sigma_3 = \{3, 1\}$$

## Example 2: policy = $\{\sigma_2, \sigma_2, \sigma_2 \sigma_1\}$

*time:*  $t = 3$

*lifetime:*  $L = 3$

*policy:* **Use pod 2**



$$e_1(3) = 2 \quad \bigcirc \bigcirc$$

$$e_2(3) = 1 \quad \bigcirc$$

$$e_3(3) = 0$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

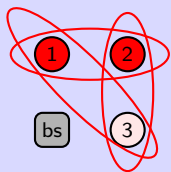
$$\sigma_3 = \{3, 1\}$$

## Example 2: policy = $\{\sigma_2, \sigma_2, \sigma_2 \sigma_1\}$

*time:*  $t = 4$

*lifetime:*  $L = 4$

*policy:* **Use pod 1**



$$e_1(4) = 1 \quad \bigcirc$$

$$e_2(4) = 0$$

$$e_3(4) = 0$$

$$\sigma_1 = \{1, 2\}$$

$$\sigma_2 = \{2, 3\}$$

$$\sigma_3 = \{3, 1\}$$

# Informal Problem Statement

The simple preceding examples demonstrate the importance of sensor scheduling in network lifetime maximization

# Informal Problem Statement

These simple examples show importance of sensor scheduling in network lifetime maximization

# Sensor Scheduling: Time Invariant Pods

## Theorem

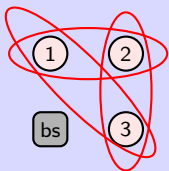
The optimal system lifetime  $L$  and corresponding policy  $x$  are determined by solving the following integer linear program (ILP):

$$\max_{x \in \mathbb{N}^{n_P}} L : Ax \leq E, \quad L = \sum_{j=1}^{n_P} x_j \quad (1)$$

- $A_{ij} = 1 \iff$  sensor  $i \in$  pod  $\sigma_j$ .
- $e_i$  denote the initial available energy in sensor  $i$ .
- $x_j$  denote the number of times pod  $\sigma_j \in \mathbb{S}$  is used.
- $E = [e_1 \ \cdots \ e_{n_S}]^*$  and  $x = [x_1 \ \cdots \ x_{n_P}]^*$

# Sensor Scheduling: Time Invariant Pods

## Proof idea:



$$A = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \quad E = \begin{bmatrix} 2 \\ 4 \\ 3 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$\begin{aligned} x^\circ &= \arg \max_{x \in \mathbb{N}^3} x_1 + x_2 + x_3 \\ &\text{s.t. } Ax \leq E \\ &= \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix} \end{aligned}$$

- $A_{ij} = 1 \iff$  sensor  $i \in \text{pod} \sigma_j$ .
- $e_i$  denotes the initial available energy in sensor  $i$ .
- $x_j$  denotes the number of times pod  $\sigma_j$  is used.
- $E := [e_1 \quad \cdots \quad e_{n_S}]^*$  and  $x := [x_1 \quad \cdots \quad x_{n_P}]^*$ .

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

**Problem Relaxation:**

- ILPs are NP-hard.

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

**Problem Relaxation:**

- ILPs are NP-hard.
- Standard relaxation: let  $x$  live in  $\mathbb{R}^{nP}$ .

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

**Problem Relaxation:**

- ILPs are NP-hard.
- Standard relaxation: let  $x$  live in  $\mathbb{R}^{nP}$ .
- Let  $\underline{x}$  denote the solution of this LP.

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

**Problem Relaxation:**

- ILPs are NP-hard.
- Standard relaxation: let  $x$  live in  $\mathbb{R}^{nP}$ .
- Let  $\underline{x}$  denote the solution of this LP.
- $\lfloor \underline{x} \rfloor$  is a feasible point for the ILP, where the floor is taken component-wise.

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

## Problem Relaxation:

- ILPs are NP-hard.
- Standard relaxation: let  $x$  live in  $\mathbb{R}^{nP}$ .
- Let  $\underline{x}$  denote the solution of this LP.
- $\lfloor \underline{x} \rfloor$  is a feasible point for the ILP, where the floor is taken component-wise.
- $\underline{L}$  = lifetime corresponding to feasible point  $\lfloor \underline{x} \rfloor$ .

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

## Problem Relaxation:

- ILPs are NP-hard.
- Standard relaxation: let  $x$  live in  $\mathbb{R}^{n_P}$ .
- Let  $\underline{x}$  denote the solution of this LP.
- $\lfloor \underline{x} \rfloor$  is a feasible point for the ILP, where the floor is taken component-wise.
- $\underline{L}$  = lifetime corresponding to feasible point  $\lfloor \underline{x} \rfloor$ .
- Then, it is easy to show that  $L^\circ - \underline{L} \leq n_P$ .

# Sensor Scheduling: Time Invariant Pods

**Remark:** Optimal schedules are non-unique.

## Problem Relaxation:

- ILPs are NP-hard.
- Standard relaxation: let  $x$  live in  $\mathbb{R}^{n_P}$ .
- Let  $\underline{x}$  denote the solution of this LP.
- $\lfloor \underline{x} \rfloor$  is a feasible point for the ILP, where the floor is taken component-wise.
- $\underline{L}$  = lifetime corresponding to feasible point  $\lfloor \underline{x} \rfloor$ .
- Then, it is easy to show that  $L^\circ - \underline{L} \leq n_P$ .

## Theorem

*This relaxation will offer suboptimal policy that is with a constant of  $n_P$  of the optimal lifetime.*

# Sensor Scheduling: Time-varying Pods

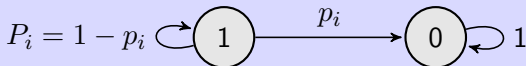
## Theorem

*The lifetime  $L$  is achievable on the horizon  $N$  if and only if the following ILP over the decision variables  $x(t)$  and  $N$  is feasible:*

$$Ax \leq E, \quad L = \sum_{t=1}^N \sum_{j=1}^{n_P(t)} x_j(t), \quad \sum_{j=1}^{n_P(t)} x_j(t) \leq 1 \quad \forall t \quad (2)$$

# Sensor Random Failure Model

- We allow for sensors to abruptly fail random times.
- The binary valued stochastic process  $\{\theta_i(t)\}_t$  models the occurrence of an **irreparable malfunction at sensor  $i$** .



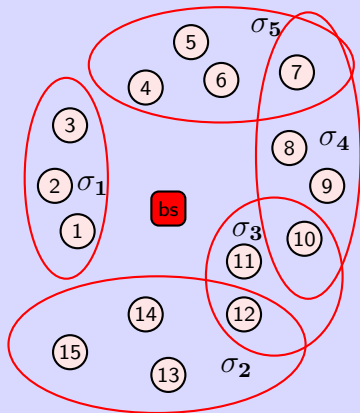
- Sensor  $i$  will cease functioning if the  $\theta_i$  transitions to the absorbing state  $\theta_i = 0$ .
- $\theta_i(t) \perp \theta_j(s) \quad \forall i \neq j, s, t$

# Example of Scheduling under Random Failure

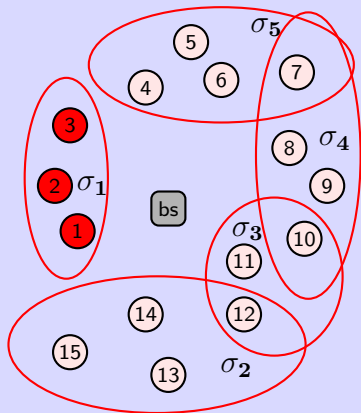
**time:**  $t$

Base-station communicates with a pod  
from the family

$$\mathbb{S}(t) = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5\}.$$



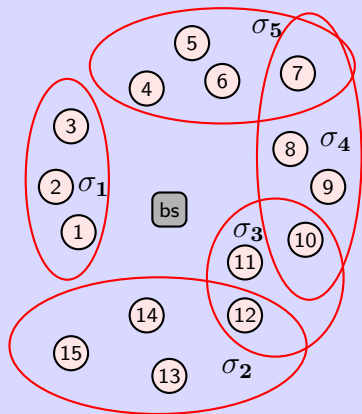
# Example of Scheduling under Random Failure



**time:**  $t$

Chosen pod  $\sigma_1$  then transmits to base-station, depleting the energy of its sensors.

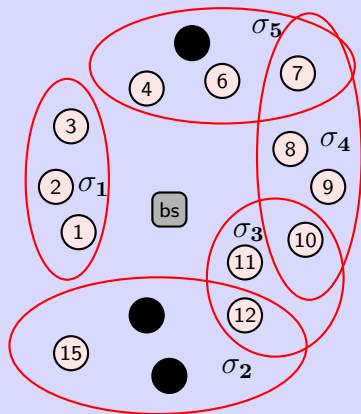
# Example of Scheduling under Random Failure



**time:**  $t$

Network returns to wait mode.

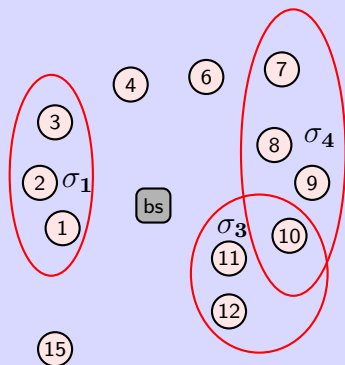
# Example of Scheduling under Random Failure



**time:**  $t$

Sensors may randomly fail.

# Example of Scheduling under Random Failure



**time:**  $t + 1$

Process repeats with remaining viable pods,  $\mathbb{S}(t) = \{\sigma_1, \sigma_3, \sigma_4\}$ .

## Definition

With random sensor failures, we shall be concerned with the **expected lifetime**  $\mathcal{L} = \mathbf{E}[L]$

- Clearly  $L$  depends on scheduling policy which is a function of the energy states of the sensors.
- Requires that we have access to the true sensor energy states  $\{e_i(t)\}$  for all  $t$ .

# Certainty Equivalence Assumption

- Our scheduling policies require that the base-station can **detect sensor failures instantaneously**.
- In practice, however, we would at best receive **noisy measurements** of  $e_i(t)$  at times  $t$  when sensor  $i$  is interrogated.

# Certainty Equivalence Assumption

- Given a measurement of  $e_i(t)$  at time of the most recent sensor interrogation and using a simple battery model, we can compute an optimal estimate  $\hat{e}_i(t+m)$  of the energy state:

# Certainty Equivalence Assumption

- Given a measurement of  $e_i(t)$  at time of the most recent sensor interrogation and using a simple battery model, we can compute an optimal estimate  $\hat{e}_i(t+m)$  of the energy state:

$$\hat{e}_i(t+m) = \begin{cases} e_i(t) \cdot P_i^m & \text{MV estimate} \\ e_i(t) \cdot \mathbb{I}\{P_i^m \geq 1/2\} & \text{ML estimate} \end{cases}$$

# Certainty Equivalence Assumption

- Given a measurement of  $e_i(t)$  at time of the most recent sensor interrogation and using a simple battery model, we can compute an optimal estimate  $\hat{e}_i(t+m)$  of the energy state:

$$\hat{e}_i(t+m) = \begin{cases} e_i(t) \cdot P_i^m & \text{MV estimate} \\ e_i(t) \cdot \mathbb{I}\{(P_i^m \geq 1/2)\} & \text{ML estimate} \end{cases}$$

- Inspired by certainty-equivalence ideas, we can use the estimates  $\hat{e}_i(t)$  as surrogates for the true energy state  $e_i(t)$  in our scheduling policies at time  $t$ .

# Dynamic Programming: Recursive Definition of Optimal Expected Lifetime

## Definition

$\mathcal{L}_i(e_1, \dots, e_{n_S})$ , expected lifetime resulting from an **initial use of pod  $i$**  and **optimal pod usage thereafter**.

# Dynamic Programming: Recursive Definition of Optimal Expected Lifetime

## Definition

$\mathcal{L}_i(e_1, \dots, e_{n_S})$ , expected lifetime resulting from an **initial use of pod  $i$**  and **optimal pod usage thereafter**.

Using this we can write a recurrence formula for optimal lifetime:

$$\mathcal{L}^\circ(e_1, \dots, e_{n_S}) = \max_{i \in \mathcal{S}} \mathcal{L}_i(e_1, \dots, e_{n_S})$$

# Dynamic Programming: Recursive Definition of Optimal Expected Lifetime

## Definition

$\mathcal{L}_i(e_1, \dots, e_{n_S})$ , expected lifetime resulting from an **initial use of pod  $i$**  and **optimal pod usage thereafter**.

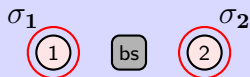
Using this we can write a recurrence formula for optimal lifetime:

$$\mathcal{L}^\circ(e_1, \dots, e_{n_S}) = \max_{i \in \mathbb{S}} \mathcal{L}_i(e_1, \dots, e_{n_S})$$

This approach has a *time complexity* of  $O(c^{n_S})$  where  $c = \max_i e_i$ .

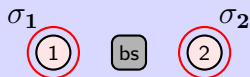
# Properties of Optimal Expected Lifetime Under Random Failure

For the case of a network consisting of two singleton pods (i.e. one sensor in each pod), we have



# Properties of Optimal Expected Lifetime Under Random Failure

For the case of a network consisting of two singleton pods (i.e. one sensor in each pod), we have

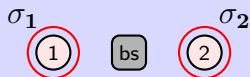


where, for  $i = 1, 2$ ,

$e_i :=$  energy of sensor  $i$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of a network consisting of two singleton pods (i.e. one sensor in each pod), we have



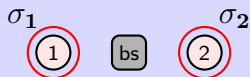
where, for  $i = 1, 2$ ,

$e_i$  := energy of sensor  $i$

$p_i$  := failure probability of sensor  $i$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of a network consisting of two singleton pods (i.e. one sensor in each pod), we have



where, for  $i = 1, 2$ ,

$e_i$  := energy of sensor  $i$

$p_i$  := failure probability of sensor  $i$

$P_i$  :=  $1 - p_i$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of two singleton pods (i.e. one sensor in each pod), we have

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of two singleton pods (i.e. one sensor in each pod), we have

$$\mathcal{L}_1(e_1, e_2) = 1$$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of two singleton pods (i.e. one sensor in each pod), we have

$$\mathcal{L}_1(e_1, e_2) = 1 + \mathbf{P}\{\text{sensors 1 and 2 live}\} \mathcal{L}^\circ(e_1 - 1, e_2)$$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of two singleton pods (i.e. one sensor in each pod), we have

$$\begin{aligned} \mathcal{L}_1(e_1, e_2) = 1 &+ \mathbf{P} \{ \text{sensors 1 and 2 live} \} \mathcal{L}^\circ(e_1 - 1, e_2) \\ &+ \mathbf{P} \{ \text{sensor 1 lives and sensor 2 dies} \} \mathcal{L}^\circ(e_1 - 1, 0) \end{aligned}$$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of two singleton pods (i.e. one sensor in each pod), we have

$$\begin{aligned}\mathcal{L}_1(e_1, e_2) = & 1 + \mathbf{P} \{ \text{sensors 1 and 2 live} \} \mathcal{L}^\circ(e_1 - 1, e_2) \\ & + \mathbf{P} \{ \text{sensor 1 lives and sensor 2 dies} \} \mathcal{L}^\circ(e_1 - 1, 0) \\ & + \mathbf{P} \{ \text{sensor 1 dies and sensor 2 lives} \} \mathcal{L}^\circ(0, e_2)\end{aligned}$$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of two singleton pods (i.e. one sensor in each pod), we have

$$\begin{aligned}\mathcal{L}_1(e_1, e_2) = 1 &+ \mathbf{P} \{ \text{sensors 1 and 2 live} \} \mathcal{L}^\circ(e_1 - 1, e_2) \\ &+ \mathbf{P} \{ \text{sensor 1 lives and sensor 2 dies} \} \mathcal{L}^\circ(e_1 - 1, 0) \\ &+ \mathbf{P} \{ \text{sensor 1 dies and sensor 2 lives} \} \mathcal{L}^\circ(0, e_2) \\ &+ \mathbf{P} \{ \text{sensor 1 dies and sensor 2 dies} \} \mathcal{L}^\circ(0, 0)\end{aligned}$$

# Properties of Optimal Expected Lifetime Under Random Failure

For the case of two singleton pods (i.e. one sensor in each pod), we have

$$\begin{aligned} \mathcal{L}_1(e_1, e_2) = 1 &+ \mathbf{P} \{ \text{sensors 1 and 2 live} \} \mathcal{L}^\circ(e_1 - 1, e_2) \\ &+ \mathbf{P} \{ \text{sensor 1 lives and sensor 2 dies} \} \mathcal{L}^\circ(e_1 - 1, 0) \\ &+ \mathbf{P} \{ \text{sensor 1 dies and sensor 2 lives} \} \mathcal{L}^\circ(0, e_2) \\ &+ \mathbf{P} \{ \text{sensor 1 dies and sensor 2 dies} \} \mathcal{L}^\circ(0, 0) \end{aligned}$$

which reduces to

$$\mathcal{L}_1(e_1, e_2) = 1 + P_1 P_2 \mathcal{L}^\circ(e_1 - 1, e_2) + P_1 p_2 \mathcal{L}^\circ(e_1 - 1, 0) + p_1 P_2 \mathcal{L}^\circ(0, e_2)$$

# Properties of Optimal Expected Lifetime Under Random Failure

*Properties of optimal lifetime  $\mathcal{L}^\circ(e_1, e_2)$  for two singleton pods.*

# Properties of Optimal Expected Lifetime Under Random Failure

*Properties of optimal lifetime  $\mathcal{L}^\circ(e_1, e_2)$  for two singleton pods.*

$$\textcircled{1} \mathcal{L}_1(e_1, e_2) = 1 + P_1 P_2 \mathcal{L}^\circ(e_1 - 1, e_2) + P_1 p_2 \mathcal{L}^\circ(e_1 - 1, 0) + p_1 P_2 \mathcal{L}^\circ(0, e_2)$$

# Properties of Optimal Expected Lifetime Under Random Failure

*Properties of optimal lifetime  $\mathcal{L}^\circ(e_1, e_2)$  for two singleton pods.*

- 1  $\mathcal{L}_1(e_1, e_2) = 1 + P_1 P_2 \mathcal{L}^\circ(e_1 - 1, e_2) + P_1 p_2 \mathcal{L}^\circ(e_1 - 1, 0) + p_1 P_2 \mathcal{L}^\circ(0, e_2)$
- 2  $\mathcal{L}^\circ(e_1, e_2) = \max_{i=1,2} \mathcal{L}_i(e_1, e_2)$

# Properties of Optimal Expected Lifetime Under Random Failure

*Properties of optimal lifetime  $\mathcal{L}^\circ(e_1, e_2)$  for two singleton pods.*

- 1  $\mathcal{L}_1(e_1, e_2) = 1 + P_1 P_2 \mathcal{L}^\circ(e_1 - 1, e_2) + P_1 p_2 \mathcal{L}^\circ(e_1 - 1, 0) + p_1 P_2 \mathcal{L}^\circ(0, e_2)$
- 2  $\mathcal{L}^\circ(e_1, e_2) = \max_{i=1,2} \mathcal{L}_i(e_1, e_2)$
- 3  $\mathcal{L}^\circ(e_1, 0) = 1 + P_1 + \dots + P_1^{e_1-1} = (1 - P_1^{e_1})/p_1$

# Properties of Optimal Expected Lifetime Under Random Failure

*Properties of optimal lifetime  $\mathcal{L}^\circ(e_1, e_2)$  for two singleton pods.*

- 1  $\mathcal{L}_1(e_1, e_2) = 1 + P_1 P_2 \mathcal{L}^\circ(e_1 - 1, e_2) + P_1 p_2 \mathcal{L}^\circ(e_1 - 1, 0) + p_1 P_2 \mathcal{L}^\circ(0, e_2)$
- 2  $\mathcal{L}^\circ(e_1, e_2) = \max_{i=1,2} \mathcal{L}_i(e_1, e_2)$
- 3  $\mathcal{L}^\circ(e_1, 0) = 1 + P_1 + \dots + P_1^{e_1 - 1} = (1 - P_1^{e_1})/p_1$
- 4  $\mathcal{L}_1(e_1, e_2 - 1) = \mathcal{L}_2(e_1 - 1, e_2)$

# Sensor Scheduling: Two Non-intersecting Pods ( $\sigma_1 \cap \sigma_2 = \emptyset$ ) with Different Failure Probabilities

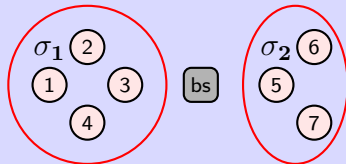
## Theorem

The optimal scheduling policy for maximizing the expected lifetime  $\mathcal{L}$  is given by

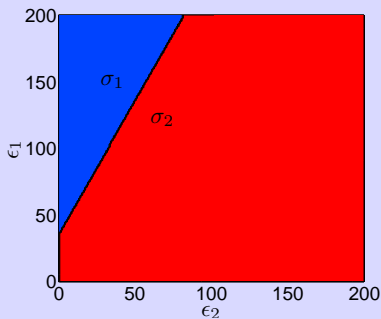
$$i^\circ(t) = \arg \min_{i \in \{1,2\}} \epsilon_i(t) \log(1 - \pi_i) - \log(\pi_i) \quad (3)$$

where

$$\epsilon_i(t) := \min_{j \in \sigma_i} e_j(t) \quad \text{and} \quad \pi_i := 1 - \prod_{j \in \sigma_i} P_j$$



# Sensor Scheduling: Two Non-intersecting Pods ( $\sigma_1 \cap \sigma_2 = \emptyset$ ) with Different Failure Probabilities



## Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

- WLOG, we will represent each pod as a single sensor (a pod fails  $\iff$  at least one of its member sensors fails).

## Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

- WLOG, we will represent each pod as a single sensor (a pod fails  $\iff$  at least one of its member sensors fails).
- For clarity, let  $e(t) = (m, n)$  and  $(p_1, p_2) = (p, q)$  and the theorem can be restated as

## Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

- WLOG, we will represent each pod as a single sensor (a pod fails  $\iff$  at least one of its member sensors fails).
- For clarity, let  $e(t) = (m, n)$  and  $(p_1, p_2) = (p, q)$  and the theorem can be restated as

use sensor 1 if  $pQ^n > qP^m$

use sensor 2 if  $pQ^n < qP^m$

use either sensor if  $pQ^n = qP^m$

## Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

- WLOG, we will represent each pod as a single sensor (a pod fails  $\iff$  at least one of its member sensors fails).
- For clarity, let  $e(t) = (m, n)$  and  $(p_1, p_2) = (p, q)$  and the theorem can be restated as

use sensor 1 if  $pQ^n > qP^m$

use sensor 2 if  $pQ^n < qP^m$

use either sensor if  $pQ^n = qP^m$

- We first show by induction on the total energy  $E = m + n$  that

$$pQ^n \geq qP^m \implies \mathcal{L}_1(m, n) \geq \mathcal{L}_2(m, n)$$

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

- The assertion is easily verified for the base step  $E = 1$ .

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

- The assertion is easily verified for the base step  $E = 1$ .
- Fix  $(m, n)$  and suppose  $m + n = E^*$ .

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

- The assertion is easily verified for the base step  $E = 1$ .
- Fix  $(m, n)$  and suppose  $m + n = E^*$ .
- Assume that the result holds for all energy states with total energy  $E < E^*$ .

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (Proof)

- The assertion is easily verified for the base step  $E = 1$ .
- Fix  $(m, n)$  and suppose  $m + n = E^*$ .
- Assume that the result holds for all energy states with total energy  $E < E^*$ .
- Define  $\Delta = \mathcal{L}_1(m, n) - \mathcal{L}_2(m, n)$ .

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

We have

$$\Delta = PQ(\mathcal{L}^\circ(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m$$

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

We have

$$\begin{aligned}\Delta &= PQ(\mathcal{L}^\circ(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\ &\geq PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m\end{aligned}$$

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

We have

$$\begin{aligned} \Delta &= PQ(\mathcal{L}^\circ(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\ &\geq PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\ &\quad \text{since } \mathcal{L}^\circ(m, n) \geq \mathcal{L}_i(m, n) \quad \forall i \end{aligned}$$

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

We have

$$\begin{aligned}
 \Delta &= PQ(\mathcal{L}^\circ(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\
 &\geq PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\
 &\quad \text{since } \mathcal{L}^\circ(m, n) \geq \mathcal{L}_i(m, n) \quad \forall i \\
 &= PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}_1(m, n-1)) + pQ^n - qP^m
 \end{aligned}$$

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

We have

$$\begin{aligned}
 \Delta &= PQ(\mathcal{L}^\circ(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\
 &\geq PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\
 &\quad \text{since } \mathcal{L}^\circ(m, n) \geq \mathcal{L}_i(m, n) \quad \forall i \\
 &= PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}_1(m, n-1)) + pQ^n - qP^m \\
 &\quad \text{by the induction hypothesis, } m+n-1 < E
 \end{aligned}$$

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

We have

$$\begin{aligned}
 \Delta &= PQ(\mathcal{L}^\circ(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\
 &\geq PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}^\circ(m, n-1)) + pQ^n - qP^m \\
 &\quad \text{since } \mathcal{L}^\circ(m, n) \geq \mathcal{L}_i(m, n) \quad \forall i \\
 &= PQ(\mathcal{L}_2(m-1, n) - \mathcal{L}_1(m, n-1)) + pQ^n - qP^m \\
 &\quad \text{by the induction hypothesis, } m+n-1 < E \\
 &\geq 0 + pQ^n - qP^m
 \end{aligned}$$

giving us the desired result.

# Sensor Scheduling: Two Non-intersecting Pods with Different Failure Probabilities (**Proof**)

A symmetric argument shows that using sensor 2 at energy state  $(m, n)$  is optimal if  $pQ^n \leq qP^m$ , completing the proof.  $\square$

# Sensor Scheduling: Many Non-intersecting Pods

$(\sigma_i \cap \sigma_j = \emptyset \quad \forall i \neq j)$  with Identical Failure Probabilities

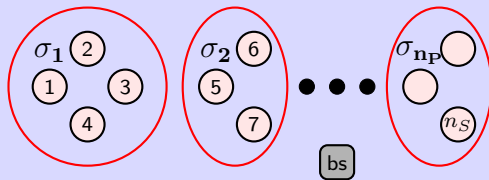
## Theorem

*The optimal scheduling policy for maximizing the expected lifetime  $\mathcal{L}$  is to use the most energetic sensor first, i.e.*

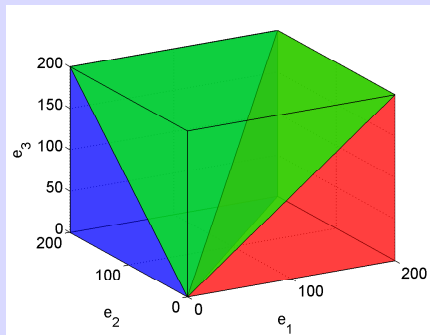
$$i^\circ(t) = \arg \max_{i \in \mathbb{S}} \epsilon_i(t) \quad (4)$$

where

$$\epsilon_i(t) := \min_{j \in \sigma_i} e_j(t)$$



# Sensor Scheduling: Two Non-intersecting Pods ( $\sigma_1 \cap \sigma_2 = \emptyset$ ) with Different Failure Probabilities



# Sensor Scheduling: Many Non-intersecting Pods with Identical Failure Probabilities (**Proof**)

- For transparency, we prove the theorem for the *three* sensor case.
- Let  $p$  be the failure rate for any sensor and let the energy state be  $e(t) = (m, n, k)$ .
- We prove by induction on the total energy  $E = m + n + k$  that

$$m \geq n \implies \mathcal{L}_1(m, n, k) \geq \mathcal{L}_2(m, n, k)$$

$$m \geq k \implies \mathcal{L}_1(m, n, k) \geq \mathcal{L}_3(m, n, k)$$

# Sensor Scheduling: Many Non-intersecting Pods with Identical Failure Probabilities (**Proof**)

- The assertion is easily verified for the base step  $E = 1$ .
- Fix  $(m, n, k)$  and suppose  $m + n + k = E^*$ .
- Assume that the result holds for all energy states with total energy  $E < E^*$ .
- Define  $\Delta^{12} = \mathcal{L}_1(m, n, k) - \mathcal{L}_2(m, n, k)$ .

# Sensor Scheduling: Many Non-intersecting Pods with Identical Failure Probabilities (**Proof**)

We now have

$$\begin{aligned}
 \Delta^{12} &= P^3 \underbrace{(\mathcal{L}^\circ(m-1, n, k) - \mathcal{L}^\circ(m, n-1, k))}_{T^a} \\
 &+ P^2 p \underbrace{(\mathcal{L}^\circ(m-1, n, 0) - \mathcal{L}^\circ(m, n-1, 0))}_{T^b} \\
 &+ P^2 p \underbrace{\left( \begin{array}{l} \mathcal{L}^\circ(m-1, 0, k) - \mathcal{L}^\circ(m, 0, k) + \\ \mathcal{L}^\circ(0, n, k) - \mathcal{L}^\circ(0, n-1, k) \end{array} \right)}_{T^c} \\
 &+ P^2 p^2 \underbrace{\left( \begin{array}{l} \mathcal{L}^\circ(m-1, 0, 0) - \mathcal{L}^\circ(m, 0, 0) + \\ \mathcal{L}^\circ(0, n, 0) - \mathcal{L}^\circ(0, n-1, 0) \end{array} \right)}_{T^d}
 \end{aligned}$$

## Sensor Scheduling: Many Non-intersecting Pods with Identical Failure Probabilities (**Proof**)

- The terms  $T^a$  and  $T^b$  are  $\geq 0$  by the induction hypothesis as the total energy is  $m - 1 + n + k < E^*$ .
- The term  $T^d$  simplifies to  $T^d = P^{n-1} - P^{m-1} > 0$  as  $m > n$ .
- The difficult term is  $T^c$ .

# Sensor Scheduling: Many Non-intersecting Pods with Identical Failure Probabilities (**Proof**)

- Using previously defined identities, we can write a difference equation for  $\psi(m, k) = L(m, 0, k) - L(m - 1, 0, k)$  and obtain the closed form expression:

$$\psi(m, k) = \begin{cases} P^{m-1} - P^{2m-k-1} + P^{2m-2k} X(k) & m \geq k \\ \frac{pP^m(1 - P^{2k-2m})}{1 - P^2} + P^{2k-2m} X(m) & m \leq k \end{cases}$$

where

$$X(k) = \left( P^{4k-1} \left( \frac{p}{1 - P^3} \right) + P^k \left( \frac{1 - P^2}{1 - P^3} \right) \right)$$

# Sensor Scheduling: Many Non-intersecting Pods with Identical Failure Probabilities (**Proof**)

- We can now write  $T^c = -\psi(m, k) + \psi(n, k)$ .
- After much algebra, it can be verified that  $m \geq n \implies T^c \geq 0$ .
- As a consequence we have  $\Delta^{12} \geq 0$ .
- A similar argument shows that  $m \geq k \implies \Delta^{13} \geq 0$ .
- As a result using sensor 1 is optimal at energy state  $(m, n, k)$  if  $m > n$  and  $m \geq k$ .  $\square$

# Conclusions

Some general principles emerge from our results.

- All sensors equally energetic  $\implies$  use the least reliable sensor first.
- All sensors equally reliable  $\implies$  use the most energetic sensor first.
- i.e. use the sensor with the greatest expected energy loss.
- Equitable risk distribution across sensors.

# Future Work

Many challenging problems remain open:

# Future Work

Many challenging problems remain open:

- How to compute  $\mathbb{S}(t)$  in different application domains?

# Future Work

Many challenging problems remain open:

- How to compute  $\mathbb{S}(t)$  in different application domains?
- Connections with model predictive control when we have a dynamic model of  $\mathbb{S}(t)$ .

# Future Work

Many challenging problems remain open:

- How to compute  $\mathbb{S}(t)$  in different application domains?
- Connections with model predictive control when we have a dynamic model of  $\mathbb{S}(t)$ .
- Link failure.

# Future Work

Many challenging problems remain open:

- How to compute  $\mathbb{S}(t)$  in different application domains?
- Connections with model predictive control when we have a dynamic model of  $\mathbb{S}(t)$ .
- Link failure.
- How does packet loss and latency affect system lifetime?

# Future Work

Many challenging problems remain open:

- How to compute  $\mathbb{S}(t)$  in different application domains?
- Connections with model predictive control when we have a dynamic model of  $\mathbb{S}(t)$ .
- Link failure.
- How does packet loss and latency affect system lifetime?
- For multiple base stations competing to use the same sensor resources, we naturally obtain multi-objective problems in sensor selection and scheduling. What is a good metric of lifetime in this case? How do we optimize lifetime in case limited inter-base-station communication?