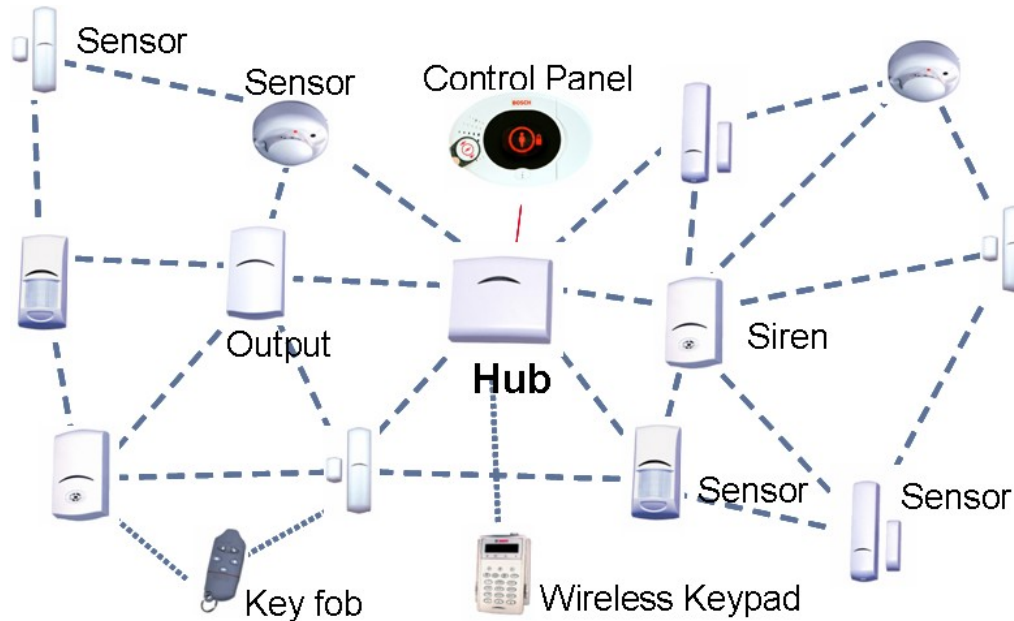


# Efficient Sensor Networks for Smart Environments



Huang Lee  
Wireless Sensor Networks Lab  
Stanford University



**BOSCH**  
Invented for life

# Wireless Sensor Networks

## ■ Wireless Sensor Networks for Smart Environment Applications

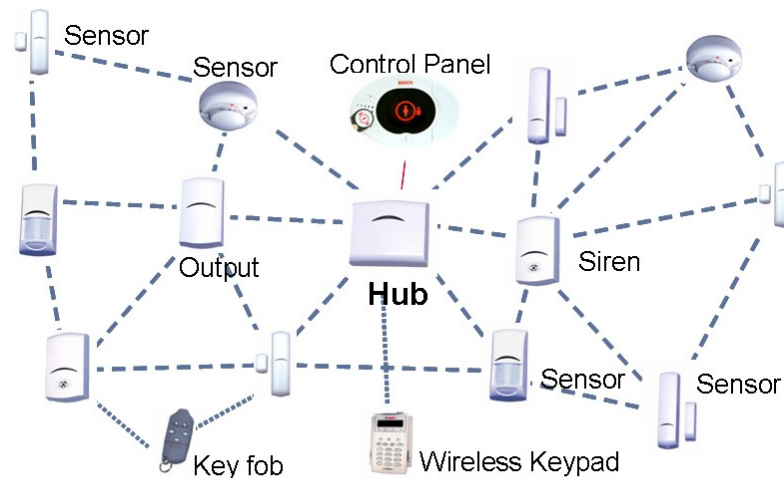
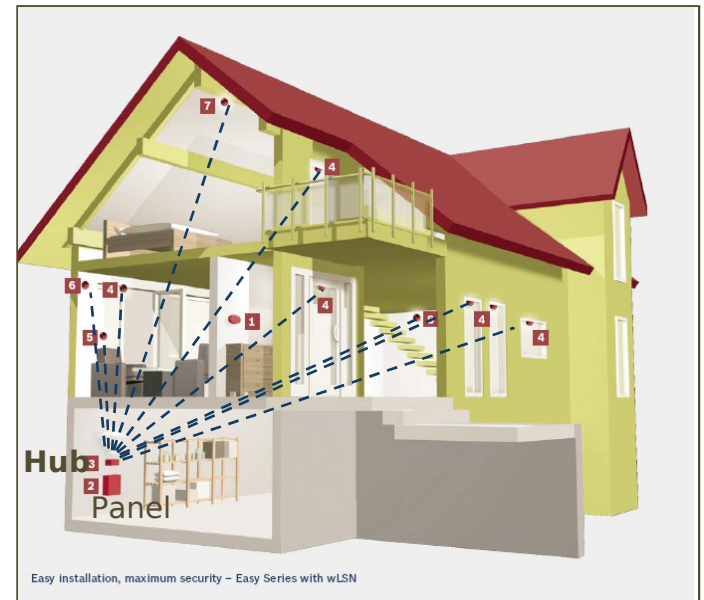
- Fire alarm systems
- Smart surveillance systems
- Intelligent control systems

## ■ Features

- Battery-powered sensors
- Hub (Base station)
  - Ac-powered
  - Computational power

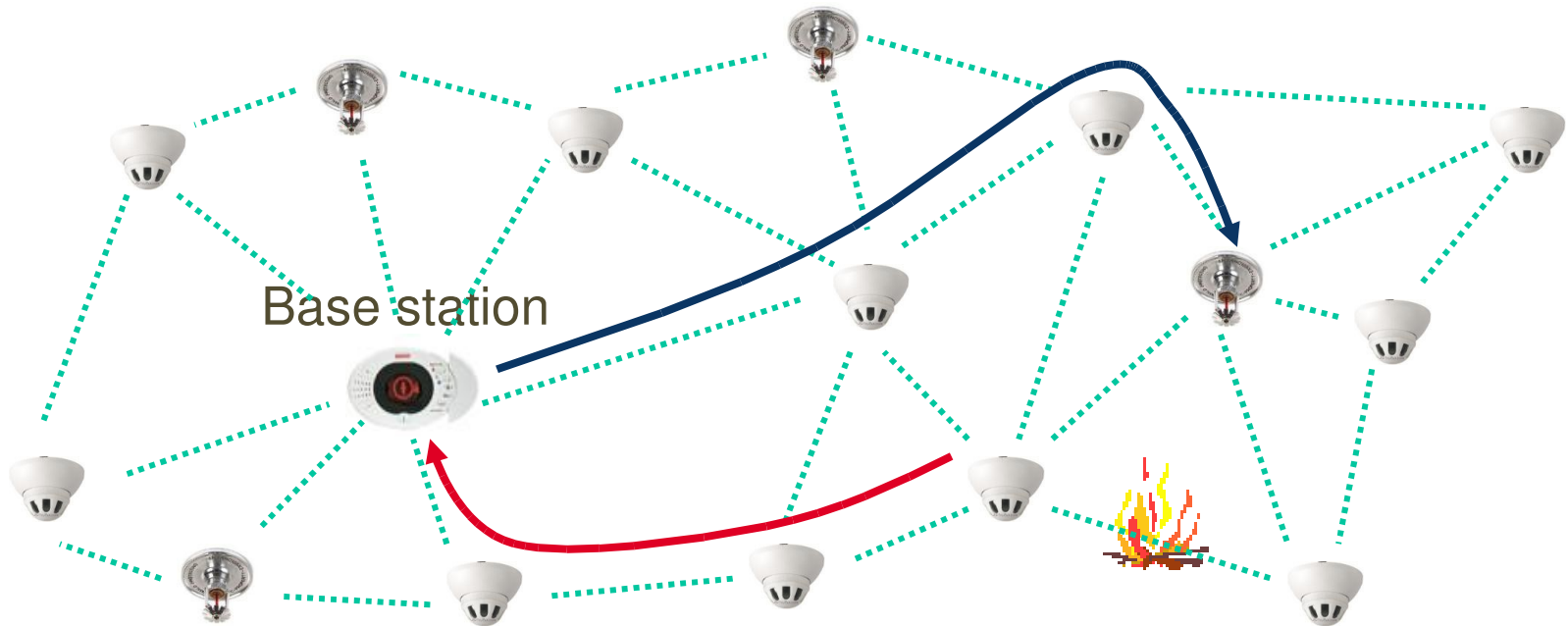
## ■ Operations

- Event-driven operation
- Time-driven operation



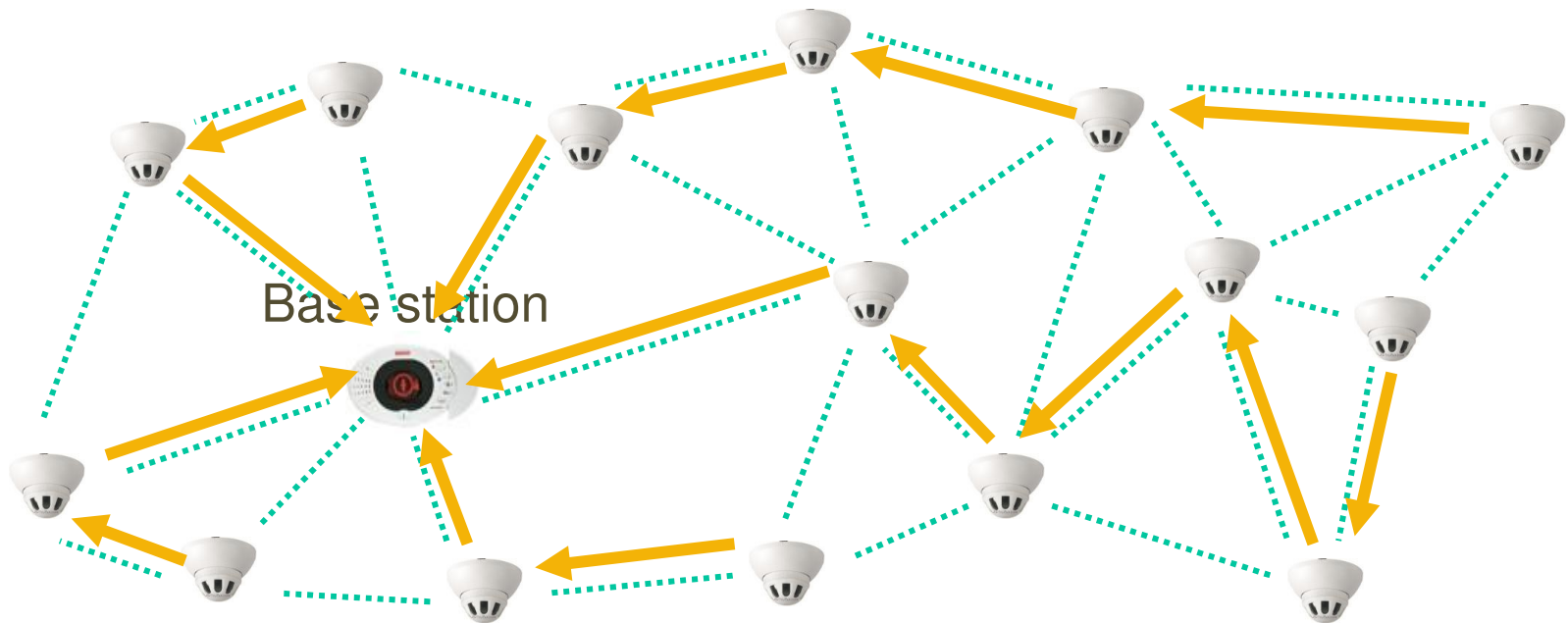
# Event-Driven Operation: Motivation

- Sensing and actuation application
  - For example, fire alarm sprinkler systems
  - Detect and notify **rare but urgent** events
  - Receive **delay sensitive** actuation commands



# Time-Driven Operation: Motivation

- Periodic data gathering
  - Nodes report data via the data aggregation tree
  - One of the most common operations
  - Take 30%-60% of energy consumption of a node



# Network Operations

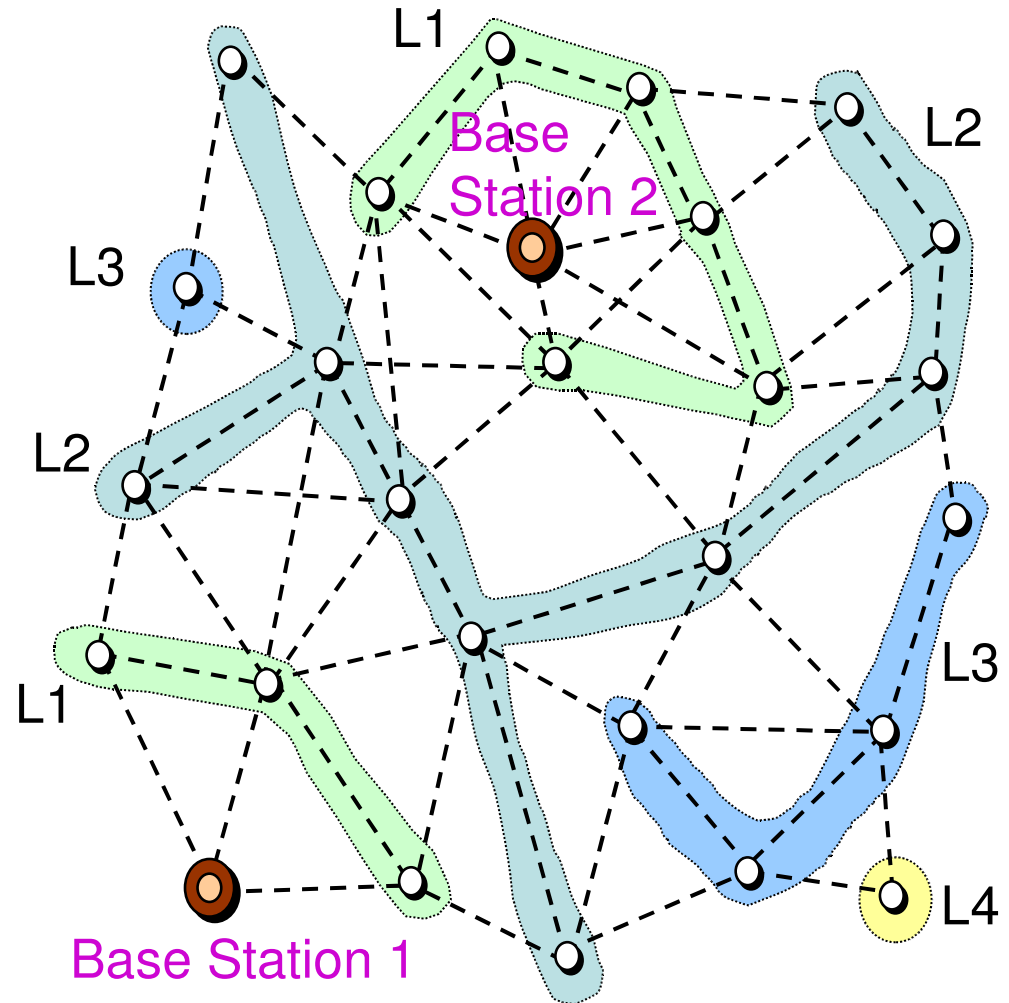
	Event-driven	Time-driven
Application	Sensing and actuation	Periodic data gathering
Topology	Communication path	Data collection tree
Direction	Two-way traffic	One-way traffic
Traffic	Bursty	Periodic
Feature	Delay sensitive	Energy hungry
Issue	How to optimize the trade-off between the delay and power?	How to optimize the energy (lifetime) for data collection?

# Outline

- Event-driven operation
  - Energy-efficient wake-up scheduling
  - Efficient algorithm to double the performance of delay and power trade-off
- Time-driven operation
  - Energy-efficient data aggregation
  - Near lifetime-optimal and practical algorithm
- Intelligent light control
  - Light control based on camera sensor observations

# Network Topology

- Stationary
- Synchronized
- Dense network
- Levels (L1 – L4) are assigned in breadth first order



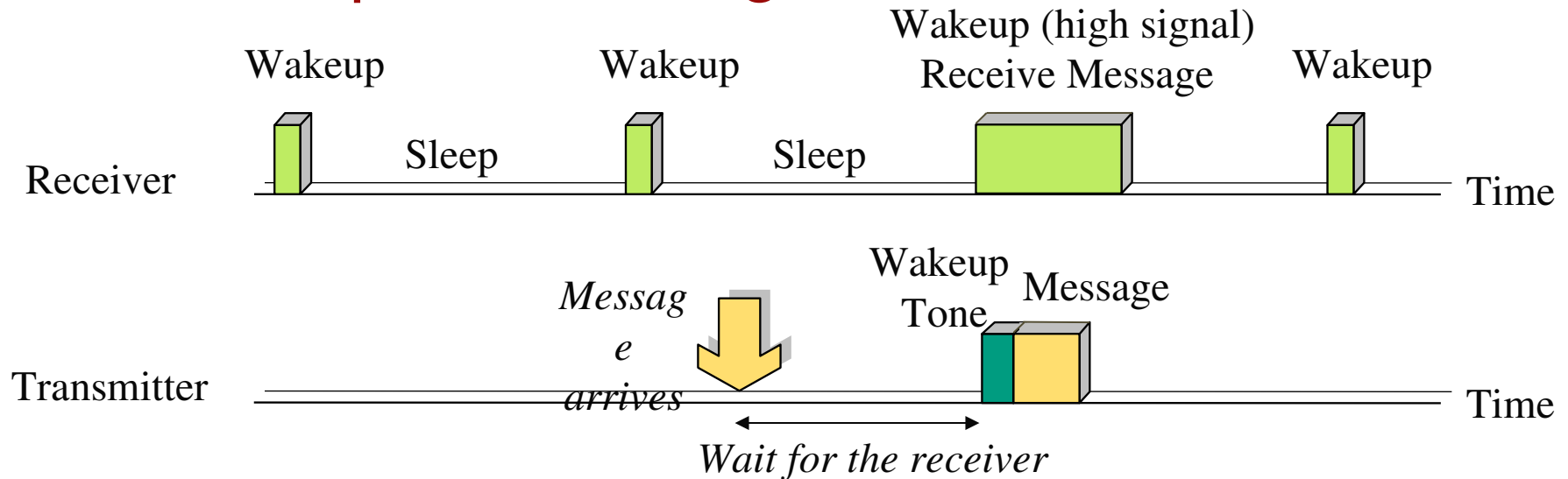
# Outline

- Event-driven operation
- Time-driven operation
- Intelligent light control

# Outline

- Event-driven operation
  - Wake-up scheduling
- Time-driven operation
- Intelligent light control

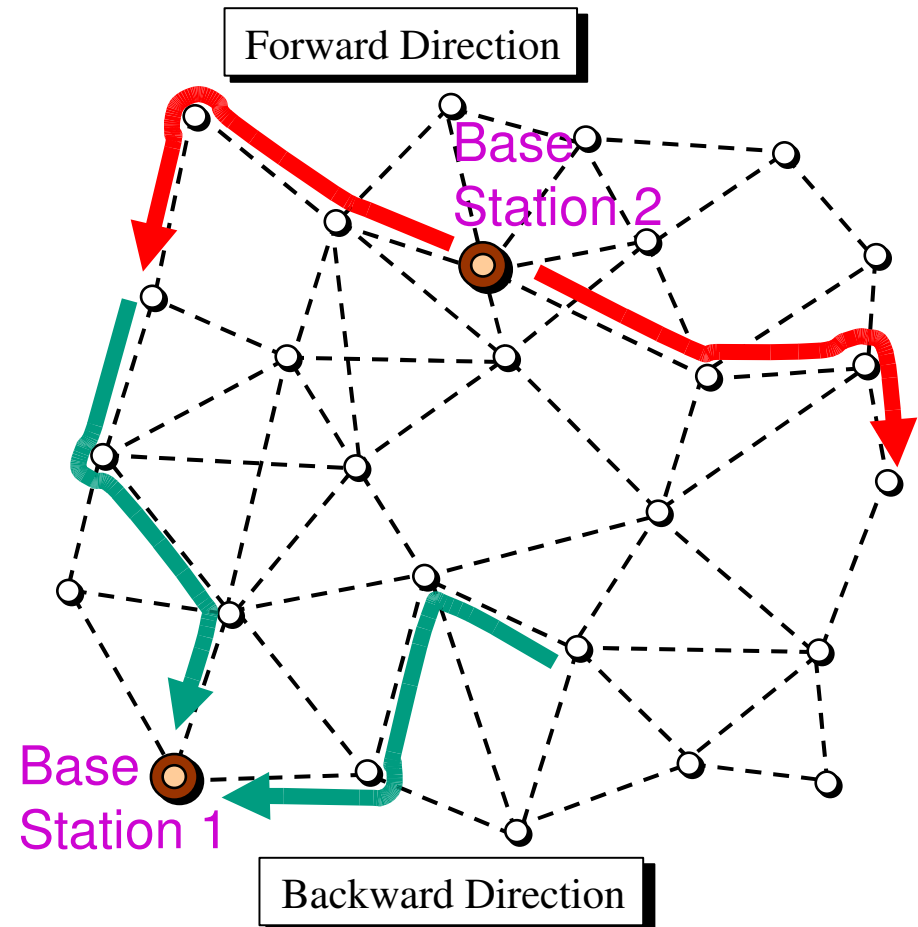
# Wake-up Scheduling



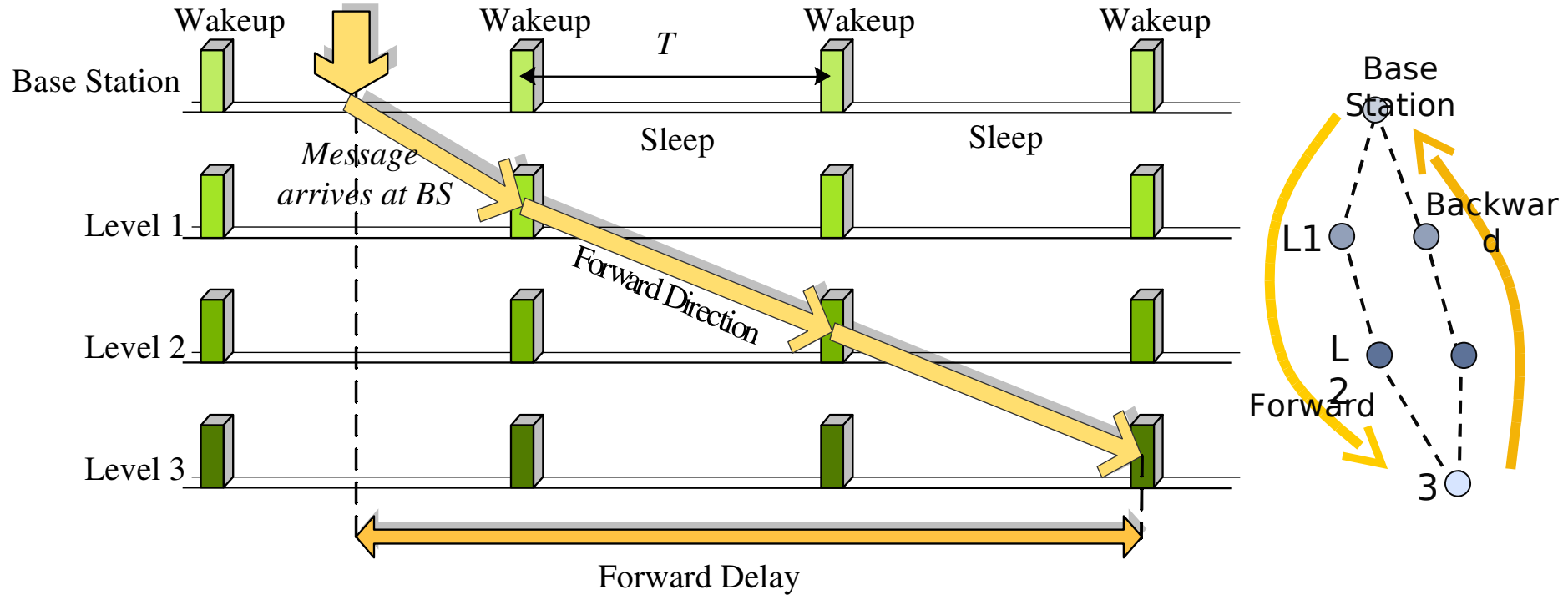
- Receiver: Wakeup periodically to check for any activity
  - Sample RSSI (received signal power)
  - If  $RSSI > \text{Threshold}$  stay in active mode to receive the message
  - Otherwise switch to low-power mode
- Sender: Awaken the receiver and send the message
  - Send a tone (at correct time) to awaken receiver
  - Send the message after the receiver is awakened
- **Trade-off between the delay and power**

# Traffic Model

- Two communication paths
  - Forward direction (downlink)
    - Commands/Queries
  - Backward direction (uplink)
    - Event notifications, Alarms
- Messages arrive at random time asynchronously
- **Delay**: time between generation of a message till its delivery at the destination
  - Worst delay depends on the maximum number of hops



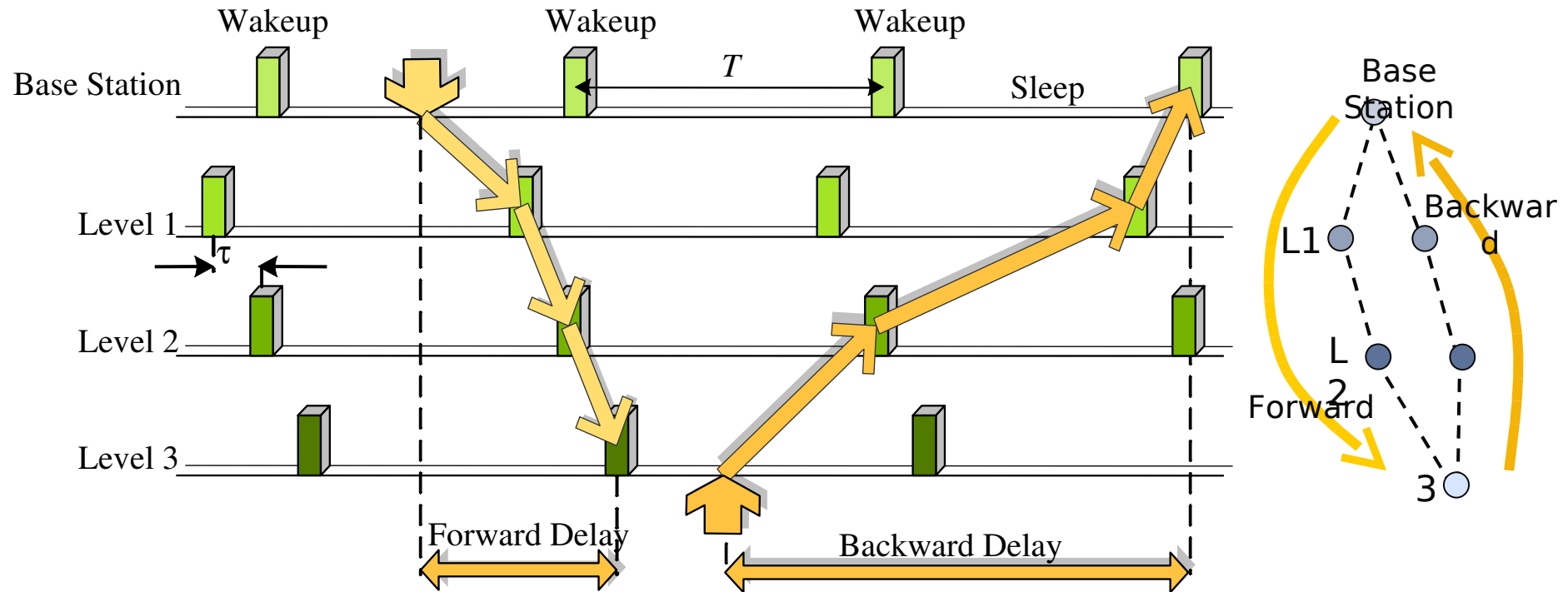
# Wakeup Patterns – Fully Synchronized



[ W. Ye, J. Heidmann, and D. Estrin 2002 ]

- Delay due to wakeup over multiple hops
- Simple but worst performance!

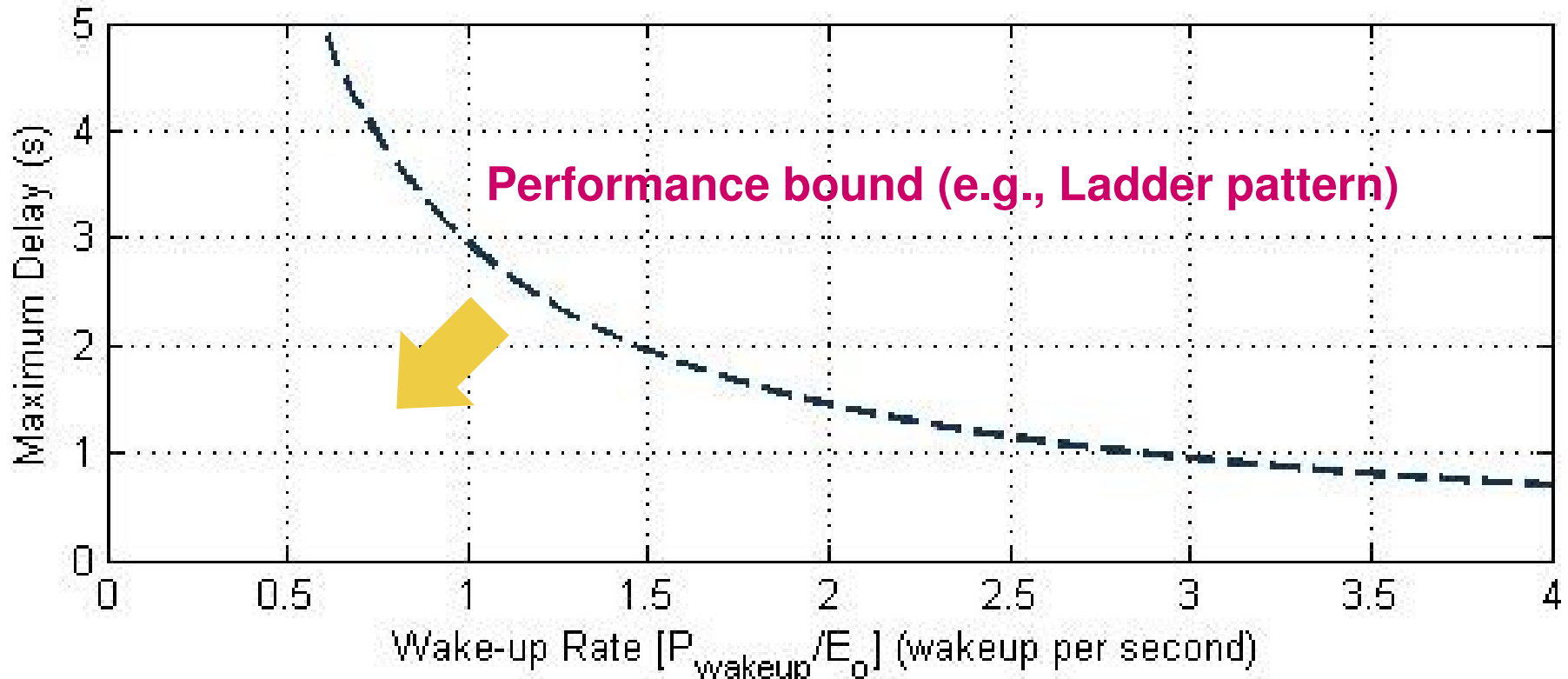
# Wakeup Patterns – Ladder pattern



[ G. Lu, B. Krishnamachari, and C. Raghavendra  
2004 ]

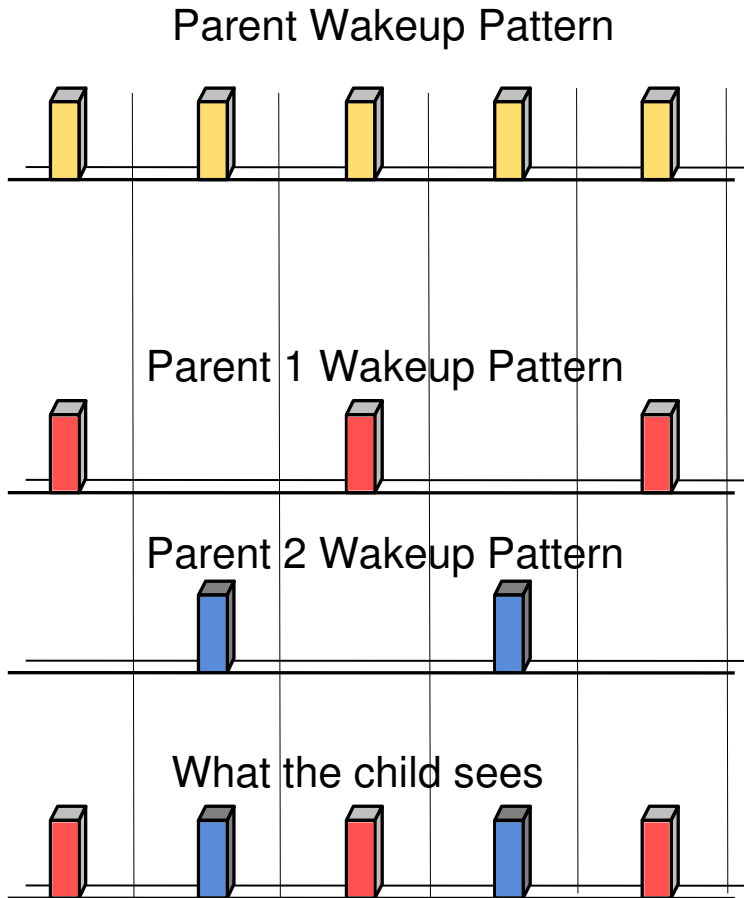
- Forward delay is significantly improved
- Backward delay is not so good

# Delay and Power Trade-off Curve

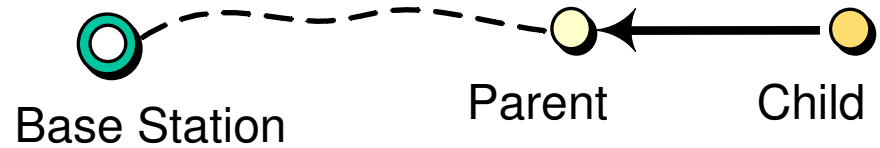


- How to improve the performance?

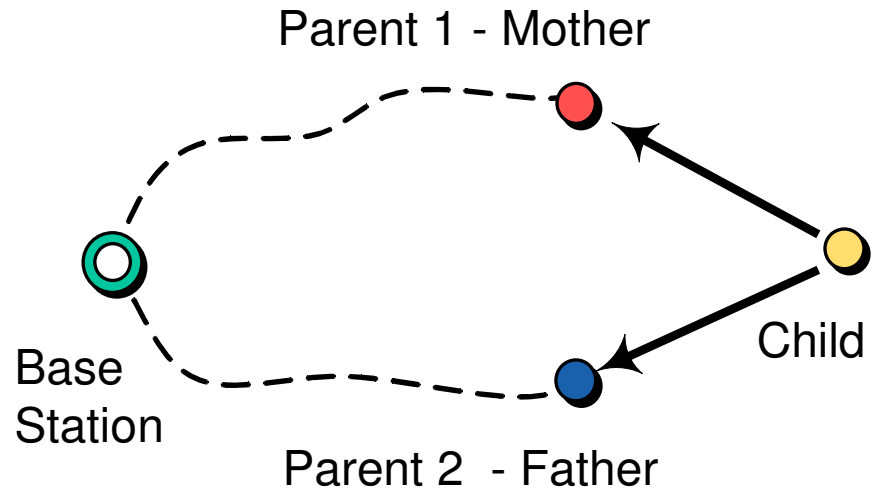
# Multi-Parent Method – Basic Idea



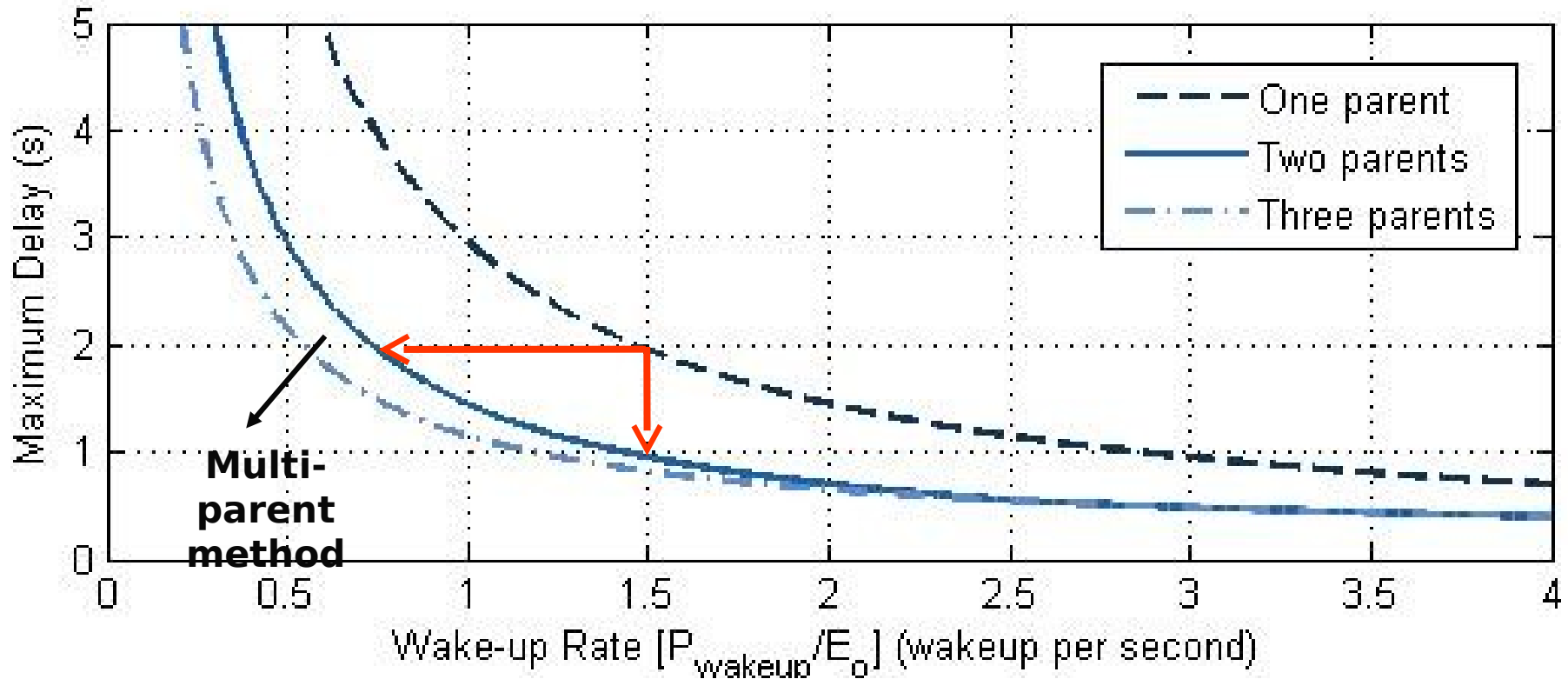
## Single-parent Case



## Multi-parent Case



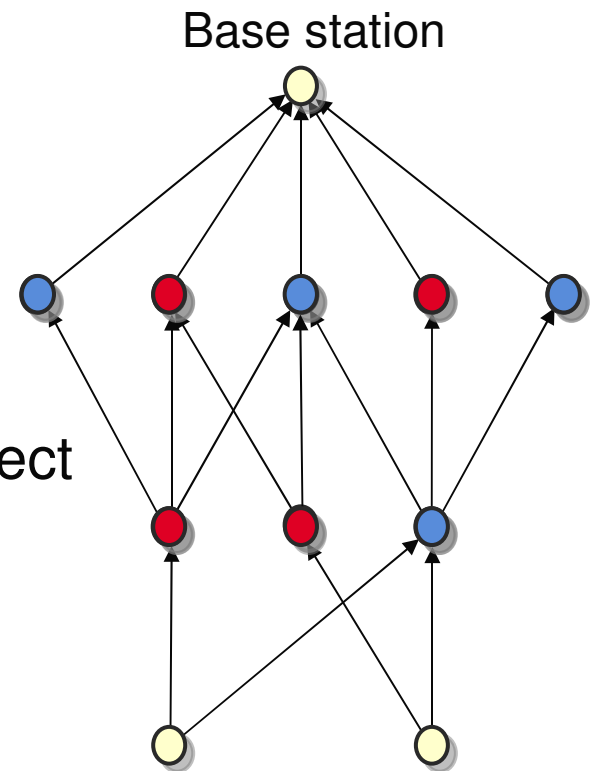
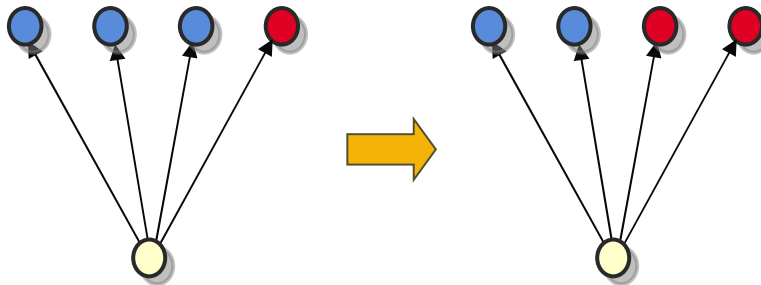
# Delay and Power Trade-off Curve



- Most of the benefit is from one parent to two parents
- How to assign the (two) parents?

# Graph Coloring Problem

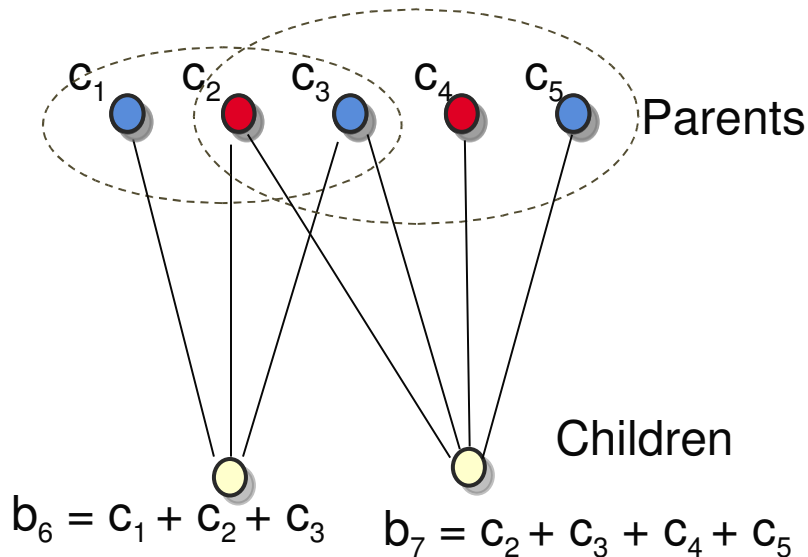
- Color all nodes in the network either red or blue
- ✓ Each node has at least one red parent and one blue parent
- Balance the number of red and blue parents of each node
  - Have balanced number of parents to select



- Easier to solve

# Graph Coloring Algorithm – Formulation

- Define a variable  $c_i$  for each node  $i$



$$c_i = \begin{cases} +1 & \text{node } i \text{ blue} \\ -1 & \text{node } i \text{ red} \end{cases}$$

$$b_i = \sum_{j=1}^N p_{ij} c_j$$

- $p_{ij} = 1$  if node  $j$  is a parent of node  $i$ , otherwise it is 0!

- ✓ If the parent colors of node  $i$  are balanced,  $b_i$  will be close to 0

# Graph Coloring Algorithm

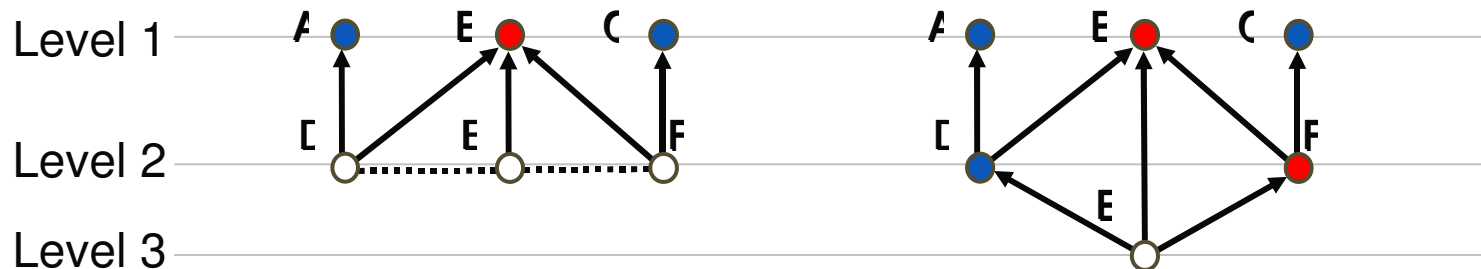
- Minimize the weighted sum of square of  $b_i$ s

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N'} w_i b_i^2 = c^T P^T W P c \\ \text{subject to} \quad & c_i^2 = 1, i = 1, \dots, N \end{aligned}$$

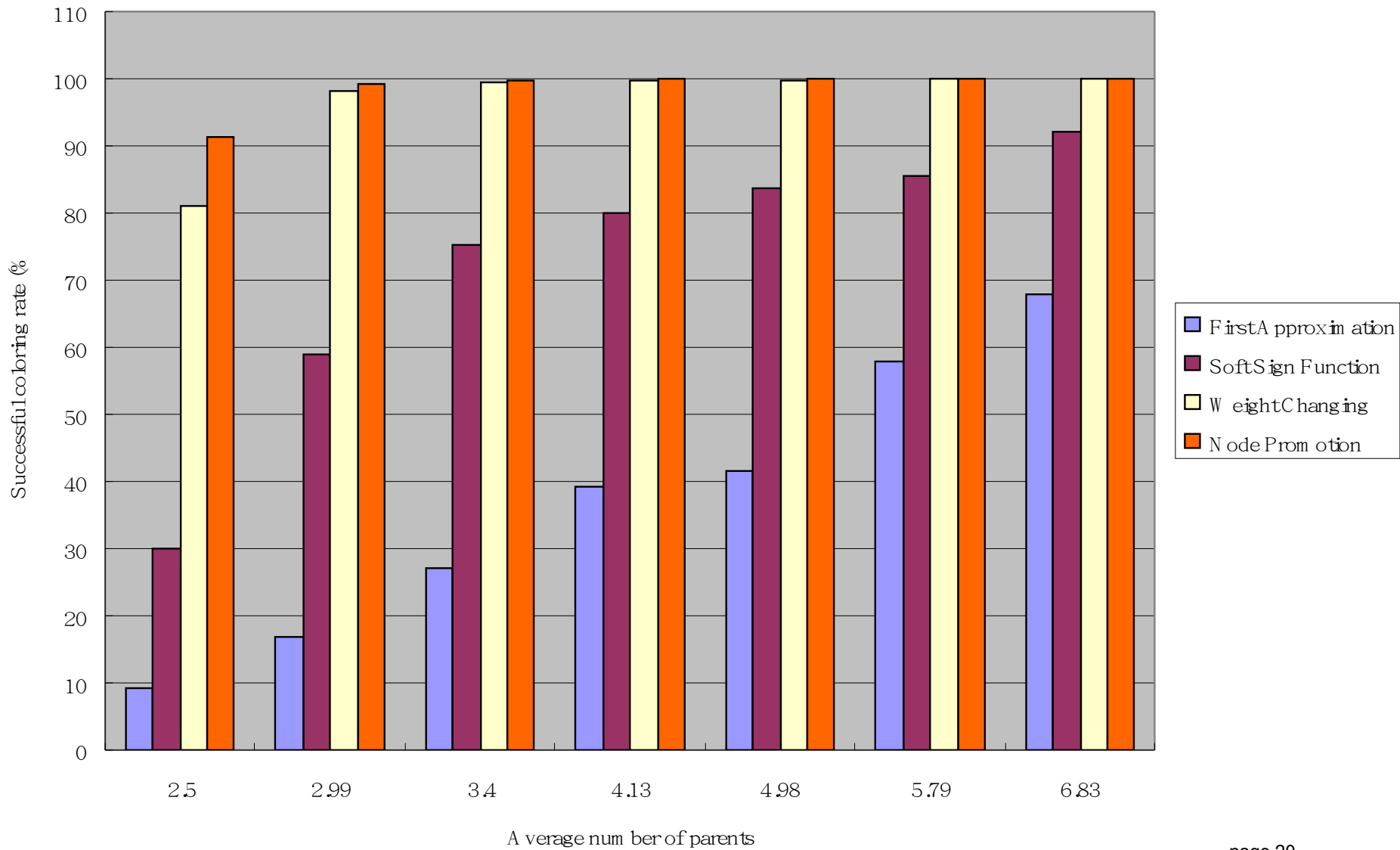


$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N'} w_i b_i^2 = c^T P^T W P c \\ \text{subject to} \quad & \sum_i c_i^2 = N \end{aligned}$$

- Heuristic:
  - Approximate the problem with relaxation
  - Refine the solution based on the value of  $c_i$
  - Assign weights to node and adaptively change the weights
  - Move the node to higher levels (e.g., level 2  $\rightarrow$  level 3)



# Graph Coloring Algorithm - Results



# Event-Driven Operation Summary

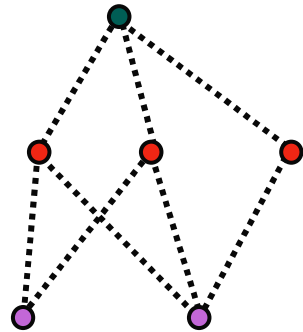
- Wake-up is an energy consuming process for many delay-sensitive applications
- Multi-parent method
  - Significantly improves the performance
  - Partitioning of the nodes (NP-complete)
- Heuristic coloring algorithm
  - Efficiently assigning the colors (at network initialization)
- Joint design of MAC and topology

# Outline

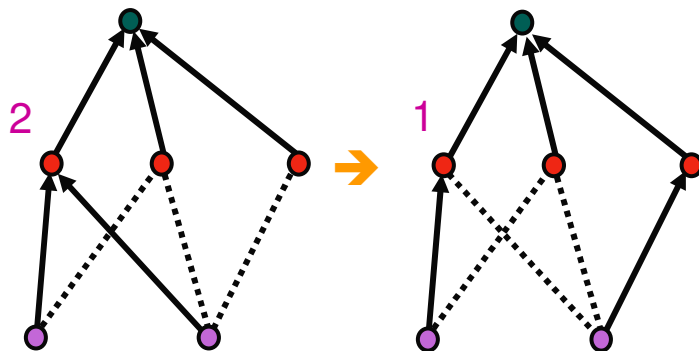
- Event-driven operation
- Time-driven operation
  - Periodic data aggregation
- Intelligent light control

# Data Aggregation Tree

Base Station



- Build tree for data aggregation
  - The energy cost (load) depends on the number of children
- Balance the load among nodes

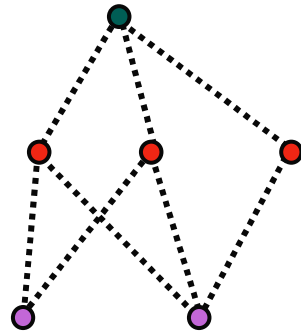


- ✓ Data centric routing  
[B. Krishnamachari, D. Estrin, and S. Wicker 2002]
- ✓ Maximize network lifetime  
[Y. Wu, S. Fahmy, and N. B. Shroff 2008]

Different trees gives different performance

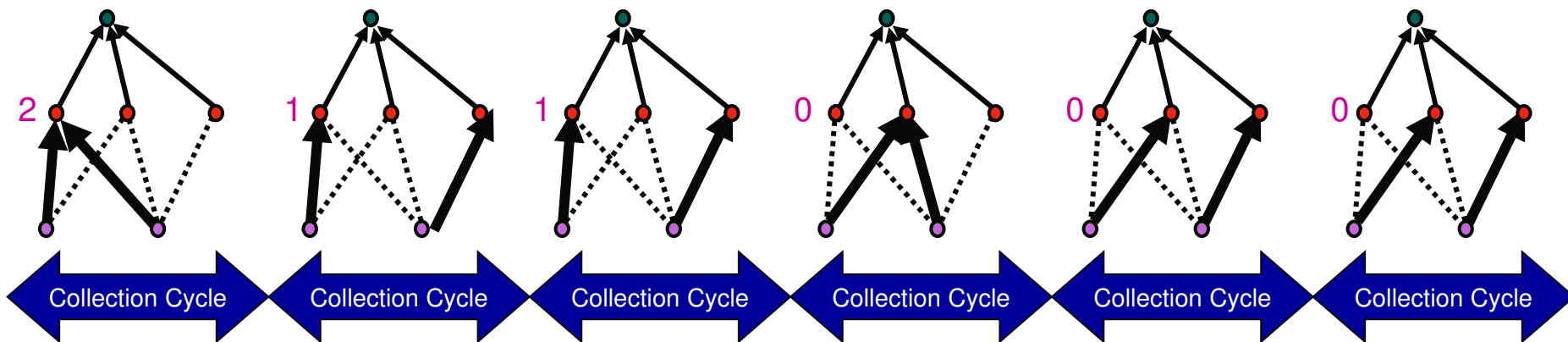
# Proposed Data Aggregation Trees

Base Station



➤ Balance the load over time

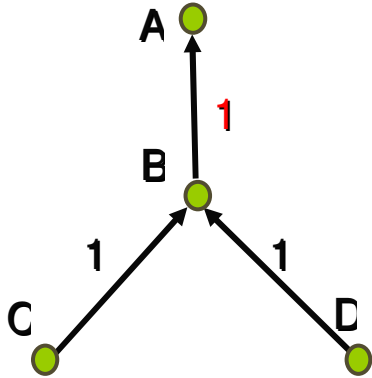
- Example (left node): First cycle 2 children, second and third cycles 1 child, last three cycles no child
- On average =  $(2+1+1)/6 = 2/3$  child



- How to construct the set of trees to maximize the lifetime?

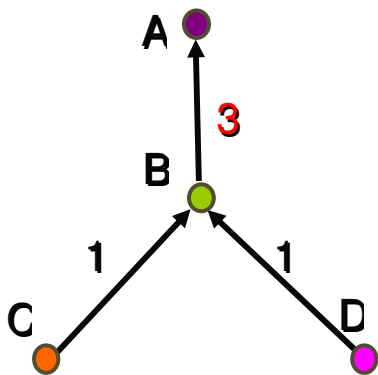
# Data Aggregation Models

- Before we talk about tree construction...



- ✓ All data can be combined
  - For example, network supervision
- An algorithm using a set of trees is proposed

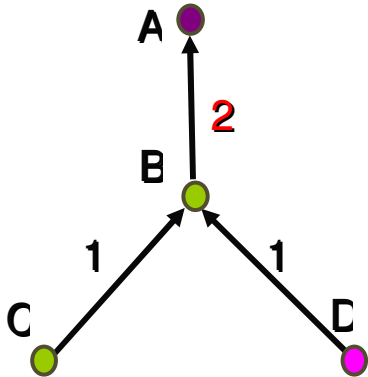
“Towards Energy-optimal and Reliable Data Collection via Collision-free Scheduling in Wireless Sensor Networks,” at Infocom 2008



- ✓ No data can be combined
- An algorithm based on DAG (Directed Acyclic Graph) structures is proposed

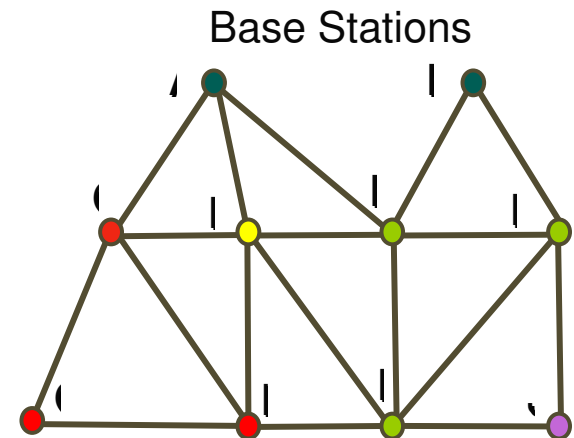
“Near Lifetime-Optimal Data Collection in Wireless Sensor Networks via Spatio-Temporal Load Balancing,” submitted to Trans. On Sensor Networks

# Data Aggregation and Network Models



- ✓ Only data from same group of nodes are combined
  - Most general (e.g., temperature in each room)
  - Considered in the tree construction (routing)

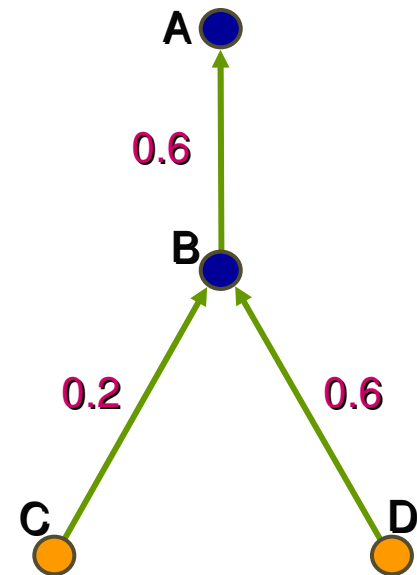
- Nodes are divided into groups
- Relay Nodes
  - May be used if it's more energy-efficient
- Example
  - Base Stations: A, B
  - Relay node: D
  - Source nodes in 3 groups: (C,G,H), (E,F,I), (J)



# Flow Calculation - Principles



- One unit of data from each sensor
- Flow conservation
  - All the data should be routed to the base stations
- Data fusion
  - Relay B:  $0.6 < 0.2 (C) + 0.6 (D)$
  - For the flows from the same group, only the “maximum flow” has impact on energy consumption:  $\max(0.2, 0.6)$
  - **Effective flows**  $x_{ij}$  on each link: Sum of the “maximum flow” from each group



# Flow Calculation - Formulation



- Formulated as a convex optimization problem
- Optimize the network lifetime

- $f_{ij}^n$  : flow from node  $i$  to  $j$ , generated by node  $n$
- $x_{ij}$  : effective flow from node  $i$  to  $j$
- $q$  : the maximum energy consumption
- $R_n$  : set of nodes that receive packet from node  $n$
- $S_n$  : set of nodes that send packet from node  $n$

minimize  $q$

subject to  $\sum_{j \in R_n} f_{nj}^n = 1, \forall n \in \mathcal{S},$

$$\sum_{j \in S_n} f_{jn}^n = 0, \forall n \in \mathcal{S},$$

$$\sum_{j \in R_i} f_{ij}^n = \sum_{j \in S_i} f_{ji}^n, \forall n \in \mathcal{S}, \forall i \in \mathcal{N} \setminus \{n\},$$

$$x_{ij} \geq \sum_{k=1}^g \max_{n \in G_k} (f_{ij}^n), \forall n \in \mathcal{S}, \forall i \in \mathcal{N}, \forall j \in R_i,$$

$$\sum_{j \in R_i} e_{ij}^t x_{ij} + \sum_{j \in S_i} e_{ji}^r x_{ji} \leq q, \forall i \in \mathcal{N},$$

$$f_{ij}^n \geq 0, \forall n \in \mathcal{S}, \forall i \in \mathcal{N}, \forall j \in R_i,$$

} Flow equations

} Energy equation

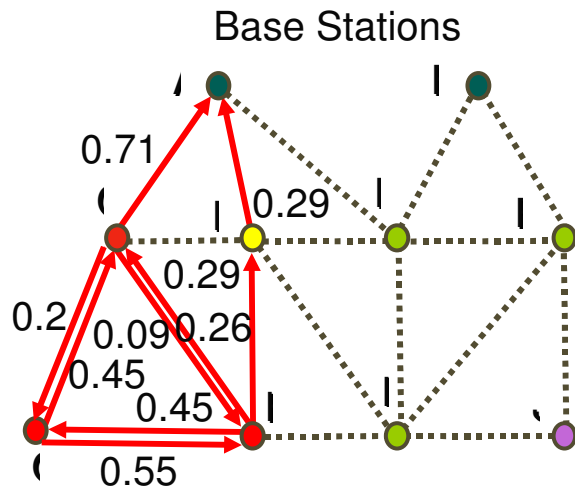
$e_{ij}^t$  : tx energy

$e_{ij}^r$  : rx energy

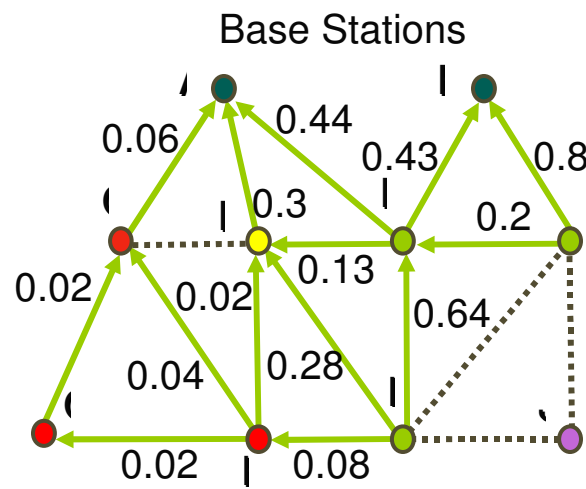
# Flow Calculation



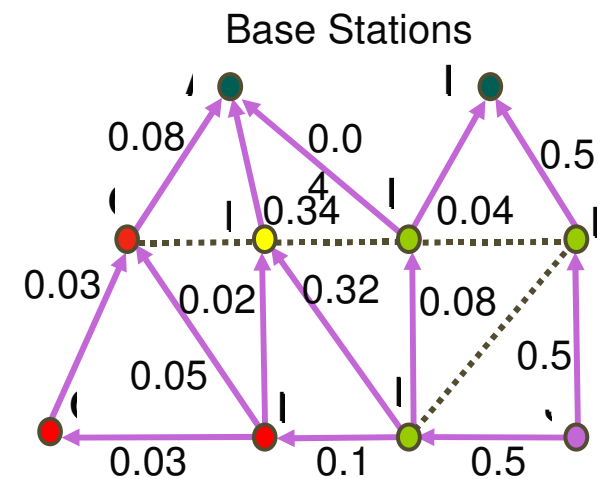
- We calculate the optimal flows and their directions on all the links
- Flows from different group are jointly optimized



Flows from group red



Flows from group green

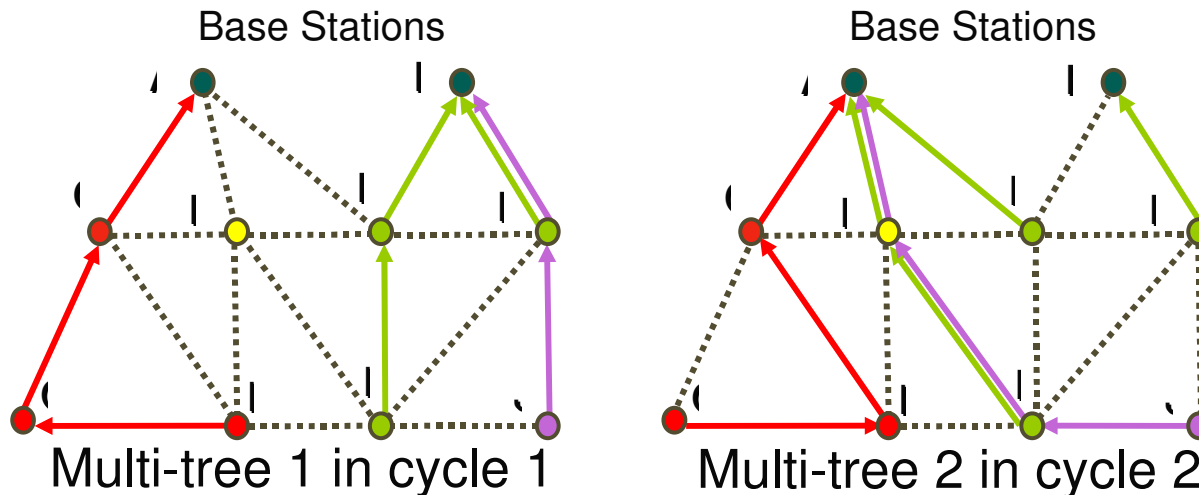


Flows from group purple

# Multi-tree Construction



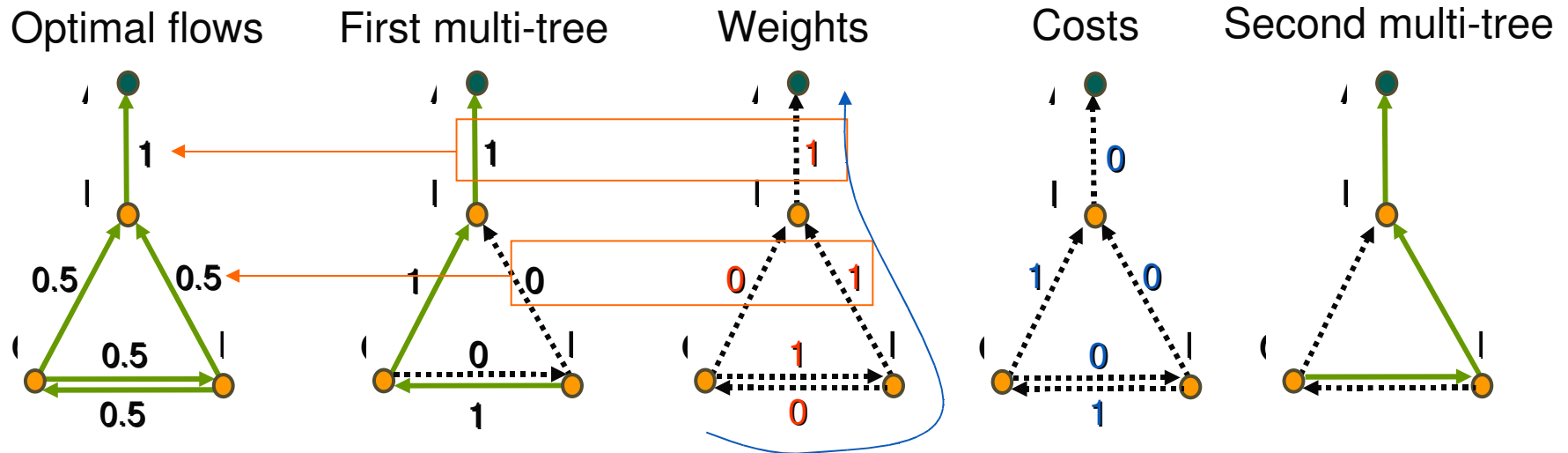
- In each cycle, we construct one multi-tree
- The energy consumption of nodes depending on the TX/RX pairs
  - Node G and H exchange roles in different cycles



# Multi-tree Construction – Basic Idea



- Goal: make the average link usage in all cycles close to the optimal flow
- Construct multi-trees one by one (iteration)
  - Given one multi-tree, how to construct the second multi-tree?



# Multi-tree Construction – Find the Weights



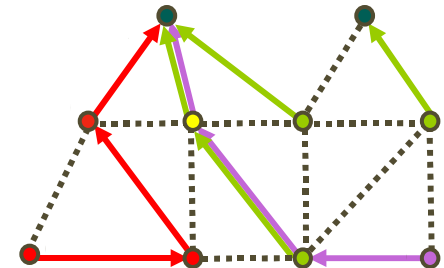
- Given  $m - 1$  multi-trees, find the weights for the  $m^{\text{th}}$  multi-tree
  - ✓ Count the number of use of each link for all the  $m - 1$  multi-trees
  - ✓ Find the weight via convex optimization formulation
  - ✓ Change the weight to cost for each link (e.g.,  $1 - w_{ij}$ )

$$\begin{aligned}
 &\text{minimize} && \sum_{k=1}^g \sum_{\forall j \in R_i, \forall i \in \mathcal{N}} \left\| \frac{1}{m} \left( w_{ij}(k) + \sum_{p=1}^{m-1} z_{ij}(k, p) \right) - \max_{n \in G_k} \left( \hat{f}_{ij}^n \right) \right\|_2^2 \\
 &&& + \sum_{\forall j \in R_i, \forall i \in \mathcal{N}} \left\| \sum_{k=1}^g \frac{1}{m} \left( w_{ij}(k) + \sum_{p=1}^{m-1} z_{ij}(k, p) \right) - \hat{x}_{ij} \right\|_2^2 \\
 &\text{subject to} && \sum_{j \in R_i} w_{ij}(k) = 1, \quad k = 1, \dots, g, \quad \forall i \in \mathcal{S} \cap G_k, \\
 &&& \sum_{j \in R_i} w_{ij}(k) = \sum_{j \in S_i} w_{ji}(k), \quad k = 1, \dots, g, \quad \forall i \in \mathcal{N} \setminus (\mathcal{S} \cap G_k), \\
 &&& w_{ij}(k) + w_{ji}(k) \leq 1, \quad j \in R_i \text{ and } i \in R_j, \quad k = 1, \dots, g, \\
 &&& 0 \leq w_{ij}(k) \leq 1, \quad \forall i \in \mathcal{N}, \forall j \in R_i, \quad k = 1, \dots, g,
 \end{aligned}$$

# Multi-tree Construction – Algorithm



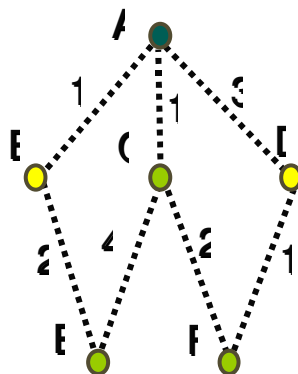
- Given the **cost** on each link, we construct multi-tree
- Each multi-tree contains several Steiner trees
  - A tree only connects some nodes with minimum cost
- The Takahashi and Matsuyama Algorithm



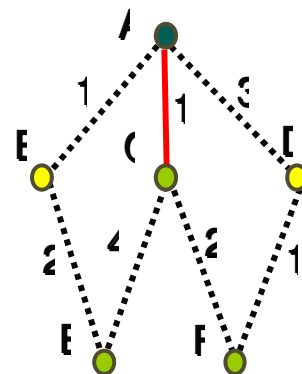
Base station:  
A

Relay nodes:  
B, D

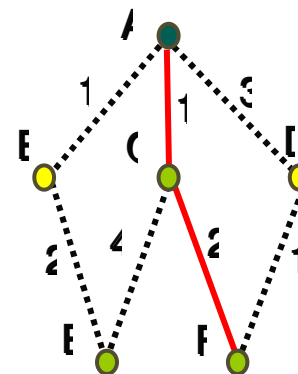
Source nodes:  
C, E, F



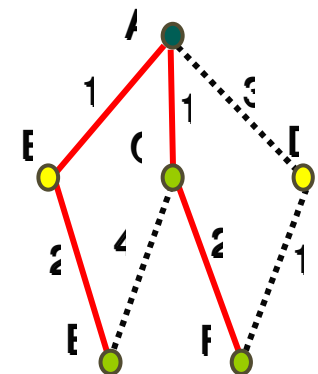
Step 1



Step 2

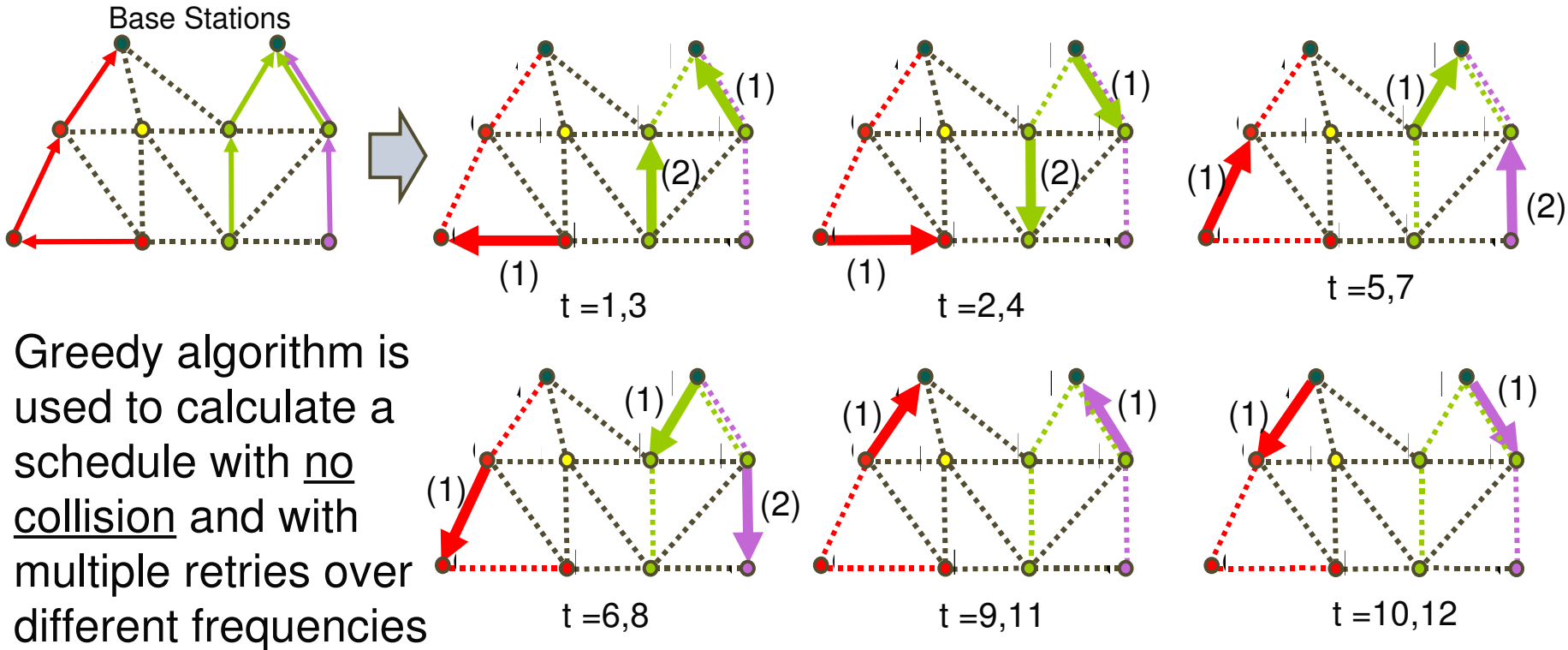


Step 3



Step 4

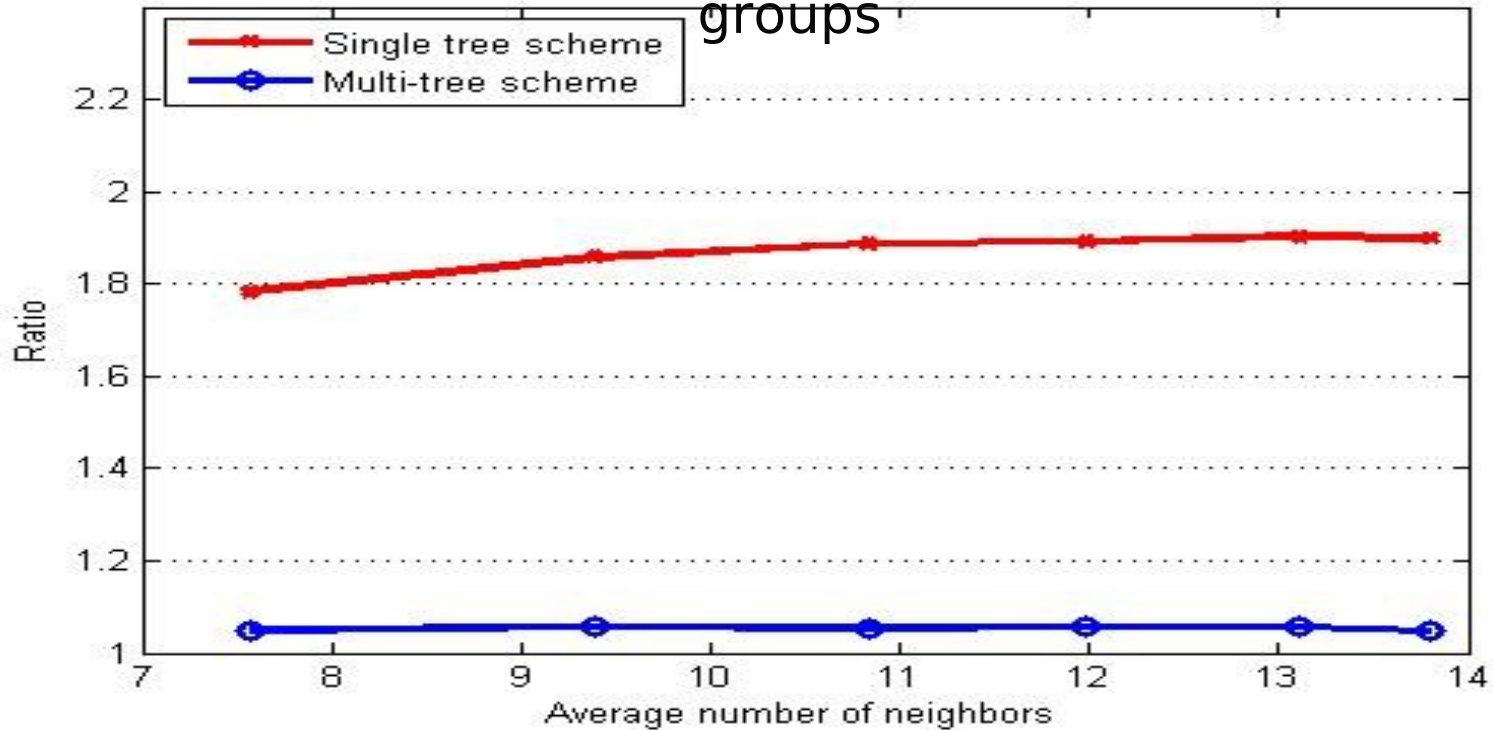
# Scheduling



- Greedy algorithm is used to calculate a schedule with no collision and with multiple retries over different frequencies

# Performance Analysis

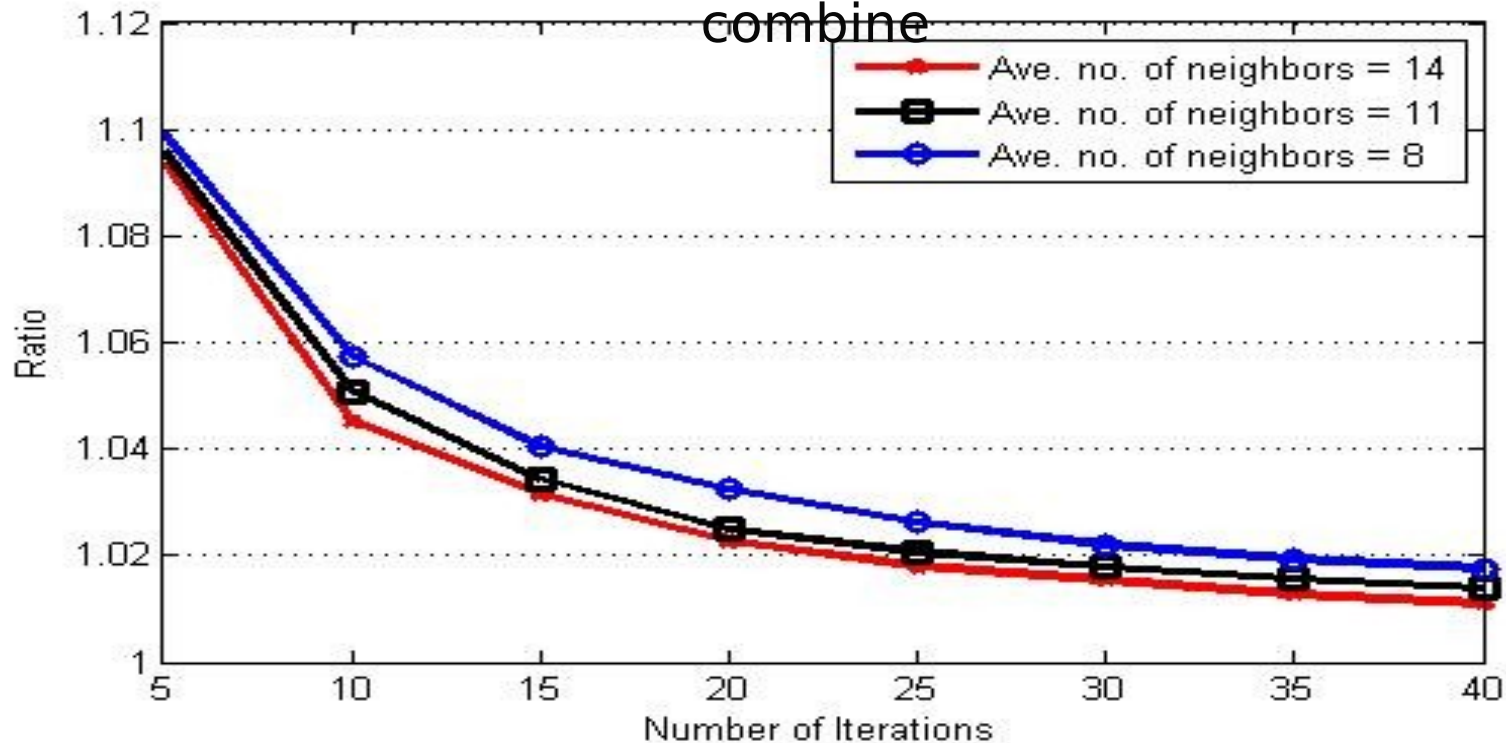
Number of nodes = 45, 10% relay nodes, 3 groups



- Ratio of the real lifetime to the theoretical optimal value
- Multi-tree structure gives much better performance

# Performance Analysis

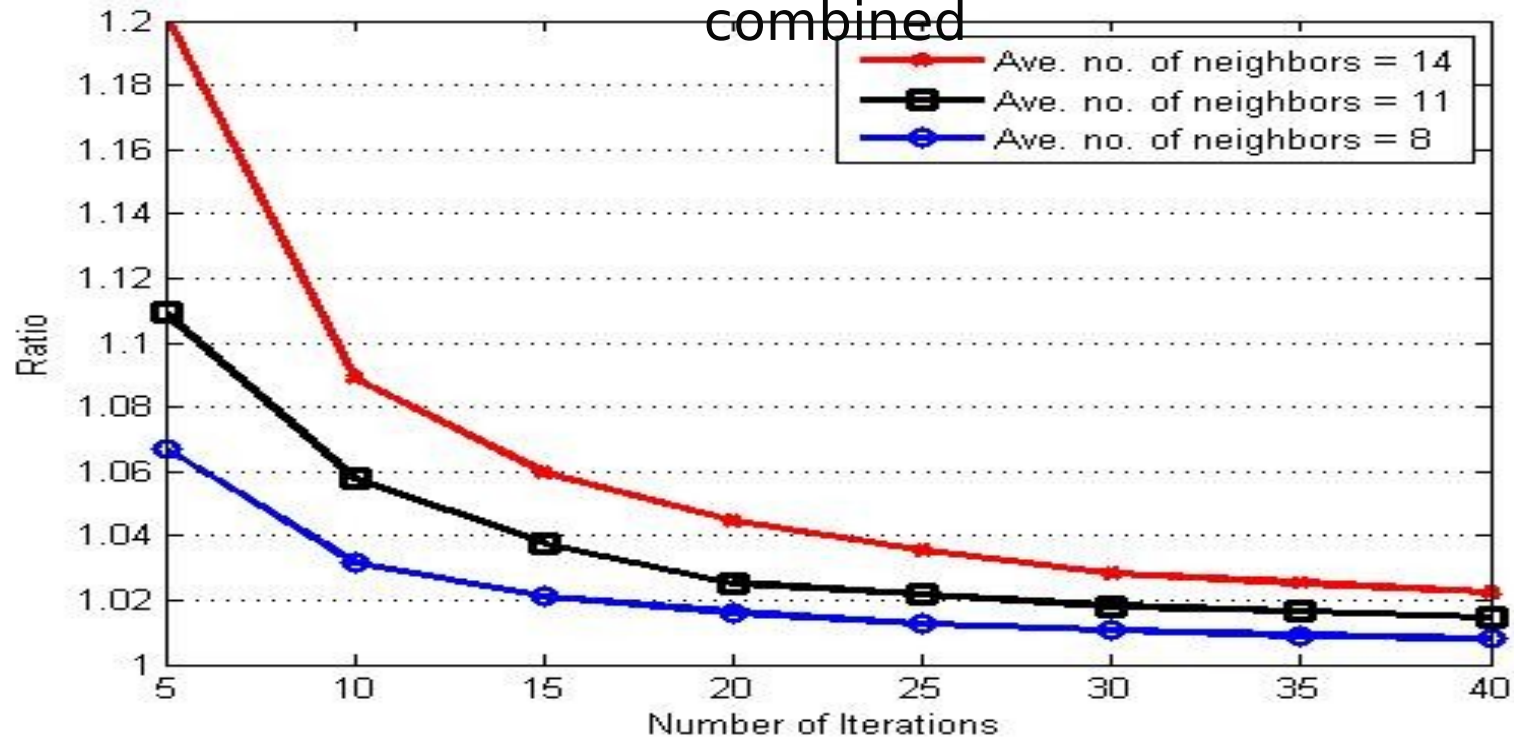
Number of nodes = 40, No relay node, All data are combine



- Obtaining near-optimal performance with around 35 multi-trees

# Performance Analysis

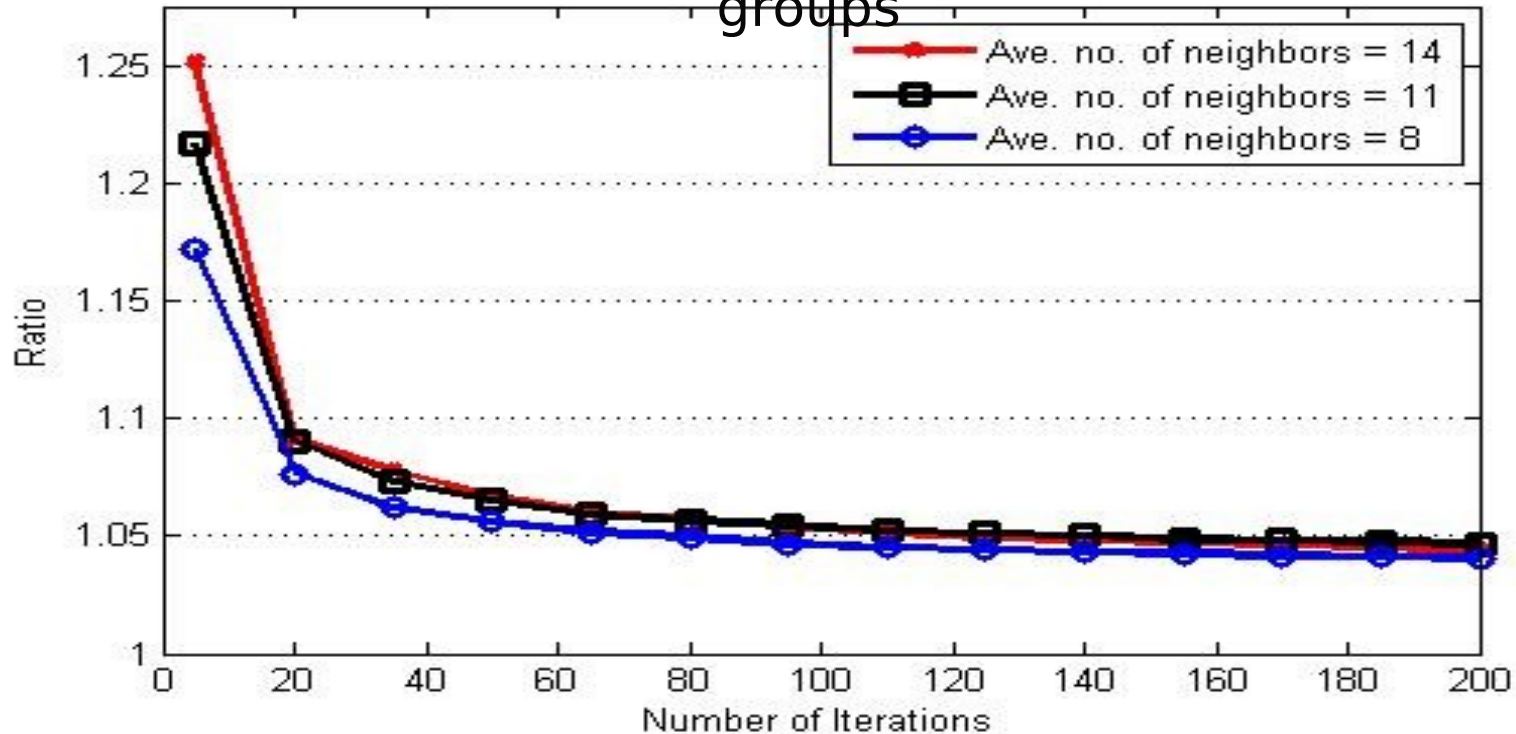
Number of nodes = 50, No relay node, No data are combined



- Obtaining near-optimal performance with around 35 multi-trees

# Performance Analysis

Number of nodes = 45, 10% relay nodes, 3 groups



- NP-complete problem solved by heuristic algorithm
- Obtain 5%-to-optimal performance with around 100 multi-trees

# Time-driven Operation Summary

- Data gathering is one of the most common tasks
  - Very energy-consuming
- Near energy-optimal and practical data fusion algorithm
  - Data aggregation models
  - Flow calculation / Multi-tree construction / Scheduling
- By changing the aggregation trees over time, we can balance the traffic load and achieve near-optimal lifetime
- Joint design of MAC and routing

# Outline

- Event-driven operation
- Time-driven operation
- **Intelligent light control**

# Light Control Scenario

- Wireless nodes are deployed over a floor
- Light control based on camera observations

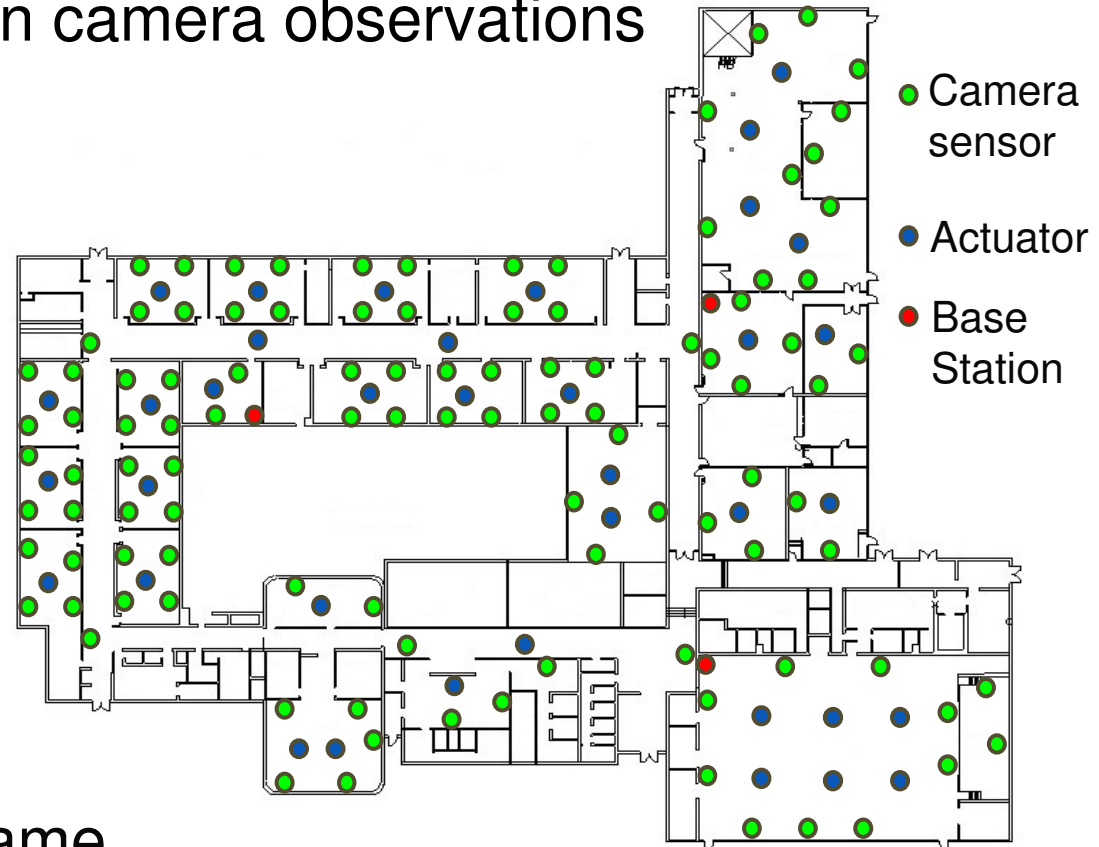
- Goal:

- ✓ Comfort light setting
- ✓ Energy efficiency

- Camera sensors:

- ✓ Rich information
- ✓ Multiple functions

- Only data from the same room can be combined



# Light Control System

## Camera nodes

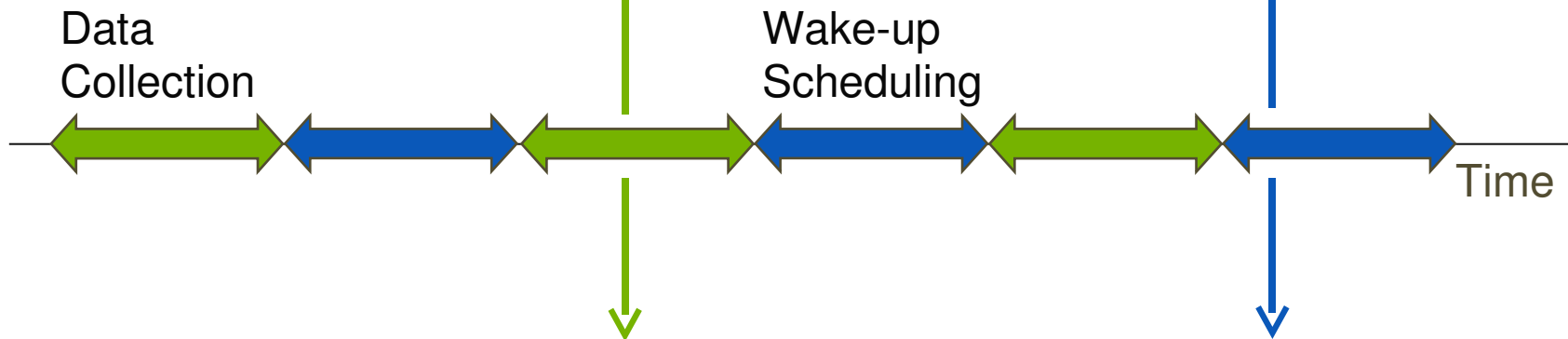
- Making observations
- Local processing (Region of interest)

## Base stations

- Occupancy sensing
- Activity analysis
- Optimization

## Actuators

- Light control

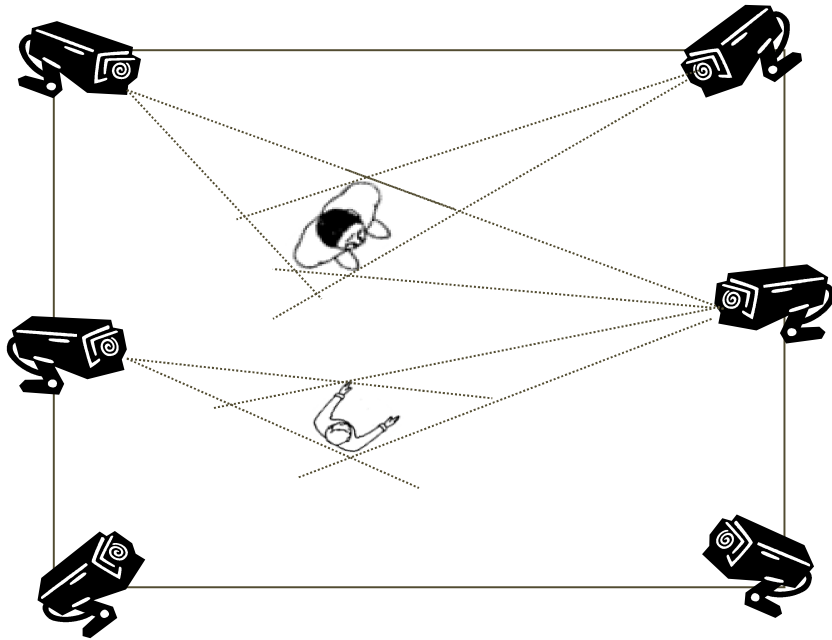


- Camera sensors regularly send processed observations to the base stations

- Base stations send commands to controllers when the light setting has to be changed

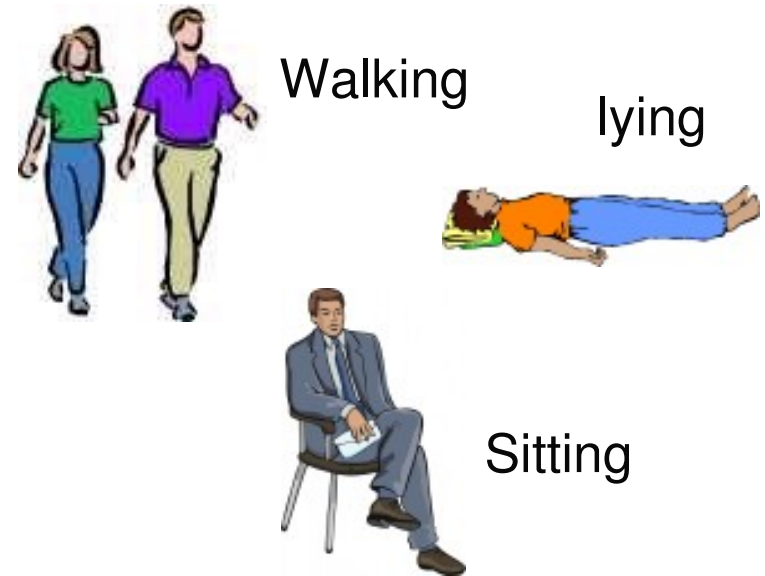
# Vision Analysis

## Occupancy Sensing



[ A. Ercan, A. El Gamal, and L. Guibas 2003 ]  
 [ D. Yang, H. Gonzalez-Banos, and L. Guibas 2003 ]

## Activity Analysis

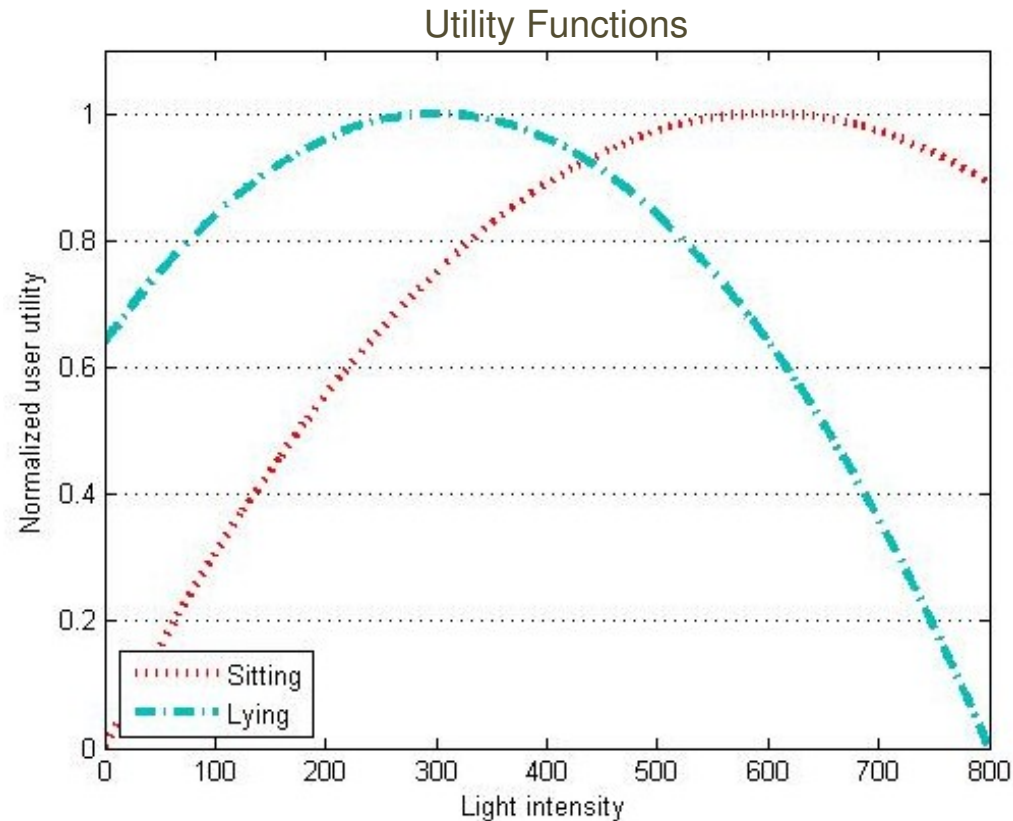


[ W. Chen and H. Aghajan 2009 ]

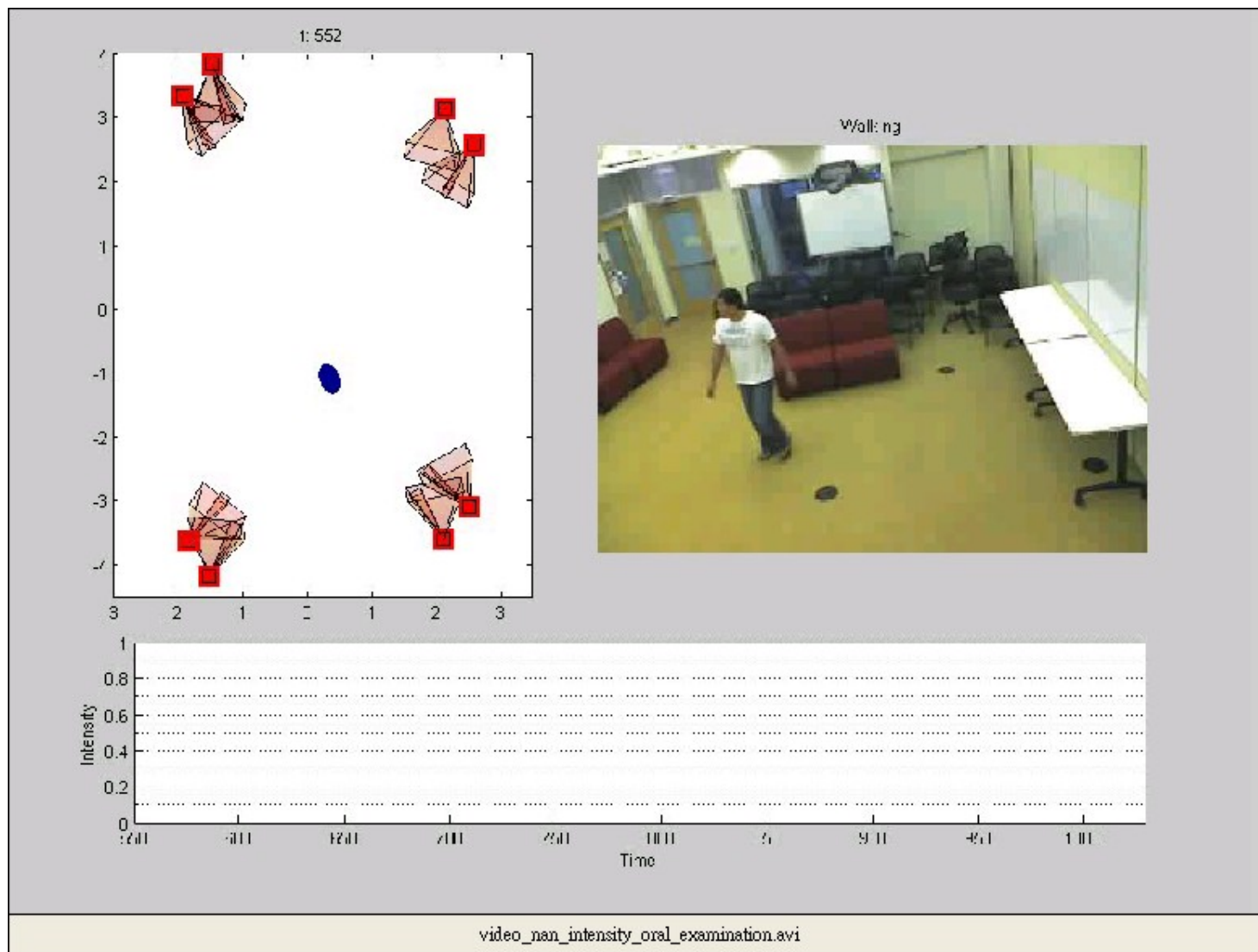
- Base station obtains the user location and activity

# Optimization Formulation

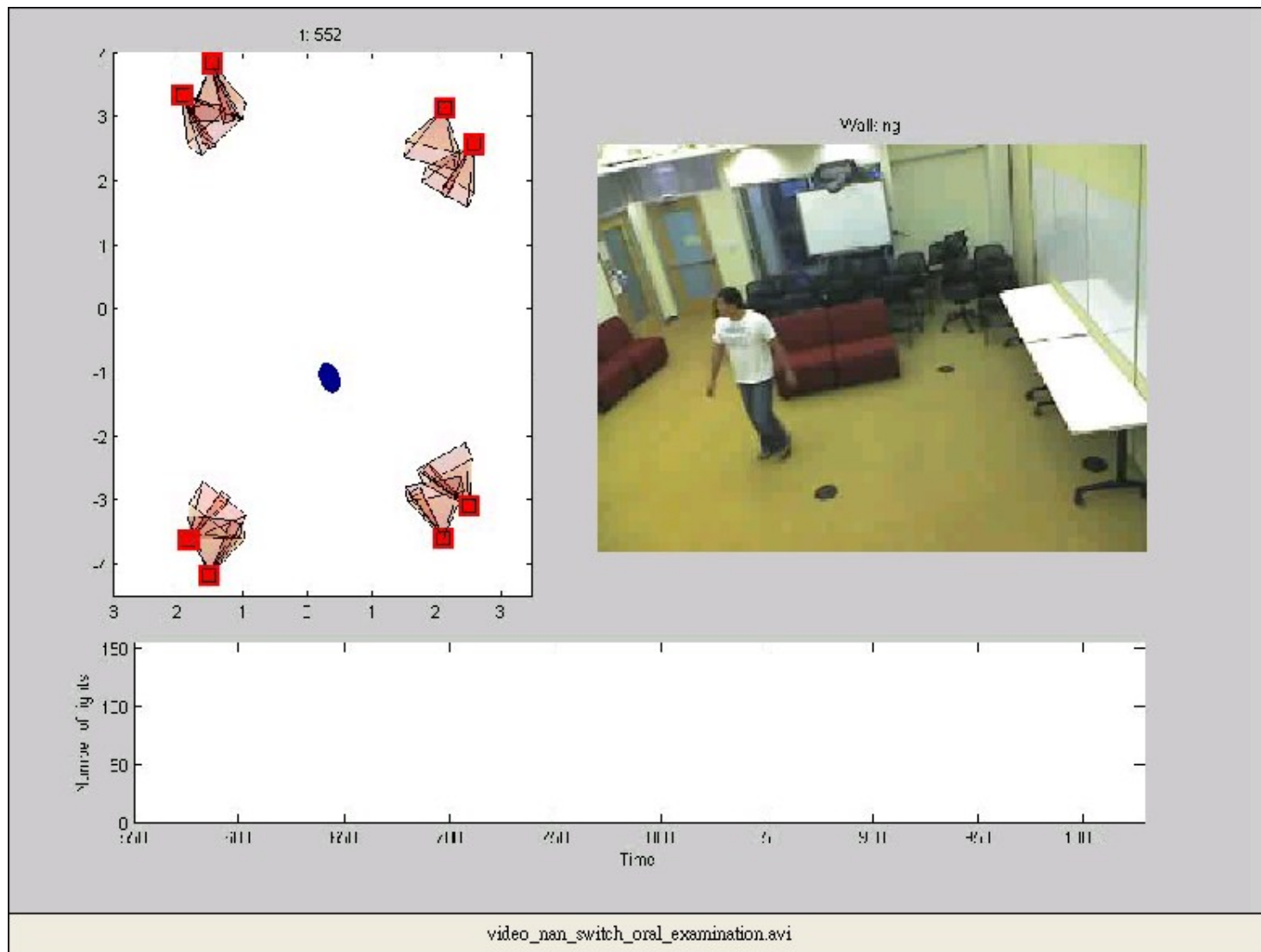
- Energy vs. Satisfaction
  - Utility Functions
    - Walking, sitting, lying
    - Obtained from learning
- [ V. Singhvi, C. E. Dept, J. H. Garrett 2005 ]
- Intensity control
  - Switch control



# Simulations: Intensity Control



# Simulations: Switch Control



# Light Control Summary

- A camera sensor network for light control
  
- Operations
  - Data collection cycle
  - Wake-up scheduling cycle
  
- Energy and satisfaction Trade-off
  - Utility functions
  - Intensity and switch control
  
- Simulations with artificial lights

# Conclusions

- Wireless sensor networks for smart environment
- Event-driven operation
  - Multi-parent method in wake-up scheduling
  - Improve the energy and delay trade-off
  - Efficient graph coloring algorithm
- Time-driven operation
  - Data aggregation models
  - Near lifetime-optimal and practical data aggregation
  - Dynamic data aggregation trees
- Light control using camera sensors

# Q & A

