A Distributed Algorithmic Framework for Coverage Problems in Wireless Sensor Networks

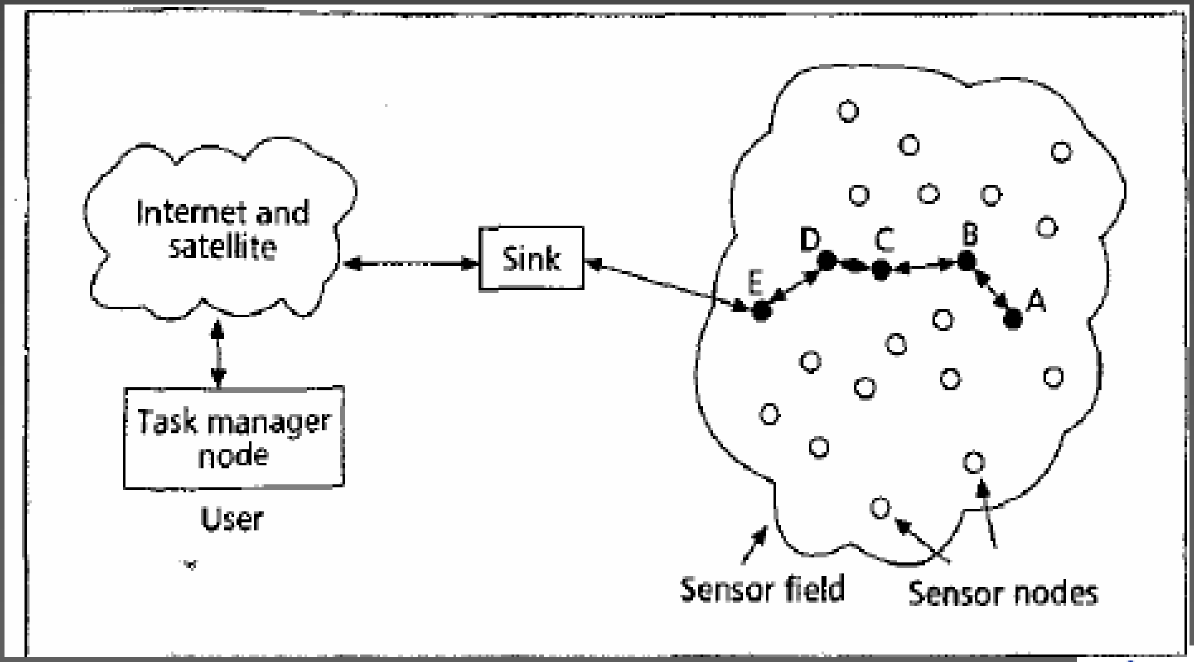
Akshaye Dhawan and Sushil K. Prasad

APCM, IPDPS ‘08

Introduction

- Sensor Networks : Consist of a large number of low cost sensor nodes connected to one or more sinks
- Dense
- Constraints:

  Energy.



Introduction

- The fact that sensors are deployed in large numbers means that there is significant overlap of their monitoring regions
- *Idea:* Use a subset of these sensors
- Shuffle this active set periodically

Max Lifetime Problem

- Come up with a sleep-sense schedule for all sensors such that the lifetime of the network is maximized while maintaining coverage of all targets.
- Shown to be a NP-complete problem [5]

Existing work

- Centralized approaches are typically based on Linear Programming [3, 5].
- Distributed algorithms are based on rounds where at the beginning of a round a sensor negotiates with its neighbors usually on some simple greedy criteria.
- Greedy criteria: Choose sensor with largest battery, largest number of uncovered targets etc.

Contributions

- **Main idea**: Instead of just looking at greedy solutions, look at how solutions effect each other
- Generalizes our work in HiPC'07 [12].
- Applies it to other coverage problems
- Expresses it as a framework that can possibly used on other network/graph problems that exhibit a similar structure

Definitions

- $G=(V,E)$ $V=\{s_1, s_2, \ldots, s_n\}$
- $e=(s_i,s_j) \in E$ iff s_i is in communication range of s_j
- $b(s)$: Battery available at sensor s
- Cover C: minimal subset of sensors that meets coverage objective (target/area)
- Lifetime of a cover $lt(C)$: $\min_{s \in C} b(s)$
- s is called the *bottleneck*

Framework

- **Step 1: Applicability**
- So what problems can we use this on?
- Must have a property that compatible local solutions when combined give a globally feasible solution.

Framework

- **Eg: Coverage**
- When covering targets, the local problem is to cover local targets.
- All local solutions combined -> all global targets covered
- Same is true of covering local area.
- Eg Cannot -> MST problem

Framework

- **Step 2: Modeling local solutions and their interdependencies**
- Model local solutions: for coverage these are all the local covers (for local targets/area)
- May be possible to represent entire space
- Account for the dependency between local solutions - Envisioned graph model: *Dependency Graph*

Lifetime Dependency Graph

G = (V, E) where nodes in V are local covers and edges in E are sets of common sensors between pairs of nodes, that is, an edge e between two covers C_1 and C_2 is $C_1 \cap C_2$. Also,

$w(\mathrm{e}) = \min_{s \in C1 \cap C2} b(s)$, the weight of an edge e.

$d(C) = \sum_{e \in E \text{ and incident to } C} w(e)$, the degree of a cover C.

Framework

- **Step 3 – Prioritize local solutions**
- Decide which local solutions to use
- Define a priority function that takes into account properties of the *dependency graph*

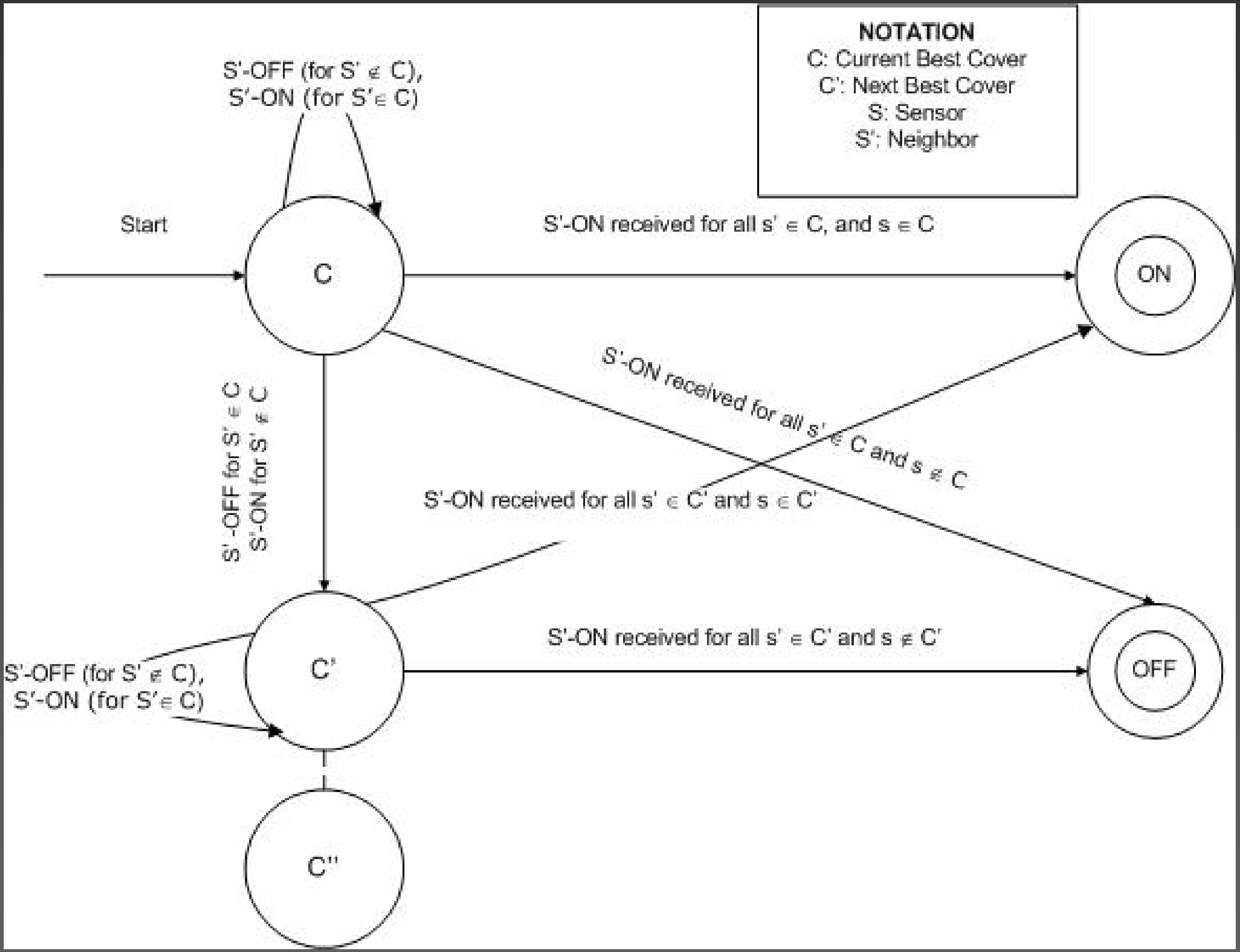
Framework

- Priority function for LD Graph
- *Lower d(C)* – means a smaller impact on other covers
- Degree same then other criteria like battery, remaining sensors, id can be used to break ties
- Some other improvements possible [12]

Framework

- **Step 4: Negotiate with neighbors**
- *Setup* – exchange information with 1|2 hop neighbors, construct dependency graph and prioritize
- Negotiation involves taking neighbors and your preferences into account to decide which local solution to use.

For coverage



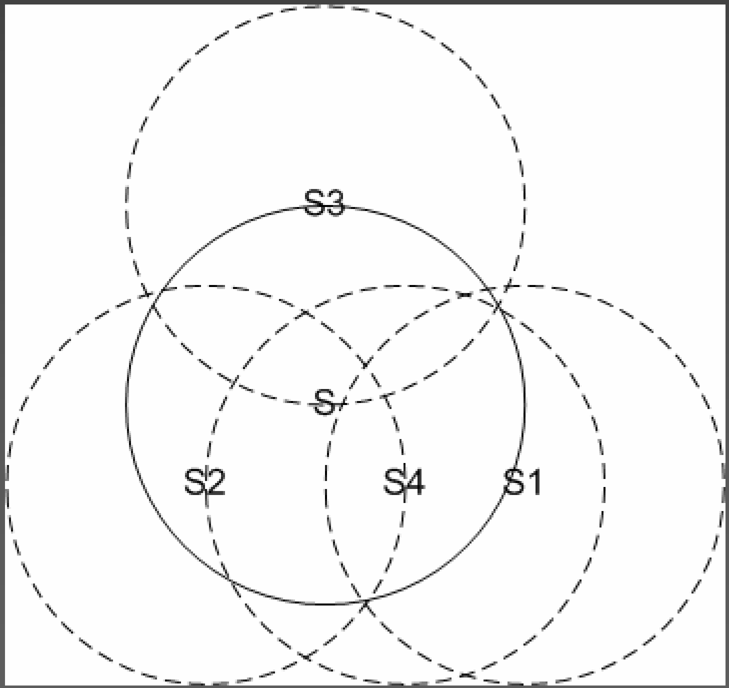
Target coverage formulation

- Initial Setup: Communicate with neighbors b(s), T(s). Using this find N(s,1) for T(s). Construct LD graph for these covers and calculate the degree
- Negotiate using automata, where each cover C is a local cover that covers all local targets

Area Coverage

- Sponsored area computed
- Certain part of a sensors disk covered by its neighbors [3].
- Use to compute covers.
- Eg. $\{s\}$, $\{s_1,s_2,s_3\}$,

  $\{s_2, s_3, s_4\}$

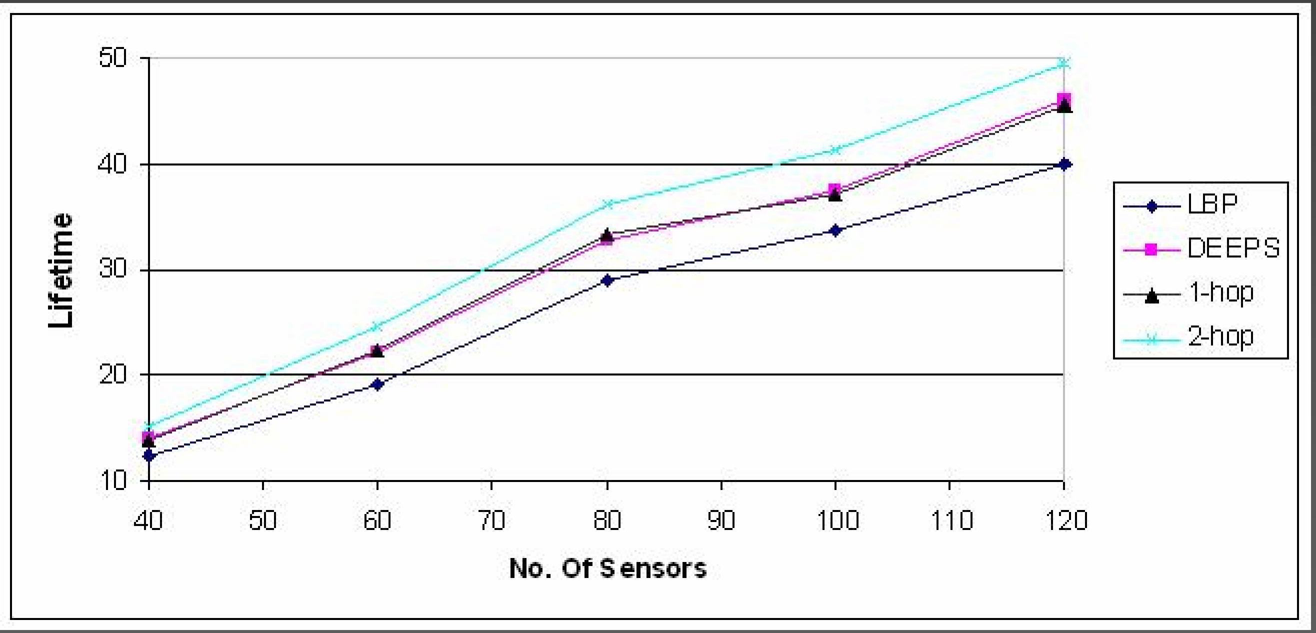


Alternative formulation

- A field as a set of points that are covered by the same set of sensors.
- Discretize the area into a grid
- Once points have been grouped into fields, all that is needed is to ensure that all fields are covered.
- Hence, each field corresponds to a virtual target and the problem can effectively be reduced to that of target coverage.

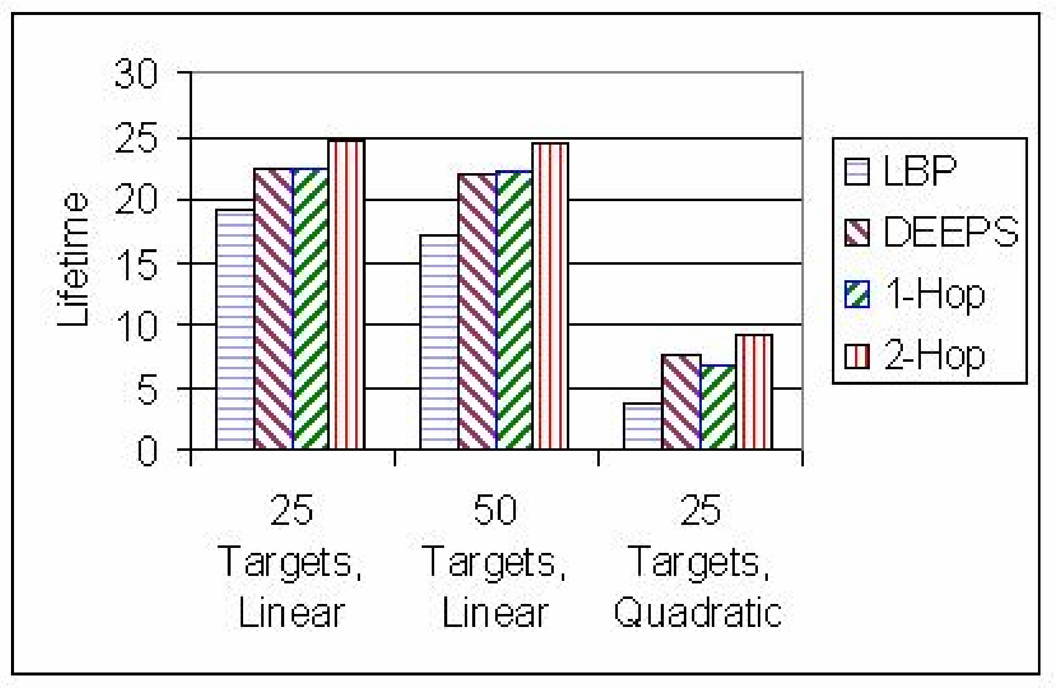
Results

- 25 targets, Linear energy



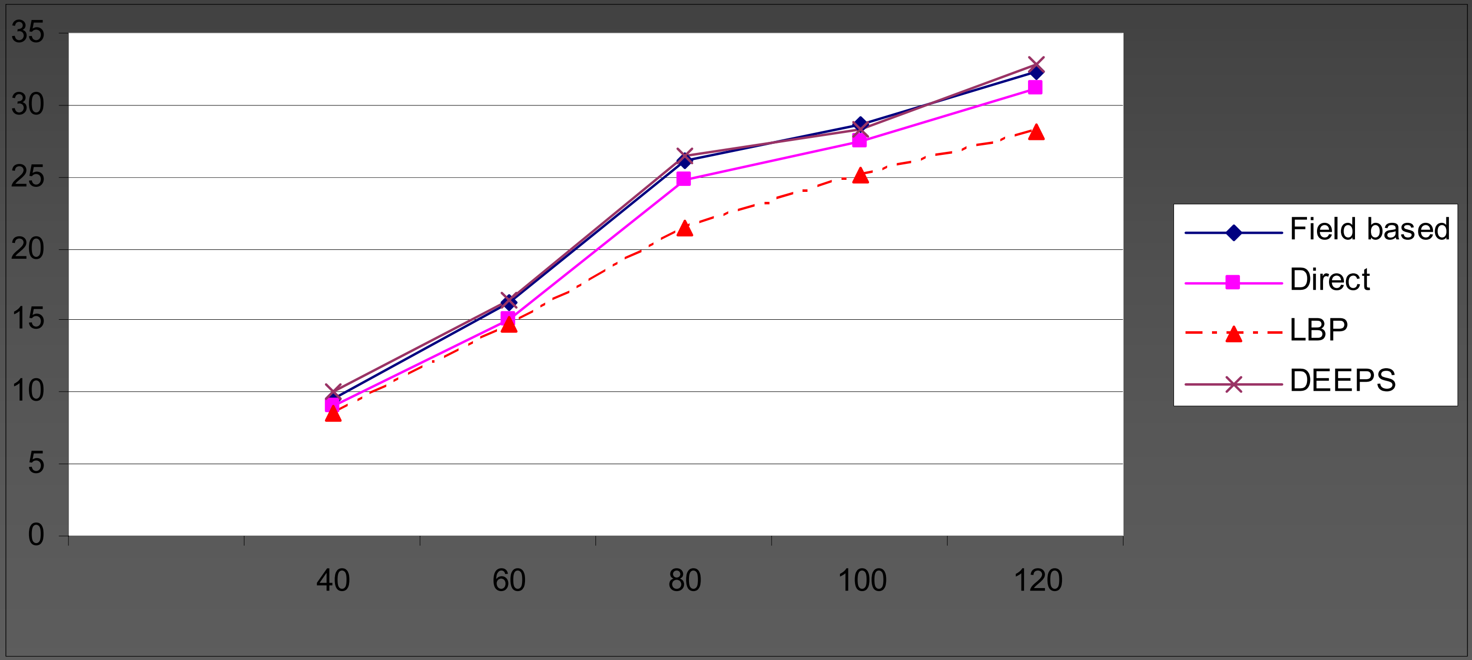
Target coverage

- 60 sensors. Energy, target varying.



Area cover

- Linear Energy



Conclusion

- New framework for developing heuristics
- *Dependency* Graph formulation presented to model interdependencies between local solutions
- Gives an insight into the problem structure instead of focusing on purely greedy approaches

Questions?