A clustering method for wireless sensor networks

S. Fouchal[1] Q. Monnet[2] D. Mansouri[3]
L. Mokdad[2] M. Ioualalen[3]

[1]LSIIT (BFO) University of Strasbourg (FR)

[2]LACL University of Paris-Est (FR)

[3]LSI USTHB (DZ)

Context

New clustering algorithm

- General method to cluster any type of data
- **FFUCA** (***F****ast and* ***F****lexible* ***U****nsupervised* ***C****lustering* ***A****lgorithm based on ultrametric properties*)

applied to

Wireless Sensor Networks (WSNs)

- WSNs: frequent use of clustering
- purpose: better results than existing algorithms

Outline

Wireless Sensor Networks (WSNs)

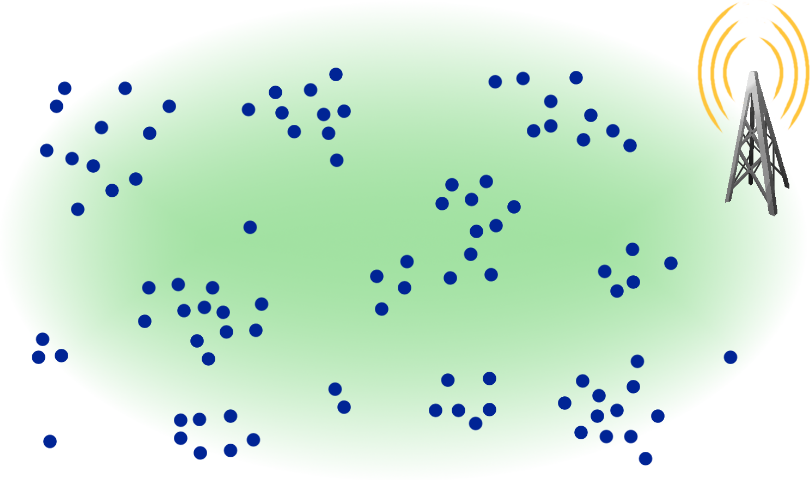
Small devices

- realize measurements (sensors)
- ad-hoc communication
- linked to a base station (BS)

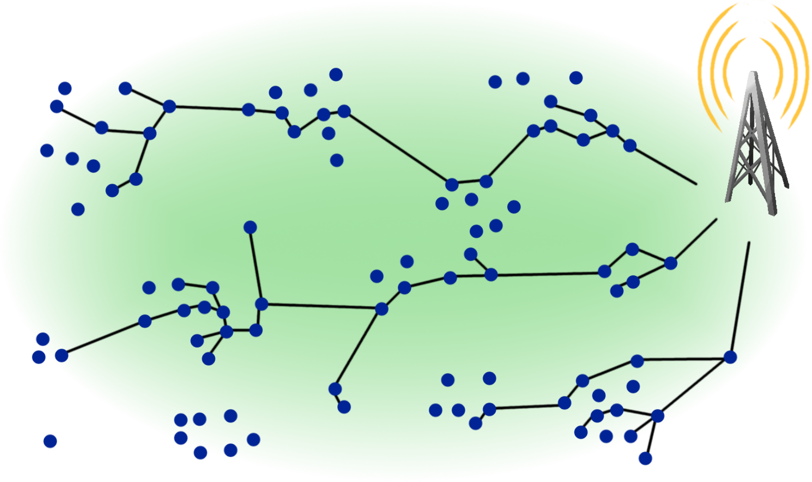
Restricted resources

- few computation capabilities
- few memory available
- few energy available (battery)

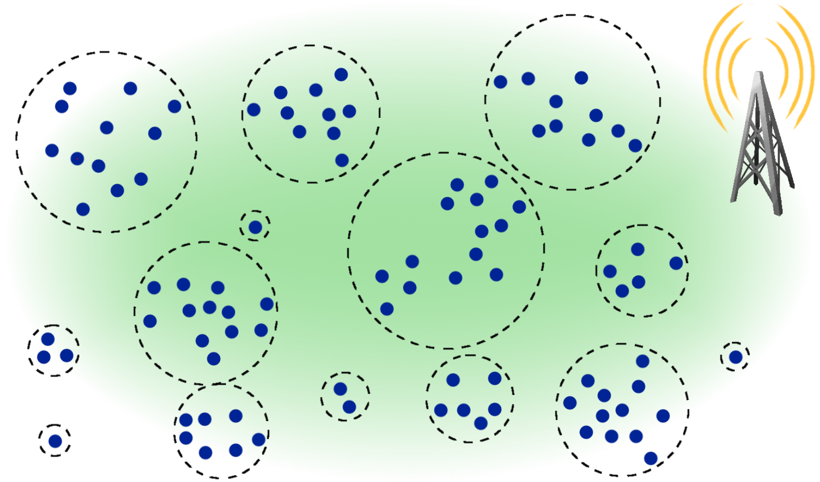
Routing in WSNs



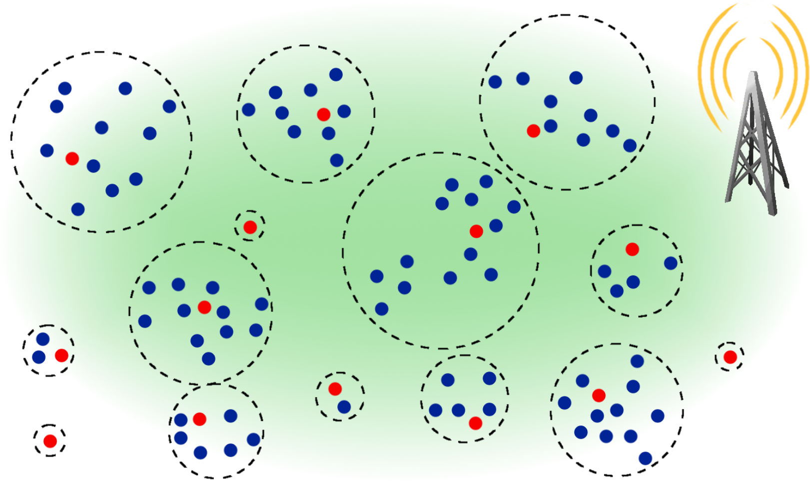
Routing in WSNs



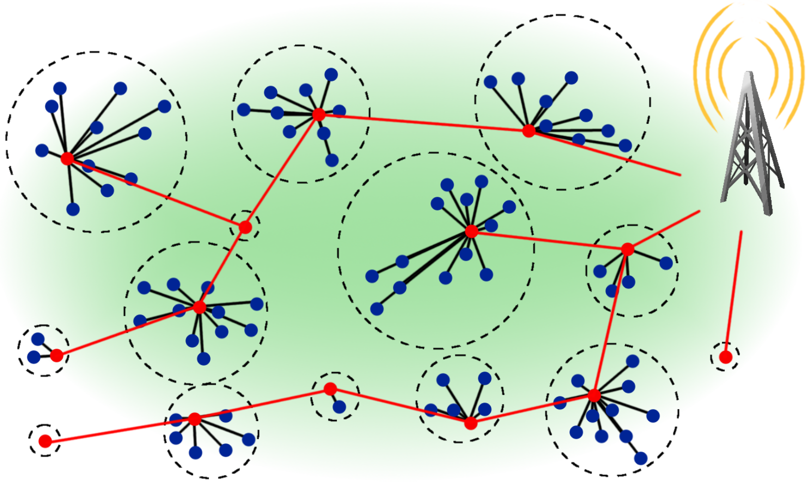
Clustering in WSNs



Clustering in WSNs



Clustering and routing in WSNs



Definitions

What is "ultrametric distance"?

Distance

d is a distance if and only if for any couple $(s_1; s_2)$:

- $d(s_1, s_2) = d(s_2, s_1)$ (symmetry)
- $d(s_1, s_2) \geqslant 0$, and $d(s_1, s_2) = 0 \Leftrightarrow s_1 = s_2$ (positive definiteness)
- $d(s_1, s_2) \leqslant d(s_1, s_3) + d(s_3, s_2)$ (triangle inequality)

Definitions

What is “ultrametric distance”?

Distance

d is a distance if and only if for any couple $(s_1; s_2)$:

- $d(s_1, s_2) = d(s_2, s_1)$ (symmetry)
- $d(s_1, s_2) \geqslant 0$, and
  $d(s_1, s_2) = 0 \Leftrightarrow s_1 = s_2$ (positive definiteness)
- $d(s_1, s_2) \leqslant d(s_1, s_3) + d(s_3, s_2)$ (triangle inequality)

Ultrametric distance

d is an ultrametric distance if and only if for any triple $(s_1; s_2; s_3)$:

- d is a distance
- $d(s_1, s_2) \leqslant \max(d(s_1, s_3), d(s_3, s_2))$ (strong triangle inequality)

The main steps of FFUCA

6 main steps:

1. *Choose uniformly at random a sample elements from the global set*
2. *Execute a classic hierarchical clustering algorithm with d on the sample*
3. *Represent the distances in the resulting dendogram: the ultrametric space is built*
4. *Deduce the clusters' intervals (thresholds)*
5. *Choose uniformly at random one representative per cluster from the result of Step 2*
6. *Pick the rest of data and compare them, according to d, with the clusters' representatives;*
    - *if it is close ($d(s_i, s_j) \leqslant threshold_j$) to one or more representative, then add it to the same cluster*
    - *else create a new cluster*

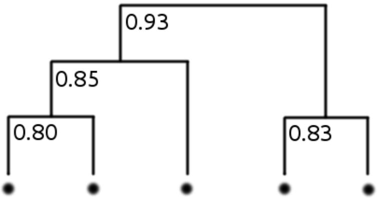
Principle of FFUCA

Principle

1. use energy consumption for communication between two nodes as a distance
2. choose a sample elements from whole set and order them with classic algorithm (steps 1, 2, 3)
3. use the ordered elements as a basis to build the clusters (steps 4, 5, 6)

Ultrametric distance

- ordering $\Rightarrow$ indiced dendogram $\Rightarrow$ ultrametric space
- distances on the opposite illustration are ultrametric



Application of FFUCA (1)

Step 1

Choose uniformly at random a sample elements from the global set

Application of FFUCA (1)

Step 1

Choose uniformly at random a sample elements from the global set

- application: choose a sample nodes from the network
- for instance: with $n = 1000$, choose $m = 20$ random nodes

Application of FFUCA (1)

Step 1

Choose uniformly at random a sample elements from the global set

- application: choose a sample nodes from the network
- for instance: with $n = 1000$, choose $m = 20$ random nodes

Remarks

- m depends on n: the larger m, the pettier m is compared to n. For instance:
  - $n = 100\ 000$, $d_e(s_i, s_j) \in [0; 0.5]$, $m = 500 = \frac{n}{200}$
  - $n = 600$, $d_e(s_i, s_j) \in [0; 300]$, $m = 15 = \frac{n}{40}$

Application of FFUCA (2)

Step 2

Execute a classic hierarchical clustering algorithm with d on the sample

Application of FFUCA (2)

Step 2

Execute a classic hierarchical clustering algorithm with d on the sample

- classic algorithms: *UPGMA*, *WPGMA*
  ((**U**n)**W**eighted **P**air **G**roup **M**ethod with **A**rithmetic mean)

Application of FFUCA (2)

Step 2

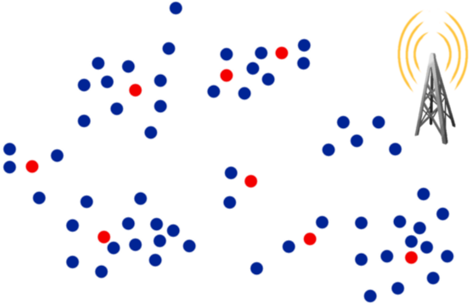
Execute a classic hierarchical clustering algorithm with d on the sample

- classic algorithms: *UPGMA*, *WPGMA*
  ((**U**n)**W**eighted **P**air **G**roup **M**ethod with **A**rithmetic mean)
- for instance: apply *WPGMA* to the sample nodes
- now sample nodes are ordered

Application of FFUCA (3)

Step 3

Represent the distances in the resulting dendogram: the ultrametric space is built

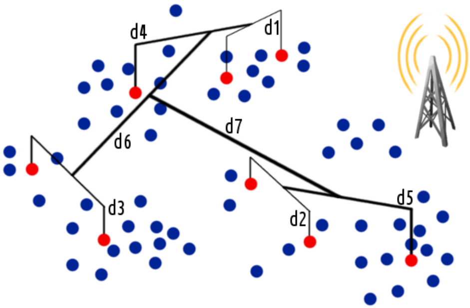


Application of FFUCA (3)

Step 3

Represent the distances in the resulting dendogram: the ultrametric space is built

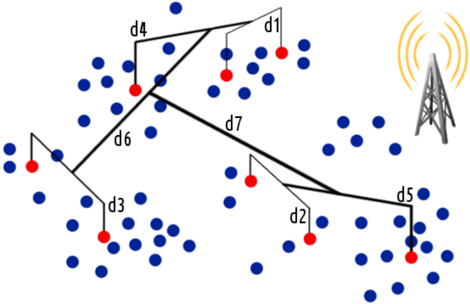
- let's do it. . .



Application of FFUCA (4)

Step 4

Deduce the clusters' intervals (thresholds)

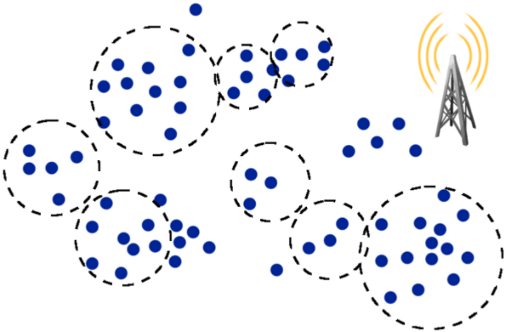


Application of FFUCA (4)

Step 4

Deduce the clusters' intervals (thresholds)

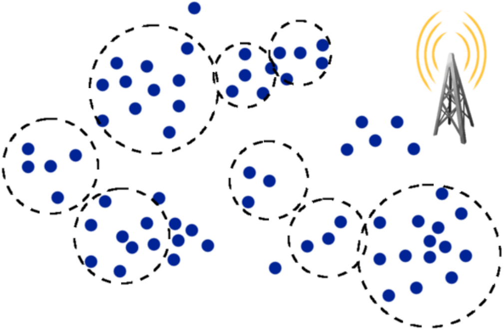
- the sample nodes set the thresholds with regard to their respective distances and to the order resulting from Step 3
- each node from sample sets its own threshold



Application of FFUCA (5)

Step 5

Choose uniformly at random one representative per cluster from the result of Step 2

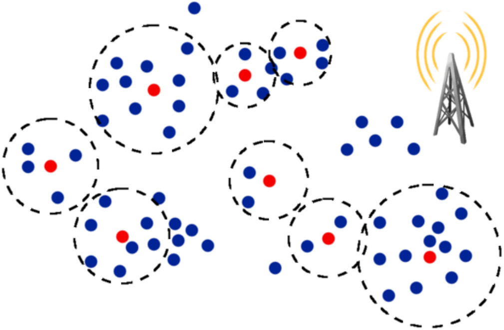


Application of FFUCA (5)

Step 5

Choose uniformly at random one representative per cluster from the result of Step 2

- choose sample nodes from Step 1 as representatives
- those nodes become cluster heads (CHs)

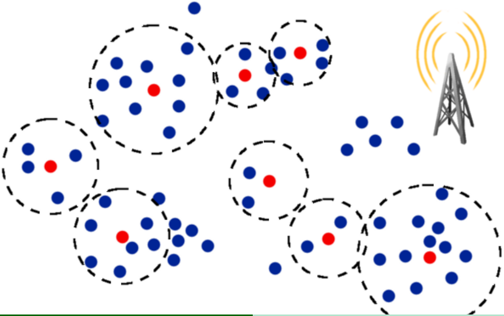


Application of FFUCA (6)

Step 6

Pick the rest of data and compare them, according to d, with the clusters' representatives;

- *if it is close to one or more representative, then add it to the same cluster (i.e. if $d(s_i, s_j) \leqslant threshold_j$)*
- *else create a new cluster*

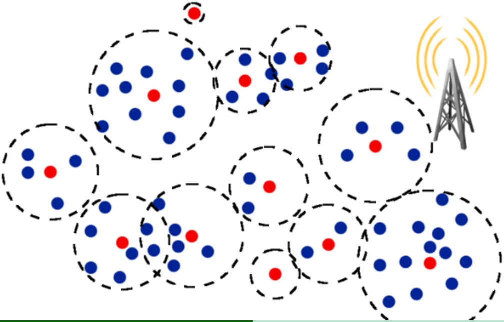


Application of FFUCA (6)

Step 6

Pick the rest of data and compare them, according to d, with the clusters' representatives;

- *if it is close to one or more representative, then add it to the same cluster (i.e. if $d(s_i, s_j) \leqslant threshold_j$)*
- *else create a new cluster*



Complexity of FFUCA

Several cases

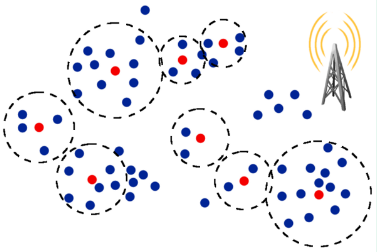
- rare worst case: $\mathcal{O}(n^2) + \mathcal{O}(m^2)$
- most cases: $\mathcal{O}(n) + \mathcal{O}(m^2) = \mathcal{O}(n) + \epsilon$

Worst case happens when clustering provides only singletons.

Comparison with LEACH

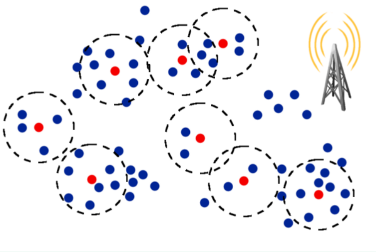
FFUCA

- different thresholds according to sample nodes order
- "sticks better" to nodes repartition



LEACH

- every CH emits with the same power, so all clusters have the same radius



Conclusion and future works

Application of FFUCA to WSNs

- Clustering is made according to "energy consumption" distance.
  $\rightarrow$ energy efficient
- Complexity in most case is $\mathcal{O}(n) + \epsilon$ (rare worst case: $\mathcal{O}(n^2) + \epsilon$).
  $\rightarrow$ fast, scalable

We are now working on...

- detailing and improving the implementation
- simulating on ns-3 network simulator
- comparing with LEACH algorithm
- comparing with other clustering algorithms (HEEDS...)

The end

Thank you!

Questions?