Locality helps sleep scheduling

Jukka Suomela

Helsinki Institute for Information Technology HIIT
Department of Computer Science, University of Helsinki, Finland

jukka.suomela@cs.helsinki.fi
http://www.cs.helsinki.fi/jukka.suomela/

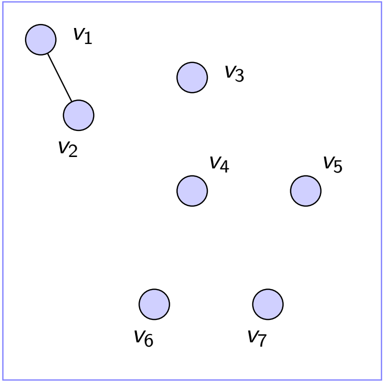
Workshop on World-Sensor-Web (WSW 2006)
31 October 2006

Sleep scheduling problem

- Given: a sensor network
- Assumption: there may be some redundant nodes
- Objective: find a sleep schedule that maximises the lifetime of the network
- Constraints:
  - Energy-constrained nodes
  - At any point of time, a node can sleep only if it is redundant

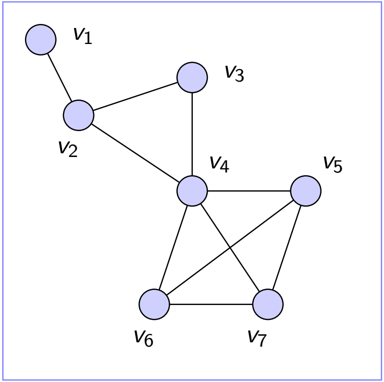
Redundancy graphs

- Focus: pairwise redundancy of the nodes
- Example: if v_1 is active then v_2 may be asleep and vice versa (e.g., sensors are close to each other)



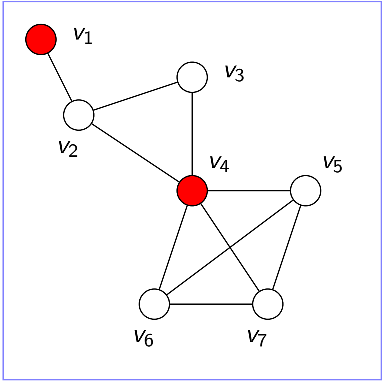
Redundancy graphs

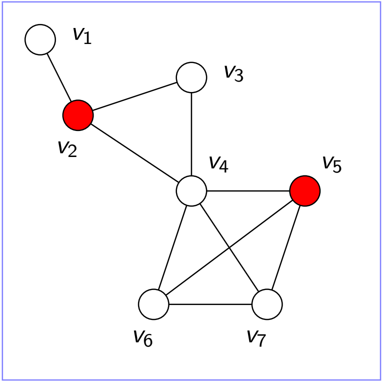
- Redundancy relations can be represented as a graph



Redundancy graphs

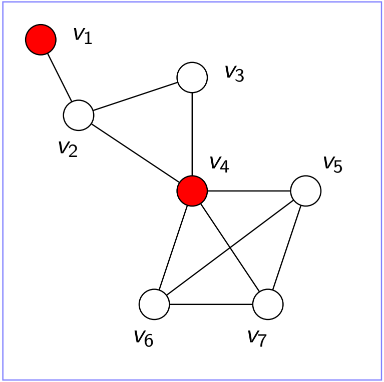
- A valid set of active (nonsleeping) nodes
  = a dominating set in the redundancy graph
- Two examples; red = active, white = asleep:



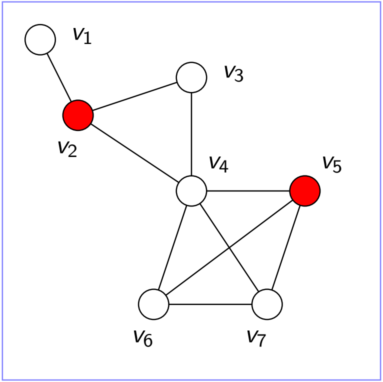


Redundancy graphs

- Sleep schedule = a time interval for each dominating set
- No single node is active for more than 1 unit of time
- An example of a sleep schedule of length 2:



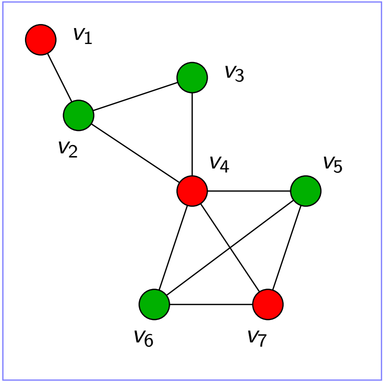
Active for 1 time unit



Active for 1 time unit

Domatic partition

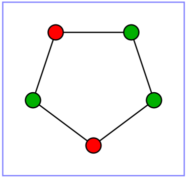
- One possible solution: find a domatic partition
- That is, partition the nodes into disjoint dominating sets (maximum number of such sets = domatic number)
- Assign 1 time unit to each such set
- Length of sleep schedule = number of such sets

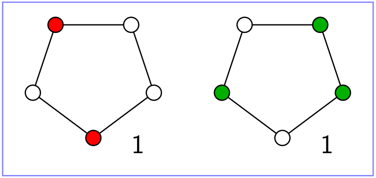


Green nodes are a dominating set and red nodes are another disjoint dominating set

Domatic partition

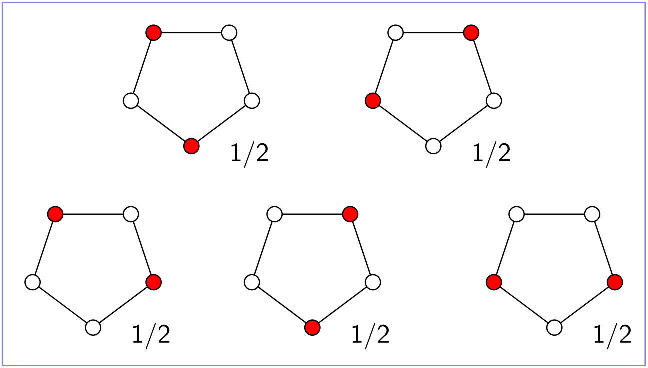
- However, domatic partition does not necessarily give an optimal sleep schedule
- Example: ring of 5 nodes (neighbours pairwise redundant)
- Domatic number is 2
- We obtain a sleep schedule of length 2:





Fractional domatic partition

- We have to allow fractional solutions
- This is an LP relaxation of domatic partition (fractional domatic partition)
- An optimal sleep schedule of length 5/2:

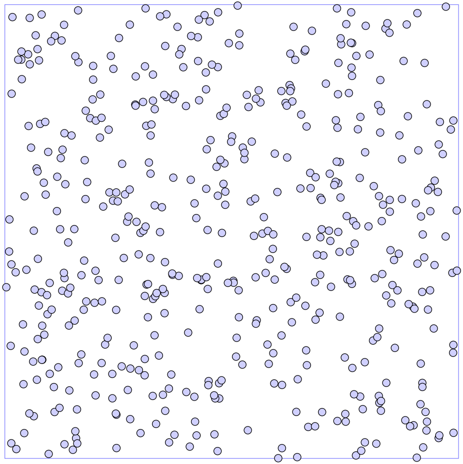


Fractional domatic partition

- Sleep scheduling with pairwise redundancy = fractional domatic partition of the redundancy graph
  - There is a lot of research on finding domatic partitions (and more general set cover packings, set K-cover), but little research on the fractional versions
- Unfortunately, both domatic partition and fractional domatic partition in general graphs are hard to approximate within factor $(1 - \epsilon) \ln |V|$
- Solution: observe that realistic redundancy graphs are not arbitrary graphs
- We focus on what we call (d, N)-local graphs

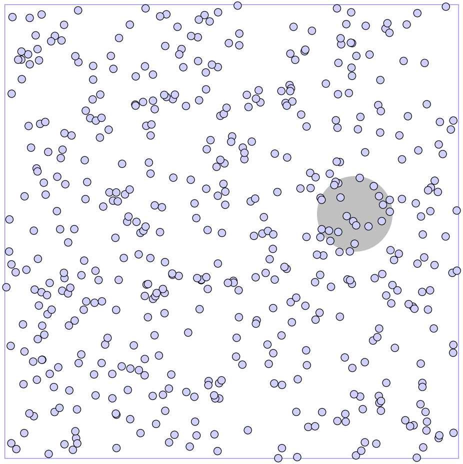
Local graphs

- Nodes are points in a *d*-dimensional space



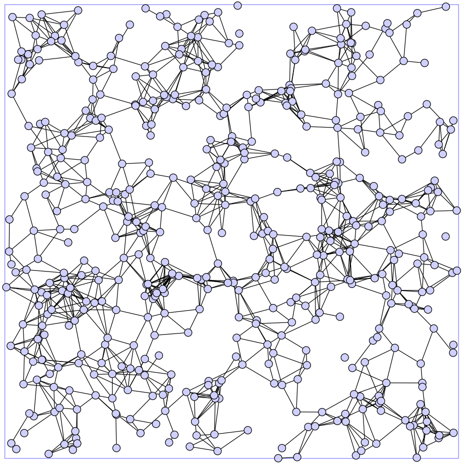
Local graphs

- Nodes are points in a d-dimensional space
- No more than N nodes in any unit disk



Local graphs

- Nodes are points in a d-dimensional space
- No more than N nodes in any unit disk
- No edges longer than 1 unit



Fractional domatic partition in local graphs

Main result:

- Polynomial-time approximation scheme (PTAS) for fractional domatic partition in local graphs
  - That is, for any $\epsilon > 0$, there is a polynomial-time $(1 + \epsilon)$-approximation algorithm

Techniques:

- Garg-Könemann LP approximation scheme:
  - Problem reduced to minimising *weighted dominating set*
- Divide-and-conquer technique based on modular grids:
  - Multiple partitions of the plane
  - Solve weighted dominating set optimally in each cell
  - Nodes near borders of the cells may do extra work
  - However, at least one of the partitions is good: there is not too much weight near the borders

Summary

- Focus on pairwise redundancy
- Sleep scheduling in sensor networks
  = fractional domatic partition in redundancy graphs
- Hard to solve or approximate in general
- Focus on (d, N)-local graphs
- Polynomial-time approximation scheme (PTAS)
  for fractional domatic partition in local graphs
- Assumptions on locality help with sleep scheduling

Jukka Suomela
Helsinki Institute for Information Technology HIIT
Department of Computer Science, University of Helsinki, Finland

jukka.suomela@cs.helsinki.fi
http://www.cs.helsinki.fi/jukka.suomela/