Naps: Scalable, Robust Topology Management in Wireless Ad Hoc Networks

Brighten Godfrey and David Ratajczak

IPSN 2004, Berkeley, CA
April 27, 2004
What is Naps?

- Naps is a simple, randomized algorithm that “thins” an ad hoc network to a desired density of nodes per unit area without knowledge of the underlying density or node location.

- Potential applications: reducing contention among radios, smoothing sensing coverage

- Application in this paper: power saving
  - Nodes deployed at density $\lambda$
  - Density $\lambda_t < \lambda$ sufficient for multi-hop routing connectivity
  - Use Naps to thin network to density $\lambda_t$
  - Thinned nodes “sleep” (turn off their radios)
Model

*Geometric random graph:*

- Nodes distributed uniformly at random in a square region
- Average of $\lambda$ nodes per unit area
- Unit radius connectivity

Naps performs well empirically even under relaxed assumptions.
Intuition

• $\lambda = \text{underlying density}, \lambda_t = \text{target density}$

• Easy way to thin to desired density: leave each node on with probability $\lambda_t/\lambda$ (others sleep)
  
  – Nodes distributed like Poisson process with intensity $\lambda$
  
  – Poisson thinning property: waking set is like Poisson with $\lambda_t$

• Problems:
  
  – Needs global knowledge of $\lambda$
  
  – Node density may vary over space and time

• Naps uses an adaptive local estimate of underlying density
The Naps algorithm in words

Executed at each node:

• Iterate over time periods:
  – Broadcast HELLO message
  – Listen for HELLO messages from neighbors
  – If $c$ HELLO messages received, sleep until end of period
• Initially and every 10 periods thereafter, period length is uniform-random $\in [0, T)$; otherwise period length is $T$.

Two parameters:

• **Neighbor threshold** $c$ proportional to target density (e.g. $c = 6$)
• **Time period** $T$ controls rate of turnover (e.g. $T = 10$ minutes)
The Naps algorithm in pictures

Here $c = 4$. 

$0 \quad \rightarrow \quad \text{time}$
The Naps algorithm in pictures

Here $c = 4$. 
The Naps algorithm in pictures

Here $c = 4$. 

Hello!
The Naps algorithm in pictures

Here $c = 4$. 

Hello!
The Naps algorithm in pictures

Here $c = 4$. 

Hello!
The Naps algorithm in pictures

Here $c = 4$. 

Hello!
The Naps algorithm in pictures

Here $c = 4$. 
The Naps algorithm in pictures

Here $c = 4$. 

Hello! Hello!

1 2 1

Hello!

0 t t+T
The Naps algorithm in pictures

Here $c = 4$. 

![Diagram showing the Naps algorithm with time intervals and 'Hello!' at specific points.](image-url)
The Naps algorithm in pictures

Here $c = 4$. 

Hello! Hello!

Hello! Hello!

Hello! Hello!

Hello! Hello!

Hello! Hello!
The Naps algorithm in pictures

Here $c = 4$. 

0 $t$ $t+T$
The Naps algorithm in pictures

Here $c = 4$. 
The Naps algorithm in pictures

Here $c = 4$. 

Hello!  Hello!  Hello!  Hello!

0 1 2 t 1 2 3 4  t+T

Zzzzz...
The Naps algorithm in pictures

Here \( c = 4 \).
Why it works

• Suppose node $v$ has $d$ neighbors
• HELLO messages received by node $v$ are uniformly distributed
• $\implies$ expected time between two messages is $\frac{T}{d+1}$
• Node $v$ stays awake for $c$ of these intervals per period $T$
• $\implies$ awake for fraction $\frac{c}{d+1}$ of time
• $E[d] = \pi \lambda$
• $\implies$ average node stays awake for fraction of time $\approx \frac{c}{\pi \lambda}$
• i.e. for target density $\lambda_t$, pick $c = \pi \lambda_t$
Using Naps for power saving

- Goal: Turn off as many nodes as possible such that multi-hop routing still works, i.e.
  - (almost all) waking nodes are in a connected component and
  - (almost every) sleeping node has a waking neighbor

- Property of geometric random graphs: there is a critical density $\lambda_c$ above which a large fraction of nodes are in a connected component w.h.p.

- Set $\lambda_t$ above critical threshold $\Rightarrow$ almost all waking nodes are connected

- Random graphs produced by Naps are not geometric random graphs...

- ...but we prove they also have a critical threshold above which connectivity is good.
Simulation: connectivity

MCA = Maximum Component Accessibility = fraction of nodes in or adjacent to the largest waking component
Area = 625. 1st percentile is minimum of 100 samples within a time period, then averaged over 20 trials.
Simulation: power savings

Network lifetime is time that $\text{MCA} \geq 0.9$.

$r = \text{ratio of sleeping power to waking power.}$

Area $= 900$, $c = 6$, waking node lifetime $= 100T$. 5 trials.
Summary

• Naps selects a rotating set of “waking” nodes of a desired density

• Advantages
  – Low communication (one message sent per node, $\Theta(\lambda_t)$ received)
  – Simple, robust (performs better in mobile setting)
  – No location information necessary

• Disadvantages
  – Only probabilistic guarantees
  – Isn’t optimal in terms of number of nodes turned off (but more efficient schemes are costly)

• Future work
  – Test performance in a real network
  – Estimate target density adaptively
Simulation: mobility

Area = 256, $\lambda = 5$, $c = 6$, $r = 0.1$, and waking node survives for time $100T$. Averaged over 5 trials.
Simulation: scaling

1st percentile is over 100 samples within a time period. $\lambda = 5$. 
Simulation: fraction of nodes awake

Area = 625.
Simulation: MCA vs. time

\[
\text{Area} = 625, \ c = 6, \ r = 0.1, \text{ and a waking node survives for time } 100T.
\]