Approximate Neighbour Counting in Radio Networks

Calvin Newport, Georgetown University

Chaodong Zheng, Nanjing University

Knowing neighbourhood is important

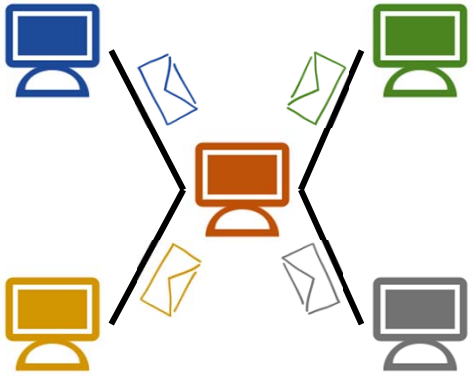
- Distributed algorithms may need to gather information from each neighbour to compute output.
  E.g.: aggregate sensor readings to compute an average.

- Distributed algorithms may require neighbourhood size as input/runtime parameter.
  E.g.: Luby's algorithm for computing MIS.

Gathering information from neighbourhood:

Wired vs Wireless

- Easy in wired networks
  - One round in the LOCAL model



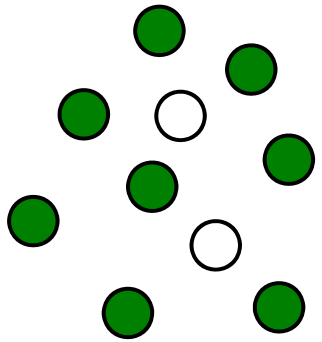
- Challenging in wireless networks
  - Messages prone to **collisions**
  - Knowing neighbourhood's size can help avoid collisions!
  - Close approximation of this size usually suffices.

Neighbourhood size estimation is in many algorithms, implicitly…

- In a *single-hop* wireless network there are N nodes in total.
  We wish to send a message m to all nodes.
  Some (say x) already know m, but others don't.
  We know N, but not x, how to get the job done?
- The DECAY procedure. [Bar-Yehuda et al. 87]

```
for (i = 1 to lg N)
    if (already know m)
        broadcast m with probability 1/2^i
    else
        listen
```

- When $i \approx \lg x$, we will likely succeed.
  We estimate x with 2^i.



time slot 1 2 3

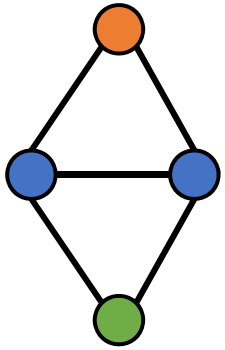
But…

- Lack systematic and comprehensive study
  - Embedded within other algorithms, not a standalone procedure.
  - Happy with “good enough” solutions, don’t know optimal solutions.
  - In fact, complexity lower bounds are unknown for many cases.

- Our paper tries to fill this gap
  - We study the problem for several common scenarios.
  - We devise standalone algorithms for approx. neighbour counting.
  - We have matching lower and upper bounds.

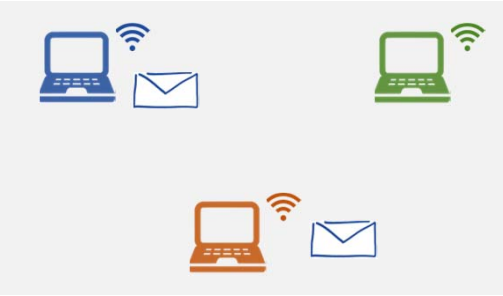
Wireless communication model

- A *synchronous* radio network modelled as an undirected graph G
- Each vertex is a radio device, $(u, v) \in E(G)$ iff u and v can talk to each other
- The network can be single-hop (G is a clique), or multi-hop (G has diameter ≥ 2)
- n is the size of the network, n_u is the number of neighbors of u, $n_\Delta = \max_u\{n_u\}$
- N is an upper bound of n, N_Δ is an upper bound of n_Δ



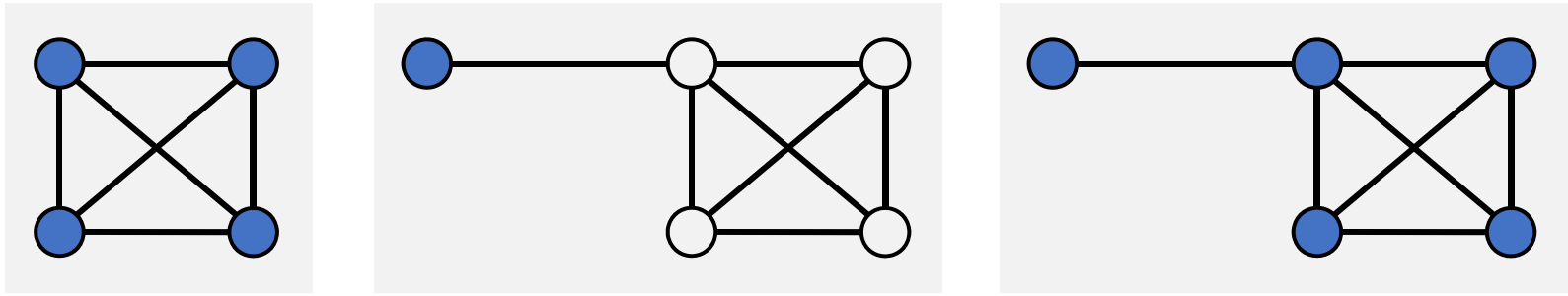
Wireless communication model

- Each radio device has a *half-duplex* radio transceiver.
  In each round, each node can either broadcast or listen, but not both.
- A broadcasting node gets no feedback regarding channel status.
- A listening node hears **silence** if **none** of its neighbours broadcast.
- A listening node hears a **message** if **exactly one** of its neighbours broadcasts.
- With collision detection, a listening node hears **noise** if **multiple** neighbours bcst.
- Without collision detection, a listening node hears **silence** if **multiple** neighbours bcst.



Approximate Neighbour Counting

- Need selected node(s) to obtain a *constant factor approx.* of their neighbourhood size(s).
  E.g., for node u, need $\hat{n}_u$ s.t. $n_u \leq \hat{n}_u \leq c \cdot n_u$ for $c \in \Theta(1)$.
- G is single-hop, all nodes need an identical estimate.
- G is multi-hop, a single designated node needs an estimate.
- G is multi-hop, every node needs an estimate.
- We consider randomized alg., and allow some chance of err.



Summary of results

		with constant probability		with high probability	
		no-CD	CD	no-CD	CD
all nodes in single-hop	lower bound	$\Omega(\lg N)$	$\Omega(\lg \lg N)$	$\Omega(\lg^2 N)$	$\Omega(\lg N)$
	upper bound	$O(\lg n)$	$O(\lg \lg n)$	$O(\lg^2 n)$	$O(\lg n)$
designated node in multi-hop	lower bound	$\Omega(\lg N_\Delta)$	$\Omega(\lg \lg N_\Delta)$	$\Omega(\lg^2 N_\Delta)$	$\Omega(\lg N_\Delta)$
	upper bound	$O(\lg n_w)$	$O(\lg \lg n_w)$	$O(\lg^2 n_w)$	$O(\lg n_w)$
all nodes in multi-hop	lower bound	—	—	$\Omega(\lg^2 N_\Delta)$	$\Omega(\lg N_\Delta)$
	upper bound	—	—	$O(\lg^2 n_u)$, $O(\lg^3 N)$	$O(\lg^2 n_u)$

- In almost all cases, we have matching bounds.
- Collision detection helps a lot.
- Better correctness guarantee needs more time.
- Leader does not help in approx. counting!

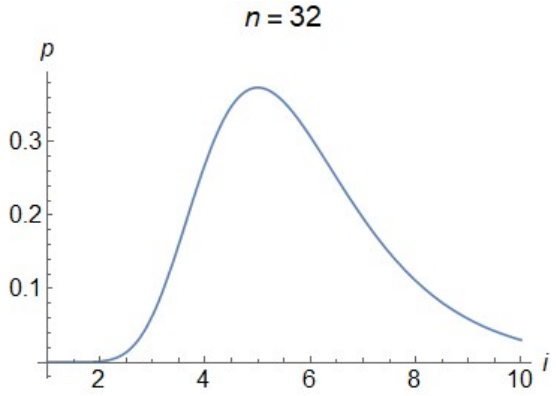
A brief tour of upper bounds

Linear guess and verify, the most basic approach

- Consider all nodes counting in single-hop network, without CD and require constant success probability.

(Simplified) Count-SH-noCD-Const:

Set $i = 1$.
Repeat
  Broadcast with probability $1/2^i$, and listen otherwise.
  Set $i = i + 1$.
Until a clear message is heard.
Return 2^i as the estimate.



$$p = n \cdot (1/2^i) \cdot (1 - 1/2^i)^{n-1}$$

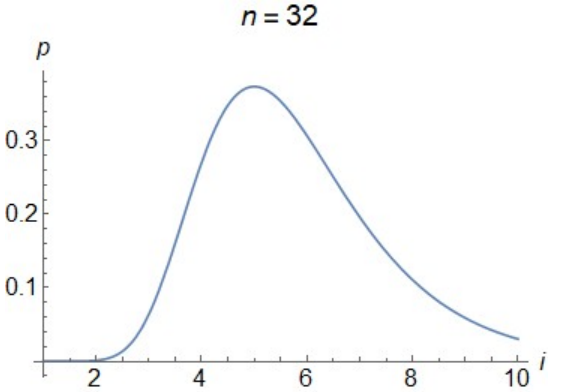
- Success probability reaches peak value when $2^i \approx n$.
- Success probability drops rapidly as 2^i deviates from n.
- Remember to inform the single broadcasting node to terminate…

Linear guess and verify, improve correctness guarantee

- Consider all nodes counting in single-hop network, without CD and require high success probability.
- For each estimate we repeat **multiple** slots, and use **fraction** of clear message slots to determine accuracy.

(Simplified) Count-SH-noCD-High:

Set $i = 1$.
Repeat
  Repeat following for $\Theta(i)$ slots:
    Broadcast with probability $1/2^i$, and listen otherwise.
  Set $i = i + 1$.
Until fraction of clear message slots exceeds $1/2e$.
Return 2^i as the estimate.



$$p = n \cdot (1/2^i) \cdot (1 - 1/2^i)^{n-1}$$

- When $i < \lg(n/\lg n)$, too many nodes will broadcast w.h.p.
- Once $i \geq \lg(n/\lg n)$, fraction value concentrated to expectation w.h.p.
- Fraction value reaches peak value ($\approx 1/e$) when $2^i \approx n$.

Guess and verify, boosted via binary search

- Consider all nodes counting in single-hop network, with CD and require constant success probability.
- Without CD, silence provides no info regarding the nature of the failure:
  - Is the estimate too large or too small?
- CD allows us to gain info from channel feedback:
  - Hearing noise imply current estimate is too small;
  - Hearing silence imply current estimate is too large.
- We can now use binary search, but we need some upper bound of n.

Guess and verify, boosted via binary search

- Consider all nodes counting in single-hop network, with CD and require constant success probability.
- We can now use binary search, but we need some upper bound of n.
- Use **linear** guess and verify approach to estimate $\lg n$. [Willard 86]

(Simplified) EstUpper-SH:

Set $i = 1$.
Repeat
  Broadcast with probability $1/2^{2^i}$, and listen otherwise.
  Set $i = i + 1$.
Until a clear message is heard.
Return 2^i as the estimate of $\lg n$.

- Nodes will not terminate when $i \leq \lg\lg n - 1$, w.h.p.
- Nodes will terminate when $i = \lg\lg n + O(1)$, w.h.p.
- We have a polynomial upper bound N of n, w.h.p. ($N \in n^{O(1)}$.)

Guess and verify, boosted via binary search

- Consider all nodes counting in single-hop network, with CD and require constant success probability.
- We can now use binary search, and we have $N \in n^{O(1)}$.

(Simplified) Count-SH-CD-Const:

Set $l = 1, h = N$.
Repeat
  Set $m = (l + h)/2$.
  Broadcast with probability $1/2^m$, and listen otherwise.
  If broadcast and heard silence then set $h = m - 1$,
  else if broadcast and heard noise then set $l = m + 1$.
Until a clear message is heard or $l > h$.
Return 2^m as the estimate.

- When m is too large or small, nodes will realize and adjust accordingly.
- When $m \approx \lg n$, nodes will hear clear message and stop.
- We success within $O(\lg \lg n)$ slots, with constant probability.

A “random walk” approach

- Consider all nodes counting in single-hop network, with CD and require high success probability.
- ~~Repeat each estimate for multiple slots to boost success probability?~~
- Move closer to correct estimate when current one is wrong, but stay still when current estimate is already correct. [Nakano and Olariu 02]
- Over a time period, most frequent estimate will be a correct one!
- $O(\lg n)$ slots ensures the most frequent estimate is correct w.h.p.

(Simplified) Count-SH-CD-High:

Set $\hat{n} = N$, where N is the upper of n returned by EstUpper-SH.
For (each slot from 1 to $\Theta(\lg N)$)
  Broadcast with probability $1/\hat{n}$, and listen otherwise.
  If broadcast and heard silence then set $\hat{n} = \hat{n}/4$,
  else if broadcast and heard noise then set $\hat{n} = 4 \cdot \hat{n}$.
Return the most frequent $\hat{n}$ as the estimate for n.

Move towards multi-hop networks…

- Relatively easy for designated node counting
  - Adjust single-hop algorithms, designated node helps coordination.
- Challenging for all nodes counting (success w.h.p.)
  - Termination detection is challenging: nodes certainly shouldn't stop too late, but they also shouldn't terminate too early.
  - High correctness guarantee is also challenging: nodes have non-uniform error probabilities when we have no knowledge of n.

All nodes counting in multi-hop networks

- Collision detection is *not* available
  - If we know N_Δ or N, or are willing to give up termination:
    - Run a variant of Count-SH-noCD-High at each node.
      (linear guess and verify, use fraction of clear msg. slots as measure)
    - For each node u, know $\Theta(n_u)$ in $O(\lg^2 n_u)$ time, w.h.p. in n_u.
  - If we know N:
    - Devise a new algorithm that uses "double counting" trick.
    - For each node u, know $\Theta(n_u)$ in $O(\lg^3 N)$ time, w.h.p. in N.
- Collision detection is available
  - Recall u shouldn't stop immediately when itself has estimate of n_u.
  - But with CD, we can let neighbours of u that have *not* finished yet to *reliably* inform u that they wish u to continue.
  - In many cases, even without knowing N_Δ or N, each node u obtains $\Theta(n_u)$ in $O(\lg^2 n_u)$ time and eventually terminates, w.h.p. in n_u.

A few words on lower bounds

Lower bounds

- Most followed by reduction from contention resolution:
  if we can solve approx. counting, then we can also solve contention resolution, with limited overhead.
- But this “limited overhead” is too large in some cases!
- Create custom lower bounds via combinatorial arguments.
- Leader does not help in approx. counting!

		with constant probability		with high probability	
		no-CD	CD	no-CD	CD
all nodes in single-hop	lower bound	$\Omega(\lg N)$	$\Omega(\lg \lg N)$	$\Omega(\lg^2 N)$	$\Omega(\lg N)$
	upper bound	$O(\lg n)$	$O(\lg \lg n)$	$O(\lg^2 n)$	$O(\lg n)$
designated node in multi-hop	lower bound	$\Omega(\lg N_\Delta)$	$\Omega(\lg \lg N_\Delta)$	$\Omega(\lg^2 N_\Delta)$	$\Omega(\lg N_\Delta)$
	upper bound	$O(\lg n_w)$	$O(\lg \lg n_w)$	$O(\lg^2 n_w)$	$O(\lg n_w)$

Summary

- Useful algorithms, matching lower bounds, for many cases.

		with constant probability		with high probability	
		no-CD	CD	no-CD	CD
all nodes in single-hop	lower bound	$\Omega(\lg N)$	$\Omega(\lg \lg N)$	$\Omega(\lg^2 N)$	$\Omega(\lg N)$
	upper bound	$O(\lg n)$	$O(\lg \lg n)$	$O(\lg^2 n)$	$O(\lg n)$
designated node in multi-hop	lower bound	$\Omega(\lg N_\Delta)$	$\Omega(\lg \lg N_\Delta)$	$\Omega(\lg^2 N_\Delta)$	$\Omega(\lg N_\Delta)$
	upper bound	$O(\lg n_w)$	$O(\lg \lg n_w)$	$O(\lg^2 n_w)$	$O(\lg n_w)$
all nodes in multi-hop	lower bound	—	—	$\Omega(\lg^2 N_\Delta)$	$\Omega(\lg N_\Delta)$
	upper bound	—	—	$O(\lg^2 n_u)$, $O(\lg^3 N)$	$O(\lg^2 n_u)$

- Better lower and/or upper bounds for all nodes counting in multi-hop?
- How termination requirements affect the complexity?