Computing average value in ad hoc networks

Daniel Letkiewicz

Institute of Engineering Cybernetics

Mirosław Kutyłowski

Institute of Mathematics

Wrocław University of Technology

Ad hoc networks

mobile devices (like handy), communicating via radio channels

new application areas:

- sensor networks (monitoring the environment, production process, ...)
- mobile devices (traffic support, ...)
- military operations (electronic devices on a battlefield, ...)

Ad hoc networks

Advantages:

- ad hoc (deployment costs ≈ 0)
- self configuration and initialization
- robust against failures and adversary

Ad hoc networks

Advantages:

- ad hoc (deployment costs ≈ 0)
- self configuration and initialization
- robust against failures and adversary

Disadvantages:

- algorithmic issues non-trivial

Communication features of ad hoc networks

- communication through shared radio channels
- communication collisions possible
  $\Rightarrow$ (random) noise
  (no-collision detection model)
- a station is either transmitting or listening or idle

Communication features of ad hoc networks

- communication through shared radio channels
- communication collisions possible
  $\Rightarrow$ (random) noise
  (no-collision detection model)
- a station is either transmitting or listening or idle

- single-hop network: a message transmitted can be received by every station

Units/stations of an ad hoc network

- the number of participating stations known only approximately, (or even unknown!)
- the stations are not labeled by consecutive numbers,
- a station may fail or join the network during protocol execution

Typical situation:

- stations with unique ID's in the range $[1, N]$,
- $\Omega(N)$ of them active

Phone cost scenarios

- private channels versus shared channels

Phone cost scenarios

- private channels versus shared channels
- pay for a message really transmitted

Phone cost scenarios

- private channels versus shared channels
- pay for a message really transmitted
- pay already for calling

Phone cost scenarios

- private channels versus shared channels
- pay for a message really transmitted
- pay already for calling
- pay for the time when waiting for calls

Design goals

- no central control
- robust against station failures
- robust against communication failures
- low time and energy complexity

Design goals

- no central control
- robust against station failures
- robust against communication failures
- low time and energy complexity
- energy cost should be kept low (battery exhaustion)
  energy cost: the maximal number of steps during which a station either transmits or listens

Computing average value

problem statement: each station i is given an integer T_i,
replace each T_i by

$$T = \sum_{\text{station } i \text{ active}} T_i / n$$

($n =$ the number of stations in the network)

Computing average value

problem statement: each station i is given an integer T_i,
replace each T_i by

$$T = \sum_{\text{station } i \text{ active}} T_i/n$$

($n =$ the number of stations in the network)

integer values: if such a T is not an integer, round it up, so that the total sum does not change
(avoiding a drift)

Applications

- aggregation of measurements in a sensor network before sending it out
- making decisions (voting with weights)
- ...

Design problems

1. active stations unknown
2. station failures, communication failures may occur
3. low energy cost must be maintained
4. it must be robust against a adversary disturbing communication

Design problems

1. active stations unknown
2. station failures, communication failures may occur
3. low energy cost must be maintained
4. it must be robust against a adversary disturbing communication
   (MAC codes can be used to authenticate against forging messages by an adversary)

Naive solution

- initialize the active stations with consecutive numbers
- sum up the numbers T_i

Problems with naive solutions:

- an initialization of the network is difficult,
- an adversary may block some crucial moments of communication, for instance he can scramble communication with some station,
- $\Rightarrow$ crucial moments in the computation are undesired,
  but typical data collection algorithm have “tree architecture”!

Solution paradigm

1. rounds

Solution paradigm

1. rounds
2. during a round many stations form pairs, each pair balance their values (replace T_i and T_j with $\lceil \frac{1}{2}(T_i + T_j) \rceil$ and $\lfloor \frac{1}{2}(T_i + T_j) \rfloor$)

Solution paradigm

1. rounds
2. during a round many stations form pairs, each pair balance their values
(replace T_i and T_j with
$\lceil \frac{1}{2}(T_i + T_j) \rceil$ and $\lfloor \frac{1}{2}(T_i + T_j) \rfloor$)
3. rounds repeated until all values become (almost) equal

Solution paradigm

1. rounds
2. during a round many stations form pairs, each pair balance their values
   (replace T_i and T_j with
   $\lceil \frac{1}{2}(T_i + T_j) \rceil$ and $\lfloor \frac{1}{2}(T_i + T_j) \rfloor$)
3. rounds repeated until all values become (almost) equal

how many rounds are necessary?

how to implement a round?

Round

- R radio channels, the number of active stations is $n = \Omega(N)$,$n \leq N$
- N communication slots used for a round ($3N/R$ steps needed), each slot consists of 3 substeps,

Round

- R radio channels, the number of active stations is $n = \Omega(N)$,$n \le N$
- N communication slots used for a round ($3N/R$ steps needed), each slot consists of 3 substeps,
- each station chooses
  - a slot t and
  - to *initiate* or *respond* and
  - an auxiliary slot t'

Execution of slot j

substep 1 each station u that has chosen $t = j$ and to *initiate* transmits its T_u,

Execution of slot j

substep 1 each station u that has chosen $t = j$ and to *initiate* transmits its T_u,

substep 2 each station v that has chosen $t = j$ and to *respond* transmits its T_v

Execution of slot j

substep 1 each station u that has chosen $t = j$ and to *initiate* transmits its T_u,

substep 2 each station v that has chosen $t = j$ and to *respond* transmits its T_v

substep 3 each station that has chosen $t' = j$ listens during the first two substeps and now retransmits the messages heard,
the stations that have chosen $t = j$ listen

Execution of slot j

substep 1 each station u that has chosen $t = j$ and to *initiate* transmits its T_u,

substep 2 each station v that has chosen $t = j$ and to *respond* transmits its T_v

substep 3 each station that has chosen $t' = j$ listens during the first two substeps and now retransmits the messages heard, the stations that have chosen $t = j$ listen

end processing each initiator u and receiver v that can hear both T_u and T_v at the 3rd substep replaces its value by $\lceil \frac{1}{2}(T_u + T_v) \rceil$ or $\lfloor \frac{1}{2}(T_u + T_v) \rfloor$

Idea of a round

- with a constant probability exactly one station chooses $t = j$ as initiator, and exactly one station chooses $t = j$ as responder, and exactly one station chooses $t' = j$
- if this happens, then there is no communication collision and the initiator and responder know about it and balance their values,
- otherwise there is no change of values – an inconsistency is avoided

Similar ideas - load balancing

- for load balancing we are happy with approximately equal load values
- fixed geometry, no stations leaving or joining
- a very similar load balancing algorithm: Gosh, Mutukrishnan, JCSS, 1996

Our result

$O(\log N)$ rounds (i.e. $O(\log N \cdot N/R)$ steps) suffice so that

- either all stations hold at most 2 values, or
- there are three values left, and the number of stations holding the smallest and the largest value is at most ϵN for a small ε

each station sends/receives $O(\log N)$ times

Optimality

$\Omega(\log N)$ trials are necessary for transmitting from each station if each transmission can fail with a constant probability.

So time and energy cost are optimal.

Limitations

- If initially all T_i are 0, except a single 1 and a single -1, then a lot of time required so that they meet each other.
- stopping with at most 3 values makes sense

Analysis outline

- use potential functions to measure quality
- three virtual phases:
  - for $x_i = T_i - T$ decrease potential

$$\sum x_i^2$$

    below αn for some constant α
    (basically Gosh Mutu...)
  - cut off the values below $T - \beta$ and $T + \beta$
    (where β is a constant)
  - one by one cut off the extreme values

Technicalities - phase 1

decreasing the first potential function

- in one round the expected value of the potential decreases by a constant factor (if no rounding)
- rounding can increase the expected value by at most $O(n)$
- (ugly terms cancel out)

Technicalities - phase 2

removing deviations higher than β

- change the potential function to

$$\sum \tilde{x}_i^2$$

  where $\tilde{x}_i = |x_i| - \beta$ if $|T_i - T| \geq \beta$ and $\tilde{x}_i = 0$ otherwise,

- balancing a pair causes decrease of its contribution to the potential function at least by a constant factor

Technicalities - phase 3

Removing extreme values one by one

- pick an extreme value, say biggest value b
- if a station with b balances with a station with a value different from b and $b-1$, then both get values different from b
- no new station can get value b

Link failures & adversary

- the probability of balancing values in a pair becomes lower,
- no change in the *structure*
- since communication pattern is random – an adversary can only make collisions at random moments
  algorithm structure unaffected

Changing values - dynamic version of the problem

- the analysis works,
- consider *old values* and *new values*
- balancing the new values does not disturb balancing the old values

End processing

- when at most 3 values left:
  run a similar minimum finding algorithm

Conclusions

- a robust algorithm
- a general purpose paradigm for algorithms in ad hoc networks
- security features are easy to deploy

Problems

- multi-hop