Adversary Immune Algorithms for Single-hop Radio Networks

Mirek Kutyłowski   Wojtek Rutkowski

Wrocław University of Technology

presentation at North Eastern University, Boston, 2004

Algorithms and Security Group at TU Wroclaw

Ongoing research areas:

- anonymity communication protocols - traffic analysis, encoding
- hardware cryptography - side channel attacks, test technology
- key distribution
- watermarks
- law regulations concerning computer security

Algorithms and Security Group at TU Wrocław

Ongoing research areas:

- security in P2P
- ad hoc networks
  - **immunity against disturbing communication for ad hoc networks**
  - **energy efficient algorithms for ad hoc networks**
- random walks, rapid mixing
- 3G telecommunication networks - network planning, ...
- *online algorithms*
- *fuzzy sets and optimization algorithms*

People

- 3 professors
- 6 *adiunkt*s
- about 10 PhD students
- very smart master students

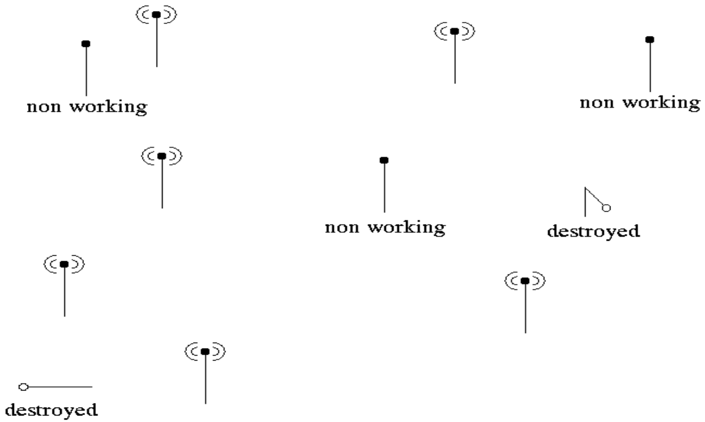
Wrocław (Breslau)

- about 600.000 inhabitants
- 300 km to Berlin, 350 km to Warsaw, 300 km to Prag
- over thousand years old, one of the richest European cities in the middle ages,
  under Polish, Czech, Austrian, German, and Polish rule
- computer science – Wrocław University, Wrocław University of Technology
- IT grows, mainly software companies

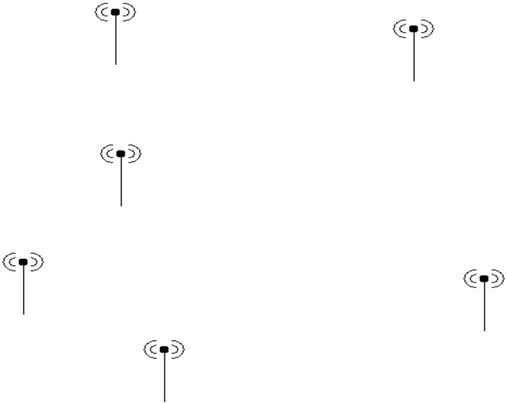
Talk Overview

- Model
- Self-organization tasks
- Performance goals
- Adversary model
- Tricks
- Overview of algorithms
- Current work

Single-hop Radio Network



Single-hop Radio Network



Single-hop Radio Network

- possible status of a station:
  - broken
  - inactive for some reason (doing something else)
  - active
- status of the stations – unpredictable
- the status may change
- serial numbers - available, but ...

Single-hop Radio Network

- communication via a **shared broadcast channel**
- a signal from a station can reach everybody **-single-hop**
- one cannot simultaneously transmit and listen

Single-hop Radio Network

- communication via a **shared broadcast channel**
- a signal from a station can reach everybody -**single-hop**
- one cannot simultaneously transmit and listen
- if two stations send then **collision** - no message comes through
- one cannot distinguish collision from a random noise - **no-CD model**

Single-hop Radio Network

- communication via a **shared broadcast channel**
- a signal from a station can reach everybody -**single-hop**
- one cannot simultaneously transmit and listen
- if two stations send then **collision** - no message comes through
- one cannot distinguish collision from a random noise - **no-CD model**
- common clock, **synchronous** communication

Single-hop Radio Network

- communication via a **shared broadcast channel**
- a signal from a station can reach everybody -**single-hop**
- one cannot simultaneously transmit and listen
- if two stations send then **collision** - no message comes through
- one cannot distinguish collision from a random noise - **no-CD model**
- common clock, **synchronous** communication
- communication can be **temporarily broken** by burst errors and malicious stations through collisions

Self Organization

- no central control
- initially - no distinguished stations
- the stations may have some preloaded shared knowledge (secret keys ...)
- computational power may vary

Other Related Definitions

Related notions

- ad hoc networks
- sensor networks
- mobile networks

Major differences

- computational power: sensor devices versus laptop computers
- energy: battery operated versus communication devices in cars
- assistance from centralized networks? (3G ...)

New Application Areas

- military
- rescue
- traffic
- new communication features

Motivations

often said:

- no central control, so resistant against failures and attacks
- dynamically adopting

Motivations

often said:

- no central control, so resistant against failures and attacks
- dynamically adopting

but the truth is:

- while hardware is available, algorithmic ideas are not at the same stage of development
- our previous experience has been focused on distributed/parallel systems with
  - centralized control
  - low dynamics
  - reliable communication
  - not many “bad guys” in the system
- unclear business case

We are in danger that ad hoc networks will fail to suceed, just as it happened with parallel systems.

State of the Art

- not many papers
- energy problem – some techniques known

State of the Art

- not many papers
- energy problem – some techniques known
- system dynamics – limited knowledge yet

State of the Art

- not many papers
- energy problem – some techniques known
- system dynamics – limited knowledge yet
- resilience to communication failures limited knowledge yet

State of the Art

- not many papers
- energy problem – some techniques known
- system dynamics – limited knowledge yet
- resilience to communication failures limited knowledge yet
- resilience to a malicious adversary – limited knowledge yet

State of the Art

- not many papers
- energy problem – some techniques known
- system dynamics – limited knowledge yet
- resilience to communication failures limited knowledge yet
- resilience to a malicious adversary – limited knowledge yet
- insider attacks – countermeasures to be developed

The algorithms need to be very **homogenous**, with no party playing a significant role.

A Bibliography Reference

There is an old story written by Stanislaw Lem about astronauts landing on a planet, where the machines extinguished biological life, and then fought among themselves. Finally, only ad hoc systems were left.
Humans had to withdraw quickly: shocked and defeated.

An exciting story about smart dust winning against an advanced human technology.

Self-organization of a Network

We start in a situation when:

- we do not know which stations are active
- we do not know the number of active stations
- no roles are assigned to the active stations

Goal:

- build a logical infrastructure so that we can run algorithms on this basis.

It is like “booting” an ad hoc network.

Booting Tasks

- estimating/counting the number of active stations
- leader(s) election
- initialization (assigning consequtive numbers to stations)
- ...

Size Approximation

find a number N such that

$$n/c \leq N \leq c \cdot n$$

where n is the (unknown) number of the active stations

Leader Election

- *exactly* one station gets the status *leader*
- the other active stations receive the status *non-leader*.

It would be useful not only to elect a leader, but also a subleader for the case when the leader becomes inactive.

Initialization

given a network consisting of n stations

goal:

- each station gets an ID number in the range $1..n$
- each number is used exactly once

Performance Measures

time - the number of time slots

Performance Measures

time - the number of time slots

energy cost - the maximal number k such that some station transmits **or** listens k times during algorithm execution

Performance Measures

time - the number of time slots

energy cost - the maximal number k such that some station transmits **or** listens k times during algorithm execution

- communication consumes almost all energy used (processor and sensors usage - negligible)
- energy required for transmitting and listening of the same magnitude
- battery exhaustion is a problem

Adversary Model

- random transmission errors, or burst errors, or even a malicious adversary knowing the algorithm

Adversary Model

- random transmission errors, or burst errors, or even a malicious adversary knowing the algorithm
- legitimate stations have a secret that is not known by the adversary
  $\Rightarrow$ keyed MAC can be used to prevent faking messages by the adversary

Adversary Model

- random transmission errors, or burst errors, or even a malicious adversary knowing the algorithm
- legitimate stations have a secret that is not known by the adversary
  $\Rightarrow$ keyed MAC can be used to prevent faking messages by the adversary
- the adversary attempts to cause collisions so that the algorithm fails

Adversary Model

- random transmission errors, or burst errors, or even a malicious adversary knowing the algorithm
- legitimate stations have a secret that is not known by the adversary
  $\Rightarrow$ keyed MAC can be used to prevent faking messages by the adversary
- the adversary attempts to cause collisions so that the algorithm fails
- an adversary cannot use much higher communication resources than other users

Adversary Model

- random transmission errors, or burst errors, or even a malicious adversary knowing the algorithm
- legitimate stations have a secret that is not known by the adversary
  $\Rightarrow$ keyed MAC can be used to prevent faking messages by the adversary
- the adversary attempts to cause collisions so that the algorithm fails
- an adversary cannot use much higher communication resources than other users
- the adversary may detect collisions

Tricks - Cryptographic Tools

- legitimate stations share a secret

Tricks - Cryptographic Tools

- legitimate stations share a secret
- all messages are enciphered *and indistinguishable from a random noise*

Tricks - Cryptographic Tools

- legitimate stations share a secret
- all messages are enciphered *and indistinguishable from a random noise*
- the secret can be used to initialize a pseudorandom number generator
  each station generates the same pseudorandom sequence

Tricks - Time Windows

- within a group of k time slots only one used for communication
- which slot is used depends on a secret pseudo-random value computed from the secret and the current time

For an adversary it is difficult to make a collision at the right moment!

Drawbacks of Time Windows

- an adversary may try many times and collide somewhere
- time increase, waste of communication bandwidth

Tricks - Interleaving Time Windows

a technique used when groups of stations perform independent computations in parallel

- a time window of length k used simultaneously by k groups
- for communication, group i uses slot

$$f(\mathrm{secret}, t; i)$$

$f(\mathrm{secret}, t; -)$ - a cryptographic pseudorandom permutation

Tricks - Interleaving Time Windows

a technique used when groups of stations perform independent computations in parallel

- a time window of length k used simultaneously by k groups
- for communication, group i uses slot

$$f(\mathrm{secret}, t; i)$$

$f(\mathrm{secret}, t; -)$ - a cryptographic pseudorandom permutation

Advantages:

- each slot used – no waste of time
- from a point of view of a group – the same behavior as for time windows

Group Transfer

- instead of a single station A transmitting to B, there is a group $\mathcal{A}$ of k stations
- each station in group $\mathcal{A}$ transmits c times within a period of $c \cdot k \cdot 2$ time slots
- a station wishing to receive from A chooses s moments within this period and listens

Colliding random s transmissions is hard.

Pairs and Collision Detection by the Sender

- arrange stations in pairs
- within a pair: the sender sends, his partner listens
- if the message comes through, it sends a confirmation during the next step

Problems:

- it is hard to arrange pairs (the next problem to be solved!)
- the problem of Byzantine generals

Problems in Algorithm Design

- all processors need to have a consistent view of the situation
- for many previous algorithms the most efficient attack is to confuse about which stations are active - then the algorithm goes crazy and the non-malicious stations destroy their communication by themselves

Size Approximation

- energy efficient solution known (energy cost $O(\log \log n)$, but fragile (EuroPar'02, Jurdziński, Kutyłowski, Zatopiański)
- adversary immune – a paper under construction

Leader Election Algorithm (ESA'2003)

- energy cost – $O(\sqrt{\log N})$
- time complexity – $O(\log^3 N)$
- the outcome might be faulty with probability $O(2^{-\sqrt{\log N}})$ for an adversary with energy cost $O(\log N)$

Leader Election Algorithm (ESA'2003)

- energy cost – $O(\sqrt{\log N})$
- time complexity – $O(\log^3 N)$
- the outcome might be faulty with probability $O(2^{-\sqrt{\log N}})$ for an adversary with energy cost $O(\log N)$
- additional feature – it produces a group of $\Theta(\log N)$ numbered stations

Leader Election Algorithm (ESA'2003)

- energy cost – $O(\sqrt{\log N})$
- time complexity – $O(\log^3 N)$
- the outcome might be faulty with probability $O(2^{-\sqrt{\log N}})$ for an adversary with energy cost $O(\log N)$
- additional feature – it produces a group of $\Theta(\log N)$ numbered stations

Fragile algorithms:

- energy cost $O(\log^* n)$ when the number of active stations is known approximately (PODC'2002)
- going already below $\log n$ is non-trivial

New Paradigm

- instead of a tree structure with reducing the number of candidates at each step
- a ring structure – in a small group of candidates everybody learns everybody

Leader Election Algorithm - Overview

$v = \Theta(\sqrt{\log N})$

- **preprocessing** - we choose at random v small groups (each of size at most $O(\log N)$) of (pairs of) candidates for the leader
- **group elections** - *group election* phase executed in group 1, then in group 2, then ...

  The first group that succeeds in choosing a group leader "**attacks**" all subsequent group election phases preventing another leader to be chosen.

Preprocessing

- each station decides at random to be either a sender or a receiver
- for each of $d = v \cdot k$ rounds:
  - a station decides to turn on the radio with probability $2/N$
  - an active sender sends a message, an active receiver listens and confirms
- if exactly one sender
  and exactly one receiver during a round: a pair emerges with
  TempID= the step number

Preprocessing

- each station decides at random to be either a sender or a receiver
- for each of $d = v \cdot k$ rounds:
  - a station decides to turn on the radio with probability $2/N$
  - an active sender sends a message, an active receiver listens and confirms
- if exactly one sender
  and exactly one receiver during a round: a pair emerges with
  TempID= the step number
- a station tries to get TempID only once, then it remains idle for the rest of preprocessing

Preprocessing – Analysis

- at each step a new pair emerges with a probability close to a known constant ($\approx 1/e^2$, if the number of stations is exactly N)
- (mathematical) problems with estimations due to the fact that a station tries at most once
- with high probability we get $\Omega(d)$ pairs with TempID assigned

Preprocessing – Analysis

- at each step a new pair emerges with a probability close to a known constant ($\approx 1/e^2$, if the number of stations is exactly N)
- (mathematical) problems with estimations due to the fact that a station tries at most once
- with high probability we get $\Omega(d)$ pairs with TempID assigned

An adversary can prevent creating t pairs with energy cost at least t.

Group Election Phase

It consists of two stages:

- arranging all active stations from a group in one or more *chains*
- merging chains
- a chain that encompasses more than half of the address space determines a leader

for a better protection: TempId numbers are “rotated” in a pseudo-random way

Relay Procedure

- a station with TempID$= i$ looks for the stations with the TempID's closest to i:
  1. at round i it listens:
     a station with the bigest TempID $j < i$ transmits j
  2. if nothing received, then station i starts a new chain
  3. otherwise: station i confirms receiving j and takes over
  4. during the following rounds, station i transmits i until somebody responds

Relay Procedure

- a station with TempID$=$ *i* looks for the stations with the TempID's closest to *i*:
  1. at round *i* it listens:
     a station with the bigest TempID $j < i$ transmits *j*
  2. if nothing received, then station *i* starts a new chain
  3. otherwise: station *i* confirms receiving *j* and takes over
  4. during the following rounds, station *i* transmits *i* until somebody responds
- each station learns its neighbors
- energy bound: a station *i* does not transmit *i* for more than $\sqrt{(}\log n)$ times
  therefore chaining might be broken

Relay Procedure and an Adversary

- an adversary may cause collisions during communication between servers j and i
- careful design of confirmations so that in a case of irregularities:
  - station i does not take over and is excluded from the chain, or
  - a new chains starts with station i

Building Chains

- based on the relay procedure
- each step of the relay procedure executed using 4 time windows of size $\Theta(\log^{3/2} N)$

An adversary has very limited chances to disturb the chain construction, but even he suceeds there are no inconsistencies but only breaking a chain into pieces.

Merging Chains

- In a suitable time slot the last station in a chain informs the current and the next chain about all members of its chain.
- If a new chain was started due to an attack, then the next chain is able to receive this information.

Disabling Later Groups - Internal Attack

- Successfull chain is blocking the later groups from getting a chain that is big enough to elect a leader.
- There are enough stations to act as an adversary without exceeding the energy limit.
- The method of the attack: causing irregularities that force starting new chains and later attack the merging process at these points.

Disabling Later Groups - Internal Attack

- Successfull chain is blocking the later groups from getting a chain that is big enough to elect a leader.
- There are enough stations to act as an adversary without exceeding the energy limit.
- The method of the attack: causing irregularities that force starting new chains and later attack the merging process at these points.

Adversary cannot turn off the internal attack - too many places for changing the nature of irregularities.

Additional Feature

- the algorithm yields a group of $\Omega(\log N)$ active stations which know each other,
- it can be used to choose “vice leaders” at no cost

Initialization (ESAS'2004)

- energy cost $O(\sqrt{\log N})$
- time $O(N)$
- the outcome is faulty with probability $O(2^{-\sqrt{\log N}})$ in a presence of an adversary with energy cost $O(\log N)$.

time N is necessary – each active station must show up

most difficult: time $O(N)$ despite of extra measures against an adversary

Initialization - Overview

Idea: gradually increase the set of initialized stations

Initialization - Overview

Idea: gradually increase the set of initialized stations

Phase 1: initialization performed concurrently in $k = \Theta(N/\log^3 N)$ groups of polylogarithmic size;

Initialization - Overview

Idea: gradually increase the set of initialized stations

Phase 1: initialization performed concurrently in $k = \Theta(N/\log^3 N)$ groups of polylogarithmic size;

Phase 2: joining the groups from Phase 1 into a single set of $D = \Theta(n/\log^2 N)$ initialized stations whp;

Initialization - Overview

Idea: gradually increase the set of initialized stations

- Phase 1: initialization performed concurrently in $k = \Theta(N/\log^3 N)$ groups of polylogarithmic size;
- Phase 2: joining the groups from Phase 1 into a single set of $D = \Theta(n/\log^2 N)$ initialized stations whp;
- Phase 3: 4 subphases, each of them increases the number of initialized stations by a factor of $\Theta(\sqrt{\log N})$ whp;

Initialization - Overview

Idea: gradually increase the set of initialized stations

- Phase 1: initialization performed concurrently in $k = \Theta(N/\log^3 N)$ groups of polylogarithmic size;
- Phase 2: joining the groups from Phase 1 into a single set of $D = \Theta(n/\log^2 N)$ initialized stations whp;
- Phase 3: 4 subphases, each of them increases the number of initialized stations by a factor of $\Theta(\sqrt{\log N})$ whp;
- Phase 4: $\Omega(N)$ stations already initialized; use them to initialize the remaining stations similarly as in Phase 3.

Phase 1: Initialization

- each station chooses independently a group from $1..k$,
- each group runs leader election (ESA'2003)
  – with interleaving instead of windows,
- result: $N/\Theta(\log^3 N)$ groups of $\Theta(\log N)$ stations initialized.

Phase 1: Initialization

- each station chooses independently a group from $1..k$,
- each group runs leader election (ESA'2003)
  – with interleaving instead of windows,
- result: $N/\Theta(\log^3 N)$ groups of $\Theta(\log N)$ stations initialized.

What can an adversary do:

- attack at most $O(\log N)$ groups,
- even attacking a single group difficult – leader election is adversary immune!

Phase 2: Joining Initialized Sets

- counting the number of initialized stations
- the group i gets a number x of the initialized stations in the groups 1 through $i-1$ and initializes its stations with
  $$x+1, x+2, \ldots,$$
  and informs the group $i+1$
- if something goes wrong – the whole group i is discarded

Phase 2: Communication between Groups $i-1$ and i

- information processed in many time slots
- a representative of group i listens at a random subset of slots
- the representative repeats x, while all stations from groups $i-1$ and i listen

Phase 2: Adversary

- the adversary may cause discarding a small number of groups
- but **he cannot make the computation inconsistent**

Phase 3:

Overview:

- 3a already initialized stations split into *collection groups*,
  each groups collects yet uninitialized stations
- 3b collection groups are merged – similarly as during Phase 2

Phase 3a - Overview

- collection groups formed
- each collection group has a number of auxiliary stations – *servants* that maintain communication,

Phase 3a - Overview

- collection groups formed
- each collection group has a number of auxiliary stations – *servants* that maintain communication,
- each uninitialized station chooses a collection group and a step number

Phase 3a - Overview

- collection groups formed
- each collection group has a number of auxiliary stations – *servants* that maintain communication,
- each uninitialized station chooses a collection group and a step number
- inside a group: relay procedure used to collect some number of uninitialized stations that have chosen this group

Relay Procedure

step t of a collection group:

- a servant informs about the number of stations collected so far
- each uninitialized station that has chosen this group and t responds
- if no collision, then the servant sends an acknowledgment and one station has joined the collection group.

Relay Procedure

step t of a collection group:

- a servant informs about the number of stations collected so far
- each uninitialized station that has chosen this group and t responds
- if no collision, then the servant sends an acknowledgment and one station has joined the collection group.

design problems:

- a servant used only $O(\sqrt{\log n})$ times
- switching the roles between the servants
- an adversary cannot cause inconsistencies – even if some of the messages get scrumbled

Final Remarks

- if the adversary detects an encoded transmission – to late for collision, our techniques still work,
- small network sizes: a combination of the same tricks but tuned for the size of n
  (e.g. $\sqrt{n}$ might be smaller than $\log^2 n$).

Open Problems

- attack from an insider
- very frequent errors
- stations becoming inactive
- ...

Thanks for your attention!