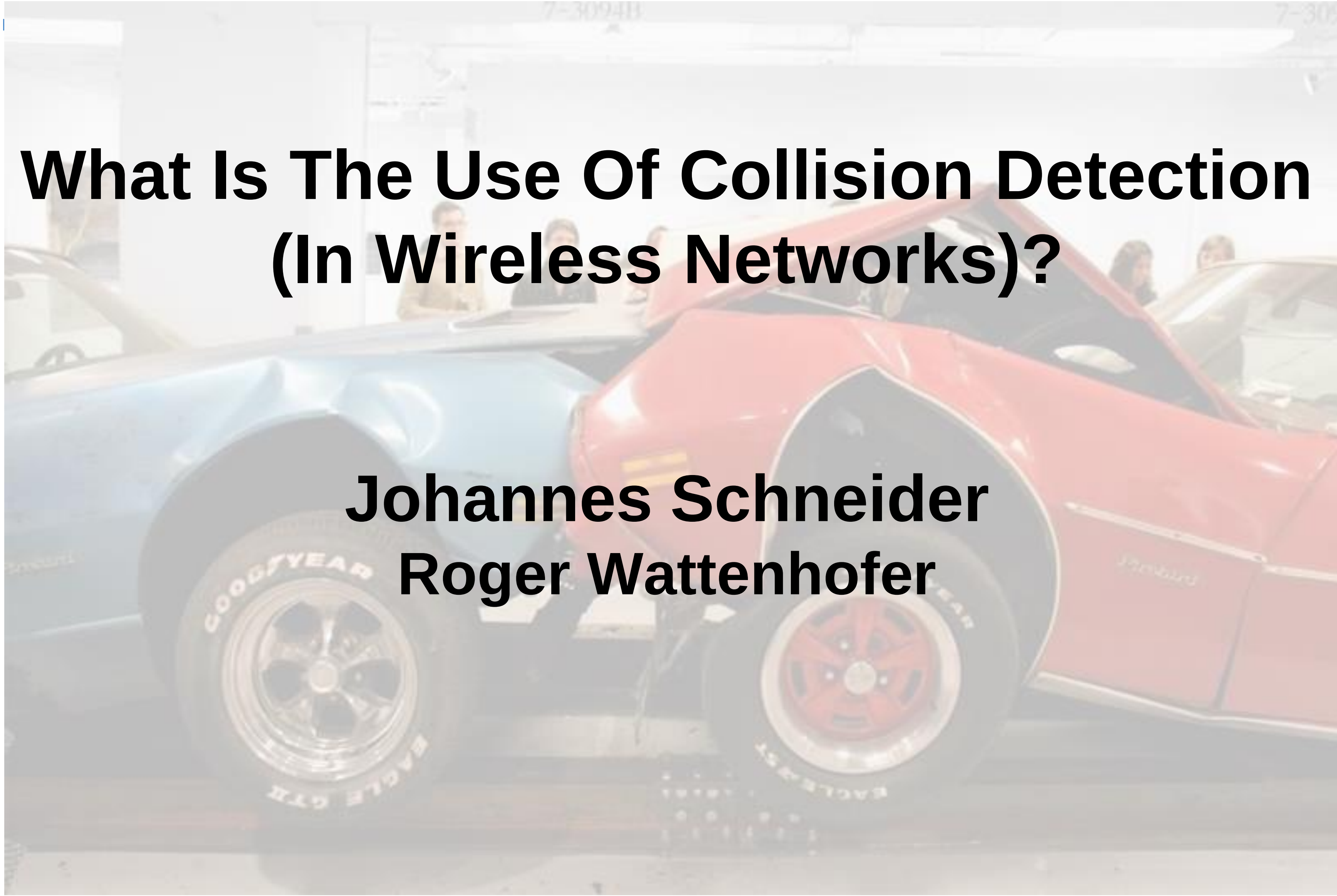


What Is The Use Of Collision Detection (In Wireless Networks)?

Johannes Schneider
Roger Wattenhofer

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Motivation

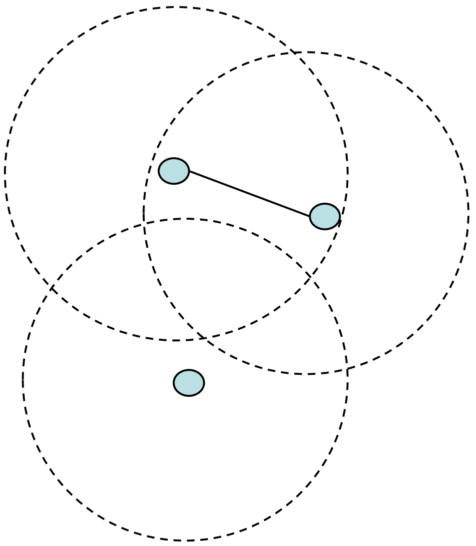
- Communication models
  - Local model
    - No interference
    - A node can transmit one distinct message to each neighbor
  - Radio network model
    - Interference
    - cannot distinguish many transmitters from none
- Reality?
  - Interference is an issue
  - But: carrier sensing allows at least to distinguish between 0, ≥1 transmitter(s)

⇒ Collision Detection Model
  - Interference
  - A node can distinguish: 0, ≥1 transmitter(s)
  - adds complexity to devices

    ⇒ **Is it worth it?**

Graph based connectivity model

- General graph
  - Too pessimistic
- Unit Disk Graphs(UDG)
  - Geometrical graph



- Problems in UDG easier than in general graph?
  - We show: lower bounds for general graphs hold for UDG

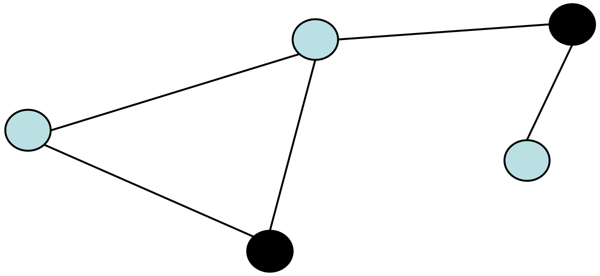
Contribution and related work

- Collision detection has been studied before
  - E.g. leader election, broadcasting

- Investigate time complexity of 3 fundamental problems

Upper and Lower Bounds		
Problem	With Collision Detection	Without
MIS	$O(\log n)$ det. [This paper]	$O(\log^2 n)$ ra. [9]
	$\Omega(\log n)$ [This paper]	$\Omega(\log^2 n / \log\log n)$[5]
$\Delta + 1$ Col.	$O(\Delta + \log^2 n)$ ra. [11]	$O(\Delta + \log^2 n)$ ra. [11]
	$\Omega(\Delta + \log n)$ [This paper]	$\Omega(\Delta + \log n)$ [This paper]
Broadcast	$O(D \log n)$ det. [This paper]	$O(n \log n)$ [6] det.
	$\Omega(D + \log n)$ [This paper]	$\Omega(n \log D)$ det. [2][This paper]

Maximal Indenpendent Set (MIS)



- Model: First, synchronous start of all nodes
- Algorithm: Adaption of [PODC'08] for local model

- Communicate in parallel despite interference?
  - Yes, if transmit bit by bit …

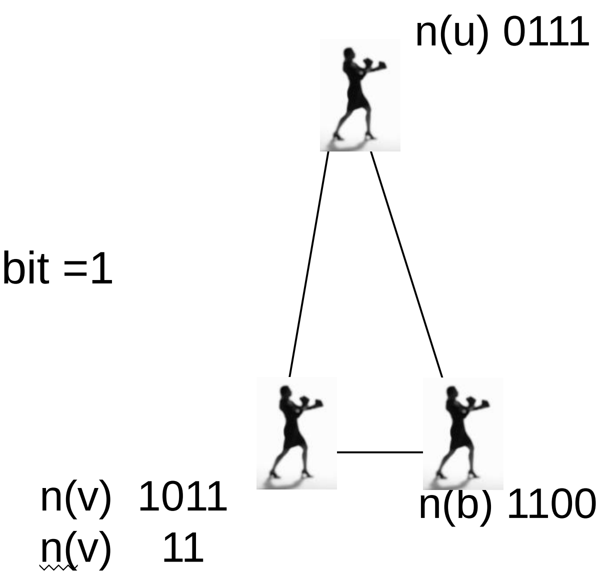
MIS Algorithm

- Each node *v* has a number *n(v)* that it uses for fighting

- Each node v updates *n(v)* after a fight to get a new number *n*(v)
  - If n(v) maximum among neighbors node wins
  - If n(v) < n(u)
    - *n*(v) is the minimum position
      where n(v) has a bit =0 and n(u) has the same bit =1
    - Example: Position 4 3 2 1

Position	4	3	2	1
n(v)	1	0	1	1
n(u)	0	1	1	1

$\Rightarrow$ n(v) = 3 =11 (binary)



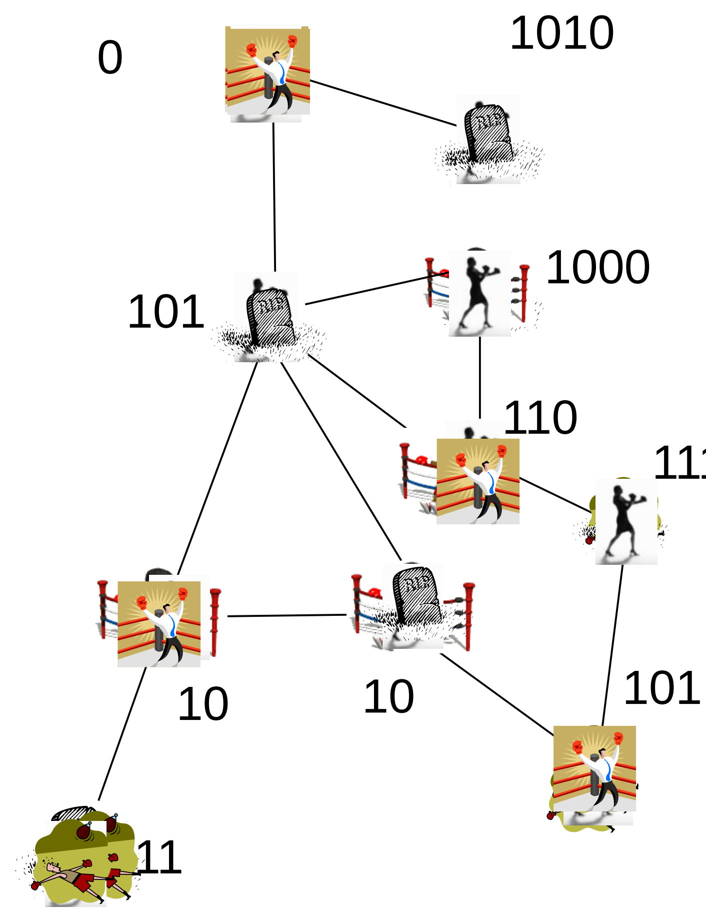
- Fight = Transmit number *n(v)* bit by bit
  - Transmit in round k if kth position = 1, otherwise listen

$\Rightarrow$ Can detect if a neighbor has bit 1, if detect collision

MIS Algorithm continued

 Every node fights against all its neighbors using some number

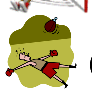
- Number is updated after every fight
- First number = ID



 If have (strictly) smallest number of all neighbors, join MIS

 If a neighbor , stop fighting

 If have smallest number and a neighbor has the same number, start over  with ID

 If a neighbor , wait until all neighbors  or  then start over  with ID

 Else compute a new number and continue fighting

How to deal with asynchrony?

- Nodes can wake up at an arbitrary point in time
- A newly woken up node should not disturb an ongoing computation

⇒ Iterate a few slots periodically to synchronize

  - Use 1 slot to execute synchronous MIS Algorithm
  - Other slots to inform nodes about the current state

Upon wake-up:

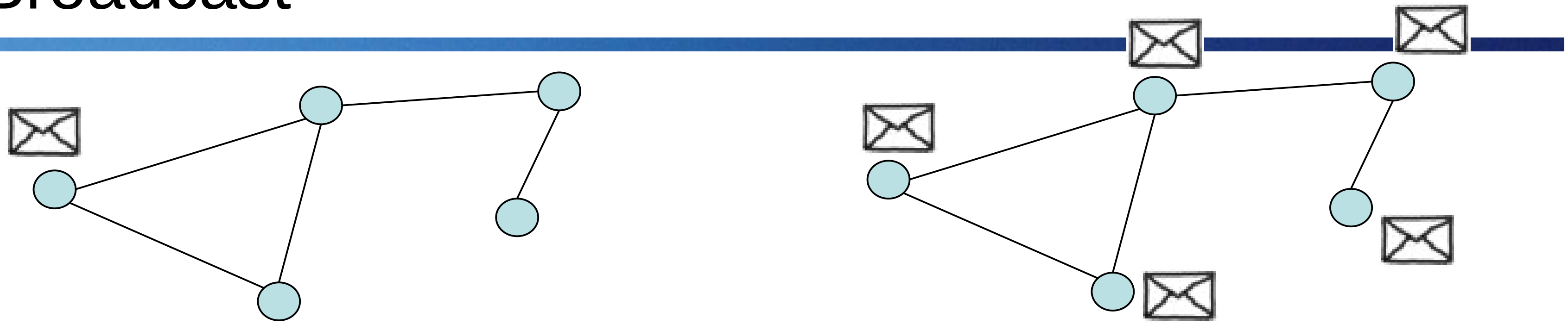
Listen until no transmission detected for 7 consecutive rounds
if ever detected transmission for 2 consecutive rounds **then** $s_v := N_{MIS}$ **else** $s_v := executing$; SixRoundSchedule() **end if**

SixRoundSchedule():

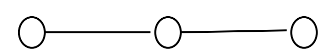
loop forever

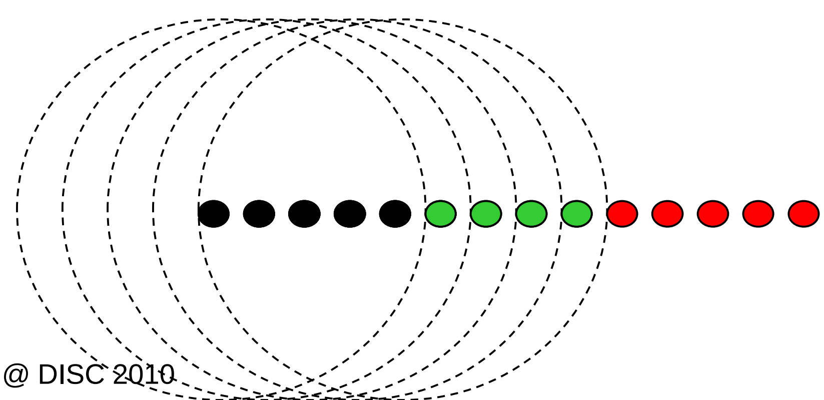
1: **if** $s_v = executing$ Synchronous MIS **then** Transmit **else** Sleep **end if**
2: Sleep
3: **if** $s_v = executing$ Synchronous MIS **then** Execute 1 step in Algorithm Synchronous MIS **else** Sleep **end if**
4: Sleep
5,6: **if** $s_v = MIS$ **then** Transmit twice **else** Sleep two rounds **end if**

Broadcast



- Candidate to transmit := node having message and neighbor lacking it
- Algorithm: Repeat:
  1) Compute MIS for candidates
  2) Nodes in MIS transmit
- But…
  – A node might have several neighbors in the MIS
  – This can be arbitrarily slow



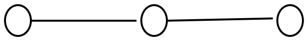


Broadcast

- Low priority candidate :=
  - node having msg and a neighbor lacking it
- High priority candidate :=
  - node having msg and a neighbor lacking it

    AND just received msg

- Nodes in MIS transmit concurrently

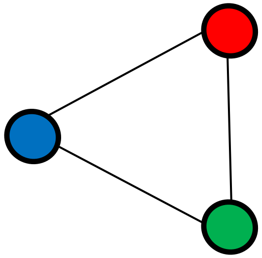


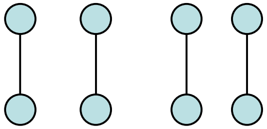
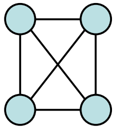
$\Rightarrow$ Use strongly selective families F = {F_0, F_1 ,…,$F_{\log n}$}

F_i = predetermined set of nodes

Node v transmits in round i if v in F_i

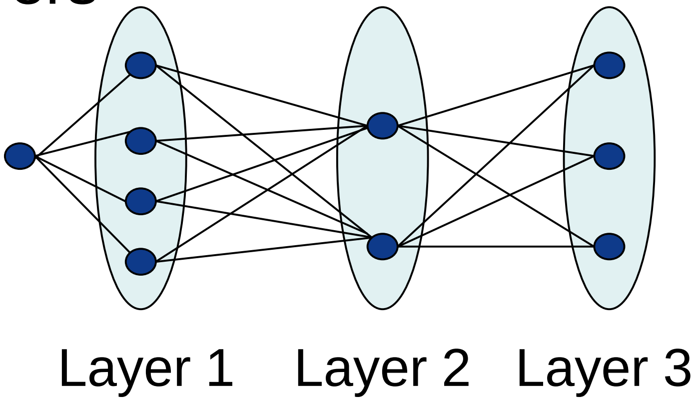
Lower bound coloring



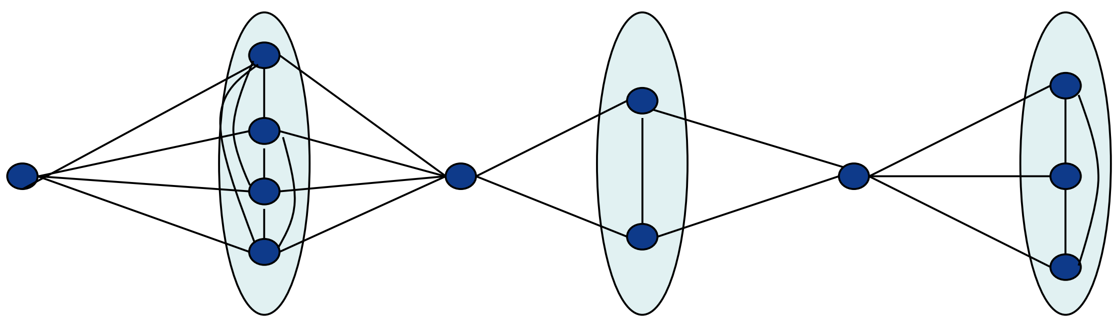
- Proof uses:
  - Information theory
  - Make algorithm only gain little information per round
- Thm: Ω(log n) time for O(1) coloring
  - Graph of pairs: 
- Thm: Ω(Δ) time for Δ+1 coloring
  - Clique: 

Lower bounds for broadcasting without collision detection

- So far: For general graphs using layers
- Difficulty for algorithm
  - Determine number of nodes in a layer



- Do they hold for UDGs?
  - Yes, can turn them into clique and add additional layers



Conclusions

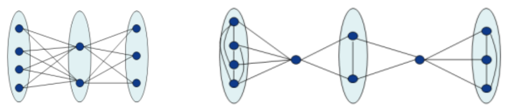
- Model: theory vs. practice

  Theory *is when you think you know something but it doesn't work.*

  Practice *is when something works but you don't know why.*

  Usually we combine *theory and practice: nothing works and we don't know why.*

  – UDG vs. general graphs



- Is collision detection useful?
  – Generally good for fast deterministic algorithms
- How useful?
  – Depends on the problem..
  – Some cases not worth it…
  …for others large speed up

Upper and Lower Bounds		
Problem	With Collision Detection	Without
MIS	$O(\log n)$ det. [This paper]	$O(\log^2 n)$ ra. [9]
	$\Omega(\log n)$ [This paper]	$\Omega(\log^2 n/\log\log n)$[5]
$\Delta+1$ Col.	$O(\Delta+\log^2 n)$ ra. [11]	$O(\Delta+\log^2 n)$ ra. [11]
	$\Omega(\Delta+\log n)$ [This paper]	$\Omega(\Delta+\log n)$ [This paper]
Broadcast	$O(D\log n)$ det. [This paper]	$O(n\log n)$ [6] det.
	$\Omega(D+\log n)$ [This paper]	$\Omega(n\log D)$ det. [2][This paper]