

Final Presentation: "Jamming Resistant MAC protocol"

Ilya Rivkin

Agenda

- Problem Presentation.
- System Model.
- Algorithm Presentation.
- Algorithm analysis.
- Additional applications.
- Conclusions and open issues.

2

Problem Presentation (1)

- Jamming attacks on networks are a problem that has to be fought.
- Jamming PHY layer:
 - Simple concept.
 - Relatively efficient.
 - Demands high transmit power.
 - Wide range of protocols can be jammed by a single system.
 - Possible to overcome (Spread Spectrum, Frequency hopping etc.)



3

Problem Presentation (2)

- Jamming MAC layer:
 - Cheap
 - Demands protocol knowledge
 - Very hard to overcome
 - Mostly SW implementation



4

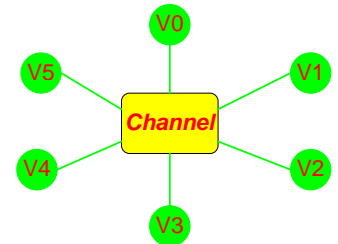
Problem Presentation (3)

- MAC Jamming – How it works?
 - Modern wireless protocols allow Collision.
 - Whenever Collision occurs – the node that intended to transmit reduces its transmission probability.
 - To disrupt communication – Just keep initiating transfers!!

5

System Model (1)

- Single hop network model:
 - Each node transmits to all vertices at once – shared bus.
 - Synchronous time model.
 - At each time step a node can:
 - Sense the channel (receive a packet).
 - Transmit a packet.



6

System Model (2)

- Single hop network model:
 - Sensing the channel has 3 possible outcomes:
 - "Idle" – the channel is free.
 - "Collision" – two or more nodes transmit.
 - "Receive" – if exactly one node transmits – a packet is received.
 - Receive succeeds – Only if 1 node transmits at a time step.
 - Collision - two or more nodes transmit at the same time step.

7

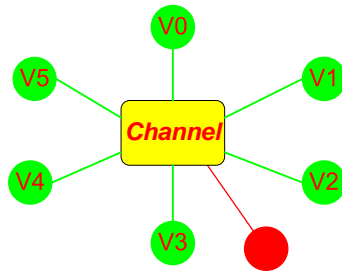
System Model (3)

- Random back off protocol:
 - Common protocol for many networks.
 - When a node has to send a message:
 1. Attempt to transmit – with probability p
 2. If transmission fails
 3. Decrease p – for example $p := p^2$ until p reaches the lower bound.
 4. Goto 1

8

System Model (5)

- The adversary (1):
 - Knows and works by the protocol.
 - Adaptive in behavior – jams at will and not by pattern.
 - Jamming – will be sensed as “Collision” state.



9

System Model (6)

- The adversary (2):
 - (T, λ) bounded – $T \in \mathbb{N}$, $0 < \lambda < 1$
For window size $w \geq T$, the adversary can jam at most λw time steps of the window.
 - All nodes always wish to transmit.
 - In our model the adversary is $(T, 1 - \varepsilon)$ bounded.
 - Honest nodes cannot distinguish between a normal “Collision” and one caused by jamming.

10

System Model (7)

- Our Goal:
 - To create a constant competitive protocol against any $(T, 1 - \varepsilon)$ bounded adversary.
 - The nodes have a common parameter:
 $\gamma = O\left(\frac{1}{\log T + \log \log n}\right)$ - a very rough estimate of their number and T .
 - The protocol is symmetric – no IDs, same rules.

11

Algorithm presentation (1)

- Algorithm Background:
 - Any node v at any time step with probability $p_v \leq p'$.
 - Now let's define:
 - $p = \sum_{v \in V} p_v$
 - Probability for “idle” – q_0
 - Probability for “receive” – q_1

12

Algorithm presentation (2)

- Claim 1 – for any time step t :

- $q_0 p \leq q_1 \leq q_0 p / (1-p)$

- Proof:

$$q_0 = \prod_{v \in V} (1 - p_v), \quad q_1 = \sum_{v \in V} p_v \prod_{\substack{w \in V \\ w \neq v}} (1 - p_w)$$

$$q_1 \leq \sum_{v \in V} p_v \frac{1}{1-p} \prod_{w \in V} (1 - p_w) = \frac{q_0 p}{1-p}, \quad \text{assuming } p_v = p \text{ (at most)}$$

$$q_1 \geq \sum_{v \in V} p_v \prod_{w \in V} (1 - p_w) = q_0 p \quad \text{counting } p_v \text{ as well}$$

13

Algorithm presentation (3)

- If there is a similar number of steps with "idle" and with "receive" then from claim 1 :

- $q_0 p \leq q_1$

- $q_0 \approx q_1$

- $p = \sum_{v \in V} p_v \approx 1$

- Each node adjusts p_v until the number of "idle" and "receive" states is similar.

14

Algorithm presentation (4)

- Using this principle, naïve algorithm is given:

- At each step, each node sends with p_v .

- If it decides not to send a message:

- If the channel is idle $\rightarrow p_v := (1 + \gamma) p_v$

- If the channel is single $\rightarrow p_v := (1 + \gamma)^{-1} p_v$

- Ignores time steps with collisions.
- At high load – none of the states will be observed – the nodes get "stuck" in endless Collision state.

15

Algorithm presentation (5)

- There is a solution for the problem of p getting too high:

- A threshold is set for each node – T_v

- If the node does not observe a successful transmission for T_v steps, this node's p_v is decreased.

- The nodes have no info about the size of the adversary time frame T , so T_v is set adaptively.

16

Algorithm presentation (6)

- The authors propose the following rule:
 - If a node v senses a successful transmission $\rightarrow T_v := T_v - 1$
 - v senses no successful Tx for T_v time steps $\rightarrow T_v := T_v + 1$
- Now we are ready to define the full algorithm.

17

Algorithm presentation (7)

- Each node has the following variables:
 - Transmit probability - p_v
 - Time threshold - T_v
 - A counter - c_v
- Common parameters for all nodes:
 - A rough estimate - $\gamma = O\left(\frac{1}{\log T + \log \log n}\right)$
 - Upper transmit prob. bound - $0 \leq p' \leq 1/24$

18

Algorithm presentation (8)

- At each step the node:
 - Decides with probability p_v to transmit.
 - If it decided not to transmit:
 - If channel is idle
 - $p_v := \max\{(1+\gamma)p_v, p'\}$
 - If v successfully receives a message
 - $p_v := (1+\gamma)^{-1} p_v$
 - $T_v := \max\{1, T_v - 1\}$
 - $c_v := c_v + 1$
 - If $c_v > T_v$
 - $c_v := 1$
 - If there was no success in the last T_v steps
 - $p_v := (1+\gamma)^{-1} p_v$
 - $T_v := T_v + 1$

19

Algorithm Analysis (1)

- The main theorem :
 - Theorem : "The algorithm is constant competitive with high probability under any $(T, 1-\varepsilon)$ bounded adversary if the protocol is executed for at least $\Theta\left(\frac{1}{\varepsilon} \log N \max\left\{T, \frac{1}{\varepsilon \gamma^2} \log^3 N\right\}\right)$ time steps, $N = \max\{T, n\}$ ".
 - Example: for $N=1000, T=200, \varepsilon = 0.1$: $\gamma = 0.35$, At least 66120 cycles are needed.

20

Algorithm Analysis (2)

- Proof review (definitions):
 - Let V be the complete set of nodes.
 - For some node v_0 , $U = V \setminus \{v_0\}$
 - At time t : $p_t = \sum_{v \in U} p_v(t)$
 - We study competitiveness for a time frame $L = \left(\frac{1}{\varepsilon} \log N \max \left\{ T, \frac{1}{\varepsilon \gamma^2} \log^3 N \right\} \right)$

21

Algorithm Analysis (3)

- Further definitions:
 - The proof is based on induction over time frames.
 - I – time frame, consists of subframes I' .
 - The size of I' is f .
 - The size of I is F .
 - $F = \frac{\alpha}{\varepsilon} \log N$, α – large constant

22

Algorithm Analysis (4)

- Proof flow:
 - A time frame or a step is called “good” if $p_t \leq 9$, “bad” otherwise.
 - At any I' where initially $p_t \geq \frac{1}{f^2(1+\gamma)^2\sqrt{f}}$ at the end of the frame still $p_t \geq \frac{1}{f^2(1+\gamma)^2\sqrt{f}}$ w.h.p.
 - In a “good” subframe I' – all non-jammed time steps are good w.m.p.

23

Algorithm Analysis (5)

- The proof shows that in a good subframe at least 1/8 of non jammed time steps have a constant p_t w.m.p.
- For constant p_t there is a constant probability a message is sent.
- By the use of Chernoff bounds the authors show that at least a constant part of the “good” subframes results in constant competitiveness.

24

Algorithm Analysis (6)

- Finally, by induction the proof shows that the algorithm is constant competitive at a frame level – each frame has enough “good” time steps.
- The size of the frame is $L = \left(\frac{1}{\varepsilon} \log N \max \left\{ T, \frac{1}{\varepsilon \gamma^2} \log^3 N \right\} \right)$ and that proves the initial claim of constant competitiveness.

25

Algorithm Analysis (7)

- Energy Efficiency:
 - Energy spent on sending a message is defined as a sum of tries made until successful transmission.
 - Lemma: “Consider permanent jamming from time t_0 where $p = \sum p_v, \max T_v \leq T'$, the energy spent by all nodes is bounded by $O\left(\frac{pT'}{\gamma} + \log N\right)$ (!!!)

26

Algorithm Analysis (8)

- Let’s check an example to get impressed:
 $p = 1, T' = 100, N = \max(n, T) = 1000, n = 1000, T = 100$
- For these arbitrary numbers, the energy spent converges to $O\left(\frac{pT'}{\gamma} + \log N\right)$, meaning $O\left(\frac{1 \times 100}{0.4} + 3\right) = 250$ regardless of the length of the attack!!

27

Additional application (1)

- Leader election – selecting a single node to be the network coordinator.
 - Based on our algorithm.
 - Each node keeps a counter s_v and state variable $state \in \{unknown, follower, leader\}$
 - At start $s_v = 0, state = unknown$
 - At each send, s_v is attached.

28

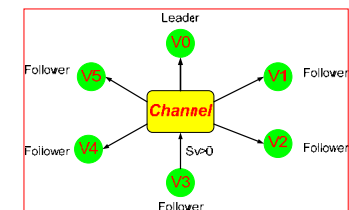
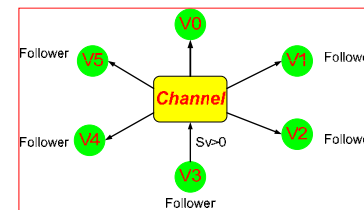
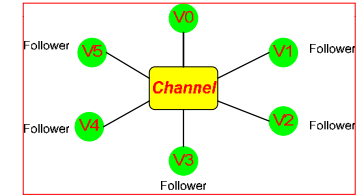
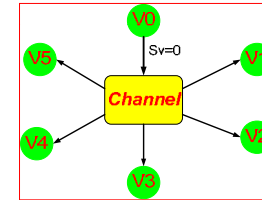
Additional applications (2)

- Each node runs our algorithms as described.
- If v receives a message from w:
 - If state = unknown
 - If $s_v \geq s_w$ then state = follower
 - Else state = leader
 - In any case, $s_v = \max(s_v, s_w) + 1$

29

Additional applications (3)

- The following example should help:



30

Additional applications (4)

- Establishing fairness – problem intro:
 - In the original protocol – some nodes will dominate with time:
 - All nodes sensing a TX – lower their p.
 - The sending node doesn't.
 - Conclusion – the nodes that send a lot in the beginning will keep high p values.
 - Fairness is lost.
- The problem is solved

31

Additional applications (5)

- Each node holds the following parameters:
 - The parameters of the main algorithm.
 - Counter of successful transmissions s_v
 - Counter of the nodes it has seen so far m_v
 - State variable – {covered, uncovered}.
 - The last counter it has received - olds.

32

Additional applications (6)

- Transmit with probability p_v , add s_v and state_v.
- If the channel is idle
 - If v is uncovered
 - Set $p_v = \max\{(1+\gamma)p_v, p'\}$ (increase p_v)

33

Additional applications (7)

- If v receives a message with some s_w
 - if state_w = uncovered and $s_w \neq olds$ (node w did not successfully receive any message)
 - Set $m_v = m_v + 1$ (add a node to the network size)
 - If state(v)=covered
 - Set $p_v = p' / m_v$
 - If state_v = uncovered and $s_w > s_v$ (the first message received by v)
 - state(v) = covered
 - Set $m_v = m_v + 1$
 - Set $p_v = p' / m_v$

34

Additional applications (8)

- If state_v = uncovered and $s_w \leq s_v$ (v made no successful transmission – nobody “sees” the node v).
 - Set $p_v = (1+\gamma)^{-1} p_v$
 - Set $T_v = \max\{1, T_v - 1\}$
- Set $olds = s_w$
- Set $s_v = \max\{s_v, s_w\} + 1$
- Proceed as in the original algorithm.

35

Additional applications (9)

- The following claims hold (without proof):
 - At any time s_v is equal to the total number of successful transmissions so far.
 - At any time m_v is equal to the number of nodes that have successfully transmitted so far.
 - A node becomes covered only if it has completed one successful transmission and reception.
 - After sufficient time, all nodes have the same transmit probability.

36



Conclusions

- The first MAC protocol that keeps a constant competitive rate against any jammer.
- The protocol provides an energy saving solution even against constant jamming.
- Slightly modified versions can be used to solve other network problems s.t leader election and improved fairness.
- The algorithm is thoroughly studied over a single hop network model.

37



Open issues

- Can the algorithm be extended to multi-hop networks?
- No simulations of the algorithm operations were made – open call (??).
- Application on the algorithm to ad-hoc networks – not studied yet.
- Still, some estimate of T and n is needed – is it possible to lose that dependency?

38