Packet Forwarding in an Ad-hoc Network of Selfish Nodes

Utku Günay Acer

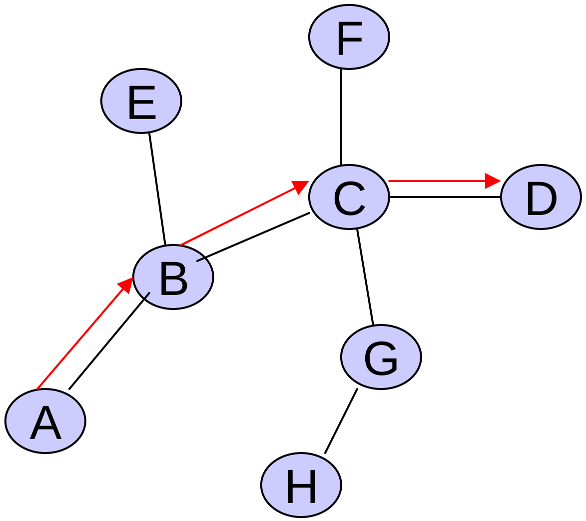
Outline

- Ad-hoc Networks and Game Theory
- Generous TIT-FOR-TAT
- Effect of the network topology on the cooperation
- A punishing mechanism that maintains cooperation
- Conclusion

Wireless Ad-hoc Networks

- The network is composed of independent nodes
- There is no an established infrastructure
- Each node is its own authority
- All the decisions are made locally
- The mechanisms should be implemented in a distributed fashion
- Networking services are provided by nodes themselves

Packet Forwarding



- Node A needs nodes B and C to relay packets to ensure communication with node D
- Why would nodes B&C cooperate?
  - Cooperation is not grated

Objectives and Issues

- Maximize throughput
- Minimize the energy consumption
- If the nodes behave selfishly, they might not spend their energy in forwarding other nodes’ traffic
- Not forwarding any packets on the other hand adversely affect the network functioning

Game Theory

- The nodes trade-off energy consumption, network throughput and the collaboration offered to the network by the neighbors on packet forwarding
- Packet forwarding is an infinitely repeated game
- The players are the nodes
- Prior to choosing its next action, a node can analyze
  - Past behavior of its neighbors
  - Its priorities in terms of energy consumption and throughput

Uncertainty and Available Information

- A node doesn't know about
  - Actions of other nodes in the future
  - Future locations of other nodes
  - Traffic in the entire network
- Each node is endowed with the information on
  - Its neighbors and their past behaviors
  - Traffic it sent, has to send and received

How to Maintain Cooperation

- Incentive based techniques
  - Rewards and punishments
- Can cooperation without incentives exist?
  - What are the conditions

Generous TIT-FOT-TAT (GTFT) Model

- Time is slotted
- A finite population of N nodes among K classes
- n_i is the number of nodes in class $1 \leq i \leq K$
- ρ_i is the average power constraint of class *i*, $\rho_1 > \rho_2 \ldots > \rho_K$
- M is the maximum number of relays
- *q(l)* is the probability that the source requires *l* hops
- A session is of type *j*, if at least ones of the nodes involved belongs to class *j* and the class of any other node is less than or equal to *j*

- The source requests the relay nodes to forward its packet to the destination
- $\phi_h^j(k)$: The ratio of the number of relay requests for type *j* sessions made by node *h* which are accepted to the number of requests for type *j* sessions made by this node
- $\psi_h^j(k)$: The ratio of the number of relay requests for type *j* sessions made to node *h* which are accepted to the number of requests for type *j* sessions made to this node

Rational and Pareto Optimal Optimal Operating Point

- τ_{ij} the probability for a node of class *i* accepts a relay request for a session of type *j*.
- Average energy per slot spent by the node *p* as a source of type j session $e_{pj}^{(s)}$

$$e_{pj}^{(s)} = \frac{1}{N}\sum_{l=1}^{M}\sum_{h_1,\ldots,h_j} q(l)\Gamma(l; h_1,\ldots,h_j)\tau_{1j}^{h_1}\ldots\tau_{jj}^{h_j}$$

- Similarly, the average energy per slot spent by the node p as a relay is

$$e_{pj}^{(r)} = \frac{1}{N}\sum_{l=1}^{M}\sum_{h_1,\ldots,h_j} lq(l)\Gamma(l-1; h_1,\ldots,h_j)\tau_{1j}^{h_1}\ldots\tau_{jj}^{h_j}\tau_{class(p)j}$$

- The feasible region for τ_{ij} is defined by

$$\sum_{j=1}^{K}(e_{pj}^{(s)} + e_{pj}^{(r)}) \le \rho_{class(p)} \text{ and } \tau_{class(p)\,j} \in [0,1]$$

- The Pareto optimal values can be obtained by imposing the equality relation
- For self interested nodes $\tau_{ij} = \tau_{jj} = \tau_j$ $1 \le i \le j \le K$

GTFT Algorithm

- Nodes maintain a record of their past experience using $\phi_h^j(k)$ and $\psi_h^j(k)$ *and M=1*
  - *If* $\psi_h^j(k) > \tau_j$ *or* $\phi_h^j(k) < \psi_h^j(k)-\varepsilon$ *Reject*
  - *Else accept*
- For multiple relays define for node *h* of class *i*

$$L_{ij} = \frac{\text{Prob}(h \text{ is served in a type } j \text{ session})}{\text{Prob}(h \text{ accepts to relay a type } j \text{ session})}$$

- M-GTFT for a request for a type *j* session in class *i* node
  - *If* $\psi_h^j(k) > \tau_j$ *or* $\phi_h^j(k) < L_{ij}\psi_h^j(k)-\varepsilon$ *Reject*
  - *Else accept*

Nash Equilibrium

- If all nodes except node *h* employs m-GTFT, then

$$\limsup_{k\to\infty} \phi_h^{(j)}(k) \le \tau_j$$

- If all nodes employ m-GTFT, then all $\phi_h^j(k)$ converge to τ_j
- All nodes employing m-GTFT constitutes a Nash Equilibrium

An Approach Considering the Network Topology

- The preceding results assume a random connection setup
- The topology of the network is abstracted away
- The interactions are not random, they are determined by the network topology and communication pattern

Model

- Time is divided into slots
- Each node *i* chooses a cooperation level $p_i(t) \in [0, 1]$
- Assume there exists a route r with source node *s* and *l* intermediate nodes $f_1, f_2, \ldots, f_l$ in time slot *t*
- $T_s(r)$ is the constant amount of traffic *s* wants to send on *r*
- Define

$$\tau(r,t) = T_s(r)\prod_{k=1}^{l} p_{f_k}(t) \text{ and } \hat{\tau}(r,t) = \frac{\tau(r,t)}{T_s(r)}$$

Utility

- Payoff of the source node on r in t is $\xi_s(r,t)=u_s(\tau(r,t))$, where u_s is a non-decreasing function
- Payoff of the *j*-th intermediate node is

$$\eta_{f_j}(r,t) = -T_s(r)c\hat{\tau}_j(\mathrm{r},\mathrm{t}) \text{ where } \hat{\tau}_j(\mathrm{r},\mathrm{t}) = \prod_{\mathrm{k}=1}^{\mathrm{j}} p_{f_k}(t)$$

- The total payoff a node *i* in t is

$$\pi_i(t) = \sum_{q\in S_i(t)} \xi_i(q,t) + \sum_{r\in F_i(t)} \eta_i(r,t)$$

- Strategy Function

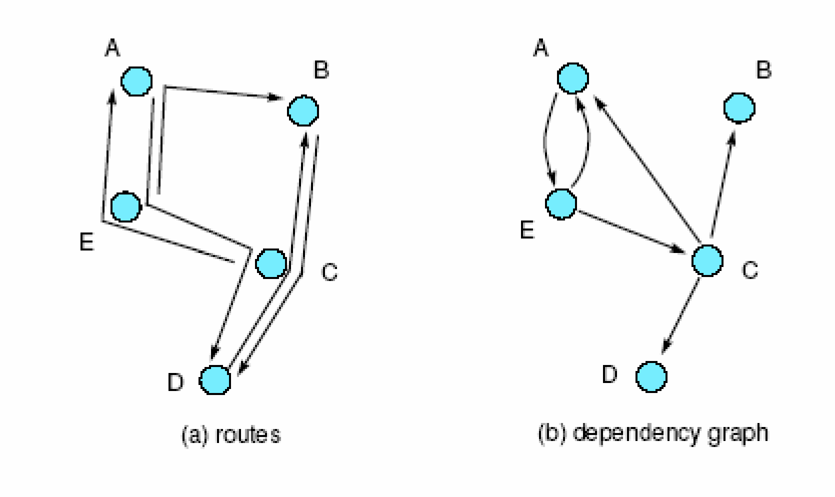
$$p_i(t) = \sigma_i\left(\left[\hat{\tau}(r,t-1)\right]_{r\in S_i(t-1)}\right)$$

Strategy Space

- Assume the input to the strategy function is a scalar
- Some possible strategies are
  - *Always Defect (AllD)* $\sigma_i(in)=0$
  - *Always Cooperate (AllC)* $\sigma_i(in)=1$
  - *Tit-For-Tat (TFT)* $\sigma_i(in)=in$

Dependency Graph

- Each vertex is a network node
- There is a directed edge from vertex *i* to *j* *(i,j)*, if there exists a route where *i* is an intermediate node and *j* is the source



Dependency Loop

- A dependency loop *L* for node *i* is a sequence $(i,v_1),(v_1,v_2),\ldots,(v_l,i)$
- Nodes have two types of dependency loops depending on the strategies played by other nodes
  - Reactive loops
  - Non-reactive loops
- If node *i* has no dependency loops, the cooperation level chosen by *i* has no effect on the normalized throughput it experiences

Resuts

- Φ is the set of all forwarder nodes
- If a node *i* is in Φ and it has no dependency loops, then its best strategy is *AllD*
- If a node *i* is in Φ and it has only non-reactive dependency loops, then its best strategy is *AllD*.

- If every node *j* ($j \neq i$) plays *AllD*, then the best response of *i* to this is *AllD*. Hence, every node playing *AllD* is a Nash Equilibrium.
- Assuming that node *i* is in Φ, the best strategy for *i* is to *AllC* in each time slot if
  - For every $r \in F_i$, there exists a dependency loop that contains the edge *(i,src(r))*
  - Every node in Φ (other than *i*) plays TFT
- If the first condition holds, all nodes playing TFT is a Nash equilibrium

Strategy

- The probability for the first condition to hold for every forwarder is very small (0)
- The nodes for which first condition holds play *TFT*
- The others play *AllD*
- The avalanche effect (effect of *AllD* players) is mild when the amount of traffic generated in the network is high.

An Incentive Based Mechanism

- A route between a source node *s* and the destination node *d* is of the form $\pi(s,d)=(s,n_1,\ldots n_k,d)$
- Node *j* forwards packets with a fixed probability, γ_j

$$p(s,d,\gamma)=\prod_{j=1}^{k}\gamma_j$$

- The set of intermediate nodes before n_j on this route is $S(s,d;n_j)=(n_1,\ldots,n_{j-1})$
- Define $p(s,d;i,\gamma)=\prod_{j\in S(s,d;i)}\gamma_j$

- Let *O(i)* denote all the paths in which node *i* is an intermediate node and let the rate at which source *s* creates packets for destinations *d* is given by λ_{sd}
- The rate at which packets arrive at node *i* to be forwarded is $\xi_i(\gamma)=\sum_{\pi(s,d)\in O(i)}\lambda_{sd}p(s,d;i,\gamma)$

- Let E_f be the total energy needed for forwarding a packet
- The utility of node *i* is

$$U_i(\gamma) = \sum_{n:(i,n)\in O} \lambda_{in} f_i(p(i,n;\gamma)) + \sum_{n:(n,i)\in O} \lambda_{ni} g_i(p(n,i;\gamma)) - aE_f \xi_i(\gamma)$$

- where f_i and g_i are utility functions that depend on the success probabilities associated with node *i* as a source and as a destination respectively

Punishing Policy

- Consider a given set of policies $\gamma_i = \gamma$ for all *i*
- For any choices of strategy γ for all nodes, define (γ_i', γ^{-i}) to be the strategy obtained when only player *i* deviates from γ to γ_i'
- All other nodes use a punishing policy and decrease their forwarding probability to γ'

Punishing Algorithm

- Time is slotted
- Every node computes its forwarding probability that maximizes the utility in the beginning of the slot
- As soon as a neighboring node is detected to misbehave, the node computes its forwarding probability as $\gamma_i^* = \min\{\gamma_i, \min_{j \in \mathcal{N}(i)} \gamma_j\}$
- The punishment propagates in the network until all the nodes settle to a common forwarding probability
- At the beginning of the new slot, nodes calculate their forwarding probabilities as if there is no misbehaving node

Conclusion

- It is not possible to force a node to forward more packets than it sends in average