Evaluating Algorithms in MANETs

The Server Determination Algorithm

Oliver Wellnitz

IBR
Technische Universität Braunschweig

March 12th, 2007

Overview

- Motivation
- Server Determination Algorithm
- Evaluation ideas
- Simulation
- Conclusions

Mobile Ad-hoc Networks

- Radio networks spontaneously created by mobile devices
  - Wireless radio networks (e.g. WLAN)
  - Radio networks are error-prone
- Mobile devices communicate directly with each other
  - No infrastructure needed, no communication costs
- Multi-hop communication for increased range
  - Cooperative behaviour required
- Users may freely roam around
  - Mobile ad-hoc network are subject to change at any time

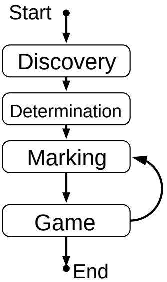
Motivation

- Game servers in mobile ad-hoc networks
  - Traditional single game servers are single points of failure
  - Fully distributed architectures may have problems with
    - Available bandwidth in mobile networks
    - Cheating & Control of the game
- Zone Servers:
  - Game servers for distinct areas of the network
  - Minimize traffic between zones
    - Zone servers can aggregate, delay, or drop traffic
  - Mixture between Client/Server and P2P
- Problem:
  - How many servers do we need?
  - Which mobile nodes should be chosen?
  - $\rightarrow$ Need for server determination algorithm

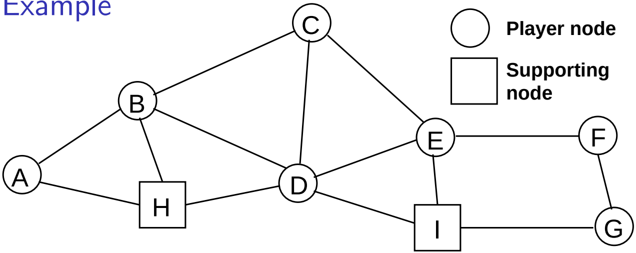
Overview of the Algorithm

Description

- Dominating set algorithm
- It comprises mainly 3 phases
    - Discovery Phase
    - Determination Phase
    - Marking Phase
- Mobility Management during the game



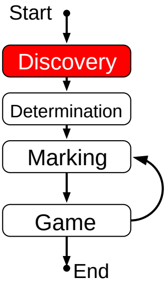
Example



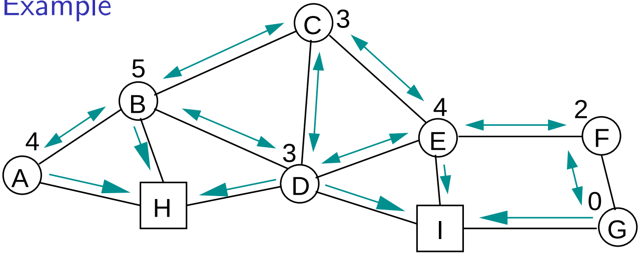
The Server Determination Algorithm (1/4)

Discovery Phase

- Determine available resources (weight)
  - May include CPU, RAM, battery, ...
- Determine number of neighbours (degree)
- Start periodically broadcasting this information to neighbouring nodes
- Build neighbour list based on received info



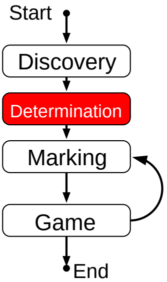
Example



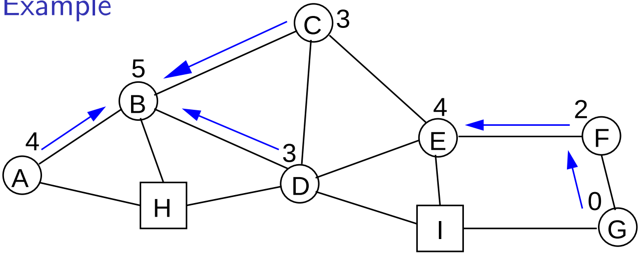
The Server Determination Algorithm (2/4)

Determination Phase

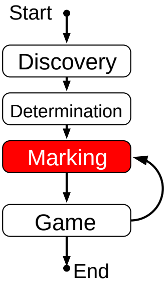
- Sort neighbour list
  - Order by weight, degree, node ID
- Calculate best server
  - First node on neighbour list with degree > 1
- Send ticket to best server



Example



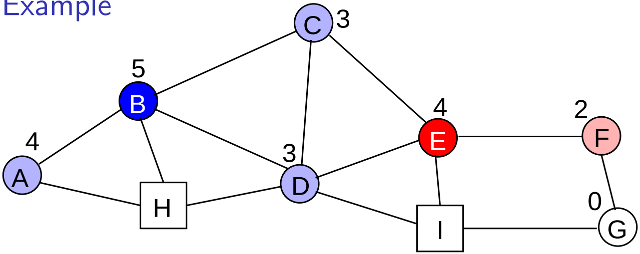
The Server Determination Algorithm (3/4)



Marking Phase

- Set tag
  - Server, Server neighbour, or Empty tag
- Add tag to the information broadcasted to neighbours

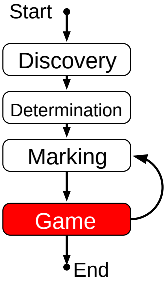
Example



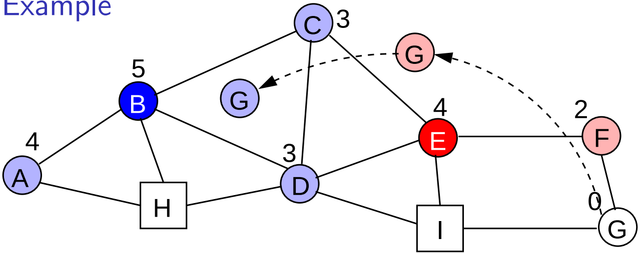
The Server Determination Algorithm (4/4)

During the game

- Continuous updates
  - Create new servers
  - Degrade servers back to players
  - Can be integrated in game protocol itself
- Mobility may require handoff to a better server
  - Decision to be made by the game
  - Threshold-based mechanism is advised



Example



Evaluation of the Algorithm

- Parameters of the algorithm:
  - Time between packet transmission, phase transition timeouts
- Server Synchronisation
  - Simple traffic forwarding
  - Region of Interest (allows for traffic reduction)
- Optimize parameters for
  - Packet size $\times$ traversed links (total traffic)
  - Fraction of available link capacity used
  - Minimal number of server
- Mobility
  - Handoff threshold, hysteresis
- Initial results look good, but evaluation is ongoing work

How to evaluate?

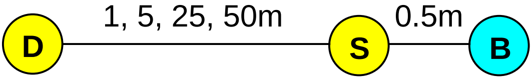
- WLAN measurements are not reproducible
- Simulations provide a good alternative
  - Scalable & reproducible, require computing instead of manpower
- But careful creation of simulation necessary
  - Number of nodes and initial placement
  - Environment and node mobility
  - Background traffic distribution
  - Verify the results (error estimation)
- Scenarios are indispensable to determine network parameters
  - Train stations, trains, busses, school yard
- First steps
  - Select simulator
  - Recreate simple scenario to evaluate simulators

Simulation of MANETs

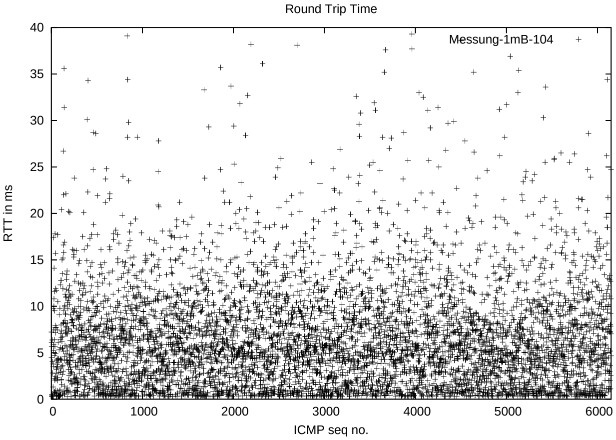
		Real world	NS-2 Std.	NS-2 INRIA
MAC	802.11	a/b/g	b	a
	Rate Adapt.	Various	–	ARF, AARF
	Retries	Multi-Rate	Same Rate	Same Rate
Radio model	Propagation	Realistic	FreeSpace TwoRay Shadowing	FreeSpace *Shadowing*
	Multipath	X	–	–
	Fading	X	–	–
	Reception	Realistic	SNRT	Energy, SNRT BER
	Noise	Realistic	Other signals	Thermal noise + oth. signals
Env.	Obstacles	LOS/NLOS	LOS only	LOS only

Scenario

- WLAN measurements
  - Goal: Realistic data as input for our simulation
- Experiment setup
  - 3 notebooks, Atheros-based WLAN PCMCIA card
  - Use notebooks as source, destination and for bg traffic
  - TCP connection for background traffic
  - ICMP echo request/response (ping) with various packet sizes
  - Measure round-trip-time, jitter and packet loss
- 104 byte data traffic @ 10 Hz + background traffic



Measurement

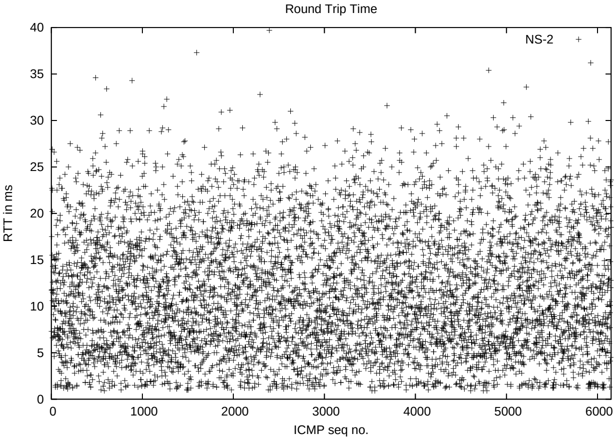


Average RTT
6.86 ms

Avg.Jitter
5.62 ms

Packet loss
0.15 %

NS-2 Std.

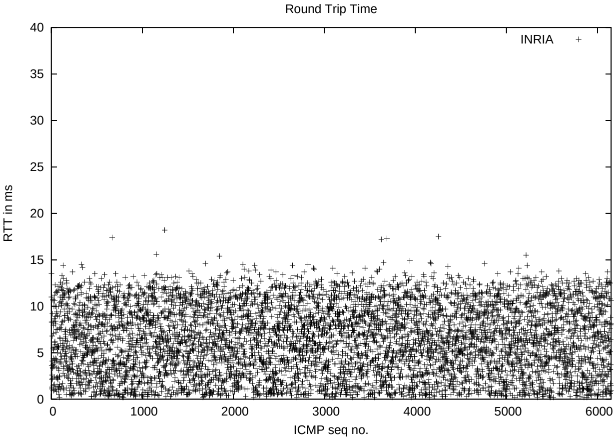


Average RTT
11.62 ms
(+4.76 ms)

Avg. Jitter
6.98 ms
(+1.36 ms)

Packet loss
3.7 %
(+3.55 %)

NS-2 INRIA + Shadowing



Average RTT

6.38 ms

(-0.48 ms)

Avg. Jitter

3.83 ms

(-1.79 ms)

Packet loss

0.0 %

(-0.15%)

Simulation Results

- NS-2 Std. only supports speeds of up to 11 MBit/s
  - RTT is too high (... even for 802.11b)
  - All packets are retransmissions!
  - Works fine in unimpaired environments
- NS-2 INRIA
  - BER-based reception model is more accurate than SNRT
  - Average RTT at measured level
  - MAC layer retries are not implemented
  - Significantly slower than NS-2 Std.
- ACKs are sent at the same speed as data
- Lessons learned
  - 802.11a measurements are easier than 802.11b/g
  - NS-2 INRIA seems more useful than NS-2 Std.
  - Retries are essential for realistic behaviour

Conclusion

- Game in MANETs require redundancy
- Distributed Server Determination
  - Dominating set algorithm
  - Scalable approach with low overhead
  - Initial evaluation shows good results
- Recreating simple scenarios is not straightforward
  - Even the widely-used NS-2 simulator fails in our scenario
- Essential simulator features
  - Layer 2 retries are necessary for realistic latencies
  - Rate adaptation useful to determine link capacity
  - BER mechanism is superior to SNRT but more expensive

Questions? Comments?

Evaluating Algorithms in MANETs

The Server Determination Algorithm

Oliver Wellnitz <wellnitz@ibr.cs.tu-bs.de>