MOTION

A Formal Model for the Simulation of Mobile Networks

Emanuele Covino
Dipartimento di Informatica, Universitá degli Studi di Bari, Italy

A short resume

Emanuele Covino is an Assistant professor at the
Dipartimento di Informatica, Universitá degli Studi di Bari, Italy.

Research: Implicit computational complexity, Template metaprogramming and partial evaluation, Mobile networks.

Teaching: Foundations of computer science, Computability and complexity, Programming languages, Web programming, Algorithm and data structures.

Projects: Erasmus+ Computing Competences: "Innovative learning approach for non-IT students" (agreement n° 2018-1-PL01-KA203-051143);
Horizon Europe Seeds: "Freedom of speech, new technologies, and consensus formation";
"Computational complexity of Generic programming".

Table of contents

Mobile Ad-hoc NETworks (MANET) and routing protocols

Mobile Ad-hoc NETworks (MANET)

A mobile ad hoc network (MANET) or a wireless ad hoc network (WANET) is a

- self-configuring
- self-organizing
- infrastructure-less

network of mobile devices, with no physical connections.

Each device participates (dynamically) in routing by forwarding data for other nodes, depending on network connectivity and on the routing algorithm in use.

More on MANETs

Some obvious advantages of MANETs over centralized networks, like

- mobility
- self-organization and scalability
- flexibility (rapid deployment and low-cost infrastructures)
- robustness and reliability in critical conditions

provide the scope for deployment in applications such as

- environmental monitoring
- disaster relief
- military communications
- VANET (Vehicular mANET)
- SPAN (SmartPhone Ad-hoc Networks)

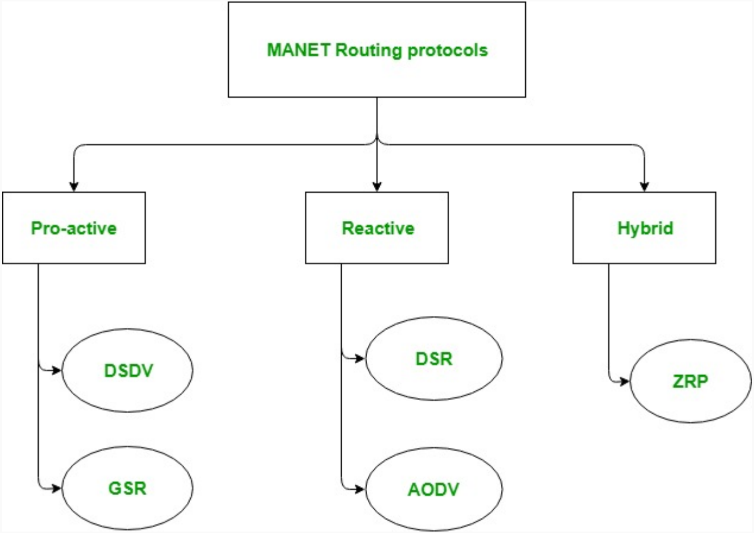
Routing protocols (1)

When building a MANET, each device must maintain the information required to properly route the traffic. This is done by means of

1. proactive protocols (nodes find routes by exchanging network topology information, periodically)
2. reactive protocols (nodes discover the routes only when it's needed)
3. hybrid

that need to limit the search into the state's space.

Routing protocols (2)



Routing protocols (3): Ad-hoc On-demand Distance Vector , AODV

It's the combination of Dynamic Source Routing (DSR) and Destination-Sequenced Distance Vector (DSDV).

- quick adaptation to dynamic link condition
- low CPU consumption and memory overhead
- low network utilization (pure on-demand route acquisition)
- loop free by using destination sequence numbers

Several types of AODV

1. Reverse-AODV
2. Secure-AOVD or Trusted-AODV
3. Not ACKnowledgment AODV (NACK-AODV)
4. Blackhole-free NAODV (BN-AODV)
5. ...

Routing protocols (4): AODV routing table

In AODV, nodes discover routes in request-response cycles. Each node maintains a routing table, containing

1. a **next hop node:** all packets destined to the destination are sent to this node;
2. a **sequence number:** acting as a form of time-stamping, and is a measure of the freshness of a route;
3. and a **hop count:** it represents the current distance to the destination node.

The AODV protocol maintains the following property:

If a and b are nodes, and b is the next hop of a to some destination d; if the sequence number and hop count of the routes to d at a and b are $(seq_a, hcnt_a)$ and $(seq_b, hcnt_b)$, respectively.

$$(seq_a < seq_b) \vee (seq_a = seq_b \wedge hcnt_a > hcnt_b)$$

b either has a newer route to d than a, or b has a shorter route that is equally recent.

Routing protocols (4): AODV operations

Three types of messages defined for AODV:

1. **Route Request - RREQ.** A node requests a route to a destination by broadcasting a RREQ message to its neighbours; they forward the request until it reaches a node with a route to the destination.
2. **Route Reply - RREP.** This node responds with a RREP, which contains the number of hops to reach the destination and the sequence number for the destination most recently seen by the node generating RREP.
3. **Route Error - RERR.** If a node looses connectivity to the next hop, the route is invalidates by sending a link failure message to all nodes that received its RREP.

On receipt of the three messages, the nodes update their next hop, sequence number and hop count in such a way as to satisfy the partial order constraint mentioned above.

Routing protocols (5): Not ACKnowledgment AODV, N-AODV

A node *n* is aware of the existence of a node *m* only when *n* receives a RREQ that originates by (or is directed to) *m*.

1. if a RREQ originated by *n* and directed to *m* is received by *p*, *p* sends the NACK packet back to *n*;
2. *n* (as well as all the nodes in the path to it) receives fresh information about the existence and the position of *p*; on receiving the NACK, all the nodes in the path to *p* add an entry in their routing tables, or update the pre-existing entry.

N-AODV improves the network awareness of each node; it has been validated through simulations, showing that the number of RREQ decreases, wrt the AODV protocol.

Routing protocols (6): Blackhole-free N-AODV, BN-AODV

When a malicious node detects an RREQ message, it sends a false route reply message (RREP) back to the sender, *with the maximum sequence number* before other nodes send an actual true one. The sender of RREQ thinks that the route discovery is accomplished and begins to transmit packets to the malicious node.

The Black hole-free N-AODV protocol uses two control packets:

1. each node *n* receiving an RREP verifies the trustworthiness of the nodes in the path followed by the RREP, producing a *challenge packet* (CHL) for the destination node; only the destination can produce the correct *response packet* (RES);
2. if *n* doesn't receive RES, the next node towards the destination is a potential black hole.

Simulators Vs Formal methods

Simulators (software-based)

Used for

1. evaluation of performance
2. compare different solutions
3. stress test of the network
4. easy monitoring on results

CONS

1. no network simulator is accurate
2. strong assumptions on node mobility
3. simulation size
4. implementation only at low abstraction level
5. each simulator has its own language

Formal methods

They provide a formal definition of the MANET

1. process calculus (Singh et al.)
2. CMN - calculus for mobile ad-hoc networks (Merro)
3. AWN - algebra for wireless networks (Fehnker et al.)
4. Petri nets (Erbas et al., Xiong et al.)

CONS

1. a gap between the abstract model and the actual level at which the MANET has to be analyzed
2. various physical aspects are omitted
3. realistic potential problems may be abstracted away when not reflected in the formal model

Abstract State Machines for MANETs with Asmeta

AODV by Börger and Raschke

"We use [AODV] protocol as case study to illustrate how, by stepwise developing the components of an ASM model, one can explain complex intended system behavior from scratch, gently but accurately, supporting a correct understanding of the requirements by the programmers and of the high-level system behavior by the users of the system."

Börger and Raschke, *Modeling Companion for Software Practitioners*

Asmeta

Asmeta is a framework for the Abstract State Machines method.

1. it's composed of different tools: editor, simulator, validator, animator, model checker, ...
2. it is based on the definition of a metamodel (AsmM)
3. uses a concrete syntax (AsmetaL) as a notation to write ASM models in a textual form
4. and an interpreter (AsmetaS) to execute AsmM models.

MOTION: MOdeling and simulaTIng mObile ad-hoc Networks

Putting all together

- We define AODV (as well as N-AODV and BN-AODV) within the ASM model,
- using AsmetaL to define the network protocol and AsmetaS to run the model,
- and introducing a visual interface that shows the progress of the simulation.

How MOTION works

The executions of MOTION and ASMETA are interleaved:

1. MOTION captures the parameters of the network (number of nodes, level of mobility) and includes them into an AsmetaL file;
2. it runs AsmetaS (executing an ASM move over the network's configuration);
3. saves the information related to the move (new positions of the nodes, sent/received requests, relations among the nodes);
4. visualizes the current topology of the network (shows the successful communication attempts between pairs of nodes, the connections established, and the failed attempts);
5. repeats from 2;

At the end of the simulation, MOTION reads the final log file, parses it, and stores the collected results in a csv file.

MOTION: examples of rules for AODV (1)

```
MAIN RULE AODV =
    forall a ∈ Nodes do AODVSPEC(a)


AODVSPEC(a)=
  forall dest ∈ Nodes with dest ≠ a do
    if WaitingForRouteTo(a, dest) then
      if Timeout(a, dest) > 0 then
        Timeout(a, dest) := Timeout(a, dest)-1
      else
        par
          WaitingForRouteTo(a, dest) := false
          ca-fail(a, dest) := ca-fail(a,dest)+1
        endpar
      endif
  if WishToInitiate(a) then PREPARECOMM(a)
  if not Empty(Message) then ROUTER
```

MOTION: examples of rules for AODV (2)

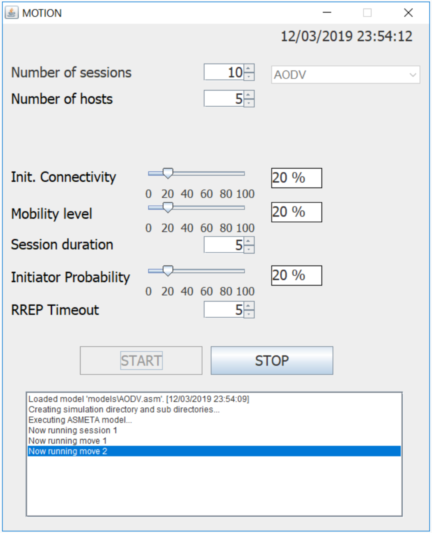
```
PREPARECOMM(a) =
  forall dest ∈ Nodes with dest ≠ a do
    choose wantsToCommWith ∈ Boolean with true do
      if wantsToCommWith then
        par
          if not waitingForRouteTo(a,dest) then
              ca-tot(a, dest) := ca-tot(a, dest) + 1
          endif
          if knowsActiveRouteTo(a,dest) then
              par
                StartCommunicationWith(dest)
                waitingForRouteTo(a, dest) := false
              endpar
          else
              if not waitingForRouteTo(a, dest) then
                par
                  GenerateRouteReq(dest)
                  WaitingForRouteTo(a, dest) := true
                  Timeout(a,dest) := Timeout
                endpar
              endif
          endif
```

MOTION: examples of rules for BNAODV

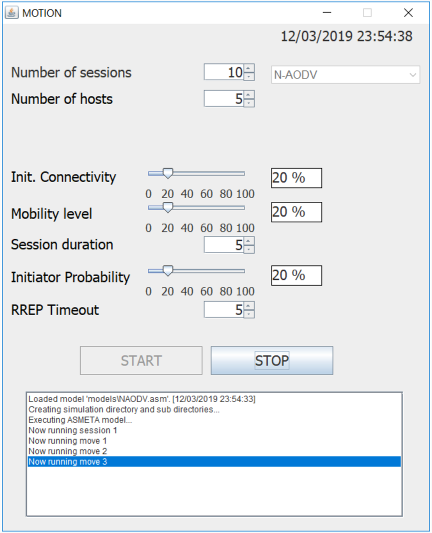
```
main rule rmain =
    forall bh in Blackhole do rBlackHoleProgram(bh)
    forall c in Colluder do rColluderProgram(c)
    forall a in Honest do rHonestProgram(a)

rule rBlackHoleProgram(bh) =
  if(notEmpty(Message)) then
    let(queue = {m in Message | messageType(m) = RREQ and isLinked(self,sender(m))
      and isConsumed(self,m)=false : m} rreq = chooseone(Message)) in
      while(notEmpty(queue)) do
        seq
          rreq:= chooseone(queue)
          queue := excluding(queue, rreq)
          par
            if (hasNewReverseRouteInfo(rreq)) then rBuildReverseRoute[rreq]
            endif
            rGenerateRouteRep[rreq]
            maliciousoverhead(self) := maliciousoverhead(self)+1
            rConsume[rreq]
          endpar
        endseq
    endlet
  endif
```

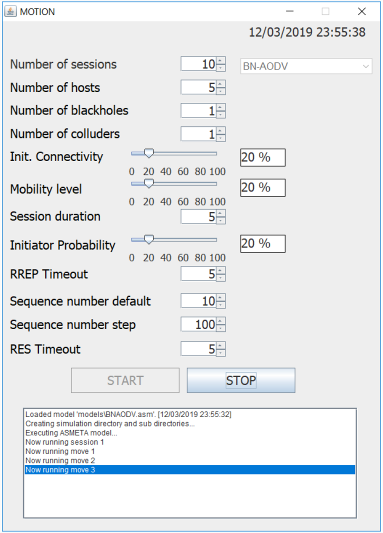
MOTION: the interface for AODV



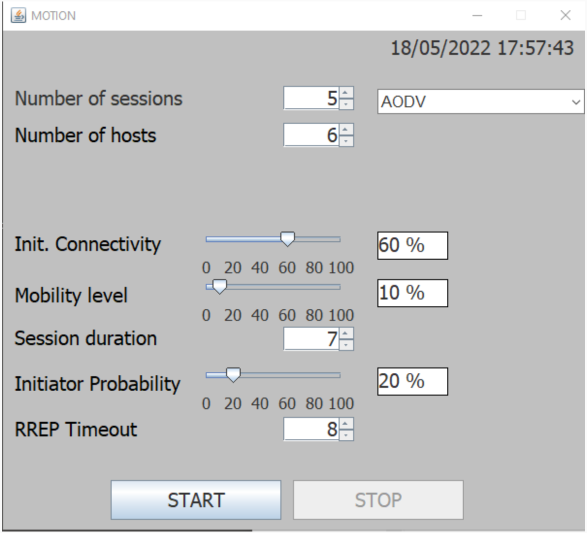
MOTION: the interface for NAODV



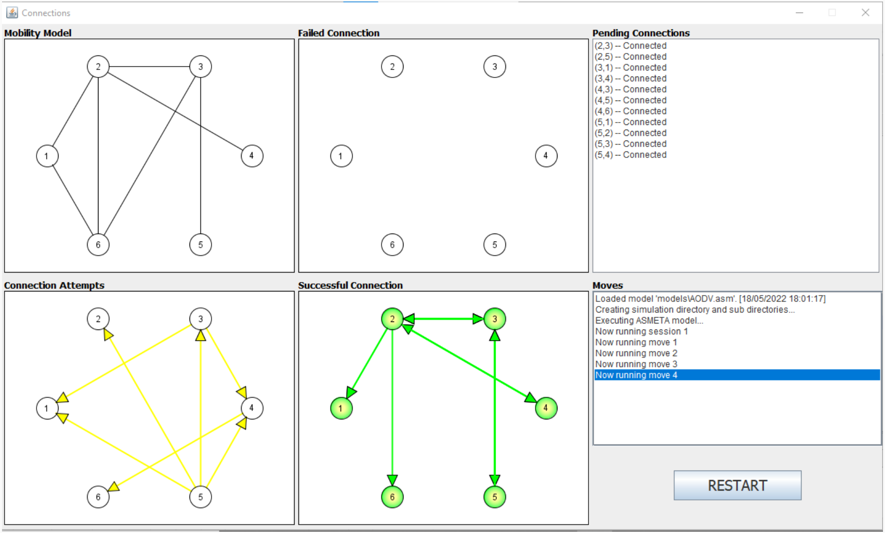
MOTION: the interface for BNAODV



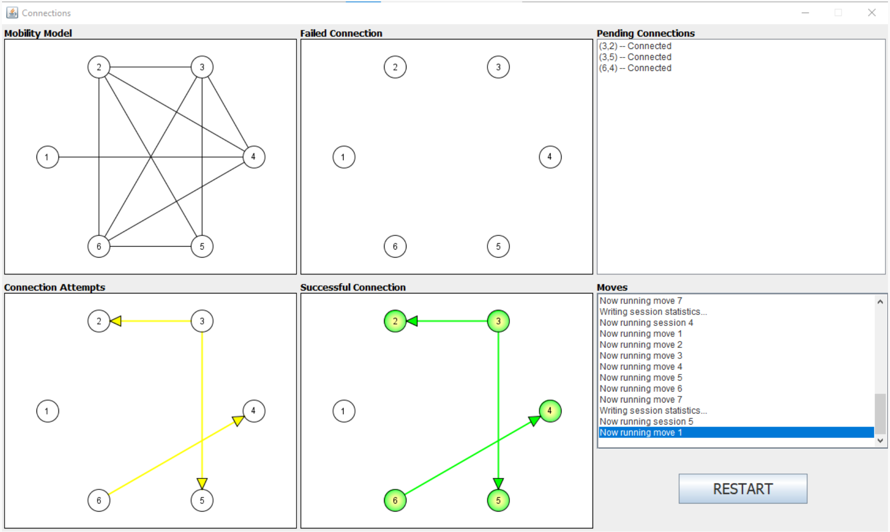
MOTION: a new interface for AODV



MOTION: visual evolution of the network (1)



MOTION: visual evolution of the network (2)



An interesting application: Social network's analysis

Social network analysis

Social structures can be investigated by means of methods and tools of *social network analysis*, a key technique in modern sociology, demography, communication studies, market economy, sociolinguistic, cooperative learning.

Graphs (or networks) are often used, with

- *nodes* associated to people or agents, and
- *arcs* representing any kind of relation, interaction or influence between pairs of agents.

Many studies are executed with simulators, in order to compare different social structures and several scenarios, according to the parameters of the network.

Social network analysis

MOTION (as well as other models of mobile networks' protocols) provides *methods* to define these kind of networks, and *algorithms* that allow to broadcast a message from a source to a destination, mimicking the spread of information, opinions, or consensus into a group of social agents.

This tool could be used by social scientists to represent a social group and to study the related interactions.

For instance,

- a high value of the *initial connectivity* parameter and a low level of *mobility* represent strong ties within a very cohesive group;
- a high *mobility* means that the group is prone to change opinions very easily;
- the *initiator probability* measures how much a member of a social group is inclined to spread information inside the network.