A Framework for a Comprehensive Evaluation of Ant-Inspired Peer-to-Peer Protocols

Amos Brocco
Department of Innovative Technologies, University of Applied Science of Southern Switzerland

Ingmar Baumgart
Institute of Telematics, Karlsruhe Institute of Technology (KIT), Germany
Outline

- Introduction & motivation
  - Bio-inspired protocols
  - The need for a simulation framework
- OverSim overlay framework
- The OverSwarm extension
- Evaluation example: BlåtAnt
- Conclusion
Introduction

Scenario: complex P2P distributed systems

A Framework for a Comprehensive Evaluation of Ant-Inspired Peer-to-Peer Protocols
Introduction

Bio-inspired solutions

- self-organization
- emergence
- fault-tolerance
Examples of bio-inspired solutions

- Messor - load balancing
- AntNet, AntHocNet - routing
- Self-Chord, Self-CAN - DHT
- BlåtAnt - overlay management
- Antares - clustering
- SemAnt - resource discovery
Motivation

- We want to validate bio-inspired P2P protocols based on the paradigm of social insects (ant colonies)
  - simplify their implementation
  - simulation under realistic conditions
  - reproducibility of results
  - comparison with “traditional" solutions
The problem

...out of 36 considered publications in the field of bio-inspired network systems, 19 present results obtained using custom simulators.

Validation and comparison is difficult!
How to proceed?

A Framework for a Comprehensive Evaluation of Ant-Inspired Peer-to-Peer Protocols
What is OverSim?

- OverSim is an overlay framework
  → providing common functions for rapid prototyping

- Once a protocol is implemented in OverSim, you can...
  ...use OverSim as an overlay simulator
    → simulations with up to 100,000 nodes
    → strong GUI support (e.g. for debugging)
    → gathering of statistical data

...run your protocol in real networks
  → deployment in PlanetLab

...emulate an overlay network
  → connection to real network devices
  → for demonstrational purposes

→ all without any code modification!
OverSim in a Nutshell

- Based on OMNeT++
- OverSim runs on
  - Linux, Windows, Mac OS X, Nokia Internet Tablets (Maemo)
- Written in C++
- Well documented
- Open Source (GPL)
- Actively used by community
  - more than 10,000 downloads and over 200 citations
- OverSim already includes a large number of structured, unstructured, and event distribution protocols
Flexible Architecture

- Layered architecture
  - Underlying network
  - Overlay layer
  - Application layer

- Consistent interfaces between layers
  - UDP between network and overlay
  - Common API between KBR overlay and application

→ Exchange of one component is transparent to all other components
Underlay Abstraction

- OverSim offers different underlay models
- Simple Underlay
  - Low computational overhead
  - Coordinate-based delays calculated from CAIDA/Skitter measurements
  - Logical access network
- INET Underlay
  - Based on the INET framework
  - Complete IP stack is modeled
  - Backbone simulation
  - Extendable by INET framework models, e.g. 802.11
Visualization and Debugging

- Topology visualization
- Message visualization
- Node state debugging

A Framework for a Comprehensive Evaluation of Ant-Inspired Peer-to-Peer Protocols

Institute of Telematics
Modeling of Churn

- Several Churn models provided, based on different lifetime distributions
  - Weibull
  - Exponential
  - Pareto

- Simulations with several churn generators possible
  - Used to generate different types of nodes
    - e.g. overlay partitions, landmarks, i3-server
Statistics

- Central model for gathering statistics
- Post-processing scripts for statistical data
  - facilitate generation of publication quality plots: scatterplots, vectorplots, lineplots
  - gnuplot compatible output
The missing piece

A Framework for a Comprehensive Evaluation of Ant-Inspired Peer-to-Peer Protocols

Institute of Telematics
The OverSwarm framework

- Rapid development and evaluation of bio-inspired protocols
- Extension for the OverSim framework
  - Benefits from OverSim/OMNeT++ features, like a flexible architecture, GUI, …
  - Developed as separate project to keep OverSim codebase small
  - Supports agent migration, pheromone trails and provides its own programming language
Support for different architectures

**Standalone Protocol**

- Application
- **Swarm Based Overlay**
- Underlay

Example: BlåtAnt, …
Support for different architectures

Example: Ozmos, …
Support for different architectures

Example: Self-Chord, …
Bio-inspired features

- Support for strong, transparent migration
  - Traditional: Nodes exchange messages
  - Different view for bio-inspired protocols
    - Agents (“ants”) are migrated between nodes
  - Migration process
    - Suspend execution of agent
    - Transfer complete state (e.g. stack) to new node
    - Resume execution

- Pheromone management
  - Ant-inspired protocols use pheromone trails to mark paths
  - OverSwarm provides various reinforcement and evaporation models for pheromone trails
Toolchain

Programming language (Lisp-like)
- dynamic typing, automatic memory management
- strong, transparent migration
- pheromone API

Compiler (agent language -> C++)
Programming

Ant Agent's Behavior

With probability 50% the agent will migrate to the successor, otherwise it migrates to predecessor. In case it doesSomething, then it also migrates to the successor. If the result of doingSomething was greater than 0, then doThis; otherwise doThat.

OMNeT++ / OverSwarm
Benchmark: simulation time

- OverSwarm
- OverSim
- PeerSim

Simulation time (seconds)

Number of nodes

- 100
- 1000
- 10000

OverSwarm: 9, 104, 1323
OverSim: 5, 46, 646
PeerSim: 2, 13, 159
Benchmark: memory usage

![Graph showing memory usage with different protocols and node counts.](image-url)
Example: BlåtAnt

Overlay management protocol

Self-structured overlay

bounds distances between each pair of peers
limits small cycles
pure peer-to-peer (no superpeers, hubs)

Ant-like mobile agents
Example: BlåtAnt

For a user defined parameter D

Create and remove logical links:

**Connection Rule**
connect two nodes if distance (hops) ≥ 2D – 1

**Disconnection Rule**
break cycle if length < 2D - 2
Example: BlåtAnt

Discovery ants collect and disseminate information across nodes...
Example: BlåtAnt

Infer **minimal distances to** (non-neighbor) nodes

- **Connection Rule:** if distance $\geq 2D - 1$, create a new connection

![Diagram of connections between nodes](image)

- $d(W,F) = 2$
- $d(E,F) = 3$
- $d(S,F) = 3$
- $d(O,F) = 2$

= neighbor of F
Example: BlåtAnt

Infer **minimal distances between** (neighbor) nodes in the overlay (not crossing the node itself)

- **Disconnection Rule:** if distance < 2D-2, disconnect one of the neighbors

\[ d(A,M) = 5 \]

\[ \text{F} \quad \text{A} \quad \text{W} \quad \text{E} \quad \text{S} \quad \text{O} \quad \text{M} \quad \text{O} \]

\[ \square = \text{neighbor of F} \]
Example: BlåtAnt

```c++
(while 1 (begin
  (if (<= steps 0) (break))

;; Pass the information to the node
(inform vector)

;; Address of the current node
(var currentNode (getThisNode))
(push vector currentNode)

;; Trim vector (if necessary)
(if (> (len vector) vectorlength) (erase vector 0))

;; Now, choose next step
(var nextStep nil)
(var candidates (getNeighbors))
;; Remove previously visited nodes
(foreach v in vector (begin
  (remove candidates v)))

;; No good destination? Forget all information, and start over
(if (= (len candidates) 0) (begin
  (set! candidates (getNeighbors))
  (set! vector []))

;; Count this step
(set! steps (- steps 1))
;; Exploration or exploitation?
(if (< (random) kappa)
  (migrate (getLowestGammaTrail candidates))
else
  (migrate (choose candidates))))

"alpha->inform" as "inform",

```c++

```c++
osvPtr antFunction(Alpha::inform, osvPtr vector)
{
  OsvList v = unwrapOsvList(vector);
  return wrap(inform(v));
}

bool Alpha::inform(OsvList& info)
{
  int distanceFromNeighbor = 2;

  if (info.empty()) return false;

  int count = (info.size() - 2);

  for (int i = count; i > 0; i--) {
    TransportAddress el = unblobTransportAddress(info.at(i));
    if (el.isUnspecified()) {
      continue;
    }
    if (isNeighbor(el)) {
      distanceFromNeighbor = 2;
    } else {
      Entry& e = table->getOrNew(el);
      e.distance = distanceFromNeighbor;
      e.entryAge = getTimeStamp();
      distanceFromNeighbor++;
    }
  }

  distanceFromNeighbor = info.size();
```
Example: network overhead

BlåtAnt: max 8 neighbors
GIA: min 3, max 20 neighbors
Example: Dealing with packet loss

~ 512 Nodes / Weibull lifetime churn (10'000s mean, k=1)

10% packets lost
BlåtAnt: max 8 neighbors
GIA: min 3, max 20 neighbors

100.0%
90.0%
80.0%
70.0%
60.0%
50.0%
40.0%
30.0%
20.0%
10.0%
0.0%

500 1500 2500 3500 4500 5500 6500 7500 8500 9500 10'500

Time (seconds)

% Nodes in LCC

Sent Maintenance Bytes/s per node

120
100
80
60
40
20
0

BlåtAnt
GIA
Chord

Bytes/s
Example: number of edges

~ 512 Nodes / Weibull lifetime churn (10'000s mean, k=1)

- No packet loss
- 10% packet loss
Conclusions

- Lots of bio-inspired protocols proposed, but no widely accepted simulation tool available
- OverSwarm is an extension to the well-known OverSim framework
- Supports rapid prototyping and evaluation of bio-inspired protocols
  - Transparent agent migration
  - Pheromone based communication
- Solution to compare and validate bio-inspired with traditional peer-to-peer protocols
Give it a try…

http://syscall.org/doku.php/overswarm

Several protocols are included: BlåtAnt (overlay management), Ozmos (load balancing), Self-Chord (DHT), Messor (load balancing), Peer Sampling Service (Newscast, etc.), Cyclon,...