The Anthill Project

Part II: The Anthill Framework

Alberto Montresor

University of Bologna
Department of Computer Science

Preface

- **The IT community**
  - has recognized that most of the efforts in the peer-to-peer arena have to confront with common problems
  - is promoting several P2P initiatives:
    - P2P-WG: development of new standards
    - JXTA: open-source infrastructure
    - OpenP2P: repository of information about P2P
- **These initiatives are sponsored by the industry:**
  - their support is limited to the development of new protocols and standards
  - they do not support scientific investigation of the properties of these systems

The Anthill Project

- **The Anthill project was born on the basis of these considerations:**
  - We want to support researchers in the *study*, the *design* and *analysis* of P2P systems
- **In order to pursue our goal:**
  - We have introduced a new approach for designing P2P systems, based on the *ant colony* paradigm
  - We are developing a framework supporting this approach, enabling researchers to design and test new P2P algorithms

Anthill: Why a New Approach?

- **P2P networks can be described as *complex adaptive systems***
  - *a P2P system is composed of a collection of simple elements which cooperate through a small set of interactions*
  - *yet a P2P system exhibits a complex behavior that goes far beyond the capabilities of its components*
- **Unfortunately, only traditional techniques are being applied to them:**
  - Multicasting in Gnutella
  - Centralized broker in Napster
- **In our opinion, techniques borrowed from complex adaptive systems may be more effective in dealing with the inherent complexity of P2P systems**

Anthill: Why a Framework?

- **Our framework helps researchers in the design of new protocols**
  - provides a set of interfaces to be implemented and used
- **Our framework provides researchers with an infrastructure supporting the implementation of P2P protocols**
  - provides basic implementations of the main interfaces
- **Our framework support researchers in the analysis of the behavior of P2P protocols**
  - simulation tools

Why Ants?

- **The Anthill project builds upon the similarities between P2P systems and social colonies of ants**
- **A "live" example:**
  - Ant colonies are generally competing for the control of the territory
  - There are ant species, however, in which large networks of interconnected but autonomous social units cooperate
  - *Formica Yessensis*
    - A super-colony of Yessensis is reported to live on the coast of Japan since 1979
    - 45.000 interconnected nests, 1.080.000 queens, 306.000.000 workers, 270 hectares

Ant Colony Algorithms

- **Agent-based**
  - Artificial ants of limited individual capabilities move across a network of nodes trying to solve a particular problem
  - While moving, they build (partial) solutions and modify the problem representation by adding collected information
- **Complex adaptive**
  - Individual ants are unintelligent and have no problem solving capability
  - Nevertheless, ant colonies manage to perform several complicated tasks

A Parallel Between Ant Colonies and Anthill

Ant colonies	*Anthill systems*
Real ants need food to survive	Anthill ants need resources to survive (i.e., documents, cpu, etc)
Environment composed of nests, food sources and land	Anthill networks composed of nests (which are both nest and food sources) and links between them
Real ants move in the environment in the search of food	Anthill ants move across the network looking for resources
Pheromone trails linking nests and food sources	"Information" trail linking nests and pointing to resources
Real ants move food from the source to their originating nest	Anthill ants (may) copy a resource from nest to nest

Anthill Goals and Applications

- **The aim of the Anthill project is:**
  - to support scientific investigation of peer-to-peer systems
  - to support researchers in the *study*, the *design* and *analysis* of P2P systems
- **For which P2P applications?**
  - Content-sharing networks
  - Storage-sharing networks
  - CPU-sharing networks
  - Any combination of these three
- **For the moment, we are not interested in human presence**

Anthill Description

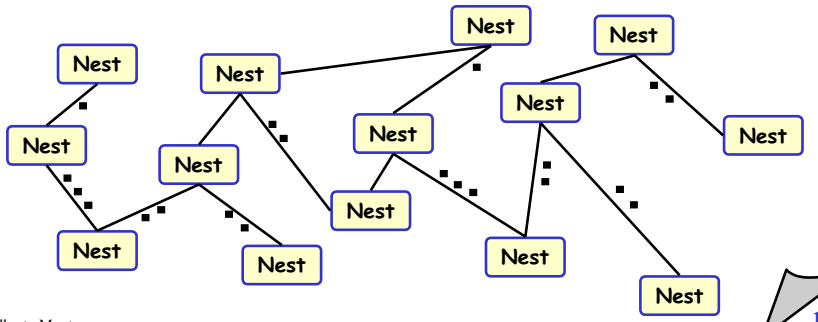
- **The Anthill project is based on Java**
- **Why Java?**
  - Portability
  - Possibility of remotely download code on demand
  - Rich security features
    - Execution sandboxing
    - Cryptography extensions
  - Possible future integration with JXTA

The Anthill Framework

- **The Anthill framework is composed of four main elements**
  - The *infrastructure*
    - Supports the development and the execution of ant algorithms
  - The *ant algorithms*
    - Written by researchers in order to implement P2P protocols aimed at performing a particular task
  - The *evaluation framework*
    - Helps researchers in the evaluation of new protocols
  - The *evolution framework*
    - Helps researchers to obtain new ant algorithms through evolution techniques

The Anthill Infrastructure

- **The Anthill Infrastructure:**
  - is composed by a *network* of *nests* and *links* between nests
    - nests are P2P applications run by users on their machines
    - two nests are linked when they know each other



Nests

- **Each nest is capable of:**
  - performing computations
  - storing information
  - communicating with other nests
    nest communicate through ant exchange
- **A nest can act on behalf of:**
  - its user
  - foreign ants coming from nests run by different users
- **Each nest is associated with a unique *nest identifier***
  - In Internet: IP address + port number

Nest Networks

- **Neighbor nests:**
  - A nest *A* that knows the identifier of another nest *B* may communicate directly with it
  - We say that:
    - B is *neighbor* of A
    - A is *connected* to B
- **The set of neighbors of a nest may be dynamic**
  - new neighbors may be added at run-time by notifying their identifiers to the nest

Requests

- **A user may ask its nest for specific services by:**
  - performing requests
  - listening for specific responses
- **The Anthill API**
  - does not impose any particular format for requests and responses
  - does not specify which services a nest should provide
- **Request interpretation and satisfaction are delegated to ants**
  - P2P services may be developed by implementing new ant algorithms based on the facilities offered by the Anthill infrastructure

Examples of Requests

- **In a music sharing network:**
  - A request could be a query for the songs of an artist
  - The response could contain:
    - a set of URLs from which these songs may be downloaded
    - the songs themselves (for anonymity)
- **In a distributed computing project (ex. the flu vaccine)**
  - A request could be a query to perform some computations, such as a virtual experiment
  - The response could be the result of the computation

The Nest Interface

- **Users interact with a nest through the Nest interface**

```
public interface Nest {
   NestId getId();
   void request(Request request, ResponseListener listener);
   void addNeighbor(NestId id);
   void removeNeighbor(NestId);
   NestId[] getNeighbors();
}
```


Nest implementation

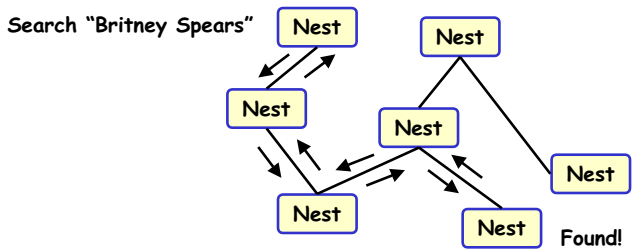
- **Nest implementations consist of three modules:**
  - a document storage
    - responsible for storing documents
  - an ant manager
    - takes care of scheduling computations of visiting ants
  - a gateway
    - the communication module, responsible for:
      - sending/receiving ants
      - monitoring the reachability of remote nests
- **Nest implementations constitute the infrastructure used to build ant algorithms**

Ant Algorithms

- **Ants are autonomous agents capable of:**
  - moving across a nest network
  - interacting with the nest they visit to pursue their goals
- **Individual ants are characterized by:**
  - their algorithm ("species")
  - a small amount of memory containing the ant's state
- **The behavior of an ant:**
  - depends on its algorithm and its current state
  - may be non-deterministic
    - i.e., probabilistic

Ant Algorithms

- **Ants are generated by nests in response to requests:**
  - ants try to satisfy the requests for which they have been generated
  - ants move from nest to nest until they reach their goal
  - successful ants returns to the nest that generated them in order to deliver a response to the user



Ant Algorithms

- **Ants implements the `Ant` interface:**

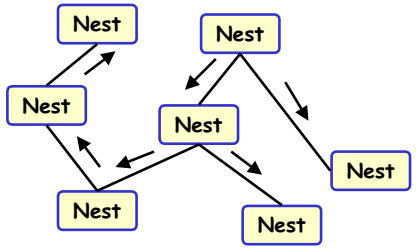
```
public interface Ant {
   void run(AntView view);
}
```

- **The generic `run()` method**
  - contains the ant's algorithm
  - is invoked by the ant manager of the current nest hosting the ant
  - after the algorithm completes, the local copy of the ant is removed from the current nest

Ant Movements

- **When an ant moves from a nest to another:**
  - The ant state is transmitted from the source nest to the destination, where the ant is re-created
  - If not known, the ant algorithm is transmitted as well
  - Ants may move from one source to multiple destinations
    - In this case, the ant is replicated

A Gnutella-like ant



Ant Views

- **Executing downloaded code is dangerous**
- **Ants and security**
  - The ant manager is responsible for confining the ant execution to a controlled environment:
    - code signing
    - no access to valuable resources (as disks, I/O, net)
    - possible operations are limited
- **The only methods that can be invoked are those contained in interface AntView**

Ant Views

- **Operations permitted to ants:**
  - move to another nest specified by its identifier
  - query the local document storage
  - inform the nest of the existence of other nests
  - obtain the list of neighbors known to the local nest
  - get and set the "pheromone" information associated to links to other nests
  - insert new documents in the storage
  - notify the nest about a request for which the ant has produced a response

The AntView Interface

```
public interface AntView {
    void move(NestId  id);
    Document[] request(Request request);
    void addNeighbor(NestId id);
    NestId[] getNeighbors();
    Pheromone getPheromone(NestId id);
    void setPheromone(NestId id, Pheromone pheromone);
    void addDocument(Document doc);
    void result(Request request, Response response);
}
```


Pheromone

- **Pheromone information:**
  - The Anthill framework does not specify any particular format for pheromone information
  - Ant developers are free to select a format that is appropriate
- **Examples:**
  - Gnutella-like ants:
    - Information about recent queries
  - Freenet-like ants:
    - "Pheromone" information computed on the basis of document keys

The Evaluation Framework

- **Using the evaluation framework, researchers may**
  - simulate the behavior of their ant algorithms
  - assess them
- **Each simulation is called a *scenario*, composed of:**
  - a collection of interconnected nests
  - a scheduling of requests to be performed
- **Scenarios:**
  - proceed by executing the requests and moving ants across the simulated network
  - are generated randomly following some distribution of probability

The Evaluation Framework

- **Several parameters may be monitored**
  - Number of requests issued/satisfied
  - Average number of hops per request
  - Number of ant hops performed
  - Number of documents copied
  - ….

The Evolution Framework

- **We further extend the analogy between P2P systems and the natural world using *evolution-based* paradigms**
  - genetic algorithms
  - genetic programming
- **Ants behavior is governed by their algorithms**
  - Algorithms may be parameterized in various way
    - Exploration probability
    - Document copy probability
  - Genetic algorithms may simplify the task of tuning the set of parameters of an ant algorithm for a particular task

The Evolution Framework

- **Chromosome**

  the set of parameters of a particular ant algorithm
- **The evolution framework**
  - maintain a population of chromosomes
  - at each generation, each chromosome is evaluated
    (using it as a set of parameters for the ants used in a scenario)
  - fittest chromosomes are selected, and a new generation is obtained from them through crossover/mutation
  - the process continues until
    - a fixed number of generations is executed
    - some fitness objective is met

Evolution Techniques at Run-Time

- **Can genetic techniques be applied at run-time?**
- **Example - In order to satisfy a request:**
  - several different ants may be launched
    - Example: ants with different chromosomes and/or algorithms
  - ants may be rated using a local fitness criterion
    - Example: time needed to satisfy a request
  - nests could steal the algorithms and the chromosomes of visiting ants, and use them to generate new ants
- **In this way, we obtain a P2P network that genetically selects its algorithm**

Anthill: Status and Future Work

- **The Anthill project started recently**
  - We have defined the framework API
  - We have a prototype implementation of the evaluation framework
  - We have the first ant algorithms
  - We are writing a prototype implementation of the nest infrastructure
- **We are looking for help!**
  - We have student thesis proposal
  - Anthill is (will be) an open-source project

Anthill "Wish list"

- **Development of new and more efficient ant algorithms**
  - Example: "crossover" between Gnutella and Freenet
- **Implementation of the complete nest infrastructure**
  - Downloading of code of new ants
  - Security
    - execution sandboxing
    - code signing
    - use of cryptography techniques for communication
- **Extension of the evaluation framework**
  - Scenario generators
  - Integration with existing simulation application

Anthill "Wish list"

- **Implementation of run-time evolution**
  - Adding the possibility of managing different ants algorithms in the same nests
- **Existing P2P network analysis**
  - Interfacing with existing Gnutella networks for studying their behavior (ex. Jtella)
  - Interfacing with the Freenet network
- **Implementation of existing P2P approaches in Anthill**
  - Gnutella-like and Freenet-like ants
  - Analysis of their behavior
    - Example: is document-movement important in Freenet?

For More Information

- **The Anthill web page:**

  http://phd.cs.unibo.it/anthill
- **You may find:**
  - These presentations (P2P and Anthill)
  - A position paper about Anthill
  - The Javadoc-generated API of Anthill
  - The current source code