Self-Organizing P2P Systems Inspired by Ant Colonies

by Carlo Mastroianni

We show how bio-inspired algorithms can be used to enhance the flexibility and efficiency of peer-to-peer computer networks and present Self-Chord, a peer-to-peer system in which resources are distributed, in an adaptive and semantic-aware fashion, by means of statistical algorithms inspired by the behavior of ant colonies. This strategy enables the efficient execution of complex queries, improves the load balance among peers, and strengthens the system's ability to self-organize and rapidly adapt to environmental changes.

Modern peer-to-peer (P2P) systems are based on a structured overlay, meaning that peers are organized in accordance with a predefined logical structure – ring, multi-dimensional grid, tree – and resources are assigned to peers with a precise strategy. This allows discovery requests to be driven, exploiting P2P interconnections, directly to the peers that store the desired resources. This type of organization is adopted by all the most popular P2P platforms (Kademlia, BitTorrent, Freenet), and recently also by Cloud companies for their distributed storage systems (for example, the Dynamo system adopted by Amazon).

Structured P2P systems are generally based on Distributed Hash Tables (DHT), whose purpose is to assign keys to resources, and indexes to peers, by means of hash functions. Each peer is responsible for a portion of the key space, and is required to manage the keys belonging to this portion. However, the use of hash functions has an important drawback: since similar resources are mapped to completely different keys, they are generally assigned to peers that are located far from each other in the overlay. This hinders the efficient execution of queries for resources having similar characteristics, or “range” queries.

At the ICAR-CNR Institute, in collaboration with the Politecnico di Torino, we have devised a new technique to implement structured P2P systems without using DHT functions. Instead of being computed with a hash function, resource keys may be given a semantic meaning, for example the value of a resource attribute. In this way, the keys of similar resources can be assigned to the same or neighbour peers, thus enabling the efficient execution of range queries. The technique is based on statistical operations of mobile agents whose behaviour is inspired by ant colonies, and follows the swarm intelligence paradigm. Ants hop from peer to peer, pick/drop resource keys, and sort them on the underlying P2P structure. Sorting of keys ensures their prompt discovery.

Ants Swarming in a Ring: the Self-Chord System

This ant-inspired approach has been adopted in the Self-Chord P2P system. Self-Chord uses the ring-shaped overlay of Chord, the forefather of structured P2P systems, to establish P2P interconnections, but distributes resource keys among peers using the operations of ant-inspired mobile agents. The underlying principle of this mechanism is illustrated in Figure 1. In this example, keys can be assigned integer values in the range [0..99], and the key space is toroidal, so that key 0 is the successor of key 99. Each peer computes its centroid as the key value that minimizes the sum of the distances between itself and the keys stored in the local area. In the example, an ant arrives at the peer with centroid C=30, and picks key 55, because its value is dissimilar to the centroid. Actually, the pick operation is subject to a Bernoulli trial, whose success probability is inversely proportional to the distance between key and centroid. The agent carrying the picked key exploits the long distance links of the underlying Chord structure, hops to a region of the ring where peers are supposed to have centroid values close to the key, and then tries to drop the key in this new region. Note that these operations assume that keys and centroids are already sorted in the ring. The power of the ant algorithm is that, by making this assumption, the keys will actually be ordered, even when starting from a completely disordered network. In stable conditions, the sorting of keys guarantees their logarithmic discovery. The base principles for this behavior are the self-organizing

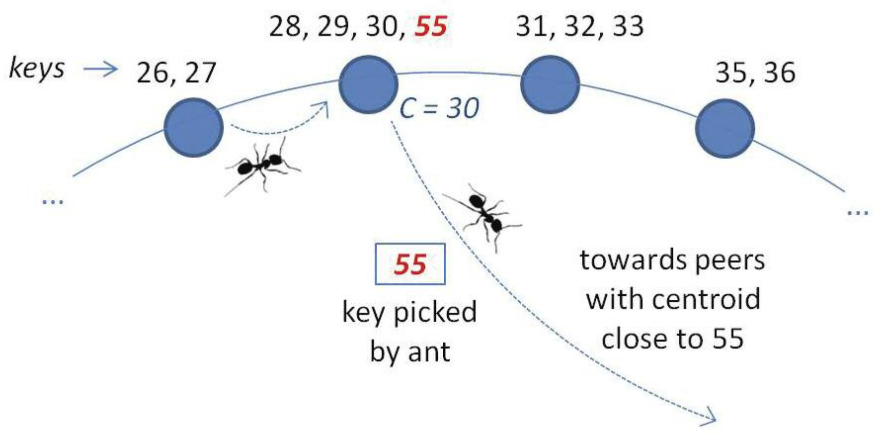

Figure 1: Example of a Self-Chord operation. A mobile agent arrives at a peer with centroid C=30, picks key 55, and then hops to a region of the ring where peers are supposed to have centroids close to 55.

nature and the positive feedback mechanism of ant algorithms.

In addition to serving range queries, this approach has further advantages:
- the range of values that can be assigned to keys is arbitrary
- the keys are fairly balanced among peers. In DHT-based networks, this feature is hindered by the non uniform distribution of resources and corresponding keys
- the network traffic is stable because it only depends on the frequency of ant movements. In DHT-based networks, the traffic is strongly affected by the frequency of peer connections and reconnections.

Ants Swarming in Multi-Dimensional and Hierarchical Structures

The ant approach can be adapted to any kind of overlay. In particular, we are analysing its application on multi-dimensional and hierarchical structures. A multi-dimensional overlay fits very well to situations in which resources are described by multiple and independent attributes: for example, computers may be described by their amount of memory, CPU speed, number of cores, etc. The ant-inspired approach allows resources to be described by multi-dimensional keys, corresponding to the attribute values, and keys to be sorted over the multi-dimensional structure. This enables the efficient execution of multi-attribute range queries, a sophisticated functionality that is very challenging in DHT-based systems.

The second case under study is when resources are described by hierarchical keys: this may be the case of distributed XML databases, in which documents are built in accordance with a hierarchical schema. For this kind of application, an appropriate overlay is that of Pastry. In Pastry, P2P interconnections are organized in a logical tree, and they allow agents to jump between peers whose indexes have a common prefix of any specified length. Mapping XML instances to this structure means that it will be possible to efficiently serve complex XML queries.

Links:
http://self-chord.icar.cnr.it
http://en.wikipedia.org/wiki/Distributed_hash_table
http://en.wikipedia.org/wiki/Swarm_intelligence

Please contact:
Carlo Mastroianni
ICAR-CNR, Italy
Tel: +39 0984 831725
E-mail: mastroianni@icar.cnr.it

Dialogue-Assisted Natural Language Understanding for a Better Web

by Mihály Héder

***By studying and improving the process of digital content creation, we can develop more intelligent applications. One promising step ahead is to acquire machine representation of content as early as possible - while the user is still typing and is available to answer questions*.**

Internet triggered a major revolution in everyday communication. The basic goal has not changed- to reach other people and to share our ideas- but everything else is different.

Computers, as mediators, have been incorporated into communication channels and users turn directly to them, and indirectly to other users when they need information. Clearly, the better the computers understand the content, the more useful and convenient they will become. However, to achieve some sort of understanding, they first need at least 1) an adequate machine representation of the content and 2) background knowledge. In this article, I will present a means of achieving the former.

In general, the life cycle of digital content can be divided into three stages: 1) the creation and formatting of content carried out by the user; 2) the attempt to create a machine representation of the content, carried out by indexers, the information extractor and a text mining software (typical components of a search engine); and finally, 3) the content can be consumed by other users (see Figure 1).

We have to face the fact that the second step of this process is very hard to carry out in a fully automated manner, as it would require unassisted natural language understanding to achieve the best results. Fortunately, when the content is yet to be written, we have an alternative option.

In the frame of a project in SZTAKI, we try to merge the first two phases of the content creation process detailed above. Our experimental software processes content on the fly while the user is still typing, and tries to ask relevant questions from him/her using everyday language. The aim of these questions is to clarify what the machine representation of the text should be (see Figure 2). While there are a multitude of semantic annotator tools, we think that the dialogue-assisted input method makes our solution a rather unique one. Also, this same property is the key to enabling lay users to create semantic annotations.

Our system consists of a rich text editor built around TinyMCE which is capable of visually annotating the content, and presenting questions/suggestions to the user. On the server side, we have a UIMA-compatible text processing system which relies on various UIMA Analysis Engines. One of these is the Magyarlanc, a tokenizer, lemmatizer and part of speech (POS) analyser for Hungarian. We also have a lan-