CatNet

ITS-2001-34030

Evaluation of the Catallaxy paradigm for decentralized operation of dynamic application networks

Delivery 2: Catallaxy Simulation Study

Covering period 1.9.2002-30.1.2003

Report Version: 0.1

Report Preparation Date: 30.01.03

Classification: public

Contract Start Date: 01.03.2003 Duration: 12 months

Project Co-ordinator: Universitat Politècnica de Catalunya

Partners: Albert-Ludwigs-Universität Freiburg

Project funded by the European Community under the "Information Society Technologies" Programme (1998-2002)

Table of contents:

DELIVERABLE SUMMARY SHEET

Project Number: ITS-2001-34030

Project Acronym: CatNet

Title: Evaluation of the Catallaxy paradigm for decentralized operation of dynamic application networks

Deliverable N°: 2 – Catallaxy Simulation Studies

Due date: 02/2003

Delivery Date: 02/2003

Short Description:

This study describes the configuration of the simulator and the design of the main experiments carried out. We describe how we mapped real world application networks into scenarios, which we have run on the simulator. We indicate the way the experiments are configured in the simulator and describe which parameters we measure. Also, we give some illustrations on the management of the experiments and the trace collection mechanism we used in the simulations. In the analysis of the simulation traces, we have observed that additional dimensions of the system can be explored which might provide further insight on the performance of both control mechanism.

Partners owning: Albert-Ludwigs-Universität Freiburg

Partners contributed: Universitat Politècnica de Catalunya

Made available to: Public

1. Introduction

The goal of this project is to compare the performance of a decentralized economic approach based on the Catallaxy paradigm to manage distributed applications with a centralized baseline system. To do this, both approaches are evaluated experimentally by means of simulations. The experiments consist of the simulation of an application layer network under different scenarios, while being operated under the Catallaxy paradigm and the baseline system. The analysis of the simulation results is used to compare how the application layer network behaves under both paradigms.

In this report we describe in section 2 the different scenarios of application networks, which we have evaluated experimentally. In section 3 we explain how these scenarios are mapped into experiments carried out with the simulator. We describe the configuration of the simulator regarding the input configuration and topology used. Then, we explain the parameters we measured and how these parameters can contribute to the comparison of both systems. In section 4 we outline the design and the management of the experiments. In section 5 we discuss some of our observations and lessons learned. We conclude in section 6.

2. Scenarios

Real world distributed applications like multimedia content distribution networks (for instance Akamai [1]), Grid implementations, and Peer-to-Peer systems (for instance Gnutella) can be characterized in a simplified form by a number of a few common features, which inspired the design of the application layer network implemented in the simulator. Though different in many particular mechanisms, these real world applications can be mapped to the two-dimensional design space given by 1) the node dynamics; and 2) the node density of the application layer network.

Node dynamics measures the degree of availability of service-providing nodes in the network. Low dynamics mean an unchanging and constant availability; high dynamics are attributed to a network where nodes start up and shut down with great frequency.

Node density measures the relation of resource nodes to the total number of network nodes. The highest density occurs when every network node provides the described service to others; the lowest density is reached if only one resource node in the whole network exists.

Describing application layer networks as points in a two-dimensional space formed by node dynamics and node density, we can identify content distribution applications such as Akamai as networks which have low node dynamics (quasi-static) and low node density. Low node dynamics due to the fact that resource nodes are highly available, being permanently connected to the network. Low node density due to the fact that resource nodes are few in terms of the total number of nodes, which form the application layer network.

On the opposite side of this explanation space, we can locate Peer-to-Peer (P2P) networks, which we can describe as networks with high node dynamics and high node

density. In this case, high node dynamics is given by the high level of connection and disconnection found in P2P networks. High node density is due to the fact that each peer carries out the function of a client, service provider, and resource, such that the number of resources is high in terms of the total number of nodes.

Considering content distribution networks as examples for low node density networks, we note that low node density comes along with high capacity of the resource nodes, where each resource is able to serve a high number of clients. On the other hand, considering P2P networks as example for high node density networks, the resources are many, but each of them has a small capability, being able to serve well only to a limited number of clients.

In Figure 1 we illustrate our approach on how we map real world application networks in a two-dimensional space. This classification allows, mainly by means of different setup parameters of the simulation, to simulate different application layer network scenarios.

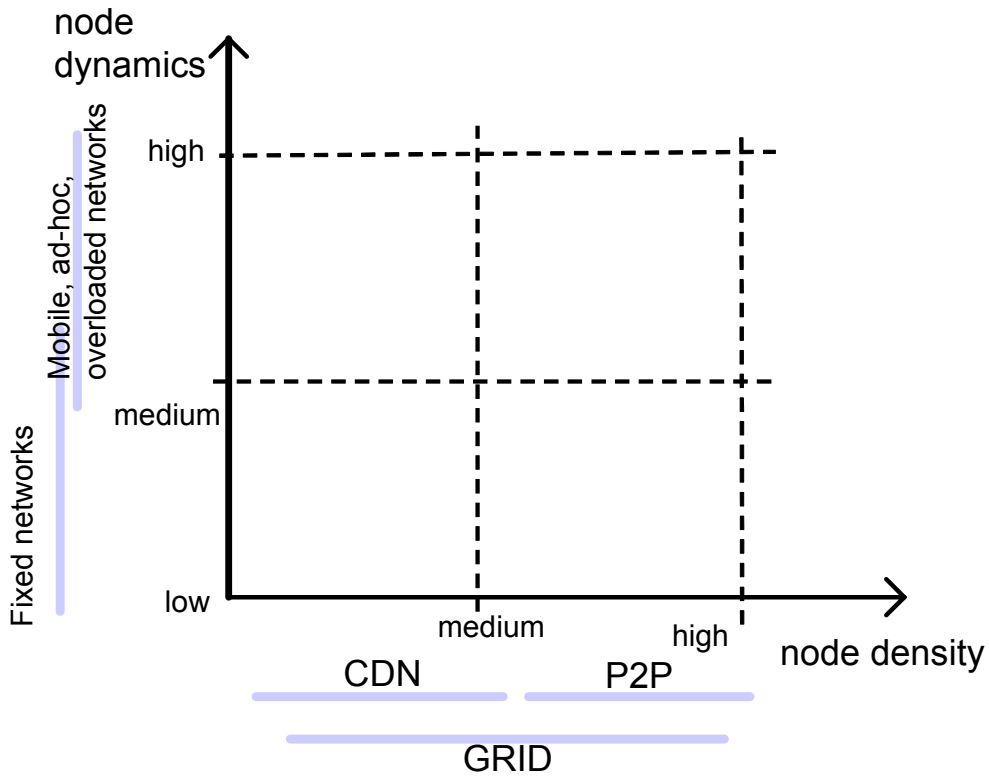

Figure 1. Mapping of real application layer networks into a two-dimensional design space.

3. Configuration of the simulator

3.1 Network topology

The application layer network is build on top of a physical network topology. The physical network topology is specified in the input of the simulator. The topology could be random, such as the topologies generated by the BRITE [2] network topology generator, or having a determined structure specified by the user.

In Figure 2 we show one of the physical topologies, which we used in the experiments. This topology uses a central ring of nodes. On each central node, another ring of nodes is attached. Each of the attached nodes has a certain number of leaves.

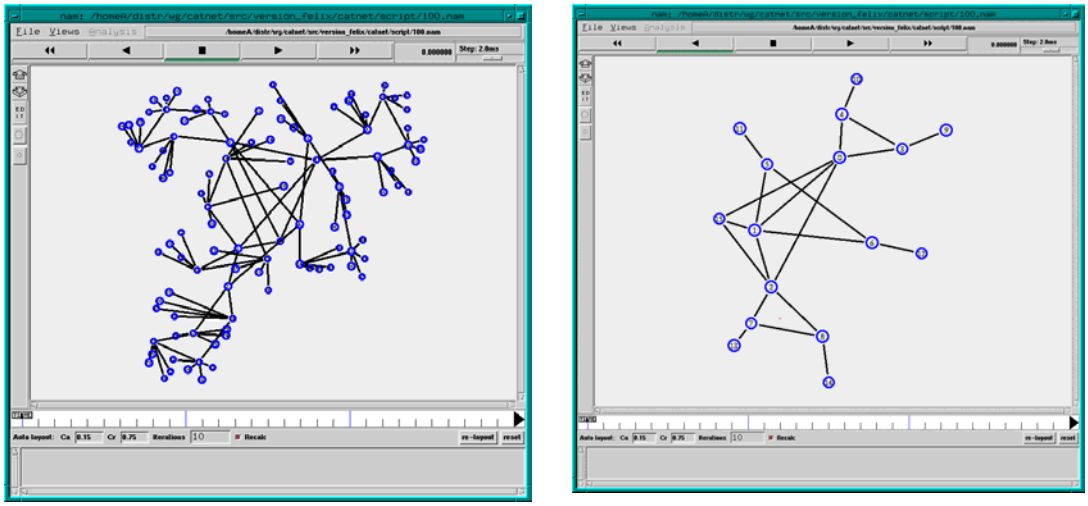

Figure 2. Examples of network topologies: left: approx. 100 nodes; right: 17 nodes.

On top of the physical nodes, a number of different software agents are created, which form the application layer network. The software agents are Clients, ServiceCopies, and Resources [3]. Each node can host Clients, Resources, and/or ServiceCopies. A node can host several agents or none at all. In the latter case, the node just acts as a router.

The application layer network formed by these agents is varied in the experiments. In order to simulate the node density of the application layer network, we vary the number of Resource agents and ServiceCopies available to Clients. In order to simulate the dynamics of the application layer network, we connect and disconnect during the simulation the available ServiceCopies.

For each agent, particular data such as the capacity of the Resources can be specified in the initialization of the simulator. Recall that the capacity of the resources is high in the low node density scenario, and low in the high node density scenario, due to the

correspondence with content distribution networks and peer-to-peer networks, respectively, such as discussed in section 2.

3.2 Input

The input of the simulator is on one hand the service demand trace, on the other hand all the data needed to set-up the network including the configuration of the physical network in terms of node topology and the application layer network in terms of the agent layout, the parameters concerning the network dynamics, and the initial prices.

The service demand trace of the simulator contains the service requests from the clients. The clients specify the required service. The simulator allows the clients to detail the requested service through some additional parameters. The service demand trace can be generated by the simulator, which includes a tool to generate random demand traces. For controlled experiments, the demand trace can be generated manually.

Variations of the service demand trace consist in the number and specific parameters of the requests, the number of different clients involved and the time when these clients send their request into the network.

The initial prices of Clients, ServiceCopies, and Resource agents are specified in the initialization of the simulator. Clients also dispose of an initial budget.

The type of control mechanism is another parameter specified in the setup of the simulator. The main control mechanisms implemented in the simulator are the Catallactic and the baseline approach. The modular design of the simulator, however, also allows testing variations of them to investigate the effect of different parameters in each control approach.

3.3 Scenario simulation

Our approach is to map different application layer networks from real world into a two-dimensional design space consisting of the node dynamics and node density.

The range of the node dynamics goes from quasi-static (content delivery networks) to very dynamic (peer-to-peer networks). The level of dynamics is set in the configuration of the simulator.

The range of the node density goes from low node density (content delivery networks) to high node density (peer-to-peer networks). The node density is set in the configuration of the simulator.

The parameter, which governs the dynamics of the network, affects the connection and disconnection of ServiceCopies. The node density is given by the number of resource agents available in the network.

As approximation, the following correspondence of node density to real world scenarios can be established:

- Low density (which also means high resource concentration) may correspond to web service scenarios with one or a few web servers.
- Medium density may correspond to a CDN or a Grid.
- High density (which also means spread resources) may correspond to a P2P network, or an extreme case of Grid.

3.4 Parameters measured

The parameters measured in the simulations should allow assessing how the Catallactic and the baseline system behave in the different scenarios, mainly in terms of resource allocation efficiency, response time, and communication cost, which may help to compute more elaborate parameters.

During the simulation, events are recorded in simulation traces for later analysis. In the general format, the events contain a time stamp and the value of the measured parameter.

In the following, we indicate some of the parameters we measure in the simulations.

Response Time (REST):
This parameter is the time observed by the client to get a service.

Resource allocation efficiency (RAE):
This parameter indicates the ratio of service demands, for which the network provides a service to all sent service demands.

Communication cost (CC):
This parameter reflects the number of hops used by the control messages.

Price:
This parameter shows the evolution of prices in the network during the simulation.

Client-Resource assignment:
This parameter contains the number of hops between a Client and Resource once a successful service provision is achieved.

3.5 Practical set-up of the simulator

In order to allow fast experimentation, the set-up of the simulator is divided into a number of Tcl scripts, such that an experiment can be assembled in a modular way.

The configuration of the simulator is split into the set-up of the physical network topology, the set-up of the application layer network, i.e. the specification of Clients, ServiceCopies, and Resource agents, and the set-up of the service demand. When executed together these scripts initialize and run the simulation.

4. Experiments

The experiments are executed simulating different scenarios of application layer networks, both using the Catallactic and baseline control mechanism. We evaluate the influence of the following parameters in the application layer network performance:

- Network dynamics
- Node density

This leads to the following series of main experiments, which we can view in terms of node dynamics variation or in terms of node density variation.

Varying node dynamics
1.1 In a "low density" system, the "dynamics" parameter is changed. 3 cases: null, medium, high
1.2, 1.3 (as 1.1) with other values of "density": medium, high

Varying node density
2.1 In a "quasi static" system, the "node density" parameter is changed. 3 cases: low, medium, high
2.2, 2.3 (as 2.1) with other values of "dynamics": medium, high

In total, each of the two control mechanism is simulated with 9 scenarios, as illustrated in Figure 3, which leads to 18 basic experiments.

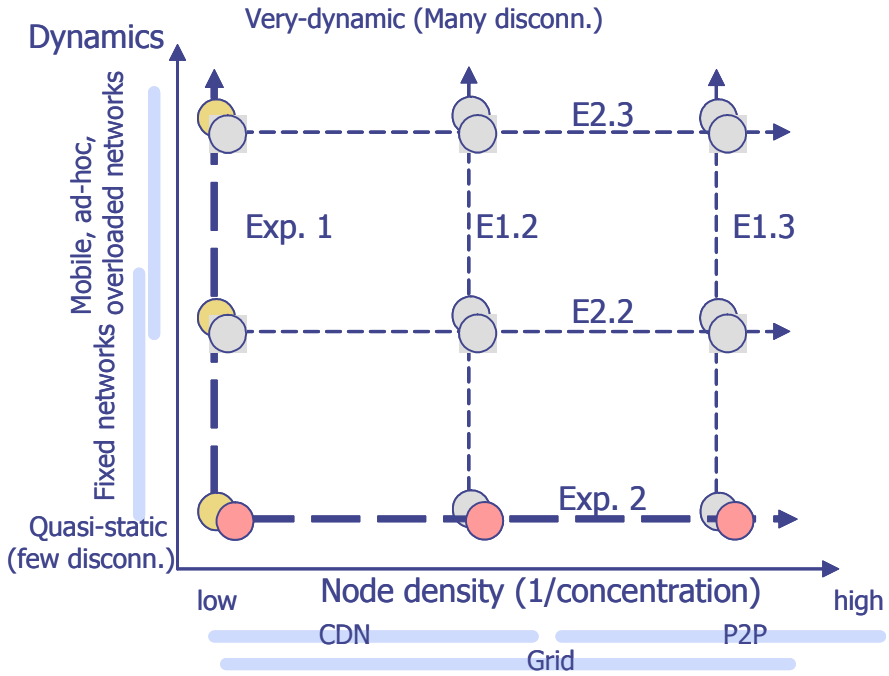

Figure 3: Illustration of the main experiments in a two-dimensional design space.

In order to manage the number of experiments, we have used an experiment template. For each experiment, the fields in the following template must be completed:

- Description:
- Input:
- Topology:
- Parameters:
- Results:
- Graphics:
- Observations:

For illustration, we show in Table 1 the description of an experiment.

Experiment
Description: In a "low density" (=high concentration) system, the dynamics parameter is changed. 3 cases: null, medium, high Therefore, it is a simple setting, a few nodes concentrating all resources. The effect of dynamics will be demonstrated.
Input: 500 service requests at different time steps. Requests are for the same service Id. Uniform initial price distribution on Clients, ServiceCopies, Resources.
Topology: 106 nodes. 75 Clients; 5 ServiceCopies, 5 Resources with 60 resource units each.
Parameters: record all
Results: Low node density (few resource nodes with large capacity) are favourable to the baseline case. With increasing dynamics the Catallactic system gets an advantage.
Graphics: automatic
Observations:

Table 1: Example for an experiment specification.

4.2 Simulation traces

Two techniques are applied to collect simulation data. First, we obtain simulation traces by means of log files, which contain the traced events in a temporal order. The second way to collect simulation data is by writing the simulation data into a data base.

The log files allow a quick analysis of the simulation data. The simulator can be specified to provide two main types of log files: The first type of log file records data on the application level, such as response time, node assignment and so on. The second log files record data on the network level, such as the number of packets, type of packets etc. The simulator computes parameters such as the resource allocation efficiency directly after the simulation and the main graphics are generated automatically. In Figure 4 we show for illustration purpose a fragment of a trace file.

…

2.395781430992124	n99/client/	89	96	10396	312	28.928217358858284	4
2.4258656546173087	n57/client/	126	58	10358	179	39.020000195503236	3
2.457956324645995	n82/client/	110	79	10379	268	39.42999980449677	4
2.6503574309921225	n98/client/	75	92	10392	667	39.312000370025636	5
2.7510183292999244	n36/client/	135	30	10330	377	39.312000370025636	5
2.875391276260374	n77/client/	147	79	10379	423	30.277999845027924	4
2.887439276260374	n92/client/	158	90	10390	312	34.858528202271344	3
3.0217752762603736	n91/client/	170	92	10392	308	37.001459927687534	3

…

Figure 4: Fragment of a simulation log file of the catallactic system. (column 1: time stamp, c4: response time, c6: price, c7: hops between client and resource.

In addition, some data is also saved on an external sql database. The use of the data base for trace collection eases the management and the later comparison of a large number of experiments.

5. Analysis of traces and discussion

The obtained traces are used to compare the performance of the Catallactic and the baseline system in the different scenarios. The comparison itself will be reported separately in D3 [4].

Although the main simulation parameters which we vary are the node dynamics and node density, we observed that actually the design space which could be considered for both systems is much larger. Other parameters we have considered to evaluate are, for instance, the effect of scale on the coordination mechanisms, the influence of particular characteristics of the demand trace, design parameters of the baseline system to handle highly dynamic environments, and parameters of the strategy used in the catallactic coordination to determine prices.

If we consider a n-dimensional design space, the experimental exploration of this space can be done by means of a heuristic search, in order to investigate additional

effects of different parameters and particular configurations based on the experimental results.

6. Conclusions

In this study, we have described how we mapped real world application networks into scenarios, which we have run on the simulator. We have indicated the way the experiments are configured in the simulator and described which parameters we measure. Also, we have given some illustrations on the management of the experiments and the trace collection mechanisms we used in the simulations. Although initially we explored a mainly two-dimensional design space of the system, we have also considered additional variations, which might provide further insight on the performance of both systems.

7. References

[1] Akamai: URL=http://www.akamai.com

[2] Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: Universal Topology Generation from a User's Perspective" Technical Report BUCS-TR2001 -003, Boston University, 2001.

[3] CatNet, ITS-2001-34030, Delivery 1: Simulator, 9/2002.

[4] CatNet, ITS-2001-34030, Delivery 3: Catallaxy Evaluation Report