Aggregation Dynamics in Service Overlay Networks

Pietro Michiardi
Institut Eurecom, France
Email: michiardi@eurecom.fr

Paul Marrow, Richard Tateson and Fabrice Saffre
BT Group CTO, Pervasive ICT Research Center, UK
Email: First.Last@bt.com

***Abstract*— In this work we analyze the characteristics of service overlay networks generated by uncoordinated service providers that deploy different service replicas on overlay nodes across the Internet. Our approach differs from previous works, that generally rely on application-level routing, in that we allow nodes to autonomously re-wire the service overlay to make it capable of absorbing a heterogeneously distributed workload that would otherwise result in some nodes with a specific service being overloaded and others remaining idle. We provide a game theoretic model of the overlay creation process and propose several optimization methods to achieve Nash equilibrium topologies. Equilibrium overlays are characterized by interconnected clusters of nodes that instantiate the same service replicas. Hindered by the computational complexity of finding stable wirings, we propose a simple distributed heuristic that allows the study of overlay networks with a realistic size and with several service instances. We show the ability of our re-wiring strategy to promote the emergence of a clustered global topology whilst running locally. We also argue that the lack of incentives for nodes to participate in the overlay creation might lead to several types of misbehavior, of which some representative cases are analyzed. Finally, both scalability and diversity (in terms of service instances) issues that might affect our distributed heuristic are evaluated in detail.**

I. Introduction

In recent years, the Internet has witnessed an evolution towards a commercial infrastructure of service delivery, as opposed to its original intent of simply providing host connectivity. Different forms of overlay networks [1], [2], [4], [10], [14] have been developed to offer attractive service provisioning solutions, which are difficult to be implemented and deployed at the IP-layer. These networks not only provide application-level data routing but also value-added services, such as content distribution services, media compression/transcoding, language translation, encryption or decryption services, etc... In our work we address the realistic scenario where independent and uncoordinated economic entities such as service providers deploy and manage their service instances at multiple locations on the Internet.

We term the logical connections among service provider nodes a *service overlay network*. Overlay nodes offering one or more services receive and process user requests that are fulfilled until a finite service capacity that characterizes each node is reached. In this context, issues of scalability and load-balancing across service replicas become an important domain of research, that has been recently addressed in the literature for example in [16]. Although the literature is rich in examples [3], [5], [6], [15] that achieve load balancing in the traditional client-server model, in [16] the problem of how load can be shared across service replicas in an overlay network is treated as an overlay routing problem. On one hand, previous works do not consider the costs of relaying user traffic (requests and responses) that might affect the latency of the service overlay. On the other hand, all current efforts share a common characteristic that dictate an immutable service overlay network wherein links are created based on the existence of a service level agreement between overlay nodes. Instead, in this work we propose a different approach wherein each node participates to the construction (and maintenance) of the service overlay so that an unevenly distributed workload can be absorbed by the service overlay without relaying user traffic in the overlay.

In particular, the focus of this paper is to study the aggregation dynamics that arise in the context of service overlay networks when nodes are involved in the process of constructing the overlay in a distributed manner. We show that the problem of service overlay construction can be casted as a network creation game, first defined in [11]. We present an *overlay creation game* that accommodates the service load experienced by the nodes and we study equilibrium overlay topologies that are built through their uncoordinated interaction. In our work, we explore several methods for obtaining stable solutions ranging from the traditional iterated *best response* method (for which, inspired by the work in [12], we also give an integer linear programming formulation) to a constrained search method.

Our analysis on the service overlay creation game is hindered by the complexity of computing the best response of a node, which is proven to be NP-hard [11]. Hence, we present a simple heuristic introduced in [17] based on a randomized local algorithm that allows studying the *emergence* of service overlays for systems with a realistic size.

In this work we analyze the ability of a selected set of rules (based on our game theoretic formulation and on our local heuristic) to foster self-organization of what is originally a random graph into a structured network. Scalability issues with respect to the key parameters of system size and diversity (in terms of deployed services) are extensively discussed.

We demonstrate that both an overlay rewiring process based on selfish decisions and on purely local decisions and interactions can result in an efficiently organized structure prone to achieve load-balancing without central planning. We conclude by discussing the implications of some nodes acting maliciously by not following our prescribed local algorithm.

The contributions of this paper can be summarized as

follows:

- we suggest a novel approach that lets nodes self-organize in a service overlay network capable of absorbing a heterogeneous workload generated by clients using the overlay;
- we propose a game theoretic model of service overlay construction wherein the uncoordinated strategic actions of nodes result in a stable wiring of an initially random overlay;
- we evaluate a decentralized heuristic, designed to overcome the computational requirements of the game theoretic approach, that is capable of promoting the emergence of a global overlay topology characterized by clusters of nodes sharing similar traits;
- we study the impact of the lack of incentives for nodes to participate in the re-wiring process of the overlay, and suggest different misbehavior types.

The remainder of this paper is structured as follows: in Section II we detail the system model and architecture; in Section III we give a formal definition of the overlay creation game while in Section IV we present the randomized local search algorithm. In Section V we describe the numerical evaluation of our model and present our results. We illustrate related works in Section VI and conclude in Section VII.

II. Uncoordinated service overlay creation

Inspired by the model proposed in [16], in Figure 1 we show an example of a service overlay network: individual service providers deploy their services at overlay nodes which establish logical links to form an overlay network. In the figure we show what we term *exit nodes*, that is those nodes that receive client requests for a service or a set of services.

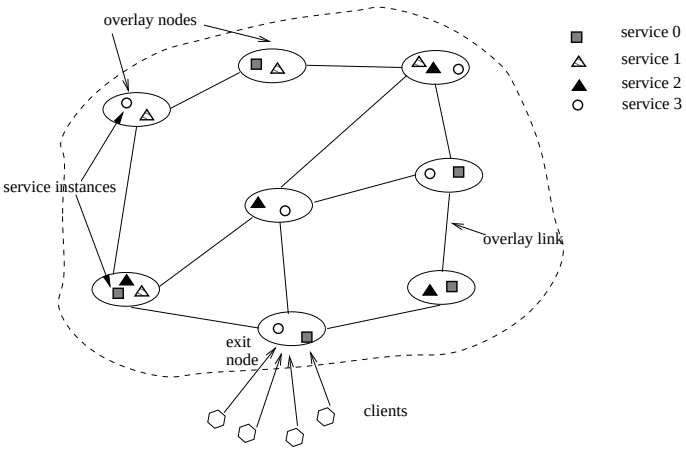


Fig. 1. System architecture under consideration with an example service level path.

While there are several challenges, for example, in the context of *service composition* [16] to enable rapid development of new applications, in this paper we focus on the construction and maintenance of the overlay network. We assume that overlay creation is achieved by the uncoordinated interaction of nodes that wish to minimize their cost, expressed in terms link, communication and load, incurred in the creation and maintenance of overlay links. Given a random unstructured overlay network, the problem we address in this paper is to find *stable* overlay configurations that take into account both load and connectivity constraints. Note that in this work we do not discuss mechanisms to *dispatch* user requests so as to achieve a fair load balancing: the overlay creation process should be seen as an optimization phase prior to the execution of such a mechanism.

A. Assumptions

To improve the tractability of the problem, we limit the scope of our study to physical topologies in which every node can communicate with every other node, that is the *communication graph* (i.e., the underlay physical network) is connected.

With the aim of simplifying the problem formulation we assume that only one service instance is deployed on each overlay node. Hence, in the following a node is equivalent to a service. We define a *type* t_i associated with a node i as the encoded description of the service that it can provide.

Every node i is characterized by a *nominal capacity* λ_n^i which defines the end-user request load a node can handle; λ_c^i identifies the current load experienced by node i. In the following we assume that:

$$\Lambda_n = \sum_{i=1}^{n} \lambda_n^i \, , \Lambda_c = \sum_{i=1}^{n} \lambda_c^i \, , \Lambda_c \leq \Lambda_n \qquad (1)$$

Equation 1 simply states that the system is able to support the requested load, that is, the system does not saturate. In this work we assume no additional end-user requests to be made at run-time.

III. Overlay creation game

In this Section we model overlay creation as a non-cooperative game with n nodes (i.e, players) whose strategies are to select which nodes to connect to. Our model extends the one presented by Fabrikant *et. al.* in [11]. Formally, there is a finite set of players $N = \{1, ..., n\}$, a finite set of player types $T = \{t_1, t_2, ..., t_n\}$ and the strategy space of player $i \in N$ is the list of other players to connect to, i.e., the set $S_i = \{(s_{ij})_{j \neq i} | s_{ij} \in \{0, 1\}\}$ where $|S_i| = 2^{(n-1)}$. Players simultaneously announce the list of other players with whom they wish to be connected. Their decisions generate a (directed) graph $G(s) = (N, s)$: the direction of a link from node i to node j indicates that node i will share its excess load with node j. However, for communication purposes, the link can be seen as bidirectional.

Note that this is a *single-stage* game with *simultaneous* announcements. Our game model requires complete information of the service overlay graph as well as node types and assumes players to be computationally unbounded. In the game, each player selects its strategy to minimize its cost.

The cost incurred by player i when all other players adopt strategy s_{-i} is additive in the cost l_j of a link to players

of different type, in the cost l_j of a link to players of the same type modulated by a threshold function based on load information, as well as in the sum of the costs of reaching all other players:

$$c_i(s) = \alpha \sum_{j \in NB_i} (1 - \delta_{t_i,t_j}) l_j + \alpha\, \sigma(\lambda_c^i, \lambda_n^i) \sum_{j \in NB_i} \delta_{t_i,t_j} l_j + \\ + \sum_{j=1}^{n} d_{G(s)}(i,j) \quad (2)$$

where NB_i is the set of overlay neighbors for which a direct link exists, $d_{G(s)}(i,j)$ is the shortest path distance from node i to node j in the graph $G(s)$, and δ_{t_i,t_j} is the Dirac-Delta function $\delta_{t_i,t_j} = 1$ if $t_i = t_j$ and 0 otherwise.

The connection cost α represents the relative importance of player i's direct links to others and is the only parameter in the model.

The function $\sigma(\lambda_c^i, \lambda_n^i)$ is defined as follows:

$$\sigma(\lambda_c^i, \lambda_n^i) = 1 - \frac{1}{1 + e^{k(\lambda_c^i - \lambda_n^i)}} \quad (3)$$

$\sigma(\lambda_c^i, \lambda_n^i)$ is used to modulate the linking cost to a node running a replica of the service instantiated in i. When $\lambda_c^i \leq \lambda_n^i$, linking costs can be easily reduced to the model defined by Fabrikant in [11]. However, when $\lambda_c^i \geq \lambda_n^i$ link costs to other nodes with the same instance of the overloaded service in i significantly decrease. Hence, as long as node i can support its current load, it will seek at minimizing communication and linking costs; when the requests load λ_c^i approaches and exceeds the threshold λ_n^i, node i will prioritize the creation of links to overlay nodes that can support requests for the overloaded service.

The total cost of the graph G is then defined as:

$$C(G) = \sum_{i=1}^{n} c_i(s) \quad (4)$$

We now define what constitutes a solution of the game, that is, which overlay topologies result from the overlay creation game. When networks arise from the unilateral action of players, standard Nash equilibrium analysis can be informative about the structure of the networks that emerge. Let $s = s_N = (s_i, s_{N \setminus i})$ and let ζ designate the set of all undirected networks on N.

Definition 1: A network $G(s) \in \zeta$ is a Nash equilibrium network if there exists a strategy s that supports $G(s)$ where $c_i(s) \leq c_i(s'_i, s_{N \setminus i}) \forall i \in N$ and $s'_i \in S_i$.

A. Game optimization process

In this Section we describe three alternative methods that we used to find stable overlay configurations as described in Definition 1. The three methods under investigation share a common algorithmic structure shown in Algorithm 1[1]. The algorithm accepts as input the total number of nodes N that form the service overlay network, the number of edges E for each node i and the total number of types $\hat{T}$ in the overlay. We then generate a random initial overlay configuration and store it as an adjacency matrix. The algorithm outputs a selfishly optimized adjacency matrix where $Strategy^*$ indicates the best response for a node in each iteration. We use a stop criterion based on the definition of the Nash Equilibrium given in Section III.

Algorithm 1: Game Optimization Algorithm

Input: N, E, $\hat{T}$
Output: $UpdateAdj$
$Adj \leftarrow generateRandomOverlay(N, E)$;
$TypeVector \leftarrow generateRandomTypes(\hat{T})$;
$UpdateAdj \leftarrow Adj$;
$StopCriteria \leftarrow 0$;
$TotalCost \leftarrow 0$;
$iteration \leftarrow 0$;
while $StopCriteria = 0$ **do**
  $iteration \leftarrow iteration + 1$;
  for $i \leftarrow 1$ **to** N **do**
    $CurrentCost \leftarrow evaluateCost(i, UpdateAdj)$;
    $StrategySpace \leftarrow$
    $generateSSpace(i, UpdateAdj)$;
    $Strategy^* \leftarrow -1$;
    foreach $Strategy \in StrategySpace$ **do**
      $TmpMatrix \leftarrow UpdateAdj$;
      $TmpMatrix(i,:) \leftarrow Strategy$;
      $CostNew \leftarrow evaluateCost(i, TmpMatrix)$;
      if $CurrentCost - CostNew > 0$ **then**
        $CurrentCost \leftarrow CostNew$;
        $Strategy^* \leftarrow Strategy$;
      end
    end
    if $Strategy^* \neq -1$ **then**
      $UpdateAdj(i,:) \leftarrow Strategy^*$;
    end
    $TotalCost(iteration) \leftarrow$
    $TotalCost(iteration) + CurrentCost$;
  end
  if
  $TotalCost(iteration) \leftarrow TotalCost(iteration - 1)$
  then
    $StopCriteria \leftarrow 1$;
  end
end

In the following we detail three alternative approaches that differ in the way the strategy space available at each overlay node is generated. Our first method is based on an integer linear programming (ILP) formulation of the overlay creation game. This method provides exact solutions to the overlay creation game. In the second method we exhaustively explore all the possible strategies available to a player, which will then select the best response to all other players actions. Although

[1]For clarity of exposure nodes execute the algorithm sequentially. In our implementation nodes execute the algorithm in random order.

very simple to implement, the computational complexity of this approach is an obstacle the study of service overlay when the number of nodes is high. Lastly, we present a variation of the iterated best response method wherein we restrict the strategy available to a node at each round. This method allows to increase the number of nodes participating in the rewiring of the overlay at the cost of finding a selfishly optimized overlay which do not always meet the Nash condition given in Definition 1.

1) ILP formulation: In this Section we formulate of the overlay creation game based on the work presented in [12]. The following general method is used to obtain the best response of a node i when the degree of the connections established by that node (*i.e.,* the out-degree) is bounded by k_i. A wiring for a node i can be defined using $n-1$ binary unknowns Y_l, $1 \leq l \leq N$, $l \neq i$: $Y_l = 1$ iff i wires to l, and 0 otherwise. Define also the binary unknowns X_{lj} : $X_{lj} = 1$ iff i has l as a first-hop neighbor on a shortest path to j. Let $G(s)_{-i} = G(s) - s_i$ denote the *residual graph* obtained from $G(s)$ by taking away node i's outgoing links. A best response for node i under residual graph $G(s)_{-i}$ can be obtained by solving the following Integer Linear Program (ILP):

$$\texttt{minimize}:$$
$$c_i(G(S)_{-i}, X) = \sum_{j=1, j\neq i}^{n} p_{ij} \sum_{l=1, l\neq i}^{n} X_{lj} \cdot (\alpha + d_{G(s)_{-i}}(l, j))$$

$$\texttt{subject to}:$$
$$\sum_{l=1, l\neq i}^{n} X_{lj} = 1, \forall j \neq i; \sum_{l=1, l\neq i}^{n} Y_l = k_i; X_{lj} \leq Y_l, \forall l, j \neq i; \quad (5)$$

where $p_{ij} = 1$ iff $t_i \neq t_j$ and $p_{ij} = \sigma(\lambda_c^i, \lambda_n^i)$ iff $t_i = t_j$. Furthermore, $d_{G(s)_{-i}}(i, j)$ is the shortest path distance from node i to node j in the residual graph $G(s)_{-i}$.

Although the ILP formulation of the selfish optimization problem, as well as the limit on the out-degree of a node help in improving the algorithm execution time, we note that the overlay creation game is hindered by the number of times the Dijsktra algorithm needs to be executed for the evaluation of the cost associated to each feasible strategy. As a consequence, when the number of nodes that form the overlay and the number of service types instantiated on each node increase substantially, the optimization framework adopted in this Section cannot be used.

2) Exhaustive search, Iterated Best-response: Ideally, all of the strategy space available to a node should be examined to determine the action(s) yielding the lowest cost, as defined in Equation 2. The method presented in this Section enumerates for every node i, all the connection vectors that are within the feasible region of the optimization problem and associates a cost computed using Equation 2. Node i will select the connection vector (the strategy) that minimizes its cost. This simple approach limits the size of the network that can be studied, as the time complexity to find all possible strategies a node can have is exponential with the number of nodes (the problem is somehow similar to [11], where it is shown that computing the Nash equilibria for the network creation game is NP-hard).

3) Local search, Iterated Best-response: The following method builds on the Iterated Best-response approach but limits the strategy space of a node i to its (overlay) two-hop neighborhood. The function that associates a cost to each feasible strategy is the one defined by Equation 2. Node i will select the connection vector (the strategy) that minimizes its cost. It should be noted that this local search strategy may converge to overlay configurations that do not meet the Nash conditions. Each node periodically performs two operations: a *Link Drop* and a *Link Add* operations. The link drop procedure computes the cost for node i when each of its neighboring node is dropped. If the difference between the cost c_i of the current configuration and the new configuration without the dropped link is positive, node i updates its neighborhood by dropping the link to the node yielding the lowest cost. We describe the link drop procedure in Algorithm 2.

Algorithm 2: Link Drop procedure for node i

Input: i, $OverlayAdjMatrix$
Output: $UpdateAdj$
$CurrentMatrix \leftarrow OverlayAdjMatrix$;
$NodeToDrop \leftarrow -1$;
$CurrentCost \leftarrow evaluateCost(i, CurrentMatrix)$;
$MinCost \leftarrow CurrentCost$;
foreach $j \in CurrentMatrix(i, :)$ **do**
  $tmpMatrix \leftarrow CurrentMatrix$;
  $tmpMatrix(i, j) \leftarrow 0$;
  $CostNew \leftarrow evaluateCost(i, tmpMatrix)$;
  if $MinCost - CostNew > 0$ **then**
    $MinCost \leftarrow CostNew$; $NodeToDrop \leftarrow j$;
  end
end
if $NodeToDrop \neq -1$ **then**
  $UpdateAdj(i, NodeToDrop) \leftarrow 0$;
end

Using the link addition procedure node i randomly selects a 2-hop neighbor, that is the neighbor of a node to which she has a direct link, that is not the node dropped with the link drop procedure. If the cost c_i evaluated by adding a link to the randomly selected node decreases, then node i updates its current connections by adding a (direct) link to the randomly selected node. We describe the link addition procedure in Algorithm 3.

Note that, as opposed to the method described in Section III-A.2, the selfishly constructed overlay graph depends on the initial number of links E established by each node.

Algorithm 3: Link Add procedure for node i

Input: i, $OverlayAdjMatrix$
Output: $UpdateAdj$
$CurrentMatrix \leftarrow OverlayAdjMatrix$;
$CurrentCost \leftarrow evaluateCost(i, CurrentMatrix)$;
$j \leftarrow$
$rand(getTwoHopNeighbors(i, CurrentMatrix))$;
$tmpMatrix \leftarrow CurrentMatrix$;
$tmpMatrix(i, j) \leftarrow 1$;
$CostNew \leftarrow evaluateCost(i, tmpMatrix)$;
if $CurrentCost - CostNew > 0$ **then**
  $UpdateMatrix(i, j) \leftarrow 1$;
end

IV. Local Re-wiring of the Service Overlay

The optimization method exposed in Section III-A.3 appears to be very close to a local heuristic, given the structure of Algorithms 2 and 3. However, we cannot adopt this method for the implementation of a distributed algorithm to achieve overlay rewiring: indeed, the hidden characteristic of this method is that local decisions taken by the nodes depend on the cost function defined in Equation 2, which in turns depends on the *entire* overlay graph. In this section we design a completely distributed algorithm that mimics the behavior of the method discussed in Section III-A.3, but which uses local information only. Precisely, we focus on a a very simple algorithm that ignores the costs defined in Equation 2 and that is inspired by the procedures of adding and dropping one overlay link described in the previous Section.

Hence, the goal of this Section is to study the emergence of a given global topology arising from local decision-based rewiring of the service overlay. The literature is rich of local algorithms capable of promoting the emergence of networks characterized by some global properties, one of the most prominent examples being the "preferential attachment" rule proposed in [7] that produces scale-free topologies characterized by a power-law distribution of node degree. However, these algorithms often assume that global knowledge of the network topology (assumption that we also make in our game theoretical model) is available at each node.

In contrast, in this Section we assume that nodes do not have a global view of the system topology nor of the global service type distribution and we study the properties of overlay topologies obtained using a simple heuristic that stems from our game theoretic formulation of the problem.

In Section IV-A we detail our algorithm, called RLS (that we first introduced in [17]), based on *randomized local search* while in Section IV-B we outline possible attacks to the normal execution of the algorithm.

A. Randomized local search (RLS)

Given an initial service overlay configuration, nodes execute a distributed algorithm based on local information only, restricting the visibility of a node in a similar way to the one described in Section III-A.3. We introduce three node roles: the *initiator*, the *mediator* and the *target* role. Initiators and mediators interact so as to match the type associated to the initiator and the target nodes, at the expense of the mediator node that loses one connection. The effect of this rewiring strategy is that the node degree of the initial overlay configuration is preserved. The details of the randomized local search algorithm are given in Algorithm 4[2].

Algorithm 4: Randomized Local Search Algorithm

Input: N, E, T, $MaxIterations$
Output: $UpdateAdj$
$Adj \leftarrow generateRandomOverlay(N, E)$;
$Type\_Vector \leftarrow generateRandomTypes(T)$;
$UpdateAdj \leftarrow Adj$;
$StopCriteria \leftarrow 0$;
while $StopCriteria = 0$ **do**
  $iteration \leftarrow iteration + 1$;
  for $i \leftarrow 1$ **to** N **do**
    $mediator \leftarrow rand(getOneHopNeighbors(i))$;
    while $isempty(target) \&\& TypeVector(i) \neq TypeVector(target)$ **do**
      $target \leftarrow rand(getOneHopNeighbors(mediator))$;
    end
    % Initiator adds link to Target
    $UpdateAdj(i, target) \leftarrow 1$;
    $UpdateAdj(target, i) \leftarrow 1$;
    % V.1: Initiator removes link to Mediator
    $UpdateAdj(i, mediator) \leftarrow 0$;
    $UpdateAdj(mediator, i) \leftarrow 0$;
    % V.2: Mediator removes link to Target
    $UpdateAdj(target, mediator) \leftarrow 0$;
    $UpdateAdj(mediator, target) \leftarrow 0$;
  end
  if $iteration = MaxIterations$ **then**
    $StopCriteria \leftarrow 1$;
  end
end

In Algorithm 4 we sketch two variants of our method: in the first, the Initiator and Mediator nodes lose their connections when a matching node is found, in the second, the Mediator and the Target nodes lose their connections.

B. Rational attacks

Provided that there is no incentive for nodes to take up the role of Mediators, what is the impact of node misbehavior during the execution of the randomized local search algorithm? We try and address this question in the following.

Lack of incentives to follow the rules prescribed by the RLS algorithm are a direct consequence of the fact that a Mediator node might be reluctant to give up a connection (be

[2]For clarity of exposure nodes execute the algorithm sequentially. In our implementation nodes execute the algorithm in random order.

it with the Initiator or the Target) if this might be harmful for subsequent iterations of the algorithm in which the Mediator could switch to the role of Initiator. As an illustrative example consider the scenario in which two node types are present in the service overlay. If the Initiator and the Mediator share the same type, then there is no incentive for them to interact. If the Initiator and Mediator are of different type, for the simple fact of giving up a connection to the Initiator, the Mediator can preclude the possibility of gaining a new neighbor with the same type when its turn to be an Initiator arrives. The same apply if a connection with the Target node is lost.

We thus identified two simple mediator misbehaviors that we describe next:

- Misbehavior 1: the Mediator systematically ignores switch requests by Initiators;
- Misbehavior 2: the Mediator systematically fails in providing the Initiator with a Target sharing the same type.

We analyze the effects of these two node misbehavior when a fixed fraction of the overlay populations is misbehaving in Section V.

V. Numerical evaluation

In this section we present the experimental setup used to characterize service overlays created both by the selfish optimization algorithm presented in Algorithm 1 (together with Algorithms 2 and 3) and by the RLS algorithm outlined in Algorithm 4. Our goal is to compute the Nash equilibria topologies of the overlay creation game and the stable topologies of the randomized local search algorithm using the methods described in Section III-A.2 and in Section IV. The overlay network graph $G(s)$ is represented by an adjacency matrix.

A. Experimental set-up

The initial conditions of the optimization problem are given by a set of N nodes, a node type vector that identifies which services are instantiated on the N nodes and a randomly generated service overlay. We used several models to generate an initial random deployment of the service overlay, but in this Section we will focus on *k-regular graphs* where the number of randomly chosen outgoing links E is a simulation parameter. We define a maximum number ($\hat{T}$) of different services available in the overlay and uniformly distribute them on the nodes. For every simulation run we randomize on the initial service overlay: in the following all plots are the results of several (50) algorithm executions, while the graphical representation of an overlay depicts an instance of a stable topology.

B. Metrics

In this work we are interested in studying the graph properties of the equilibrium service overlay. In Section V-C and Section V-D and we evaluate the *node degree distribution* that characterize the final stable configuration of the service overlay, when relevant. Furthermore, we study the *overlay homogeneity*, defined as the fraction of links connecting two overlay nodes of the same type. The homogeneity metric indicates to some extent the clustering degree of the overlay: values close to one are preferred over values close to zero.

C. Results for the selfish optimization algorithm

In this Section we discuss on the service overlay topologies that have been obtained with the local best response algorithm discussed in Section III-A.3. For illustrative purposes, we focus on overlay networks with $\hat{T} = 2$ types in the system and we assign equal nominal service capacities to each node, that is $\lambda_n^i = \lambda_n \forall i \in \{N\}\backslash\{i,j\}$, except for two nodes of each of the two types in the system that have a significantly higher nominal capacity. We expect these two nodes to have a high number of incoming links, as this implies current request load to be shared from an overloaded node to a high capacity node. The request load λ_c^i assignment is $\lambda_c^i = \lambda_c = 2 \cdot \lambda_n \forall i \in N$. We specify the linking cost defined in Equation 2.

Figure 2(a) shows the initial service overlay topology where $N = 30$ nodes are randomly deployed with $E = 4$ links per node. Node types are uniformly distributed, and two random nodes of different type are assigned a high nominal capacity, $\lambda_c = 100$: in Figure 2 node 1 and 16 are high capacity nodes. The wiring cost has been set to $\alpha = 100$.

Figure 2(b) shows that high capacity nodes have a high number of incoming links, while all other nodes sharing the same type are highly connected[3]. This is a direct consequence of the cost function of Expression 2: when a node is overloaded, the cost to connect to other nodes of the same type goes to zero, while communication costs still have an impact on the selection of which node is best to connect to. Figure 2(b) clearly shows two highly connected clusters of nodes sharing the same type. This new overlay configuration is prone to be used as a basis for a load dispatching mechanism and link directions indicate that excess load will flow towards nodes with high nominal capacity. It should be noted that the selfishly constructed overlay graph allows messages (*e.g.* maintenance or discovery signaling) to be exchanged by any node at a minimum cost.

The results presented in this Section are in line with those obtained using the simplified method (RLS), that are we extensively reported in the following Section.

D. Results for the randomized local search

This Section summarizes the results for the global self-aggregation dynamics arising from the RLS algorithm described in Section IV. Scalability issues with respect to key parameters such as system size and diversity (in number of types) are extensively studied. Lastly we show the impact of node misbehavior on self-aggregation dynamics.

1) Impact of the initial random overlay structure: Figure 3 show, for a sufficient number of time-steps for the graph to be stable, respectively the aggregated service overlay graph (Figures (a) and (d)), the node degree distribution (Figures (b) and (e)) and the homogeneity (Figures (c) and (f)) for

[3]In the Figure, a link without direction is a bidirectional link.

(a) (b)

Fig. 2. Rewiring process for the local iterated best response algorithm: initial (left) and equilibrium (right) topologies.

an overlay comprised of $N = 500$ nodes and $T = 3$ types. The simulation parameter is the number of random edges E assigned to each node during the system bootstrap.

Figures 3(c, f) show that the configuration of the initial overlay has a negligible impact on the convergence speed to a homogeneous network (in the sense of the definition given in section V-B) and the final homogeneity attained.

2) Impact of the system size and system diversity: In this section we show the sensitivity of the randomized local algorithm to variations in the system size and diversity, when $N = 500$, $T = 3$ and $E = 6$.

Figure 4 a) shows the impact of the system size on the dynamics of aggregation in terms of system homogeneity.

We observe that Algorithm 4 may support service overlay comprised of a very high number of nodes without the convergence speed being affected by the system size, confirming that our simple re-wiring strategy is highly scalable.

In Figure 4 b) we show the impact of the number of types in the system, that is the number of different services instantiated in the service overlay, on the homogeneity of the system.

Figure 4 b) indicates that increasing the number of service types deployed on the overlay may lead to very slow convergence times: results can improve when the initial random overlay is highly connected, at the risk of obtaining an final overlay with disconnected domains.

3) Impact of node misbehavior: In this section we assume a service overlay formed by $N = 500$ nodes (when not specified otherwise) with an initial number of edges $E = 6$ per node. We study the convergence speed of Algorithm 4 varying both the system size and diversity when a fraction of the population is misbehaving following the two models discussed in Section IV-B. In out analysis we vary the (fixed) fraction of misbehaving nodes m in the set $m = \{0.2, 0.5, 0.8\}$, but we plot only results for the two extreme cases for clarity of presentation.

In Figure 5(a,b) we show the impact of an increasing fraction of misbehaving nodes of type 1 when the number of nodes is a fixed parameter and we increase the number of service types instantiated in the overlay.

Figure 6(a,b) shows the impact of misbehaving nodes of type 2 under the same simulation settings described above.

When mediator nodes systematically fail in responding to requests made by initiators wishing to connect to nodes of similar type, the apparent effect as shown in Figure 5(a,b) is a slow convergence speed to a clustered overlay. Only when a very high fraction of nodes misbehave the performance degrades significantly from the case when no illegitimate node are present in the overlay.

The effects of a mediator providing a wrong target to the initiator are more pronounced: even a small fraction of misbehaving nodes of type 2 can disrupt the correct functioning of Algorithm 4, as shown in Figure 6(a,b). Clusters of nodes sharing the same service instance, thus being able to absorb the workload of overloaded nodes, are hardly formed.

In both cases of Figures 5 and 6, the negative effects of a malicious activity are proportional to the number of service types instantiated in the overlay.

In Figures 7(a,b) and 8(a,b) we show the impact on the system size of an increasing fraction of misbehaving nodes of type 1 and type 2 when we vary the (fixed) fraction of misbehaving nodes m.

By observing Figures 7(a,b) and 8(a,b), we conclude that when the system size scales up to a high number of participants, the effects of both types of misbehavior are less pronounced, although the second type of misbehavior is more disruptive.

VI. Related work

Traditional load balancing has a long history in distributed systems literature. A small sample of previous results, that focus on coordinated and often centralized solutions, includes the following: [15] investigates the online assignment of unit length jobs under the L_∞ norm; [3] considers off-line assignments of unit length jobs; [5] consider greedy assignments of weighted jobs under the L_p norm, where the client-server graph is complete bipartite while [6] considers arbitrary client-server graphs and uses the L_2 norm.

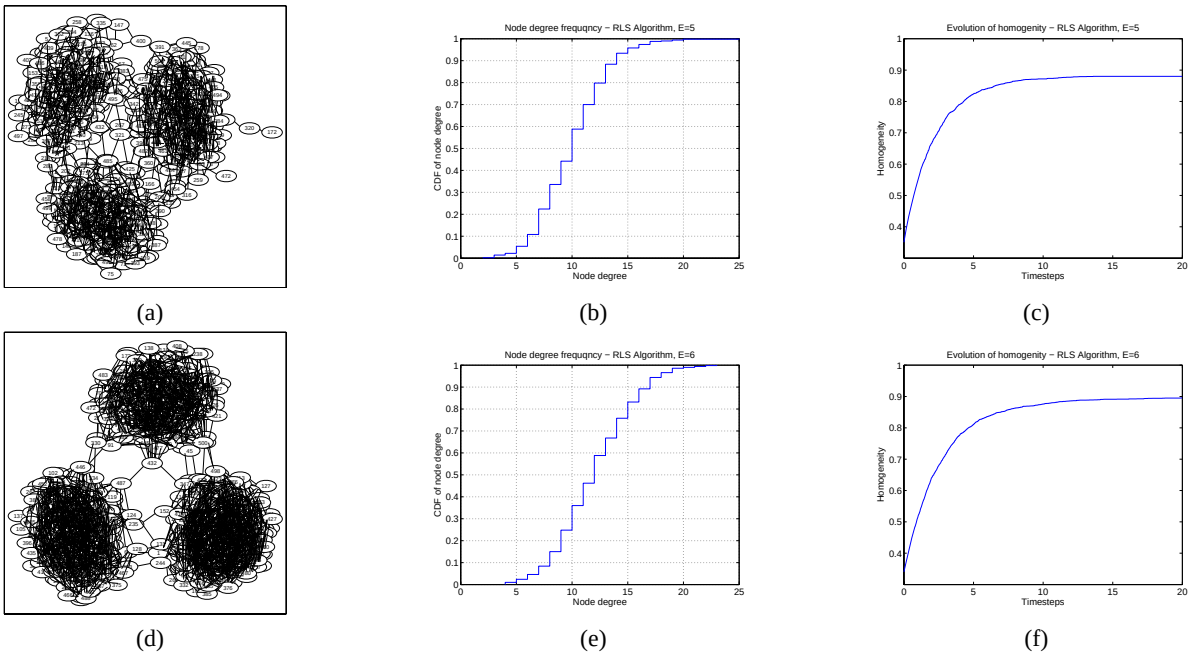


Fig. 3. Stable service overlays (left), Node degree distribution (middle) and Homogeneity plots for overlays with $E = 5$ (first row) and $E = 6$ (second row).

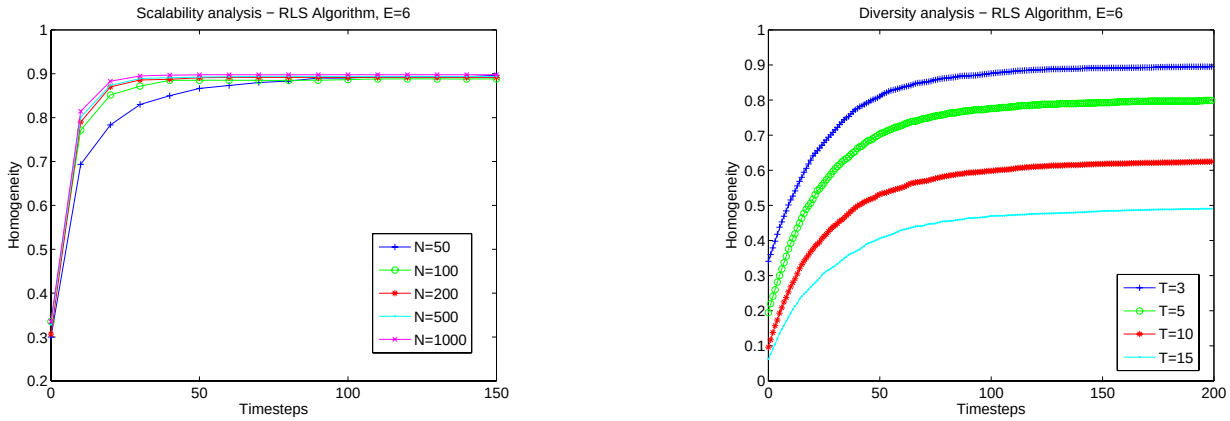
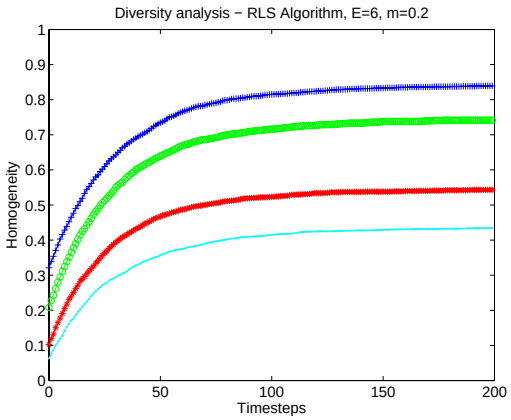


Fig. 4. Impact of system scalability (a) and diversity (b) on the randomized local search algorithm.

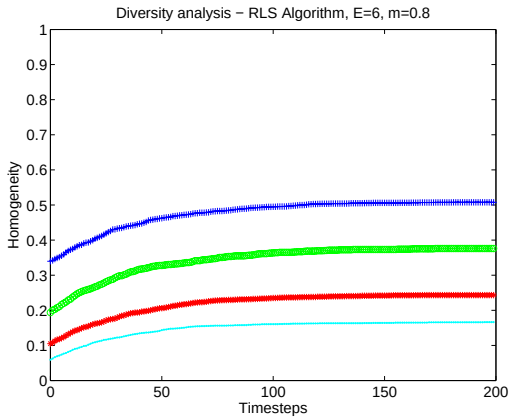
Uncoordinated load balancing has been studied, using game theory for example, in [18], [19]. Congestion games are at the heart of these works: uncoordinated, selfish clients compete to select servers providing the lowest response time (latency). Client latencies are related to the server loads. Server response time is assumed to be inversely proportional to the speed of the server, but grows with the p-th power of the number of users connected to the server. As we point out in Section I this framework is not suitable to meet the connectivity requirements of the service overlay.

The work presented in [16] from which we borrow the system model achieves load balancing among service replicas through overlay routing. The link metric adopted to select overlay service paths is inversely proportional to the capacity available at nodes on the service path. Note that load information is disseminated in the overlay so as to update the link cost metric.

The works that are closely related to ours are presented in [8], [9], [11], [13]. Fabrikant *et. al.* first present the network creation game that they use to characterize Internet topologies [11]. Their model accounts for link creation and communication costs and the resulting graph is generated by the unilateral interaction of selfish players. This model is extended in [9], where the authors consider equilibrium topologies that arise from the bilateral interaction of selfish players; in their model, link cost is paid by both vertices of an edge in the equilibrium graph. [8] presents an extension of the basic model in [11] wherein an application-level routing overlay is constructed by selfish overlay nodes. In [13], the authors study a class of simple protocols that aim to self-

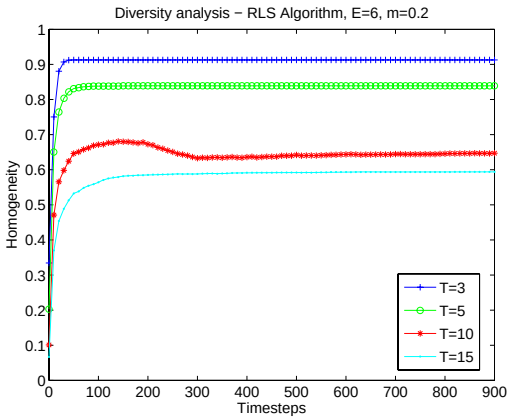


(a)

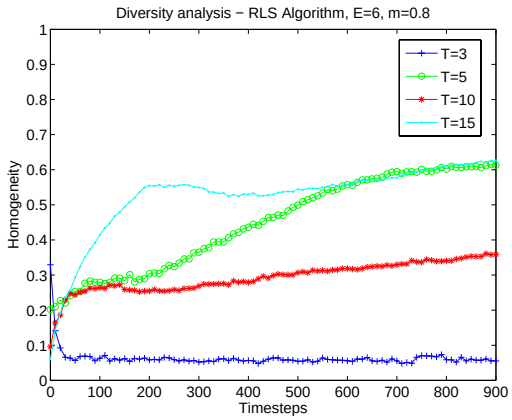


(b)

Fig. 5. Impact of an increasing fraction of misbehaving nodes of type 1 on the randomized local search algorithm for a service overlay with different number of service types ($\hat{T}$).

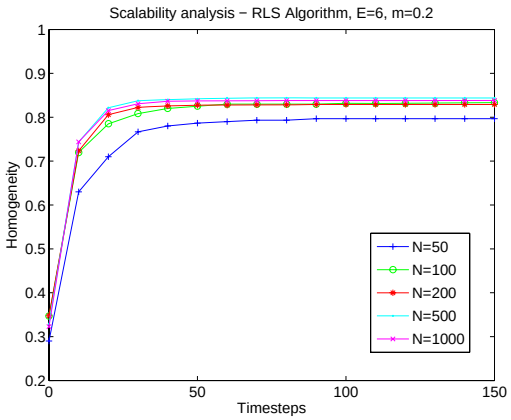


(a)

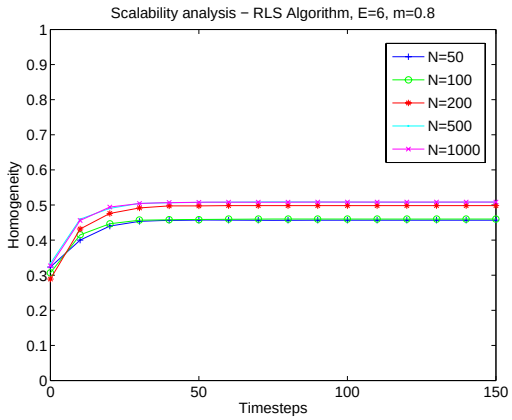


(b)

Fig. 6. Impact of an increasing fraction of Misbehaving nodes of type 2 on the randomized local search algorithm for a service overlay with different number of service types ($\hat{T}$).



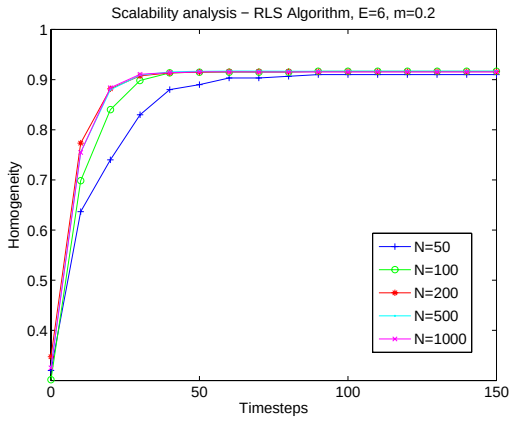
(a)



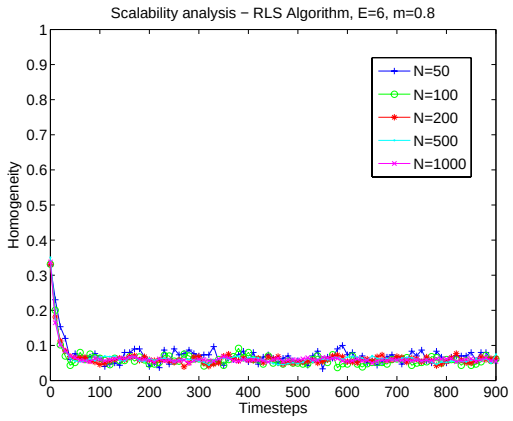
(b)

Fig. 7. Impact of an increasing fraction of Misbehaving nodes of type 1 on the randomized local search algorithm for different sizes of the service overlay.

organize P2P networks into clusters of altruistic nodes that help each other to complete jobs requiring diverse skills. The authors also use a modeling approach akin to game theory, but explore the emergence of clusters when peers mimic the strategies of neighboring nodes. Our work is in line with [13] regarding the definition of a method where nodes select



(a)



(b)

Fig. 8. Impact of an increasing fraction of Misbehaving nodes of type 2 on the randomized local search algorithm for different sizes of the service overlay.

behaviors for the good of the overlay even though their actions are based on individual greedy utility maximization.

VII. Conclusion and Future Work

In this paper we presented a game theoretic model and a distributed algorithm for the formation of service overlay network. We analyzed three methods to attain equilibrium overlay topologies based on variants of the iterated best response algorithm and on an ILP formulation. The game theoretic analysis presented in the first part of this work allowed to formalize the problem of the overlay creation and characterize its structure, while providing intuitive results. However, the optimization methods proposed using the game theoretic framework had the underlying problem of requiring the global knowledge of the overlay graph. Hence, with the goal of presenting a fully distributed algorithm and improving the computational tractability of the service overlay formation process we proposed a local heuristic which mimics the game theoretic approach and studied the emergence of structured overlay topologies, wherein nodes sharing similar traits aggregated to form clusters. We studied the scalability of our heuristic and the impact of the system diversity, showing that node aggregation based on service types is efficient even when the system is large and when the number of different service instances is high. As a glimpse to another problem akin to a game theoretic formulation we studied aggregation dynamics when a fraction of overlay nodes misbehave following different models and showed that node misbehavior due to lack of incentives may jeopardize the aggregation process. As part of our future work, we seek to enrich our approach by assuming a time-varying workload and study overlay creation in the context of repeated games. We intend to explore incentive mechanisms to foster the participation of overlay nodes in the aggregation process.

Acknowledgments

This work has been partially funded by the Integrated Project CASCADAS (FET Proactive Initiative, IST-2004-2.3.4 Situated and Autonomic Communications) within the 6th IST Framework Program.

References

[1] Planet lab. http://www.planet-lab.org/.
[2] Snap: Structured overlay networks application platform. http://www.planet-lab.org/.
[3] N. Alon, Y. Azar, G. J. Woeginger, and T. Yadid. Approximation schemes for scheduling. In *Proc. of ACM SODA*, 1997.
[4] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proc. of ACM SOSP*, 2001.
[5] A. Avidor, Y. Azar, and J. Sgall. Ancient and new algorithms for load balancing in the l_p norm. *Algorithmica*, 29:422–441, 2001.
[6] A. Awerbuch, A. Azar, E. F. Grove, P. Krishnan, M. Y. Kao, and J. S. Vitter. Load balancing in the l_p norm. In *Proc of IEEE FOCS*, 1995.
[7] A. L. Barbasi, R. Albert, and H. Jeong. Mean-field theory for scale-free random networks. *Physica*, A 272:173–187, 1999.
[8] B-G. Chun, R. Fonseca, I. Stoica, and J. Kubiatowicz. Characterizing selfishly constructed overlay routing networks. In *Proc. of IEEE INFOCOM*, 2004.
[9] J. Corbo and D. Parkes. The price of selfish behavior in bilateral network formation. In *Proc. of ACM PODC*, 2005.
[10] Z. Duany, Z. Zhangy, and Y. T. Houz. Service overlay networks: Slas, qos and bandwidth provisioning. In *Proc. of IEEE ICNP*, Paris, France, November 2002.
[11] A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. On a network creation game. In *Proc. of ACM PODC*, 2003.
[12] N. Laoutaris, G. Smaragdakis, A. Zestravos, and J. W. Byers. Implications of selfish neighbor selection in overlay networks. In *Proc. of IEEE Infocom*, 2007.
[13] A. Marcozzi and D. Hales. Emergent social rationality in a peer-to-peer system. Technical report, Department of Computer Science, University of Bologna, 2006.
[14] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *Proc. of IPTPS*, Cambridge, MA, USA, March 2002.
[15] S. Philips and J. Westbrook. Online load balancing and network flow. In *Proc. of ACM STOC*, 1993.
[16] B. Raman and R. H. Katz. Load balancing and stability issues in algorithms for service composition. In *Proc. of IEEE INFOCOM*, 2003.
[17] F. Saffre, J. Alloy, M. Shackelton, and J. L. Deneubourg. Self-organized service orchestration through collective differentiation. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 26(6):1237–1246, 2006.
[18] S. Suri, C. D. Toth, and Y. Zhou. Selfish load balancing and atomic congestion games. In *Proc. of SPAA*, 2004.
[19] S. Suri, C. D. Toth, and Y. Zhou. Uncoordinated load balancing and congestion games in p2p systems. In *Proc. of HOT-P2P*, 2004.