A BROKER BASED CONSUMPTION MECHANISM FOR SOCIAL CLOUDS

Ioan Petri[1], Magdalena Punceva[2], Omer F. Rana[3], George Theodorakopoulos[3] and Yacine Rezgui[1]
[1]School of Engineering, Cardiff University, UK
[2] Institute for Computer and Communication Systems, University of Applied Sciences, Western Switzerland
[3] School of Computer Science & Informatics, Cardiff University, UK
contact author: petrii@cardiff.ac.uk

Abstract

The new consumption without ownership paradigm is leading towards a "rental economy" where people can now rent and use various services from third-parties within a market of "shared" resources. The elimination of ownership has increased the marginal utility of consumption and reduced the risks associated with permanent ownership. In the absence of ownership the consumption in the global marketplace has become more dynamic and has positively impacted various economic and social sectors. The concept of "consumption without ownership" can also be used in the area of cloud computing where the interaction between clients and providers generally involves the use of data storage and computational resources. Although a number of commercial providers are currently on the market, it is often beneficial for a user to consider capability from a number of different ones. This would prevent vendor lock-in and more economic choice for a user. Based on this observation, work on "Social Clouds" has involved using social relationships formed between individuals and institutions to establish Peer-2-Peer resource sharing networks, enabling market forces to determine how demand for resources can be met by a number of different (often individually owned) providers. In this paper we identify how trading and consumption within such a network could be enhanced by the dynamic emergence (or identification) of brokers – based on their social position in the network (based on connectivity metrics within a social network). We investigate how offering financial incentives to such brokers, once discovered, could help improve the number of trades that could take place with a network, thereby increasing consumption. A social score algorithm is described and simulated with PeerSim to validate our approach. We also compare the approach to a distributed dominating set algorithm – the closest approximation to our approach.

Keywords: Cloud computing; Social Networks; Dominating Set; Economic Models; Consumption; Ownership.

1. INTRODUCTION

In recent years the mode of acquisition and use of resources has changed significantly, with consumers expecting to use a product from one vendor for a short amount of time, and renting rather than owning the product. Resources/products which fall within this remit have ranged from cars to movies, games and music recordings. Such a change in emphasis has been influenced by variability in markets affected by aspects such as seasonality and the temporary nature of exchange. Consumers are therefore motivated to participate in a leasing economy where products are used for a shorter period significantly preferring to rent than to purchase (Bendell, 2007), (Levenson, 2007). The ability to participate in such a sharing economy also provides greater choice for both the consumer and the provider, enabling a much greater flexibility in being able to switch between multiple market offerings, thereby also likely to increase consumption from consumers by not being restricted to products or price constraints from a single vendor. The absence of ownership also enables access to some existing services that may be inaccessible previously due to high cost of ownership (Living Planet, 2012). By engaging in such a non-ownership market, consumers can have access to greater and increased social status with less cost (Moore, 2008), (Russell, 2007).

Consumption without the cost of ownership has been identified by economists (Winsper, 2007), (Zukin, 2008) as a new paradigm in the emerging "sharing economy". Increasing use of social networks, data mashing/aggregation, availability of software platforms that facilitate such service/data aggregation and the availability of handheld devices providing easy access to such platforms enable users and providers of resources/services to discover each other and utilize trust relationships developed over time. Such trust relationships are often encoded in interaction patterns and behaviours that can be derived from (on-line) social networks. These relationships therefore provide the basis for evaluating people that one can trade with. Increasingly, there is also reluctance in making large capital purchases of equipment and hardware, making it more lucrative for users to monetize their time and assets. Trust and reputation play a central role in an economy based on the "consumption without ownership" model, therefore these pre-established relationships would be essential to encourage greater transactions between participants.

In a sharing market being proposed in this paradigm, market actors (users requesting services and providers offering them) can also act as micro-entrepreneurs or brokers. Enabling discovery of suitable providers able to meet particular, often specific and individual demand from consumers, becomes an essential tenet in such systems, especially with the ability to also associate some degree of confidence in the likelihood of the provider being able to meet their advertised capability. Consumers become independent contractors, working for themselves with control over their working time and working conditions. An example is the case of ride-sharing companies such as Lyft, Sidecar or UberX which own no cars themselves but they sign up instead ordinary car owners: when people need a ride, they can use mobile apps to find a driver nearby and ask to be picked up. Airbnb represents another example of over 300,000 listings from people making their apartments and homes available for short-term rent, similarly SnapGoods makes it possible for people to borrow consumer goods from other people in their neighbourhood or social network. A variety of examples exist today in other sectors (Benny, 1973), (Surowieski, 2013).

As brokers play a key role in such a “consumption without ownership” paradigm, identifying where such brokers should be situated and how many are needed become important challenges. Such brokers should also enable trust relationships to be established between consumers and providers to allow concerns about liability and competence to be addressed. Ride sharing companies such as Sidecar, Lyft and Uber often need to also implement and conform to certain safety and driver regulations. We believe an equivalent capability is needed for other domains.

As the demand for data and computational services increases, the benefits of Cloud computing become substantial. However, Cloud computing capabilities (as currently provisioned) can prove to be limited when accessed through a single provider. Due to vendor lock-in and specialist data models required from a single vendor, it is in a user’s interest to explore and interact with multiple possible Cloud providers. Extending capabilities of Clouds by using user owned and provisioned devices can address a number of challenges arising in the context of current Cloud deployments – such as data centre power efficiency, availability and outage management. We have investigated such “Social Clouds” in a number of contexts previously (Chard et al., 2011), (Petri et al., 2012). Social Clouds are developed using the observation that like any community, individual users of a social network are bound by finite capacity and limited capabilities. In many cases however, other members (friends) may have surplus capacity or capabilities that, if shared, could be used to meet the fluctuating demand. A social cloud makes use of trust relationships between users to enable mutually beneficial sharing. Social Clouds are defined in (Chard et al., 2011) as “a resource and service sharing framework utilizing relationships established between members of a social network.” The availability of storage resources and access latency are also significantly improved – as storage resources may be found in closer proximity to a user. The establishment of such Peer-to-Peer (P2P) community Clouds requires a robust mechanism for controlling interactions between end-users and their access to services. For instance, in the context of such a Cloud model, end-users can contribute with their own resources in addition to making use of resources provided by others (at different times and for access to differing services) (Grivas et al., 2010). There is also increasing interest in developing “distributed Cloud” platforms, which are able to orchestrate capability across multiple federated Cloud systems, see for instance work on such a Cloud orchestration system from Ericsson in the European UNIFY project (UNIFY, 2013).

In previous work, we have also investigated incentive models for users to provide services to others (Chard et al., 2011), (Punceva et al., 2012) – which can range from bartering of resources, improving the social standing of a participant within a community or obtaining a financial reward. We focus on the last of these incentives in this work. Often in such markets it is necessary for a client to discover suitable providers of interest. This is generally undertaken through the use of either a registry service (centralized) to the use of a discovery request being propagated across the network (a variety of approaches have been considered, ranging from flooding, controlled “gossiping” to multiple federated registries). We propose a decentralized approach whereby some sellers or buyers may become brokers (or “traders”) in order to improve their own revenue within a market place, based on their social connectivity within a network. We consider a number of graph theoretic measures (such as connectivity degree, centrality, etc) to identify how nodes within a social Cloud which were initially buyers or sellers could turn into brokers – to improve their own revenue and satisfy service requests within the market. We map our problem into a dominating set problem in graph theory and show how our results compare with a distributed implementation of this algorithm.

In section II we identify the role of brokers within a P2P market – and how the number of brokers influences the interaction dynamics within the network. The main concepts of our approach are outlined in section III. In section IV we outline our overall methodology, with a description of the social score algorithm and the metrics (degree & centrality) used within the algorithm to identify nodes that could be potential brokers. A description is also provided of the PeerSim simulator we used to evaluate various scenarios. Results are presented in section V, with Conclusions and future work in section VI.

2. RELATED WORK

Intermediaries or brokers bring together participants (users and providers) who have not directly interacted with each other. Brokers have been used extensively in service-based systems, primarily as an alternative to service "exchanges" and registry services. Such brokers may be managed by external third parties, who have an economic incentive to provide accurate matchmaking support (utilizing both the functional capability provided/required including other non-functional attributes which have been acquired by a broker over time, such as failure rate, performance, cost, etc) between the capabilities of a provider and the demands of a user. Brokering solutions are now beginning to emerge in Cloud systems also – especially with the emergence of Web sites such as CloudHarmony (which can support performance benchmarking across over 100 different Cloud providers). With Social Clouds (as identified in section I), broker-based interaction becomes even more important, as providers can exist over shorter time frames and offer specialist capability (Sundareswaran, 2012), (Nair et al., 2010).

Sotiriadis et al. (Sotiriadis et al., 2013) propose a meta-brokering decentralized approach to manage interactions between interconnected Clouds. The objective is to support Cloud interoperability and resource sharing. In this framework a broker acts on behalf of the user and generates requests for resources from the Cloud system, based on the contacted SLAs. The authors demonstrate that the meta-broker model outperforms a standard broker when the system contains a high number of concurrent users and cloudlets submissions.

Sundareswaran et al. (Sundareswaran et al., 2012) propose a broker-based architecture where brokers help end users select and rank Cloud service providers based on prior service requests, enabling users to negotiate SLA terms with providers. An efficient indexing structure called the CSP (Cloud Service Provider) index is used to manage the potentially large number of service providers, utilizing similarity between various properties of service providers. The CSP-index can subsequently be used for service/provider selection and service aggregation.

STRATOS (Pawluk et al., 2012) is a broker service to facilitate multi-cloud, application topology platform construction and runtime modification in accordance with a deployer's objectives. STRATOS allows an application deployer to specify what is important in terms of Key Performance Indicators, so that Cloud system offerings can be compared and ranked based on these indicators. The authors demonstrate how an application distributed across multiple Clouds can decrease the cost of deployment. Duy et al. (Duy et al., 2012) propose a benchmark-based approach to evaluate and compare cloud brokers. A benchmark called Cloud Broker Challenge (CBC) is employed to describe the cloud providers, cloud consumers, across 5 difficulty levels – inspired by the successes and impact of Semantic Web Service Challenge (a set of benchmark problems in mediating, discovering, and composing web services) . By introducing difficulty levels for Cloud brokering and associated scenarios, the authors try to abstract the fundamental properties of various Cloud providers to better understand how broker-based solutions could be applied across multiple providers simultaneously (Leskovec, 2010).

Our approach complements these situations, in that we already assume that brokers play an important role within a Cloud-based resource sharing environment. Our key objective, instead, is to understand how many brokers should co-exist within a system to enable better interaction between users and providers, whilst at the same time ensuring that the number of brokers is limited.

3. APPROACH

A resource trading network has a particular relevance in Social Clouds – as some resource users & providers may have a more dominant position in the system, with greater access to social opportunities for intermediation. The question of where brokers should be placed within such a social network becomes significant – primarily to: (i) increase the flow of 'goods' (i.e. facilitate resource exchange); (ii) increase social welfare within the community. Social welfare, in this case, measures the number of potential resource users who are able to find providers that match their requirements, within their budgets. We consider a marketplace where buyers and sellers can interact through an intermediate broker T. The broker receives commission for each transaction that it facilitates – the broker's objectiveis therefore to increase the number of transactions they participate in and the commission per transaction that they receive.
Our approach focuses on not having a pre-defined list of brokers – but understanding how the strategic position of a node within a network can lead it to be become a broker – which we refer to as "broker emergence". Our approach makes use of two stages to achieve this:

1) Node selection – Select nodes with the highest social score (as described in section IV-A).

2) Risk assessment – Evaluating the broker's capacity of making profit and the associated (financial) risk to lose the investment.

Broker emergence may be formulated as a dominating set problem. A dominating set for a graph G = (V, E) is a subset D of V such that every vertex not in D is joined to at least one member of D by some edge. The problem minimum Dominating Set (MDS) requires finding a dominating set of minimum size. Our goal is to find a set of nodes that will act as brokers among a network of socially connected nodes here every node is either a buyer or a seller. The selected set of brokers should satisfy the following condition:
Market Accessibility: Every non-broker node should be connected to at least one broker.

Our Market Accessibility condition ensures that every non-broker node can participate in a transaction either as a buyer or as a seller. Therefore our problem can be formulated as finding a Minimum Dominating Set (MDS): given a graph G(V, E) that represents a social network where vertices represent users and edges represent friendship links, finding the set of broker nodes corresponds to finding the minimum dominating set. The dominating set problem is a classical NP-complete problem and several approximation algorithms exist for finding MDS. Kuhn et al. (Kuhn, 2005) propose a distributed approximation algorithm for finding a dominating set of minimum size which means every node uses only local information when executing the algorithm. This algorithm is particularly suitable for large-scale decentralised networks and we use it here.
As nodes can change their roles of buyers/sellers, a broker may be connected to buyers only or sellers only. We consider two alternatives to overcome this: (i) brokers will attempt to connect to other brokers (perturbing the original social structure); (ii) apply the (approximation) algorithm for finding a connected minimum dominating set instead of a minimum dominating set (which may not be connected). Such algorithm although not distributed is presented in Guha et al (Guha, 1998). It ensures that every broker node is connected to at least one other broker node.

4. Methodology

We consider a network with an associated set of peer-nodes $P=\{p_1, p_2, p_3,\dots,p_n\}$, and a sub-set $S=\{p_1, p_2, p_3, \dots, p_m\}$, $m < n$, $S \subset P$, where S represents the set of non-leaf nodes from P . We use two algorithms for selecting broker nodes: social score algorithm and dominating set algorithm.

A. Social Score Algorithm

We apply the social score selection algorithm over the set of non-leaf nodes S. The selection process can be modeled as a function $f(x) : S \rightarrow T$, where the result is a sub-set of peer-nodes T with the highest social score which we consider as brokers. The selection protocol for brokers is built around the notion of social score.

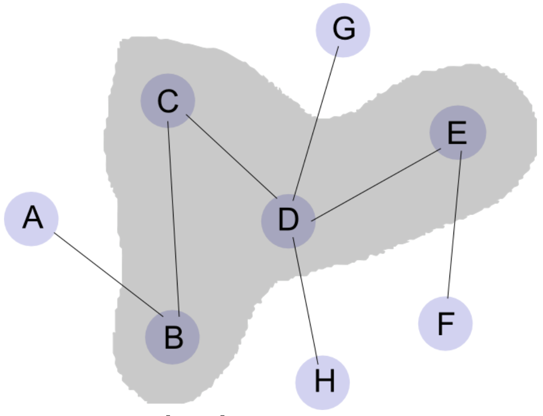

Figure 1. The selection

We use social score as a metric to evaluate nodes and select brokers. Social score is calculated as an average of three metrics used to assess the connectivity of a node within a graph – a node that has greater potential to link other nodes with each other has a higher social score. The metrics we use are a node's: (1) Degree Centrality, (2) Betweenness Centrality and (3) Eigenvector centrality. Within a graph G(V, E) where V is the set containing the number of vertices and E is the set containing the number of edges, we define the following metrics:

Node's degree centrality – is simply the number of links incident to the node:

$$DC(v) = \deg(v)$$

Node's betweenness centrality – Betweenness centrality quantifies the number of times a node acts as a bridge along the shortest path between two other nodes and is calculated as the fraction of shortest paths between node pairs that pass through the node of interest:

$$BC(v) = \left(\frac{p(s,t/v)}{p(s,t)}\right) \qquad s,t \in V$$

where p(s, t/v) is the number of shortest paths between users s and t that pass through node/user v, and p(s, t) is the number of all the shortest paths between the two users s and t.

Node's Eigenvector centrality – defines the influence of the node within a network – i.e. it measures how closely a node is connected to other well connected nodes. It assigns relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes. Hence the objective is to make x_i proportional to the centralities of its n neighbors, i.e.:

$$EC(x_i) = \frac{1}{\lambda}\sum_{j=1}^{n} A_{i,j}x_j$$

λ is a constant. In vector notation this can be written as X = λ Ax, where λ is an eigenvalue of matrix A if there is a non-zero vector x, such that Ax = λx. Thus, we classify nodes from the perspective of their social score which is calculated as:

$$SS = \frac{BC + DC + EC}{3}$$

Figure 1 illustrates the set of non-leaf nodes for which we calculate the social scores as a basis to support broker selection. Each selected node is given a certain amount of capital which can either be the same for all nodes, or in proportion to their social score. Algorithm 1 explains how brokers are selected based on the social score associated with nodes. The variable degree represents the number of current connections whereas capacity represents the maximum number of connections a node can support. This variable can be either specified in a configuration file or set as the maximum degree of the social graph: *capacity = max(degree).*

Algorithm 1: Brokers Selection

```
1:  len:=node.capacity;
2:  pos:=0;
3:  set:=null;
4:  for i := 0 to networkSize by 1 do
5:      len := pos; max := maxRounds; found := false;
6:      while (!found) and (len> 0) and (max> 0) do
7:          max− −; r[i] := selectNewNode(); rpeer := null;
8:          size:=getNodeDegree(r[i]);
9:          brokerObserver.calculateScore(r[i].getIndex());
10:         if (r[i].isActive()) and (capacity < r[i].capacity) then
11:             rpeer := getNodeId(r[i]);
12:             markNode(rpeer,r[i]);
13:             addToSet(rpeer,r[i],set);
14:             found:=true;
15:         else
16:             if (degree ≤ r[i].degree) and (r[i].degree <
                rpeer.getTarget()) then
17:                 markNode(rpeer,r[i]);
18:                 addToSet(rpeer,r[i],set);
19:                 found := true;
20:             end if
21:         end if
22:     end while
23:     if (rpeer := null) or (r[i].IsBroker()) or (r[i].Size ≥
        rpeer.getTarget()) or (r[i].isActive()) then
24:         removeNode(rpeer);
25:     end if
26: end for
```

We use a set variable for storing nodes and a marking mechanism *markNode* for identifying all those brokers over a set of simulation rounds *maxRounds*. The algorithm starts by excluding the leaf nodes and calculating the social score for each the non-leaf nodes. The brokers are then selected as the nodes with the highest social score out of all non-leaf nodes.

Algorithm 1 attempts to identify a minimum number of brokers within the network. The algorithm uses a classification criteria based on the social score measure introduced above: nodes with higher social score are considered better candidates as brokers. The target set of brokers is composed by the minimum set of nodes with highest social score whose total capacity is sufficient to cover all the remaining nodes (sellers and buyers).

B. *Dominating Set Approximation: Distributed Algorithm*

The dominating set distributed algorithm is an approximation algorithm for solving the dominating set problem from (Kuhn, 2005). The algorithm relies on a linear programming (LP) formulation of the problem and consists of two parts/algorithms: first algorithm calculates the fractional solution to the LP problem and the second algorithm does the rounding part. The algorithm runs in constant time and has a provable approximation ratio

Algorithm 2: LPMDS Approximation

```
1:  $x_i := 0$;
2:  calculate$\delta(i)$ ;
3:  $\gamma^{(2)}(v_i) := \delta_i + 1; \delta(v_i) := \delta_i + 1$;
4:  for l:=k-11 to 0 by -1 do
5:      $(*\delta(v_i) \leq (\Delta + 1)^{(l+1)/k}, z_i := 0*$;
6:      for m:=k-1 to 0 by -1 do
7:          if $(\delta(v_i) \geq \gamma^2(v_i)^{l/l+1})$ then
8:  send 'active node' to all neighbors;
9:          end if
10:         $a(v_i) := |j \in N_i | v_j$ is activenode | ;
11:         if $color_i$ = gray then then
12:             $a(v_i) = 0$;
13:         end if
14:         $a(v_i)$ to all neighbors;
15:         $a^{(1)}(v_i) := \max_{j \in N_i} a(v_i)$;
16:         $*a(v_i), a^{(1)}(v_i) \leq (\Delta + 1)^{m+1/k}*$
17:         if $\delta(v_i) \geq \gamma^{(2)}(v_i)^{l/l+1}$ then
18:             $x_i := \max x_i, a^{(1)}(v_i)^{(-m+1)}$;
19:         end if
20: send $x_i$ to all neighbors;
21:         if
22:             $color_i$ := gray ;
23: send $color_i$ to all neighbors;
24:             $\delta(v_i) := |j \in N_i | color_j$ = white |;
25:         end if
26:         $*z_i \leq (1 + (\Delta + 1)^{1/k})/\gamma^{(1)}(v_i)^{(l/l+1)}*$
27: send $\delta(v_i)$ to all neighbors;
28:         $\gamma^{(1)}(v_i) := \max_{j \in N_i} \delta(v_j)$;
29: send $\gamma^{(1)}(v_i)$ to all neighbors;
30:         $\gamma^{(2)}(v_i) := \max_{j \in N_i} \gamma^{(1)}(v_j)$
31:     end for
32: end for
```

. For an arbitrary possibly constant parameter k and maximum node degree Δ, the algorithm computes the

dominating set of expected size O (kΔ2k log(Δ)|DSOPT |) in O(k2) rounds. Where |DSOPT | represents the size of the optimal dominating set.The output of the algorithm is the vector x defined for all
$v_i \in V$ which has values 0 or 1 and indicates whether a node is in the dominating set or not: if $x_i = 1$ then node v_i is in the dominating set and if $x_i = 0$ the node is not in the dominating set. Initially each node independently runs the first part of the algorithm (Algorithm 3 from Kuhn et al. (Kuhn, 2005)) and as a result returns a fractional value between 0 and 1 for x_i variables. In accordance with this approach we use the following notations. Initially all nodes are colored white. A node is colored gray if the sum of the weights of x_j for $v_j \in N_i$ exceeds 1, i.e., as soon as node is covered. The degree of a node v_i is denoted δ_i. The largest degree in the network graph is denoted Δ. The notation $\delta^{(1)}_{=} = \max_{j \in N_i} \delta_j$ is the maximum degree of all nodes in the closed neighborhood N_i of v_i. Similarly $\delta^{(2)}_{=} = \max_{j \in N_i} \delta^{(1)}_j$ is maximum degree of all nodes at distance at most two from v_i. Note that it is assumed each node knows its 2-hop neighbors and these values therefore can be computed in at most two communication rounds. A dynamic degree of a node v_i is denoted by $\bar{\delta}$ and represents the number of white odes in Ni, the neighborhood of vi. The output of the first part Algorithm 3 in (Kuhn, 2005)) are fractional values for xi and the second part (Algorithm 1 in (Kuhn, 2005)) does the rounding: it takes fractional xi values as input and rounds them to 0 or 1.

C. Modeling trading process

In our framework each trade is defined as a function $f(t) : S \rightarrow D$ where S represents a domain containing originating nodes and D the domain containing destination nodes. Each transaction f(t) brings an associated revenue for brokers and is scheduled to happen at a specific simulation cycle. Within the protocol we enable peer-nodes to change roles over time such that buyers and sellers can become brokers or brokers can become buyers or sellers. In order to validate our hypotheses, PeerSim (Jelasity et al., 2010) was chosen as a framework for simulating a number of different scenarios. The PeerSim simulator uses separate source files for programming different needed controllers of the simulation process. We therefore employ a number of different parameters and controllers for simulating the scenarios reported in section V. We use an initialization controller defining the various types of events that can happen during the simulation and which need to be scheduled during the simulation. Another controller is used for defining the network variation at each simulation cycle (e.g. how the network changes when adding new nodes to the network) for each round of trading.
An additional controller is allocated as an observer that collects the results for each experiment. The configuration file also contains a number of simulation parameters:

•cycles: defines the maximum number of simulation cycles for each experiment.
•maxCapacity defines the maximum number of connections allowed for any given node.
•minCpacity defines the minimum number of connections allowed for any given node.
•minTrades defines the minimum number of trades scheduled to be run within the system as a whole.
•maxTrade defines the maximum number of trades scheduled to be run within the system as a whole.

To support a dynamic network formulation – whereby nodes may be added or removed from the network, we used an additional network dynamics module.

- control.c1 peersim.dynamics.DynamicNetwork
- control.c1.type v_{type}
- control.c1.maxsize v_{max}
- control.c1.add v_{add}
- control.c1.step v_{step}
- control.c1.from v_{from}
- control.c1.until v_{until}

The *DynamicNetwork* is a module provided within PeerSim which enables us to define a simulation with a differing number of nodes at each simulation cycle. It includes various Java packages initializing a network or modifying it during simulation. The type parameter represents the type of the node to be added, the *maxsize* parameter represents the maximum number of nodes that one simulation process can use; the *add* parameter defines the number of nodes injected at each step; the *step* parameter defines the frequency in cycles for each injected node. The parameter *from* specifies the starting number of nodes to simulate while the *until* parameter defines the maximum limit on the number of nodes that the simulation can use.

Table 1. Simulation data set from epinions.com

Nodes	75879
Edges	508837
Nodes in largest WCC	75877 (1.000)
Edges in largest WCC	508836 (1.000)
Nodes in largest SCC	32223 (0.425)
Edges in largest SCC	443506 (0.872)
Average clustering coefficient	0.2283
Number of triangles	1624481
Fraction of closed triangles	0.06568
Diameter (longest shortest path)	13
90-percentile effective diameter	5

We use two different metrics to measure the status of the system:

(i) Volume of trades: defining the total number of trades *f(t)* taking place within the system;

(ii) Average revenue: measures the average revenue per broker. In equation 5 *n* defines the number of trades within the system, m defines the number of brokers and *val(f(t_i))* represents the associated revenue of each trade.

$$AR = \frac{1}{m}\sum_{i=1}^{n}(val(f(t_i)))$$

5. SIMULATION AND RESULTS

Our simulation makes use of social network data about epinions.com obtained from the Stanford Network Analysis Platform (SNAP) project (SNAP, 2012).

We use this particular data set as it exposes trust relationships that are formed within a social network for product recommendation, exposing the Web of Trust between individuals.

We also felt that this data set is representative of the types of buyer-seller-broker relationship that we could foresee within a Social Cloud – based on the referrals or recommendations made between people. Table I provides a description of the *Epinions* data set.

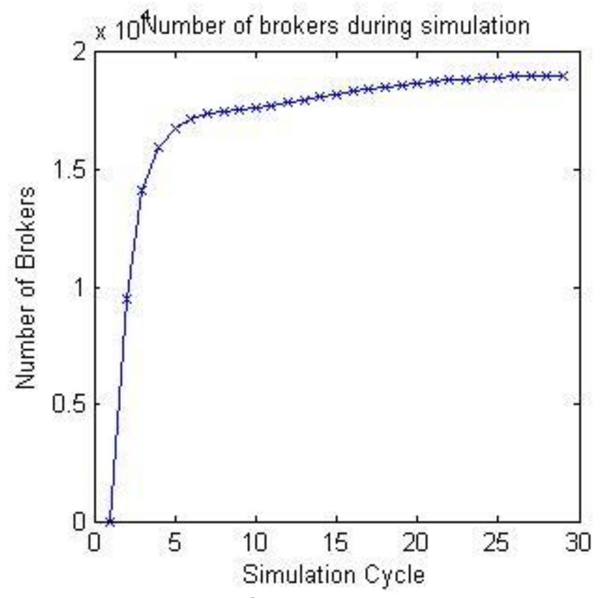

Figure 5a. Brokers emergence

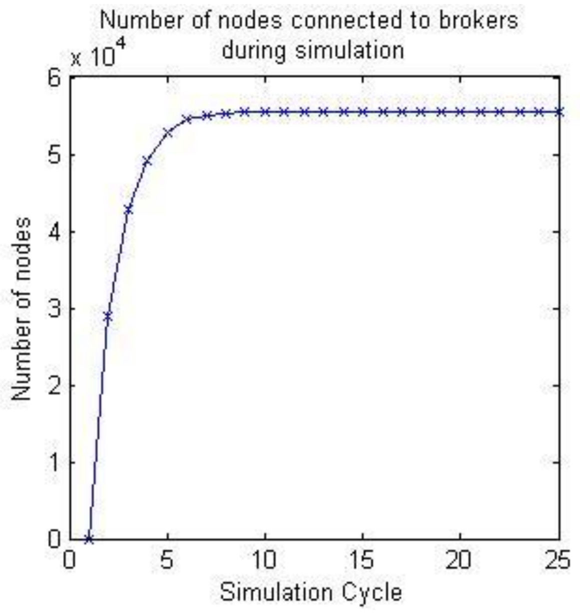

Figure 5b. Degree of brokers

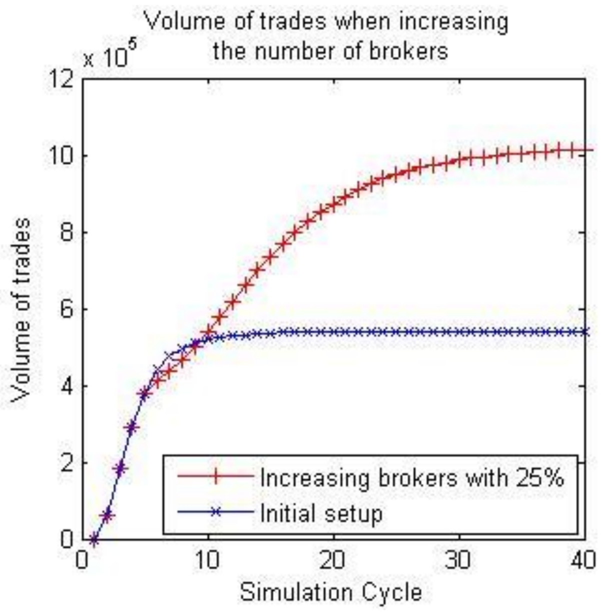

Figure 5c. Volume of trades with brokers

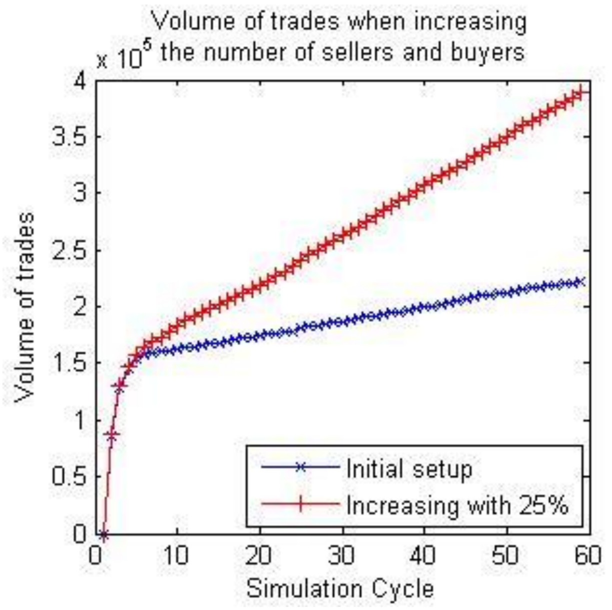

Figure 5d. Volume of traders with buyers and sellers

Experiment 1: This experiment measures how the number of brokers evolves during the simulation in relation to the initial network configuration – containing only buyers and sellers. The number of nodes, number of edges, average network degree provided in table I are used to initially start the network in bi-partite (buyer, seller only) mode. Brokers are gradually selected based on the algorithm 1 presented in section IV-A. From figure 5a it can be observed that the simulator needs around 6 simulation cycles to select brokers. During simulation, the process of broker selection works in parallel to the actual trading (i.e. trading starts when the first broker has been identified and continues during simulation). After 6 simulation cycles the number of brokers within the system becomes stable – although trading within the network still continues to take place.

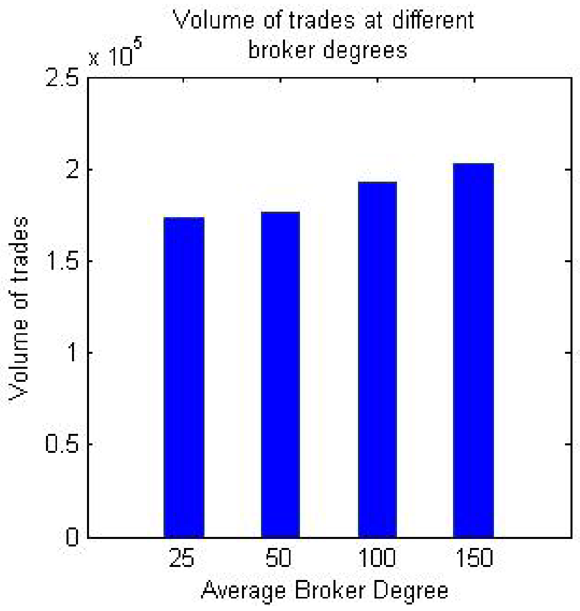

Figure 5e. Volumes of trades at broker degrees

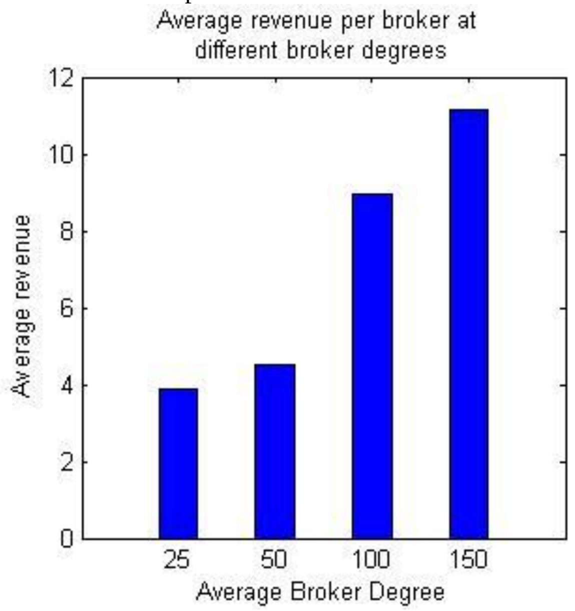

Figure 5f. Average revenue per broker

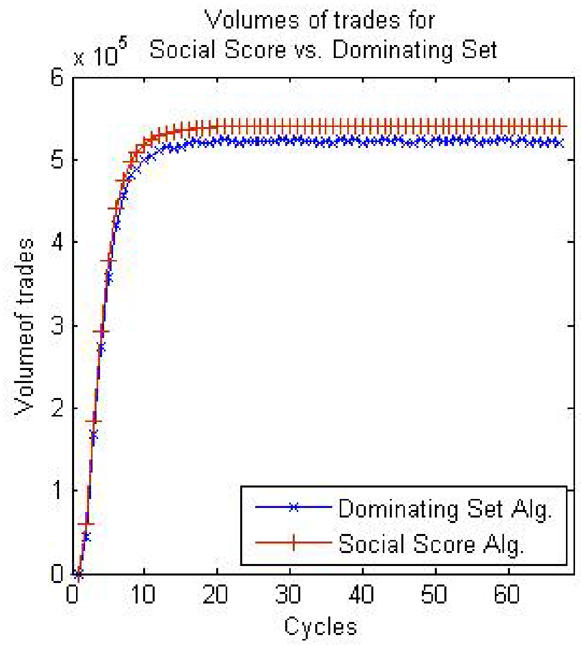

Figure 5g. Number of brokers: Social score vs. Dominating set problem

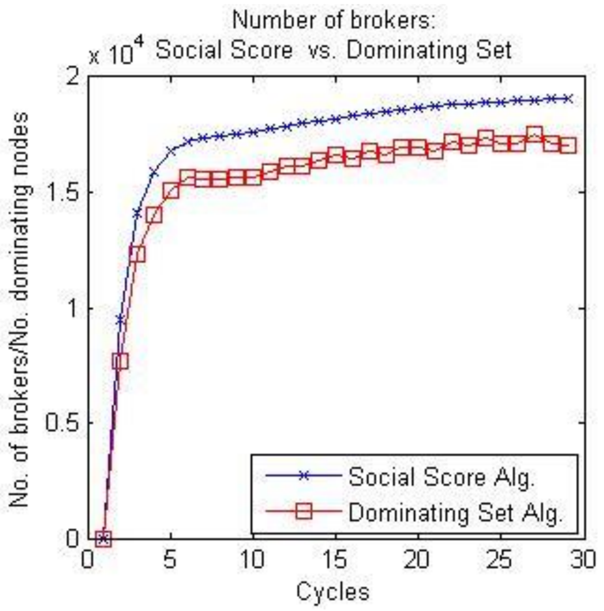

Figure 5h. Volume of trades: Social score vs. Dominating set algorithm

Experiment 2 – This experiment presents how the average number of nodes connected to brokers evolves during the simulation – with the initial setup provided in table I.
The average numbers of nodes connected to brokers identify sellers and buyers within the network.
From figure 5b we observe that the number of nodes connected to brokers gradually increases within the first 6 cycles of the simulation. Hence, as the number of brokers increases, the number of nodes (buyers/sellers) associated with a broker changes. This process of a change in node interactions (buyer/broker, seller/broker) is strongly related to the process of broker emergence.

Experiment 3 – Volume of trades when increasing the number of brokers. In this experiment we measure the volume of trades when increasing the number brokers within the network but keeping a fixed number of buyers and sellers. For running this experiment we extended the capacity of the network by adding new brokers to the simulation process.
This is ensured by the *dynamics controller* presented in section IV-C with the specific parameter *type* specifying broker as the type of node to be added. From 5c it can be observed that an increase in the number of brokers by 25% has a direct impact on the volume of trades. When adding more brokers to the network, the routes between sellers and buyers increases significantly, thus an additional volume of trades is identified. In this experiment the initial setup and the associated number of brokers are specified by the social score calculated for each node.
Experiment 4 – Volume of trades when increasing the number of buyers and sellers. In this experiment we measure the volume of trades when increasing the number of sellers and buyers within the network but keeping a fixed number of brokers.
Experiments 3 and 4 are the only two simulations where we change the structure of the network during the simulation to better understand the impact of: (i) varying number of intermediaries (Exp. 3); (ii) a change in demand/ supply over time (Exp. 4). Whereas experiment 3 investigates how an increase in the number of brokers impacts the volume of trades, in this experiment we evaluate how the volume of trades change when expanding the number of buyers and sellers. As in the previous experiment, the increase of nodes is handled by employing the *dynamics controller* with the specific parameter *type* set to node. As illustrated in figure 5d an increase of 25% in the number of buyers and sellers causes an increase in volume of trades. When more buyers and sellers are added, the number of possible trade options increases.
However, even if the increase of buyers or sellers causes an increase in volume of trades, as the number of brokers and the associated capital are limited the impact on volume of trades is less significant than the increase in brokers presented in previous experiment.

Experiment 5 – Volume of trades with regard to broker degrees. In this experiment we investigate how the volume of trades evolves with reference to a (broker) node degree. Brokers are selected according to their social score. However, each broker has an associated degree parameter specifying the number of current connections.
In this experiment we analyze the relationship between the average broker degree and the volume of trades. Figure 5e presents various levels of trades with reference to the degree of a broker. It can be observed that the volume of trades increases for brokers with higher degrees such as 50 or 75 connections. The impact on volumes of trades for those brokers with lower degrees is reduced as identified for degrees of 15 and 25.

Experiment 6 – This is used to measure the average broker revenue with regard to broker degrees. In addition to measuring the volumes of trades we also try to quantify the revenue for each broker. As it can be observed from figure 5f, the average revenue for brokers is strongly related to their network degree. Whereas for brokers with degree of 15 respectively 25 connections, the impact is low, for brokers with higher connectivity average revenue is significantly increased. A higher degree for a broker gives an increased number of options for performing trades thus leading to an increased revenue.

Experiment 7 – The number of brokers compared to the number of nodes in the dominating set when comparing the Social Score Algorithm with The Dominating Set algorithm.
In this experiment we compare the performance of the Social Score Algorithm with Dominating Set Algorithm from the perspective of number of brokers respectively number of nodes in dominating set. As presented in figure 5g, the dominating set has better performances than the social score algorithm. It can be observed that at cycle 5 the number of nodes in dominating set is with 11% lower than the number of brokers whereas at cycle 30 the difference is around 12.5%. The performance differences are determined by two important particularities: (i) graph properties and (ii) evaluation metrics.

Experiment 8 – Volume of trades when comparing the Social Score Algorithm with The Dominating Set Algorithm. In this experiment we evaluate the social score algorithm and the dominating set algorithm from the perspective of the volume of trades they generate. Figure 5h shows that the social score algorithm generates a higher volume of trades than the dominating set algorithm. This happens because the social score algorithm considers a number of different network metrics for selecting the brokers applied on a predefined social graph whereas the dominating set algorithm seeks to optimize the dominating set for an ad-hoc network.

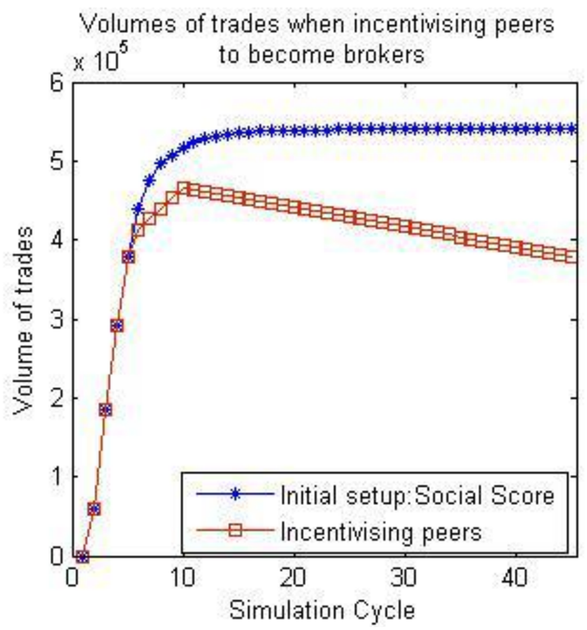

Figure 5i. Volume of trades: Social Score vs. Incentivising peers approach

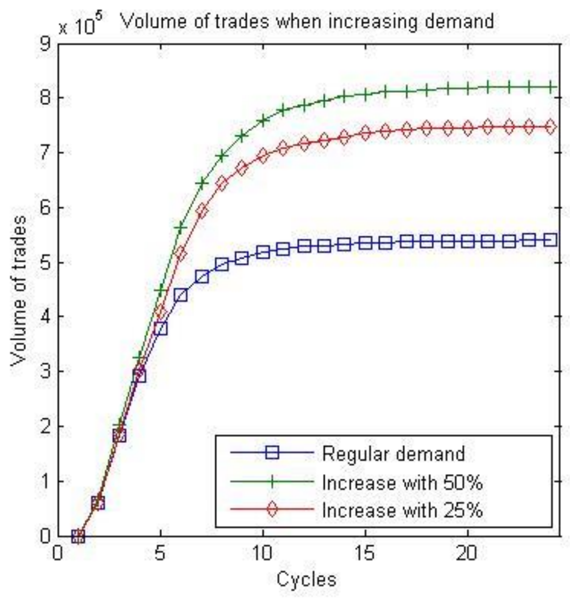

Figure 5j. Volume of trades when increasing demand

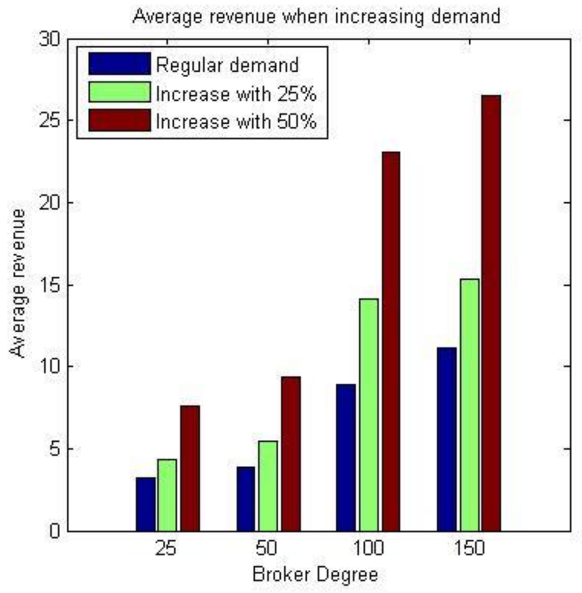

Figure 5k. Average revenue per broker when increasing demand

Experiment 9 – Volume of trades when comparing the Social Score Algorithm with the approach when we incentivize peer-nodes to become brokers. For running this experiment we assign an incentive to each peer-node in order to become a broker. Hence, a peer-node i can decide to become a broker because according to its subjective decision function f(x), the broker role enables it to maximize its revenue. Figure 5i shows that the social score algorithm generates a higher volume of trades than the incentivising approach.

This happens because the social score algorithm considers a number of different network metrics for selecting the brokers applied on a predefined social graph whereas in the incentive approach some of the brokers can derive from peer-nodes with poor network attributes such as low connectivity, low centrality degree, etc. It can be also observed that in the incentive approach the volume of trades starts to decay after a certain simulation cycle.
This happens because the brokers derived from peers with low connectivity are unable to generate a constant volume of trades within the system.

Experiment 10 – Volume of trades when increasing the demand. In this experiment we measure the volume of trades when increasing demand within the system but keeping a fixed number of buyers and sellers. In previous experiments we used a fixed demand identifying a process where one broker can intermediate a single trade between a buyer and a seller. Here, we consider that between each buyer and each seller more than one broker-intermediated trade can take place. Figure 5j illustrates a comparison between the base case where there is a regular demand and the cases where we increase the demand by 25% and 50%, respectively. We observe that the highest differential increase in volume of trades is identified when increasing demand by 25%.
This differential increase is determined by the capacity parameter associated with every broker. In this experiment, we assume that one broker has a configured capacity of trades that can be intermediated. When increasing the demand by 25% there is still enough capacity for brokers to intermediate trades whereas when increasing the demand by 50% the brokers, due to limited capacity, cannot

intermediate all the trades. The request for resources increases when the demand is increased, hence brokers will intermediate more trades generating an increase in volume of trades.

Experiment 11 – Average revenue per broker with an increase in demand. In this experiment we investigate how demand impacts the average revenue per broker. As outlined in experiment 10, an increase in demand leads to an increase in trade volume, and is affected by the degree of the broker (Figure 5k). When a broker has a degree of 25, it can be observed that the impact of demand on average revenue is limited. When using a broker degree of 50 it can be observed that the demand correspondingly impacts average revenue. This difference between various levels of demand in terms of average revenue is determined by the broker degree. Although the demand is increased, in some cases a broker can support only a specific number of trades in relation to the configured capacity, thus the impact is often limited in practice.

6. Conclusion

Consumption without ownership represents an emerging economical approach with applicability in many contexts. Enhancing consumption with a broker based market intermediation is a process commonly used in various market scenarios to enable better interaction between buyers and sellers. This concept has found applicability in P2P markets with extension to Social Clouds where a number of sellers and buyers are able to use and provide resources – driven primarily by economic incentives and their reputation in the market. We investigate a specific mechanism of broker emergence – whereby nodes in a Social Cloud can change role from buyers or sellers to brokers – in order to improve their revenue. We identify the associated benefits for supporting such broker emergence within a P2P environment. We also describe how the identification of such brokers can lead to an improved social welfare within a community.

A number of scenarios are simulated in PeerSim, by employing a heuristic social score algorithm for determining the number of brokers within the network and the associated generated volume of trades. We investigate how the algorithm performs when adding more brokers respectively buyers/sellers by measuring the volume of trades and the average revenue. In addition we compare the social score algorithm with a distributed dominating set algorithm. Broker emergence provides a useful alternative to the pre-identification of "brokers" within a Cloud system – and could lead to a dynamic environment which adapts the number and types of brokers available over time (as the system connectivity and trade volumes (based on supply/demand) change.

7. References

Stelios Sotiriadis, Nik Bessis, Nick Antonopoulos, Decentralized meta-brokers for inter-cloud: Modeling brokering coordinators for interoperable resource management, 9th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 2462-2468, Chongqing, China,2012.

Sundareswaran, S., Squicciarini, A., Lin, D., ”A Brokerage- Based Approach for Cloud Service Selection,” 2012 IEEE 5th International Conference on Cloud Computing (CLOUD) , pp.558-565, 24-29 June 2012.

Pawluk P., Simmons B., Smit, M. Litoiu M., Mankovski S.; ”Introducing STRATOS: A Cloud Broker Service,” Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on Cloud Computing (CLOUD) , vol., no., pp.891-898, 24-29 June 2012.

Ngan Le Duy, S. Tsai Flora, Keong Chan Chee, Kanagasabai Rajaraman; , ”Towards a Common Benchmark Framework for Cloud Brokers,” 2012 IEEE 18th International Conference on Parallel and Distributed Systems (ICPADS), vol., no., pp.750- 754, 17-19 Dec. 2012

Fabian Kuhn and Roger Wattenhofer, Constant-Time Distributed Domianting Set Approximation, Springer Journal for Distributed Computing, Volume 17, Number 4, May 2005.

S. Guha and S. Kuhler, Approximation Algorithm for Connected Dominating Sets, Algorithmica, Volume 20, Number 4, 1998.

SNAP, http://snap.stanford.edu/index.html, Last accessed: October 2012.

J. Leskovec, D. Huttenlocher, J. Kleinberg: Signed Networks in Social Media. 28th ACM Conference on Human Factors in Computing Systems (CHI), 2010.

M´ark Jelasity, Alberto Montresor, Gian Paolo Jesi and Spyros Voulgaris. "The Peersim Simulator". Downloadable at: http: //peersim.sf.net, 2010.

Chard, K. Bubendorfer, S. Caton and O. Rana, "Social Cloud Computing: A Vision for Socially Motivated Resource Sharing", IEEE Transactions on Services Computing, 2011. IEEE Computer Society Press.

Ioan Petri, Omer F. Rana, Yacine Rezgui & Gheorghe Cosmin Silaghi, "Trust Modelling and analysis in peer-to-peer Clouds", International Journal of Cloud Computing, No. 2/3, 2012. Inderscience.

Magdalena Punceva, Ivan Rodero, Manish Parashar, Omer F. Rana & Ioan Petri, "Incentivising Resource Sharing in Social Clouds". 21st IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2012, Toulouse, France, June 25-27, IEEE Computer Society Press, 2012, pp 185-190.

Living Planet Report 2012, http://www.wwf.org.uk/what_we_do/about_us/living_planet_report_2012/

Berry, L. L., & Maricle, K. E., Consumption without ownership: What it means for business. Management Review, 62(9), 44, 1973..

James Surowieski, Uber Alles, The New Yorker, Financial Page, 2013.

Grivas, S.G.; Kumar, T.U.; Wache, H., "Cloud Broker: Bringing Intelligence into the Cloud, *IEEE 3rd International Conference on*," *Cloud Computing (CLOUD)*, vol., no., pp.544,545, 5-10 July 2010

Nair, S.K.; Porwal, S.; Dimitrakos, T.; Ferrer, A.J.; Tordsson, J.; Sharif, T.; Sheridan, C.; Rajarajan, M.; Khan, A.U., "Towards Secure Cloud Bursting, Brokerage and Aggregation," *IEEE 8th European Conference on Web Services (ECOWS)*, vol., no., pp.189,196, 1-3 Dec. 2010

Belk, Russell W.., “Why Not Share Rather Than Own?,” *The ANNALS of the American Academy of Political and Social Science,* 611, 126 – 140, 2007.

Bendell, Jem and Anthony Kleanthous, “Deeper Luxury: Quality and Style When the World Matters,” 2007, ttp://www.wwf.org.uk/deeperluxury/_downloads/DeeperluxuryReport.pdf

Levenson, Eugenia, “The Fractional Life: With jewelry, yachts, and vineyards available by the slice, even the superrich are learning to share,”, 2007, http://money.cnn.com/magazines/fortune/fortune_archive/2007/03/05/8401282/index.htm

Moore, Amy and Michael Taylor, “Why buy when you can rent? A brief investigation of difference in acquisition mode based on duration,” *Applied Economics Letters*, pp. 1-3, 2008

Trendwatching.com, “Transumers: Consumers Driven by Experiences”, 2006, http://www.trendwatching.com/trends/transumers.htm

Winsper, Jeff, “The 6 P’s of Luxury Marketing: The Advanced Model for Measuring Consumer’s Buying Behavior for Luxury Brands,” , 2007, http://www.winsper.com/6Ps/

Zukin, Sharon and Jennifer Smith Maguire, “Consumers and Consumption,” *Annual Review of Sociology,* 30, 173 – 197, 2008.

European FP7 UNIFY Project – “Unifying Cloud and Carrier Networks”. Available at: http://www.fp7-unify.eu/. Last accessed: December 20, 2013.

Authors

Ioan Petri holds a PhD in 'Cybernetics and Statistics' from Babes-Bolyai University. He has worked in industry, as a software developer at Cybercom Plenware and then as a research assistant on several research projects. Starting with 2009, he collaborated with the School of Computer Science & Informatics, Cardiff University as an internship researcher in Distributed and Parallel Computing. Currently he is working in School of Engineering as an associate researcher in Computational Engineering. His research interests are cloud computing, peer-to-peer economics and distributed systems.

Omer F. Rana is a Professor of Performance Engineering in School of Computer Science & Informatics at Cardiff University and Deputy Director of the Welsh e-Science Centre. He holds a Ph.D. in ‘‘Neural Computing and Parallel Architectures’’ from Imperial College (University of London). He has worked in industry, as a software developer at Marshall BioTechnology Limited and then as an advisor to Grid Technology Partners. His research interests extend to three main areas within computer science: problem solving environments, high performance agent systems and novel algorithms for data analysis and management.

Magdalena Punceva is a Senior Scientists at the Institute for Computer and Communication Systems (ISIC), HE-Arc, HES-SO, Switzerland. She holds a PhD in peer-to-peer networks and distributed information systems from the Swiss Federal Institute of Technology in Lausanne (EPFL). She has worked as a Postdoctoral Researcher at CERN and spent a year as a Fulbright Visiting Reaserch Scholar at Rutgers University, New Jersey, US. Her research interests are in the area of large-scale networks and algorithms including social networks, cloud computing and distributed systems.

George Theodorakopoulos is a Lecturer at the School of Computer Science & Informatics, Cardiff University, since 2012. From 2007 to 2011, he was a Senior Researcher at the Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland. He is a coauthor (with John Baras) of the book Path Problems in Networks (Morgan & Claypool, 2010). He received his Ph.D. (2007) in electrical and computer engineering from the University of Maryland, College Park, MD, USA. His research interests are in privacy, security and trust in networks.

Professor Yacine Rezgui is a Professor in School of Engineering at Cardiff University and a BRE (Building Research Establishment) Chair in 'Building Systems and Informatics'. He is a qualified architect with an MSc (Diplôme d’Etudes Approfondies) in “Building Sciences” (obtained from Université Jussieu - Paris 6) and a PhD in Computer Science applied to the construction industry, obtained from ENPC (Ecole Nationale des Ponts et Chaussées).

He has then worked as a researcher for CSTB (Centre Scientifique et Technique du Bâtiment) and was involved in a number of national and EU research projects in the field of document engineering (DOCCIME), product modelling and Computer Integrated Construction (ATLAS).