The Peer-to-Peer Paradigm Applied to Hydrogen Energy Distribution

Michele Amoretti, *Member, IEEE*

Abstract: **The shift from centralized power plants to fully distributed generation, storage and provision of energy may be the greatest innovation of next decades. To succeed, it must be supported by the completion of the decarbonization process, using hydrogen as main energy carrier. The envisioned new energy economy (hydrogen economy) promises to eliminate global warming, pollution and dependency on oil reserves.**

Among the many issues that still have to be solved, in this paper we focus on the ICT aspect. Distributed energy production and sharing requires efficient distributed software architectures. To this purpose, the peer-to-peer paradigm is highly suitable for the realization of robust and scalable systems.

Index Terms**: decentralized energy distribution, hydrogen economy, peer-to-peer.**

I. INTRODUCTION

Decarbonization is the changing ratio of carbon to hydrogen atoms with each succeeding energy source [11]. Global primary energy use has evolved from reliance on traditional energy sources to being based on fossil fuels, first coal and steam then oil and natural gas, and more recently (but to a lesser extent) on nuclear. The journey of decarbonization is completed by hydrogen itself, which burns without producing carbon dioxide (CO_2).

Another great advantage of hydrogen is that it is much more easy to store it than any other *energy carrier*. For example, electricity can be stored in batteries, which are high energy density devices but release their potential slowly, and capacitors, which are able to release current very quickly but are low energy density devices. In 1992, the Fraunhofer Institute for Solar Energy Systems (ISE) in Germany created the first solar home, using hydrogen for long-term energy storage [15].

Finally, the great social benefit of adopting a hydrogen energy regime is that energy would be as cheap in one part of the world as another [3]. Since hydrogen can be produced from any primary source, it could improve the reliability of the energy supply and stabilize the energy market [4].

Hydrogen is the most abundant element in the universe (75% of the mass of the universe and 90% of its molecules), but rarely exists free-floating and alone, as do coal, oil, and natural gas. It only occurs in nature in combination with others elements, mainly with oxygen in water. Today, nearly half the hydrogen produced in the world is derived from natural gas via a *steam-reforming* process. The natural gas reacts with steam in a catalytic converter, in which hydrogen atoms are stripped away, leaving carbon dioxide as the by-product. Another issue of this solution is that increasing demand of electricity will likely cause the price of natural gas to rise [5]. *Electrolysis* is the other way to produce hydrogen, with the advantage that fossil fuels are not used in the process. Two electrodes, one positive (anode) and the other negative (cathode), are submerged in pure water that has been made more conductive by the addiction of an electrolyte. When direct current is applied, the hydrogen bubbles up at the cathode and oxygen at the anode. Electrolysis has not been widely used because the cost of the electricity used in the process makes it uncompetitive with the natural-gas steam-reforming process. But in fact it is the cost of generating electricity in large, centralized power plants that makes electrolysis so costly [2, 13]. However, in a *distributed generation* scenario, many issues must be reconsidered.

For example, in a distributed generation scenario, a feasible solution to generate the electricity that is used in the electrolysis process to split water into hydrogen and oxygen would be to use renewable forms of energy that are carbon-free, like photovoltaic (PV), wind, hydro, geothermal, and biomasses. Again, if natural-gas production peaks, a point could be reached where using renewable sources of energy to produce electricity for the electrolysis process could be cheaper [1]. In recent years, solar- and wind-power-based electrolysis systems have been set up in Germany, Italy, Spain, Switzerland, Finland, the United States, and even Saudi Arabia [14].

The most important point to be emphasized is that renewable resources converted into hydrogen-stored energy can be applied in concentrated forms whenever and wherever needed, and with zero CO_2 emissions. This is made possible by *fuel cells*, which convert the chemical energy of hydrogen (in general, of a fuel) that is fed into them to generate electricity. Fuel cells are expensive, because (like any new technology) their production has not yet reached the critical threshold where economies of scale kick in to significantly reduce the cost per manufactured unit [1, 2]. However, in the longer run, hydrogen-powered fuel cells will eventually assume dominance and become energy leader in the market of distributed generation.

To conclude this premise, we argue that the shift from the centralized power plants scheme to distributed generation may be the greatest innovation of next decades, with two main effects: lowering the cost of hydrogen production by electrolysis, and also providing incentives for the diffusion of fuel cells for the extraction of energy from stored hydrogen.

This argument is the main motivation for the hydrogen energy distribution framework we propose in this paper. We envision a software layer made of distributed nodes, interacting in a peer-to-peer fashion, and a physical layer made of energy producing and storage units. In the following we provide the detailed definition of the framework, together with a comprehensive classification of peer-to-peer overlay schemes. Then we describe the energy sharing process in systems based on our framework, showing how resource discovery in the virtual network is affected by the topology of the physical energy network. As an example, we introduce a rather simple unstructured overlay scheme, characterized by epidemic message propagation, and we evaluate it by means of simulation.

II. THE FRAMEWORK

We propose a *microgrid* design strategy, for which small-scale power supply networks provide power each one to a small community. The definition of the 'small community' may range from a typical housing estate, isolated rural communities, to mixed suburban environments, academic or public communities such as universities or schools, to commercial areas, industrial sites and trading estates, or municipal regions [6].

We envision microgrids in which passive energy users are becoming freelance energy producers [7], being connected to one another with the help of a distributed software application.

A study done for the U.S. Department of Energy [16] concluded that the biggest obstacle to creating such an interactive energy network is "the long-standing regulatory policies and incentives designed to support monopoly supply and average system costs for all ratepayers". In the two years since that report was issued, the situation has begun to change. A few power companies are already beginning to modify their mission, exploring a new role as bundler of energy services and coordinator of energy activity on distributed generation systems. In the new scheme of things, power companies would become *Distribution Network Operators (DNOs)*, assisting end users by connecting them with one another and helping them share their energy surplus profitably and efficiently. DNOs for distributed generation networks should be like ISPs for the Internet.

Silberman [7], Rifkin [1] and others compare the distributed generation model to the WWW, for their bottom-up development dynamics. More concretely, we envision systems that are characterized by two network layers:

- the *physical layer*, consisting of connected nodes, each one equipped with electrolyzers (for the production of hydrogen), hydrogen storage facilities, fuel cells (for the extraction of energy from hydrogen), sensors and a computing unit (a common PC); fuel cells form energy network, while computing units are connected through the Internet; connection and rewiring of such physical nodes requires human intervention; each links among two nodes has a transmission cost, decided by DNOs;
- the *virtual layer*, consisting of software nodes, each one hosted by the computing unit of a physical node, forming a highly dynamic overlay network; each software node collects sensor data to provide up-to-the-moment information on local energy conditions, supporting a distributed energy discovery process which allows all software nodes to find the most convenient (cheapest and nearest) energy producer.

Both the physical and the virtual network are based on the ***peer-to-peer paradigm***, which has recently emerged as a highly appealing solution for scalable and high-throughput resource sharing among decentralized computational entities [17]. A peer-to-peer system is a large set of collaborating programmed elements, which implement distributed algorithms to the purpose of sharing resources efficiently. Furthermore, it is a complex system, *i.e.* a system composed of interconnected parts that as a whole exhibit one or more properties (*e.g.* behavior) which are not easily inferred from the properties of the individual parts.

From a topological perspective, the physical network layer defined in our framework is a mesh that can be represented as an colored undirected graph, each node being physically connected (with bidirectional links associated with a cost) to *k* other nodes, with *k* being randomly distributed because of the unstructured network growth. We consider mesh topologies because they are more robust than traditional star topologies, in which a breakdown of the central hub renders the network inoperable, and also because our microgrids are made of peers rather than clients and servers.

On the other side, the virtual network layer can be represented in general as a directed graph, depending on the adopted overlay scheme whether knowledge links among neighbors are bidirectional or not. With respect to the physical network layer, virtual peers can be disconnected and reconnected many times, changing the network topology without affecting too much its functioning. In the following section we provide a classification and description of the most important peer-to-peer overlay schemes.

III. PEER-TO-PEER OVERLAY SCHEMES

The overlay scheme defines how peers are connected and how messages are propagated among nodes to share resources and information about them. Most first generation overlay schemes were based on flat unstructured models, for which all peers have equal responsibilities and are randomly connected. Information for locating resources was stored in centralized directories, or flooded among peers. Second generation schemes are based on hierarchical models, where some peers provide more functionalities than others. Third generation schemes constrain topologies and routing strategies in a logically structured fashion.

We classify overlay schemes with respect to three aspects: presence/absence of a functional hierarchy, distribution degree of information about shared resources, and relation of the latter with the topology of the overlay network. According to this, we propose the following taxonomy for peer-to-peer architectural models:

- Flat
    - Unstructured
        - Centralized
        - Decentralized
    - Structured
- Hierarchical

We point out that, in the hierarchical scheme, each layer is organized according to a flat model.

The Unstructured Centralized Model (UCM) is a first generation overlay scheme which was made popular by Napster [18]. The resulting architecture (illustrated in Fig. 1) is hybrid: community peers connect to one or more central directories to publish information about resources they offer for sharing. The central directory matches each resource request with the peer offering the best answer. The best peer could be the cheapest, fastest, or most available one, depending on user needs. The UCM approach has evident scalability limits, because more performing servers maintaining the central directory are required when the number of requests increases. Napster's experience showed that this model is quite efficient, while not secure and robust. BitTorrent [19] file sharing systems are also based on UCM, but central servers store only the essential information about trackers, *i.e.* agents which are responsible for helping downloaders find each other. OpenNap [21] extends the Napster protocol to allow sharing of any media type, and the ability to link servers together. Another UCM-based protocol is eDonkey [22].

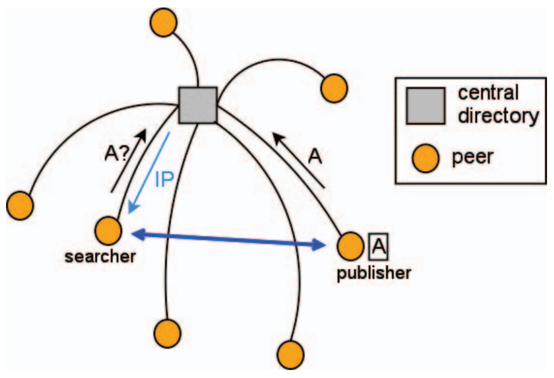

Fig.1 – Unstructured Centralized Model (UCM).

In the Unstructured Decentralized Model (UDM), illustrated in Fig. 2, information about resources is evenly distributed among peers. Each peer propagates requests to directly connected peers, according to some strategy. For example, by flooding the network with messages, a strategy adopted *e.g.* by Gnutella [20] that consumes a lot of network bandwidth, and hence does not prove to be very scalable, but it is efficient in limited communities such as company networks. To improve scalability, caching of recent research requests can be used.

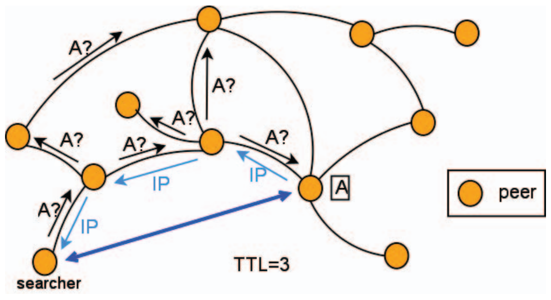

Fig.2 – Unstructured Decentralized Model (UDM).

P2P systems based on the Structured Model (SM) implement a Distributed Hash Table (DHT). Each peer has a randomly assigned ID and has a logically structured list of neighbours (as illustrated in Fig. 3).

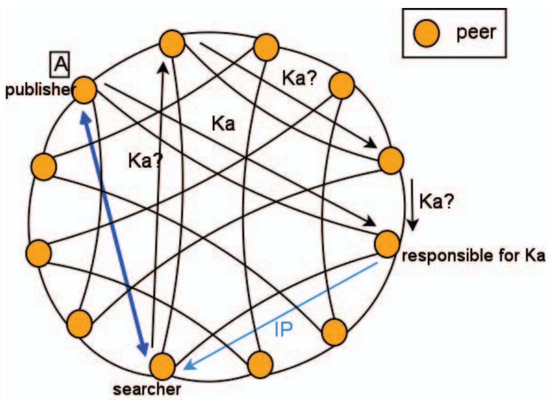

Fig.3 – Structured Model (SM).

When a resource is published (shared) in such systems, its description is hashed to obtain an ID. Each peer then routes the resource description towards the peer with the ID that is most similar to the resource description ID, according to a specific metric. This process is repeated until the nearest peer ID is the current peer's ID. When a peer requests a resource from the P2P system, the request goes to the peer with the ID most similar to the description ID. The process is repeated until a matching resource is found. Examples of SM-based networks are those defined by the Distributed K-ary Search (DKS) [26] family of protocols, and Kademlia [27], each one defining its specific metric to compute distances between IDs.

A rather different approach is the Hierarchical Model (HM), which uses a flow control algorithm for query propagation, searching for peers which have the highest probability of fulfilling the resource requirements (Fig. 4). The distributed control algorithm is usually implemented only by peers with higher bandwidth and process capacity, which are named super-peers or supernodes. This model has been adopted for example in FastTrack [23], Gnutella2 [24], Skype [25] and JXTA [31].

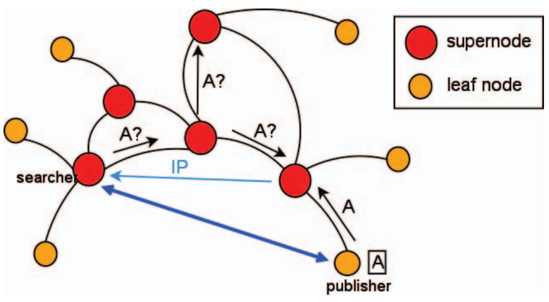

Fig.4 – Hierarchical Model (HM).

The SM may be merged with the HM, resulting in a hierarchical network with supernodes implementing a DHT [28].

IV. ENERGY SHARING PROCESS

We assume that each peer is a consumer but also a producer of energy. The load profile of each peer, representing its average power demand over time, is considered to be the superimposition of (1) an almost constant component, which is the energy produced by the peer itself and immediately consumed or stored using hydrogen; and (2) demand peaks, for which the energy locally stored by the peer may not be sufficient.

When a peer has a demand peak, it starts searching the virtual network for a peer which is able to provide the energy it needs at the lowest cost. In general, energy distribution is not direct from producer to consumer: other peers in the physical network layer mesh are involved in the transmission process. For this reason, the selection of the energy provider depends not only on the cost of the energy it sells but also on its distance. For simplicity, we assume that each energy request must be completely satisfied by a single peer, not by a "consortium" of peers (this much more complex case will be addressed in a future work).

Formally, our energy sharing strategy comprises four steps:

1. peer X requires an amount E_X of energy, described in terms of average power and duration ($E = P{\cdot}D$);
2. peer X searches the virtual network and finds n possible providers, forming a set $X^n = \{X_i, i=1,..,n\}$;
3. supposing the physical network layer topology is known, peer X computes the minimum distance between itself and each provider in X^n;
4. peer X selects peer X_k in X^n as provider, such that $l(X,X_k) = min_i\{l(X,X_i)\}, i=1,.., n$, where $l(X,Xi)$ is the *least-cost path* between X and Xi.

In graph theory, the least-cost path problem is the problem of finding a path between two vertices (or nodes) such that the sum of the weights of its constituent edges (links) is minimized. Note that if all edges in the graph have the same cost, the least-cost path is also the shortest path. One way to classify routing algorithms is according to whether they are *global* or *decentralized* [30]. Routing algorithms with global state information are often referred to as link-state (LS) algorithms. The most famous is *Dijkstra's algorithm* [8], which computes the least-cost path from one node (the source) to all other nodes in the network. Creating a priori knowledge of all possible $l(X,Y)$ least-cost paths makes step 4 of the energy sharing process very fast to complete. If the physical network topology changes, least-cost paths must be recalculated (which is an endurable disadvantage, being the network relatively small). To avoid this computational burden, decentralized routing algorithms may be adopted, in which the calculation of the least-cost path is carried out in an iterative, asynchronous and distributed manner [30]. Another advantage of this approach is its ability to adapt to link-cost changes and link failures.

V. EPIDEMIC MESSAGING

In this section we propose an example of system based on our framework, enabling efficient and scalable sharing of energy, and in general of consumable resources, *i.e.* resources that cannot be acquired (by replication) once discovered, but may only be directly used upon contracting with their hosts. In computer science, an example of consumable resource is disk space, which can be partitioned and allocated to requestors for the duration of a task, or in general for an arranged time.

The overlay scheme is UDM-based. Algorithm 1 illustrates the protocol executed by each virtual peer involved in a energy discovery process, which is based on epidemic propagation of queries [9]. Each query message generated by a peer is matched with local resources and eventually propagated to neighbors, with a time-to-live (TTL) which is the remaining number of hops before the query message itself expires (*i.e.* it is no longer propagated). Moreover, each peer caches received queries, in order to drop subsequent duplicates. In general, bio-inspired epidemic protocols have considerable benefits as they are robust against network failures, scalable and provide probabilistic reliability guarantees.

Algorithm 1 - Energy discovery protocol executed by each virtual peer (P = power, D = duration)

```
1: if (peer ≡ initiator) && (timeout expired) then
2:         select provider among discovered ones
3:         exit
4: endif
5: if (peer ≡ initiator) then
6:           define requested energy: $P$, $D$
7:           define query timeout
6: end if
7: if (¬propagation)
8:         if ($P \le P_{local}$) then
9:                   occupy ($P_{local}$ - $P$) for $D$
10:                  exit
11:        endif
12:        if (a > local A) then
13:                  occupy ($P_{local}$) for $D$
14:                  reset discovery: $P \leftarrow P - P_{local}$
15:        endif
16: endif
17: if (propagation) then
18:        if ($P \le P_{local}$) then
19:                  add peer to list of possible providers
20:                  exit
21:        endif
22: endif
23: if (TTL > 0) then
24:        propagate query with TTL ← TTL-1
25: endif
```

The resource discovery process is affected by the following parameters:

• f_k = fraction of neighbors targeted for query propagation

• TTL_{max} = max number of hops for messages

Traditional epidemic algorithms use fixed values for parameters, which are set in advance according to the results of some tuning process which should guarantee good performance with high probability. More sophisticated approaches envision random initialization of parameters, and periodic updates based on distributed evolutionary algorithms (we are currently working on the definition of a framework for such systems, which will be the argument of a future work).

VI. SIMULATION EXPERIMENTS

Creating working prototypes of peer-to-peer energy sharing systems based on the proposed framework may be very expensive, nowadays. Nevertheless, simulation can be used to gain insight to their functioning, and to optimize their performance by tuning epidemic messaging parameters and the energy resources that each participant should be equipped with [10].

To evaluate the strategies illustrated in sections IV and V, we used the Discrete Event Universal Simulator (DEUS) [12]. This tool provides a simple Java API for the implementation of nodes, events and processes, and a straightforward but powerful XML schema for configuring simulations.

We considered a energy sharing network with fixed population (5000 nodes), with static physical network layer topology. At the virtual network layer, we connected peers one after another, assuming that each new peer connects to 3 peers, randomly chosen among those which are already connected. Connections at the virtual layer are always bidirectional. Figure 5 illustrates an example of resulting node degree distribution, for a virtual network grown by randomly connecting each new peer to M=3 already connected peers.

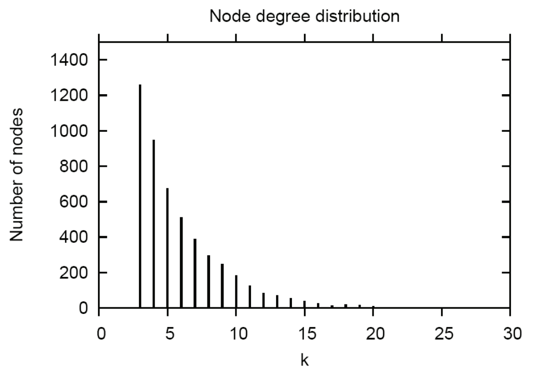

Fig.5 – Example of node degree distribution of a network with 5000 virtual peers, each one connecting with 3 already connected peers.

Table 1 provides a summary of energy requirements (in terms of Watts, *i.e.* capacity or demand at a point in time) for a number of different consumers, ordered by increasing demand [29].

Table 1	
Number of Watts	**What they will power**
1 W	1 Christmas tree light
100 W	1 standard light bulb
1 kW	2 refrigerators, 1 box of servers in a server farm
100 kW	25 homes or 1 fast food
1 MW	1 office building
100 MW	10 server farms
1 GW	100 factories or a city
10 GW	35 Nicaragua or New York city

Each simulated peer is characterized by either 5kW or 10kW of power which can be made available on-demand for a duration of 1.2h on average. We introduced a parameter C for which

• $f_k = C/(2*C_{mean})$ or C/ C_{mean}

• $TTL_{max} = C$

In general, each peer may have a random *C* value, but here we considered all peers set with the same *C* value ($\equiv C_{mean}$), thus f_k = ½ or 1, *i.e.* messages are propagated to half or all the neighbors, respectively. We measured, for *C* ranging from 1 to 5, and for different values of the initial number of connections, the evolution over time of the average QHR, considering peers which have sent at least one query (with timeout of 2 seconds per query).

We simulated about 14 h of the life of a network of peers, with random energy query distributed over time according to a Poisson process with fixed rate of one request each 20 sec. Each query is a request for a randomly generated amount of resource (no more than 10kW for a duration of 1.2h, *i.e.* a maximum energy consumption of 12kWh).

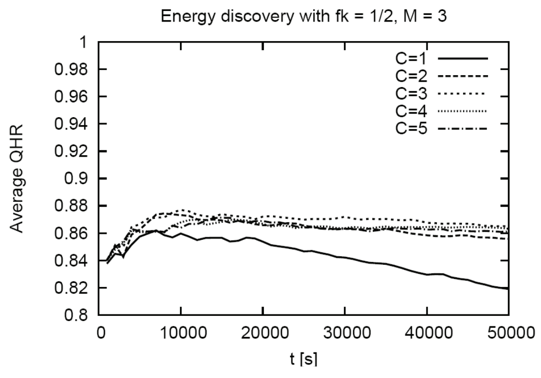

Fig.6 – Average Query Hit Ratio (QHR) in a 5000 nodes network, with f_k=1/2 and *M*=3. The *C* parameter influences the depth of the search.

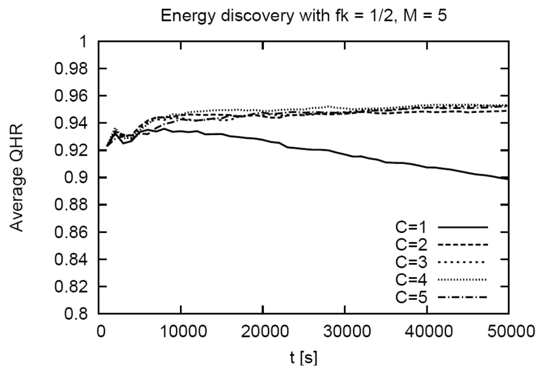

Fig.7 – Average Query Hit Ratio (QHR) in a 5000 nodes network, with f_k=1/2 and *M*=5. The *C* parameter influences the depth of the search.

Figures 6-9 shows the results of the simulations, in terms of average Query Hit Ratio (QHR), which is the fraction of query hits versus the number of sent queries (of course the average is computed considering only those nodes which have performed at least one discovery). Since we considered f_k = ½ or 1, and *M*=3 or 5 (where M is the number of random connections each virtual peer creates when joining the network), there are four different cases. For each case, we considered five values for *C*, ranging from 1 to 5.

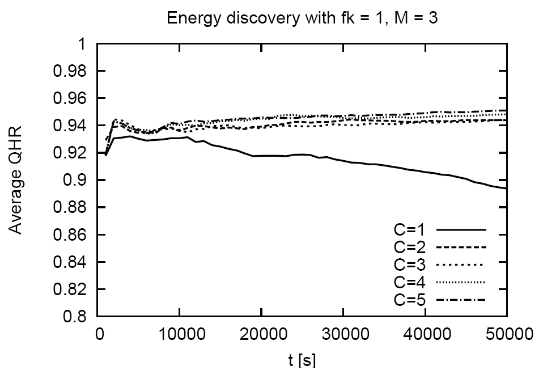

Fig.8 – Average Query Hit Ratio (QHR) in a 5000 nodes network, with f_k=1 and *M*=3. The *C* parameter influences the depth of the search.

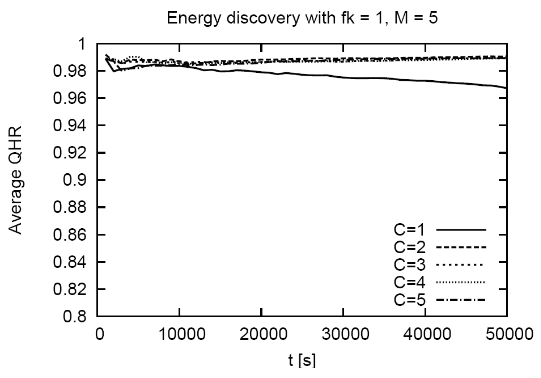

Fig.9 – Average Query Hit Ratio (QHR) in a 5000 nodes network, with f_k=1 and *M*=5. The *C* parameter influences the depth of the search.

As it was foreseeable, increasing the width (f_k) and depth (TTL) leads to best results (figure 9). Such kind of simulations allows to perform a first tuning of parameter *C*, and also to estimate the average quantity of energy that each peer must produce to satisfy the load configuration. For the one we considered, we see that resources shared by peers are sufficient (otherwise requests would not be satisfied even with f_k=1, *M*=5 and TTL=5).

VII. CONCLUSIONS

In this paper we have proposed a microgrid design strategy, based on two network layers (physical and virtual), for distributed energy generation supported by hydrogen storage. We have shown that the peer-to-peer paradigm can be applied at the virtual layer, allowing energy providers to share their resources.

We illustrated the general framework of the energy sharing process, and an example of unstructured overlay scheme characterized by epidemic messaging. We finally illustrated how simulation experiments are useful in the tuning phase of the design process of such systems.

Future work will mainly focus on the refinement of the proposed framework, with particular interest for overlay schemes based on adaptive evolution, and for third generation structured overlay schemes. With respect to traditional peer-to-peer applications, this specific context requires to carefully take into account the physical layer (which conveys energy from one node to another) in relation with the virtual layer (in which information is spread).

REFERENCES

[1]. Rifkin J. The Hydrogen Economy. Tarcher / Penguin. New York, 2003.

[2]. Bossel, U. Does a Hydrogen Economy Make Sense? Proceedings of the IEEE, 94 (10) (2006), p. 1826-1837.

[3]. Hoffman P. Tomorrow's Energy: Hydrogen, Fuel Cells, and the Prospects for a Cleaner Planet. The MIT Press. Cambridge, MA, 2001.

[4]. Royal Belgian Academy Council of Applied Science. Hydrogen as an energy carrier. 2006.

[5]. Chambers A. Distributed Generation: a Nontechnical Guide. PennWell. Tulsa, OK, 2001.

[6]. Abu-Sharkh S., Arnold R. J., Kohler J., Li R., Markvart T., Ross J. N., Steemers K., Wilson P., Yao R. Can microgrids make a major contribution to UK energy supply? Renewable & Sustainable Energy Reviews 10 (2006), p. 78-127.

[7]. Silberman S. The Energy Web. Wired 9 (7) (2001).

[8]. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C. Introduction to Algorithms (2nd edition). The MIT Press. Cambridge, MA, 2001.

[9]. P.T. Eugster P. T., Guerraoui R., Kermarrec A.-M., L. Massoulie L. From Epidemics to Distributed Computing, IEEE Computer 37 (5) (2004), p. 60-67.

[10]. Banks J., Carson J., Nelson B. L., Nicol D. Discrete-Event System Simulation (4th edition). Prentice-Hall (2004).

[11]. Nakicenovic N. Freeing energy from carbon. Daedalus 125 (3) (1996), pp. 95-115.

[12]. Amoretti M., Agosti M. Discrete Event Universal Simulator (DEUS). http://code.google.com/p/deus

[13]. Institute of Gas Technology. Survey of Hydrogen Production and Utilization Methods. 1975.

[14]. Dunn S. Hydrogen futures: towards a sustainable energy system. Int'l Journal of Hydrogen Energy 27 (3) (2002), p. 235-264.

[15]. Fraunhofer ISE. Solar House Freiburg, 1992. http://www.ise.fhg.de/about-us/history/1992-2000

[16]. Alderfer R. B., Starrs T. J., Eldridge M. M. Making Connections - Case Studies of Interconnection Barriers and their Impact on Distributed Power Projects. United States Department Of Energy Distributed Power Program, NREL Report (2000).

[17]. Androutsellis-Theotokis S., Spinellis D. A Survey on Peer-to-Peer Content Distribution Technologies. ACM Computing Surveys 26 (4) (2004), p. 335-371.

[18]. McCourt T., Burkart M. When creators, corporations and consumers collide: Napster and the development of on-line music. Media, Culture and Society 25 (3) (2003), p. 333-350.

[19]. Cohen B. BitTorrent. http://www.bittorrent.org

[20]. Saroiu S., Gummadi P. K., Gribble S. D. A measurement study of peer-to-peer file sharing systems. Multimedia Computing and Networking (2002).

[21]. D. R. Scholl. OpenNap homepage. http://opennap.sourceforge.net.

[22]. A. Klimkin. eDonkey protocol v0.6.2. http://kent.dl.sourceforge.net/pdonkey/eDonkey-protocol-0.6.2.html, April 2004.

[23]. N. Zennstrom, J. Friis, and J. Tallinn. FastTrack protocol. http://cvs.berlios.de/cgi-bin/viewcvs.cgi/gift-fasttrack/, September 2003.

[24]. M. Stokes. Gnutella 2 developer network wiki. http://g2.trillinux.org, September 2003.

[25]. Sharman Networks. Skype. http://www.skype.com

[26]. Onana Alima L., El-Ansary S., Brand P., and Haridi S. DKS(N,k,f) A family of Low-Communication, Scalable and Fault-tolerant Infrastructures for P2P applications. Proceedings of the 3rd Int.'l Workshop "Global and P2P Computing on Large Scale Distributed Systems", Tokyo, Japan, May 2003.

[27]. Maymounkov P., Mazieres D. Kademlia: a Peer-to-peer Information System Based on the XOR Metric. Proceedings of the 1st Int.'l Workshop on Peer-to-peer Systems, Cambridge, MA, USA, March 2002.

[28]. Han S., Park S. A dynamic layer management scheme for a superpeer ring with a loosely-consistent dht. Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, B.C., Canada, August 2007.

[29]. Edison Electric Institute. Electric Power Research Institute. CIA World Factbook 2000.

[30]. Kurose J. F., Ross K. W. Computer Networking (3rd edition). Addison Wesley, 2005.

[31]. Traversat B., Arora A., Abdelaziz M., Duigou M., Haywood C., Hugly J.-C., Pouyoul E., Yeager B. Project JXTA 2.0 Super-Peer Virtual Network. Project JXTA Technical Reports, 2003.

Michele Amoretti received the Dr. Eng. degree in Electronic Engineering in 2002, and the Ph.D. degree in Information Technologies in 2006 from the University of Parma, where currently he serves as a Research Assistant. His research activity focuses on distributed systems, with particular interest for the peer-to-peer paradigm and for self-organization issues. Application fields include Grid Computing, Service-Oriented Architectures, Ambient Intelligence and Networked Robotics. He is member of the IEEE since 2000.

.