An Approach to Decentralizing Search, Using Stigmergic Hyperlinks

Artur Marques[1], José Figueiredo[2]

[1] ESGTS (Management and Technology School of Santarém, Polytechnic Institute of Santarém)
ESGTS, IPS, Polytechnic Institute of Santarém
Complexo Andaluz, Apartado 295
2001-904 Santarém, PORTUGAL
am@arturmarques.com

[2] IST (Lisbon Institute of Technology, Technical University of Lisbon)
CEG-IST (IST Engineering and Management Research Center)
Av. Rovisco Pais, 1
1049-001 Lisboa, PORTUGAL
jdf@ist.utl.pt

Abstract. A stigmergic hyperlink, or “stigh”, is an object that looks and behaves like a regular HTML hyperlink, but runs at the server side. A system of stighs displays interesting emergent behaviors, of some complexity, but a stigh alone is very simple: it has a life attribute, only reinforced when users click it, and methods to provide meta-information about its destination. We reason that stigmergic hyperlinks could support a more decentralized approach to the Web search problem, particularly for addressing the “Deep Web”, which we consider all the WWW that is uncharted by search engines.

We discuss vertical and horizontal solutions for the “Deep Web” and present a specialized system that makes searchable the publications hiding at the biggest Portuguese digital magazines site. The index that the system builds, feeds the search related methods of a stigmergic hyperlink linking to that destination.

Our contributions are, to make the case for a broader “Deep Web” concept, that goes beyond databases hiding behind HTML forms; describe an approach that could decentralized Web search, based on stigmergic hyperlinks and a supporting business model; and to exemplify one specialized system that enables searching the biggest Portuguese digital magazines website.

Keywords: World Wide Web; Deep Web; business model; hypertext; hyperlink; stigmergy; stigh ; decentralization ; delegation.

1 Introduction

1.1 The Deep Web

The Deep Web is the part of the World Wide Web that the search engines ignore. While some contents explicitly request to be ignored, i.e. not to be crawled, others are served with technology that makes finding them hard by itself, like being stored in databases that only unlock when asked specific queries, posed by specific users.

In general, all dynamic content – having to be computed before being served – presents a challenge to Web crawlers, namely because most search engines’ original architectures, including Google’s [1], require a resource to have a dedicated URL in order to be indexed: this means that if two or more resources share the same address, and are delivered in result of different client side interactions, there might be no way to represent them in the system.

Web forms and interactive elements in general pose difficulties to crawlers. Assuming that the usage of interactivity facilitating client side (e.g. Javascript) and server side technologies has been increasing, then today's World Wide Web (WWW) is harder to search not only from the quantitative perspective of its growing volume of data, but also from the qualitative view that considers search barriers, intentional or not.

The disproportion between the many "deep" webs and the very few available search services makes it extremely difficult to think of an umbrella solution that can handle all the variety. Maybe there is an alternative to the centralized umbrella: a distributed network of specialized search agents, whose findings could feed search engines –stigmergic hyperlinks are objects that support that.

1.2 Stigmergic Hyperlinks

Stigmergy – from the Greek *stigma* (mark) and *ergon* (work) – means "the mark of work". The expression was introduced by the French biologist Pierre-Paul Grassé [2], who observed that termites, when building a nest, modify their local environment by aggregating mud balls, marked with pheromones. The termites don't have to communicate directly: the environment is their indirect and shared communication medium. Chances are that a mud ball will be put near other pheromone-marked balls, simply because there is a more invitational marking there. This (stigmergic) behavior, where the environment is a shared communication medium, that signals the work that has been done and simultaneously conditions future work, without the need of direct communication between worker agents, was an inspiration, is very common in nature and has many potential applications [3, 4].

We thought of web page visitors performing like ants or termites, leaving a digital pheromone on hyperlinks, whenever they clicked them – the idea was then executed and a new Social Web object [5], the stigmergic hyperlink or "stigh", was born.

Stighs look and behave very much like regular HTML hyperlinks [6], but they are server side objects with a life attribute that increases when they are clicked and decreases when they are not used, relatively to the other stighs in the same page. We had already found that a recommendation system behavior emerges from a set of stighs and we are now at the stage of enabling the prototype objects' meta-information methods and studying their potential.

1.3 Enabling vertical search

Vertical or specialized solutions for searching the Deep Web are usually limited to one site or to a small set of sites, while horizontal solutions are focused on particular web technologies (e.g. HTML forms querying databases), ignoring others.

Vertical solutions would be impracticable if they all had to be developed by the few global search engines on operation; however, if the task could be delegated on many entities, yet as few as one solution provider per site, searching could become much more decentralized.

Because stigmergic hyperlinks are simultaneously linking devices and search objects, they are adequate for supporting such a network of solution providers that blends with the current WWW structure and feel.

We would like that, in the end, all it takes for a stigh to be able to search its destination, deep or not, is the availability of a corresponding index: a structure that, at least, matches expressions with locations.

It is important to note that this decentralized approach gives a role to many new agents, but doesn't erase the role of today's leading search engines: people will always value the simplicity in having a central starting point for their searches.

2 The Deep Web

2.1 Concept

The “Deep Web” is the part of the World Wide Web (WWW) beyond the reach of the conventional search engines – Michael Bergman is credited with having coined the expression [7]. The original concept was about pages that search engines “can not see” because they “do not exist until they are created dynamically”, which is often translated to databases behind HTML forms [8].

We take a broader approach, considering “deep contents” all those that aren’t being indexed/found by the major search engines, due to whatever obstacles, ranging from architecture to any kind of browsing interfaces or filters, including forms in general and CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) in particular.

2.2 Scale

A 1997 study estimated hotbot.com as the most reaching search engine, covering 34% of the “indexable Web” [9], ahead of five alternative search services, of which lycos.com was the lowest rated , “seeing” less than 5% of the available contents.

Bergman expects the “Deep Web” to be 400 to 500 times larger than the surface Web [7]. More recently, in their estimate of how big the “Deep Web” is [10] Patel et al. tested 100,000 IPs (Internet Protocol Addresses), to identify those running http servers (we presume by doing a port 80 scan), then crawled to depth 10 down the found ones, counted their number of different database querying interfaces, then the corresponding number of web databases and, finally, reached a number of deep webs, which was extrapolated to the entire WWW, on the assumption that a total of around 2.2 billion IPs were valid and assignable. In raw numbers and using a 99% confidence interval the study, which was run during April 2004, expected 236000 to 377000 deep websites and a 32% deep Web coverage for both Google and Yahoo search engines.

There aren’t many details about the tools that were used, other than that wget [11] was the crawler and that it was modified to report the crawling depth. Their approach, assumedly, doesn’t account for web servers running on virtual hosts, because the IP is shared.

This Deep Web scale estimate should be an underestimation, at least because:

- The Web has been expanding since – IP version 6, the new version of the Internet Protocol, capable of addressing a much bigger network, is now a reality [12]. Moreover, as e-commerce grows and Management applies new value creation frameworks, like the 6 Cs (Commerce, Content, Communication, Computation, Community, Connectivity), dynamic content becomes a must [13].
- Virtual hosting (shared IP) is more affordable than dedicated hosting (dedicated IP), so ignoring it should equate to ignore a large set of web servers;
- Not only there is the issue of shared IP, there is also the issue of one single web server being able to serve many different web sites at the same time, deciding which one to deliver in function of the domain name requested, meaning that an even greater number of locations can’t be scanned by IP alone;

Other authors, as Patel et al. [10] explicitly state that forms for login, registration, polling and message posting were not considered query interfaces. Madhavan et al. also exclude forms requiring user information and consider only those they call “informational forms” [8].

However, a login/password form does query a database and guards access to protected content, so the “query interfaces” that matter is subjective, dependent on goals. For example, some authors [14] present a system that is successful in performing automated crawling and identity theft attacks against popular social networking sites in order to gain access to a large volume of personal user information – to achieve so, they had to automatically create forged profiles, interacting with login and registration forms, to harvest hidden relationships.

2.3 Solutions

Regarding how to search the Deep Web, there are specialized/vertical solutions that only work for a specific site or set of sites, and general/horizontal solutions, that intend to cover most, if not all, the webs. The directory at http://completeplanet.com claims to index 70,000 vertical search engines [15]. An omnibus general solution doesn't exist yet but some sub-problems are being addressed in a fully automatic fashion, like in Google's "surfacing" [8].

Two classes of vertical tools are identifiable: dedicated scripts that retrieve data from a particular site, and deep web crawlers, which are seen as a more scalable approach, on the assumption that they are like regular crawlers, until the time they face HTML forms – when that time arrives, their key problem is considered to be providing proper values to the form's elements [16] so that, when submitted, the form returns interesting results.

The brute force or naive attitude to explore forms, would be to input a whole dictionary to each open-ended element (where the user can type freely, like text boxes), but doing so would be stressful to the web server and inefficient because, probably, a much smaller number of carefully selected expressions can achieve the same returns. In [16] the authors present algorithms that generate representative keywords for sites, to be used in search forms. Their goal is to use such keywords in queries that siphon all or most of the available content.

Also having to compute keywords, but using a solution of their own, [8] describe one Deep Web "surfacing" system, that Google is already using: in their sense, "surfacing" is to pre-compute URLs suitable for inclusion in the Google index, corresponding to meaningful form submissions.

2.4 Broadening the approach

It often there has to be an already available HTML search form interface, ready to be submitted. So, in a way, the "Deep Web" that is being addressed is not that hidden and openly provides mechanisms to be searched, although most search engines can't yet automate their usage, to index the corresponding contents and make them more easily available to a wider community.

The "surfacing" approach has the great advantage of being compatible with the dominant search engines' architecture, that can only index resources that have a dedicated URL, but there are hidden contents to be found in many web sites that will remain uncharted using the retro-compatibility mode alone.

Forms can be "surfacing" resistant. We identify the "post" and the "action" problems.
The HTML form element supports an attribute named "method" that can only have one of two values: "get" or "post". When no method is specified, forms will default to "get". When using "get", forms unveil all the data that is being passed to the server, on submission, in an environment variable, also exposed at the browser's address bar.

For example, consider the following HTML search form, with a single user input element, which is a text box, named "query" having "love" written on it.

```
<form action="http://server/searchTool">
      <input type="text" name="query" value="love">
      <input type="submit">
</form>
```

When submitted, this form will build a so-called QUERY_STRING. In this case it will be
http://server/searchTool?query=love
For any other keyword *k*, just replace "love" by *k*, hence all possible usages of the form have different corresponding URLs – this is the ideal, for current search engines.

However, on "post" forms, there is no QUERY_STRING environment variable – the bytes are streamed to the server application. This means that all the possible usages of the form will expose the same single URL: http://server/searchTool

“Post” forms are used when the data to be sent is too big for an environment variable to handle, as in file uploads, and/or when there is sensitive data that shouldn’t be exposed at the browser’s address bar, like passwords, but can also do everything that “get” forms can. Search engines, such as Google, can not “surface” the “post” forms, because there aren’t different URLs to index.

As for the “action” challenge, it can be difficult to know what a form’s action is. For example, established client side development patterns call for client side validation; i.e. code that first runs some algorithms to decide if it is ok to go bother the server, instead of doing the call directly, so there might be no explicit URL for the action attribute – the form’s submission will be invoked programmatically, for example using Javascript.

Moreover, server side technologies, such as Microsoft’s ASP .NET, allow what appear to be forms with regular client side controls, such as text boxes and buttons, but aren’t – they are instead server side controls.

These techniques and technologies play a role in the deepening Web, as do many others. We think of the “Deep Web” not only as hidden data in Web databases with HTML interfaces, but in the much broader sense of all the deep contents that are being ignored.

2.5 Decentralizing (deep) search

Specialized solutions are limited to the site or sites they cover. They have the advantages of being focused and doing the job, but are seen as impractical from the perspective of search engines, because it is considered that each domain will require a mediator form and particular semantic mappings to its data sources, and that it is very difficult to identify site specific relevant queries. This is on the assumption that the hidden data is on databases behind HTML forms and that the search engines will have to do the effort alone, but what if the search engines could delegate the deep indexing on third parties? Given enough third party agents, the scale impracticality of vertical solutions for the Deep Web could fade.

The hyperlink is the most fundamental element in hypertext, so if it could some how extend its value from linking to also searching, then the way the WWW is approached and the search problem in particular could change. One of the functions that we envisioned for the stigmergic hyperlink [5] is the retrieval of meta-information about its destination, like an index – a structure that (at least) maps expressions with locations. By having access to an index of site *s*, stighs can directly search *s* and/or service that index to other agents who request it.

If stigmergic hyperlinks were a pervasive object and at least a few of those linking to deep websites would also build and provide the corresponding deep indexes, no matter if obtained by extremely narrow vertical solutions, then search engines would be able to abstract that problem and just focus on importing the third party data.

This scenario is a decentralization of the search with three types of economic agents:

- Solution providers via stighs, which take the challenge of indexing (deep) webs and/or service the indexes while linking to the destinations;
- Content providers, which publish (deep) content;
- Search engines, which provide the means to search the (deep) Web in general.

We can think of the following positives and negatives for each agent type:

Provider	**Positives**	**Negatives**
Solution	Traffic from search engines. Potential access to previously non existent revenue streams.	Having to pay to some content providers. Risk of litigation with some content providers.
Search	Access to deep content that otherwise would remain ignored. Immediate value creation for users. Abstraction from highly specialized vertical solutions, costly to develop and non scalable. Increased networking with other entities. Competition and redundancy of providers: as many as all the links to a destination can provide search functions over it.	If a system to reward vertical providers isn't put in place, there might not be enough incentive to reach a significant or good enough Deep Web coverage; but if such a system gets traction, it could have the perverse effect of pushing some contents to hiding, so that someone can profit from finding them.
Content	Traffic from both solution and search providers. Potentially increased revenue streams, because of potentially more users.	Increased content scrutiny that might unveil security flaws and raise data sensitivity issues. Risk of litigation.

2.6 Business model considerations

When the agents are in for the money, this should be the flow:

Content **ó** Solution **ç** Search

Search pays for the solution. The solution might pay to access the content and/or be payed to make the content more visible. The content provider might be payed for allowing access, but might also want to become more visible and pay for a solution.

Technology is one factor driving the WWW; business models and Marketing are others. Contents are locked and made part of the Deep Web for many reasons, like privacy concerns or because they generate income, as in subscription based business models, that demand a fee for access.

For those in the business of knowing the Web, like search engines, ad networks, or Marketing firms who want to better understand online behavior, the Deep Web can be a negative. There is less value to be delivered in a search when the search function omits a valuable (enough to be charged for) part of the search universe and can't even start to address its relevance. There is also less value in any Marketing function when it has poor or incomplete data about its desired communities. These reasons explain some of the flows sketched above.

3 One vertical example

We describe a specialized search system that enables searching over the biggest Portuguese digital magazines website (assineja.pt), which is for profit and charges subscription fees for all magazines. The site is a deep web, "hiding" behind .NET forms, never offering full magazine downloads, delivering content always as high quality JPG images, that don't support text searching or bookmarking. Still, the system managed to index all the text available in all the digital magazines until early March 2010. In total, 12 GB of data, corresponding to 19 titles and 321 magazines, were downloaded and made searchable.

We first identified and exploited a security flaw, in order to download every page of every magazine, then performed OCR (optical character recognition) on each page and built one PDF file per magazine. These stages are executed by modules, programmed in PHP with libcurl [17] and FPDF [18].

The security flaw was found after careful observation of the http traffic between the browser and the server, for free samples. Basically, every page has a direct URL that will be served upon request, without a previous check for a valid login. The URL has a rigid syntax that doesn't change, not even for the restricted magazines. Although the address isn't totally mechanic, i.e. it has parts that are random, those sequences are very short in length and can be brute force guessed.

Having understood the direct URL structure, all it took to download the pages was a couple of iterating algorithms. Among the unexpected hurdles was the fact that the web server in question never responds a "404"/"URL not found" for magazine pages that don't exist, so it isn't possible to control when a magazine's page is the last one, just by reading the http return code, because the site simply redirects to the home page. The solution was to check the first bytes before redirection: non existing pages are delivered as 1x1 pixels GIF files (GIF89a), while valid pages are JPGs.

The set of PDF files was then indexed by a keyword based search engine, written in C#, that supports HTML, text and PDF files, and can also perform on intranets. As it is, the search engine has file level granularity that only states in which files there are matches, not detailing beyond that, e.g. mentioning the page or the line.

The results are finally sorted by the number of hits of the search expression in the searched files, which is a common artificial measure that we intend to replace by "stigmergic natural relevance": if the search results were themselves stigmergic hyperlinks (and not plain links to the matching files), over time they should exhibit a recommendation system behavior, that is expected to capture their effective usage.

The index file was finally made available to a prototype stigh, with two search related methods:
exposeIndex(), which allows any requesting agent to download the index in use;
search(expression), by which the stigmergic hyperlink searches its active index for a given expression, using the search engine's algorithm. This search method must have a corresponding search form that automatically appears *onmouseover* ; i.e. when the mouse goes over the hyperlink. The search results appear in the same overlayed region, replacing the form.

3.1 Insights from the example

3.1.1 Stigmergic natural relevance

There is a reciprocal relationship between stigmergic hyperlinks and search functions: on one hand, search adds value to a stigmergic hyperlink, enabling it with meta-information about its destination; on the other hand, when search results are stigmergic, they can automatically evolve from an originally artificial ranking of findings to a "natural relevance" order, reflecting the preferences of the community that is using them – this is the stighs' emerging recommendation system behavior in action.

3.1.2 One pseudo-solution for the security flaw

The specialized indexing tool for the Portuguese magazines website made us aware of one security flaw that might be surprisingly common, because we also found it in international sites for digital publications (e.g. coverleaf.com). In those sites there are many differences, but the direct URL availability without a previous login check remains. This might be because of performance issues with BLOBs (Binary Large Objects) in databases [19].

One pseudo-solution for this vulnerability would be to increase the length of the random sequences used in the URLs. It is "pseudo" because it only works to a point: a large sequence does make the brute-force guessing not viable, but the root cause is still there.

4 Conclusions

Content can hide behind by many techniques in many technologies, and the “Deep Web” concept should embrace all the heterogeneity. Broadening the concept makes sense, but emphasizes how challenging the Web is for horizontal solutions.

Specialized or vertical solutions can’t usually scale beyond the website(s) they address and can’t be a one entity’s job. We hypothesize that with a proper reward system in place, using objects that can simultaneously link to destinations and search them, maybe a network of solution providers arises and contributes to decentralize search.

We propose both a reward system and an object for that matter – the stigmergic hyperlink.

We exemplified how one very specialized solution overcame its hurdles and articulated with a stigh.

References

1. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. (1997)

2. Grassé, P.P.: La reconstruction du nid et les coordinations interindividuelles chez bellicositermes natalensis et cubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. Insectes Sociaux **6** (1959)

3. Callon, M., Muniesa, F.: Les marchés économiques comme dispositifs collectifs de calcul. Réseaux **122** (2003) 189-233

4. Marco Dorigo, T.S.: From real to artificial ants. Ant colony optimization. The MIT Press (2004)

5. Marques, A., Figueiredo, J.: Stigmergic Hyperlink: a new social web object. IJISSC - International Journal of Information Systems and Social Change (2010)

6. Wardrip-Fruin, N.: What hypertext is. Conference on Hypertext and Hypermedia, Santa Cruz, CA, USA (2004) 126-127

7. Bergman, M.K.: White Paper: The Deep Web: Surfacing Hidden Value. JEP - The Journal of Electronic Publishing **7** (2001)

8. Madhavan, J., Ko, D., Kot, Ł., Ganapathy, V., Rasmussen, A., Halevy, A.: Google's Deep-Web Crawl. VLDB 2008. ACM, Auckland, New Zeland (2008)

9. Lawrence, S., Giles, C.L.: Searching the World Wide Web. Science **280** (1998) 98-100

10. He, B., Patel, M., Zhang, Z., Chang, K.C.-C.: Accessing the Deep Web. Communications of the ACM, Vol. 50. ACM.ORG (2007) 7

11. GNU.ORG, Nikšić, H., Cowan, M., Vol. 2010 GNU Wget is a free software package for retrieving files using HTTP, HTTPS and FTP, the most widely-used Internet protocols. It is a non-interactive commandline tool, so it may easily be called from scripts, cron jobs, terminals without X-Windows support, etc.

12. IETF, T.I.E.T.F.-. Vol. 2010 (2003) IPv6 is short for "Internet Protocol Version 6". IPv6 is the "next generation" protocol designed by the IETF to replace the current version Internet Protocol, IP Version 4 ("IPv4").

13. Krishnamurthy, S.: E-Commerce Management: Text and Cases. South-Western College Pub (2002)

14. Bilge, L., Strufe, T., Balzarotti, D., Kirda, E., Antipolis, S.: All Your Contacts Are Belong to Us: Automated Identity Theft Attacks on Social Networks. WWW 2009, Madrid, Spain (2009)

15. completeplanet.com - The Deep Web Directory. (2010) Discover over 70,000+ searchable databases and specialty search engines.

16. Barbosa, L., Freire, J.: Siphoning Hidden-Web Data through Keyword-Based Interfaces. Brazilian Symposium on Databases, Brasília, DF, Brazil (2004)

17. LibCurl. Vol. 2010 (2010) libcurl is a free and easy-to-use client-side URL transfer library, supporting FTP, FTPS, HTTP, HTTPS, SCP, SFTP, TFTP, TELNET, DICT, LDAP, LDAPS, FILE, IMAP, SMTP, POP3 and RTSP. libcurl supports SSL certificates, HTTP POST, HTTP PUT, FTP uploading, HTTP form based upload, proxies, cookies, user+password authentication (Basic, Digest, NTLM, Negotiate, Kerberos4), file transfer resume, http proxy tunneling and more!

18. FPDF. Vol. 2010 (2010) FPDF is a PHP class which allows to generate PDF files with pure PHP, that is to say without using the PDFlib library. F from FPDF stands for Free: you may use it for any kind of usage and modify it to suit your needs.

19. Shapiro, M., Miller, E.: Managing Databases with Binary Large Objects. 16th IEEE Symposium on Mass Storage Systems (1999)