

FindEngine™ white paper



hapax™

Version 2.3

First published in September 2001

Table of contents

Fact retrieval — a new paradigm in IR	4
Fact Retrieval as opposed to Document Retrieval	4
Semantic awareness through text analysis	5
FindEngine — system description	6
Overview	6
Query management	6
Matching	9
Organizing search results	10
Text analysis	10
Characteristics of methodology	12
Minimal domain dependency	12
Deterministic approach	13
Comparison to other question answering systems	13
The future of question answering	14
Conclusion	16

Fact retrieval: a new paradigm in IR

Fact Retrieval as opposed to Document Retrieval

This document is about retrieving information from unstructured text data. Today, the “IR community” generally defines an Information Retrieval (IR) system according to Lancaster’s definition from 1968:

'An information retrieval system does not inform (i.e. change the knowledge of) the user on the subject of his inquiry. It merely informs on the existence (or non-existence) and whereabouts of documents relating to his request.'

(F.W. Lancaster, Information Retrieval Systems: Characteristics, Testing and Evaluation, Wiley, New York, 1968)

Defining IR this way conceptually equates “information” with “document(s)”. Such a definition of “information” relieves an IR system of the ambition to enhance the user’s knowledge. It leaves it to the user to browse through the found documents. We will refer to these IR systems as “**Document Indexing and Retrieval Systems**” or **DIRS**.

An alternative definition of IR could be the process of transforming **data** (most often by extraction, summarization, or a combination thereof) into **information** that is useful for human decision making. Here the document is a meaningful entity only as a repository of data. Information consists of the reported facts in the text, truths, lies or conjectures about the state of the world, that can change the user’s knowledge about the world, and that decisions can be based upon. For convenience, this document will use the word “**facts**” to refer to such propositions or reported facts. Since the term IR system has been reserved by convention to the definition above, we will refer to the kind of system that can retrieve facts from unstructured text data as a “**Fact Retrieval System**”.

All conventional DIRS make use of the idea of a topic in order for the user to reach the data of interest for creating the information. A document’s content is supposed to be best described by the themes or topics in the document, by way of at least one of the following:

- Abstract representation - as with latent semantic indexing (LSI) and pattern recognition techniques;
- A topic’s “typical words” - as with a Boolean search tool; or
- Explicit literal representation - as with a traditional catalogue, found in many libraries and research databases.

Needless to say, modeling the information into more formal intermediary representation involves a loss of information, as in most any transformation of data into a more compact form. In this case, it is the **factual content** of the document that is lost.

With FindEngine™, Hapax is changing the existing search engine paradigm by abandoning index terms and document topics as the signifiers of content, in favor of the facts themselves. FindEngine™ does not operate on topics that are assigned to documents, or on the occurrence of words in documents, but on what the text actually says about its topic. FindEngine™ does not search for documents, but facts. By making the facts searchable, the FindEngine can perform IR in the complete fashion that enables information output to be directly used in decision making.

Facts can be found with FindEngine™ using natural language questions, natural language phrases, or natural language commands, depending on the user's specific interest. Facts are always delivered in the form of sentences directly quoted from the original text. This is important to the user, since it enables him or her to understand the context in which the proposition appears. If a broader context is needed, the user clicks a link that points to the original source document.

Semantic awareness through text analysis

The task of retrieving facts is accomplished by simulating aspects of how people understand text. Human understanding of text takes into account not only the word tokens themselves, but also how the words relate to each other in the text. On reading a sentence of text, a human quickly understands who does what to whom, where, when, how, and possibly why, i.e. the functional relationships between the entities that are described by the words used.

FindEngine™ takes advantage of the fact that human language has a structure that expresses these relationships. FindEngine can match different ways of stating the same thing, and can distinguish between similar sets of words that signify different facts or propositions. "Kennedy was shot by Oswald" is equal to "Oswald shot Kennedy", whereas "Kennedy shot Oswald" is a different proposition, since the roles of the entity 'Kennedy' and the entity 'Oswald' have been reversed.

In this sense, FindEngine can be said to have semantic awareness in a way that a conventional DIRS (that treats search terms as a "bag of words") does not.

FindEngine™ - system description

Overview

FindEngine is a fully automatic system for making unstructured text databases searchable for facts. Content is collected from internal or external sources e.g. the internet. Content may come in a variety of binary formats, so the first step is to translate the files into a common format that contains only the raw text of the documents, together with limited structural mark-up, such as paragraphs, headings, etc. and any meta-data that is supplied with the documents.

The text then goes through a text analysis process, which will be described in more depth in the following sections. The analyzed text, which is the raw text with additional mark-up for its linguistic structure, is stored in a proprietary database format.

When a user queries the system, the query is analyzed in a similar fashion. Depending on the nature of the query, matching rules are applied and **the system searches the database for facts that match the query**. The sentences containing facts that satisfy the matching criteria are extracted from the database and presented to the user, together with links to the original documents.

Query management

When people communicate a request for information can be phrased in several different ways. The query “Who invented sliced bread?” could be phrased “Do you know who invented sliced bread?”, “Sliced bread – I wonder who invented that!”, or “I would give anything to know who invented sliced bread”. It may be argued that the ultimate fact retrieval system should be able to respond to such requests for facts regardless of the way that the request is phrased.

In practice, however, a user knows that he or she is interacting with a computer system and has little need for conversational querying capabilities. Since automated fact retrieval is an unfamiliar concept to most users, it is especially important that the behavior of a fact retrieval system be predictable. The user should not have to guess on how to best formulate a query to get the results that he or she is looking for. The FindEngine™ therefore imposes some structural requirements on queries, supporting a set of query types that represent simple and straightforward ways of querying:

- **Natural language phrases:** A natural language phrase is very different from the quote-enclosed “fixed phrase” or “string search” of the traditional Boolean search engine. It is also different from a set of keywords. A phrase is a clause fragment that expresses entities or actions that could be of interest to the user. One important type of phrase is the noun phrase. A noun

phrase is a sequence of words that identify and qualify an entity i.e. “Tony Blair” or “high-fiber whole-meal bread loaf”. The noun phrase is a familiar concept in IR, and noun phrase extraction is a technique that has been used for a few years in conventional DIRSs to identify multiword units that may be used as indexing terms.

When FindEngine is queried for “rocket speedboat”, it will return sentences containing occurrences of e.g., “rocket speedboats”, as well as “rocket powered off-shore speedboats”. Since it understands the difference between looking for a boat and an engine, FindEngine will not, however, return sentences speaking of “speedboat rocket engines”. Furthermore, a combination of words in the text such as “...tuning the trim tabs is not rocket science - speedboats are generally...” would not result in a match, because there is no mention of a rocket speedboat here, even if the words “rocket” and “speedboat” are located close to each other in the text. Hapax’s recognition and matching of noun phrases are based on a thorough understanding of how natural language works in practice, and not on simple word co-occurrence or proximity.

In the same manner, FindEngine will intelligently handle any clause segment, for example a verb with an object (“disassemble crank case”) or a verb sequence (“must negotiate”). A phrase could also be as short as one single word (in which case, however, the word sought for should be very rich in information, in order for results to be specific enough to be useful to the user).

- **Natural language commands:** Natural language commands are used to search the text for certain relationships that may not be readily sought for by a natural language direct question. One is the “relate”-command, which is useful for finding semantic relations between two entities that can be directly deduced from text structure. The query “Relate Apple and Microsoft” will result in answers where Apple and Microsoft are semantically related, like “Apple has always released more minor versions of its OS than Microsoft” and “As it raked up profits in the late 1980s, Apple lost its marketing momentum, setting the stage for Microsoft”.
- **Natural language questions,** including asking for:
 - Subjects of specified actions (Who is selling hybrid cars?)
 - Objects of specified actions (What does Toyota sell?)
 - Statements by a specific person (What does Tony Blair say about European policy?)
 - Actions by a specific person (What does Tony Blair do?)
 - Actions affecting a specific person (What happens to Tony Blair?)
 - Descriptions or identities of a specific person (Who is Tony Blair?)

These examples above are a subset of the question answering capability of FindEngine™. Customer specific question types can be tailored to specialized information retrieval needs. The standard set of natural language question types is continually expanding towards the satisfactory support of all types of direct questions including *when* specific events have taken place (temporal relationships), and *where* and *how* question handling.

Which type of query is best to use depends on the particular interest of the user. We will use the following sentence as an illustrative example:

“Tony Blair is transforming the British governmental system, putting forward the most radical constitutional proposals since the 1920’s”

Any user who is interested in the facts contained in the sentence above may be coming from different angles or perspectives. A user may want to simply find everything about Tony Blair, or want to know what Tony Blair is doing, or maybe want to find everything that has been written about the British system of government. The following examples describe some of the versatility with which the FindEngine™ can be used to find specific information.

- **Phrase search and relations between entities:** Using the example sentence above, let’s assume that a user has a particular interest in the concept “radical proposals”. This user could just enter “radical proposals” into Hapax FindEngine, to obtain the sentence above as a search result. Note that “radical constitutional proposal” as it is stated in the original sentence, is a semantic variant of “radical proposal”. This is an example of *noun phrase search*, which will find facts relating to any given entity, regardless of the role that the entity plays within the fact. Pairs of entities may also be of interest to the user. Facts that describe a semantic relationship between two entities, e.g., “constitutional proposals” and “Tony Blair”, as in the example sentence, can be extracted by instructing FindEngine to “Relate Tony Blair and constitutional proposals”.
- **Question answering:** If a user is interested in, say, a definition of the “British governmental system”, the user would simply ask the system “What is the British governmental system?”. This question would not return the sentence above, simply because it does not answer the question. On the other hand, the question “What is done to the British governmental system” would return the example sentence. Also, asking the system “What does Tony Blair do?” would return the sentence, as well as “Who is transforming the system?”. These are examples of Hapax’s question answering (QA) capabilities.

These examples show that text carries much information above and beyond what is contained in its “bag of words”. Roles and relations between entities can be readily decided from the syntactic structure of the text, together with some understanding of what possibilities exist for different words. The example sentence contains propositions relating to several different entities, for example “the British governmental system”, “Tony Blair”, and “radical constitutional proposals”. It is evident that “the British governmental system” is the object of Mr. Blair’s act of transformation; “Tony Blair” himself is the party that is acting – the subject; “radical constitutional proposals” is the object of the Mr. Blair’s “putting forward”. FindEngine™ recognizes the meaning built in to these roles; and makes use of it when a user searches for facts. **FindEngine™ enables the user to get directly to the facts that he or she is interested in, eliminating the noise that is intrinsic to all systems that disregard the structure of human language as a carrier of information.**

Matching

As an example of how the FindEngine™ answers a natural language direct question, we will use the question: “**What does Tony Blair say about the Middle East?**”. The applicable FindEngine matching rule will be:

1. Find [Tony Blair]
2. As a subject
3. Of any verb of communication
4. Where [Middle East] occurs in what is communicated.

In any typical news database, hundreds of instances of “Tony Blair” may be found. Below are 7 sample sentences containing “Tony Blair”:

- A. Following John Smith's death, Margaret Beckett ran both for leader and deputy leader but cruelly won neither role, having to make way for **Tony Blair** and John Prescott.
- B. **Tony Blair** is transforming the British governmental system, putting forward the most radical constitutional proposals since the 1920s.
- C. Peace can be achieved in the Middle East only if the parties show sufficient goodwill, **British Prime Minister Tony Blair** said Saturday.
- D. Shortly before her death, Princess Diana accepted a long-sought offer to serve as a special ambassador for Great Britain, **Prime Minister Tony Blair** said Sunday, promising that her memory and good works would endure.
- E. At the press conference, **British Prime Minister Tony Blair** expressed that Japan should follow the example of his country's "big bang" and boldly open up its financial markets.
- F. **British Prime Minister Tony Blair** may play a more direct role in the Northern Ireland peace talks as the April 9 deadline for agreement nears, aides said Tuesday.
- G. Britain's foreign secretary on Sunday denied reports that **Prime Minister Tony Blair** intervened with the Italian government on behalf of media tycoon Rupert Murdoch.

Applying criterion 2 to this set of candidate answers will exclude sentence A, where Tony Blair is not found in the subject role. Criterion 3 will rule out sentences B, F and G, where the verbs connected to Tony Blair as a subject are “transform”, “play” and “intervene”. Now sentences C, D, and E remain, with Tony Blair “saying” or “expressing” something. Finally, by applying criterion 4, we see that only sentence C qualifies as an answer to the question and would, therefore, be the only sentence presented by FindEngine™.

Organizing search results

In the case of traditional document retrieval, relevance is considered to be a matter of degree. In most cases, document based IR systems base their ranking of search results either by estimating degrees of similarity between the query pseudo-document and the individual response documents, or by calculating the relative information weight of the search terms in the response documents. With FindEngine's fact retrieval, much more useful organizing can be accomplished. The default algorithm for organizing search results is based on a set of discrete criteria that take word order, verb tenses, and inflections of nouns into account. These criteria support grouping of results in terms of shared properties, as well as ordering within these groups according to degree of similarity to the question. Opportunities for further filtering, categorization, and other manipulation of search results are achieved by utilizing the power of the FindEngine's linguistic mark-up in post-processing of search results. The system can also be configured to order responses based on document meta-data, e.g., date of publication of the document in which the response is found.

Text analysis

The examples above demonstrate that the FindEngine's ability to answer questions, and to perform fact extraction from query phrases or commands, rests on accurate identification of the different entities, actions, and circumstances in the text, as well as the functional relations between them. The part of the FindEngine™ system that makes this possible is the text analysis module. The text goes through 6 steps of analysis:

- *Text structure assignment* - in which the binary document is converted to normalized text;
- *Tokenization* - which serves to identify individual words, and to normalize misspellings and other orthographic problems. Character strings such as *I'm* and *doesn't* are recognized as *I_ 'm* and *does_ n't*, abbreviations (p. m., p.m., p m) are recognized as single tokens, as are numerals (150 000, 150000, 150,000). Sentence delimiters are also identified;
- *Morphological analysis* - in which each word token is considered individually, determines the full set of possible analyses for the word. For example, morphological analysis of the word token *drove* results in:
 - drove = past tense of the verb drive
 - drove = singular of the noun drove
- *Part-of-speech tagging* - in which one of the possible analyses is finally determined, taking the context of the word token into account;
- *Shallow parsing* - which recognizes text units such as clauses and phrases (noun phrases, temporal or geographical prepositional phrases, etc.);

- *Functional parsing* – which serves to identify the functional roles of entities and other elements in the text. Functional parsing is the process that finally sorts the text elements into the categories of “Who does what to whom, where, when, how and why?”.

The text analysis process transforms the original text into text with linguistic mark-up, denoting the result of the analysis. This information enhancement makes it possible to perform the previously described matching across the entire text database.

Several patents have been granted (with several more pending) for methods pertaining to the different steps of the text analysis process. There are other commercially available text analysis software packages for OEM application that perform the full sequence of analysis, as well as parts of it. Dominating search engine providers use such linguistic software packages. These toolboxes are sold with linguistic resources, such as lexicons, and can perform tokenization, morphological analysis (for normalization of index terms) and noun phrase extraction (for construction of complex index terms).

The text analysis process used by Hapax FindEngine is designed specifically for the purpose of fact finding, and is superior to other commercially available systems in the way that shallow parsing and functional parsing is combined. Most text analysis tools deteriorate in performance as sentences grow longer, or as vocabulary gets unfamiliar. Hapax technology deals with these problems in more effective ways than other technology, as will be described in the following sections. This is essential, since the quality of fact retrieval builds on the quality of the text analysis. Each step of text analysis builds on the preceding step. Every step of analysis is optimally tuned to serve its purpose as building the foundation for fact retrieval.

Characteristics of methodology

Minimal domain dependency

One of the points of criticism conventionally directed against employment of linguistic technology in information retrieval is its domain dependency. This notion is valid for the currently most popular linguistic software toolboxes, but Hapax technology is built with domain independency as an explicit design parameter.

People do not always conform to the strict rules of spelling and grammar they were taught in school. Sometimes words are inflected in the “wrong” way, words are misspelled, and sometimes people are creative enough to, for example, derive a completely new verb from a noun. The schoolmaster’s view on these anomalies would be that they should be pointed out, so that they can be corrected. Such normative grammar is the methodological foundation for most commercial language technology. This is useful when designing a spell-check program or a grammar module for word processing software, but too inflexible for the efficient analysis of unrestricted text.

The analysis methods that Hapax uses are **descriptive**. They are originally designed to pragmatically deal with text that can be understood by people, but may or may not adhere to strict rules. They are also designed to deal with unknown words. When someone reads a technical text on a subject matter that is new to the reader, he or she will encounter new words – words that are “out of vocabulary”. Even though the reader does not know the exact meaning of the word, or its definition, he or she will usually be able to say that this word is, say, a verb, and not an adverb or a proper name. The reader will also be able to guess, with a fair degree of accuracy, how to inflect the new verb or even how a new noun could be formed from the new verb. The methods that Hapax uses mimic this aspect of human language understanding, and therefore minimize domain dependency. Much of the secret lies in **Hapax’s one level morphological analysis**, which is far better than competitors’ two-level methods at handling unknown words, because:

- It has more elaborate algorithms to intelligently handle morphemes as the building blocks of words. This way, important aspects of the language’s innate capacity to create new word forms is effectively modeled, which results in recognition of words that are not in the lexicon; and
- It enables self-extending lexicons based on new analyzed texts. Since the lexicons are derived from actual texts, they can hold frequency information about the number of times each analysis of a word form is observed in a text collection.

With FindEngine’s widely superior lexical coverage, and its intelligent guessing mechanisms, the system can effectively handle text pertaining to any knowledge domain.

Deterministic approach

Much linguistic effort has been spent on exploring the semantic ambiguities of natural language. Because of these ambiguities, natural language has been considered a sub-optimal way of representing knowledge, and much linguistic research has been devoted to finding alternative ways of representing knowledge, using artificial symbol systems that are strict, unambiguous, and seem to lend themselves better to for logic operations. At Hapax, we recognize the ambiguity of human language, but firmly believe that human language, written as well as spoken, is understandable. Although examples that demonstrate ambiguity can be easily found, most of our written and spoken communication is perfectly understandable, and Hapax does its best to capture the meaning of text at the surface level, much in the same manner that a human being quickly disregards less plausible possible interpretations of a piece of communication.

Normally, text analysis tools output several possible analyses for a given input sentence, because of the local structural ambiguities in a sentence, which often result in combinatorial explosions of ambiguities. Our text analysis is deterministic, in the sense that ambiguities are eliminated, and only one single analysis is finally chosen for each element of text. The secret lies in the FindEngine's proprietary techniques for pruning the growing 'tree' of ambiguity at the right moment – to avoid combinatorial explosion and, at the same time, keep plausible alternatives just long enough not to miss out on the correct analysis.

Comparison to other question answering systems

Since 1999, The Text Retrieval Conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and the Defense Advanced Research Projects Agency (DARPA), has hosted a yearly competition between question answering systems - the so-called QA-track. The systems participating in the QA-track have demonstrated some ability to locate text snippets of 50 or 250 bytes containing answers to specific questions. The answers were known to exist in the text. In 1999, even the best performing systems failed to located answers to 27% of the questions for the 50 byte task, and 23% of the answers for the 250 byte task. Official results have not yet been published for 2000, but participants' reports indicate marginal improvements on these figures. These results show that fact finding is not a trivial task.

All of the described participating systems that make use of text analysis apply this text analysis only as a second stage, in post-processing the output of a traditional word based document retrieval system. As so much information is lost in the first stage, this may explain some of the failure to find answers. The FindEngine™ is different, in that the full text is analyzed at indexing - so that the entire text database is ready to be searched utilizing the full linguistic capabilities. This up-front indexing feature of the FindEngine™ also makes it viable in practical usage. The FindEngine™ is designed for response times less than 5 seconds for more than 95% of the queries. In practical usage, average response time is below 0.5 seconds. Though the TREC QA-systems are not evaluated on response times, one of the top-performing systems reported an average processing time of 6 seconds, with processing times ranging

from 1 up to 540 seconds. Effective response times are clearly essential for useful commercial application.

Examining the competing systems a little closer, it becomes evident that they all make use of some form of multi-layered model of information, where the information is supposed to be useful only in some sublimated form. FindEngine™ adheres to a different principal: whatever can be done on the surface level, should be done there. Another important difference is that the FindEngine™ delivers facts as sentences, i.e. units of text that has meaning to the user, not as fixed length text snippets, in which the answer can be found.

The future of question answering

Hapax believes that fact extraction will form the basis of tomorrow's question answering systems. This seems to contrast with a commonly held view with the community associated with TREC, DARPA's TIDES (Translingual Information Detection Extraction and Summarization) program and ARDA's Advanced Question & Answering Program. In short, the argument against the usefulness of pure extractive technology is that natural language does not lend itself to effective reasoning, so knowledge must be formally represented. It is assumed that extractive technologies must be combined with extensive ontologies, i.e. formal catalogues of the objects of the world, their relations and attributes¹. This is in line with the AI tradition of the 60's and 70's, as well as with the developments in object oriented modeling made during the 80's and 90's. Hapax believes these convictions to be short sighted for at two main reasons:

1. The power of "pure fact extraction" is underestimated. Vast resources are already being spent, by Government agencies and others, on ontology building for the purpose of question answering, before the extractive path has been fully tried. In a sufficiently large text database, chances are high that the answer is there, spelt out clearly. The cost with which a system such as the FindEngine™ can make such a text database into a repository of searchable facts is negligible compared to the significant cost involved in building up ontologies or "knowledge bases" formally modeling entire systems of world objects.
2. The advantages of direct reasoning, i.e., performing reasoning operations directly on facts that have been extracted from text, is disregarded. Natural language can be used for reasoning. In fact, it is the method for reasoning that we are most familiar with, and it need not be any less accurate or precise than formal reasoning. By applying reasoning directly on the text itself, information is not lost as when the text is transformed into formal representation. Also, the reasoning can easily be verified by directly referring back to the original facts as they are written in text.

¹ The difference between these world object ontologies and the lexical resources of semantic classification, such as synonym and hyponym lexicons, should be noted. The former describe the objects in the world, and their relations and attributes, the latter only deal with describing the words. The former tend to need continuous updating as the world changes, whereas the latter refer to the system of symbols, which - although language is ever evolving - tends to more stable.

With the enhancement of direct reasoning, fact extraction is the straightest way forward. Hapax is currently devoting significant effort to incorporate more inference and other reasoning capability into its technology. Although ontologies may be very useful for particular applications in restricted domains, intelligent handling of natural language can take us a long way, and will be economically viable also in application areas other than the least cost sensitive. In the areas where the use of world object ontologies does make sense from a cost/benefit point of view, fact retrieval is an excellent method to help build them.

Conclusion

This document has described some of the aspects of Hapax technology that make using FindEngine™ more efficient from a user perspective than systems based on document retrieval.

With conventional IR, if you are lucky to phrase your search query in the optimal way, you would get references to documents that have been deemed “similar” - either because the same words appear in them or because they have been catalogued with similar topics by a mathematical method. These referenced documents are likely to be more or less relevant to your specific information need. With fact-finding, however, whether a specific fact answers a question or not is not a matter of degree. A statement is either an answer to a given question or not. To get to such information refinement, and to deliver facts instead of document references, the syntactical information in the text must be taken into account – something conventional techniques, regardless of mathematical sophistication, simply cannot do.

Already in 1990, Sembok and Rijsbergen noted that:

'The keyword approach with statistical techniques has reached its theoretical limit and further attempts for improvement are considered a waste of time.'

(T. Sembok and C. Van Rijsbergen, SILOL: A simple logical-linguistic retrieval system, Information Processing and Management, 26(1): 111-134, 1990)

This statement may have been premature, since the developments in the keyword approach has enhanced the performance of traditional IR tools over the course of the last decade. There is, however, a theoretical limit to how effective such tools can be, and today's commercially available systems seem to be approaching this limit. Hapax is proposing a different and more effective solution – a precise yet flexible and fast way to find specific information in unstructured text databases.

Today, researchers and so-called information specialists spend a significant amount of time scanning articles and reports in order to keep themselves updated on the latest developments in their field of interest. Machine-enhanced fact retrieval systems Like FindEngine™ is one of the technological advancements that will change the future of such knowledge work, be it in the academic environment, corporation or government agencies. By directly extracting the required facts, time will be freed up for more focused exploratory work, oriented towards specific tasks. Knowledge workers will be able to span broader knowledge domains, and inter-disciplinary research will be more efficient.

The core economic value of these prospects can hardly be over estimated. Machine-enhanced fact extraction will have significant impact on how knowledge and intelligence is organized – and, in today's modern, connected and information rich society, fact based IR systems will prove central to the quality of life for all of us.



Hapax Limited
Caparo House
103 Baker Street
London W1U 6LN
UK

T: +44 (0)20 7317 0151

info@hapax.com

www.hapax.com