Product Matching DSS

Concept, Opportunities & Design

Draft: © 2 October 2004

Tasks to completion:

1. Discuss visualization and how Serendipity ideas can be integrated into the design.

   Probably will add a short chapter on visualization. Meeting with Cristy on Tuesday October 5.

2. Incorporate James Thompson's material into the "Present Practice" chapter, chapter 2.

   Drafted. Waiting for comments from Jimmy.

3. Edit the material in Part II.

4. Complete the "Information Retrieval" chapter (it will be brief).

5. Complete the "Information Extraction" chapter

6. Complete the "Activities and Schedule" chapter by adding cost and level of effort estimates.

7. Finish editing the appendix, "A Note on Automated Support for Product Application Discovery"

   Good news. This paper has just been accepted a WITS, a strongly refereed conference.

8. Complete the executive summary.

   This is the last thing to be done.

Steven O. Kimbrough, 103 Bentley Avenue, Bala Cynwyd, PA 19004–2805. Tel: (215) 898-5133 or 610-667-8894. Fax: (215) 898-3664. Email: kimbrough@wharton.upenn.edu. Web:
`http://opim-sun.wharton.upenn.edu/~sok/`.

Note: This document has been compiled in draft mode.

`$Id: master-dupontdesign.tex,v 1.18 2004/10/02 21:45:53 sok Exp $`

Contents

List of Figures

List of Tables

Executive Summary

- Product Matching (PM) problems

  1. Product Finding
  2. Product Placing
  3. Product Hybridization

  Potentially huge opportunity. Seek to "change the game"; speed up discovery of marketable uses of IP.

- Philosophy

  1. Software for extracting information, focusing on text.
  2. DSS. No magic bullets. Aim is to build software that supports human analysts. Speed, thoroughness, focus.
  3. Comprehensiveness vs exhaustiveness. We aim at the former.
  4. Intensive processing possible. Need not stick with fast IR techniques. Batch indexing runs of several days are workable if the value they extract is good.

- Heuristics. Key concept.

  Identify demonstrably profitable heuristics and provide demonstrably effective software to support analysts in reasoning with the heuristics.

  Heuristics must be linked to, and justified by, process models for Product Matching.

- Purpose of this document: articulation and justification of a PM DSS concept; preliminary design sufficient to justify complete design and implementation; plan and schedule for completion of a fieldable PM DSS.

Key issues addressed by this document:

1. The promise and purpose of the envisioned PM DSS.

   Short answer: To support PM studies by gaining information faster, reducing labor, and improving quality. And thereby "changing the game" of commercializing IP. Addressed in detail in [list chapters].

2. High-level design of the envisioned PM DSS.

   Short answer: use cases. Addressed in detail in [list chapters].

3. Algorithmic basis.

   How is it possible to provide the computational support called for in the use cases?

   Short answer: The DuPont-Sizatola team has originated and assembled key concepts that meet this need. Addressed in detail in [list chapters].

4. Implementation plan.

   What are the steps needed to produce an operational and validated PM DSS?

   Addressed in detail in [list chapter].

Part I

Product Matching

Chapter 1

Product Matching Concepts

1.1 Product Matching Problems

PM, or product matching, problems

Every use of every product requires that someone solves a product matching (PM) problem by recognizing a fit between the product and its use. If essence of lemon is to appear in bottled water, someone first has to see that essence of lemon can be added to bottled water and that doing so is potentially valuable. Follow-up is of course necessary. Is it safe? Do customers want it? Will it be profitable? What will competitors do in response? These and other questions have to be answered favorably if the use of a product in a particular application is to be a commercial success. But there is often little point in answering these questions unless and until someone has made an interesting match between a product and a use for the product.

Product matching problems are pervasive and fundamental. They are also non-trivial. New, previously unconceived uses for old products regularly appear. If the product matching problem were straightforward firms would never be puzzled about how to productize their patents. In fact, matching products to uses is generally recognized as daunting and problematic. There is even a small industry of consultants specializing in helping firms to identify promising uses for the products they own—typically patents or other forms of intellectual property.

No one maintains that the state-of-the-art for product matching is fully satisfactory. The methods available rely almost entirely on human expertise and creativity, aided only at the margins by supporting technology. The methods are consequently expensive and not amenable to significant improvement over time. What can be done?

Human expertise and creativity, of course, will not be replaced by technology any time soon. But can human skills be augmented, leveraged, enhanced by innovative uses of computation applied to product matching problems? The aim of this paper is to sketch the case that they can.

1.2 Distinctions

If there is to be computational support for PM problems, algorithms (computational procedures) and data will need to be identified such that when the algorithms are executed on the data, PM-useful results regularly occur. What are the promising algorithms and where are data on which they are to operate? Some analysis and frameworking of PM problems will be useful in approaching these questions. If we are to support a process, we need a workable model of that process. If we are to find helpful algorithms, we need to identify what they should do.

PP, or product placing, problems

Product matching problems may conveniently be distinguished into three types. First, *product placing* problems arise when new uses are sought for a given product. Typically, the product in question is potentially useful as a component or ingredient in other, end-user products. How can promising applications be identified? Think of duck (aka: duct) tape, or Teflon, or fiber glass, or various kinds of plastics, or glues, and their many uses. Have all commercially attractive applications been discovered? Likely not. Could they be used in shoes, or ships, or sealing wax, or any of the thousands of end-user products now on the market? Note the considerable creativity and knowledge required to find the many existing and familiar uses of these component products.

PF, or product finding, problems

Second, *product finding* problems are the obverse of PP problems. PF problems occur when a firm considers how to improve a given end-user product. What can be done with shoes, or ships, or sealing wax to make them more valuable commercially?

PC, or product composition, problems

In PP and PF problems we innovate by modifying existing end-user products. In the third kind of PM problem, *product composition* we seek to create a more or less entirely new end-user product. We might, for example, combine pens with radios to create an FM radio pen as an end-user product. (Westminster, Inc., Atlanta, GA, sells one.) The PC-PP and PC-PF distinctions are, of course, not absolute and are matters of degree. Still, it will be useful to keep the distinctions in mind. PP and PF will be labels for incremental, relatively modest innovations. We reserve the PC label for comparatively major departures.

Gastronomy, the art of food preparation, offers good illustrations of these distinctions. Innovators in the kitchen, especially leading chefs, are in the business of solving PM problems. They are variously described as being in the "innovative cooking" or "creative cuisine" business. "What new and innovative dish can I create today?" is a PM question. More specificity is required if the question is to lead to progress. The suggestion at hand is that three more or less distinct follow-on questions constitute a step in this direction:

1. PP: "To what new use can I put this ingredient?" Perhaps there is an excess of strawberries on hand. What can be done with them? Put them on pizza? Bake them with a roast?

   Think of the product placing problem as the What-do-we-do-with-all-these-strawberries question. Or, given an ingredient, i, find one or more dishes, d, that can host it.

2. PF: "Pizza tonight. How can I make it interesting?" Drop the tomato sauce and use a thickened mash of strawberries?

Think of the product finding problem as the What-can-we-put-on-the-pizza question. Or, given a dish, d, find one or more ingredients, i, that go with it.

3. PC: "Any new dishes I can create with what's in my kitchen?" *The New York Times Magazine* of August 10, 2003, features descriptions of exciting culinary innovations from Spain ("A laboratory of Taste" by Arthur Lubow, pp. 38ff.). A photo appears on page 40 of "An Adrià creation: Grilled Melon With Ice Plant and Pink Grapefruit."

   Think of the product composition problem as the Let's-make-something-new question. Or, given a list of ingredients, $i_1, i_2, \ldots, i_n$, find a new dish, d, that can be made from the ingredients.

We shall have to refine this characterization. Even so, it and indeed the domain of creative cuisine will remain apt for our discussion. Note, as a first step in revision, that an ingredient need not be a foodstuff. It may be a mode of treatment—Adrià *grilled* the mellon—or style of presentation—Adrià put the ice plant on top of the grapefruit, itself on top of the melon.

1.3 Heuristic Intuitions

A heuristic (aka: meta-heuristic) is a worthy yet imperfect procedure for solving a problem. Absent perfect solution procedures (not worth bothering with here because not plausible), heuristics are the next best thing. What might be promising heuristics for PM problems? Our purpose here is to discuss this question not in detail—by specifying the algorithms involved—but in general. The aim is to present heuristics abstractly—as "heuristic intuitions"—with the promise that details can be forthcoming. Before examining the trees, we first take a look at the forest.

1.3.1 PP: Product Placing Problems

There are fundamentally just two PP heuristic intuitions: substitutes and complements. Let parsley be the ingredient we wish to place. If a recipe (dish, product) calls for basil and you use parsley instead, you have substituted parsley for basil. Given an ingredient, i, we wish to place, the substitutes heuristic intuition enjoins us to find a list L of ingredients $i_1, i_2, \ldots$ that are appropriately similar to our i. Then, we find where the ingredients in L are used and determine whether i might profitably be substituted. Points arising:

1. What counts as "appropriately similar"? This, too, has to be determined by a heuristic. Besides subjective judgment, there are two principles, which may be used individually or in combination.

2. Under the first principle of similarity, i may be judged similar to j if they share, to a sufficient degree, certain properties, the relevant properties being determined by the purposes to hand. As attested by many restaurant menus, beef, chicken, pork, shrimp, scallops, and tofu are appropriately similar for many recipes. They may differ in price, but each is a popular source of protein.

3. Under the second principle of similarity, i is judged similar to j because they are closely related under a given classification scheme. Perhaps it is always possible in principle to reduce this second principle to the first. After all, the classification scheme had to be arrived at somehow. The point here is a practical one. Classification schemes may embody much valuable knowledge and it may often be expedient to use them directly, rather than to appeal to a "fundamental" approach and have the burden of reconstructing this knowledge.

Imagine instead that we have a recipe (dish, product) that does not call for parsley, yet we add it in. We have placed parsley as a *complement* to the recipe. Given an ingredient we wish to place, the complements heuristic tells us to find a list D of dishes $d_1, d_2, \ldots$ to which we might add our ingredient. Points arising:

1. "Adding an ingredient" should be understood expansively. Teflon coating on airplane toilets counts. Apple's adding style and variety to the design of their computers counts.

2. Complements and substitutes are both product (recipe) modifying heuristics. Complementing is more fundamental; it is addition without corresponding subtraction. Complementing may occur with other modifications in the host product. One might usefully organize a PP study around the kinds and cases of complements identified: substitutions, simple additions, complex additions, and so on.

3. Heuristics for finding productive complements are generally more challenging than heuristics for substitution. Think of discovering how pineapple complements ham. Perhaps a principle is involved, such as sweets complement salty things.

4. Other heuristics for finding complements may exploit known requirements or problems. Airplane toilets face the problem of maintaining cleanness with minimal washing fluid. Solution approaches include: better cleaning fluids, better application of the fluids, and a more receptive environment for application of the fluids.

5. Plausible heuristics for finding productive complements can be imagined for recipes. Is it possible to create a recipe book for other kinds of products? We address this in the sequel.

6. In any event, notice that known requirements and known problems should be useful in constraining search for complements.

1.3.2 PF: Product Finding Problems

PF is the obverse of PP, product placing. The heuristics employed for one can be used for the other. Given a starting or host product (dish, recipe) we can identify its components and search for substitutes. Points arising:

1. Understanding of the host product (dish, recipe), with its known requirements, problems, and other components, can help to constrain and inform the search for substitutes.

2. In more difficult cases, the starting product may not exist or may be known only partially. These are sometimes called "Holy Grail" products (white ink is an example). They don't exist and it is not known how to create them. Some of their aspects—components, properties, requirements—are more or less well understood. Something—some poorly understood "principle of science" perhaps—blocks the assembly of the envisioned product. Finding and circumventing the blockage is a the center of these problems.

How may complements to a starting product be discovered? The problem is not substantially different from finding complements in the PP case. See the discussion in the previous section.

1.3.3 PC: Product Composition Problems

PC problems are so different in degree from PP and PF problems that they differ in kind. In PP we focus on a single ingredient and try to find uses for it. In PF we focus on a single product (dish, recipe) and try to find improvements to it. We might think of product composition as beginning with a list of ingredients, which we select and combine in search of new products. This, however, is feckless in any straightforward sense. The combinatorics will defeat us. Given 100 ingredients there are 2^{100} (minus 1) ways to combine some or all of them. This is more than 10^{30}: 1,000,000,000,000,000,000,000,000,000,000. It is simply impossible to examine such a large number of things.

Heuristics are available. They fall into several categories, which may be combined, including the following:

1. Generative grammar. A grammar is a set of rules that tells us which combinations (e.g., of words) are permitted and which are not. Using a grammar and criteria of goodness or badness it may be possible to generate and select useful combinations from what becomes a much smaller search space.

2. Embryo. The embryo heuristic treats the problem as a more thoroughgoing, extensive PF problem. A starting product is identified and sequentially modified, producing increasing different products, until a stopping condition is reached. (An interesting stopping condition: incorporation of a given ingredient, i.) The embryo heuristic is named by analogy with a presumed mechanism of biological development. It has been used with considerable success by John Koza to discover electrical circuits. His programs have produced circuits that duplicate or infringe upon patented discoveries.

3. Hierarchical ascent. A random collection of paragraphs is much more likely to yield a meaningful, useful document than is a random collection of words. Paragraphs are meaningful on their own, words are not. Similarly, if we begin with a sentence and randomly insert, delete, substitute, etc. words we may well not progress as quickly as we would if we began with a document and modified it by inserting, substituting, etc. meaningful paragraphs.

> The biologist Lynn Margulis, and others, have assembled an impressive body of data supporting the argument that speciation usually, if not always, occurs by genome acquisition, rather than by incremental changes in gene frequency. If correct, the theory implies that speciation is saltational (non-gradualistic), and this is just what the historical record seems to indicate. The thought is that viruses and bacteria, which routinely insert their genomes into host cells, may occasionally enter into a sustainable, intimate, symbiotic association with a host organism. This may "change the game" and result in a new species. (See *Acquiring Genomes: A Theory of the Origins of Species*, by Lynn Margulis and Dorion Sagan, Basic Books, 2002.)
>
> Note that viruses and especially bacteria are more like paragraphs than like words: they stand on their own and have independent meaning in the sense that they have gene complexes that operate collectively to do useful things. Cows can eat grass because their stomachs contain bacteria that process the grass for the cows. Plants can fix nitrogen because they live in association with bacteria that fix the nitrogen. Put two functional things together and you have a decent chance of getting a functional result; better at least than if you put two non-functional things together.
>
> Analogs in the world of commerce? Accessories, for ladies as well as others. Meal planning, which combines individual recipes. There are others.

1.4 Particular Heuristics

1.4.1 Sizatola

Heuristic intuition: *Given documents that mention attributes of interest, the heavily populated categories they fall into are themselves interesting.*

Subject of a patent filing. Create a document collection of "hit files", documents that match a query of interest and are believed relevant to the problem. Map these documents to a classification scheme. (We currently support USPTO and Library of Congress.) Create a "browser" for exploring the hits and their mappings to the classification scheme. (Karen Chung wrote this code for us.)

1.4.2 Extracts by Phrase List

Create a document collection of "hit files", documents that match a query of interest and are believed relevant to the problem. Run an extraction program (now written by Ann Kuo) that finds and highlights the phrases listed, including only the abstract and claims sections of the patent.

This should be especially useful for phrases indicating uses of products or ideas.

Heuristic intuition: *Find things that are like the product we are trying to place, and see what these products are thought to be useful for and why. Suggestion is that these uses are candidates for our product.*

1.4.3 Related Companies

Create a document collection of "hit files", say patent documents that match a query of interest and are believed relevant to the problem. Run an extraction program (not currently existing) to find the assignees of these patents. Investigate the products and industries of the most frequently occurring assignees. Find and explore their Web sites.

Note: This is an extraction sok could easily take care of, easily do.

1.5 Remarks, Concluding and Previewing

The art of the kitchen is more than an apt metaphor for the product matching problem. Gastronomy is an instance, one that affords valuable lessons for the problem at large. In addition to creativity and an imaginative pallet, master chefs have to help them long lists of detailed product descriptions in the form of recipes and reviews. They have lists of ingredients and knowledge of their properties. They may draw on experience—their own and that of others, perhaps recorded—with combinations of ingredients and combinations of menu items.

Can the infrastructure of gastronomy be created in other areas? If this were done, is there reason to think that computational support could be devised that would materially "change the game" of product innovation?

Beginning with the second question, definite computational procedures are available for each of the heuristic intuitions described above. The procedures are proprietary but can be demonstrated. The technology involved is promising, but recent and in need of wider testing and refinement.

Can something like the infrastructure of gastronomy be made available in other domains—materials, plastics, pharmaceuticals, etc.—so that computational procedures may be brought to bear in supporting the heuristic intuitions? *The Joy of Cooking* is available in libraries and bookstores. Is it possible and if so what will it take to write *The Joy of Commerce*?

Much useful material is available in the form of document databases (corpi), lists, and classification schemes. Here are a few examples:

- The US Patent and Trademark Office (USPTO) maintains all recent patents in electronic form, on-line. The patents are organized by a classification scheme that identifies more than 4,000 different product categories.
- The Office of Management and Budget, has proposed development of a Comprehensive and Integrated North American Product Classification System. See:
  `http://www.whitehouse.gov/omb/fedreg/napcs1.html`
- From the Schedule B Export Codes description. See:
  `http://www.census.gov/foreign-trade/schedules/b/index.html`

  > About Schedule B Codes... There are millions of trade transactions occurring each year. These transactions are classified under approximately 8,000 different products leaving the United States. Every item

> that is exported is assigned a unique 10-digit identification code. Every 10-digit item is part of a series of progressively broader product categories. For example, concentrated frozen apple juice is assigned a 10-digit identifier that is aggregated into a broader category assigned a 6-digit identifier described as apple juice. The 6-digit identifier described as apple juice is aggregated into a broader category assigned a 4-digit identifier described as fruit juices and vegetable juices, etc. The 4-digit identifier is further aggregated into a broader category assigned a 2-digit identifier described as Preparations of Vegetables, Fruit, Nuts etc.

- The US Department of Defense maintains the MILSPEC standards, requirements and descriptions for thousands of products subject to DoD procurement. See:

  `http://www.dscc.dla.mil/Programs/MilSpec/`

- International standardization efforts are underway, sponsored by the United Nations, to describe and classify products and business processes involved in international trade.

In sum, the heuristic intuitions are valid. Computational procedures to support them have been identified and shown to have genuine promise. The infrastructure—information in the form of text and data—requirements are reasonably well understood and can be met, either from existing sources or by proprietary creation.

Product matching—the process of finding new uses for things—is poised for a profound advance.

Chapter 2

Present Practice

2.1 Overview of 3rd Party Intellectual Property Management Services

Much of the motivation—the value proposition—for the intellectual property management services industry is well captured in the following passage.

> To profit from intellectual property, companies must know whether it can be usefully and valuably applied in other industries and, if so, whether sales deals can convert that potential into revenue. At present, the market knowledge and deal-making infrastructure of most companies come up short because they rely on their internal business-development staffs and on narrow technical specialists to manage the sales of their intellectual assets. Much as companies look outside their own organizations to find the lawyers and bankers who manage their stock offerings, so too should they look outside to find the experts who can identify market applications for intellectual assets and convert these ideas into revenues. [ESV02]

Product Matching may be thought of as a component of intellectual property management services. In addition to "market knowledge and deal-making infrastructure", "companies must know whether [intellectual property] can be usefully and valuably applied in other industries and, if so, whether sales deals can convert that potential into revenue." Product Matching is our name for this latter capability, an essential component of intellectual property management.

Useful reviews of, and comments on, the intellectual property management industry are readily available (see [DE04, ESV02, Tri02]), so we will not duplicate that material. We do note that the size of the industry is estimated to be on the order of $100 billion:

> When a company has identified its marketable assets and assessed their approximate value, a network of conversion partnersintellectual-property brokers, consolidators, and business builderscan help it strike licensing and equity deals with buyers. The dozens of mostly small firms that oc-

> cupy this fastgrowing niche have pushed US licensing revenues to an estimated $100 billion a year. [ESV02]

At the same time, the customers of this industry are increasingly aware of the value of proper management of intellectual property.

> An increasing number of companies in sectors ranging from consumer products to high technology are discovering what companies in life sciences already know: the licensing of intellectual property can generate healthy revenues and bolster profits. [DE04]

Further, we note that most of the focus has been on the phases to commercialization after "a company has identified its marketable assets and assessed their approximate value." Authors will enjoin managers to identify and assess IP, but have very little to say about how to do it, other than to talk to the right sort of people, viz.:

> Companies aiming to extract the maximum value from intellectual property should begin by reviewing all of their patents, processes, and technologies and be prepared to do so at least once every three years. They will need two types of on-call knowledge partnersbroad-based technologists and industry specialiststo suggest applications for their technologies across a range of industries and then to confirm that each application is viable and to estimate its business impact in every relevant industry. That process should yield an A-list of ideas warranting immediate attention. [ESV02]

The efforts that are the subject of this report aim to provide automation to improve the quality of and reduce the time and cost required for these recommended reviews. In what follows we describe briefly the principal companies in the intellectual property management space. A short discussion concludes the chapter.

2.2 IP Asset Management Companies

The following companies use similar business models: managing the IP portfolio of their clients and aggressively seeking commercialization by third parties. None, with the possible exception of Information Holdings, appear to be particularly successful. However, despite their lack of success BTG plc and CTT have been in the business for more than 20 years. Thinkfire and ipValue Management are newly created subsidiaries with substantial financial backing.

2.2.1 Competitive Technologies, Inc. (CTT)

An Intellectual Property asset management firm, CTT manages its clients' IP and is also responsible for commercialization of selected technologies. CTT does this by licensing the technology, selling it, or establishing a company to commercialize it. Revenue is generated by receiving royalty payments, equity ownership in a company established to commercialize the technology, or both.

- $659,455 in revenue, $545,729 net loss (3 months ending April 30, 2003)
- $1,031,879 net loss (3 months ending April 30, 2002)
- 2003 financial status skewed due to award received in Materna case

2.2.2 BTG plc

BTG acquires IP from universities, federal institutions, and companies. The firm offers patent protection services, further technology development, and commercialization of the secured IP. Commercialization is driven primarily through licensing and venturing. Revenue is generated by milestone payments, royalties, and a commission on licensing agreements.

- Success stories include licensing of Magnetic Resonance Imaging (MRI) and the Hovercraft
- Total revenue for year to March 31, 2003 was £31.5 million (2002: £33.2 million)
- Pre-tax loss was £36.2 million (2002: £22.6 million)
- Net funds of £61.1 million (2002: £97.5 million)
- Number of employees down 30% from 234 to 164
- Profitability aimed for 2006

2.2.3 Thinkfire

Thinkfire is an “IP advisor” that manages the IP portfolio of companies, among which is Lucent. Analyzing the portfolios, Thinkfire helps its clients to identify technologies with high potential and engages in licensing negotiations, from which it generates revenue if successful.

- $1,900,000 in sales
- 19 total employees
- Thinkfire is a portfolio company of Intellectual Ventures, a private entrepreneurial partnership founded by two former Microsoft executives

2.2.4 ipValue Management

ipValue Management is a subsidiary of iFormation Group, which was formed by a partnership of BCG, Global Atlantic Partners, and Goldman Sachs. ipValue is an IP asset management firm with a bsiness model similar to the above.

- Founded in late 2001
- Recently contracted in 2002 to manage and license BTs patent portfolio (about 14,000 patents). This was ipValues first contract
- $30 million dollar backing by founder companies
- Revenue generated by sharing in royalties from licensing deal
- BT recently announced largest licensing agreement to date with LG Electronics (advised by ipValue)

2.2.5 Chipworks

ChipWorks performs a similar service to that of IPValue and Thinkfire in that it assesses the commercial viability of a companys IP portfolio, finds potential applications, and uses its legal abilities to enact deals. ChipWorks limits its services to the semiconductor industry. They also offer services helping companies to identify companies that might possibly be infringing on their patents in the hopes of obtaining licensing fees from those companies.

2.2.6 Information Holdings

Information Holdings (http://www.informationholdings.com/) is an Intellectual Property Management company that helps companies manage their intellectual property in a variety of ways. The company has four major businesses: MicroPatent, Master Data Center (MDC), IDRAC, and Liquent.

> Information Holdings Inc. is a leading provider of intellectual property and regulatory information products and services for professional end users in corporate and legal markets.
>
> The companys data businesses, which include MicroPatent, Master Data Centerł and IDRAC, provide a broad array of databases, information products and complimentary services used by intellectual property and regulatory professionals to research and manage information on a global basis. The companys Liquent business is a leading provider of life science regulatory intelligence and publishing solutions.
>
> Information Holdings Inc. is listed on the New York Stock Exchange and trades under the symbol: IHI.

In more detail (http://www.informationholdings.com/companies.html):

> The Intellectual Property (IP) Group is a leading provider of products and services meeting the IP needs of more than 10,000 of the world's leading corporations and law firms. The products and services we offer help our clients with IP research, management, licensing, and registration. Our leading brands include MicroPatent, Trademark.com , Aurigin, Master Data Center TM, and Liquent

> MicroPatent including its trademark.com and Aurigin divisions, is a global leader in the distribution of patent and trademark information. We provide cutting-edge solutions that streamline research, document delivery and document management. Uniting today's technology with patent expertise since 1989, MicroPatent delivers comprehensive, easy-to-use, and cost-effective systems. We are committed to developing IP systems that add value to Legal, R&D, Information Management, Competitive Intelligence and Marketing functions.
>
> Since 1971, Master Data Center TM has made nearly 2.5 million patent annuity payments for the largest companies and law firms in the world. With over 700 clients using its Patent Management and Trademark Management Software, and more than 300 annuity payment clients, MDC has a network of agents, foreign associates, employees and clients unique in the business.
>
> Liquent is the leading provider of content assembly and publishing solutions for the Life Sciences industry. Built on proven, world-class rendering technology that transforms proprietary content into open formats such as XML and PDF, Liquent software and services are used by 15 of the top 20 global pharmaceutical companies. Liquent's IDRAC database service is the leading source of regulatory information for the pharmaceutical and biologics industry.

2.2.6.1 MicroPatent

The most relevant to SizaTola is MicroPatent. MicroPatent provides the software tools that companies use to search for patents, legal information, and other research data. Their Aureka software provides a comprehensive software solution that allows companies to search patents, generating reports that allow them to do industry analyses based on the patents that they find. The major tool in their software is known as ThemeScape. This tool allows users to search through groups of specified patents to identify word patterns and relationships among the documents. It then presents the data in the form of an intuitive topological map so that the user may identify trends in and the important aspects of a particular group of patents. It also allows the user to identify the white spaces in the map that might possibly lead to commercialization opportunities.

MicroPatent also provides consulting services (formerly of Aurigin) to these customers so that they may get more out of the Aureka software. The consultants use a combination of the Aureka software as well as the software of company partners to analyze the technology trends of their competitors and of their industry, as well as to help the company to direct future innovation and/or M&A efforts to maintain their technological edge.

2.3 Online Technology Transfer Exchanges

The first three online marketplaces—Yet2.com, 2xfr.com, and NewIdeaTrade.com—represent the remainder of dozens of exchanges created in the late 1990s. Currently,

Yet2.com seems to be the most successful with the largest number of companies regularly posting or browsing. However, none of the online exchanges have lived up to expectations and few transactions have been made using their services.

2.3.1 Yet2.com

Yet2.com is a technology transfer marketplace that offers consulting and other services as a means of differentiating itself from other Internet marketplaces. Registered members can have IP portfolios analyzed and/ or identify technology needs and find answers to those needs. Essentially, yet2.com attempts to integrate itself into the technology transfer process. The company actively pursues technology providers, or large corporations who list technologies exclusively on the yet2.com website. There are currently about 500 technology providers. Yet2.com generates revenue on membership fees and royalty fees collected.

- $12,000,000 in sales; 75 total employees
- Recently acquired by Scipher plc for £6.8 million and renamed QED Intellectual Property

2.3.2 2xfr.com

2xfr.com is another tech transfer marketplace. It prides itself on simplicity and extensiveness in its tech listings. As opposed to yet2.com, 2xfr.com does not participate in the tech transfer process and merely provides a post for companies to find each other. However, it does claim to aggressively market listed technologies. It does not charge commission on deals and generates revenue solely from membership fees. It was established in 1996.

- 2xfr.com is part of a larger IP network called PatentCafe
- PatentCafe currently is a privately owned company funded by internet.com Fund II and Gray Cary Fund
- No financial information available; not widely mentioned

2.3.3 NewIdeaTrade.com

NewIdeaTrade.com is an online forum in the manner of Yet2.com and 2xfr.com, NewIdeaTrade.com generates revenue from advertisements, requiring no membership fees, browsing fees, or transaction fees. However, there is a fee for posting a technology.

2.3.4 NineSigma

NineSigma facilitates open market innovation on the behalf of its clients by using its network of resources and services to locate outside R&D sources. Current clients include DuPont and P&G. NineSigma and P&G recently entered into a strategic relationship agreement where NineSigma would use its services to help P&G reach a goal of generating 50% of its R&D from outside sources.

- $1,200,000 in sales
- 15 total employees
- Of 82 projects completed or in progress, has facilitated transactions on 26

2.3.5 Innocentive

InnoCentive is a company based in Andover, Massachusetts that helps large biology- and chemistry-based firms solve R&D issues. The firm was started up by Eli Lilly in 2001 and features an online database of small R&D problems posted by large firms including Eli Lilly, BASF, Dow Chemical, and Procter & Gamble in the hope that one of its 25,000 registered members may solve the problem.

The business model works by first charging the "seeker" company (the company seeking an R&D solution) a nominal fee of about $2,000 to post their problem on the database. The seeker company will offer an award ranging from $5,000 to $100,000 to the person who provides the best solution to their problem. Upon solution, the company pays a service fee to InnoCentive. A solution is found in approximately 40% of cases.

According to D&B, InnoCentive is a private company employing 13 people and generating revenues of about 1.7 million dollars a year. Since the company is private, the profit figures are hard to come by, but the growth of this company appears to be strong as it has expanded to include multiple large chemical firms and also has expanded to allow participation by solvers in China, India, Korea, and Japan.

Essentially, InnoCentive is limiting the scope of their business by only using one avenue of finding the solutions to a project. It additionally limits the scope of its business by choosing to only pursue chemically- or biologically-related R&D issues. It would take very little for the company to address all research issues by providing an online forum for any group willing to pay a reward and a service fee to find a solution from the greater research community.

2.3.6 UTEK Corp

UTEK Corp assists in the identification and acquisition of technology for client companies. Uses a U2B process that links universities with businesses. UTEKs services encompass working with both the universities and the businesses to commercialize technology for the universities and find technologies for acquisition for businesses. Similar to NineSigma in that it partners with companies to find technology for that company, but it also works from the other side, getting contracts from universities to license their technology.

- In Q1 Y03, UTEK had revenues of $504,271 but saw a $3 million decrease in operating assets, or about $.97 a share
- This is up from Q1 Y02 when there was only a $512,852 decrease in operating assets

2.4 Technology Development

2.4.1 Research Corporation Technologies (RCT)

Focusing primarily on biomedical fields, RCT works directly with universities and research institutions to commercialize early-stage technologies. Commercialization comes in both venture investment and licensing forms. RCT generates revenue through royalties and return on investments.

- (2001) $90 million in revenue with a total gain of about $11 million
- (2001) $286 million+ in total assets

2.4.2 New Venture Partners LLC

NVP is a venture capital firm investing in seed and early stages. Working directly with partner companies, NVP takes technologies out of their R&D labs and establishing companies to develop these technologies. NVP is backed primarily by Coller Capital and was created from Lucents New Venture Group. Currently, NVP has an exclusive relationship with BT to create ventures from its R&D lab.

- Less than $2 million in sales
- 11 to 20 total employees
- 4 ventures already spun out of BTs R&D labs

2.5 IP Sale/ License-Back

2.5.1 Duff & Phelps Capital Partners

Known as “DuffCap,” Duff & Phelps Capital Partners is recognized as one of the leaders in IP management. DuffCap has recently developed an innovative IP management strategy known as IP Sale/License-Back. This strategy calls for the complete acquisition, either by cash payment or equity, of patent or licensing rights and then licensing back the IP on a case-by-case basis. Concurrently, DuffCap collects the IP into a pool, and upon reaching critical mass, DuffCap proceeds to commercialize the IP through licensing to third parties and share a portion of the royalties with the contributing patent owners. Im not sure if this is the exact model as the financial arrangement is quite complex. I dont fully understand what they are trying to do. However, this is an innovative design and is the first to apply sale-leaseback to the IP management industry.

- Recently secured $1 billion in financing
- Claims to be finalizing first transactions

2.5.2 IP2IPO.com

Formed in 2001, IP2IPO invests in universities in order to commercialize any technologies or spin-out companies that arise from research the school does. IP2IPO has four long-term partnerships with universities, including a partnership with the University of Oxfords Chemistry Department whereby IP2IPO is entitled to 50% of the equity of companies spinning out of the Chem Dept. until 2015, and a 25-year partnership with Kings College London whereby IP2IPO will receive 20% of KCLs interest in spin-out companies and technologies.

- £222,000 revenues in fiscal year ending December 31, 2003 (2002: nil).
- Losses reduced to £583,000 (2002: £1.56 million).
- Six new spin-out companies formed in 2003, each of which IP2IPO received equity stakes in.

2.5.3 InteCap

By employing people with eclectic backgrounds in fields as diverse as finance and technology transfer, InteCap hopes to appeal to a wide range of companies in need of IP solutions. However, instead of keeping people with different backgrounds segregated and only using them for projects they might be familiar with, InteCap employs a team approach, which stimulates creativity, and since InteCap operates as a meritocracy, it further encourages creative thinking.

2.5.4 Edengene

Edengene was formed to give guidance to enterprises in such ways as disposing of non-strategic assets, restructuring portfolios, and make strategic acquisitions. As such, Edengene formed Edengene Finance to help oversee these operations for companies in fields such as telecommunications, media, technology, financial services, utilities, heavy engineering, and industrial goods; all transactions fall below £250 million. Edengene also shares risks and rewards by developing compensation models that match the complexity of the transactions.

- Top-line growth of 25% to nearly £6 million, for fiscal year ending January 21, 2004.
- 135% increase in profits, up to £850,000.
- Over half of Edengenes revenues come from developing innovative new products in the clients core markets.
- Survey conducted by Edengene among FTSE 100 companies revealed that 78% plan to invest substantially to create growth in the next 12 months.

2.5.5 TechEX

Founded at Yale University, TechEX is an internet-based exchange for buying and selling biomedical information. However, as opposed to some other exchanges which focus on undervalued or unused technology, TechEX focuses on emerging technologies, which allows for a bit more security, as the emerging technologies can often be a safer investment. Sort of like a digital classifieds section, TechEX allows institutions or corporations to register and then post online their technologies, or peruse and purchase already-posted technologies.

In May 2002, TechEX was acquired by UTEK Corporation, which uses its U2B process to help companies go from initial research and early-stage technologies to full-fledged corporations and help get the technologies out the door. This is a good partnership between TechEX and UTEK because TechEX is really just a posting area, and UTEK gives members of the TechEX listing a means by which to easily use the technology they recently purchased.

- Specific financial information regarding TechEX is unknown, but the UTEK Corporation, for the fiscal year ending December 31, 2003, had total revenues of $3.8 million (2002: $3.38 million), costs of $746,000 (2002: $763,000), and gross profit of $3 million (2002: $2.6 million).

- Currently, 326 universities and medical research organizations have signed on to be members of the TechEX.com listing. 2,130 new technologies have been posted online in the past 365 days, and to this date there have been 855 new inventions posted this year so far.

2.5.6 Delphion

Formed in May 2000 by Internet Capital Group (ICG) and IBM, Delphion is a wholly-owned company which specializes in the analysis, management, and extraction of IP information and technologies. Delphion uses IBMs technologies and was started with $35 million in funding from ICG. Delphion maintains lists of patents and employs a search engine for customers to quickly and easily extract pertinent information. The patent database is vast and includes both pending patents as well as granted patents; further, the patents are not limited to the US, as the database includes patents from Europe (including separate databases for Germany and Switzerland), and Japan.

- Although specific financial information is unavailable to the public, Hoovers.com estimates Delphion makes between $5-7 million in sales each year.

- Partners include CHI Research, Inc. (http://www.chiresearch.com), IBM (http://-www.ibm.com), IP.com (http://www.ip.com), Patent Awards (http://www.patent-awards.com/), Powerize.com (http://www.powerize.com), Verity, Inc. (http://-www.verity.com/), Wisdomain (http://www.wisdomain.com/index.htm), and Yet2.com (http://www.yet2.com). Yet2.com markets Delphion research in Europe.

- Although TechEX specializes in biomedical patents and IP, it seems to be a direct competitor of Delphion, in the sense that both companies are built to allow for easy searching of information. However, whereas Delphion merely lists patents, TechEX is more of a listing of technologies which may or may not be patented or up for a patent, and which companies can bid on. Another difference between the companies is that TechEX works in conjunction with its parent company, UTEK Corporation, to help customers go from technology to marketable product; Delphion, on the other hand, gives customers the tools with which they can search for technologies and then attempt to implement them on their own.

2.6 Other

2.6.1 ClearForest

From their website:

ClearForest Corporation is a provider of text-driven business intelligence solutions, supplying the analytical bridge between two previously disconnected worlds of information– unstructured text and enterprise data. Our award-winning solutions offer manufacturers, publishers, federal, chemical & financial service organizations critical links to situational context buried in text for use in Business Intelligence [BI] systems.

Adding this situational context to enterprise data systems empowers organizations to: uncover hidden relationships, evaluate events, discover unforeseen patterns and facilitate problem identification for rapid resolution. Applying this intelligence enables organizations to avoid loss of profit margins due to preventable write-offs, customer churn, legal settlements, warranty claims or inefficient product development cycles.

2.6.2 Invention Machine

www.invention-machine.com

2.6.3 Ideation International

http://www.ideationtriz.com/

2.6.4 Concept Net

http://www.conceptnet.org

2.7 Discussion

Supplementing the material above, Trippe [Tri02] is informative on current technology and decision support in the intellectual property management services industry. The technology focuses, as do we, on extracting information from collections of text. The collections of text described, however, are invariably collections of patents, from the US as well as abroad. Three types of software tools are in evidence:

- Document management systems
- Information extraction tools for patents
- Visualization tools

The leading software system appears to be Aurigin, recently acquired by Information Holdings, Inc. from a chapter 11 proceedings. The platform is described as follows (http://www.aurigin.com/static/test.htm):

> Aurekas interactive platform allows you to:
>
> - Gather IP you need to know about
> - Organize and annotate your data
> - Use advanced patent analysis tools
> - Use data and text mining to assess IP
> - Report on your findings and provide evocative visualizations

What is suggested in this passage appears upon closer examination to be the case: Aurigin (and other systems in the industry):

- Are focused on patent documents;
- Bring to bear mostly generic document management and informational retrieval tools; and
- Are not aimed at Product Placing exercises, but at other stages of the IP management process

The level of effort available for this report does not permit a definitive analysis. On the information available, the technical approaches described in the remainder of this report are richer and more powerful than existing methods in the industry. Continued attention should be paid, however, to current practices, including more thorough investigation of software and methods now in use. Our development plan, chapter **??**, envisages a number of activities. Continued investigation of current practice is covered by algorithm development (AD), 4d, page 90.

Chapter 3

Product Placing Report Template

The focus of this report is how the Product Placing (PP) process may be supported with a text-oriented DSS (decision support system). Each project aims to find new and innovative uses of a product, P. The DSS will present to the user a PP project template. The project template will encompass two kinds of information:

- Project-specific setup information: data and documents associated with the target product, P.
- Findings of the project, mainly: potential uses for P and supporting documentation.

The project template will also serve as a user interface to the DSS—its information stores and information extraction facilities—through which the user will produce, and then record, the project's findings.

Specifically, the template will support the following *project setup* information:

1. Collections of exemplary documents pertaining to P.

   An exemplary document is a specially-chosen document that with high quality describes a subject of interest and thus may be used to facilitate further inquiry into the subject (e.g., by providing key terms for search). Exemplary documents for a target product P may include engineering, design, or marketing documents, among others. The system will support document-based queries based on exemplary documents, or even passages from them. See [BK02] for details on the concept of an exemplary document.

2. One or more attribute lists describing P.

   These will normally be ascertained by human analysts, aided by the exemplary documents as well as interviews.

3. Known uses of P.

4. By known uses, attribute lists capturing the aspects of P that make the uses possible or attractive.
5. By known uses of P, collections of exemplary documents pertaining to the uses.
6. Conjectured possible uses of P.
7. By conjectured uses, attribute lists capturing the aspects of P that make the uses possible or attractive.
8. By conjectured uses, collections of exemplary documents pertaining to the uses.

The DSS, and specifically the project template, will support the following *project findings* information:

1. Potential uses found.
2. By potential uses found:
   (a) Exemplary documents
   (b) Project notes and documentation
3. Project reports from the DSS and other general documents produced during the analysis.

Part II

Heuristics for Product Matching

Chapter 4

Product Matching Heuristics

4.1 Central Rôle of Heuristics

- Multiple, related meanings and uses. From the Wikipedia (http://en.wikipedia.org/wiki/Heuristic):

  > The word comes from the same Greek root as "eureka", meaning "to find". A heuristic for a given problem is a way of directing your attention fruitfully to a solution. It is different from an algorithm in that it merely serves as a rule of thumb or guideline, as opposed to an invariant procedure. Heuristics may not always achieve the desired outcome, but can be extremely valuable to problem-solving processes. Good heuristics can dramatically reduce the time required to solve a problem by eliminating the need to consider unlikely possibilities or irrelevant states.

- The programme is to identify useful heuristics, and then provide computerized support for applying them.

4.2 Uses of Similar Products

Q is our product, the component product we wish to place.

Heuristic 1 (Uses of Similar Component Products) *If Q is similar to product R, and R is used for X, then Q may be useful for X.*

Kinds of Support (in This Context)

- Retrieval
- Extraction
- Thesauruses and query expansion

- Classification
- Heuristic matching
- Visualization

Using this heuristic in the context of Product Placing requires:

1. One or more document collections that contain descriptions of products similar to *Q* in interesting ways.

   Patents are one source of such documents. Web queries, e.g., with Google's API, are another. [What else?]

2. A description of *Q*, the source component product, for which we are trying to find new uses.

   Typically the description will be in the form of a list of *descriptor terms* (key words or phrases associated with *Q*), in which case we call it *termdes(Q)*. The description of *Q* may also be in the form of documents describing *Q*, in which case we call it *docdes(Q)*. In any event the description is originally obtained manually, by analysts working with relevant experts.

3. A way of finding documents, *R*, that are similar to *Q*: $\{R : sim(Q,R)\}$.

   The *sim(Q,R)* function may be realized in any of a number of ways, including:

   (a) Boolean retrieval using *termdes(Q)* (keyword description of *Q*) against an appropriate document collection.

   (b) Forms of information retrieval other than boolean, using *termdes(Q)* (keyword description of *Q*) against an appropriate document collection.

   [Say more here. Important to rank documents by degree of similarity/relevance. Then test: how far down to go?]

4. A way of finding the uses of the *Q*s, and extracting them for analysis.

   Possibilities include:

   (a) Simply have the analysts read the documents.

   (b) Use a concordance/KWIC program to focus on promising passages. (Ann Kuo's program, e.g.)

   (c) Employ information extraction techniques.

Also, in all of this we need a system, or even a doctrine, for recording and organizing information. For example, making a list of candidate uses and recording where they were found.

Note: opportunity here for empirical testing: (a) what methods work best for locating uses? (b) what methods work best for finding similar documents? All of this should be investigated wrt return for effort.

4.3 Sizatola

Heuristic 2 (Sizatola (Related Categories)) *Given: (a) a product or use, P, (b) a* collection *of documents,* **Q***, similar to P, (c) a categorization (or classification) scheme, CS, and (d) a categorization of* **Q** *based on CS, cat(***Q***, CS). Then, the more active a category is in cat(***Q***, CS) (e.g., the higher the 'hit count' of documents in the category), the more likely the category is pertinent to P.*

Note: The "Sizatola (Related Categories)" heuristic makes essential use of the IP in the Sizatola patent application.

Heuristic 3 (Similar Uses) *If P may be useful for X and X is similar to Y, then P may be useful for Y.*

Note: The "Similar Uses" heuristic makes essential use of the IP in the Sizatola patent application.

Heuristic 4 (Market Identification) *If P may be useful for X and X is a product in the M market or industry, then P may be useful for other products in M.*

Heuristic 5 (Product Finding) *If Q meets the requirements for specification S, and P is similar to Q, then P may be able to meet the requirements for specification S.*

Heuristic 6 (Competition Matching) *If Q competes with P in some market M, then if Q is useful for X in any market, so P may be useful.*

Heuristic 7 (Similar Attributes) *If Q is used for X which has attributes $\mathcal{A}$, and these attributes are important for, or characteristic of, Y, then Q may be useful for Y; and specifically:*

If Q is used for X for the sake of attributes $\mathcal{A}$ and these attributes are important for Y, then Q may be useful for Y.

Heuristic 8 (Matching Attributes) *If component product Q has attributes $\mathcal{A}$ and these attributes are important for, or characteristic of, use Y, then Q may be useful for Y.*

4.4 Product Similarity

Heuristic 9 (Product Similarity) *If component product Q is similar to end-use product X, then Q may be useful for X.*

There are a number of ways in which similarity between Q and a series of Xs might be measured, including:

- Q is represented by a patent (or other descriptive document), a product CTB is available, and the product categories are ranked by distance from Q.

- Q is represented by a list of (possibly weighted) characterizing attributes, a product CTB is available, and a retrieval engine uses the list of characterizing attributes to rank order either (a) all the documents in the product CTB, or (b) the product categories in the product CTB.

  Note: The characterizing attributes of a product might be automatically generated (at least in part) by a program that takes as input: (a) a file of attributes, and (b) a patent (or other file describing the product of interest), and that returned a 'hit list' with counts of matched attributes. This hit list might then be used to formulate a query, e.g., for Lemur, to produce (a) a ranked list of relevant patents, or (b) a ranked list of relevant product documents (from a product CTB), or (c) a ranked list of relevant products (aggregating the documents in a product CTB), or (d) a ranked list of companies from a company CTB, or all of the above.

4.5 Related Firms

Heuristic 10 (Related Firms 1) *If P may be useful for X and X is made by firm F and Y is made by firm F, then P may be useful for Y.*

Heuristic 11 (Related Firms 2) *If component product Q is similar to (descriptions associated with) firm F and X is made by firm F, then Q may be useful for X.*

Points arising:

- Both heuristics require a determination of the form *X is made by firm F*. Doing this automatically is akin to, but likely much more difficult than, discovering what a patent is used, or useful, for. The number of relevant firms, however, may be small enough that this determination can be made manually.
- Similarity between a component product Q and a firm F may be assessed in any of several ways, including:
  - Q is described by a patent, a collection of similar patents is assembled (retrieved), and the firms associated with these similar patents are identified.
  - A company CTB is available, as is a description of Q (e.g., a patent, a set of attributes, etc.). The description is used with the company CTB to produce (a) a ranked list of relevant company documents, or (b) a ranked list of relevant companies (aggregating the documents in the company CTB),
- Many refinements of method are possible. For example, the description of Q might be used to rank patents, the top companies associated with the ranked patents extracted, and the description of Q used to rank these companies via a company CTB.

$Id: pm-heuristics.tex,v 1.9 2004/09/29 01:29:12 sok Exp $

Chapter 5

Design for Heuristic #1: Uses of Similar Products

See §**??**, page **??**, for an outline of a report template for Product Placing studies. Ideally, a Product Matching DSS would provide support, in some form (if only for recording and organizing information), for each item in the report template list.

Recall Heuristic #1, page 27, repeated here for convenience. *Q* is our product, the component product we wish to place as part of a Product Placing effort.

Heuristic 1 (Uses of Similar Component Products) *If Q is similar to product R, and R is used for X, then Q may be useful for X.*

This heuristic might be represented in an abstract, *logical* form using pseudo-Prolog notation.

Logical Representation 1 (Uses of Similar Component Products)
```
maybe(usefulfor(Q,X)) :- similarto(Q,R), usefulfor(R,X).
```

In English: If *Q* is similar to *R* and *R* is useful for *X*, then it may be that *Q* is useful for *X*. This logical, pseudo-Prolog-like form should be taken as provisional. It serves the purposes, however, of concisely representing the heuristic and of suggesting how, later, it may be possible to implement interpreted languages to support the heuristic.

The question now is how to implement Logical Form 1 (for Heuristic #1). Points arising:

1. As suggested by Logical Form 1, the problem may be decomposed into two parts:

   - Similarity determination.
     Determining the *R*s that are similar to a particular *Q*, and (given that),
   - Use determination.
     Determining what the identified *R*s are useful for.

2. Similarity determination may be implemented in a number of ways. Both *Information Retrieval* and *Document Classification* methods are potentially useful. Each of these approaches comes in several forms.

3. Use determination is most appropriately implemented using one or another *Information Extraction* methods.

4. The `similarto` relation is naturally operationalized by representing a particular Q as a 'query' (as it is called in the Information Retrieval literature). Then q, a particular Q represented as a query, is supplied to a similarity mapping engine and applied to a document collection. The engine returns R, a set of documents, similar to q.

5. Thus we can think of the design space for implementing the `similarto` relation as

   Queries $\times$ *Similarity Mapping Methods* $\times$ *Document Collections*

6. This design space should also inform the user interface design. Basic use case:

   (a) User enters a query (or loads it from a file) .

   (b) User chooses a similarity mapping method.

   (c) User chooses a document collection.

   (d) User directs system to apply the query to the collection via the similarity mapping method, and to output the matching files (or pointers thereto) to a specific location (for use by the `usefulfor` subsystem).

7. Queries themselves may be in a number of different forms and of a number of different kinds, all of which may usefully co-exist in a single Product Placing study.

   In particular, queries may be single terms or Boolean combinations of terms or they may be sentences or even entire documents. Different similarity mapping methods are appropriate for different kinds of queries. Multiple queries may be present and of interest as, for example, analysts explore different aspects (uses, properties, industries) associated with a product. Also, it will be normal during a study to refine and modify various queries.

8. For the initial prototype, the design space should be kept simple, while keeping in mind the need to implement various parts of the space.

9. Even so, the initial prototype should allow some exploration of the design space, in order to afford experimental testing and comparison.

10. Initially, two similarity mapping methods should be available:

    boolean retrieval and tf.idf (a ranked retrieval method, based on the cosine rule).

    Both of these are available, I believe, in Lemur. They may also be available in Infomap NLP Software: An Open-Source Package for Natural Language Processing (http://infomap-nlp.sourceforge.net/).

11. Initially, two document collections should be available:

    US patents and retrieved Google pages (using the Google API plus filtering).

12. Initially, queries should be either single terms, or boolean combinations of terms, or raw term lists (as appropriate for a ranking algorithm such as tf.idf).

13. `usefulfor` operator. This also can be done in a variety of ways, by applying various kinds of Information Extraction techniques. The prototype should foresee and allow the user to pick among a list of possibilities, just as in the cases of the document collections and the similarity mapping methods. See the basic use case, item 6, above, page 32.

    Initially, we should identify predictive patterns, e.g., "used for", "used in", etc. and employ regular expression matching techniques to grab promising passages, and then produce a readable report. In this, we should draw extensively on the Python program written by my student, Ann Kuo.

14. Basic use case for the `usefulfor` operator.

    (a) Having completed an execution of the `similarto` operator (see item 6, above, page 32), the user selects an available `usefulfor` operator.

    (b) The user indicates the file set (or pointer set) produced by the `similarto` operator of interest.

    (c) Optionally, the user specifies matching paraters (e.g., which phrases the regex express should look for, how much text to grab around each hit, and so on).

    (d) The user indicates the file name and location for the output file, and directs the system to produce the file per directions.

    (e) Reading the file, and using a text editor, the user assembles, or adds to, a list of promising uses for *Q*, the product that is the subject of the exercise.

15. In addition, there will have to certain system administration functions performed. These include:

    (a) Creation of document collections.

    Patents. Complicating issues: the large number of patents makes it sensible to sample from them in traunches[1] of, say, 10,000. We should want to enable the user to sample from the patents by running queries on individual traunches or groups of traunches, as well as on the entire collection.

    Google. In the case of document collections derived from Google it will be necessary to feed the (or at least a) query to our software, which then goes to Google and downloads (and filters) a collection. Because of the limitation of 1000 queries per day (at a max of 10 hits per query), it might be necessary to support multi-day or multi-key queries, but not now, not at the beginning.

[1] http://www.investopedia.com/terms/t/traunch.asp, "One of many influxes of cash that is part of a single round of investment. . . . For example, you might hear: 'The $6.3 million is the first traunch of an $8 million round, with the extra $1.7 million expected over the next three months.' "

(b) Indexing of document collections.

Patents. Lemur wants one giant file to index, consisting of all the documents, separated by some markup. This is easily done, but recall the traunch strategy for patents. At least at first, I'd favor simply indexing by traunch only.

Google. Indexing is trickier because the downloaded documents may be of different types and formats, e.g., including PDF and Word. At first, it's best to filter for HTML and text only. Let's see what that brings us. Note that typically a user relying on a Google collection will submit a query and have to come back sometime much later to do the `usefulfor` operation (or specify it ahead of time and wait a while). That seems OK, seems fine. We just have to build the system so as to set expectations properly.

16. KISS! The most important thing is to get some end-to-end functionality asap. Then we can build in more functionality. The output of the system is a list of candidates products for which our product, *Q*, may be useful. We need to move with dispatch in order to generate such lists with facility. Then we can test and improve the methods, the design, and so on.

Part III

Information Sources

Chapter 6

Documents and Classification Schemes

Product Matching in general, and Product Placing in particular, are problems for which most of the available electronic information is in text (document), rather than in data (record), format. In consequence, the primary information bases for the Product Matching DSS must be collections of (textual) documents. The following document collections are the most important.

1. US Patents

   US patent documents are available commercially in "scrubbed" and well-formed HTML (i.e., a vendor has cleaned up what the Patent Office has begun). Sizatola has purchased 5 years of patents in this form, totaling about 820,000 distinct patents, and has used this collection for Product Matching with good results. The collection should be expanded to include all available patents.

2. Commercial Products

   What is perhaps the most important and valuable information base for Product Matching does not exist: a comprehensive knowledge base about *products*. If a new use is to be found for a component product, P, knowledge of the end-use products, which might use P as a component, is crucial. Chapter 7 describes the innovation—the CTB, or categorized text base—that will permit the construction of product knowledge bases for the purpose of Product Matching.

3. Firms

   Several of the heuristics given in chapter 4 require knowledge bases about firms and their products. As in the case of products, no appropriate comprehensive knowledge base exists, but one may be constructed as a CTB, again using the concepts and methods described in chapter 7.

4. Industries

As in the case of firms, several of the heuristics given in chapter 4 require knowledge bases about industries and their products and firms. As in the case of products, no appropriate comprehensive knowledge base exists, but one may be constructed as a CTB, again using the concepts and techniques described in chapter 7.

5. Special or Proprietary Collections

   The firms that develop and own IP for component products will typically produce a large amount of proprietary relevant documentation, including information on design and manufacturing, test data, market assessment information, and so on. This information is normally kept electronically and is in principle available for use in a Product Matching DSS. Specialized professional fields and associated technologies have even developed to manage such document collections. Most recently called *knowledge management*,[1] the older terms *special collections*[2] and *records management*[3] are still viable and apt.

   Regardless of terminology, internal, proprietary documents are potentially of great and differentiating value in undertaking Product Matching exercises. Many of the heuristics listed in chapter 4 may make use of proprietary collections and by requirement the DSS should be able to exploit them when available.

In addition to document collections, there is much valuable and pertinent information available in categorization schemes.[4] Categorization schemes are developed in many specialized fields and embody a great deal of specialized information, which may be exploited in the Product Matching process. Consider for example the ICD-9 (and soon ICD-10) classification system.

> There are two related classifications of diseases with similar titles, and a third classification on functioning and disability. The International Classification of Diseases (ICD) is the classification used to code and classify mortality data from death certificates. The International Classification of Diseases, Clinical Modification (ICD-9-CM) is used to code and classify morbidity data from the inpatient and outpatient records, physician offices, and most National Center for Health Statistics (NCHS) surveys. (http://www.cdc.gov/nchs/icd9.htm)

Many of the heuristics listed in chapter 4 may make use of classification schemes, in conjunction with document collections. The Product Matching DSS should, by requirement, be able to represent and exploit arbitrary classification systems in support of the heuristics. Finding these classification schemes, and obtaining copies, can be problematic, since many of them are built for very specialized purposes and made-to-order. They are often hard to find and even proprietary.

---

[1] See for example http://www.brint.com/km/, http://www.kmresource.com/, and http://www.mccombs.utexas.edu/kman/.

[2] E.g., http://www.uidaho.edu/special-collections/Other.Repositories.html.

[3] E.g., http://www.arma.org/.

[4] One may think of a classification scheme, e.g., for biological entities or library books, as a hierarchical categorization scheme. When the difference in meaning is not important we shall use either term interchangeably. Context will normally make clear which sense of *classification scheme* is meant.

Sizatola has obtained two important classification schemes and has represented them electronically in convenient format: the Library of Congress classification system, and the USPTO classification system. Other classification systems, especially pertaining to products, firms, and industries, are discussed in chapter 7. In addition, we note that a thesaurus is a common and important kind of classification system.

During phase 2 of this project, it will be important to systematically canvass for, and collect, classification schemes.

$Id: docs-classification.tex,v 1.1 2004/09/30 00:30:41 sok Exp $

Chapter 7

Categorized Document Bases

7.1 Background: The Sizatola Concept

Automation and decision support for Product Matching require that relevant information be available in machine-processable format. The great majority of such information resides in text documents. How is this information to be collected, organized and used for purposes of Product Matching? The Sizatola concept—subject of a patent application authored by Kimbrough, MacMillan, and Ranieri—offers an important and innovative approach to this question. The purpose of this chapter is to articulate in some detail how the Sizatola concept may be applied to the problem of collecting, organizing, and using textual information for purposes of Product Matching.

The core insight of the Sizatola concept is that classification schemes will often contain valuable information. This information, when combined properly with document collections, may be used to great advantage for a number of purposes, including those of Product Matching. Such uses rely essentially on heuristics and the discovery process is necessarily a noisy and approximate one. To illustrate, the US Patent and Trademark Office (USPTO) has developed a classification scheme into which every patent is uniquely placed. If Q is a patented component product and R is a patented component product in the same (or perhaps nearby) USPTO category as Q, then plausibly (heuristically) what R is used for may be something that Q could be used for.

7.2 An Example

Consider, for example, an exercise to find uses for a surfactant, call it Q. The USPTO classification scheme has 44 categories whose title contains the word *surfactant* (or *surfactants*):

1. 62.14.45 subclass 154 Surfactant or wetting agent
2. 70.1.9.37.42.31 subclass 31.59 Specified surfactant containing
3. 70.1.9.37.43.39 subclass 31.89 Specified surfactant containing

4. 90.1.13.12.4 subclass 22.14 With organic treating agent (e.g., solvent, surfactant, or reactant yielding soluble product, etc.)
5. 90.1.13.16 subclass 22.19 With organic treating agent (e.g., solvent, surfactant, or reactant yielding soluble product, etc.)
6. 113.2.20.45 subclass 270.1 Injecting a composition including a surfactant or co-surfactant
7. 113.2.35 subclass 300 Chemical inter-reaction of two or more introduced materials (e.g., selective plugging or surfactant)
8. 312.15.57 subclass 70.19 Two or more designated surfactant containing
9. 312.15.58 subclass 70.21 Amphoteric or zwitterionic surfactant containing
10. 312.15.59 subclass 70.22 Anionic surfactant containing
11. 312.15.60 subclass 70.27 Cationic surfactant containing
12. 312.15.61 subclass 70.31 Nonionic surfactant containing
13. 318.12.31.31.14.29.33 subclass 115 Identified adjuvant, i.e., surfactant, etc.
14. 318.33.111.146.111 subclass 493 Surfactant, emulsifier, or solvent
15. 323.5.32.62.34 subclass 112 Utilizing surfactant fatty acids or fatty acid esters (i.e., having seven or more atoms)
16. 362.3.3.10.20.16 subclass 155 Plural surfactant components (e.g., organic sulfate and sulfonate, sulfonate and amine oxide, etc.)
17. 362.3.3.48.63.55 subclass 289 Polyoxyalkyene containing surfactant devoid of covalently bonded anionic substituents
18. 362.3.3.48.63.56 subclass 290 Sulfur-containing anionically substituted surfactant
19. 362.3.3.48.73.74.28 subclass 331 Nonionic oxygen containing surfactant or polyacrylamide component
20. 362.3.3.48.76.79 subclass 340 Plural nonsoap organic surfactants (e.g., nonionic and anionically substituted, diverse nonionic surfactants, etc.)
21. 362.3.3.48.76.79.30 subclass 341 Nitrogen containing organic surfactant devoid of covalently bonded anionic substituents (e.g., cationic, nonionic, etc., surfctant)
22. 362.3.3.48.83 subclass 350 Nitrogen containing surfactant devoid of covalently bonded anionic substituents which is admixed with a diverse non-soap surfactant
23. 362.3.3.48.84 subclass 351 Sulfur containing anionically substituted surfactant which is admixed with a diverse non-soap surfactant

24. 362.3.3.48.84.82 subclass 352 Plural sulfur-containing, anionically substituted surfactants

25. 362.3.3.48.85.84 subclass 355 With non-soap surfactant component

26. 362.3.3.48.86 subclass 356 Oxygen containing surfactant devoid of covalently bonded anionic substituents (e.g., polyethoxylated alcohol, amine oxide, etc.)

27. 362.3.3.48.87 subclass 357 Sulfur-containing, anionically substituted surfactant

28. 362.3.13.69.107 subclass 413 Polyoxyalkylene containing surfactant devoid of covalently bonded anionic substituents

29. 362.3.13.69.108 subclass 414 Sulfur containing anionically substituted surfactant

30. 362.3.13.76 subclass 421 Polyoxyalkylene containing surfactant devoid of covalently bonded anionic substituents

31. 362.3.13.76.109 subclass 422 With diverse non-soap surfactant

32. 362.3.13.76.109.90 subclass 423 Nitrogen or phosphorus in organic surfactant devoid of covalently bonded anionic substituents

33. 362.3.13.76.109.91 subclass 424 Sulfur containing anionically substituted surfactant

34. 362.3.13.76.109.91.31 subclass 425 With soap or diverse sulfur containing surfactant component

35. 362.3.13.77 subclass 426 Sulfur containing anionically substituted surfactant

36. 362.3.13.77.110 subclass 427 With diverse non-soap surfactant

37. 362.3.13.77.110.92 subclass 428 Plural anionically substituted sulfur containing surfactants

38. 362.3.13.77.110.92.32 subclass 429 Sulfonate surfactant with sulfate monoester surfactant

39. 362.3.13.78 subclass 433 Nitrogen in organic surfactant devoid of covalently bonded anionic substituents

40. 362.3.15.89.117 subclass 450 With anionically substituted nonsoap surfactant and soap component

41. 362.4.30 subclass 535 Surfactant composition for cleaning agents (other than raw soap)

42. 362.4.30.119 subclass 536 Sulfoxy containing anionically substituted surfactant component

43. 364.120 subclass 975 CHARACTERIZED BY THE DESIGNATED SURFACTANT USED

44. 368.12 subclass 911 SURFACTANT FOR OTHER THAN POLYURE-THANE CELLULAR PRODUCT

The classification scheme even has a category for things that are not surfactants!

- 312.15.56 subclass 70.11 Polymer containing (nonsurfactant, natural or synthetic)

Note: In the items above there are three elements. The first is a "dotted number" indicating an exact address in the classification tree. This number was assigned by a Sizatola software program. Shorter numbers are higher up in the classification tree. The second element, prototypically "subclass 911", is taken from the USPTO classification scheme. It has meaning only in a larger context (subclass numbers are only unique within a class, which is not shown here). The third element is a bit of descriptive text, constituting the USPTO's brief description of the category.

Points arising:

1. Patents typically mention what they may be used for. Heuristically, it is reasonable (in looking for potential uses of the surfactant Q) to examine patents in the categories listed above (including the "nonsurfactant" category) and consider what they mention as their uses.

2. Neighboring categories or other categories suggested by the surfactant categories above may also be interesting for finding uses of Q.

3. If the number of patents in the candidate categories is too large, the number may be reduced by filtering. For example, an IR system might be used in various ways to rank by relevance the patents falling in the categories of interest. The ranking, combined with noting the number or density of 'hits' of retrieved documents by category, may even be used to identify which categories are of greatest interest.

4. Unlike the case of a surfactant, it may well be that the product of interest, Q, is not easily describable with a keyword contained in the USPTO category descriptions. Here it is possible to identify candidate documents, e.g., with an IR system, and then to map them to the USPTO classification scheme. Sizatola has developed general-purpose software that does this.[1] The resulting categories 'hit' by the candidate documents may then be examined as above.

Thus, the USPTO classification scheme, used in conjunction with the associated patents, may be used productively in finding new uses for things. But the patent text base and its classification scheme hardly exhaust the available information for Product Matching. How can the example just given be generalized and extended? The key to doing that lies in the concept—underlying the Sizatola idea—of a *categorization text base* or CTB.[2]

[1] The software written by Karen Chung.

[2] More generally, a categorization document base, CDB. Since the discussion here is limited to textual sources we will conduct the discussion with the term CTB.

7.3 CTBs: Categorization Text Bases

A Sizatola database—or a CTB—consists of three kinds of elements:

1. Documents (here we assume they are text documents)
2. A categorization system
3. A mapping between each document and the categorization system.

In general, a categorization system may be represented by a network of nodes (or vertices) corresponding to categories and edges (or arcs) corresponding to categorical relationships, such as subsumption (aka: is-a, as in an otter is a mammal). Categorization systems may themselves be categorized. They include:

- Simple lists of categories.

  Example: a list of uses for a component product.

- Categories organized as a classification tree.

  Examples: The Library of Congress classification system, the USPTO classification scheme, biological classification systems.

- Faceted indexing systems.

  "A faceted classification uses clearly defined, mutually exclusive, and collectively exhaustive aspects, properties, or characteristics (a.k.a. facets) of a class or specific subject".[3]

  Foskett [Fos82] and Taylor [Tay00] are good introductions to faceted indexing.

- Generalized networks, in which the meaning of concepts (nodes, or vertices) emerges in part from their associations with other concepts.

  Example: social networks.

  Barabási's popular science book is usefully suggestive [Bar03]. Newman has an excellent technical overview of recent developments in networks [New03].

We shall find simple lists and classification trees more directly, or at least more immediately, useful than either faceted indexing systems or general networks. Further points arising:

1. The mapping between documents and a categorization system should be complete: every document should have a categorization image. We allow, and this essentially requires, that categorization systems all have a NULL category, analogous to the relational database concept.
2. The mapping between documents and a categorization system should be invertible. Given a document its image in the categorization system may be retrieved. Given a category in the categorization system the associated documents may be retrieved.

[3] `http://www.slais.ubc.ca/courses/libr517/02-03-wt2/projects/faceted/`

3. The mapping between documents and a categorization system may be many-to-many. One document may be mapped to many vertices (concepts) in a categorization system (e.g., because the document is 'about' several topics of interest). Similarly, many documents may be mapped to one vertex.

4. A CDB (Sizatola database) may have many document collections, many categorization schemes, and many mappings from documents to categorization schemes. (Although the latter will be seen rarely.)

5. Document (text) collections need not be homogeneous. For example, certain documents mapped to a category may be singled out with special properties, e.g., the *exemplary documents* described by Blair and Kimbrough [BK02].

7.4 Finding, Creating, and Populating CTBs

The USPTO patent text base is especially useful for Product Matching. There is, as noted above, a classification scheme for patents and it is relatively easy to identify references in the patents to the classification scheme. This enables the Sizatola concept. In addition, USPTO published patents have a regular style that facilitates extracting index information from them. For example, authors, dates, patent owners (typically companies) may with relative ease be extracted automatically and used to support Product Matching heuristics, particularly in Heuristics

#10 Related Firms 1, page 30

#11 Related Firms 2, page 30

7.4.1 Finding Existing Text Databases

More generally, are there highly useful text bases, other than the USPTO patents,[4] that exist and may be collected or that do not exist but may be created? Potentially attractive existing text databases include:

1. Archives and *special collections* (term of art), especially in corporate environments, may contain useful documents.

   The Society of American Archivists (`http://www.archivists.org/`) and `http://www.archives.gov/` focus on archives in the public sector and for nonprofit organizations. The term of art in the business (for profit) world is *records management.* ARMA (`http://www.arma.org/`) is the main professional society, holding a large annual conference.

   Obtaining access to private archives will be problematic. On the positive side, if a close business relationship were developed with a firm having interesting corporate archives (such as DuPont, e.g.), access to the archives could be used for competitive differentiation.

[4] Or patent databases from other countries.

2. Standards, specifications, and regulations may prove useful sources of text data for Product Matching. These will often be created by government agencies and in consequence freely available or available for a small fee. The firm IHS (`http://www.ihs.com/`), for example, specializes in collecting and selling standards, specifications and regulations. Their collections may be of enormous value.

3. Commercial text bases, e.g., LexisNexis, ABI-Inform, and much else. The Penn library has a rich collection of these sources, listing 87 "Databases & Article Indexes" in the "Business & Management" category.

4. Governments and quangos,[5] e.g., the UN, produce often valuable text bases (besides patents). Again, the Penn library has a good listing online, under "E-Resources, Primary Business & Management (General) Government Information."

Although it is clear that these sources are very promising, there is not sufficient time available during phase 1 of this project (under which this report is undertaken) to canvass them. That should be a high-priority task during phase 2.

7.4.2 Creating and Populating CTBs

One further possibility remains to be discussed: creating useful CTBs for purposes of Product Matching. Just as the USPTO patent classification scheme is usefully (for us) populated with patents and thus is the basis for a CTB, so it would be useful to have, for example, a product classification scheme populated with documents having to do with the various product categories. Thus, the Sizatola concept may be employed to augment a classification scheme with a collection of text, as well as (e.g., above) to augment a collection of text with a classification scheme.

Creating such a CTB is in principle straightforward:

A Obtain a categorization system (e.g., product classification system) in machine-readable form.

B Use relevance ranking IR (Information Retrieval) techniques (e.g., Google's search engine) in conjunction with descriptions of the category to find documents highly relevant to each category. Use these documents to populate the CTB.

Points arising:

1. Requirement: the categories must be described with enough text to be used as the basis for using IR techniques to find related documents. See Figure 7.1, page 50, for the (minimal) associated text in an example product classification system.

   There is enough text here for a useful Google (or other search engine) query. Also, (a) there may be product classification systems endowed with more extensive descriptions, and (b) bootstrapping, manually or under program control, should be possible.

[5] "quango - QUAsi Non-Governmental Organization; an organization that is financed by the government yet acts independently of the government"—`http://www.thefreedictionary.com/quango`.

2. A product CTB so constructed ought to be especially useful for a number of our heuristics, in particular Heuristics

   #3 Similar Uses, page 29;

   #10 Related Firms 1, page 30;

   #11 Related Firms 2, page 30;

   #4 Market Identification, page 29;

   #5 Product Finding, page 29;

   #6 Competition Matching, page 29.

   And perhaps most importantly:

   #7 Similar Attributes, page 29;

   #8 Matching Attributes, page 29

   #9 Product Similarity, page 29.

   Note: A CTB (categorized text base) for products would be especially helpful here, and presumably of much greater use than the USPTO patents.

3. Categories and instances. Classification schemes will typically specify a hierarchy of categories, but be silent on particular instances. This is entirely appropriate. A biological classification scheme should contain a category corresponding to domesticated cats (*felix domesticus*?), and should not mention any particular pet cat. In the case of a product classification scheme, particular products will not belong to it. Tyvek, Nomex, Barri-cade, Responder, and Chemrel are particular commercial products. They will not be present in any product classification system. Given a product CTB, however, it would be valuable to extract, or otherwise obtain, a list of instance products and associate them with the appropriate categories.

4. Characteristic attributes. See below: the attributes "Comfortable. Disposable. Re-usable. Breathable. Waterproof. Elastic." are apparently important for coveralls. It will be valuable to extract or otherwise obtain a list of important attributes by category, and record the associations.

   Even if an explicit list is not created, a categorized collection of documents will contain much useful information in this regard, which can be algorithmically accessed in various ways. A similar point applies to specific products.

5. Scale. A product CTB would involve hundreds if not thousands of products, with multiple documents for each. As such it would be on a scale requiring machine representation and processing. The heuristic inferences it would support need, thus, not be subtle or difficult. Simple results extracted from a large scale database may be very useful.

   An anecdote may be help illustrate the point. We undertook an arbitrary Google query on "Coveralls" (from the list in Figure 7.1). The sixth entry returned, at

`http://www.chiefsupply.com/coveralls.phtml`,

was for "Tyvek Protective Coveralls and Boot Covers". This would be interesting, for example, if your product were similar to Tyvek. Perhaps your product could be used for coveralls and boots.

Further the (literal) query in Google, "used in coveralls", returns very interesting results, e.g., passages from selected hits:

- "Aprons are available in materials such as those used in coveralls and chemical-resistant gloves."
- "The aramid fibers used in coveralls garment construction provide superior flame resistance. The fibers will not melt, burn, drip or support combustion in air."
- "From my discussion with Dupont, the Tyvek used in coveralls, envelopes and building paper is the same resin woven in a different way" `http://solstice.crest.org/discussion/greenbuilding/-199910/msg00096.html`
- "Choina contacted du Pont, a maker of fire-resistant fabrics including Nomex Woven, a fairly heavy fabric used in coveralls, and Nomex Spunlaced, a far lighter" `http://www.ca5.uscourts.gov/opinions/pub/93/-93-03500-cv0.htm`
- From the sponsored link at `www.automotiveworkwear.com`: "Quality Red Kap Coveralls and speedsuits. Embroidery available." Suggesting that coveralls are similar to speedsuits.
- From the sponsored line by ShuBee & Tyvek Coveralls: "Comfortable. Disposable. Re-usable. Breathable. Waterproof. Elastic." Suggesting important attributes of coveralls.

6. Other commercial uses. A comprehensive product CTB, especially one mapped to different kinds of documents and even data (e.g., government records, import-export records, and so on) might be valuable commercially and certainly in the value-adding "downstream" activities of Product Matching, such as market assessment.

Will it actually be useful for Product Matching? This has to be tested, but the odds have to be judged quite favorable. We can strongly recommend that an effort be begun promptly to design and prototype a product CTB.

An interesting question is What other CTBs might be useful for Product Matching?

7.5 Specification of a Product CTB Prototype

At: `http://www.census.gov/epcd/www/naics.html` we find:

```
Cable-laying machinery
CAT scanners
Chemical contraceptives
Chemical reagents
Chick peas
Chippers
Civil engineering machinery
Clinical products
Coffins
Colour-flow doppler
Compacting machinery
Compression equipment
Computer servers
Construction machinery
Construction management services
Construction project management services
Construction work for elevated highways
Contraceptives
Contractor's all-risk insurance services
Cooling equipment
Copper ores
Coveralls
Credit and surety insurance services
Credit insurance services
CT scanners
Damage or loss insurance services
Data management services
Data network management services
Decoration works
Dermatological devices
Dialysis solutions
Diesel fuel
Diesel oil
Doppler equipment
Dresses
Dried peas
Drilling cement
Drinking-water distribution
```

Figure 7.1: From `Other_code.doc`, "Codes having the same description but another code" found at `http://simap.eu.int/EN/pub/src/main5.htm`, linked to from "Product Classification Systems," `http://faculty.philau.edu/russowl/product.html`, accessed July 18, 2004. For the complete list see Appendix B.

> A new North American Product Classification System (NAPCS) is presently under development, with initial focus on products of service industries. (NAPCS will focus on manufacturing products at a later date; for existing census codes, see the Numerical List.)

NAPCS, when it is ready, will likely be a classification system we would want to use for a Product CTB. Since it is not ready some other system must be found. An acceptable alternative may be the classification scheme available from the United States International Trade Commission (USITC), which is used to classify products for purposes of international trade. The key Web site is called "Tariff Affairs and Related Matters" and is found at `http://www.usitc.gov/taffairs.htm`, which has useful links to downloads. In particular see the "Tariff Database Download Area" at

`http://reportweb.usitc.gov/tariff/tariff_form.jsp`.

There, there is a link, "Description of Tariff Database and codes", leading to

`http://reportweb.usitc.gov/tariff/readme_hts.htm`

> This database is expressed in terms of the Harmonized Tariff Schedule of the United States (HTS) as of January 1, 2004. ... NOTES AND CAUTION: This database is being provided as an advisory tool only. For complete legal product descriptions and enacted/proclaimed tariff rates to be used on Customs Service documents, you must consult the current HTS and any supplements thereto, as well as any applicable Customs regulations and decisions.
>
> Additionally, the database provides tariffs at the tariff-line (8-digit HTS item) level. Therefore, where there is more than one "first" unit of quantity (i.e. statistical line items are collected in different units of quantity), the unit of quantity will appear as "NA" for the 8-digit HTS. More than one line of data will appear for those 8-digit HTS items having a change within the year in a special program rate or eligibility (such as for the GSP program) or change in any other data element for the HTS item.

Thus, the underlying classification scheme is the Harmonized Tariff Schedule of the United States (HTS). A portion of the relevant database has been downloaded (USITC.txt). Figure 7.2 shows a representative section of the file, i.e., of the HTS classification scheme.

The item numbers uniquely identify leaves in the HTS classification scheme, which itself is described in "Harmonized Tariff Schedule of the United States (2004) Supplement 1–Effective July 1, 2004" at

`http://hotdocs.usitc.gov/tariff_chapters_current/toc.html`.

The documents here are complete, but in PDF. Browsing, we note chapter 34, which is about

> Soap, organic surface-active agents, washing preparations, lubricating preparations, artificial waxes, prepared waxes, polishing or scouring preparations, candles and similar articles, modeling pastes, "dental waxes" and dental preparations with a basis of plaster

33079000 Depilatories and other perfumery, cosmetic or toilet preparations. nesoi

34011110 Castile soap in the form of bars, cakes or molded pieces or shapes

34011150 Soap, nesoi; organic surface-active products used as soap, in bars, cakes, pieces, soap-impregnated paper, wadding, felt, for toilet use

34011900 Soap; organic surface-active products used as soap, in bars, cakes, pieces; soap-impregnated paper, wadding, felt, not for toilet use

34012000 Soap, not in the form of bars, cakes, molded pieces or shapes

34013010 Organic surface-active products for wash skin, in liquid or cream, contain any aromatic/mod aromatic surface-active agent, put up for retail

34013050 Organic surface-active products and preparations for washing the skin, in liquid or cream form, put up for retail sale, nesoi

34021120 Linear alkylbenzene sulfonates

34021140 Anionic, aromatic or modified aromatic organic surface-active agents, whether or not put up for retail sale, nesoi

34021150 Nonaromatic anionic organic surface-active agents (other than soap)

Figure 7.2: Fragment of HTS classification scheme. From USITC.txt. (Note: ‘nesoi’ is apparently an acronym, for “Not Elsewhere Specified Or Included”.)

In the above list identifiers beginning with '34', e.g., 34011110, fall under chapter 34. A search on Google reveals that a number of foreign countries have posted on the Web text versions of the HTS. In particular, here is the Irish site for chapter 34:

`http://www.revenue.ie/services/customs/nomenclature04/ch34.htm`

Here is a passage from chapter 34 of the PDF on the US site:

```
3401 Soap; organic surface-active products and preparations
for use as soap, in the form of bars, cakes, molded
pieces or shapes, whether or not containing soap;
organic surface-active products and preparations for
washing the skin, in the form of liquid or cream and put up
for retail sale, whether or not containing soap; paper,
wadding, felt and nonwovens, impregnated, coated or
covered with soap or detergent:
Soap and organic surface-active products and
preparations, in the form of bars, cakes, molded
pieces or shapes, and paper, wadding, felt and
nonwovens, impregnated, coated or covered with
soap or detergent:
3401.11 For toilet use (including medicated products):
3401.11.10 00 Castile soap
3401.11.50 00 Other
3401.19.00 00 Other
3401.20.00 00 Soap in other forms
3401.30 Organic surface-active products and preparations for
washing the skin, in the form of liquid or cream and
put up for retail sale, whether or not containing soap:
3401.30.10 00 Containing any aromatic or modified aromatic
surface-active agent
3401.30.50 00 Other
3402 Organic surface-active agents (other than soap);
surface-active preparations, washing preparations
(including auxiliary washing preparations) and cleaning
preparations, whether or not containing soap, other than
those of heading 3401:
Organic surface-active agents, whether or not put up for sale
```

Recall the three essential elements of a CTB:

1. A classification system

   We have available several appropriate product classification schemes for a prototype product CTB:

   (a) The list from `Other_code.doc`, Figure 7.1, page 50. The full list is given in Appendix B "Example List of Product Categories," page 107.

   (b) (A portion of) the HTS classification scheme, Figure 7.2, page 52.

(c) The USPTO patent classification scheme. (See §7.2, page 41.)

The USPTO classification system is the least appropriate because many of the patents, and classes, are for processes, not products, and many of the products covered will be component products, rather than end-use products. The HTS classification scheme is promising, but will require some processing and much cleaning of the return sets. (For example, if you Google with an entire description of a category you are most likely to get several top hits that simply link to the HTS system—from several different countries and sites!) The simplest start would be with the `Other_code.doc` list, and that's where we should begin.

2. One or more document collections

   Google has an API allowing 1000 free queries per day to registered users, each query returning up to 10 links. (The book *Google Hacks* [CD03] is especially useful on this.)

   The first thing we should try is the Google API using the `Brief_Description` field of the USITC download, e.g., "Nonaromatic anionic organic surface-active agents (other than soap)". Note that some processing of the descriptions may be necessary.

   Also, once a query is issued and a set of links returned it will be necessary to process these links (Do we want them all?) and download the associated documents. We'll need to set some policy on how many we want to download.

3. A mapping between the documents and the classification system.

   This should be recorded in a relational database. Thus for each document we download we want to record at least:

   (a) The document's file name

   (b) The document's full path or location

   (c) The classification scheme intended

   (d) The document's location in the classification scheme

   (e) The query string used to retrieve the document

   (f) What processed the query string (e.g., Google API)

   (g) The rank of the document on the query

   (h) A timestamp on the query

   In addition, we shall want to index (e.g., into Lemur) all the documents downloaded. We need to do so in such a way that we can tie IR queries with SQL database queries. For example, after having downloaded the documents and indexed them, we might want to do a query on "fireproof" and return the hit rate by category and roll up the counts, much as in Karen's program. Thus, for example, we can ask Which kinds of products are most heavily associated with the set of attributes $\mathcal{A}$? and Which product categories are associated with Tyvek?

7.6 Required CTBs

On what topics should we build CTBs, besides products? Remember that a list counts as a classification system for these purposes. We should include:

1. Products
2. Companies
3. Industries
4. 'Holy Grail' challenges

7.7 Additional Reference Material

1. "The Knowledge Management Connection"
   `http://www.kmconnection.com/DOC100100.htm`
   is particularly interesting on faceted classification.
2. Useful sites on faceted classification:
   `http://www.poorbuthappy.com/fcd/`
   `http://besser.tsoa.nyu.edu/impact/f95/Papers-projects/Papers/perles.html`
3. A federal government report on the Central Product Classification system:
   `http://www.census.gov/eos/www/napcs/papers/cpcintro.pdf`
4. Industry Classification Schemes:
   `http://newarkwww.rutgers.edu/guides/business/ind-schemes.htm`
5. Vocabulary Standards and Classification Schemes:
   `http://www.chin.gc.ca/English/Standards/vocabulary_classification.html`
6. UDDI tModels: Classification Schemes, Taxonomies, Identifier Systems, and Relationships, Version 2.04 11 December 2002
   `http://uddi.org/taxonomies/UDDI_Taxonomy_tModels.htm`
7. This site
   `http://www.intracen.org/tis/impro.htm`
   mentions "Product classifications: The ITC Thesaurus incorporates the UN SITC3 classification. See WEB site"
   `http://unstats.un.org/unsd/cr/registry/regcs.asp?Cl=14&Lg=1&Co=265`

7.8 Tasks for Prototype Product CTB, Version 0.1

This section contains a succinct task list for creating a prototype Product CTB and deploying it in a useful demonstration.

7.8.1 Task 1: Acquire Initial Collection of Documents

1. Select a *product classification scheme* for creating the Product CTB.

   Our choice for the initial prototype will be the product list from `Other_code.doc`, reproduced in Appendix B, beginning on page 107. The product descriptions are simple and for the most part well-suited to serve as search engine queries.

2. Obtain a library of document conversion software, whose members are able to convert popular formats found on the Web (principally PDF (.pdf) and Word (.doc)) to text (or HTML or XML).

   This step is optional. It is a 'nice to have' rather than a 'must have' for the initial prototype.

3. Using the product descriptions in the chosen classification scheme (step 1 above), use the Google API to iteratively retrieve the top 1000 documents for each product description. That is, use the Google API to capture the URL and rank of the top 1000 documents for each product description query, and then download each of these documents.

   Note:

   - In general it may be necessary to edit or otherwise process the classification scheme elements to make them suitable as queries. The hope is that this will be unnecessary, at least at first, using the `Other_code.doc` product list as our classification scheme.
   - Initially, this exercise need be undertaken using only the products beginning with the letters A or B. This will facilitate testing and experimentation. Once the code is working, however, a second, complete CTB should be created using the entire list from `Other_code.doc`.

4. For each of the documents retrieved in step 3, if it is not an HTML (.htm, .html, . asp, .php, &c.) or plain text (.txt) document, either

   - convert it to a text format using an appropriate converter from the conversion library (step 2), or
   - remove it if it cannot be converted to a text formal.

5. For each of the documents retrieved in step 3, record the following information:

   (a) The document's file name

   Note: If the document has been converted, e.g., from `CTscanners.pdf` to `CTscanners.txt`, the latter, i.e., `CTscanners.txt`, is the file name to record.

   (b) The length in bytes of the document file

   (c) The document's original file extension

   (d) The program used to convert the original document to the form stored for indexing in the system.

   See the conversion library, step 2 above.

(e) The document's full path or location (on the local system, relative to the home directory of the Product CTB application)

(f) The original location (full URL) of the document

(g) The classification scheme intended
`Other_code.doc` in the case of the first prototype.

(h) The document's location in the classification scheme
Since `Other_code.doc` is a flat list, not a hierarchical classification scheme, just give the document's place in the list, e.g., 5 for `Aerial spraying services`.

(i) The query string used to retrieve the document
In the first prototype, simply the product identifying string, e.g., *Aerial spraying services*. If the string has been processed, record here the result of the processing. In any event, record here the actual string used to query the associated document base (in the case of Google, the base of documents indexed by Google).

(j) What processed the query string
(e.g., Google API)

(k) The rank of the document on the query
If known.

(l) A timestamp on the query

Question for Cristy: Do you want me to specify a relational schema for this? If you want, I can just give you the SQL `CREATE TABLE` statement(s). —Steve

7.8.2 Task 2: Index with Lemur and Build Web Interface

This can be done without recourse to the information described in step 5 of Task 1, §7.8.1 (page 56).

7.8.3 Task 3: Index by Attributes by Document, Sum by Category

1. For each attribute in a given file listing attributes (one per line) and for each document, determine whether there is a match between the attribute and the document, and if so, how many matches. Record this number.

   Note:

   - Attributes may be n-grams—phrases—e.g., *able to flex*, and they may be given in boolean combinations, e.g., *flexible OR flexile OR able to flex*
   - Attribute-to-document matching should be case insensitive.
   - A nice-to-have: stemming of terms in the documents for purposes of matching.

2. Provide a report via a Web interface that, given an attribute (from the file listing attributes) and a product category, reports:

   (a) The **Hit Score** for the category and the attribute: the number of matches for the given attribute summed for all documents in the category, divided by the total number of bytes for the documents in the category (see step 5 of Task 1, §7.8.1 (page 56) and multiplied by a stored constant—say 1000—for purposes of display

   (b) The **Hit Rate** for the category and the attribute: the number of documents in the category that have at least one match with the given attribute, divided by the number of documents in the category, and multiplied by 100.

3. Provide a report of **found attributes** in the document collection by processing each document with a Part of Speech (POS) tagger and extracting nouns and adjectives. For each document, record the nouns and adjectives that appear in it as well as the number of times they appear in it. Provide a facility by which an analyst may view this report for all the documents in the collection, totaling up the number of 'hits' by term (noun or adjective), showing the term, and showing its part of speech. Provide a facility by which an analyst may remove terms from this list (e.g., terms occurring very frequently or very rarely) and then create a file that lists the remaining terms as attributes for subsequent matching (see step 1 above).

7.8.4 Task 4: Build an Attribute by Category Sortable Display

Build a Web application showing a table whose rows are attributes from a given file listing attributes, whose columns are product categories from a given file listing product categories, and whose entries present both the Hit Score (2a in Task 3, §7.8.3) and the Hit Rate (2b in Task 3, §7.8.3). Allow the user to sort by attribute and by category.

Part IV

Use Cases

Chapter 8

Basic Use Cases

8.1 Use Case: Create a List of Attributes

The course of events begins with the user reviewing the (possibly empty) collection of declared attribute lists for product P. The user decides to create a new list of attributes and so informs the system. The system responds by presenting the user with a text field. The user enters a list of attributes, one attribute per line. When finished, the user saves the list and names it. The system adds the list to the existing collection of attribute lists.

8.2 Use Case: Create a List of Attributes for Known Uses

The course of events begins with the user reviewing the (possibly empty) collection of declared attribute lists for the *uses* of product P. The user decides to create a new list of attributes and so informs the system. The system responds by presenting the user with a text field. The user enters a list of attributes, one attribute per line. When finished, the user saves the list and names it. The system adds the list to the existing collection of attribute lists.

8.3 Use Case: Create an Extended Boolean Query

The course of events begins with the user reviewing the (possibly empty) collection of declared boolean queries for product P. The user decides to create a new boolean query and so informs the system. The system responds by presenting the user with a text field. The user enters an extended boolean query (including proximity operators and regular expressions, as well as logical connectives). The user directs the system to validate the query. When satisfied, the user saves the query, names it, and optionally describes it in an associated text field. The system adds the query to the existing collection of boolean queries for the project.

8.4 Use Case: Create a Text Query

The course of events begins with the user reviewing the (possibly empty) collection of declared text queries for product P. The user decides to create a new text query and so informs the system. The system responds by presenting the user with a document selection dialog box. The user selects the documents to constitute the text query. When satisfied, the user saves the query, names it, and optionally describes it in an associated text field. The system adds the query to the existing collection of text queries for the project.

8.5 Use Case: Create a List of Known Uses

The course of events begins with the user reviewing the (possibly empty) collection of declared lists of known uses for product P. The user decides to create a new list of known uses and so informs the system. The system responds by presenting the user with a text field. The user enters a list of known uses, one use per line. When finished, the user saves the list and names it. The system adds the list to the existing collection of lists of known uses.

8.6 Use Case: Create a List of Requirements

The course of events begins with the user reviewing the (possibly empty) collection of declared lists of requirements for product P. The user decides to create a new list of requirements and so informs the system. The system responds by presenting the user with a text field. The user enters a list of requirements, one per line. When finished, the user saves the list and names it. The system adds the list to the existing collection of lists of requirements for product P.

8.7 Use Case: Associate Exemplary Documents with an Entity

Note: For the concept of an exemplary document see [BK02].

The course of events begins with the user viewing the (possibly empty) collection of exemplary documents associated with the entity of interest. (The entity may be a product, the use of a product, an industry, a particular firm, etc.) The user indicates to the system a document to be added to the collection. The system responds by adding the document, converting it to a standard format and indexing it as appropriate. The user continues to add documents until satisfied.

8.8 Use Case: Associate Firms with Products

The course of events begins with the user viewing the (possibly empty) list of associations between products and firms that produce, distribute, or sell them. The user

indicates to the system an association to be added to the list. The system responds by adding the association. The user continues to add documents until satisfied.

Chapter 9

Use Cases for Heuristics

9.1 Uses of Similar Component Products

Recall heuristic #1, page 27:

Heuristic 1 (Uses of Similar Component Products) *If Q is similar to product R, and R is used for X, then Q may be useful for X.*

9.1.1 Use Case: Find Uses of Similar Component Products

The course of events begins with a user wishing to find uses for a product P, is in possession of a description of P, and of a text base which may be mined for component products, Q, that are similar to P. The user identifies a query (either an attribute query, §8.1, page 61, or a text query, §8.4, page 62). The user identifies a text base on which to run the query (e.g., US patents). The user directs the system to run the query on the text base. The user then requests a report from the system, listing the uses for the products, Q, found in the retrieved documents.

Note: See appendix D for technical details on extracting uses from documents.

9.2 Sizatola (Related Categories)

Recall heuristic #2, page 29:

Heuristic 2 (Sizatola (Related Categories)) *Given: (a) a product or use, P, (b) a* collection *of documents, $\mathbf{Q}$, similar to P, (c) a categorization (or classification) scheme, CS, and (d) a categorization of $\mathbf{Q}$ based on CS, cat($\mathbf{Q}$, CS). Then, the more active a category is in cat($\mathbf{Q}$, CS) (e.g., the higher the 'hit count' of documents in the category), the more likely the category is pertinent to P.*

9.2.1 Use Case: Find Similar Product Categories

The course of events begins with a user wishing to find uses for a product P, is in possession of a description of P, and has access to a product CTB (Categorized Text Base). The user identifies a query (either a boolean query, §8.3, page 61, or a text query, §8.4, page 62). The user identifies a product CTB. The user directs the system to execute the query on the product CTB. The system responds with a display showing the 'hit rate' of matches to the query by category.

Note: the product CTB may be organized as a list (e.g., a list of products) or as a classification tree. In the later case, the system responds with a 'browser' that may be explored hierarchically.

9.2.2 Use Case: Find Related Categories

Note: This is a generalization of the "Find Similar Product Categories" use case, §9.2.1. Here, we assume a categorization scheme *other* than a product CTB, e.g., the Library of Congress classification scheme, the USPTO classification scheme, a thesaurus of technical terms for an industry, an industrial classification scheme.

The course of events begins with a user wishing to find uses for a product P, is in possession of a description of P, and has access to a CTB (Categorized Text Base) or a document base that may be mapped to a categorization scheme, *CS*. The user identifies a query (either a boolean query, §8.3, page 61, or a text query, §8.4, page 62). The user identifies a document base. The user directs the system to execute the query on the document base. The system responds with a display showing the 'hit rate' of matches to the query by category in the *CS*.

Note: the product CTB may be organized as a list (e.g., a list of products) or as a classification tree. In the later case, the system responds with a 'browser' that may be explored hierarchically.

9.3 Similar Uses

Recall heuristic #3, page 29:

Heuristic 3 (Similar Uses) *If P may be useful for X and X is similar to Y, then P may be useful for Y.*

9.3.1 Use Case: Find Uses Similar to a Known Use

The course of events begins with a user wishing to find uses for a product P that are similar to known uses of the product. (It is assumed that at least one such list has been prepared. See the use case in §8.5, page 62.) It is further assumed that a CTB (Categorized Text Base) of products is available. The user identifies to the system: (a) a list of known uses, and (b) a CTB of products. The user requests that the system match (a) to (b). The system responds with a tabular display in which known uses are

row headings, products are column headings, and table entries are similarity scores for use-product combinations. The user may sort the display row-wise and column-wise in order to focus on topics of greatest interest. The user may click on a table entry and the system will respond with a display giving the user access to the most relevant documents in that use-product category.

9.4 Market Identification

Recall heuristic #4, page 29:

Heuristic 4 (Market Identification) *If P may be useful for X and X is a product in the M market or industry, then P may be useful for other products in M.*

9.4.1 Use Case: Market Identification

Note: This use case presumes a CTB for markets or industries. As such it relies on the IP that is the subject of the Sizatola patent application.

The course of events begins with a user, in possession of a list of uses of product P (see the use case in §8.5, page 62), who wishes to discover which industries or markets P is used in. The user identifies to the system: (a) a list of known uses, and (b) a CTB of markets or industries. The user requests that the system match (a) to (b). The system responds with a tabular display in which known uses are row headings, industries are column headings, and table entries are similarity scores for use-industry combinations. The user may sort the display row-wise and column-wise in order to focus on topics of greatest interest. The user may click on a table entry and the system will respond with a display giving the user access to the most relevant documents in that use-industry category.

9.5 Product Finding

Recall heuristic #5, page 29:

Heuristic 5 (Product Finding) *If Q meets the requirements for specification S, and P is similar to Q, then P may be able to meet the requirements for specification S.*

9.5.1 Use Case: Product Finding

Note: This use case matches the given requirements, or specifications, of a component to a text base (categorized or not) of products. The use case in §8.6, page 62 covers the creation of a requirements list for a product, P.

The course of events begins with a user, in possession of a list of requirements for a component product P, who wishes to find products that match the given list of requirements. The user identifies to the system: (a) a list of known uses, and (b) a text

base (categorized or not) containing information about products. The user requests that the system match (a) to (b). In the case that the text base is not categorized, the system responds with a relevance ranked list of documents. In the case that the text base is categorized, the system responds with a 'browser' that may be explored hierarchically.

Further note: This heuristic may also be supported by extending the use case to allow document-based queries, replacing the requirements list with one or more documents to be used to specify the requirements.

9.6 Competition Matching

Recall heuristic #6, page 29:

Heuristic 6 (Competition Matching) *If Q competes with P in some market M, then if Q is useful for X in any market, so P may be useful.*

9.6.1 Use Case: Competition Matching

This is a variant of the Market Identification use case, §9.4.1, page 67.

9.7 Similar Attributes

Recall heuristic #7, page 29:

Heuristic 7 (Similar Attributes) *If Q is used for X which has attributes $\mathcal{A}$, and these attributes are important for, or characteristic of, Y, then Q may be useful for Y; and specifically:*

If Q is used for X for the sake of attributes $\mathcal{A}$ and these attributes are important for Y, then Q may be useful for Y.

9.7.1 Use Case: Similar Attributes

The course of events begins with a user, in possession of a list of uses of product P (see the use case in §8.5, page 62) as well as as lists of attributes for those uses (see the use case in §8.2, page 61), and who wishes to discover uses similar to the known (or conjectured or assumed or postulated) uses of P.

The user identifies to the system: (a) a known use, and (b) a list of attributes for the known use (or a collection of exemplary documents for the known use). The user requests that the system match (a) to (b). The system responds with a tabular display in which known uses are row headings, industries are column headings, and table entries are similarity scores for use-industry combinations. The user may sort the display row-wise and column-wise in order to focus on topics of greatest interest. The user may click on a table entry and the system will respond with a display giving the user access to the most relevant documents in that use-industry category.

9.8 Product Similarity

Recall heuristic #9, page 29:

Heuristic 9 (Product Similarity) *If component product Q is similar to end-use product X, then Q may be useful for X.*

9.8.1 Use Case: Product Similarity

The course of events begins with a user wishing to find uses for a product P, is in possession of a set of attributes, $\mathcal{A}$ (see the use case in §8.1, page 61), or exemplary documents (see the use case in §8.7 for P, and has access to a *product* CTB (Categorized Text Base). The user identifies a query (either a boolean query, §8.3, page 61, or a text query, §8.4, page 62). The user identifies a product CTB. The user directs the system to execute the query on the product CTB. The system responds with a ranked list of matching products.

9.9 Related Firms 1

Recall heuristic #10, page 30:

Heuristic 10 (Related Firms 1) *If P may be useful for X and X is made by firm F and Y is made by firm F, then P may be useful for Y.*

9.9.1 Use Case: Related Firms 1

The course of events begin with a user wishing to find uses for a product P, is in possession of a list of known uses for P (see the use case in §8.5, page 62) and an associated list of firms for each known use (see the use case in §8.8, page 62). The user indicates to the system the use or uses of interest, as well as the list of associated firms. The system responds with a list of products associated with the firms in question.

9.10 Related Firms 2

Recall heuristic #11, page 30:

Heuristic 11 (Related Firms 2) *If component product Q is similar to (descriptions associated with) firm F and X is made by firm F, then Q may be useful for X.*

The course of events begin with a user wishing to find uses for a product P, is in possession of a list of attributes or exemplary documents, as well as a *firm* (business organization) CTB. The user indicates to the system the attribute set or document collection of interest (describing P), as well as the appropriate firm CTB. The system responds with a list of products associated with the firms in question.

Part V

Core Technologies

Chapter 10

Information Retrieval

/* Since this is such a mature area, it's best to limit the discussion to a brief overview and to focus on the open source software that's available. Perhaps a word or two on using SOK's DCB algorithm would be appropriate. */ `http://www-2.cs.cmu.edu/~lemur/`

Chapter 11

Document Distance Mapping

There are, in product matching efforts, many situations in which measuring similarity distances among documents is useful. For example, given one or more documents describing a product, Q, it will be helpful to find documents describing similar products.

Standard IR (Information Retrieval) techniques require the analyst to extract search terms from the descriptions of Q. It is these extracted terms that are used as inputs to a retrieval algorithm. The required term extraction process takes time and effort, and is vulnerable to bias. misunderstanding, and incomplete specification of information. In essence, the analyst undertakes a feature extraction task, with little in the way of theory, computational support, or even feedback for guidance.

A strong case can be made that proper use of an entire document (or documents) for purposes of similarity mapping will perform better than using judgmentally-extracted terms from a source document as input to an IR algorithm. Theoretical arguments are available (e.g., [CV04, LCL$^+$03, LV97]) and are persuasive, at least in principle. Empirical considerations, however, must trump all others. Happily, available evidence is positive (e.g., [BCL02, BLM03, CV04, LCL$^+$03]),[1] and—as we shall discuss—experimental testing is available without undue burden. Although the best-of-breed method for measuring similarity among documents for purposes of product matching is not known, it is discoverable at minimal cost.

Further points arising:

1. Distance mapping for documents should be seen as a (particularly attractive) form of similarity mapping. This is directly usable in the case described above, as well as in other situations. For example, given a use for a product, documents may be found describing that use and distance mapping may then be used to find similar documents (and uses). More specifically, distance mapping for similarity measurement should be useful in Heuristics:

   #1 Uses of Similar Component Products, page 27;

   #2 Sizatola (Related Categories), page 29;

   #3 Similar Uses, page 29;

[1] We note that *some* of this evidence is controversial. See [Goo02].

#5 Product Finding, page 29

2. Distance mapping for similarity measurement may be used for relevance ranking, and hence for document filtering, just as IR methods are used, if they provide a ranking.

3. Document clustering, Chapter 13, is broadly useful in product matching, and all document clustering methods rely on measuring distances between documents.

4. Distance mapping for similarity measurement will typically be much more computationally expensive than IR methods. Delay and cost of computation will generally be much less a critical issue in product matching than in Information Retrieval.

In the next several sections we discuss available and promising techniques for document distance mapping for similarly measurement.

11.1 DCB

The DCB algorithm/representation was developed originally by Kimbrough and Oliver [KO94] based on a suggestion by David C. Blair [Bla74]. DCB was conceived as an IR algorithm for relevance-ranked retrieval and was explored and tested for that purpose [DKKO97, DHK+97]. Dworman's Ph.D. thesis work [Dwo99a, DKP00] used the algorithm for identifying word patterns in document collections. The DCB algorithm/representation may also be used (and has been used) for document distance mapping. Because it has not previously been reported, we describe that use of the algorithm/representation here.

The first step is indexing. This is done by determining, for every document and every keyword (or key phrase) in an appropriately thorough list, whether a document contains a keyword. Having done this, we in effect have a matrix, called K, whose entries are all 1s and 0s, whose rows correspond to keywords, and whose columns correspond to documents [SM83, Rij79]. A particular K might look like this, where: rows = keyterms and columns = documents.

$$K = \begin{pmatrix} 1 & \dots & 0 & 1 \\ \vdots & k_{i,j} & \vdots & \vdots \\ 1 & \dots & 1 & 0 \end{pmatrix} \tag{11.1}$$

Element $K_{i,j}$ is 1 if keyword i occurs (at least once) in document j; otherwise it is 0. Thus, K is interpreted as a term-by-document matrix. Given K, we can compute the L matrix as follows.

$$L = K \cdot K^T \tag{11.2}$$

L is a square, symmetric document-by-document matrix with elements $L_{i,j}$, indicating the number of documents containing both terms i and j. L is called a *co-occurrence* matrix in the general literature, because it holds information on the co-occurrence of the various terms. Note that $L_{i,i}$ is simply the number of documents containing term

i. Dworman's work is based on exploiting the L matrix for finding word patterns in documents.

Given L, we can compute the M matrix as follows.

$$L \cdot K = K \cdot K^T \cdot K = M \tag{11.3}$$

M has the same dimensions as L, and like L is a term-by-document matrix. The entries of M, $M_{i,j}$ provide an inverse distance score for the relevance of document j to term i (the larger the closer). Experiments have shown that this measure may often fit well with human judgment [DKKO97, DHK$^+$97].

The use and interpretation of the L and M matrices have been reported in the works cited. Two other matrices, however, are associated with the DCB algorithm/representation, but have not been described in previous publications. We report these here.

Given K, we can compute the J matrix as follows.

$$J = K^T \cdot K \tag{11.4}$$

J is something of a dual of L. Both are square, symmetric matrices. Where L is term-by-term, J is document-by-document. Where $L_{i,j}$ is the number of documents containing terms i and j, $J_{i,j}$ is the number of terms occurring in both document i and document j. Again, this is an inverse distance measure; the higher the score between two documents, the more similar they are purported to be.

Finally, given J, we can computer the I matrix as follows.

$$J \cdot K = K^T \cdot K \cdot K^T = I \tag{11.5}$$

I is more than the dual of M, the two matrices are transposes of each other

$$I^T = M \tag{11.6}$$

thus $I_{i,j} = M_{j,i}$. Looking at I row-wise (or M column-wise) the entries represent importance (inverse distance) scores for the terms, given the document. Notice that in I a term may have a non-zero, or even fairly high, value for a given document *even though the term does not occur in the document.* Similarly, in M a document may be deemed highly relevant to a given term *even if the term does not occur in the document.* In fact, document j may be judged more relevant to term i than document j', even though j' contains the term and j does not. This surprising feature of the DCB algorithm/representation is in fact a requirement of any good relevance ranking algorithm. See [DKKO97, KO94] for further discussion.

Points arising:

1. To date, the efficacy of the I and J matrices has not been tested at all. Testing of the L matrix has been persuasive [Dwo99a] and testing of the M matrix has produced in positive results [DKKO97, DHK$^+$97]. Much more extensive empirical testing of these uses of the DCB algorithm/representation is merited.

2. The matrix multiplications required by DCB expensive. Limiting the representation to binary entries (1s and 0s) in the K matrix allows Kimbrough's propriety algorithm to be used, thereby greatly reducing the computational costs.

3. As a consequence of the restriction of the K matrix to binary entries, DCB cannot be expected to work well if there are large documents in the collected to be indexed. What the threshold of large is is not known.

4. Very small documents would appear to be problematic, too, at least without, e.g., term augmentation with thesauruses. Experience with paragraph-length documents, however, has been very positive.

11.2 LSI: Latent Semantic Indexing

LSI—Latent Semantic Indexing (aka: LSA: Latent Semantic Analysis)—was developed at Bell Labs and has enjoyed intensive investigation. The algorithm is presented in the open literature (notably [DDF$^+$90, Dum03, GD98, LD97]) and is quite complex, so we will not describe it here. Further, the published applications of technique are impressive. LSI/A may be used for a number of interesting purposes, including distance mapping of documents.

Telcordia Technologies, Inc. (http://lsi.research.telcordia.com/) markets products based on LSI/A.[2]

The National Institute for Technology & Liberal Education (NITLE, `http://www.nitle.org/semanti` has an informative Web site pertaining to LSI. NITLE's homepage (`http://www.nitle.org/`) describes what they are about. They have produced a layman's introduction to LSI (`http://research.nitle.org/lsi/`) and it isn't bad, although there's a lot of marketing-style verbiage before they get down to nuts and bolts.

A shorter and perhaps better intro is at

`http://www.cs.berkeley.edu/~jasonh/classes/sims240/-sims-240-final-paper-lsi.htm`

Useful collections of papers and links:

`http://www.upmf-grenoble.fr/sciedu/blemaire/lsa.html`

`http://www.cs.utk.edu/~lsi/papers/`

Probably the appendix to [LD97] is the best introduction (the paper is terrific, too).

Points arising:

1. LSI relies on SVD (singular value decomposition) of term-document matrices. Once the SVD algorithm has been run and the decomposition obtained some judgment (and perhaps testing) is required in order to determine the appropriate degree of dimension reduction to be used. This is somewhat of an obstacle to its routine use in a DSS for Product Matching.

2. LSI is computationally demanding. It would be a major effort to produce usable software for it that could handle, say, 100,000 documents or more. Most of the publicly available SVD software modules cannot be expected to scale up well, although this has to be tested.

3. Is open source software available? I would think so, but a quick search produced nothing. Source Forge lists only one project, Kassandra—

[2] Web site, accessed July 14, 2004, is dated 2003. It is not clear how active the company is.

`http://sourceforge.net/projects/kassandra/`

—and it is only in the planning stage. If we can find open source LSI software, we will need to stress test it. That may well be worth doing given the evident power and success of LSI.

11.3 Compression Methods

Distance mapping by using file compression is a recently-innovated application of the theory of Kolmogorov complexity, which measures the amount of information in the syntactic sense, in a file [BCL02, BLM03, CV04, LCL+03, LV97]. The Kolmogorov complexity of a file is the number of bits in an ideally-compressed version of the file. It is a theoretical construct in the sense that the Kolmogorov complexity is uncomputable. For practical purposes, however, it may be approximated and in applications work common, everyday compression algorithms (such as ZIP) are often used.

The fundamental idea in using compression to measure distance between two documents is this. If documents A and B are very similar, then the compression of their concatenation, AB, will be relatively short, just as the compression of AA is hardly longer than the compression of A alone. Conversely, if the compression of AB is comparatively large, then A and B are presumed to be rather different. More generally, by concatenating and compressing all pairs of documents it is possible to create a document-document distance matrix.

Points arising:

1. This is a new area. It is not known which compression algorithms work best or why. Different authors use different distance measures and even these have not achieved consensus.

2. A great strength of compression methods is their ability to compare seemingly very different entities and accurately find similarities between them. One of the successes of these methods is in constructing phylogenetic trees from DNA sequences. Because evolution moves genes around, lengthens and shortens DNA, etc., the DNA from even closely related species may not directly match well at all. Compression methods have been successful in finding hidden patterns of similarity and correctly adducing known phylogenies.

3. It is not known whether and if so how standard processing of documents will affect distance rankings in a collection of documents. In the studies to date, the documents are accepted ‘as is’, no processing is done to eliminate stop words, pronouns, etc. (as is done in DCB, LSI, and IR generally). This may be appropriate when constructing phylogenies; it may not be appropriate otherwise. Note that the main successes of compression algorithms on text collections has been in constructing historical evolution of documents, e.g., [BCL02, BLM03].

4. Compression methods for distance mapping will be computationally expensive when large numbers of documents are involved. The calculations, however, may

be done in parallel and may be done incrementally. On the last point, the addition of a new document to a collection does not change any of the previously-calculated distances. This is *not* a property of DCB or LSI.

5. Compression methods hold the prospect of a more or less 'parameter-free' approach to distance mapping. Tuning and judgment may be minimized once a basic method is chosen. Thus, the indexing may be undertaken automatically and the methods may more easily be used in a DSS.

11.4 Experimental Designs

Measuring similarity is one of the main uses for distance mapping of documents in the context of Product Matching. It is a core capability that must be provided and provided well. The following approaches are worth considering:

1. IR relevance ranking methods
2. DCB
3. LSI
4. Compression methods

IR relevance ranking methods are highly available but theoretically (or at least apparently) weaker than the other methods. Also, the IR methods require careful extraction of search terms and formulation of queries. This places a knowledge burden on the analyst that may be unreasonable. The other three methods allow the analyst to begin with one or more documents describing, e.g., the product whose uses are to be found. The algorithm in each case works directly with the originating documents, eliminating the need to formulate specific queries and interpret the results.

It is not known which of the methods actually works best in the sense of returning the best set of documents needed for the purposes of Product Matching. This calls for experimental investigation. We have in mind two basic designs.

The first design might be called a *within method validation test.* The question addressed is how well the rankings obtained by a given method agree with human judgments. Kimbrough developed an appropriate experimental procedure, which is described in[DKKO97, DHK$^+$97]. Essentially, subjects are asked to judge among presented documents for relevance to a specified task. The subjects' judgments are then compared to the rankings produced by the algorithm. In the current context, subjects could be given originating documents, say descriptions of products, and asked to choose between presented patents for relevance and similarity.

The second design might be call a *between method validation test.* This could be done with a variant of the methodology for the first design. Here, subjects would be asked to choose between documents ranked by different distance mapping methods. Also, short of experiments, simply describing how the methods differ in their results would be interesting and useful.

Ann Kuo, working with Steve Kimbrough, has begun exploring and enhancing the compression-based metric reported by Benedetto et al. [BCL02]. Examining arbitrary

sequences of 200 and 500 patents has revealed a consistent pattern of bimodal distances for most of the patents examined. For each patent, typically, a few other patents will be judged close and the great majority judged to be very distant. If this patten holds up it may prove useful in filtering a large collection of documents. Of more immediate interest, we have enough data to begin experimental testing of the within method validation sort. This should be undertaken as soon as possible.

`$Id: doc-distances.tex,v 1.2 2004/07/16 04:38:26 sok Exp $`

Chapter 12

Information Extraction

[Car97a] [JM02, chapter 3]

Chapter 13

Document Clustering

/* Note: include support vector machines; see [Dum03] and references therein. [JM02, chapter 4]. Include also use of SOMs (self-organizing maps), Kohonen maps, etc. Note that these are, perhaps, the poor man's Sizatola; the classification schemes have to be imputed, rather than given exogenously, based on experience and information. */

Part VI

Development Plan

Chapter 14

Activities and Schedule

14.1 Activities

As stated at the outset, the principal goals of this report are these:

- To articulate the concept of Product Matching and to indicate something of its *potential* value.
- Briefly to examine present practices.
- To articulate the concept of a Product Matching DSS and to indicate how such a system may improve on present practice in a manner that amply rewards the required investment.
- To provide in detail a high-level design for a Product Matching DSS, including an estimated schedule of deliverables and costs.

The fourth, last goal is the subject of the present chapter. Before discussing details, we wish to emphasize several points:

1. Our approach to a high-level design has been this:

   (a) Identify key heuristics—value-adding computations—that will be the focus of the DSS.

   (b) Articulate the heuristics with characteristic use cases.

   (c) Articulate the use cases by indicating how the key component computations may be undertaken, specifying both information bases and algorithms.

   Detailed design remains to be done, using the high-level design as both a guide and specification of requirements.

2. We have assumed a 12 month period for phase 2, after which a fully working system should be available. This precludes anything like a complete Product Matching DSS. Instead, it properly focuses on creating and deploying high-value

functionality as soon as possible. Further, the level of effort and cost estimates are for an entry-level, but functioning system, rather than a complete or at least highly mature system.

3. We favor a so-called *middle-out* development strategy for the Product Matching DSS, in contrast to *top-down* and *bottom-up* approaches. Under the middle-out development strategy, system functionality is decomposed into distinct units of value and these units are delivered sequentially. The strategy places a premium on delivering—for use and evaluation—a meaningful level of system functionality at the earliest possible time, and then building incrementally on that. The method—sometimes referred to as *permanent prototyping*—recognizes that as a result it may be necessary to revise designs and even discard working code.

4. Given the high-level design and the middle-out development strategy, phase 2 of the project needs to include the following activities:

   (a) Detailed design and documentation (DD&D)

   This will be organized about the use cases. Detailed designs are needed for user interfaces, information base requirements, and algorithmic implementation of computational operators.

   (b) Coding and system testing (C&ST)

   Writing and configuring the software (some of which will be available as open source) and testing the resulting system for compliance with design requirements.

   (c) Information base acquisition and development (IBA&D)

   The information base includes: document collections (and access methods), categorization schemes, and creation of CTBs (categorized text bases). Note: cleaning, formatting, and transforming the information base will require significant programming effort.

   (d) Algorithm development (AD)

   An ongoing effort is needed to continue search for useful algorithms and software. The Product Matching concept is novel and there is essentially no literature explicitly linking text processing, information retrieval, information extraction, etc. algorithms to it. In addition, Product Matching has, as evidenced by Sizatola and Serendipity, occasioned origination of new algorithms. Especially as experience is gained, as feedback is obtained, and as design requirements become more specific, it will be rewarding to continue to seek improved algorithmic support.

   (e) Field testing (FT)

   We propose a case-based approach. Phase 2 of the project should include a series of Product Matching (specifically Product Placing) studies that use all available system technology and that are evaluated by appropriate stakeholders. This will provide a proper basis for estimating the value of Product Matching DSS.

(f) Evaluation (E)

The project needs to be evaluated from several perspectives:

i. System verification. Does the delivered software implement its design properly?

ii. System operation. Is the system stable? What are its computational requirements? Is it responsive to users?

iii. System validation. What is the value the DSS contributes to the Product Matching process? What results does it produce and how good are they?

iv. System improvement. How should the system be further developed?

5. We emphasize the immediate use and evaluation of the system as it becomes available. The schedule envisions conducting four full test cases of Product Placing exercises, which will deliver useful information to DuPont, will provide feedback to the system designers and developers, and help in the evaluation of the overall project.

14.2 Schedule

Month 1

DD&D Detailed design for basic the user interface (template), chapter 3, and basic use cases, chapter 8.

C&ST Begin work on user interface and basic use cases.

IBA&D Canvass systematically and draw up an acquisition plan for information base acquisition and development.

AD As pertains to use case 1.

FT —

E —

Month 2

DD&D Detailed design for use case 1.

C&ST Complete work on basic use cases and user interface.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 2.

FT —

E —

Month 3

DD&D Detailed design for use case 2.

C&ST Complete work on use case 1.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 3.

FT Identify and begin work on test case 1 (Product Placing) to be undertaken with system support for use case 1.

E Draw up evaluation plan for field testing of system.

Month 4

DD&D Detailed design for use case 3.

C&ST Complete work on use case 2.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 3.

FT Conduct test case 1.

E —

Month 5

DD&D Detailed design for use case 4.

C&ST Complete work on use case 3.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 4.

FT Complete test case 1. Identify test case 2.

E —

Month 6

DD&D Detailed design for use case 5.

C&ST Complete work on use case 4.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 5.

FT Conduct test case 2.

E Evaluate results from test case 1 and the system as used to support it.

Month 7

DD&D Detailed design for use case 6.

C&ST Complete work on use case 5.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 6.

FT Complete test case 2. Identify test case 3.

E —

Month 8

DD&D Detailed design for use case 7.

C&ST Complete work on use case 6.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 7.

FT Conduct test case 3.

E Evaluate results from test case 2 and the system as used to support it.

Month 9

DD&D Detailed design for use case 8.

C&ST Complete work on use case 7.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 8.

FT Complete test case 3. Identify test case 4.

E —

Month 10

DD&D Detailed design for use case 9.

C&ST Complete work on use case 8.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 9.

FT Conduct test case 4.

E Evaluate results from test case 3 and the system as used to support it.

Month 11

DD&D Detailed design for use case 10.

C&ST Complete work on use case 9.

IBA&D Continue to canvass for, acquire, and develop the information base.

AD As pertains to use case 10.

FT Complete test case 4.

E —

Month 12

DD&D —

C&ST Complete work on use case 10.

IBA&D —

AD —

FT —

E Complete system evaluation and write up final report.

`$Id: dev-plan.tex,v 1.3 2004/10/02 21:45:53 sok Exp $`

Appendix A

Open Source Sources

A.1 The Lemur Toolkit for Language Modeling and Information Retrieval

`http://www-2.cs.cmu.edu/~lemur/`

> Language modeling has recently emerged as an attractive new framework for text information retrieval, leveraging work on language modeling from other areas such as speech recognition and statistical natural language processing. Research carried out at a number of sites has confirmed that the language modeling approach is an effective and theoretically attractive probabilistic framework for building information retrieval (IR) systems.
>
> The Lemur Toolkit is designed to facilitate research in language modeling and information retrieval, where IR is broadly interpreted to include such technologies as ad hoc and distributed retrieval, cross-language IR, summarization, filtering, and classification. The toolkit supports indexing of large-scale text databases, the construction of simple language models for documents, queries, or subcollections, and the implementation of retrieval systems based on language models as well as a variety of other retrieval models. The system is written in the C and C++ languages, and is designed as a research system to run under Unix operating systems, although it can also run under Windows.
>
> The toolkit is being developed as part of the Lemur Project, a collaboration between the Computer Science Department at the University of Massachusetts and the School of Computer Science at Carnegie Mellon University.

A.2 txtkit

http://www.txtkit.sw.ofcd.com/

txtkit is an Open Source visual text mining tool for exploring large amounts of multilingual texts. It's a multiuser-application which mainly focuses on the process of reading and reasoning as a series of decisions and events. To expand this single perspective activity txtkit collects all of the users mining data and uses them to create content recommendations through collaborative filtering. The software requires Mac OS X 10.3 and Internet access.

A.3 KDTREE 2

www.arxiv.org/abs/physics/0408067

KDTREE 2: Fortran 95 and C++ software to efficiently search for near neighbors in a multi-dimensional Euclidean space

Many data-based statistical algorithms require that one find *near or nearest neighbors* to a given vector among a set of points in that vector space, usually with Euclidean topology. The k-d data structure and search algorithms are the generalization of classical binary search trees to higher dimensional spaces, so that one may locate near neighbors to an example vector in $O(\log N)$ time instead of the brute-force O(N) time, with N being the size of the data base. KDTREE2 is a Fortran 95 module, and a parallel set of C++ classes which implement tree construction and search routines to find either a set of m nearest neighbors to an example, or all the neighbors within some Euclidean distance r. The two versions are independent and function fully on their own. Considerable care has been taken in the implementation of the search methods, resulting in substantially higher computational efficiency (up to an order of magnitude faster) than the author's previous Internet-distributed version. Architectural improvements include rearrangement for memory cache-friendly performance, heap-based priority queues for large $m$searches, and more effective pruning of search paths by geometrical constraints to avoid wasted effort. The improvements are the most potent in the more difficult and slowest cases: larger data base sizes, higher dimensionality manifolds containing the data set, and larger numbers of neighbors to search for. The C++ implementation requires the Standard Template Library as well as the BOOST C++ library be installed.

A.4 FIHC 1.0

`http://www.cs.sfu.ca/~ddm/` Then click on Software and examine FIHC.

Frequent Itemset-based Hierarchical Clustering (FIHC) is a program that constructs a document cluster hierarchy from a set of unlabeled documents based on frequent itemsets. The output of FIHC is a XML file which allows user to visualize the hierarchy or serves as input data for further processing. The project is developed at Simon Fraser University. This software is free for academic and research purpose.

A.5 SVM Light

svmlight.joachims.org

SVMlight is an implementation of Support Vector Machines (SVMs) in C. The main features of the program are the following:

- fast optimization algorithm
  - working set selection based on steepest feasible descent
  - "shrinking" heuristic
  - caching of kernel evaluations
  - use of folding in the linear case
- solves classification and regression problems. For multivariate and structured outputs use SVMstruct.
- solves ranking problems (e. g. learning retrieval functions in STRIVER search engine).
- computes XiAlpha-estimates of the error rate, the precision, and the recall
- efficiently computes Leave-One-Out estimates of the error rate, the precision, and the recall
- includes algorithm for approximately training large transductive SVMs (TSVMs) (see also Spectral Graph Transducer)
- can train SVMs with cost models and example dependent costs
- allows restarts from specified vector of dual variables
- handles many thousands of support vectors
- handles several hundred-thousands of training examples
- supports standard kernel functions and lets you define your own
- uses sparse vector representation

A.6 IBM Integrated Ontology Development Toolkit

`www.alphaworks.ibm.com/tech/semanticstk?Open&ca=daw-flHnt-081204`

IBM Integrated Ontology Development Toolkit (formerly named IBM Semantics Toolkit) is designed for storage, manipulation, query, and inference of ontologies and corresponding instances. A major purpose is to establish an end-to-end ontology engineering environment tightly integrated with dominant Meta-Object Facility (MOF)-based modeling and application development tools. As such, it provides a platform for managing RDF metadata and reduces the amount of programming required for the development of metadata-intensive applications.

This toolkit contains three main components (Orient, EODM, and RStar), which are designed for users of different levels.

1. Orient, as a visual ontology management tool, is mainly used by domain experts who have limited computer knowledge but who are familiar with specific domain knowledge. It is designed as a set of loosely-coupled cooperative Eclipse plug-ins.Now Orient can run on Eclipse 3.0 or compatiable. Orient is a joint R&D project of IBM China Research Laboratory, Beijing, and APEX Data and Knowledge Management Lab, Shanghai Jiao Tong University.

2. EODM and RStar provide a set of programming APIs for programmers and IT specialists. EODM is designed to provide a high performance OO interface for the programmer. Now, it is mainly used to manage ontology-level data with limited size.

3. RStar is a high performance metadata repository based on Resource Description Framework (RDF). RStar is designed and developed to support storage and query for large-scale RDF data with high performance. It uses the RDBMS (Currently, IBM DB2 and IBM CloudScape are supported) to store data and defines RStar query language (RSQL) for retrieval. It offers programmers rich APIs to effectively load, retrieve and manage RDF data.

Orient's users need only download the eclipse plug-ins. Programmers who want to use EODM and/or RStar API need only download the corresponding packages.

A.7 HiSee

hisee.sourceforge.net

HiSee is a tool for visualizing high-dimensional datasets. HiSee projects high-dimensional data to (currently) two dimensions using one of a variety of projection techniques. By comparing the way different projection algorithms present a dataset users can gain qualitative understanding of high-dimensional structures. Consider efforts to project a globe to two dimensions. Even though each projection introduces its own distortions, by comparing them one gets a sense of the three-dimensional structure of our world. HiSee makes use of the Piccolo zoomable user interface (ZUI), which allows users to emphasize and explore specific regions of a dataset. HiSee can be used as a standalone application or as a component in other programs (like Simbrain) which generate high dimensional data in real-time. Algorithms for fast addition of new datapoints are included. The program is written in Java and is open source. We are currently seeking developers to (among other things) contribute new projection algorithms and to implement a three-dimensional view.

A.8 Tom Technology

tom.loria.fr/about.php

> Tom is a pattern matching compiler developed at INRIA. It is particularly well-suited for programming various transformations on trees/terms and XML based documents. Its design follows our research on rule based languages (R3), and our experiences on the efficient compilation of ELAN developed by the Protheo group.

A.9 Judge

www3.dfki.uni-kl.de/judge

> Judge features automatic classification and clustering of documents, optionally as a webservice.
>
> The program is written entirely in Java and makes use of the Weka machine learning toolkit.

A.10 Facet Analytical Theory (FAT)

www.ucl.ac.uk/fatks/

> Facet analysis in a rudimentary form was conceived by S.R. Ranganthan in the 1930s, although it had been preceded by similar analytico-synthetic approaches to subject classification and indexing, notably by Henry Bliss and Paul Otlet in the classification field, and Kaiser in indexing. It was developed post 1950, principally by members of the UK Classification Research Group, as a tool for the organization of document collections in technical, scientific and social scientific fields, where it was highly effective in the storage and retrieval of compound and complex subjects.
>
> Modern facet analytical theory contrasts with earlier views of knowledge as an integral whole (which is broken down into smaller and smaller units) in that it deals with individual terms or concepts which are clustered into categories to create a 'bottom-up' map of knowledge. A number of categories have been identified which are widely applicable to the terminologies of a range of subject fields; these categories are generally functional and/or linguistic in nature (e.g. entities, processes, properties, operations, agents). Compound and complex subjects are accommodated by combining individual concepts. Various forms of system syntax (links and rules for ordering and combination between categories) have been proposed to compound the individual concepts, most of which are based on natural language models; the method used in 'classical' UK facet theory depends on a formulaic inter-category order (citation order).

Currently, facet analysis is used primarily to create classifications for the physical arrangement of documents (or document surrogates). The completed classification or knowledge structure is built up from individual terms which are analysed into categories and ordered by the application of the system syntax. The resultant structure is logical and predictable, and therefore highly effective in storage and retrieval. The only widely used classification embodying these principles is the second edition of the Bliss Bibliographic Classification, which employs standard categories and citation order. It seems probable that this methodology of facet analysis can be used in a broader and more innovative way to create much deeper and more complex knowledge structures and semantic networks, by extending the range of categories and by exploring variants on combinatorial methods. Although the faceted classification is regarded by many as a structure with specific characteristics, essentially facet analysis is a technique, and different models of the same universe of discourse can be derived to meet different local or subject-specific needs using different categories and variations on the syntax.

Some research is now required into the fuller range of categories and relations that may be encountered across the complete range of disciplines. Work is also required on the problems of interdisciplinarity, and alternative approaches to the structuring of knowledge that do not depend on traditional disciplines as the first point of entry; in this area, classifications of phenomena (as opposed to the more conventional aspect classifications), and the further application of integrative level theory, require some investigation. Additional properties of digital objects, especially non-text, multimedia and images, can also provide data for categorical analysis, and may affect the potential syntax of the system. The formulae developed for the combination of terms and concepts will generate n-dimensional structures that seem appropriate to a hypertext environment. Structures generated from the expanded category base may be particularly useful in handling digital objects.

A.11 Prefuse

prefuse.sourceforge.net

the prefuse visualization toolkit

prefuse is a user interface toolkit for building highly interactive visualizations of structured and unstructured data. This includes any form of data that can be represented as a set of entities (or nodes) possibly connected by any number of relations (or edges). Examples of data supported by prefuse include hierarchies (organization charts, taxonomies, file systems), networks (computer networks, social networks, web site linkage) and even non-connected collections of data (timelines, scatterplots). Using this toolkit, developers can create responsive, animated graphical in-

terfaces for visualizing, exploring, and manipulating these various forms of data. prefuse is written in the Java programming language using the Java2D graphics library and is designed to integrate with any application written using the Java Swing user interface library.

A.12 CliniMiner

`www.alphaworks.ibm.com/tech/cliniminer?Open&ca=daw-flnt-060304`

CliniMiner is a demonstration of undirected data-mining method for detecting unexpected relationships in large data sets. This tool discovers and predicts unexpected qualitative and quantitative phenomena in large data sets by unsupervised, that is, undirected, data mining where the combinations of items to examine explodes. By "explodes" is meant that exhaustive combinations of statistical tests or directed queries could take millions, in some case zillions, of years. Such difficulties show up particularly in analysis and use of genomic and clinical data, and they represent a major bottleneck in information-based medicine. Extensive archives of records of just a mere 100 parameters or columns could, in the worst case, contain 10 to the power of 29 combinations, which could appear as significant "rules" in any data-mining output.

In contrast, if one were simply to test a hypothesis in the classical way, that is, if one suspected what was interesting in advance, it could be tested in seconds by statistical tests or tools such as DiscoveryLink; hence, CliniMiner complements these kinds of methods. Pruning heuristics and other shortcuts are possible in CliniMiner and continue to be developed. For example, "rules" are not even explored in CliniMiner if the abundance of the component items or events predicts that the "rules" would have insufficient information content. The output data are estimates, and bias is in favor of not missing a potentially useful discovery, hence classical methods must be applied in order to verify the discoveries.

This tool combines aspects of data mining, information theory, and number theory (and even ultimately quantum theory) that directly address the hot mathematical topic known as "The Theory of Expected Information," more recently referred to as "Zeta Theory" (ZT).

A.13 VisuMap

`www.visumap.net/index.aspx?p=Products`

Note: This is a commercial product.

Dedicated to exploring high dimensional data. Offers the most comprehensive implementation of Relational Perspective Map. Free for no-commercial use.

A.14 Infomap NLP Software

infomap-nlp.sourceforge.net

> The Infomap NLP Software package uses a variant of Latent Semantic Analysis (LSA) on free-text corpora to learn vectors representing the meanings of words in a vector-space known as WordSpace. It indexes the documents in the corpora it processes, and can perform information retrieval and word-word semantic similarity computations using the resulting model.
>
> The Infomap software is implemented in C and can efficiently process large corpora. It has already been used on the British National Corpus; New York Times, AP, and Wall Street Journal newswire corpora; a collection of medical abstracts; the OHSUMED corpus; and other corpora. The software has successfully been compiled and run under Solaris 7 (SunOS 5.7), Red Hat Linux 9, Debian Linux 3.0 ("woody"), and Cygwin. It should work under other Unix variants with minimal adaptation.
>
> Future releases may include a version of the software that can process parallel bilingual corpora and a web interface for convenient information retrieval and query tuning.

A.15 "Fast Parallel Matrix Multiplication - Strategies for Practical Hybrid Algorithms"

`www.f.kth.se/~f95-eeh/exjobb/background.html`

> The aim of the proposed study is to investigate implementations of different algorithms for matrix multiplication (Strassen, Winograd) and their applicability for practical cases. Parallelizing compilers will be used in order to determine near-optimal strategies for serial, vector and parallel computers - including strategies for building hybrids of known algorithms (determination of breakpoints) and study of multiplication of matrices of sizes not a power of 2 (fewer steps of recursion vs. padding-out with zeros). In cases where the full potential speed-up is not realised factors such as loss in data locality/cache effects, recursion overheads, etc. will be studied.

A.16 Other Matrix Algorithms

Courtesy of David Pensak, email, 12 May 2004:

> Abstract. In previous work R. Johnson and A. McLoughlin, by a computer-aided search, found new noncommutative bilinear algorithms for 3?3 matrix multiplication that require only 23 essential multiplications rather than

the 27 required by the conventional method. Such algorithms, like Strassen's algorithm for the 2?2 case, lead to fast algorithms for matrices of arbitrary size. It is proposed to extend the search to obtain improved upper bounds on the number of essential multiplications required for the 3?3 case and other small sizes, in particular 4?4. Implications of the results for the design of special-purpose array-processing hardware will be studied.

Computing the product of two nn matrices X and Y by straightforward evaluation of the defining expression involves multiplying n3 pairs of numbers and performing a proportionate number of other elementary operations, such as additions. A celebrated algorithm of Strassen's [1] shows that we can do substantially better, requiring a number of elementary operations that grows only in proportion to approximately n2.807 as n increases, rather than n3. The advantage grows as n increases. Even Strassen's algorithm is not the best possible, and it remains an important unsolved problem in algebraic computational complexity theory to determine the minimum number of arithmetic operations required for matrix multiplication.

Efficient algorithms for matrix multiplication are of interest not only because matrix multiplication is important in its own right, but because they automatically lead to efficient algorithms for other problems. These include not only such cornerstones of numerical analysis as the solution of systems of linear equations, triangular decomposition of matrices, matrix inversion, and computation of determinants[1, 2], but problems in combinatorics—finding the transitive closure of a relation[3]—and formal language theory—parsing of arbitrary context-free languages[4]. Strassen's algorithm is based on a method for multiplying 22 matrices that (1) uses only 7 multiplications instead of the 8 needed by the conventional method and (2) is noncommutative—i.e. does not require multiplication of matrix elements to be a commutative operation. Consequently it is possible (when n is even) to split nn matrices X and Y into blocks of size (n/2)(n/2) and regard them as 22 matrices, with smaller matrices as entries. Strassen's method computes the product Z = XY by computing the products of 7 pairs of matrices of size (n/2)(n/2) (linear combinations of the X(i, j) or of the Y(i, j)) and combining them to obtain Z. If n is large enough, these 7 smaller matrix products can in turn be computed by recursive application of Strassen"s method. The resulting algorithm runs in time proportional to na, where the exponent a is log27, or about 2.807.

Similar fast algorithms for matrices of arbitrary size can be based on noncommutative procedures for other small sizes than 22. For example, such a procedure for 33 matrices, requiring m multiplications, would lead to an algorithm for general nn matrices with running time proportional to na with an exponent a=log3m, which is an improvement over the conventional method when m ¡ 27 and would beat Strassen if m 21 could be achieved. Similarly a noncommutative algorithm for 44 matrices would yield a general algorithm with exponent a=log4m, which beats the conven-

tional method when m ¡ 64, with m 48 required to beat Strassen. Actual reductions of the exponent below Strassen's value have been achieved by rather different means with the help of several new ideas [5—13], and they leave open the minimum number of multiplications required for relatively small matrices. The algorithms with the best known theoretical exponents, in fact, become effective only for such huge matrices that they have no practical application. It therefore remains of interest to find good bounds on the number of noncommutative multiplications required for small matrix sizes.

The actual best result known for the 33 case is 23 multiplications, achieved by the algorithms discovered by Laderman [14] and by Johnson and McLoughlin [15]. Strassen's result (7multiplications) has been shown to be optimal for the 22 case [16, 17]. The algorithms presented in [15] were discovered by a computer-aided search. It is proposed to extend the search to obtain improved upper bounds on the number of essential multiplications required in algorithms for products of small matrices, in particular the 3?3 and 4?4 cases. The results have potential application not only to general-purpose numerical software but to the design of special-purpose hardware. The trade-offs involved in the two cases are somewhat different. Strassen-like algorithms require more additions than the conventional algorithm when applied directly to the smallest matrices and do not reduce the total operation count until the matrices surpass a certain minimum size. However, while multiplication instructions in modern workstations may be competitive in speed to addition instructions, the circuit complexity of a fast multiplier is considerably greater than that of an adder. The applications of results obtained to the design of special-purpose array-processing hardware will be studied. References

[1] V. Strassen, "Gaussian elimination is not optimal," Numer.Math. 13, 354–356 (1969).

[2] J. Bunch and J. E. Hopcroft, "Triangular factorization and inversion by fast matrix multiplication," Math. Comp. 28 (125), 231–236 (1974).

[3] M. J. Fischer and A. R. Meyer, "Boolean matrix multiplication and transitive closure," Proc. IEEE 12th Ann. Symp. Switching and Automata Theory, 129–131 (1971).

[4] L. G. Valiant, "General context-free recognition in less than cubic time," J. Computer and System Sciences 10 (2), 308–315 (1974).

[5] V. Pan, "Strassen's algorithm is not optimal: trilinear technique of aggregating, uniting and canceling for constructing fast algorithms for matrix operations," Proc. 19th Ann. Symp. on Foundations of Computer Science, 166–176 (Oct. 1978).

[6] V. Pan, "Field extension and trilinear aggregating, uniting, and canceling for the acceleration of matrix multiplications," Proc. 20th Ann. Symp. on Foundations of Computer Science (1979).

[7] V. Pan, "New fast algorithms for matrix operations," SIAM J. Computing 9, 321–342 (1980).

[8] D. Bini, M. Capovani, F. Romani, and G. Lotti, "O(n2.7799) complexity for n?n approximate matrix multiplication," Inform. Proc. Lett. 8, 234–235 (1979).

[9] D. Bini, G. Lotti, and F. Romani, "Approximate solutions for the bilinear form computational problem," SIAM J. Computing 9, 692–697 (1980).

[10] A. Schönhage, "Partial and total matrix multiplication," SIAM J. Computing 10, 434–455 (1981).

[11] D. Coppersmith and S. Winograd, "On the asymptotic complexity of matrix multiplication," SIAM J. Computing 11, 472–492 (1982).

[12]V. Pan, How to Multiply Matrices Faster, Lecture Notes in Computer Science 179, Springer-Verlag, 1984.

[13] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," J.Symbolic Computation 9, 251–280, 1990.

[14] J. Laderman, "A noncommutative algorithm for multiplying 33 matrices using 23 multiplications," Bull. Amer. Math. Soc. 82, 126–128 (1976).

[15] R. W. Johnson and A. M. McLoughlin, "Noncommutative bilinear algorithms for 33 matrix multiplication," SIAM J. Computing 15 (2), 595–603 (1986).

[16] J. E. Hopcroft and L. R. Kerr, "On minimizing the number of multiplications necessary for matrix multiplication," SIAM J. Appl. Math. 20, 30–36 (1971).

[17] S. Winograd, "On multiplication of 22 matrices," Linear Algebra and Appl. 4, 381–388 (1971).

A.17 "Reconstruction of organisational phylogeny from memetic similarity analysis: Proof of feasibility"

`jom-emit.cfpm.org/2001/vol5/lord_a&price_i.html`

Courtesy of David Pensak.

Abstract

A successful phylogenic reconstruction of the known pattern of descent of the main post-reformation Christian Churches has been achieved from a computerised analysis of aspects of their present day memetic pattern. The result confirms the feasibility of a new approach to organisational memetics and conducts a first empirical test of the hypothesis that, if organisations are construed as evolving, those with a common ancestor should show greater replicator similarity than more distant relatives.

`$Id: open-sources.tex,v 1.2 2004/10/02 21:45:53 sok Exp $`

Appendix B

Example List of Product Categories

From the downloaded file `Other_code.doc`, a list of product categories.

```
Accident and health insurance services
Accident insurance services
Aerial and related services
Aerial forest-firefighting services
Aerial spraying services
Air-charter services
Air-rescue services
Aircraft insurance services
Aircraft liability insurance services
Aircraft operating services
Aircraft-operation services
All-risk insurance services
Aluminium ores
Anti-pollution ship services
Antisera
Auxiliary insurance services
Average adjustment services
Bags for urine
Bathrobes
Beach cleaning services
Bituminous or oil shale
Blood bags
Blood-grouping reagents
Blood-testing reagents
Bone reconstruction cements
Bovine ear tags
Brake fluids
```

Cable-laying machinery
CAT scanners
Chemical contraceptives
Chemical reagents
Chick peas
Chippers
Civil engineering machinery
Clinical products
Coffins
Colour-flow doppler
Compacting machinery
Compression equipment
Computer servers
Construction machinery
Construction management services
Construction project management services
Construction work for elevated highways
Contraceptives
Contractor’s all-risk insurance services
Cooling equipment
Copper ores
Coveralls
Credit and surety insurance services
Credit insurance services
CT scanners
Damage or loss insurance services
Data management services
Data network management services
Decoration works
Dermatological devices
Dialysis solutions
Diesel fuel
Diesel oil
Doppler equipment
Dresses
Dried peas
Drilling cement
Drinking-water distribution
Echocardiographs
Echoencephalographs
Ejector seats
Enema preparations
Engineering insurance services
Enteral feeds
Equipment sets
Exterior cleaning work for buildings

File servers
Financial consultancy services
Financial loss insurance services
Financial markets administration services
Fire insurance services
Fire-brigade uniforms
Flatwork for cemeteries
Flower seeds
Flying footwear
Fuel oils
Fungicides
Galenical solutions
Gas oils
General liability insurance services
Geological equipment
Glands and their extracts
Glucose solutions
Gravel
Grounds maintenance services
Hard plastic containers
Health insurance services
Heating oil
Heavy-lift ship services
Hepatitis B vaccines
Highways consultancy services
Highways engineering services
Hydrated lime
Hydraulic equipment
Hydraulic installations
Immunoglobulins
Industrial clothing
Industrial equipment
Infusion solutions
Injectable solutions
Installation services of agricultural and forestry machinery
Installation services of agricultural machinery
Installation services of beverage-processing machinery
Installation services of clothing-production machinery
Installation services of construction machinery
Installation services of cranes
Installation services of food-, beverage- and tobacco-processing machinery
Installation services of food-processing machinery
Installation services of forestry machinery
Installation services of handling equipment
Installation services of leather-production machinery
Installation services of lifting equipment

Installation services of machine tools
Installation services of metallurgy machinery
Installation services of mining machinery
Installation services of mining, quarrying, construction and metallurgy mach
Installation services of paper- and paperboard-production machinery
Installation services of paper-production machinery
Installation services of paperboard-production machinery
Installation services of quarrying machinery
Installation services of special-purpose machinery and equipment
Installation services of textile-, clothing- and leather-production machiner
Installation services of textile-production machinery
Installation services of tobacco-processing machinery
Installation services of weapon systems
Insulin
Insurance and pension consultancy services
Insurance consultancy services
Insurance services relating to transport
Intravenous fluids
Isotopic reagents
Jiffy bags
Laboratory reagents
Landscaping work
Lead ores
Legal expenses insurance services
Legal insurance services
Lentils
Liability insurance services
Life insurance services
Magnetic resonance imaging equipment
Mail delivery services
Mechanical rollers
Medical clothing
Medical insurance services
Medical solutions
Microbiological cultures
Military clothing
Military uniforms
Motor vehicle insurance services
Motor vehicle liability insurance services
Motorway service area
Multi-functional equipment
Mustard seeds
Network management software services
Network servers
Nickel ores
Nicotine substitutes

Nightdresses
Non-life insurance services
Non-residential property renting or leasing services
Occupational clothing
Offshore supply ship services
Oil country tubular goods
Operation of an educational centre
Oral contraceptives
Ornamentation fitting work
Overhaul and refurbishment work
Panties
Paper for newsprint
Parcel delivery services
Parenteral feeding solutions
Parenteral nutrition products
Parking meters
Parts of steam turbines
Paving machinery
Pebbles
Pecuniary loss insurance services
Pension fund consultancy services
Pension fund management services
Pension investment services
Peptic substances
Perfusion solutions
Photographic flashbulbs
Photographic flashcubes
Pile drivers
Pile extractors
Pilot services
Planers
Plasma bags
Plastic barrels
Plastic bottle cases
Plastic caps
Plastic carboys, bottles and flasks
Plastic cases
Plastic crates
Plastic lids
Plastic medical bags
Plastic refuse containers
Plastic sacks and bags
Plastic spools or bobbins
Plastic stoppers
Plastic stoppers, lids and caps
Plastic waste sacks

Police uniforms
Polio vaccines
Polystyrene products
Polystyrene sheeting
Polystyrene slabs
Polythene medical bags
Polythene sacks and bags
Polythene storage bags
Polythene waste and refuse sacks and bags
Powdered lime
Precious-metal ores
Printer servers
Process timers
Products of wood and related articles
Property insurance services
Provitamins
Pulses
Railway insurance services
Reagents for electrophoresis
Refurbishment work
Repair and maintenance services of central heating
Repair and maintenance services of cooler groups
Residential property renting or leasing services
Rig-jacking systems
Risk management insurance services
Road rollers
Road-surfacing machinery
Safe-deposit boxes
Safety cases
Salicylic acids
Seismic equipment
Service area
Sesame seeds
Sewage-treatment consultancy services
Shorts
Skirts
Sleeving repair services
Snowblowers
Snowploughs
Snowploughs and snowblowers
Standby ship services
Steam turbines
Sugar syrups
Switching station installation work
Taxi services
Theatre seats

Time recorders
Time registers
Time switches
Tin ores
Token meters
Tool parts of wood
Topography equipment
Toxins
Transport of radioactive waste
Trousers
Typhus vaccines
Ultrasound scanners
Uniforms
Unleaded petrol
Urological reagents
Vaccines
Vaccines for veterinary medicine
Vessel insurance services
Vessel liability insurance services
Vitamins
Voluntary health insurance services
Wall lights
Weather and financial loss insurance services
Weather-related insurance services
White spirit
Wooden frames
Wooden tool handles
Wooden tool holders
X-ray contrast media
Zinc ores

$Id: other-code.tex,v 1.3 2004/08/13 00:20:55 sok Exp $

Appendix C

Information Sources

C.1 Information on Text Mining

1. http://www.cs.utexas.edu/users/pebronia/text-mining/
2. http://filebox.vt.edu/users/wfan/text_mining.html
3. http://www.text-mining.org/

$Id: info-sources.tex,v 1.1 2004/10/02 21:45:53 sok Exp $

Appendix D

A Note on Automated Support for Product Application Discovery

This appendix was written by Gary Chen, Steven Kimbrough, and Thomas Lee, 14 August 2004.

* * *

We report in what follows on an exploratory study in knowledge extraction from text. Our discussion begins in §D.1 with a description of the particular application context—computational support for product application discovery—that directly motivated this work. Following that we present and discuss our experimental setup, §D.2, and our findings, §D.3. We conclude in §D.4 with a discussion of related work, of the larger significance of this investigation (and others of its kind), and of future avenues of investigation.

D.1 Context and Motivation

For several years, researchers at the Sol C. Snider Center for Entrepreneurial Studies at The Wharton School have, in conjunction with industrial sponsors, conducted Product Application Discovery (PAD) exercises. The goal of such an exercise is to find promising new (heretofore unknown) potential uses for a given product. For example, the Snider Center was given the challenging task of finding new uses of polytetrafluoroethylene (PTFE, aka: Teflon). Perhaps surprisingly, since Teflon has been on the market for 30 years, more than a dozen new uses were identified.

It is believed that these exercises—PAD exercises—are potentially of enormous value because much intellectual property (IP) goes under- (or even un-) utilized for lack of appreciation of how it might be used. That new uses of Teflon may be discovered

today supports this widely-held belief. The studies at the Snider Center have produced results in this regard that are consistent with experiences in the small industry that provides consulting services to companies wishing to discover new uses for their IP.

A common theme among those who approach this problem is that a great portion of the relevant electronic information for PAD studies is in textual form and hence results in a labor-intensive process to extract the requisite information. The motivation for the work we report here is to provide computerized support that reduces the amount of labor involved in conducting PAD studies. We focused on just one step in the process, which we shall now explain.

An analyst working on a PAD exercise will employ a number of heuristics, aiming at finding new uses for a target product, call it Q. Important among these heuristics is the following *Uses of Similar Products* heuristic: *If product Q is like product P (in the right ways) and P is used for X, then plausibly Q may be used for X*. In practice, the patent textbase maintained by the United States Patent and Trademark Office (www.uspto.gov) is widely relied upon in deploying the Uses of Similar Products heuristic. The analyst obtains a set of terms describing (in the right way, i.e., according to a working hypothesis about what is important or promising) the target product, Q. These in hand, the analyst uses a search engine (one is provided online at www.uspto.gov, but there are others) to find similar patents, i.e., patents whose texts match the descriptive terms. Knowing that patents typically mention potential uses for the patented idea, the analyst reads the best-ranked patents and extracts the uses mentioned in them that seem most promising for Q.

This very labor-intensive process would be greatly speeded up if the analyst were asked to read a list of potential uses, rather than a list of entire patents. Such a list of potential uses need not be absolutely accurate. It may contain a fair number of false positives (i.e., have a *precision*—the fraction of true positives—considerably less than 1 and still be useful). Further, such a list need not be exhaustive (or complete). A level of *recall*—the fraction of uses mentioned in the source documents actually listed for the analyst—considerably below 1 may well be useful. The alternative of reading thoroughly a short list of documents is almost certain to have a low recall.

These assertions are plausible, but of course need to be thoroughly investigated and tested. Investigations, however, must begin somewhere. Our aims in this exploratory study were limited to determining whether it is possible to automatically extract a list of mentioned uses from a collection of patents. Until this is established there is little prospect for determining, e.g., how valuable the lists extracted are. We now turn to a discussion of our methods and findings.

D.2 Setup

D.2.1 Extraction Technique

Our approach relies on using supervised learning for discovering how to extract product use passages. Human coders (undergraduates employed by the Snider Center) were asked to extract manually (with a text editor) from patents passages that mention uses of the product being patented. 24 patents were coded by at least two different coders.

Fix /* here */.

Figure 2 /* here */

The patents coded were drawn arbitrarily (in sequence from a random start) but limited to *device* patents. We define a device patent as a patent on any machine, article of manufacture, composition of matter, or variety of plant which may be new or an improvement upon prior art. This definition encompasses a subset of terms from the traditional definition of a utility patent—namely we exclude patents on manufacturing and business processes(such as Amazon's one-click shopping patent) due to low inter-annotator agreement among uses found in such patents.

In all, 50 device patents were coded and the use passages copied to files. From these files we assembled a 'gold standard' list of human-recognized mentioned uses for each patent. The gold standard consist of sentence fragments that capture a use passage. For example, "for drying semiconductor wafers" is considered a use passage.

Figure 1: Main procedure

We developed and refined extraction rules for identifying general delimiters of product use passages. We began by populating a list with an initial set of delimiters (Fig.2). Using this initial list, use passages are extracted from the patent. These passages are then compared with our gold standard set of identified uses (Fig.3). A passage is considered a use if it matches a gold standard use based upon a parameterized comparison function (EVALUATE() in Fig.3). The metrics used for matching will be elaborated upon in the next section. Once the comparisons are done, the extraction rules are refined, and the process is repeated until rules cannot be further improved (Fig.1). Currently, there are only two possible rules for ending delimiters: extraction up to a trailing comma or to the end of a sentence (EOS). Our experiments showed that this simple approach is a reasonable heuristic for delimiting end points of stated product uses in a patent.

We used one approach along with three possible variations for discovering feature sets upon which we can derive extraction rules for delimiters. The first extraction technique is the most basic: simply extract text portions containing keywords which may delimit a product use passage. For example, phrases of the form <for use with> ... <EOS> may serve as delimiters for a possible stated product use where "for use with" is a start marker and EOS is an end point for extraction. This method entails keeping a keyword list. In our implementation, this consists of a list of two or three word phrases which begin patent use passages in our gold standard set of identified uses. A stoplist was used as a pre-processing step for filtering out uninteresting candidate delimiters and the resulting set was further pruned by an expert. Taking a bottom-up approach for refining this keyword list, we started from a specific set of keywords and tried to generalize them as much as possible. Delimiter phrases of the same word length containing one or more same-sequence, common keywords were collapsed together into a new delimiter phrase consisting of the sequence of those common

Figure 3 /* here */

keywords. For example, phrases such as “for use of”, “for use with”, and “for use as” were collapsed into a two-keyword phrase “for use” (Fig.3). At this point, the keyword list with the changes was evaluated over a group of patents to see how well it scored. If the new keyword list scored significantly worse than before, the changes were undone. Otherwise they were kept and the next set of changes was applied until no more changes could be made (Fig.1). Evaluation was done using the F-measure[CD95][Car97b] with $F = \frac{2*Precision*Recall}{Precision+Recall}$, $\alpha = 0.5$. We used a sentence boundary detector[RR97] for determining EOS periods.

There are three variations of this approach. One is to look only at phrases consisting of part-of-speech (POS) tags, discarding the underlying words. Collapsing phrases with a common sequence of POS-tags however is very difficult because it greatly increases the number of extracted use passages. A second variation is to develop a list of delimiter phrases consisting of both the keyword and its corresponding POS tag. Both of these variations can be accomplished simply by repeating the steps as above, but with a POS-tagged dataset. The last variation is to look at developing delimiters consisting of combinations of keywords and part-of-speech tags. An example of such a set of rules is given in Tables 1-3.

Table D.1: Rule List:Initial delimiters

Token#1	Token#2	Token#3
for/IN	supplying/VBG	a/DT
for/IN	separating/VBG	the/DT
for/IN	restricting/VBG	the/DT
for/IN	receiving/VBG	an/DT
for/IN	moving/VBG	a/DT

Table D.2: Rule List: One iteration of rule refinement (see Fig.3)

Token#1	Token#2	Token#3
for/IN	/VBG	the/DT
for/IN	/VBG	an/DT
for/IN	/VBG	a/DT

Table D.3: Rule List: Second iteration of rule refinement

Token#1	Token#2	Token#3
for/IN	/VBG	/DT

The substring to the left of the slash represents the raw word while the substring to the right indicates the part-of-speech from which the underlying word’s context was

drawn. Iterations of rule refinement are shown in Tables 2 and 3. Table 1 shows the original list of delimiter extraction rules. The first round of refinement (Table 2) collapses the initial list of five rules down to three by generalizing on the second token's present participle verb form (VBG). The second round of refinement (Table 3) further collapses these set of rules based upon the third token's determiner (DT) POS tag. These rule list modifications are kept or discarded based upon the resulting matches found between the extracted and gold standard uses for a training set of data. Note that unlike keyword refinement, we do not shorten the pattern length. We only collapse rules by generalizing on token keyword or its corresponding part of speech.

D.2.2 Metrics for Matching

We used three metrics for evaluating matches between extracted and gold standard use passages. The first is containment. If the extracted use contains or is contained in an identified use passage, then that extracted use is considered to match that gold standard passage. Containment allows the evaluation of whether a use passage has been extracted within the proximity of a location from which a hand-labeled use has been found. The second is the standard cosine similarity metric used in information retrieval. Each use represents a "document" of terms. For each extracted use, the cosine similarity score is the maximum score it gets comparing with all gold standard uses for a particular patent. If the maximum score is more than a threshold similarity value, the extracted phrase is considered correct. The third metric is edit distance. The edit distance of two strings is defined as the minimum number of point mutations required to change one string into another. Point mutations consist of inserting, changing, or deleting a character. For each extracted use, the edit distance is the minimum score it gets when compared with all gold standard uses for a particular patent. If the score is less than a threshold edit distance, the extracted use is checked off as a match. Both cosine similarity and edit distance allow us to see to what degree do extracted and gold standard passages match, with cosine and edit distance looking at term and character similarity respectively. We do not need to be able to extract use strings that are exactly identical to gold standard passages. Instead, using soft metrics of match is more appropriate for our purpose. We present results for each of these measures in our tests.

D.3 Experimental Results

We divided the set of 50 patents into two groups: a *Training group* for generating the rule lists and a *hold out set* for evaluation purposes. Each group consisted of 25 patents, randomly assigned. We did training by hand, walking through the process of initial rule population and refinement, then we ran automated tests on the keyword approach and all its variations were evaluated based upon each of the three metrics. We used a cosine similarity threshold of 0.8 and an edit distance threshold of 50 mutations. The results are summarized in Tables 4 and 5.

Table D.4: Training Results: 25 Patents

Approach	Number of Extracted Uses		Metrics containment	cosine similarity	edit distance
keyword list	787	Precision	0.56290	0.36976	0.46633
		Recall	0.95887	0.62987	0.79437
		F-measure	0.70937	0.46597	0.58767
POS tag list	18234	Precision	0.02534	0.01722	0.02402
		Recall	1.00000	0.67965	0.94805
		F-measure	0.04942	0.03359	0.04685
keyword/ POS tag list	784	Precision	0.57270	0.37500	0.47449
		Recall	0.97186	0.63636	0.80519
		F-measure	0.72071	0.47191	0.59711
combination keyword/POS tag list	867	Precision	0.51211	0.33910	0.44406
		Recall	0.96104	0.63636	0.83333
		F-measure	0.66817	0.44244	0.57938

Table D.5: Testing Results: 25 patents

Approach	Number of Extracted Uses		Metrics		
			containment	cosine similarity	edit distance
keyword list	157	Precision	0.37580	0.22293	0.33758
		Recall	0.11943	0.07085	0.10729
		F-measure	0.18126	0.10753	0.16283
POS tag list	14994	Precision	0.03188	0.01821	0.03108
		Recall	0.96761	0.55263	0.94332
		F-measure	0.06173	0.03525	0.06018
keyword/ POS tag list	167	Precision	0.39521	0.23353	0.35329
		Recall	0.13360	0.07895	0.11943
		F-measure	0.19970	0.11800	0.17852
combination keyword/POS tag list	718	Precision	0.50121	0.23729	0.36562
		Recall	0.41903	0.19838	0.30567
		F-measure	0.45645	0.21610	0.33297

Table 4 shows results from the training data after refinement on the rule list. Rules in the list consists of two or three token phrases with restrictions on the token type based upon the extraction approach. Tokens were restricted to words for the keyword list (as in "used" or "of"), POS tags for the POS tag list (such as "/VBN" or "/IN"), or keywords attached with their corresponding POS tags for the keyword/POS tag list (as in "used/VBN" or "of/IN"). Combination keyword/POS tag lists do not have any token restrictions and can have tokens consisting of any of the above types. The training set had 462 gold standard uses. The metrics of containment, cosine similarity, and edit distance are displayed with calculations of precision, recall, and F-measure based upon each metric's criterion for match. As can be seen, training on the keyword or keyword/POS tag list results in a higher F-value (0.70937 and 0.72071 respectively) than on combination lists(0.66817). Recall is high because rules that only match one or two particular passages in the training set are not removed from the rule list. Also while the POS tag list has a recall of 1.0, it extracts over 18,000 uses. Table 5 shows results from testing on rule lists developed from the training data. A total of 25 patents containing 494 gold standard uses exists for the hold out set.

Overall it appears from this one sample that the fourth method—combination keyword/POS tag list—performs best. We undertook a cross-validation study of that method, redoing the analysis with 5 additional different random divisions of the 50 patents into training and hold sets. The results appear in Table D.6 and conform well with the original findings.

Table D.6: Summary data for cross-validation study of combination keyword/POS tag list: means and (standard deviations) for 5 different random samples from 50 patents.

	Training Set			Testing (Hold Out) Set		
	Containment	Cosine	Edit	Containment	Cosine	Edit
precision	0.46066 (0.01436)	0.28956 (0.02227)	0.39041 (0.02258)	0.42148 (0.07617)	0.23119 (0.02553)	0.33747 (0.02933)
recall	0.97560 (0.01227)	0.56191 (0.10979)	0.75718 (0.13842)	0.44908 (0.05144)	0.25220 (0.04828)	0.36987 (0.07703)
F-measure	0.62564 (0.01172)	0.37823 (0.03052)	0.50988 (0.02940)	0.42751 (0.01920)	0.23714 (0.01481)	0.34696 (0.02456)

D.4 Discussion

Although the number of patents used in this study—50—is small, we believe the results are quite encouraging. With more time and effort (especially by the coders) this number will be increased several fold. Further, in a practical setting, coding and subsequent recalibration is something that could be undertaken on a continuing, non-burdensome basis. Precision (the rate of true positives) should be improvable by reduction in false positives during post-processing. This is itself a candidate for supervised learning methods. Having a hold out set of 50% is quite conservative and the cross-validation results are reassuring. All these are reasons for thinking that the results as indicated here will hold up and even be improved without improvements in our learning algorithm. Add in that and, again, we are much encouraged.

We note that in Table D.6 the best recall and precision numbers are slightly in excess of 40%. As just remarked, we think there is good reason to believe these numbers can be improved. Even without improvement, however, these numbers are very attractive. To see why, consider the context. The relevant alternative is to use the time of an analyst to read the patents and recover the mentioned uses. On the assumption that the time and money available to read these patents would cover only a small fraction of the interesting patents, then "reading" the remaining large majority and recovering only 40% of the mentioned uses at minimal labor cost is a very attractive option indeed. Again, this is an exploratory study and very much remains to be tested and measured. What can be seen on the 'back of an envelope' is, however, most attractive.

D.4.1 Related Work

Our approach is innovative and differs in detail from previously-reported methods. There are, however, two bodies of research containing work of a similar kind. In this section we briefly review this work. One of the two bodies of related research is the area of information extraction (IE). One view of IE is that specifying the data to be extracted is equivalent to defining a template listing slots to be filled by phrases taken from the text. For example, one template may be the following: **Last Name**: Smith. **First Name**: Joe. **Occupation**: Programmer, where Smith, Joe, and Programmer are words extracted from a document. Several IE systems have been developed that leverage shallow grammatical features and machine learning techniques to develop and refine rules for phrase extraction from a document[Cir03][FK04][CM98]. One of the earliest rule-based IE systems, RAPIER[CM98] works by learning to identify relevant strings in their entirety. RAPIER looks at specific string examples and attempts to generalize these entire strings using the underlying word, a part-of-speech (POS) tagger, and a semantic classifier (WordNet[Fel98]). LP^2 [Cir03] is another rule-based IE system that considers recognization of starting and ending delimiters for extraction as separate tagging problems and uses several features for extraction. These features include the tokens, POS information, a semantic classifier, a lemmatiser, and a case checker (whether or not the first letter of the word is capitalized). Our work also utilizes a rule-based system for extraction but with a different agenda. The rule-based IE systems described above are considered primarily for the problem of template instantiation. That is, given a template, the purpose of these systems is to fill in this template with key phrases as

accurately as possible. While our work is related, getting exact product use phrases of of a patent is not of as much concern as being able to extract a general passage relating to product use for our purposes. Our work also shows that using just two features, the token and part of speech information, our system performs at a reasonable level (with respect to precision and recall) for extracting use passages from patent text. Other work in IE is concerned with aggregating and developing text summaries over multiple documents [Gol99]. While our work is related to summarization, we are only interested in finding and listing product use passages and do not attempt to rewrite the original text nor do we do any sort of summarization over multiple patent documents.

Existing research involving the patent data set either leverages structured fields like patent classifications and inventor details or weighing and filtering of search terms for the purpose of patent retrieval [McL00][IMHH00][IMO03]. Other research in the patent domain has focused on extracting the 'problem solved' for the purpose of identifying the improvements that a patent makes over prior art [WP03b][WP03a]. However, to our knowledge, extracting usage information from patents to support product application discovery is a novel research problem.

D.4.2 Future Work

Currently, we extract from a start delimiter up to either a comma or an end-of-sentence period. Thus, one area for work is to develop extraction techniques for both start and end delimiters. Other work includes investigating additional features upon which to build the rule list. This includes adding more base features such as other shallow NLP processing tools as well as finding derived features from which to create rules on. A derived feature is a feature developed based upon some combination of base features. Data mining techniques can perhaps be utilized for finding these patterns from the base set of attributes. Another problem is that the updating procedure for rule refinement follows a greedy approach. Each candidate change is applied one at a time and if there is improvement, we keep the change. If there are interaction effects among the rule list updates, the ordering of the updates on the stack may influence the outcome of the final list. Also training requires large amounts of time because we do not parallelize the exploration of multiple solutions (the incremental changes or mutations to the rule list). Thus one possible area is to look into the application of genetic algorithms for optimizing refinement of these extraction rule lists. One last area is finding failure of transitivity patterns in product uses found in patents. This heuristic has been proposed in the past for drug placement[Swa91][WKB01], a subset of the product application discovery problem. A set of products with similar usages may discuss about topics A and B. Another set of products with similar usages may talk about topics B and C. However, there may not be products with similar usages that link topics A and C. This failure of transitivity may indicate a potentially novel new usage for the product in mind. Using IR-based similarity metrics for comparing and clustering product uses we can again have human experts see if patents classified as similar are actually similar to our target product as well as have domain experts verify the novelty of uses mined for the target product.

Finally, we note that there are two attractive kinds of prospects for extending this work. First, supporting PAD exercises in finding new uses of products by applying

these methods to documents other than patents, for example customer reviews of products, general Web documents, and technical reports in proprietary libraries. Second, finding *uses* may be generalized. Surely in many applications it will prove valuable to find *misuses*, *diagnoses*, *problems* and so on. As topics of interest are identified, they may be treated with (extensions of) the methods we have begun to explore here.

Acknowledgements

This work would not have been possible without the help of our coders Charles Han, Max Kimbrough, Matt McGrath, and their supervisor James Thompson. Special thanks to Professor Ian McMillan, director of the Snider Center.

Bibliography

[Bar03] Albert-László Barabási, *Linked: How everything is connected to everything else and what it means*, Plume Books, New York, NY, 2003.

[BCL02] Dario Benedetto, Emanuele Caglioti, and Vittorio Loreto, *Language trees and zipping*, Physical Review Letters **88** (2002), no. 4, 048702–1–048702–4, Also at http://arxiv.org/abs/cond-mat/0108530.

[BK02] David C. Blair and Steven O. Kimbrough, *Exemplary documents: a foundation for information retrieval design*, Information Processing and Management **38** (2002), no. 3, 363–379.

[Bla74] David C. Blair, *An extensional semantic analysis of document indexing*, University of California at Berkeley working paper, December 1974, 27 pages.

[BLM03] Charles H. Bennett, Ming Li, and Bin Ma, *Chain letters and evolutionary histories*, Scientific American (2003), 76–81.

[Car97a] Claire Cardie, *Empirical methods in information extraction*, AI Magazine **18** (1997), no. 4, 65–79.

[Car97b] ———, *Empirical methods in information extraction*, AI Magazine **18** (1997), no. 4, 65–80.

[CD95] N. Chinchor and G. Dungca, *Four scores and seven years ago: The scoring method for muc-6*, Proceedings of the MUC-6 Conference, 1995, pp. 33–38 and 293–316.

[CD03] Tara Calishain and Rael Dornfest, *Google hacks: 100 industrial-strength tips & tools*, first ed., O'Reilly, Sebastopol, CA, February 2003.

[Cir03] Fabio Ciravegna, $(LP)^2$*: Rule induction for information extraction using linguistic constraints*, Technical Report CS-03-07, University of Sheffield, 2003.

[CM98] M. E. Califf and R. J. Mooney, *Relational learning of pattern-match rules for information extraction*, Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing (Menlo Park, CA), AAAI Press, 1998, pp. 6–11.

[CV04] Rudi Cilibrasi and Paul Vitanyi, *Clustering by compression*, arXiv.org e-Print archive, posted on the World Wide Web, 9 April 2004., http://arxiv.org/abs/cs.CV/0312044. Accessed 14 July 2004.

[DDF+90] Scott Deerwester, Susan T. Dumais, George W. Furnas, Ghomas K. Landauer, and Richard Harshman, *Indexing by latent semantic analysis*, Journal of the American Society for Information Science **41** (1990), no. 6, 391–407.

[DE04] Meagan C. Dietz and Jeffrey J. Elton, *Getting more from intellectual property*, The McKinsey Quarterly (2004), no. 4, online, http://www.mckinseyquarterly.com/.

[DHK+97] Garett O. Dworman, Abeer Y. Hoque, Steven O. Kimbrough, Stephen E. Kirk, and Jim R. Oliver, *Report on an experiment on picture retrieval using the DCB algorithm, the PIRS software and pictures from the Clarence Laughlin Archives at The Historic New Orleans Collection*, Technical report: working paper series, University of Pennsylvania, Department of Operations and Information Management, 3620 Locust Walk, Philadelphia, PA 19104-6366, 1997, File: PIRS Exp. 7/21/97.

[DKKO97] Garett O. Dworman, Steven O. Kimbrough, Stephen E. Kirk, and Jim R. Oliver, *On relevance and two aspects of the organizational memory problem*, Technical report, available from Steven O. Kimbrough. Also, Department of Operations and Information Management working paper, University of Pennsylvania, University of Pennsylvania, Philadelphia, PA 19104-6366, 15 December 1997, File: orgmems5.tex/pdf.

[DKP00] Garett O. Dworman, Steven O. Kimbrough, and Chuck Patch, *On pattern-directed search of archives and collections*, Journal of the American Society for Information Science **51** (2000), no. 1, 14–23, File: http://grace.wharton.upenn.edu/∼sok/sokpapers/1998-9/mw99/mw99-jasis-19990514.doc.

[Dum03] Susan Dumais, *Data-driven approaches to information access*, Cognitive Science **27** (2003), no. 3, 491–524.

[Dwo99a] Garett O. Dworman, *Pattern-oriented access to document collections*, Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, 1999, Available as a working paper, Department of Operations and Information Management, [Dwo99b].

[Dwo99b] ———, *Pattern-oriented access to document collections*, Working paper 99-12-20, University of Pennsylvania, Department of Operations and Information Management, Philadelphia, PA, December 1999.

[ESV02] Jeffrey J. Elton, Baiju R. Shah, and John N. Voyzey, *Intellectual property: Partnering for profit*, The McKinsey Quarterly (2002), online, http://www.mckinseyquarterly.com/.

[Fel98] C. Fellbaum, *Wordnet an electronic lexical database*, 1998.

[FK04] Aidan Finn and Nicholas Kushmerick, *Information extraction by convergent boundary classification*, Workshop on Adaptive Text Extraction and Mining, AAAI, 2004.

[Fos82] Antony Charles Foskett, *The subject approach to information*, 4th ed., Linnet Books, Hamden, CT, 1982.

[GD98] Michael D. Gordon and Susan Dumais, *Using latent semantic indexing for literature based discovery*, JASIS (Journal of the American Society for Information Science) **49** (1998), no. 8, 674–685.

[Gol99] J. Goldstein, *Automatic text summarization of multiple documents*, 1999.

[Goo02] Joshua Goodman, *Extended comment on language trees and zipping*, arXiv.org e-Print archive, posted on the World Wide Web, 22 February 2002., http://arxiv.org/abs/cond-mat/0202383. Accessed 14 July 2004.

[IMHH00] Naomi Inoue, Kazunori Matsumoto, Keiichirou Hoashi, and Kasuo Hashimoto, *Patent retrieval system using document filtering techniques*, Proceedings of the Workshop on Patent Retrieval, ACM SIGIR, 2000.

[IMO03] Hideo Itoh, Hiroko Mano, and Yasushi Ogawa, *Term distillation in patent retrieval*, Workshop on Patent Corpus Processing, ACL, 2003.

[JM02] Peter Jackson and Isabelle Moulinier, *Natural language processing for online applications: Text retrieval, extraction and categorization*, John Benjamins Publishing Company, Amsterdam, The Netherlands and Philadelphia, USA, 2002.

[KO94] Steven O. Kimbrough and Jim R. Oliver, *On relevance and two aspects of the corporate memory problem*, Proceedings of the Fourth Annual Workshop on Information Technologies and Systems (Prabuddha De and Clarson Woo, eds.), [na], 1994, File: Oliver-sok-WITS94-r3. Also, available in the working paper series: University of Pennsylvania, Department of Operations and Information Management, 3620 Locust Walk, Suite 1300, Philadelphia, PA 19104-6366., pp. 302–311.

[LCL+03] Ming Li, Xin Chen, Xin Li, Bin Ma, and Paul Vitányi, *The similarity metric*, Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms, ACM, 2003, `http://portal.acm.org/citation.cfm?id=644253&dl=ACM&coll=portal`, pp. 863–872.

[LD97] Thomas K. Landauer and Susan T. Dumais, *A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge*, Psychological Review **104** (1997), no. 2, 211–240.

[LV97] Ming Li and Paul Vitányi, *An introduction to kolmogorov complexity and its applications*, second ed., Springer, New York, NY, 1997.

[McL00] A.W. McLean, *Patent space visualisation for patent retrieval*, Workshop on Patent Retrieval, ACM SIGIR, 2000.

[New03] M.E.J. Newman, *The structure and function of complex networks*, SIAM Review **45** (2003), no. 2, 167–256.

[Rij79] C. J. van Rijsbergen, *Information retrieval*, Butterworths, 1979.

[RR97] J. Reynar and A. Ratnaparkhi, *A maximum entropy approach to identifying sentence boundaries*, 1997.

[SM83] Gerard Salton and Michael J. McGill, *Introduction to modern information retrieval*, McGraw-Hill Book Company, 1983.

[Swa91] D.R. Swanson, *Complementary structures in disjoint scientific literatures*, International ACM SIGIR Conference on Research and Development in Information Retrieval, 1991, pp. 280–289.

[Tay00] Arlene G. Taylor, *Wynar's introduction to cataloging and classification*, 9th ed., Library and Information Science Text Series, Libraries Unlimited, Englewood, CO, 2000.

[Tri02] Anthony J. Trippe, *Patinformatics: Identifying haystacks from space*, Searcher **10** (2002), no. 9, online, http://www.infotoday.com/searcher/default.shtml, then search, e.g., using 'Trippe'.

[WKB01] M. Weeber, H. Klein, and L. Berg, *Using concepts in literature-based discovery: Simulating swanson's raynaud-fish oil and migraine-magnesium discoveries*, Journal of the American Society for Information Science and Technology **52** (2001), no. 7, 548–557.

[WP03a] Tianhao Wu and William M. Pottenger, *A semi-supervised algorithm for pattern discovery in information extraction from textual data*, Proceedings of the Seventh Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-03), 2003.

[WP03b] ———, *A supervised learning algorithm for information extraction from textual data*, Proceedings of the Workshop on Text Mining, Third SIAM International Conference on Data Mining, 2003.

Index