EXTRA! EXTRA!

ALL THE SIDES TO EVERY STORY

JONATHAN JENNINGS HARRIS

DEPARTMENT OF COMPUTER SCIENCE

A SENIOR THESIS
SUBMITTED IN PARTIAL FULFILLMENT OF
THE GRADUATION REQUIREMENTS OF
PRINCETON UNIVERSITY

ADVISED BY BRIAN KERNIGHAN

MAY 2002

ACKNOWLEDGMENTS

I dedicate this paper to my parents for their constant support
and to my sister, Amanda, for her love and encouragement.

I especially thank my advisor, Brian Kernighan, for his guidance.

TABLE OF CONTENTS

ABSTRACT

IN THIS PAPER, we present a system for the presentation of news in a fresh and unbiased manner. The system collects and compares news stories from around the Internet in real time, providing users with a panopticon that peers around the world, to the source of each breaking news story, to hear what the locals have to say about it.

Our system does not pass judgment on the correctness of a given news story, but instead provides many different sides of that story to grant the user free license to make educated decisions for himself. As a story breaks and appears on one of several predefined news web sites here in America and England, our system detects the story, retrieves it, parses it, characterizes it, and then proceeds to search specific foreign newspapers from the countries mentioned therein to find stories related to the original.

The results are often fascinating. The foreign versions of a given story frequently contradict those that are presented to us by American media goliaths like CNN, *The Washington Post*, and *The New York Times*. Fundamentally suspicious of the biases that inundate the American media, we present a system that counteracts these biases—a system to combat the increasingly homogenous nature of American news. We present a system that seeks to provide "All the Sides to Every Story". We call our system *Extra!Extra!*.

1

INTRODUCTION

1.1 BEGINNINGS

AFTER SEPTEMBER 2001, the role of the American media assumed a prominence it neither sought nor deserved. Literally overnight, the cultural relevance of Tom Brokaw surpassed the cast of Survivor. Dan Rather became more enticing than Britney Spears, and Peter Jennings induced the kind of captivation normally reserved for rock stars and supermodels. The importance of news—the art of delivering the truth—became what mattered.

So tied up in presidential sex affairs, Cuban child refugees, and California congressmen, the American media hardly knew what to do when the situation suddenly demanded integrity. In early September, the media was entrusted with the difficult and unenviable task of relating to its audience the fate of the world, day-by-day, hour-by-hour, as bombs dropped, airplanes took off, and people tried to go about their lives as usual. It was a tall order for anybody, and probably one that could never be filled. Ours is not a world of binaries, and when news breaks somewhere on the other side of the planet, there are always several sides to the story.

The very nature of newspapers demands that they take a stance. Newspapers cannot cater to every constituency, because editors need to make decisions about what to print, what specific words to use, what tone to set, and generally who to endorse. No credible news source can get away with contradicting itself in its own pages; doing so would be to surren-

der respect, authority, and readership. Therefore, in the game of providing different versions of a story, newspapers have their hands tied. We sought to create a system that could achieve what newspapers cannot.

Our system would be free from corporate sponsorships, free from editorial decision-making, free from political obligations, and free from the fear of offending. We set out to produce a system that could truly fulfill the lofty maxim, "all the sides to every story".

1.2 THE CONTEXT

ON JUNE 4TH OF 2001, the Internet research group Jupiter Media Metrix released the results of a study proclaiming the drastic consolidation of Internet traffic in America. A year before, in the spring of 2000, only 110 companies controlled 60% of American Internet traffic, according to a previous Jupiter study. A year later, in June of 2001, the number of companies controlling the same 60% of Internet traffic had shrunk to fourteen, as listed in Figure 1.

At the time of the second study, four companies alone controlled over 50% of American Internet traffic. The big four were America Online, Yahoo!, Microsoft and Napster. Napster's dominance has recently been crippled by the legal action taken by record companies (statistics have yet to be released), but the other three most likely continue to dominate statistically. Despite Napster's recession, the evidence of increasing American Internet consolidation is powerful. "The data show an irrefutable trend toward online consolidation and indicate that the playing field is anything but even," said Jupiter analyst Aram Sinnreich. "The trend will likely continue. The few businesses that dominate the market are directing traffic across their own network of sites."

As these businesses "direct traffic across their own network of sites," the cognizant user begins to notice a bland homogenization of information. Since the AOL/Time Warner merger, when one logs onto America Online, the headline news stories are nearly identical to those being run concurrently on CNN.com, a subsidiary of AOL/Time Warner. Microsoft owns MSNBC, resulting in similar parallels between their two sites, www.msnbc.com, and www.msn.com, the URL's themselves virtual heteronyms. Ultimately, the American user is left with a landscape that lacks diversity—a world of corporate interests governing the presentation of everything from gasoline, to soft drinks, to cigarettes, to motor cars, to headline news.

Top 14 Online Properties as of June 4, 2001
Percentage Share of Minutes

AOL Time Warner Network	32.0
Microsoft Sites	7.5
Yahoo!	7.2
Napster Digital	3.6
Juno	1.9
eBay	1.9
EA Online and Applications	1.6
Excite Network	1.0
iWon Inc	1.0
Walt Disney Internet Group	0.8
Lycos	0.6
About The Human Internet	0.4
Flipside Sites	0.4
CNET Networks	0.3

Source: Jupiter Media Metrix

Figure 1, American Internet traffic in 2001

As American Internet users consistently narrow the scope of the sites they visit, the result is a steady consolidation, not only of traffic, but also of news. When Americans get their news online, they tend to visit one of several sites—namely CNN, *The New York Times*, *The*

Washington Post, MSNBC, ABC, and perhaps a few others [www.100hotsites.com]. These sources tend to share largely homogenous viewpoints and perspectives, mainly for the reasons outlined above. Readers have come to expect a common consensus among the main American news sources. Were The Wall Street Journal to contradict *The New York Times* on some controversial issue, the lack of consensus could arguably discredit both papers.

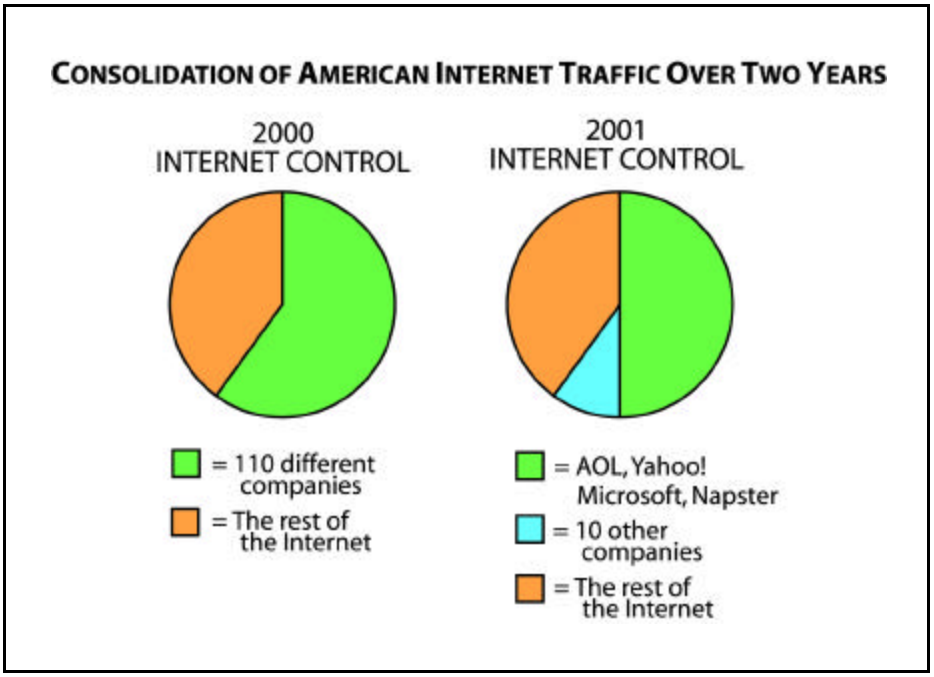

Figure 2: Consolidation of American Internet traffic

Whether propelled by corporate alliances, by political obligations, or by safety in numbers, most prominent American newspapers tend to assume a common stance on most international issues.

To find a different spin on international news, it proves essential to venture abroad—to the source of the news, as it breaks. Only by tapping into this international conglomeration of publications and web sites can one really hope to fulfill that ambitious maxim, "all the sides to every story."

1.3 PREVIOUS WORK

— 1.3.1 SEARCH ENGINE HISTORY

AS THE INTERNET has evolved from a tin-cup collection of connected computers into a multinational collaboration of millions of high powered machines, the quest to find information has been necessarily replaced by the much more daunting quest to decipher and categorize that information.

As recently as 1990, there were no search engines on the Internet, at least as we know them today. Indeed, there were no such things as web pages, but only anonymous FTP sites, which at the time were the most popular repositories of Internet files. In 1990, a McGill University student named Alan Emtage archived a listing of the web's most popular anonymous FTP sites, and he called his project "Archie", forced by the strict naming conventions of UNIX to choose a shorter name than "Archives". There were various other services that manually indexed sites, but the first "robot"—a system that could automatically find and rank these FTP sites—was not developed until 1991, when an MIT student named Matthew Gray developed his "World Wide Web Wanderer." The list formed by Gray's system came to represent the first organized database of Internet information, and he called it "Wandex" [FARRELY].

By mid 1993, three more robots, now known as spiders, were on the scene: JumpStation,

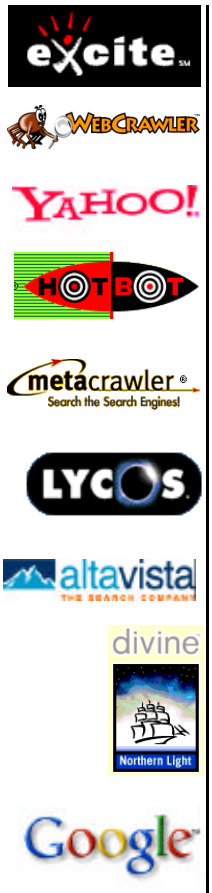

Figure 3: Some popular search engines

World Wide Web Worm, and the Repository-Based Software Engineering (RBSE) Spider. Later that year, the still popular Excite was developed by Stanford students, and soon thereafter WebCrawler was launched at the University of Washington.

Created in a Stanford dorm room by Jerry Yang and David Filo, Yahoo! had stolen the show by late 1994, with its personally developed indices of the web's top sites. Despite its ease of use, Yahoo!'s scope is limited to a mere 1% of all total web sites, making it a good choice for specialization, but a poor choice for comprehensiveness. By 1996, competitors like HotBot, MetaCrawler, Lycos, Altavista, Northern Light, and many others had emerged.

They all claimed to do the business of searching a bit differently, but the resulting saturation of the market made it difficult to imagine that yet another competitor could ever hope to enter this late in the game and steal the show. When Google arrived on the scene with its simplified interface, its excellent results, its speedy turnaround, and its PageRank system that partly bases a page's relevance on citations and backlinks, it quickly zoomed to the top of the searching game, winning over users around the world and building up an index of over two billion web pages.

Google's PageRank algorithm, while proven excellent in broad based search, fails to apply to breaking news. The PageRank algorithm uses the citation (link) graph of the web to count the number of links and backlinks to a given page, thereby determining its relevance [BRIN, PAGE]. The technique uses human input (the initial link creation process) to influence the automated search results, thereby combining intelligence with efficiency. This technique cannot apply to news, because by the time a news story has been linked to by other pages, the news story will almost certainly be obsolete. For this reason, Google's PageRank algorithm clearly could not apply to *Extra!Extra!*.

Every search engine is a glorified telephone book. They provide addresses and sometimes descriptions of sources of information around the web, and they tell how to get there. Search engines are essentially portals—tollbooths and tunnels that users pass through on the way to their destination. They are convenient and crucial, but they are not destinations, nor can they purport to be. The nature of their work is that of transience.

—1.3.2 LEXIS NEXIS

THE INTERNET IS teeming with search engines, but there are considerably fewer sites that not only search, but also recover and store. Search engines catalogue the data—but they do not save it. An exception is LexisNexis, which harvests often copyrighted information, both on and off the Internet, and sells it to users for a premium. In the act of gathering information, reformatting it, and then passing it on, LexisNexis bears similarity to *Extra!Extra!*, but LexisNexis is not a service for the everyday user. Its targeted audience includes academic institutions, researchers, corporations, independent firms, and other private ventures, but it does not pretend to be useful to random Americans seeking information in real time and on the run.

Figure 4: LexisNexis, which sells copyrighted materials to users

Extra!Extra! differentiates itself from

LexisNexis in its fundamental attitude about the business of information. LexisNexis strives to generate revenue and make profit, and in doing so, ignores many average users who have no interest in paying to get their online news. *Extra!Extra!* is a free service that helps the masses develop a clearer picture of the news. *Extra!Extra!* was created to counteract the corporatization of news, not to encourage it.

—1.3.3 GOOGLE'S HEADLINE NEWS

ON DECEMBER 19, 2001, Google unveiled a service strikingly similar to *Extra!Extra!*. [SHERMAN] This "Headline News" service:

http://www.google.com/news/newsheadlines.html

collects news headlines from roughly 100 different English language newspapers around the world, updating the record once every hour. Google crawls these predefined foreign newspapers, like *Extra!Extra!*, and decides upon the most pertinent headline stories. For each headline, Google lists between three and five related story headlines from other newspapers around the world. At this juncture *Extra!Extra!* departs from Google. Like any other search engine, Google simply links users to these headline news stories. Once there, the user has to contend with pop-up windows, advertisements, endorsements, sponsorships, unrelated links, and a whole host of other obstacles lying between him and the news. *Extra!Extra!* takes a different approach.

When a user arrives at *Extra!Extra!* to get his news, he has no reason ever to leave the site. Instead of linking to news stories like a search engine, our system retrieves the stories and stores them locally. In the retrieval process, *Extra!Extra!* discards everything but the most crucial parts of a story—text and photographs. These news stories are saved on the *Extra!Extra!* server, allowing instant retrieval and consistent presentation.

Another element that Google ignores is the relevance of the past. Many stories take several days or longer to develop, and Google's service merely catalogues the current headline news for a one-hour window. Google does not archive old news stories, nor does it allow users to trace the evolution of a given story over the course of several days. To this end, *Extra!Extra!* archives one week of headline stories, and for each new story retrieved, our system checks all preexisting stories to see if any might be related. If they are, the correlation is indicated to the user. In this manner, users can trace a story's evolution over hours and days, following the coverage of different newspapers around the world.

—1.3.4 NEWSBLASTER

PERHAPS THE MOST significant news summarization effort to date is Newsblaster (www.cs.columbia.edu/nlp/newsblaster), a system currently being developed by researchers at Columbia University's natural language programming group, under the lead of Professor Kathleen McKeown. Newsblaster scans seventeen mostly American news sources, including *The Washington Post*, BBC News, and Lycos, pulling out certain sentences and phrases from relevant articles at each news source. The goal is to create a five-sentence summary of each current news story, achieved by combining five key sentences from five different sources. Using natural language processing routines, sentences are weighted on how often similar sentences occur—the more often occurring, the more relevant the sentence. The five-sentence summaries read much like shortened Reuters news briefs—general and

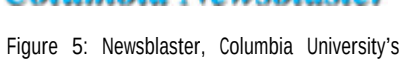

Figure 5: Newsblaster, Columbia University's attempt at news summarization

succinct—but often ending up awkward, bland or repetitive.

The system occasionally includes sentences that clearly do not belong in a news summary. In summarizing the recent Enron scandal, Newsblaster included a sentence from an opinion forum, where a user commented:

"Thanks for finally pointing out the culpability of Enron's employees in their own demise for heavily investing in Enron stock when they should have known better."

Newsblaster frequently includes multiple sentences that state the essentially same thing, as in describing R.E.M. guitarist Peter Buck's acquittal from allegedly starting a fight on an airplane:

"Buck , 45 , was found innocent of assault, and being drunk on an aircraft. R.E.M. guitarist Peter Buck sighed and wiped his brow Friday after a jury acquitted him of charges of going on a drunken rampage on a trans-Atlantic flight. Rock star Peter Buck from top-selling American band R.E.M. was acquitted Friday of a drunken mid- air rampage. Buck is charged with being drunk on an aircraft."

Despite these occasional semantic flaws, Newsblaster is fundamentally different from our system in what it sets out to do. In summarizing headline news based on seventeen mostly American newspapers, Newsblaster takes what we argue is already homogenous news, and makes it even more homogenous. *Extra!Extra!*, on the other hand, seeks to present the news in all of its multi-sided complexities, giving equal weight to opinions from all over the world.

When seen through this lens, *Extra!Extra!* really has no precedent. It has historical touchstones, points of inspiration, and features shared by other sites, but ultimately *Extra!Extra!* charts new territory, inspired by new motivations—not profit, not homogenization, but genuine concern for the creation of a well-informed society.

1.4 OUTLINE OF THIS PAPER

WE BEGIN BY describing the look and feel of the system that users see online today. With a sense of the finished product, the explanation of its evolution is easier to grasp. We proceed with a discussion of the visual design decisions that were made in constructing the *Extra!Extra!* interface. The concern for sophisticated simplicity was paramount from the start. The back-end could be a complicated quagmire, but the front-end presentation had to be obvious and easy. We were designing a system for the layperson and the professor, the soccer mom and the scientist.

We address the system's technical evolution—the main components that compose the *Extra!Extra!* news engine. We describe how ideas changed and how the system took shape over the course of several months.

Initially we needed to find a consistent and reliable general listing of all the main news stories happening in the world at any given moment. We discuss the tradeoffs between various places we considered, including the Associated Press News Feed, and then we explain our reasons for settling upon Moreover Technologies as the company to provide the crucial listing of headline news for *Extra!Extra!*.

Once we address the abstract structure of the system, we discuss the nature of the story searching algorithm, which lies at the core of *Extra!Extra!*. This algorithm allows

Extra!Extra! to decide if two web pages, written in HTML, contain textual stories that are related to each other. If so, the algorithm proceeds to decide what is important and what is superfluous. It discards the HTML, saving only the relevant story's text, and any pertinent images that happen to be contained in that text.

The algorithm's precision allows *Extra!Extra!* to archive clean versions of each retrieved story on our own server, taking the content from its original source, reformatting the words and pictures to conform to the *Extra!Extra!* template, and then saving the resultant file for one week.

Each textual story file is defined by a corresponding description file, which consists of a set of keywords that define the article. These description files label each story so that other description files can decide if the two articles are similar. The server side description file process is detailed later in this paper.

We discuss *Extra!Extra!*'s method of deciding how a given story developed, if in fact the story has been ongoing for several hours or days. If so, our system ties the new story to any related previous stories, creating an evolution of the story that the user can follow. We have found that studying the evolution of a story over several days often proves even more enlightening than reading through a story's current foreign cousins at various foreign newspapers.

Next comes a discussion of the main problems we faced—the hurdles we overcame, and the most daunting issues that still face *Extra!Extra!* today. We examine some of the general issues confronting any system that hopes to catalogue foreign news comprehensively, accurately, automatically, and quickly.

We conclude with a discussion of future work that could extend the functionality of our system, along with a final assessment of *Extra!Extra!*'s current worth.

2

CONCEPT DESIGN

Extra!Extra! demanded the creation of a system that was robust and reliable, one that would function without any human supervision, perform with consistent accuracy, and offer an interface that balanced sophistication with usability, to satisfy both the expert and the novice.

2.1 USER INTERFACE DESIGN

WE RECOGNIZED THAT ONE of the main problems facing news in the 21st century is how best to sift through the masses of information to get at the relevant stuff. Therefore simplicity was our goal in designing *Extra!Extra!*'s user interface.

The main index page offers few choices to the user. In the center of the page is a list of links under the label, "Today's Headlines". This list grows as the day progresses, with new stories appended every thirty minutes. Each link displays the story title, the originating news source, and the timestamp of initial retrieval. These links can be considered parents, and each of them leads to an area of the site devoted exclusively to that headlined story. That area contains any foreign stories that relate to the original, in addition to any old archived stories from the last seven days that relate to this new one.

In this sense, the site design of *Extra!Extra!* is modular, with strict channels and roadways that lead to independently partitioned areas of

the site, each containing all the relevant information about a single news story. This design was meant to help users envision the news in a more compartmentalized fashion, stressing that a single story can have many different interpretations from diverse sources around the world. As they are forced to navigate the site from "story group" to "story group", users come to think of news stories as organic entities that can each have many personalities and many interpretations. The site structure is meant to facilitate the kind of open-mindedness that most prominent news sources ignore, and the kind of multinational awareness that we initially set out to inspire.

Other than the listing of current headlines on the main page, users can link back to "Recent Headlines" from the last seven days. These lists are exact replicas of the headline list at midnight on a given day, when a new list is created for the day to come. These lists allow users to navigate through the last week's news, learning about specific stories that might have recently subsided. These headline lists ensure that recent news is not lost at the end of the day, and that users can easily return to a recent story that especially interested them.

A prominent graphic indicates when the site was last updated, calling attention to the sense of immediacy. Other links on the main page include "Participating News Sources", a complete record of all 66 newspapers covered by *Extra!Extra!*, along with links to each of them and groupings that declare which country they represent. A "Statement" page provides a legal disclaimer that explains the project's academic goals, and a "Purpose" page describes the mission of *Extra!Extra!*. Finally, a "Contact" page lists details about how to make suggestions regarding the general improvement of the site.

Once a user clicks a headline link on the main page, he is taken to a page containing only the text and photographs of that article. This stripping down of information allows the user to examine intensely the material at hand, undisturbed by anything extraneous.

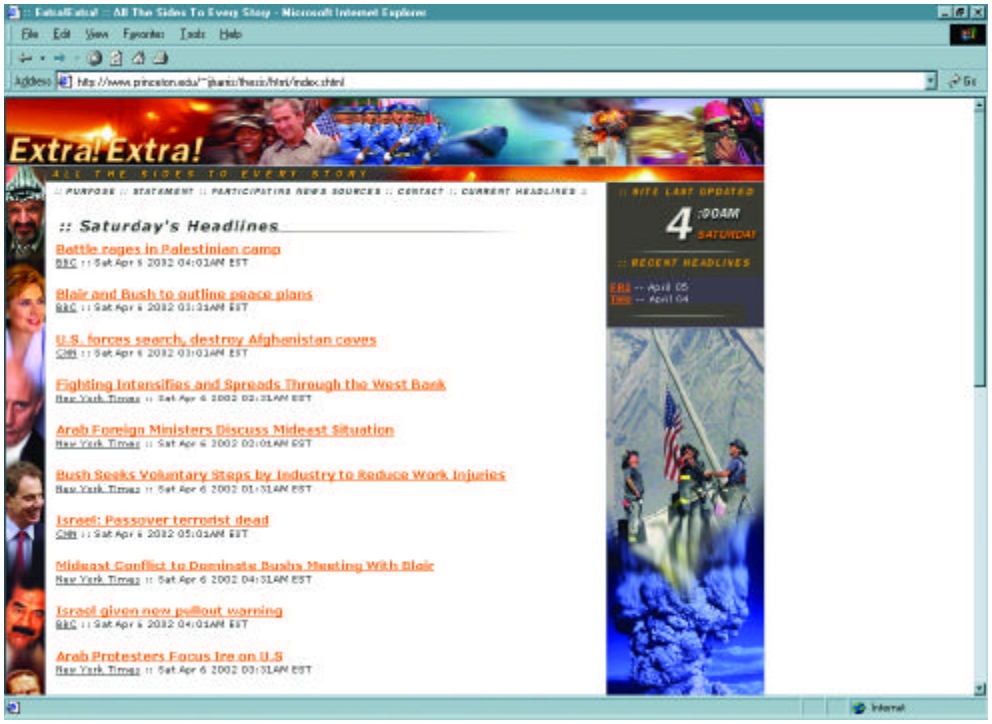

Figure 6: The main screen, featuring the day's headlines in the central window

Below the story's title are the source newspaper's logo and the timestamp of initial retrieval. Within the page's right sidebar lies the substance of our system. In this sidebar are the links to related foreign stories, under the heading, "Other Sides to the Story". In choosing this title, we were careful to stress the familiar and colloquial nature of news retrieval. We consciously rejected a traditional phrase like "Related Stories", because that sort

of phrase comes loaded with reference points at CNN and other news sites that profess to offer "Related Stories"—disparate and truly unrelated stories that happen to have a few people and places in common. Our listings are not "Related Stories", but are instead other ways of looking at the same story. This distinction is crucial, and it reinforces the realization that there can be no absolutist version of a single news story.

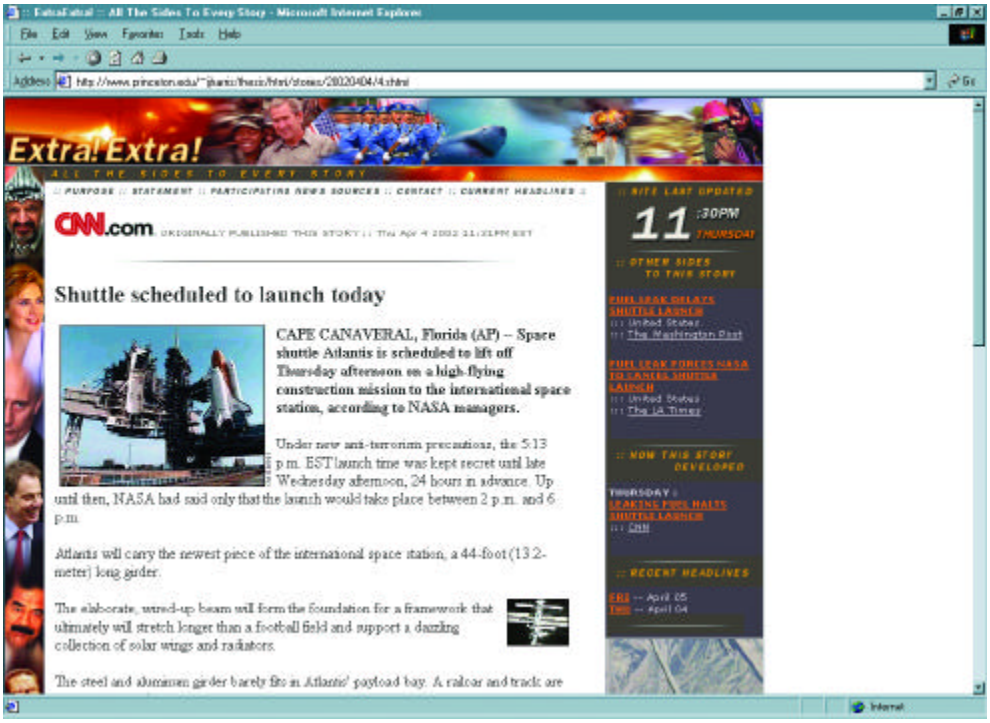

Figure 7: The news article interface, with the text and pictures in the central window. In the right sidebar are the "Other Sides to this Story" links, the "How This Story Developed" links, and the "Recent Headlines" links.

Below the "Other Sides to the Story" links are a listing of recent archived stories that bear relevance to the current one. These past stories are organized beneath the heading, "How this Story Developed", recognizing the role played by time in affecting the media's coverage of a given event. Traditional news sources often de-emphasize their initial stance on an issue if it seems to contradict their current one. *Extra!Extra!*, on the other hand, believes that a record of these initial reactions provides an excellent meter with which to gauge a given paper's account. The attention to past related coverage is unique to *Extra!Extra!*, which tries to present every thread of relevant information that exists.

There are four main elements to our news presentation system, and we have introduced all of them:

- the "Current Headlines" list
- the textual story page
- "Other Sides to this Story" links
- "How this Story Developed" archives

The entire process of news searching, retrieval, parsing, extracting, analyzing, defining, comparing, archiving and presenting has been boiled down to four sections for the user to see.

2.2 VISUAL DESIGN

DEVELOPING AN authoritative yet fresh and jazzy visual design was of utmost concern from the get-go. Existing Internet news sites tend to resemble one another in layout as much as they do in editorial content. We set out to differentiate ourselves on both accounts.

We wanted to combine a sense of unchanging and consistent authority with one of excitement and novelty. To do so, we designed a digital collage of recognizable recent news events from around the world. The montage includes the New York firefighters raising the flag in the World Trade Center's rubble, Chinese soldiers on the march, an open-jawed

great white shark, a California brush fire, an Afghan refugee, a biohazard team checking for Anthrax, and a driving Michael Jordan with his tongue extended. This banner collage could easily be changed every several weeks to reflect the current issues in world news. Additionally, several different collage designs could be available concurrently, rotating randomly with each page visit.

Along the left column of the page are the faces of the world's most prominent leaders—George W. Bush, Yassar Arafat, Vladimir Putin, Tony Blair, Aung San Suu Kyi, Colin Powell and others—alongside the world's most notorious—Osama Bin Laden and Saddam Hussein.

Besides the photographic collage, the visual design is stark and uncluttered, mostly white, and presents the news titles and stories in a staid and conservative manner. There are no flashy animations, nor gimmicks; it seemed most helpful to the user to minimize distraction and maximize simplicity.

2.3 BRANDING THE NAME

EQUALING THE IMPORTANCE of visual design was the choosing of a compelling and memorable name to market the system. Originally we considered the name "Real News", to signify the relative accuracy of our system as compared to most existing American news sources. The name, "Real News" however, sounded too much like an offshoot of Real Networks, with their omnipresent Real Player. We wanted to avoid such correlations, so we kept thinking.

We considered "Local Lore", but opted out of it because the word "Lore" is not authoritative enough for a credible news source to embrace. We thought about "Global News", but that had too many implied ties to the anti-globalization movement.

At last we settled on the name, "*Extra!Extra!*", for several reasons. Firstly, it manages to convey the sense of "News" without actually saying it. With an unorthodox name like "*Extra!Extra!*", we could happily avoid using any of the 21st century buzz words that populate the media—words like "Global", "News", and "Truth". "*Extra!Extra!*" had its own identity, its own ring, and didn't carry with it any obvious reference points.

Secondly, the name "*Extra!Extra!*" gives more bang for its buck. When most Americans hear the phrase "*Extra!Extra!*", they almost immediately follow with a subconscious "Read all about it!". In this sense, we get six words for the price of two.

Thirdly, the presence of the exclamation point instills a sense of urgency and excitement, like something big just happened somewhere, and you absolutely have to read this publication to learn the details. For these three reasons, we found the name *Extra!Extra!* to be appropriate and effective for our system.

3 TECHNICAL DESIGN

***EXTRA!EXTRA!* STARTED** as an idealistic vision of a better way to get news, but a vision with no technical direction, and no real sense of structure. We envisioned a system that could somehow harvest similar news stories from around the Internet in close to real time, and then consolidate them into one uniform layout design on one central web site that users could visit to read more than one account of a single news story. The system would eliminate the need to manually check each of many international online newspapers to approximate the whole truth about what had really happened. For example, if a bomb went off in Israel, we envisioned a system that would present CNN's account of the incident alongside the Israeli account, the Palestinian account, and maybe even the Pakistani account if Pakistan were involved.

3.1 FINDING A FEED

THE FIRST STEP was to find a reliable news feed that could supply the system with up-to-the-moment records of all current breaking news. The Associated Press operates such a feed, and it appears in a small side panel on many sites around the Internet, including *The New York Times*. This feed is very basic—generally consisting of just headlines—and goes no further than succinctly describing the subject of each story. The feed seemed perfect as a basis for *Extra!Extra!*, but the Associated Press charges a premium for on-site access to their news feed. For companies like *The New York Times*, the premium fee is pocket change, but for a senior thesis budg-

et, it busts the bank. So we had to look elsewhere. We contacted Reuters, and several other news providers, but they all run their businesses on hefty corporate contracts that demand heavy usage fees.

After much searching, we came across a company called Moreover Technologies. Based in San Francisco, they are in the business of providing up-to-the moment news from around the web for paying clients. For a fee, they supply Rich Site Summary (RSS) feeds to users. RSS feeds categorize the content of XML-equipped web pages, outputting named objects that correspond to different data contained in the page. Moreover's RSS feeds contain the headline news from several leading news sources, namely *The New York Times*, *The Washington Post*, BBC, and CNN.

We contacted Moreover Technologies, explained our project to them, and they approved of its academic merits, therefore agreeing to supply these feeds for free. They have unblocked the relevant IP addresses on Princeton's Computer Science Department server where *Extra!Extra!* is housed, allowing our program to tap into their RSS feeds.

We considered writing our own RSS feeds to maintain greater independence—to create a truly autonomous system, but we elected not

```
  <?xml version="1.0" encoding="iso-8859-1" ?>
  <!DOCTYPE rss (View Source for full doctype...)>
- <rss version="0.91">

- <channel>
  <title>Moreover - Top stories</title>
  <link>http://www.moreover.com</link>
  <description>Top stories - news headlines from the web </description>
  <language>en-us</language>

- <item>
  <title>Trains collide in California</title>
  <link>http://c.moreover.com/click/here.pl?r36649486</link>
  <description>BBC Apr 23 2002 11:24AM ET</description>
  </item>

- <item>
  <title>Talks aim to end Bethlehem siege</title>
  <link>http://c.moreover.com/click/here.pl?r36642437</link>
  <description>BBC Apr 23 2002 9:42AM ET</description>
  </item>

- <item>
  <title>Cuba-Mexico Relations Altered by Tape Recording</title>
  <link>http://c.moreover.com/click/here.pl?r36640442</link>
  <description>New York Times Apr 23 2002 9:08AM ET</description>
  </item>
```

Figure 8: A few lines of Moreover's Top News RSS feed, which breaks down HTML pages into an XML-enabled list of "items", each providing a short story headline and a link to that story's original location, accessed through a Moreover script.

to do so for several key reasons. Most importantly, Moreover Technologies has arranged a special corporate deal with *The New York Times* to allow their scripts to access the usually restricted *Times* web pages. The *Times* requires its users to login each time they visit the site, presumably to discourage scripts like ours from reaching the sub-pages of their web site. When we attempted to write our own RSS feeds, they functioned perfectly for all news sources except The *Times*. Since Princeton's server has not received authorization to access The *Times*' web site, our attempts at RSS feeds were denied. We weighed the situation and ultimately determined that *The New York Times*, as one of the world's most widely read and respected newspapers, was a crucial voice to represent with *Extra!Extra!*, so we elected to remain with Moreover Technologies and borrow their RSS news feeds.

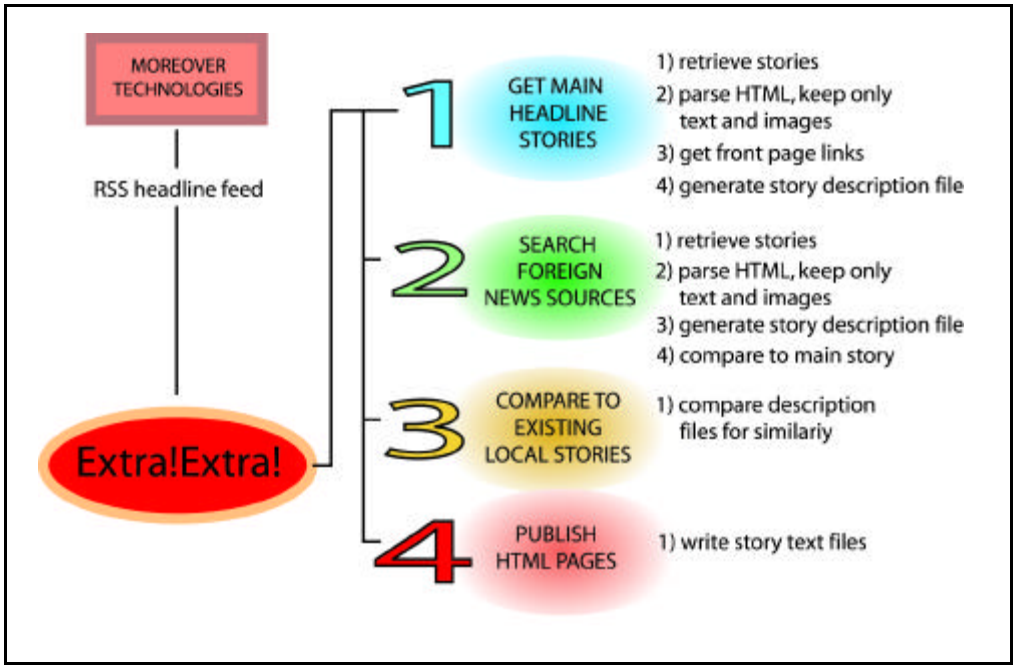

Figure 9: The program control flow structure, as outlined in Section 3 of this paper

Moreover is in the business of providing RSS feeds. As a professional corporation, Moreover will maintain its feeds with great stability and reliability. If a given newspaper decides to change its structure, Moreover will address the change quickly. We designed *Extra!Extra!* to be a completely autonomous system with no regular employees to check on things daily. For this reason, it makes sense for *Extra!Extra!* to leave the constant RSS feed maintenance to an independent company that can monitor potential changes in external newspapers as part of its job. This realization cemented our decision to leave the RSS feeds to Moreover Technologies.

3.2 THROWING OUT THE GARBAGE

THE MOREOVER NEWS FEEDS provide headlines, timestamps, and links to each article, but they simply lead to the HTML page of the original story. The Moreover links do nothing to weed through the irrelevant information on the page, and therefore we had to develop an HTML parsing routine that could sift through a given HTML page and keep only the relevant text and images. We had to invent a process intelligent enough to differentiate the nonsense from the substance.

After close examination of a sampling of news sources around the world, it became apparent that almost every online newspaper conforms to some standard set of layout protocols. Some of the more advanced sites, such as *The New York Times*, include a wonderfully obvious set of tags that surround each article's body text. The *Times* precedes each article with the string, "`<NYT_TEXT>`", and then follows each article with the string, "`</NYT_TEXT>`". Most newspapers do not

behave so cooperatively, but almost all have some mark of consistency from page to page that roughly approximates the location of the story's body text within the HTML page. Sometimes this mark of consistency is a bulleted image, or a font tag, or a closing table cell, but almost every newspaper at least has something.

We arrived at a list of roughly 65 foreign newspapers around the world for *Extra!Extra!* to cover. South America and parts of Africa have very few English language newspapers, but the rest of the world is well represented. For each of the news sources we identified, we located "starter" and "ender" strings, as in "`<NYT_TEXT>`" for *The New York Times.*

After identifying starter and ender strings for each foreign paper, we began to craft the algorithm that would parse an HTML page from a given newspaper and return a good approximation of that page's story text.

Using Perl, the obvious choice for string manipulation and web interaction, we initially created an algorithm that takes three arguments—a link to any HTML page, a starter string, and an ender string. This algorithm was portable and concise, and served to whittle much of the irrelevant (i.e. non-body text) information from any HTML document. The algorithm was handy, but we had to construct a clearer set of rules and regulations for it to follow. For certain HTML pages the algorithm returned an unacceptable amount of excess information. Especially for newspapers with unobvious starter and ender strings, there was often a glut of unneeded text surrounding the desired body text of the article.

We honed this HTML-parser algorithm to be even more precise, making a few initial assumptions:

1) We want to save the article body text.

2) We want to save the HTML page's `META-DESCRIPTION` information, as it provides the most concise and accurate textual definition of the page's content.

3) We want to save all links to images, whether relative or absolute.

4) We want to save all basic text formatting (bold, italics, underlines, `<H></H>`, `<UL></UL>`, `<BR>`, `<P>`).

5) We want to discard everything else.

Once we had these rules in place, we used Perl to execute them.

Upon visiting an HTML page at a given newspaper, the algorithm now completes the following steps, in order.

1) Look for and save the `META-DESCRIPTION` information.

2) Index the position of the predeclared starter and ender strings for the given newspaper, assuming they both appear somewhere in the HTML file. If they do not appear in the HTML file, declare the page irrelevant and proceed to the next one.

3) Once the starter and ender strings are indexed, throw away everything before the starter and everything after the ender. Presumably, what is left will be a reasonable approximation of the article's body text. If the length of the body text is less than 100 characters, we assume there was either a transmission error, or that the HTML page in question is irrelevant, so we discard it and proceed to the next one.

4) Otherwise, we search through the remaining body text string, searching for any

occurrence of <A HREF> tags, signifying images. We store every image link with a ".JPG" extension in a temporary "images" file. We ignore other image file types because ".GIFs" are generally used for logos and graphics, while ".JPEGs" are generally reserved for news photographs that pertain to the given article. If the image links are relative links, we convert them to absolute links, using the base link (e.g. http://www.cnn.com) for the given newspaper.

5) Parse the remaining article body text, searching for all tags that we want to keep, as outlined above. Upon encountering a desirable tag (<B></B>, <I></I>, <U></U>, <H></H>, <UL></UL>,
, <P>), its opening and closing brackets are converted to "<" and ">". This step ensures that they will not be deleted during the tag deletion process.

6) After preserving the good tags, the algorithm proceeds again through the body text removing all remaining tags. In this step, tables, style sheets, images, JavaScript, advertisements, banners, spans, links, and all other extraneous tags are deleted, leaving behind only the article's body text, and the previously preserved HTML tags.

7) The preserved HTML tags are reconverted to their original state as "<" and ">".

8) What results is a clean and correct version of the article's body text, with minimal text formatting, and no other garbage. Adjacent to this newly honed text file is a list of all JPEG images that we previously stored in a temporary file. At this point, the images are sequentially merged back into the text file at regular increments—specifically every second time a "<P>" tag is encountered.

We considered remembering the JPEG images' original locations in the article, but concluded that the position of photographs in online news articles rarely matters. One photograph per article is the norm, and it generally comes at the head of the article, which is where our system automatically inserts any saved images.

We considered saving any captions that might have accompanied the original image, but many images are embedded in tables that contain irrelevant phrases like, "Save this," "Print this," and "Email this" (CNN). The program would have no way of differentiating such phrases from an actual photograph caption, so we elected not to save captions.

Once we have "thrown out the garbage", we can begin to examine the actual meaning of the textual content contained in the HTML page.

3.3 FINDING MEANING

AFTER THE **HTML-PARSING** algorithm is complete, the remaining words in the article suddenly take on an added weight. Since we know there are no extraneous components to the page (links, tags, JavaScript, etc.), we can start to make judgments about what the article is about, based on the text that remains.

Before we proceed, we save the existing article into a permanent file, complete with its minimal formatting and its integrated images. Once we have saved another copy of this "publishable" version, we are free to manipulate and destroy the already existing copy in order to define its content.

To prepare the text for this process, we first convert all characters to lower case. Then we remove all non alpha-numeric characters from the string, including all punctuation. Then we remove all remaining tags, specifically the ones that we had preserved before. Then we remove all newlines and all extra spaces.

The result is a list of words, delineated by single spaces, with no punctuation and no capital letters. In this minimalist form, we can most effectively evaluate the meaning of this article. What occurs next is the heart of our algorithm.

We somehow needed to develop a "definition" file for each story. This file would have to sit on the server and represent the article to other articles that wish to compare themselves to this one. It would have to be relatively short, precise, and consistently accurate. We sought to create, for each article, a list of 20-50 words that effectively define that article's content. If we could accomplish this, the future processes of comparing various articles to one another would be simplified. Out of each article of several hundred words, we wanted to extract the 10-30 words that most clearly represent the article's meaning.

To accomplish this task, we used a word frequency list developed by Patrice Bonhomme (http://www.loria.fr/~bonhomme/sw). We also considered other lists, including the Brown corpus, but eventually settled on Bonhomme's list because it seemed to perform sufficiently well in early experiments. Bonhomme's list contains 31,886 English words and an estimate of how frequently they each tend to occur in a typical English corpus of size 593,745. The values range from 1 occurrence (e.g. "zealot") to 38,116 occurrences ("the"). We use this list to weigh the relevance of certain words in our text article.

Before doing this however, we use several other heuristics to make the definition process more effective. We use a stop-word list, also developed by Patrice Bonhomme, to automatically delete certain irrelevant words that will not add to the meaning of the article. This list contains 517 words, and includes such words such as "again", "around", "meantime", and "myself".

A third list we employed is one of common English verbs, found on the "English as a Second Language" section of About.com. This list contains 559 mostly non-descriptive common verbs, such as "brought", "were", and "dreamed".

With these three lists in hand, we could analyze our news stories. To prepare each article for its analysis, we first applied the stop word list, removing all non-subjective words from the text string. Secondly, we applied the verb list, removing all non-descriptive verbs from the article. What remained was a list of proper nouns, countries, states, people, technical terms, corporation names, scientific specifics, and other unordinary English words. This conglomeration of words provided an effective summary of an article's potential meaning.

Initially, our algorithm stopped here. We decided however, to improve its accuracy by incorporating the English word frequency list that we mentioned above. After applying the stop-word list and the common-verb list, we were left with a list of mostly proper nouns and other unusual words. The occurrence of such words generally indicates strong subjective relevance, but we had to decide how much relevance. This is where the English word frequency list came into play.

For each word in the word frequency list, we associated a ratio to indicate what percentage of the time that word tends to occur in the typical English corpus. We could then parse through the article, counting the occurrences of each word therein. If a given word appears more often (as a percentage) in the article than it typically does in the English corpus, then we add the word to the article's description file.

For example, if the word "Taliban" typically appears 0.00002% of the time in the normal English corpus, and appears 0.7% of the time in a given news article, we

conclude that the article has to do with the Taliban.

Once we have applied the two heuristics of stop words and common verbs, and once we have executed the word frequency comparison process, we have developed a reasonable approximation of an article's meaning, stored in a description file on the server. An example description file follows.

On Monday February 25th, 2002, *The New York Times* ran a story entitled, "Israelis to Keep Arafat Confined, but Loosen Reins", about the Israeli government's decision to prevent Yasir Arafat from leaving Ramallah. The *Times*' article discussed the tensions between Israel and Palestine, gauging the particular reactions of Israeli Prime Minister Ariel Sharon. The article went on to mention the stance of the United Nations, voiced by Secretary-General Kofi Annan. The entire text of the article was 1,126 words. The corresponding description file produced by *Extra!Extra!* was 68 words. The description file, as it appears on the server, is replicated below:

```
ramallah compound arafat nablus
armored pressure abdullah coalition
kofi canceled palestinians Loosen
buffer negotiator preconditions
checkpoint lieutenants issued
israel's incarceration sheetrit
killers yasir meir shimon conces-
sion palestinian zeevi nabil extra-
dition tamp hiba assassinating
Israelis peres leader's issue per-
mission abstained sharon Confined
israelis zeevi's israel quell
arafat's rudeineh sharon's annan
withdrew detaining rift rehavam
witnesses communiqué israeli iran
roadblock saudi abu humiliating
raged escalation assailed loosening
aggression Arafat forays
```

The description file provides a fairly accurate representation of the article's content in **`68/1126 = 6.03%`** of the words contained in the original article.

With the definition process complete, we began the process of searching foreign newspapers for similar stories that match the given description file.

3.4 GOING ABROAD

AS MENTIONED IN Section 3.2, we identified a set of 66 foreign newspapers around the world, storing this listing in a sub-package called "`newspapers.pm`". We set up a one-to-one correspondence between the set of newspapers and the set of starter/ender strings. We stored this information in another sub-package called "`whichPaper.pm`". We used two separate packages because each package is used at a different point in the program. `Newspapers.pm`, with its list of predefined news sources for each country, is used to develop the list of which sources to search for a given initial article, based on which country names are mentioned in that article. `Newspapers.pm` allows there to be more than one foreign paper for a give country.

`Whichpaper.pm` is invoked when it comes time to search the specified foreign sources. This process occurs one source at a time, as whichpaper.pm dictates the relevant starter and ender strings for each news source.

The next stage was to identify the relevant countries for each article. If the article was about insider trading in Stockholm, the program needed to recognize that the story concerns Sweden, and therefore the predefined Swedish newspaper should be searched for its own accounts of the same story. We used several heuristics to expedite the search for individual country names in the article.

The article's full text is copied into a temporary string, and all non-capitalized words are removed. When only capitalized words remain, the searchable string is generally reduced to city and country names, people's names, other proper nouns, and words that begin sentences. The reduction in string size that occurs by discarding non-capitalized words depends on the article in question, but testing shows that it generally reduces the searchable string to 5%-20% of its original length. This shortened list is searched for country names, and the system deduces which foreign newspapers to check.

In alphabetical order, the program retrieves the current index page for each foreign newspaper in the recently formed list. It parses the text of the index page, forming a list of every "<A HREF>" tag that appears there. Our system uses the following code, in Perl, to identify and mark all "<A HREF>" link tags that occur on the HTML page:

```
# Regular Expression to find all
link tags and mark them
$testString =~ s/
.*?<a
.*?href\s*=([“\s’])(\S*?)\1.*?>(.*?
)<\/a>
/”$2”###$3###\n\n/
ig
;
```

We make the assumption that every current breaking news story will be linked to from a newspaper's opening index page. In this sense, our program only covers "one level" of links —those that are followed from the initial index page. Once this list of main page links has been formed, we apply a series of heuristics to narrow down the number of sub-pages we will have to examine. For instance, a typical news site has links to advertisements, sponsors, horoscopes, archives, other sites, videos, surveys, maps and so on. Clearly, we are only interested in news stories, and we do not wish to concern ourselves with all of the other minutiae that tends to occur on a news web site.

To solve the problem of identifying the relevant links, we predefine for each foreign newspaper a "relevantLinkPhrase" that must appear in a given link for it to be considered a news story. These phrases vary from paper to paper, but almost always stay consistent within a given paper. For instance, every news story URL at France's *International Herald Tribune* contains the phrase "/articles/", while every story URL at *The Copenhagen Post* contains the phrase "default.asp". Building up a relevant link phrase for each foreign newspaper, we developed a quick and consistent way of examining only the potentially relevant links on a foreign web site, thereby improving performance time. We stored this relevant link phrase information alongside the other predefined data for each foreign newspaper in the sub-package "whichPaper.pm".

A typical entry in the "whichPaper.pm" package is illustrated below, this one for BBC.com:

```
if($main::thePaper eq “BBC”){
      $main::whichSource = “BBC”;

      $main::baseLink =
“http://news.bbc.co.uk/”;
$main::domainString = “bbc.co.uk”;
      $main::starter = “<DIV
CLASS=\”bodytext\”>”;
      $main::ender = “<TD><IMG
SRC=\”/furniture/nothing.gif\””;
      $main::relevantLinkPhrase =
“newsid_”;
}
```

The slowest part of the *Extra!Extra!* system is the URL retrieval of foreign web pages, using the call to Perl's "GET()". These web pages

must often be retrieved from heavily trafficked servers on the other side of the globe. The more we could narrow down which links to check, the more efficiently our program would run. These aforementioned heuristics came late in the game, but improved runtime once invoked.

Checking only the links containing the "`relevantLinkPhrase`", we sequentially examine each of the HTML pages linked to from each paper's index page. Each of those HTML sub-pages undergoes the same HTML parsing algorithm described in detail in Section 3.2, producing a bare bones text and images version of the HTML page. Once the HTML parser function has successfully finished "throwing out the garbage", we can begin to define the content of the remaining text file.

To do this, we use the same definition-creation routine that we outlined in Section 3.3, and we end up with a short list of words that accurately define the foreign news story's content. We can cross reference this foreign story's description file with that of our original local story, to see if the two are similar. After analyzing the data for many story comparisons, it became clear that a consistently reliable similarity ratio is 14%. If the two definition files have greater than 14% of their words in common, the two stories are considered similar. If the percentage of words in common is 14% or less, then the two stories are considered dissimilar, the foreign story is discarded, and the program proceeds to examine the next link in the foreign link list.

For example, an original article from *The New York Times*, entitled "President Urges Pullout With No Delay", details a speech given by Bush and Tony Blair urging Israel to withdraw its military forces immediately from their West Bank incursion. *Extra!Extra!* parses and describes this article, as described in Section 3.3, and then compares various retrieved foreign articles to this one to test for similarity. This process is demonstrated below.

```
---------- NEW STORY ------------
TITLE: President Urges Pullout With
No Delay
SOURCE: New York Times Apr 6 2002
7:52PM ET
LINK:
http://c.moreover.com/click/here.pl
?r35453156
LENGTH = 8418

----- STARTING FOREIGN PAPERS ----
The Guardian - England -
http://www.guardian.co.uk/uklatest
Retrieving foreign links...
Finished links

Comparing to foreign stories...

Title = Property Boom Establishes
Duke As Britains Richest Person
Words = 255 -->> Matching Desc
Words: 1 / 38 -->> Percentage =
0.027 %

Title = British Womans Thailand
Death Suspicious - Police
Words = 699 -->> Matching Desc
Words: 1 / 65 -->> Percentage =
0.012 %

Title = Israeli Forces Surround
More Villages Despite US Warning
Words = 285 -->> Matching Desc
Words: 6 / 27 -->> Percentage =
0.222 %
************** ----->>>> MATCHED!

Title = Cctv Footage Released Of
Missing Milly
Words = 724 -->> Matching Desc
Words: 1 / 48 -->> Percentage =
0.023 %
```

```
Title = Britain is losing too much
sleep
Words = 305 -->> Matching Desc
Words: 0 / 19 -->> Percentage = 0 %

Title = Peace Campaigners Warned To
Leave West Bank
Words = 292 -->> Matching Desc
Words: 3 / 24 -->> Percentage =
0.125 %

Title = William And Harry Tell Of
Queen Mothers Ali G Impersonation
Words = 274 -->> Matching Desc
Words: 0 / 18 -->> Percentage = 0 %

Title = Palestinians Ask Arabs to
Break Ties With Israel
Words = 815 -->> Matching Desc
Words: 12 / 62 -->> Percentage =
0.196 %  ************** ----->>>>
MATCHED!

Title = Spitfires, hymns - but no
Elton at the funeral
Words = 1029 -->> Matching Desc
Words: 2 / 73 -->> Percentage =
0.027 %
```

Note that for the sake of brevity, only the first nine links out of the 34 eventually examined were reproduced in the printed example above. The link retrieval and examination process is long and tedious, and does not lend itself particularly well to written reproduction, but the above example should at least give a feel for how the system proceeds. In the example, *Extra!Extra!* identified two apparently relevant stories from all that appeared on England's Guardian newspaper web site. Both ended up being relevant.

This process, as illustrated above, continues until every link from the foreign paper's index page has been examined. Then the next foreign paper in the list of relevant ones is examined, and the link retrieval process begins anew.

When all of this concludes, we are left with:

- the current headline news story in its bare bones format (text and images)

- a series of stories, taken from foreign newspapers, that concern the same subject matter as the original news story. Each of these stories is also in its bare bones format.

When this process had been successfully completed, our system still lacked one major component—any semblance of an archive system.

3.5 COVERING THE PAST

THE VERY NATURE OF NEWS is changeable. Reporters are not sentinels of ultimate truth, and no matter how unbiased they strive to be, they can never tell the whole story at first glance. News evolves over time, as does reporting, and often the most telling interpretations of a story are those gleaned from prolonged coverage. The major shortcoming of our system as it stood was that it only covered the present tense. As news stories would break, it would retrieve them, parse them, and go out to find relevant foreign stories, but the process would stop there. There was no sense of the past. Users had no way to follow the course of a given story over hours and days. We sought a way to change that.

It became apparent that our system's preexisting structure would lend itself to developing an archive system. Because each story on the server has a definition file associated with it, the archive creation system could be based simply on the comparisons of various story definition files. It could use the same borderline comparison ratio of 14% (that which was used in comparing foreign stories to original

news stories), and no files would have to be created or altered. Because no new URL's would have to be retrieved, all of the action could happen on the server.

The strategy seemed like a good one, so we added this third step to the process. For each new story retrieved, after developing a definition file and finding its relevant foreign links, we scanned the server-side directory structure for any "parent" stories that might match this new one. We used 14% as our threshold words-in-common comparison ratio, and we ended up with a process that retrieves three things for each current headline news story that initially comes through in the RSS news feed:

1) the current headline news story in its bare bones format (text and images).

2) a series of related foreign stories, also in their bare bones format, that provide "other sides to the story".

3) a listing of related stories from the past seven days that form the subjective timeline of "how this story developed".

With these three components in place and functioning properly, we merely had to add them to the uniform *Extra!Extra!* design template and publish them as HTML pages.

3.6 MAKING THEM PUBLIC

AFTER TEARING THROUGH a series of initial design sketches and control flow diagrams, we finally settled upon a desirably coherent visual design. We produced the graphical components using *Adobe Photoshop*, optimized them using *Adobe ImageReady*, pulled them together with HTML using *Macromedia Dreamweaver*, and then examined the HTML source code of the resulting files.

We manually added some dynamic content using JavaScript. We created a "Site Last Updated" routine that would graphically indicate the last time (on thirty minute intervals) that new headline stories were retrieved that day. We display this timestamp at the top of the page's right sidebar, reminding users of the freshness of news on *Extra!Extra!*.

Once we had developed *Extra!Extra!*'s visual design, we copied its resulting source code and pasted it into our Perl script. With the HTML structure broken down into many distinct text strings, the Perl functions could easily fashion uniform *Extra!Extra!* "template" pages for each new story processed. This step guaranteed visual consistency, and transformed the raw textual data into compelling HTML documents.

Once the HTML files are written, they are copied to a globally readable and executable `public_html` directory. In this manner, public users can access the HTML pages containing the collected news stories. They cannot, however, access any of the server-side processes that produce said text files. The system structure is removed and protected, while the user sees only what he needs to see.

If for some reason, the public HTML section of *Extra!Extra!* were to fall victim to a hacker attack, a virus, or some other unforeseen offensive, all would not be lost. Complete copies of all relevant data are housed at another location, at "`/jjharris/thesis`" and they could be summoned to replace any missing data from the public HTML side of the system.

3.7 HOW OLD IS TOO OLD?

WHILE HAVING AN archived backup system is crucial, it seemed cumbersome to save every collected news story forever. As we considered the question, however, the issue of space became less and

less of a factor. The news files stored on the server are just text strings, consuming a minimal amount of space, and the images displayed on *Extra!Extra!* are not stored at all—they reside at their source location, and our system links to them.

This expectation of minimally consumed space by simple text strings was proven incorrect by observing the system in action over the course of several days. The file size of the "text strings" ballooned much more quickly than we anticipated, eventually peaking at about 30 megabytes per day. The archive size increased so rapidly that after eight days, our disk quota had been exceeded, and *Extra!Extra!* ceased to function until the disk quota was increased.

A look at the numbers justifies these observations:

(48 original stories per day) x (20 related foreign stories for each original story) = 960 stories

(960 stories) x (13 k per file) x (2 {for the HTML version}) = 24.96 Megabytes per day

Although rapid, this size growth levels off after 14 days, which is how long *Extra!Extra!* saves stories in its archives. The upper bound archive size appears to remain around 400 Megabytes, which we decided is manageable. Besides the issue of storage space, we arrived at other reasons not to save every news story in our archives forever.

Most notably, one of *Extra!Extra!*'s main components is the "How This Story Developed" list. As described in Section 3.5, this feature allows users to trace the evolution of a given story over hours and days. We stress this feature as one of our system's most important, and its effectiveness is directly derived from its ability to report only the most relevant past.

Some stories evolve over the course of a week. For instance, a fugitive might escape and rob a bank in Dusseldorf. The authorities might follow him for a few days, and finally catch him four days later. By that night, he is back in custody, and the case is closed. A five day timeline relates the entire story—perfect for *Extra!Extra!*.

Other stories evolve over the course of weeks and months. Picture another scenario. A bomb goes off in Israel, detonated by a Palestinian suicide bomber reacting to the Israeli occupation of the West Bank. This event might correspond closely with 27 other suicide bombings over the course of the last two years. Those, in turn, might each correlate with United States involvement in the Middle East, in British interests in Saudi Arabian oil fields, and so on. Yet surely, we would not want *Extra!Extra!* to report each of these semi-related stories that have occurred over the course of 24 months. The "How This Story Developed" list would be lengthy and unmanageable. Users would be intimidated by the mass of information, and would probably feel alienated from the web site. When we considered examples like this, it quickly became apparent that we had to impose some sort of a time constraint on how long a given news story can be saved on the server and compared to new current news stories. If the time constraint were too short, the sense of evolution would be lost. If it were too long, the user would be turned off by the glut of information that is only semi-relevant. Few users care to read accounts of events that happened six months ago, and in truth, few users even care to read accounts of what happened a week ago. People are mostly concerned with the here and the now.

Based on such scenarios, we chose the period of one week as a reasonable upper bound on time. After a story has been on the server for seven days, it is no longer considered for the

"How This Story Developed" listings of new stories. After the initial story has been on the server for 14 days, it is discarded entirely. The seven-day "grace period" ensures that no broken links will occur. For example, if a story were totally discarded from the server after seven days, all of the six-day-old stories could still contain "How This Story Developed" links to the recently discarded story from one day earlier. This scenario would result in ugly broken links that put forth an unprofessional image. To avoid such situations, we allow for the seven-day grace period.

Because of the way we implemented our site structure, no links need to be updated when an old story is discarded. New stories have the potential to link to old stories under the "How This Story Developed" section, but old stories have no way to link to new stories. The linkage system is a one-way process. This allows clean deletion with no broken links when an old story is deleted.

This "garbage collection" process is initiated at run time, before the script begins to retrieve any new headline stories from the RSS feed. Although it only needs to occur once daily, it seemed simpler to include the quick garbage collection process at the front of the script.

4

PROBLEMS & PITFALLS

4.1 PROGRAMMING QUIRKS

IN THE STORY DEFINITION function that takes an HTML page and outputs a list of words to describe that page, we encountered a peculiarity that took several days to solve. While applying the HTML parsing function, one of the string simplification routines calls for the removal of all newline characters, denoted by "`\n`". Once all newlines have been deleted, we proceed to tokenize the file around whitespace characters. For the most part this worked, but for certain specific news sources, strange output ensued. Lines would be cut off at strange increments, huge sections of data would be lost, and it seemed impossible to discern what was happening.

The problematic files in question were littered with dozens of "`\r`" characters. We discovered this by counting the visible characters in a text file, and then calling the function "`length()`" on the same text string. The returned value from "`length()`" was greater than the number of characters we could count on screen. This signified to us that there were "hidden" characters in the file that were meddling with the HTML tokenizing routine. In the file's ASCII contents, we found these strange "`\r`" characters, and realized that they are "return" characters—the Macintosh version of "`\n`" (newline). It became clear that certain news sites had been developed on PC's, and others on Macintosh computers.

4.2 SLOW SERVERS

PERHAPS THE SINGLE most significant obstacle in the creation of *Extra!Extra!* was the resolution of slow calls to Perl's "`GET()`" function, which retrieves HTML source code from a given URL. Our system is automated to run every thirty minutes, and with each iteration it has to retrieve and examine a potentially huge number of individual web pages. This number depends on how many countries are mentioned in the source article, and therefore how many foreign newspapers must be scanned. Each of these foreign newspapers has an unknown number of links on its main page, and this number can vary greatly. For instance, *The Vietnam News* has 11 links on its main page, while the *LA Times* has 227.

Certain servers, especially those on the other side of the globe, are often slow in responding and occasionally give no response at all. When this occurred, our system would hang on a single call to "`GET()`", sometimes indefinitely. Waiting for the "`GET()`" call to return, *Extra!Extra!* would remain effectively frozen, unable to process any other links. This routine had the potential to eat away the entire 30-minute cycle, which would be detrimental to our system. We had to find a way to kill off exceptionally slow calls to "`GET()`".

Initially, this problem appeared trivial, and it seemed that Perl must have some qualifying addition to its `LWP::Simple` "`GET()`" function that could indicate a maximal timeout value. Surprisingly, this is not the case.

At first, we tried to invoke Perl's pseudo-C signal-handler mechanisms, using an alarm of ten seconds to kill off any slow calls to "`GET()`". Our attempt to do so looked like the following:

```
eval {
       local $SIG{ALRM} = sub { die
"GET has Timed Out" };
       alarm 10;
       $HTML = GET($URL)
       alarm 0;
};
if ($@ and $@ !~ / GET has Timed
Out/) { die;  }
```

A strange subtlety emerged. The above code works perfectly for any normal function call. Assume that, instead of "`GET($URL)`", the function being timed was one that merely sleeps for twenty seconds, and then returns. In such a case, the alarm sounds after ten seconds, and the program dies, appropriately. But the behavior of "`GET()`" is different.

For some reason, the "`GET()`" function that exists in Perl's `LWP::Simple` module overrides the signal handler, and therefore manages to ignore the alarm when it sounds. The "`GET()`" function keeps working, stubbornly, and refuses to die even when the alarm tells it to. For this reason, the pseudo-C signal-handler method did not work, and we had to look elsewhere.

Our second approach to the problem was to create two separate processes using "`FORK`", and to have the child process make the "`GET()`" call, as the parent process looks on, waiting for ten seconds to expire. Since the child process is its own independent entity, we had to introduce a `TEMP` file to hold any retrieved data for the parent to examine later. The parent waits ten seconds and then checks if the child has written to the temp file. If the `TEMP` file has not been written, the parent kills the child with "`kill 9, $pid;`" If the child has written to the `TEMP` file, the parent reads the retrieved data and begins the examination process.

In the hopes of avoiding any overprotective

Perl abstractions, we decided to break down the "GET()" call as much as we could, reducing it to the socket level. This process worked well, except that it caused every single link, no matter how speedy, to consume ten seconds of processing time. Some quick math reveals the massive time allowance required.

(6 foreign newspapers) x (100 links each) x (10 seconds per link) = 6,000 seconds
(6,000 seconds) / (60 seconds per minute) = 100 minutes

When each program cycle is only thirty minutes, this technique is clearly unacceptable.

To solve the problem, we placed the parent's "SLEEP" function into a "for" loop that sleeps for two seconds and then repeats until either the TEMP file has been written or ten seconds have been consumed. This technique was better, but still not optimal. After all, some links take only a fraction of a second to return, so why give them a minimum of two seconds? Additionally, the TEMP file checking routine was inelegant and unwieldy. There was the potential of a transmission error, of a simultaneous update and access, and the whole method generally tasted of "hack".

We settled on a technique that is more concise, more coherent, and clearly Perl's desired way of dealing with slow calls to "GET()". It utilizes Perl's LWP::UserAgent module, which essentially opens a pseudo browser within Perl, sets a maximal "timeout" value for the browser, and then retrieves HTML pages as specified. Ultimately, our solution to slow "GET()" calls is the following:

```
$browser = LWP::UserAgent->new();
$browser->timeout(10);

my $URL = @_[0];

my $request = HTTP::Request-
>new(GET => $URL);
my $response = $browser-
>request($request);
if ($response->is_error()) {
      printf "%s\n", $response-
>status_line;
}
$contents = $response->content();
```

Our desired timeout of ten seconds is set, and then the browser is asked to retrieve the given URL before ten seconds expires. If it fails to do so, the program cancels the call and awaits new input.

We embraced this "final" solution with a minor leap of faith, assuming that the promised behavior equals the actual behavior. When undocumented abstractions exist between the code and the system level actions, one can never be sure exactly what specifications are triggering what behavior. From all the testing we could manage, this third solution seems to perform well, but without access to the sockets and signals (as in our first two techniques), it is impossible to ascertain exactly why the actual (positive) behavior is happening. We decided to take it on faith and reasonable testing that this third method is sufficiently robust.

4.3 INTELLECTUAL PROPERTY

IN INITIALLY CONCEIVING this system, we assumed that the final product would violate copyright law in some way, but we decided to continue nevertheless. The potential for *Extra!Extra!* to be a significant educational landmark for innocent non-profit use outweighed the understanding that some parts of this system would verge on copyright protection issues.

It seemed, furthermore that Internet law is such a cloudy and constantly evolving entity

that it would be foolhardy to abandon this entire project because of it. All cited news sources are fully credited with their full name, logo, and web link. *Extra!Extra!* is a free service that does not benefit in any way from the reproduction of previously published news.

Additionally, existing Internet law does provide several apparent loopholes for educational entities conducting work for research purposes. As explained to the faculty of Stanford University in an October 30, 1998 memo from Condoleeza Rice, then Stanford's Provost, the Doctrine of "fair use" permits certain academic entities, under certain conditions, to circumnavigate existing copyright law:

"The 'fair use' doctrine allows limited reproduction of copyrighted works for educational and research purposes. The relevant portion of the copyright statue provides that the 'fair use' of a copyrighted work, including reproduction 'for purposes such as criticism, news reporting, teaching (including multiple copies for classroom use), scholarship, or research' is not an infringement of copyright. The law lists the following factors as the ones to be evaluated in determining whether a particular use of a copyrighted work is a permitted 'fair use,' rather than an infringement of the copyright:

- the purpose and character of the use, including whether such use is of a commercial nature or is for non-profit educational purposes
- the nature of the copyrighted work
- the amount and substantiality of the portion used in relation to the copyrighted work as a whole
- the effect of the use upon the potential market for or value of the copyrighted work

Although all of these factors will be considered, the last factor is the most important in determining whether a particular use is 'fair'" [RICE].

The "fair use" legal doctrine, as outlined by Ms. Rice, appears to grant *Extra!Extra!* the space it needs to exist. The first point, that the work be for non-profit or educational purposes, is met by *Extra!Extra!*. The second point, about the nature of the copyrighted work, seems to apply as well, given her earlier reference to "for purposes such as… news reporting." The third point, about the amount of copyrighted material being reproduced, seems to comply, as *Extra!Extra!* merely reproduces single news articles and photographs, and not entire publications. The fourth point seems to check out as well, as all news sources covered by *Extra!Extra!* are free access entities in the first place. Our system provides an alternative viewpoint, but does not explicitly attack the market share occupied by existing news sources.

5 REAL WORLD TESTING

We automated our system using standard UNIX Cron jobs, and watched it perform over multiple days to gauge its success and look for bugs.

5.1 SHORT AND UNRELATED

OTHER THAN SEVERAL minor errors that took quick fixing, the main issues arose in the "Other Sides to This Story" section. One early example involved a source article from CNN discussing the arrests of several hundred plotters in Kabul, Afghanistan. The system had retrieved an article from Iran's *Tehran Times* about the switch to daylight savings time, and marked it similar to the CNN piece. Clearly, the two should not have been linked together. The reason for this discrepancy was that the *Tehran Times* piece was a brief news flash—76 words, 335 characters. Its description file consists of only five words—"`iran`" three times, and "`tehran`" twice. When being compared to the original article from CNN, those two words held enormous weight, and the system deduced that, both being about Iran and Tehran, the two pieces must be similar. To remedy such undesirable situations, we introduced a measure that discards all articles shorter than 1,500 characters (about 250 words). This ensures that any piece considered by *Extra!Extra!* will have substantial content, and that it will not improperly trigger a similarity decision.

5.2 MISSING OUT

IN FURTHER TESTING we noticed that some stories were just not getting covered. For instance, the anti-Semitic attacks on French Jews that followed Israel's military occupation of the West Bank were covered by CNN and *The New York Times*, but *The International Herald Tribune* (our designated French news source) returned nothing. We manually visited *The Tribune* later that day and found ample mention of the attacks on French Jews, which puzzled us. We realized that *Extra!Extra!* occasionally gets a headline story from CNN, etc. immediately as the news breaks. Therefore, our system does its searching for related foreign articles very early in the story's development. At times, it is so early in the game that the foreign newspapers have not yet had the chance to update their sites with the newest news. When this occurs, *Extra!Extra!* archives the article in question with no foreign links, and when the foreign papers do get around to updating their sites, it is already too late, for the link retrieval process has already occurred. This borderline condition is less than ideal, but most stories develop over time, so subsequent headlines from the RSS feed will most likely initiate additional rounds of the article for *Extra!Extra!* to examine. This way, most stories are eventually covered by our system.

5.3 USER REACTIONS

SINCE *EXTRA!EXTRA!* was designed to be a tool for the common non-expert user, we conducted a series of user response tests to gauge the usability of our system. These tests were conducted informally on around 30 individuals, mostly Princeton students ranging in age from 18-22. A few additional tests were conducted on older users, in their late fifties.

—5.3.1 CATEGORIES

MOST USERS FOUND the system straightforward to navigate. They found that the main headline list was an intuitive way to break down the news. The most common criticism of the headline list interface was that it made no effort to categorize the news. Some users wanted different sections—"Sports", "Politics", "War", and "Culture." Such categories are staples of typical news web sites, and *Extra!Extra!* would do well to have similar sections, they argued.

Interesting problems would arise in trying to build this modification into our system. The program would have to draw conclusions on a given story's meaning, and classify it a certain way. This could probably be accomplished by noting the link's original location on the news source web site. Different sites use consistent titles or tags to demarcate sections of their page, and we could note the section in which a given link is located to decide its subjective affiliation. To do so, however, would mandate abandoning the Moreover RSS feeds, and creating our own feeds instead. When using the Moreover feeds, the provided link goes directly to the article's text page, and never shows the link in its original context on the site's main page. For reasons discussed in Section 3.1, we opted to stay with the Moreover RSS feeds and abandon the idea of categorizing news stories.

—5.3.2 LET ME NAME MY PAPER

SOME USERS WONDERED why they could not specify a given paper to be searched for a related story. For example, if there was a story about terrorism in Jordan that did not contain the word "England", how could users find out what the British had to say about the whole ordeal? Currently, *Extra!Extra!* does not support the coverage of foreign newspapers whose host country is not

explicitly mentioned in the body text of the article. To cover all 67 listed newspapers for each headline story would yield a potentially enormous number of links, many of which might have nothing to do with the story in question.

Other users suggested we provide a drop down menu containing every searchable country, allowing them to ask the program to search a give country or newspaper in real time, as they wait. This option is more viable than the first, but depending on the foreign news source and the number of main page links it contains, the search process can take anywhere from a few seconds to a few minutes. To ask a user to wait several seconds would be reasonable, but waiting several minutes would be unacceptable to most users, so we did not include the real-time search capability.

—5.3.3 WHERE'S THE VARIETY?

MANY USERS COMMENTED that the stories covered by *Extra!Extra!* were too similar in subject matter. In the testing period of early April 2002, nearly all of the stories on *Extra!Extra!* had to do with the Israeli and Palestinian conflict in the Middle East. On Friday, April 19, 2002, 16 out of 24 total *Extra!Extra!* stories concerned the Middle East. These observations highlight the apparent existence of several interesting social trends.

The reason so many *Extra!Extra!* stories in April 2002 had to do with the Middle East reflects the proportion of source news stories that had to do with the Middle East. There is no darker scheme at play here—*Extra!Extra!* simply catalogues what it finds at CNN, *The New York Times*, BBC, and *The Washington Post*. If these news sources choose to cover a single topic (the Middle East) more often than others, our system will reflect that choice. In this sense, the homogeneity of *Extra!Extra!* stories provides interesting commentary on the business of news in America and Britain.

In such a fast-paced world, teaming with information, it takes a lot to hold people's attention. In America and Britain, news is a free market business that exists for profit. To attract and maintain people's interest (and therefore turn profit), news organizations need to create a sense of urgency—the feeling that there is something to follow, something to watch. The best way to do this is to monumentalize certain stories and fool people into caring about them. The nature of the story is less important than the way it's presented. Sporadic coverage of many topics is less compelling than the detailed coverage of one apparently soap-opera-like story. Witness OJ Simpson, Elian Gonzalez, Monica Lewinsky, Gary Condit, and the summer shark attacks—all stories that gripped America until something more important came along. The media succeeds when it can sell magazines and air television specials, so the media rallies around given stories to help themselves attract readers and viewers. If there is national consensus as to what the cover of *Time* and *Newsweek* should be, then people consider that story, whatever it is, to be important enough to follow. For the sake of creating hype and driving the market, the media relishes a good story.

This hypothesis seems to be supported by the number of Middle East stories retrieved by *Extra!Extra!* in April of 2002. The issue is not whether or not the Middle East stories are important—they surely are—but the issue is that there were other things of importance as well, and *Extra!Extra!* picked up a proportionally scant amount of these other stories. The persecution of French Jews, the molestation cases surrounding Catholic priests, the controversial election of French President Jean Marie Le Pen, and the violent revolt against Venezuelan President Hugo Chavez were a

few news stories that received little to no coverage by *Extra!Extra!*, and therefore by the mainstream American and British media.

These observations could form the basis for significant social research, but further discussion of them is beyond the scope of this paper.

6 FUTURE WORK

WHEN DEALING WITH a subject as multinational as news, there are myriad ways to reinvent and improve the system. The most obvious next step would be to incorporate a breed of translation algorithms into our system. The biggest shortcoming of *Extra!Extra!* Version 1.0 seems to be the relative lack of representation of certain countries in certain areas around the world. All of South America, for example, has only one single represented newspaper — *The Santiago Times* of Chile. South America does have other online newspapers, but only *The Santiago Times* publishes an English version. We cannot expect every country to convert their national news into English. If we care about their national news, then the onus is on us to understand it. For these reasons, a reliable translation algorithm would add immeasurably to *Extra!Extra!*'s credibility and breadth.

The Babelfish algorithm, which is free and available through AltaVista, would be an obvious first step, but one that would probably prove to be insufficiently accurate, judging from preliminary testing. When a human reads the finished output, the Babelfish algorithm works moderately well. Humans have the semantic sensibility to discern the proper meaning from subtly mistaken output, but *Extra!Extra!* requires that a program reads and "understands" (makes definition files for) these translated pages. That process depends so heavily on the exact comparison of specific words that a rough translation would most likely render the story description process difficult or impossible.

It would seem that the expatriate community around the world, often responsible for the publication of foreign English language newspapers, would share a more homogenous view of the news than their sometimes more radical local counterparts. If this plausible assumption is true, then the English language newspapers that they produce (and we cover) would tend to be less controversial and less enlightening than the newspapers produced in that same country's native language. To cover these foreign language newspapers would surely result in an even more effective range of represented opinion, serving to foster a more comprehensive *Extra!Extra!*.

Another way to improve the *Extra!Extra!* process would be to modify the HTML parsing algorithm. Currently, as mentioned in Section 3.2, we manually identify "starter" and "ender" strings for each foreign newspaper. These strings are unique to each foreign news source, and they bracket the relevant article body text on a given sub-page of that web site. Our program seeks and indexes these strings, throwing away all the text before the starter and after the ender. This process works admirably for most foreign papers, but some of the less well established news sources fail to implement anything as regular as "starter" and "ender" strings on their web sites. With no "starter" and "ender" strings in existence, our system cannot extrapolate the body text from such pages. In this sense, *Extra!Extra!* fails to represent the more primitive news sites around the world. These sites still have the potential to harbor insightful local opinions, but simply because their web designers choose not to adhere to a rigid structure, their web sites are passed over and ignored. This lack of coverage is a definite shortcoming of *Extra!Extra!*, and one that could be resolved in the future. If we could develop a way to implement our HTML parser routine without the use of "starter" and "ender" strings, our system would be a stronger one. The obvious default solution would be to use the "`<body></body>`" HTML tags, but those would offer only a very rough approximation of the article's relevant text, and a great deal of garbage would probably be recorded along with the desired body text.

The issue of RSS feeds has been rigorously explained and explored already in this paper. We chose not to implement our own RSS feeds (and stay with Moreover Technologies instead) because we wanted to maintain coverage of the subscription-only *New York Times*. Using Moreover's feed allows our scripts to view the pages of *The New York Times*, whereas if we had written our own RSS feed, we would not be able to access that internationally respected newspaper. We felt that coverage of *The New York Times* was essential enough to warrant our continued partnership with Moreover. In the future however, *Extra!Extra!* would be a more self-sufficient and therefore more robust system if it were to possess its own independent RSS feed with its own paid access to *The New York Times*. In the interest of cutting cost, this step was purposely skipped in Version 1.0 of *Extra!Extra!*.

7

CONCLUSION

WE HAVE SUCCESSFULLY completed everything that was initially predicted, in addition to adding in relevant photographs for each story, a feature that was not planned in this project's initial proposal.

We have created a system that is robust and consistent, updating itself automatically every thirty minutes with the latest headline news stories from the four most prominent news sources in America and Great Britain—*The New York Times*, *The Washington Post*, BBC, and CNN.

For each news story gleaned from those four initial sources, the HTML parser routine reduces the article to relevant text and images, discarding everything else. The system creates a short list of words to describe the content of the article, and then stores that description file on the server for future referral. The article is then scanned for names of countries around the world. Each country name corresponds to one or more predefined online English language newspapers published in that country. Once this list of foreign newspapers is formed for the given article, the system proceeds to scan each foreign news source for stories that are similar to the initial one. In this manner, *Extra!Extra!* produces a list of "Other Sides To This Story".

Once the foreign stories have been collected, the system searches all archived stories already on the server, looking for similar ones. In this manner, *Extra!Extra!* decides "How This Story Developed".

This information is pulled together and presented succinctly on the *Extra!Extra!* web site. People can access this site for free, and use it to read different and potentially conflicting sides to current news stories. When people tire of the homogeneity of CNN and its cohorts, they can turn to *Extra!Extra!* and instantly discover what the rest of the world is saying.

We have succeeded in creating an online system that accurately provides "All the Sides to Every Story" in close to real time, composing its content from over 65 unique news sources from around the world.

Our system is more comprehensive than any existing news site. It resists bias, political obligations, corporate parenting, editorial pressures, and governmental allegiances. By having no editorial personality of its own, *Extra!Extra!* can assume the diversity of all its parts—disparate opinions from all across the globe.

By providing its users with all the information that exists, *Extra!Extra!* refuses to dictate the truth. It considers its users to be autonomous, intelligent beings who can formulate their own opinions based on succinctly presented evidence. By presenting "All the Sides to Every Story", *Extra!Extra!* helps users discover the one side that matters most.

The fully functioning version of *Extra!Extra!* can be viewed online at:

http://www.cs.princeton.edu/~jjharris/thesis

REFERENCES

[1] Ball, Chris. Create RSS channels from HTML news sites. http://www.perl.com/pub/a/2001/11/15/creatingrss.html, Viewed on March 24, 2002.

[2] Beare, Kenneth. English as a Foreign Language Site. About.com, http://esl.about.com/library/courses/blcourses_advanced_vocabulary.htm, Viewed on February 4, 2002.

[3] Bonhomme, Patrice. Word Frequency Lists. http://www.loria.fr/~bonhomme/sw, Viewed on February 7, 2002.

[4] Brin, Sergey & Page, Lawrence. The Anatomy of a Large-Scale Hypertextual Web Search Engine. Computer Science Department, Stanford University, Stanford, CA, 1998.

[5] Deitel, H.M. Perl: How to Program. New Jersey: Prentice Hall, 2001.

[6] Farrely, Glen. History of Search Engines. http://webhome.idirect.com/~glenjenn/search/history1.htm, Viewed on March 25, 2002.

[7] Guelich, Scott. CGI Programming with Perl. California: O'Reilly, 2000.

[8] Hall, Marty. Core Web Programming. New Jersey: Prentice Hall, 1998.

8

[9] Jupiter Media Metrix, as cited by CNN.com/SCI-TECH, on June 5, 2001. http://www.cnn.com/2001/TECH/internet/06/05/internet.consolidation, Viewed on March 28, 2002.

[10] Rice, Condoleeza. Copyright memo to Members of the Faculty, Hoover Institution Fellows, Academic Staff, and Library Directors. October 30, 1998. http://fairuse.stanford.edu/rice.html, Viewed on April 7, 2002.

[11] Sherman, Chris. Search Day. http://searchenginewatch.com/searchday/01/sd1219-links.html, Viewed on March 25, 2002.

[12] Whitehead, Paul. Perl. California: IDG Books, 2000.

HONOR CODE

I **PLEDGE MY HONOR** that I have completed my Senior Thesis in concordance with Princeton University's Honor Code.

—Jonathan Jennings Harris

9