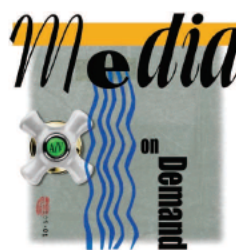


Streaming-Media Knowledge Discovery



As the amount of streaming audio and video available to Web users grows, tools for analyzing and indexing this content will become increasingly important.

*Jan Pieper
Savitha
Srinivasan
Byron Dom*
IBM Almaden
Research
Center

The proliferation of powerful Web-based search engines that efficiently index and search large document collections has made it easier for the casual user to search and browse text documents on the Web. Although several portals provide comprehensive text information, we have yet to see a similarly comprehensive resource list for locating relevant information in streaming media, whether via search engines, Web directories, or portals.

Yet the amount of Web-based streaming media grows continually. For example, a recent study estimates that live media channels on the Web number in the thousands.¹ In addition, advances in storage, computing power, and bandwidth have resulted in increasing deployment of streaming video in applications such as workplace training, distance education, entertainment, and news.

The Multimedia Knowledge Discovery group at IBM Almaden Research Center explores technologies that help locate streaming media on the Web, build effective indexing and classification tools for streaming media, and develop compelling streaming-media applications. The "What Is Streaming Media?" sidebar explores this emerging field in detail.

Our ongoing work, motivated by the Web's role as an ever-increasing information source and knowledge repository for many applications, focuses on locating, analyzing, and indexing Web-based streaming media. Frequently, knowledge management applications and information portals synthesize unstructured text information from the Web, intranets, and part-

ner sites. Given this context, we crawl a statistically significant number of Web pages, detect those that contain streaming media links, crawl the media links to extract associated metadata, then use the crawl data to build a resource list for Web media. We have used these crawl-data findings to build a media indexing application that uses content-based indexing methods.

LOCATING STREAMING MEDIA

A streaming-media link can point directly to streaming media or to a metafile, which contains one or more streaming-media links. We define a streaming-media page to be any Web page that contains a direct link to a streaming-media file or metafile. We used a statistical sampling approach to estimate the number of streaming-media pages on the Web. After crawling a significant number of Web pages and computing the streaming-media statistics on that sample, we estimated the numbers for the entire Web within a certain confidence interval. By doing so, we sought to detect streaming-media pages and study several properties related to streaming media, such as the percentage of Web pages containing streaming-media links, the distribution of links to streaming-media files, and the top-ranking categories of streaming-media pages.

Streaming-media page detection

Web pages can link to streaming media in several ways. To analyze the Web-page content, we focus on detecting both direct links to streaming media and

embedded streaming-media players. We currently ignore sites that use JavaScript to construct links to streaming media because finding such links would require parsing the JavaScript by simulating proper user input.

We view the detection task as a binary classifier function $f(x) = y$, where $x_1, x_2, \dots, x_d$ is a *feature vector* for a given document. The individual features x_i are a combination of terms that include media file extensions, plug-in syntax, and comment strings. Thus, x represents a document and $y = \{1, 0\}$ represents discrete target values, where 1 is a streaming-media page and 0 is not a streaming-media page. A single feature vector of weighted terms represents each page. We developed a simple Boolean classifier that uses a deci-

sion list with a strong empirical inductive bias toward the terms we observed to be high-confidence indicators of streaming media.

Our findings, obtained from analyzing 60 million Web pages, indicate that the RealMedia format dominates our sample data set with a 65 percent share. Only 11 percent of the pages host Windows Media Player files, and 24 percent host other formats (MPEG-1, QuickTime, SMIL). Given RealMedia's pervasiveness, the rest of our link analysis focuses on that format.

We assembled a test set of 400 pages, including both RealMedia and Windows Media Player pages, to manually verify our streaming-media detection algorithm. Table 1 shows the algorithm's accuracy.

What Is Streaming Media?

From a user's perspective, *streaming* video is viewable immediately after downloading starts. Some, however, use the term to refer only to real-time content streamed through the user datagram protocol (UDP). Others use the term to refer to any video that plays on the Web.

In reality, true streaming only works when there is enough bandwidth to play the video in real time. In terms of transport protocols, a streaming-media server uses Internet protocol (IP) multicast, real-time streaming protocol (RTSP) incorporating UDP or TCP, or TCP alone—a hybrid strategy with real-time streaming via HTTP if a firewall cannot pass UDP data.

Our work focuses on media formats that can stream regardless of the protocol used. The three main contenders in the streaming-media space are Apple's QuickTime, Microsoft's Windows Media Player, and RealNetworks' RealMedia. A recent survey of streaming-media technologies, conducted by Nielson/NetRatings, reports that RealNetworks is in the lead, followed by Windows Media Player.¹

Retrieval techniques

Researchers classify multimedia retrieval into expression-based and semantic approaches. Expression-based techniques rely on an example or a physical description of the information sought, while semantic approaches rely on the actual media content.

Another characterization of multimedia retrieval focuses on query formulation and data representation. Using a common format for representing both the query and the data, such as an image to search an image database or an audio sequence to retrieve a segment of music, is an example of an expression-based technique. On the other hand, different formats for query formulation and data representation—such as a physical description of the required information or a textual query to search an audio and video database—are examples of both expression-based and semantic approaches.

Clearly, multimedia retrieval ultimately seeks to achieve fully

automated semantic retrieval. In recent times, possibly the most significant technological advance toward this ideal has been the application of automatic speech-recognition technology to multimedia content that contains speech.

Indexing

Much of the research on content-based indexing addresses indexing archived media, in which the archiving system performs the indexing as an essentially offline operation.²⁻⁴ Although developers can use streaming technology to present the indexing results on the Web, the original media that actually undergoes the indexing typically is not streamed. In these cases, streaming media provides a more efficient user interface for searching and browsing video without download and playback. This method imposes no requirements on real-time indexing performance or streamlining the process of decoding and encoding various media formats.

In contrast, live streaming-media indexing offers the challenge of decoding and encoding, indexing, and analyzing the media in real time because the stream must be treated as effectively infinite in length, thus precluding offline processing.

References

1. G. Lawton, "Video Streams into the Mainstream," *Computer*, July 2000, pp. 12-17.
2. A. Amir et al., "Using Audio Time Scale Modification for Video Browsing," *Proc. Hawaii Int'l Conf. System Sciences (HICSS 33)*, IEEE CS Press, Los Alamitos, Calif., 2000.
3. H. Wactlar et al., "Lessons Learned from Building a Terabyte Digital Video Library," *Computer*, Feb. 1999, pp. 66-73.
4. H. Zhang, P. Aigrain, and D. Petkovic, "Content-Based Representation and Retrieval of Visual Media: A State-of-the-Art Review," *Multimedia Tools and Applications*, vol. 3, Kluwer Academic, Dordrecht, the Netherlands, 1996, pp. 179-202.

Table 1. Streaming-media detection accuracy.

Media type	False positives	False negatives
RealMedia	11.7%	2.0%
Windows Media	0.0%	0.5%
Combined	11.3%	1.0%

Streaming-media link analysis

After detecting the streaming-media pages, we next sought data on the number of streaming media files and pages on the Web. We also focused on the links connecting these two entities because the link information reveals additional characteristics about the media pages' and media links' popularity. Our results include statistics on *out links*, the number of links on a given media page that point to streaming media, and *in links*, the number of links from media pages to a given media file.

We sought information on the distributions of these entities, for example, the fraction of media pages that contain only a single out link and the fraction that contain between 10 and 100 such links. In addition to in links and out links, we also computed the *hub scores* of the media pages in our sample and the *authority scores* of the associated media files²—these are two coupled, mutually reinforcing definitions. The hub score of media page u equals the sum of the authority scores of media files v that it links to, and the authority score of media file v equals the sum of the hub scores $h(u)$ of all the media pages u that link to it. We computed the hub and authority scores for two data sets:

- media pages and the corresponding streaming-media files, and
- Internet domains that link to or embed streaming media and the corresponding streaming-media files.

In the latter case, we treated the entire domain as a single entity, effectively collapsing all media pages in a domain into a single page.

Not surprisingly, the in-link and out-link distributions skew heavily toward the low end: one media link per page. This pattern for the in-link distribution may occur because most media files attach to a single introductory Web page. Other Web pages refer to that page rather than to the media file itself.

We also compiled similar analyses for hub and authority scores. Figure 1 shows the distribution of the hub-authority score combinations corresponding to media-page-to-media-file links. We observe that few pages have both a high hub score and media links that have a high authority score. The pages with a high hub score typically correspond to media archive sites

that contain a large collection of media links. The media files with high authority scores typically correspond to current items such as news clips that several pages point to. This trend is consistent with our in-link/out-link analysis and confirms that Web pages do not link heavily to media files.

To understand the primary domains that use streaming media, we manually categorized the top-ranking hub pages and media files. The nature of the hub-authority algorithm, combined with the link structure of the media pages and files collection we used, caused these results to be drawn from a single Internet domain: music. We therefore applied the hub authority score computation to the media pages per Internet domain, which resulted in a wider coverage of categories, as Table 2 shows.

Metadata crawling

Having gained a broad understanding of the link structure of streaming media, we wanted to further examine the attributes of the streaming media contained in the metadata associated with the stream. We used the proprietary application program interfaces provided by the streaming client to access the metadata contained in the encoded stream. This metadata includes information such as author, title, duration, and whether the stream being broadcast is live or on-demand.

We developed a metadata crawler for RealMedia that fetches each media stream and extracts the relevant metadata. The information from the metadata crawler provides a valuable resource for determining several significant attributes of streaming media, such as the media's average length, number of live broadcast channels, and so on.

We used a single server to store the list of all detected media links from the original Web-page crawl and to save the extracted metadata information. The metadata crawler client connects to the server to receive a link for which metadata is to be fetched. Using the RealPlayer ActiveX Control (<http://service.real.com/help/library/guides/index.html>), the crawler tries to open the link and, once it establishes a connection to the media stream, queries the stream. Having extracted the stream's metadata, the crawler sends the information back to the server and requests a new media link. For efficiency, the metadata crawler can run on several clients simultaneously. We observed that it often was not possible to stream the link because of problems at the streaming media server or because network traffic caused time-out errors.

We estimate that a few thousand live streaming-media channels existed on the Web at the time of our study, nearly all of which broadcast audio only. From this list, we selected a handful of live broadcast channels that we monitored in an information-filtering application that uses content-based indexing methods.

INDEXING LIVE STREAMING MEDIA

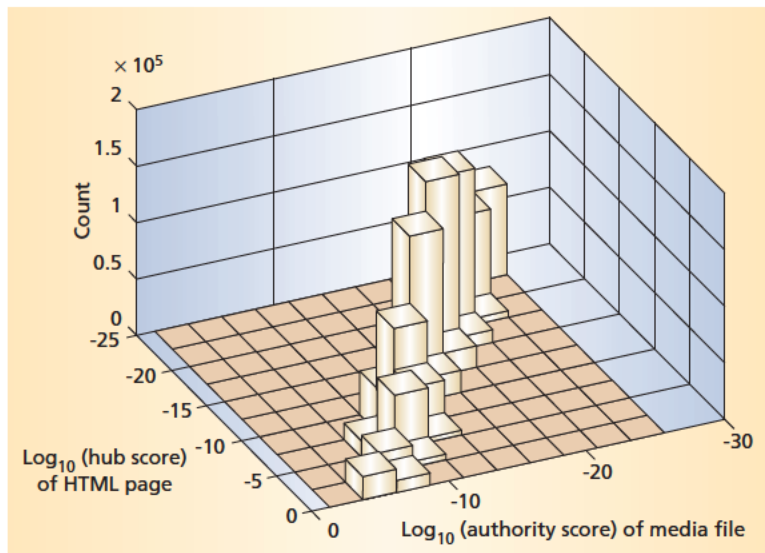
Indexing live streaming media does not afford the luxury of playing the stream multiple times during the process, as is possible with on-demand streaming media. The system must index the streaming media in a single pass, and in real time. To develop our indexing method, we first identified the types of information live media indexing can address.

Information needs

Live media content typically has a strong temporal component associated with it—being informed of a certain event is important. As an example, monitoring financial information on a live media channel to assist with business decisions involves a critical temporal component. We identify the information needs with respect to live media as follows:

- *Media on demand.* The user types a keyword query and the system retrieves the relevant archived media segments and makes them available for instant playback, starting from the time offset of the index match.
- *Delayed live.* Having determined that a media segment relevant to the query exists, a delayed live rebroadcast of the original content allows access to the entire contextual information preceding the relevant segment.
- *Live broadcast.* Using either a media-on-demand or delayed-live model to identify a media segment, direct access to the live media channel provides just-in-time information.
- *On the air.* This model queries the last 20 seconds of each channel so that a user can directly jump to the channel of interest. The last 20 seconds of any channel are not available through media-on-demand or delayed-live models because a delay exists between the live broadcast and when the system makes the broadcast available.

To address these different needs, we use two interfaces: one for queries and another for agents. The query interface lets users execute text-keyword searches on the live channels being indexed. The agent interface filters the information in the live channels by listening for user-specified text keywords. Users can issue a text query with either interface. If the interface finds a relevant match in a media segment broadcast a few minutes ago, the user can either listen to the media-on-demand clip or wait for the delayed rebroadcast, which will include contextual information preceding the match. In addition, the on-the-air mode queries the last 20 seconds of each media channel, thereby enabling an actual “live” query. If the query finds a relevant match, the user can jump directly to the live broadcast channel.



System architecture

Figure 2 shows the overall system architecture for an application that supports indexing live media streams. The system consists of three components: a back end composed of multiple speech recognition stations, application server middleware consisting of various servers that handle user requests, and a front end consisting of browser-based query interfaces and custom Java-based agent interfaces.

The back-end speech-recognition stations recognize the streaming-media content. The application server manages the indexing of the speech recognition output, the encoding of the buffered output into streaming format, and the retrieval functionality, while the front-end user interfaces provide the query functionality.

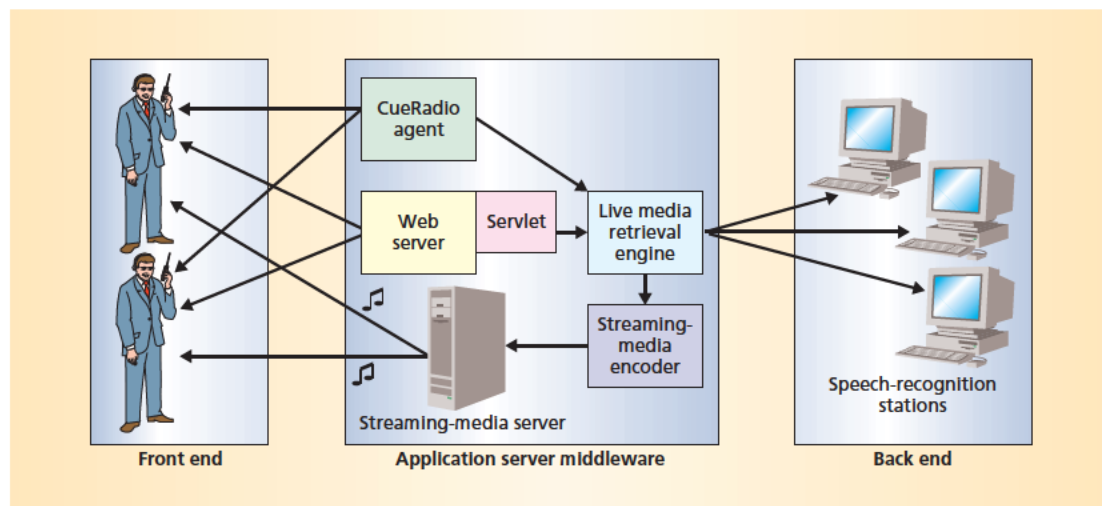
Multiple speech indexing stations drive the back end, accept streaming audio channels as input, and generate a time-aligned text transcript of the audio.

Figure 1. Histogram that shows distribution of hub-authority scores for media pages and media links. The scores reflect the concentration of media pages, their corresponding media files, and Internet domains that link to or embed streaming media and their corresponding files.

Table 2. Manually observed categories of top-scoring Internet domains.

Category	Hub Internet domains
Entertainment (includes music)	http://www.emusic.com http://www2.teenstation.com http://usatoday.hollywood.com
Sports	http://www.sportsline.com http://www.nba.com http://majorleaguebaseball.sportsline.com
News	http://www.rte.ie http://dailynews.yahoo.com http://fullcoverage.yahoo.com
Religion	http://islam.org http://www.jewishmusic.com http://www.calvarychapel.org

Figure 2. System architecture for an application that indexes live media streams. The system contains three components: a back end with multiple speech-recognition stations, application server middleware, and a browser-based front end with query interfaces and custom Java-based agent interfaces.



We use IBM's speech-recognition system, ViaVoice, with a broadcast-news language model to generate a time-aligned word transcript for each live channel being indexed. Average word error rates on these channels vary widely—between 35 and 60 percent—with word-deletion and word-substitution errors dominating.

The proprietary codecs that individual streaming technologies use proved a challenge for our recognition stations. Because streaming-media codecs are usually proprietary, we could not simply copy the media to a local machine for preprocessing and indexing. We overcame this challenge by directly piping the audio from the live media stream into our speech-indexing application. We then saved the automatically generated metadata, which consists of words and time offsets, into the media stream.

Another problem we observed with respect to live media streams is that the streaming frequently aborts. A standard streaming-media player often displays an error message when the stream stops, then the player fails to recover from such an error. To address this problem, we developed a custom streaming-media player around the RealPlayer ActiveX control. The custom player automatically catches and recovers from streaming errors, enabling 24/7 playback without human intervention.

This player thus seeks to maintain the stream in the playing state. We observed that the stream often sticks at one of the following states: stopped, paused, or seeking. When this error occurs, the player notifies our callback functions of the state change. In response to this notification, we reset the URL to be played and issue the play function call again. If the stream locks in either the contacting or buffering states, we wait for a reasonable amount of time—30 seconds—before aborting the current operation, then we reset the URL and reissue the play function call.

The application server consists of two components:

- a standard Web server that supports an HTTP interface, and
- an agent server that supports authenticated peer-to-peer instant messaging based on Vinci, an

architecture for building distributed systems using a lightweight, point-to-point, XML-like remote-procedure-call protocol.³

The Web server uses a Java servlet to analyze and pass on requests to the Live Media Retrieval engine and to format the returned results. The live-media indexing-agent server constantly filters the incoming recognized speech for user-defined keywords, sending instant messages when it finds matching results. The live media retrieval engine indexes the speech recognition text and offers text query functionality. The retrieval engine also maintains an in-memory index of the speech recognition text and continuously updates it using multiple threads by accessing the new speech-recognition text from the speech-recognition stations.

The retrieval engine returns matches to the keyword query together with the live-media channel name, a recording time stamp, the 10 words in the recognized text surrounding the matching word, and, if available, the name of the on-demand audio file and the time when the match occurs within that file. The retrieval engine groups query results by live media channel and tags them with a relevance score, which is built from the number of matching keywords within one channel and the distance between keywords. The system can use the relevance score to sort the query results before presenting them to the user.

Although the system supports live streaming-media indexing, a delay, however small, exists between the time of query and the data being queried. For on-the-air queries, this delay extends up to 20 seconds. For media-on-demand queries, the delay ranges from 5 to 20 minutes.

Our buffering scheme, shown in Figure 3, manages the delay and transition of the system from one state to another. The media and associated indexes flow through our system in one of four states: recording, encoding, query, or delayed broadcast. In the recording state, the system indexes and records the streaming media in real time. In the encoding state, the system encodes the recorded files into a streaming format. In the query state, the system queries the index

associated with the media. The delayed stream refers to the media that will be rebroadcast in the delayed live mode.

The buffering scheme consists of five packets, each five minutes long, in which a packet consists of the decoded text and the corresponding audio. The packets move from right to left as shown in Figure 3: Packet 1 represents the current time and packet 5 represents a 5-minute segment 20 minutes prior to the current time.

The speech-recognition stations transcribe the first packet and capture the corresponding audio. They encode the second packet into streaming format for delayed rebroadcast. The text transcripts corresponding to packets 3 and 4 are available for keyword queries. The corresponding audio can be played on demand. The fifth packet is used for delayed broadcast, which guarantees that query results are upcoming events in the delayed-live stream. Thus, the system issues keyword queries on 10 minutes of data, delayed between 5 and 20 minutes because the query data ages while in use. The system delays the delayed-live broadcast exactly 20 minutes.

Thus, the live streaming-media indexing system supports querying information broadcast in the last 20 seconds, and between 5 and 20 minutes before the current time. Content more than 20 seconds old but less than 5 minutes old is unavailable. Our rationale for this limitation is that a user either seeks a live event that took place less than 20 seconds ago or can wait for the near-live event that took place from 5 to 20 minutes ago.

Application interfaces

We have developed a specific application, CueRadio, that filters information through content-based indexing of live streaming-media channels. CueRadio supports two application interfaces.

Query Interface. Users can query several live-media stations for keywords. Figure 4 shows a screen capture of the query interface with query results for the keyword Clinton. The system found three matching live stations: CNN air, WABC, and CNN Headline News. The matching keywords displayed at the top were broadcast within the last 20 seconds of issuing the query. These results are not yet available for play-on-demand, but the user has the option of listening to the original live stream instead. Next, the interface presents a sorted list of all matching processed results, featuring play-on-demand and delayed-broadcast. All results show a short segment of the corresponding speech-recognition transcript to help users select results.

Querying on live content differs significantly from querying archived static-media content because results for the same keyword can vary each time a user issues a query. In particular, because on-the-air content

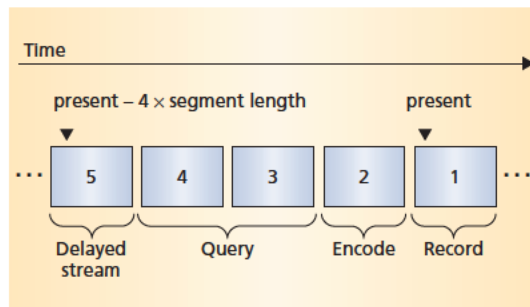


Figure 3. Buffering scheme for managing the delay and transition of the custom streaming-media system from one state to another. Media and associated indexes flow through the system in one of four states: recording, encoding, query, or delayed broadcast.

changes rapidly, a user searching for the latest news on a certain topic would need to constantly repeat the query until it returns relevant results. We use the CueRadio agent to address this temporal behavior of live data.

Agent Interface. The CueRadio agent interface allows instant-message notification when the application finds a relevant keyword in a live-media stream. The agent filters the current live data, and we can configure it to listen for certain keywords. As soon as the agent finds a matching channel, it notifies the user via instant message. The delay between the media station's broadcast of a word and the agent's notification to the user ranges from 5 to 15 seconds.

search for: go ☐ whole word only

Results for your query on: 'CLINTON'

Live on CNN air:

14:30:34 it's was financial services and other company were sponsoring bill CLINTON'S all the obvious the sponsor of they were participants

Live on WABC:

14:30:38 and is no reason why borton lawyers can deposed roger CLINTON at

WABC

14:15:07 go deal do likely with who are with though logic CLINTON had made and had an aide to included who

14:15:43 to both the mother of the prisoners been delayed the CLINTON package and the brother of the day i take

14:20:04 of investigation looking into the issue of influence with roger CLINTON and pardons and perhaps there was some shell but

score: 3.0

CNN Headline News

14:17:36 is a check our top story several elements of the CLINTON card new controversy financier mark richard refusing to appear

14:17:47 them to share details on his pursuit of clemency bill CLINTON'S partner break on his last day in office for

score: 1.77

Figure 4. CueRadio query interface. The interface generated the results in response to a query on the keyword Clinton.

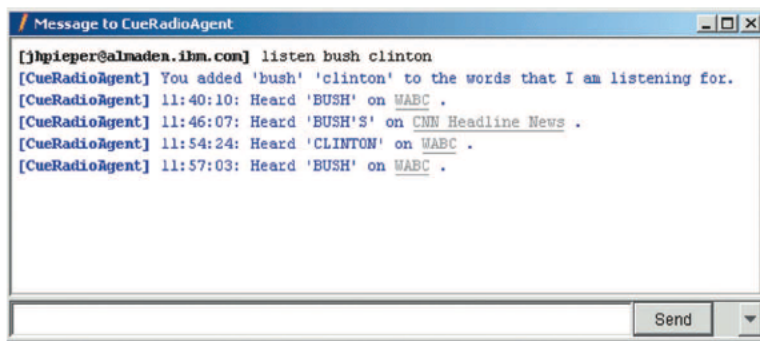


Figure 5. The CueRadio agent interface. The agent server notifies the user when it detects any of the keywords on a live-media channel, supplementing this information with a link to the channel.

Figure 5 shows the custom Java-based CueRadio agent interface. The user initiates the connection and, in this example, requests notification when the keywords Bush and Clinton are spoken on any of the live-media channels. The agent server sends to the application a notification and direct link to listen to the live stream whenever it finds any of the keywords on a live-media channel. In this example, the agent found the keyword Bush on channels WABC and CNN Headline News and the keyword Clinton on WABC.

Our initial study indicates that truly useful media content for applications such as information-extraction service providers and knowledge-management application providers may not be freely available on the Web. Such content will likely be contained within the enterprise or made available through partner sites. However, live-broadcast channels on the Web appear to be potentially valuable as an information source that real-world applications can access.

The ability to search multiple live-broadcast channels lets users locate relevant multimedia resources on the Web. Providing on-demand access to the media reduces the time of consuming irrelevant information from a streaming-media channel. Thus, access to live-broadcast data has been greatly improved.

The extensive work on indexing video archives inspires several directions for future research in this new area. Advanced audio and video indexing techniques⁴⁻⁶ can be adopted that require real-time indexing of live feed. Existing genre-classification and topic-detection methods⁷ require processing and analyzing of large parts of the document, and thus present a greater challenge when performed on live streams. Scalability plays an important role in the deployment of such systems, due to the amount of computation required for the simultaneous processing of multiple live streams. Fortunately, the rapid progress in hardware and networking will enable deployment of such systems in the next several years. ★

References

1. G. Ashour et al., "Who Is SMILing on the Web?" *Proc. 10th Int'l World Wide Web Conf.*, ACM Press, New York, 2001, pp. 90-91.

2. S. Chakrabarti et al., "Mining the Web's Link Structure," *Computer*, Aug. 1999, pp. 60-67.
3. R. Agrawal et al., "Vinci: A Service-Oriented Architecture for Rapid Development of Web Applications," *Proc. 10th Int'l World Wide Web Conf.*, ACM Press, New York, 2001, pp. 355-365.
4. A. Gupta and R. Jain, "Visual Information Retrieval," *Comm. ACM*, May 1997, pp. 70-79.
5. J.G.F. Jones et al., "Retrieving Spoken Documents by Combining Multiple Index Sources," *Proc. 19th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR 96)*, ACM Press, New York, 1996, pp. 30-38.
6. S. Srinivasan and D. Petkovic, "Phonetic Confusion Matrix-Based Spoken Document Retrieval," *Proc. 23rd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR 00)*, ACM Press, New York, 2000, pp. 81-87.
7. D. Poncelon and S. Srinivasan, "Structure and Content-Based Segmentation of Speech Transcripts," to be published in *Proc. 24th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR 01)*, ACM Press, New York, 2001.

Jan Pieper is a software engineer at the IBM Almaden Research Center in San Jose, California. His research interests include Web technologies, multimedia, and handheld devices. He received a diploma engineering degree in computer science from the University of Applied Sciences in Hamburg, Germany. Contact Pieper at jhpieper@almaden.ibm.com.

Savitha Srinivasan is the manager of the Multimedia Knowledge Discovery group at the IBM Almaden Research Center, where her work focuses on applications of speech recognition to multimedia indexing. She has several patents and publications in speech-recognition applications and technology and multimedia information retrieval. She received an MS in computer science from Pace University. Contact Srinivasan at savitha@almaden.ibm.com.

Byron Dom is a research staff member in the Multimedia Knowledge Discovery group at the IBM Almaden Research Center. Prior to that he was the manager of Information Management Principles at IBM Almaden. His research interests are in information retrieval, machine learning, computer vision, and information theory. Dom received a PhD in applied physics from the Catholic University of America. Contact Dom at dom@almaden.ibm.com.