Practical Methods for Automatically Generating Typed Links

Chip Cleary and Ray Bareiss

The Institute for the Learning Sciences
Northwestern University
1890 Maple Avenue
Evanston, IL 60201
Tel: 708-491-1500
E-mail: {chip I bareiss}@ils.nwu.edu

ABSTRACT

Our research concerns how to construct knowledge-rich hypermedia systems for use as aids to problem-solving. One of the most difficult steps in building such systems is constructing a fertile set of hypermedia links between the nodes they contain (i.e., text segments, graphics, and video clips). This paper describes the progress we have made in formalizing and automating the process of creating typed links, that is links that not only join nodes, but also label the relationship between them. We present four different methods we have developed for automated linking, each of which uses a different scheme for representing nodes, and we evaluate each method by the criteria of recall, precision, thoroughness, and ease of use. Two of these methods, designed for two different user populations, are being incorporated into the ASKTool, a hypermedia editor currently in use at the Institute for the Learning Sciences.

KEYWORDS: Automated linking, typed links, structured hypermedia system

INTRODUCTION

Our research is concerned with the practical construction of knowledge-rich hypermedia systems for use as aids to problem solving. In this paper, we describe the progress we have made over the past three years in formalizing and automating the process of building typed links between nodes in one class of hypermedia system. We begin by describing and evaluating an initial method for automating linking which used a complex AI-inspired representational framework. Although this method performed well, it proved prohibitively difficult to use. We then discuss three other linking methods we developed with the design objective of representing "just enough" about nodes to support a pragmatically useful linker. The first of these methods employs an extremely simple representational framework which, upon evaluation, cannot support automated linking. The other two, one of which was designed for expert users and the other for novices, are currently being incorporated into the ASKTool, a hypermedia editor currently in use at the Institute for the Learning Sciences (ILS).

THE ASK APPROACH

Although hypermedia offers great promise as a medium for constructing problem-solving aids, most current hypermedia systems do not provide users with enough consistent structure to prevent them from becoming "lost in hyperspace" [1]. Previous research has described a range of techniques for orienting users. In general, these techniques rely on hierarchy, showing users where their current position falls within a larger "section" of the system. To our knowledge, only a single method has been developed to orient users on a local, associative level (i.e., give their users a clear picture of the structure of the immediate neighborhood of their current position). This method is to organize links by how they relate to the current node using a consistent, task-oriented taxonomy of link types. We call systems that use this approach ***structured hypermedia systems***, since they capture the structure of the user's task in their browsing interface.

ASK systems are a class of structured hypermedia system developed at the ILS.[1] ILS has built over a dozen ASK systems in domains as diverse as business process reengineering, recent American history, and social services for Mexican immigrants. In contrast to most hypermedia systems, ASK systems are based on the metaphor of a conversation with an expert [4]. In particular, they are designed to simulate a question-answer dialog in which the user asks the questions and the system provides the answers. An interaction with an ASK system consists of two phases: first a user zooms to a story[2] relevant to his interests, then he browses the story, asking follow-up questions as desired and retrieving additional stories in response. If the user has additional top-level questions, he may return to the top-level of the system to begin another round of zooming and browsing.

To support browsing, ASK systems contain a rich network of links, each of which joins a source story which raises a question to a target story which answers it. The browsing interface in an ASK system surrounds the current story with

[1] Other examples of structured hypermedia systems include gIBIS [2] and the Collaboratory Notebook [3].

[2] We use "story" colloquially to refer to a single content element in an ASK system (i.e., a piece of text, segment of video, or graphic).

Hypertext '96, Washington DC USA
© 1996 ACM 0-89791-778-2/96/03..$3.50

specific questions it might raise for the user, grouped by question type. The specific questions mark browsing links; clicking on one takes the user to a target story which provides an answer. If a user has a specific question in mind, the question types enable him or her to quickly find it. If the user has only a vague idea of what question to ask, he or she can look under a generic question that looks promising.

Hence, in ASK systems, generic questions serve as link types, though links are also labeled with specific questions. The set of link types used in ASK systems is inspired by a simple theory of conversation which argues that at any point in a conversation, there are only a few general categories of follow-up statements that constitute a natural continuation rather than a topic shift [5]. These "conversational associative categories" (CACs) can also be thought of as the general classes of questions a person is likely to formulate in conversation. The particular link types used in ASK systems are based on a set of CACs tailored for conversations about problem-solving. Figure 1 shows the set most commonly used.

Refocusing: Adjustments to the specificity of topic under consideration.

1. **Context**: What is the big picture within which the current topic fits?
2. **Specifics**: What are the details of this situation or an example of it?

Comparison: Related topics at the same level of abstraction as the current one.

3. **Analogies**: When have other similar situations occurred?
4. **Alternatives**: What other approaches exist, or what did other experts say?

Causality: Explanations and outcomes[3]

5. **Causes (or earlier events):** How did this situation develop?
6. **Results (or later events):** What is the outcome of this situation?.

Advice: Planning knowledge for use in the problem solver's situation.

7. **Opportunities**: How can I capitalize on this situation?
8. **Warnings**: What should I watch for that might go wrong?

Figure 1: The eight CACs most commonly used in ASK systems

THE LIMITS OF MANUAL QUESTION-BASED INDEXING

Structured hypermedia systems are difficult to build. At ILS, we employ trained teams of indexers to build ASK systems who take person-months (and sometimes person-years) to construct them. The two most time-consuming and expertise-intensive steps in building an ASK system are acquiring appropriate stories and generating a rich set of links between them. [6] describes the knowledge-acquisition process; here we concentrate on the problem of building links.

The current manual process, called "question-based linking," has proven reliable. In this process, professional indexers attach to each story a list of the questions it raises and those it answers. They then create links by attaching questions raised to semantically equivalent questions answers [7]. This process (or simple variations on it) has been used to build over a dozen ASK systems. As a sign of its reliability, we are now able to accurately predict how long building a new system will take. However, the method has several drawbacks:

- **Time-consuming**: It requires approximately 5 person-weeks of effort to link each hour of video stories or 100 text stories.
- **Expertise-intensive:** Novice indexers do not pose the same breadth and quality of questions as experienced ones. The need for specialized indexing expertise becomes particularly nettlesome as we move towards the goal of having subject matter experts build ASK systems in the course of doing their jobs (e.g., a team of designers who collaboratively construct a system to capture design rationale).
- **Indexer habituation:** As indexers work with a body of content over time, they stop posing many common novice questions (such as questions about the definitions of terms) in contexts in which they are relevant.
- **Limits on "density" of content:** When a system grows to have many stories in a single topic area, the questions they answer are frequently similar, and question-based indexing becomes cumbersome.

THE GOALS OF AUTOMATED LINKING

Our eventual goal is to provide a *dynamic* ASK system, one in which a user may drop in a new story at any time and have that story become immediately available through contextually appropriate questions everywhere it is relevant. As a step in this direction, our short-term goal has been to provide a partially-automated linker which operates in partnership with a person to build static links between stories. We envision that the person will first construct representations of stories, the linker then suggest links between them, and finally the person will review and edit those links. Figure 2 uses an example link from the

[3] We group temporal order and the causal chain because people typically collapse the distinction.

Engines for Education ASK system to illustrate the specific tasks an automated linker must perform [8].[4]

Three criteria are critical when evaluating the results produced by an automated linker. *Thoroughness* measures how completely the linker supports the three tasks in the linking process. *Recall rate* measures the percentage of the links which an ASK system should have that the linker actually generates. *Precision rate* measures the percentage of links that a linker actually generates which are links that an ASK system should have. An additional criterion is useful to judge the input a linker requires. *Ease of use* judges how hard is it for an indexer to learn and use the representational scheme a linker uses.[5]

Note that the relative importance of these criteria depends on the target audience of indexers. Subject matter experts who build ASK systems as job aids require a linker that is easy to use even if it performs only moderately well on the results criteria. Professional indexers can invest the time to learn and use a more complicated linker if it produces better results.

PREVIOUS METHODS FOR AUTOMATED LINKING

Previous research has provided a variety of methods for automating linking. Some of these methods operate autonomously; others require human input or oversight. None of them, however, prove to be capable of building browsing links for structured hypermedia systems.

Similarity linkers

Several researchers have proposed using global similarity metrics as a basis for interlinking stories [9, 10] . Current similarity linkers, regardless of which specific metric they use, share three characteristics: they require a text-based stories (or a textual synopsis of them); they analyze the text to build a statistical footprint for each story based on its lexical characteristics; and they suggest links by selecting those pairs of footprints judged to be most similar according to their metric. An informal experiment conducted by Bernstein showed that his link apprentice was able to suggest a number of potentially useful links.

Unfortunately, global similarity metrics are not suitable for building browsing links for a structured hypermedia system because they rely on lexical features rather than a semantic analysis of stories. The most critical impact of this shortcoming is on thoroughness. When a global similarity metric suggests that two stories should be paired, it cannot determine which relationship holds between them. For instance, a global similarity metric might determine that the two stories in Figure 2 should be linked since they both mention the lexical term "simulator." However, the metric could not determine that the second story provided specifics about the first. Additionally, global similarity metrics will have limited recall because some stories which share few terms in common should be linked. For example, two different plans for solving a problem should be linked as *alternatives*, though the stories themselves may not share any terms.

Analogy makers

HieNet is an automated linker that makes new links by analogizing them to old ones [11] . It takes a set of manually constructed links as training data and then builds new links by replicating the patterns found in them. The key to the system (and its Achilles heel) is the method it uses to generalize from the training links. When generalizing, HieNet relies on the same surface linguistic features used by global similarity metrics. To create new links, HieNet searches for stories in the system which match the footprints of stored links.

The general approach underlying HieNet is appealing. A linker capable of making links analogous to those found in a training corpus could potentially construct a wide variety of links. However, HieNet as implemented has a significant disadvantage. Because it fails to abstract why stories were linked, it offers little ability to transfer. If an analogy-maker is to achieve transfer, it must either determine the abstract structure which underlies links or be given access to a representation which explicitly labels this structure.[6]

Rule instantiators

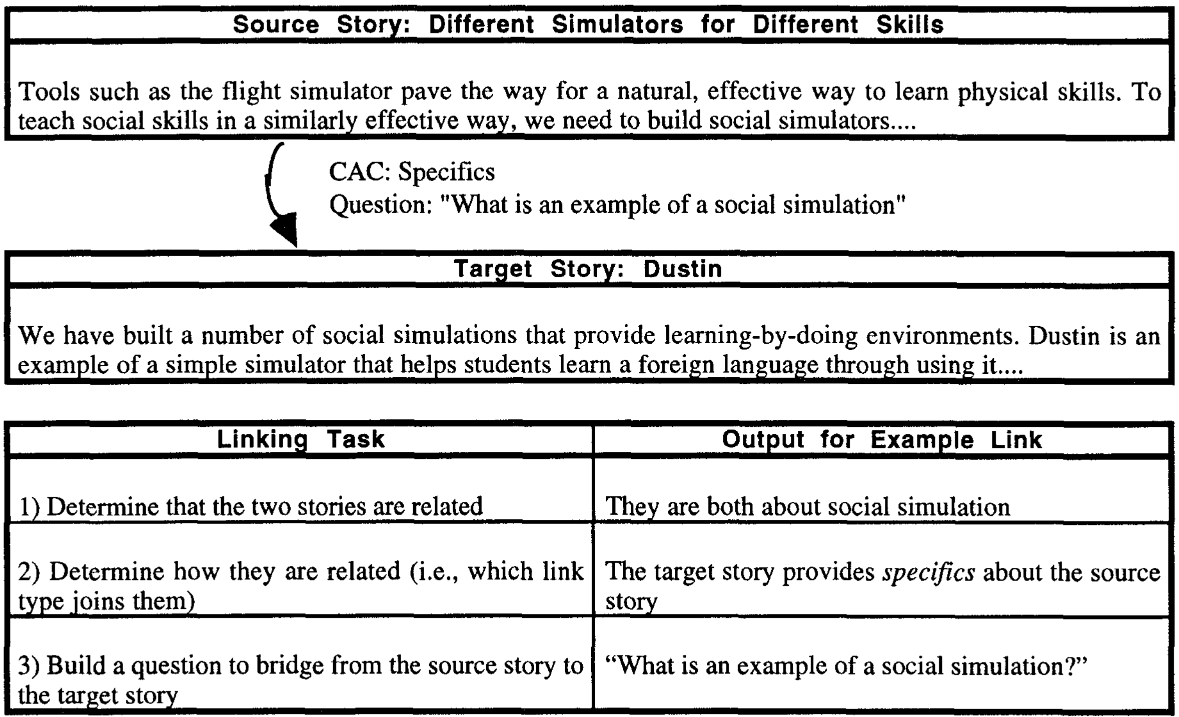
An alternative approach is to have a human develop a library of rules which encode how stories should be interlinked. [13] present a mechanism for implementing this approach. In their system, indexers build representations of interesting sections of content (thereby creating what might be termed "virtual stories") which contain a concept the section elaborates and a qualifier which indicates how. Indexers also define semantic walk expressions in terms of qualified concepts which indicate when one virtual story should be linked to another.

Nanard & Nanard's system can serve as the foundation for a useful automated linker and the four linking methods we

[4] Engines for Education deals with the role that technology can play in reforming education. It contains approximately 350 stories and 4000 associative links.

[5] When judging recall and precision, we use the "gold standard" of systems built manually by human indexers. Although manually-constructed systems are not perfect, they do provide an objective baseline. Furthermore, manually-constructed systems are not perfect in a predictable way -- they typically include few bad links but miss a number of good ones. In other words, when judged against an idealized ASK system, manually-constructed systems have a high precision rate but only moderate recall rate. These distortions have little impact when evaluating the recall rate of an automated linker. However, they provide an overly conservative measure of precision.

[6] As an alternative, an analogy-maker might use derivational analogy [12] to replicate the process which created the link rather than the surface features of the link itself.



Source Story: Different Simulators for Different Skills

Tools such as the flight simulator pave the way for a natural, effective way to learn physical skills. To teach social skills in a similarly effective way, we need to build social simulators....

CAC: Specifics
Question: "What is an example of a social simulation"

Target Story: Dustin

We have built a number of social simulations that provide learning-by-doing environments. Dustin is an example of a simple simulator that helps students learn a foreign language through using it....

Linking Task	Output for Example Link
1) Determine that the two stories are related	They are both about social simulation
2) Determine how they are related (i.e., which link type joins them)	The target story provides *specifics* about the source story
3) Build a question to bridge from the source story to the target story	"What is an example of a social simulation?"

Figure 2: The three tasks a linker must perform

discuss below share several aspects with it. In particular, we require indexers to create hand-coded conceptual representations of stories and two of our methods employ linking rules. Furthermore, one of our methods using a representation quite similar to Nanard & Nanard's qualified concepts. However, our work differs in two important ways from theirs. Their method requires a daunting amount of theory-building from its users. In contrast, we wish to provide indexers with predefined theories of linking (i.e., sets of linking rules) so that they do not need to specify how the linker should go about joining stories. Also, two of our methods use representational frameworks which offer significantly more expressive power than qualified concepts.

NARRATIVE LINKING

Our initial attempt at automating linking, narrative linking, employed a rich representation which treated stories as narratives about planful behavior. This method employed a "narrative frame" to represent stories [7, 14] which encapsulated a naive model of intentionality inspired by the "intentional chain" of [15]. The narrative frame provided a fixed set of domain-independent slots (including, among numerous others, *AgentRole*, *Goal*, *Plan*, *Enablements*, *Impediments*, *Anticipated Actions*, and *Unanticipated Outcomes*). To represent a story, an indexer instantiated the frame by inserting domain-specific fillers into its slots. The primary purpose of the frame was to encourage indexers to make use of a consistent set of features when representing stories. Accordingly, part of the burden indexers faced when constructing narrative frames was to develop a customized taxonomy of fillers for each slot appropriate to the system they were building.

To construct links between stories, the automated linker employed linking rules each of which inferred links for a specialized sense of one CAC. Conceptually, a rule performs a pairwise comparison between the frames representing two stories. For example, a rule might specify that two stories whose agents who have the same *Goal* but employ different *Plans* should be linked as *alternatives*.

We ran an informal experiment to see whether narrative linking would be effective when used by novice indexers. A team of two Northwestern University undergraduates were given a corpus of 47 stories in the domain of military logistics (a subject about which they knew nothing) and the empty narrative frame. They created taxonomies of fillers, instantiated the frame for each story, and defined inference rules for four of the eight CACs. We then ran the rules on the frames, thereby producing approximately 120 story-to-story links. Of these, approximately 80 were judged to be correct by expert indexers. Twenty more were judged to join two stories that were actually related, but not in exactly the way the proposed link indicated (e.g., the direction of causality was wrong). The remaining 20 were judged to be incorrect.

These results indicated that novices could build narrative frames and that an automated linker could use them to construct links with high precision (i.e., 20 incorrect links out of 120 proposed). Further, narrative linking was thorough, supporting each of the three linking tasks: relating stories, assigning conversational categories, and generating questions from a representational template associated with each linking rule.

Because we did not manually construct a normative set of links that a linker should create among the sample stories, the experiment did not provide a measure for recall. However, narrative linking's recall rate is limited because narrative frames cannot conveniently encode information which does not revolve around intentionality. For example, making the link in Figure 1 requires knowing that "Dustin contains a social simulation." Because it cannot capture this information, the narrative linker will be unable to build this link.

Nevertheless, the primary problem with narrative linking was not in the output it produced, but rather in the input it required. The method was not easy to use. The students worked a total of approximately 160 person-hours to represent 40 stories. An experienced indexer could link the stories by hand in roughly half the time it took the students to engineer the representation! Furthermore, the students had difficulty distilling stories down into narrative frames which were consistent with those they developed for similar stories.

Our work with narrative linking taught us that an automated linker can successfully use structured representations, but that they are difficult for novice indexers to create. It is particularly difficult to create representations which are global (i.e., which integrate a story's representation into a single overarching structure) and yet which also contain enough detail to enable an automated linker to relate stories appropriately.

SIMPLE CONCEPT LINKING

After experimenting with the detailed representations of narrative linking, we decided to back off to see how far we could get with a minimal representation of stories and a simple linking algorithm. In particular, we investigated "simple concept linking," in which an indexer represents each story as a set of the important concepts it mentions (cf. [16]). The method uses an extremely simple linking algorithm: it infers a link between two stories when their concept sets share one or more concepts.

As an example, consider the link in Figure 2. To create this link, an indexer would first need to assign concept sets to each story that shared at least one concept (e.g., "social simulation"). The linker would then suggest the stories be paired because of the shared concept. To complete the link, the indexer would need to add the link type, *specifics*, and the question "What is an example of a social simulation?".

Simple concept linking requires the indexer to construct a taxonomy of "important" concepts, which is unstructured but is otherwise roughly analogous to the taxonomies of fillers used in narrative linking. In theory, "important" concepts are those which underlie the relations between stories which should be linked. In practice, it is easier for indexers to think of them as ones which capture key distinctions in a system's content. In our experience, an indexer can produce a workable taxonomy in a single pass over a set of stories.

Simple concept linking has several attractive features. Because it uses a minimal representation, it is easy to use. It does require that indexers build a taxonomy of concepts, but to the extent that new hypermedia systems deal with idiosyncratic topics, any linking method must at least require indexers to define a customized vocabulary. To consider some examples, it would be unrealistic to expect a predefined universal vocabulary to include concepts such as *Dustin* (required for Engines for Education), *Roll-On, Roll-Off Ship* (required for TransASK), and *privatization* (required for ASK NorthWest Water).

Simple concept linking also has a high recall rate. To measure this rate, we encoded each story in Engines using a hierarchy of 180 concepts.[7] We then compared the pairs of stories which indexers linked manually to those which simple concept linking suggested. The automated method duplicated the pairings of 70% of the manual links.

Since we suspected that simple concept linking might be better able to create links for some link types (such as *specifics*) and than others (such as *alternatives*), we broke down the analysis by link type. We found that the method performed consistently across the link types.[8] Its lowest recall rate was on *warnings* (67%) and its best was on *specifics* (78%). We believe *analogies* links to be the most difficult to create of the link types listed in Figure 1, both for human indexers and automatic linkers. Good *analogies* links are often abstract, joining stories which do not share low-level features. Unfortunately, the Engines system did not use the *analogies* link type, so the experiment did not provide a recall rate for it.

Unfortunately, the method's poor performance on the remaining two evaluation criteria make it an unacceptable choice for automated linking. Its most important shortcoming is that it wildly overgenerates potential links. Only 12% of the links it suggested were duplicated by manual links. Although this is a conservative measure of precision (as discussed above) and although it is well above chance (manual links represented 3% of possible links), this rate is unacceptably low. Furthermore, simple concept linking is not thorough. After it determines that two stories are related, it relies on the indexer to manually indicate how they are related and build a bridging question between them. This shortcoming is particularly acute given that concept linking produces so many poor links.

Armed with what we have learned from our experiments with concept and narrative linking, our current strategy is to develop two different linking methods for two different populations of human indexers. For novice indexers (e.g., subject matter experts who are constructing an ASK system as an adjunct to their primary job), we have created

[7] We defined a new concept whenever the central point in a story dealt with a concept that had not been previously encountered. For example, the story "Different Simulations for Different Skills" (Figure 1) states that computer scientists know how to build good simulations of the physical world but not of the social world. This story led us to create the concepts *social simulation* and *physical simulation* as specializations of the previously defined concept *simulation*.

elaborated concept linking. This method is based on concept linking (henceforth called "simple concept linking") but adds a small amount of additional representation to address its shortcomings. For professional indexers, we have created *point linking*. This method is analogous to narrative linking, but introduces a new representational scheme which we expect will be easier to learn and use.

ELABORATED CONCEPT LINKING

Simple concept linking has three shortcomings: it wildly overgenerates links, lacks thoroughness, and fails to give users guidance about how to represent stories. Elaborated concept linking is designed to rectify these limitations.

Elaborated concept linking adds a straightforward extension to simple concept linking: it requires indexers to divide the concepts they associate with stories into two groups, concepts mentioned and concepts elaborated. The indexer must label each concept in a story's concept set as either a concept mentioned or a concept elaborated (i.e., one the story says something significant about). Furthermore, elaborated concept linking requires the indexer to specify *what* the story says about each concept elaborated by attaching a link type and question answered to it.

For example, when representing the top story in Figure 2 ("Different Simulators for Different Skills"), the indexer might add *social simulation* as a concept elaborated, attaching the link type *context* to it and the question answered "Why are social simulations valuable?" Likewise, when representing the bottom story ("Dustin"), the indexer might also add *social simulation* as a concept elaborated, but attach the link type *specifics* to it and the question answered "What is an example of a social simulation?"

Elaborated concept linking proposes links from source stories which mention some concept only to those target stories which elaborate on it (or from one story which elaborates a concept to another that also elaborates it). Hence, if the stories in Figure 2 were represented as described above, the linker would suggest that they be paired by two links: a *specifics* link from the top story to the bottom one with the question "What is an example of a social simulation?" and a *context* link from the bottom story to the top one with the question "Why are social simulations valuable?"

Improving Precision

A primary reason why simple concept linking proposes too many links is that it does not differentiate between concepts a story merely mentions and those it elaborates (i.e., says something significant about). Hence, when two stories both merely mention some concept, even if only tangentially, simple concept linking mistakenly pairs them. To avoid this error, elaborated concept linking will create links to a story only if it elaborates on the concept which underlies the link.

This extension dramatically reduces the number of suggested links. Without it, simple concept linking proposed over 20,000 links in Engines. With it, elaborated concept linking proposed less than half as many. Furthermore, the extension predicted reasonably reliably which links to eliminate. An algorithm which eliminated links at random would have eliminated good links as well as bad in the same proportions that they occurred in the set of links proposed by simple concept linking (specifically, 12%). Since only 6% of the suggested links that were eliminated duplicated manual links, the extension proved to be a useful filter.

Nevertheless, this extension only raises elaborated concept linking's precision to 16%. Although this is a large jump relative to simple concept linking's 12%, it still leaves significant room for improvement. Accordingly, we are investigating a series of additional extensions which further raise precision but do not compromise the ease of use offered by concept linking methods. The next extension we are planning adds linking rules which consider not just *whether* stories elaborate some concept, but also *how* they do. It will enable the linker to determine, for example, that it makes sense to link a story which describes a problem to one which describes its solution, but not to link a story which only gives a detail about a problem to one which describes an analogy to it. If this extension does not raise precision sufficiently, we are also planning a third which views the task of building a set of links as a constraint satisfaction problem. Instead of deciding whether to include each potential link in isolation, this extension will collect the best overall set of links for each story, trading off the strength of each potential link against others attached to the same story through the same link type.

Improving Thoroughness

The second shortcoming of simple concept linking is that it requires the indexer to attach a link type and a question answered to each of the links it suggests. Elaborated concept linking employs a simple expedient to address this shortcoming. As the example above illustrates, it asks indexers to attach link types and questions not directly to links, but instead to elaborated concepts. When it creates a link to some particular concept, it then retrieves the link type and question answered from the concept and inserts them into the link. This method, which does not involve any inference, saves indexing effort because many links are typically built to each concept elaborated.[8]

This expedient assumes that the link type and question answered assigned to an elaborated concept will be appropriate to use in suggested links regardless of the relationship between the source and target stories. Although generally true, this assumption is not flawless. In particular, it mislabels *alternatives* links, since such links typically join stories which elaborate the same concept in the same way. For instance, the same corpus of stories that includes the Dustin story of Figure 2 also includes a story about Yello, another computer program which contains a

[8] This technique could not be used in simple concept linking because simple concept linking does not include the notion of an "elaborated concept."

social simulation. An indexer who attached the concept elaborated *social simulation* to the Dustin story, marking it with the *specifics* question "What is an example of a social simulation?" would likely do the same to the Yello story. Hence, the linker would pair these stories with dual *specifics* links. However, they should be linked by *alternatives* links (e.g., "What is another example of a social simulation?") Practically speaking, we believe that this limitation is manageable since the *alternatives* case appears to account for the bulk of mislabeled links and this case may be caught automatically.

Providing Representational Guidance

Simple concept linking's final shortcoming is that it provides no guidance about which concepts indexers should include in the concept hierarchy. To provide guidance, we have developed a hierarchy of common concept types (shown in Table 1) which indexers may use as general categories under which they may install specialized domain-specific topics. The goal of this hierarchy is not to include every possible type of concept, but rather to provide a skeletal framework which contains just the types of concepts that are most important in linking. We have found two types of concepts to be particularly useful for linking: goal/plans and agents (which is not surprising since the link types in ASK systems are targeted to problem-solving). Hence, those sections of the predefined hierarchy are richer than others.[9]

In elaborated concept linking, indexers must not only attach concepts to stories, they must attach link types and questions to elaborated concepts. To enable the linker to also provide guidance about appropriate link types and questions, we are developing a taxonomy of common questions, categorized by link type, for each predefined concept.[10] When an indexer enters an elaborated concept into the linker, this taxonomy will enable the linker to offer suggestions of already-typed questions in response.

Improving Recall

Even though the mentioned/elaborated distinction proved a useful filter for eliminating poor links, it also eliminated a small percentage of good links. Because the distinction eliminated so many links, even this small percentage caused a significant reduction in recall. Whereas simple concept linking reproduced 70% of the manually-constructed links, elaborated concept linking duplicated only 53%.

This reduction in recall is significant enough to represent a new shortcoming which we desired to address. To investigate how to improve recall, we analyzed a sample of 100 manually-created links which elaborated concept linking failed to duplicate to determine the cause of the failure.

- Attribute
  - Mental Attribute
- Domain
- Event
  - Mental Event
    - Goal/Plan
      - Acquiring Something
      - Building Something
      - Changing Something
      - Managing Something
      - Measuring Something
      - Preventing Something
      - Providing Something
      - Pursuing Something
- Parable
- Object
  - Conceptual Object
    - Policy
    - Role
    - Theory
    - Tool
  - Value Judgment
  - Physical Object
    - Agent
      - Organization
        - Formal Organization
        - Informal Organization
      - Person
- State

Table 1: The predefined concept hierarchy

This analysis isolated three reasons why elaborated concept linking failed. However, the first two shed little light on how to improve it. Thirty of the missed links were caused by incomplete concept sets (i.e., concept sets which failed to include a concept needed to make the link). Although these links would have been made given perfect concept sets, it is unrealistic to expect humans to produce flawless representations, even if they are given automated assistance.

For another thirteen of the supposedly "missed" links, elaborated concept linking actually behaved correctly. Upon review, the manually-constructed links which the method "missed" were themselves found to be flawed; they joined source stories to targets that were not relevant to them.[11]

However, the remaining (and most common) cause of missed links does suggest a way to improve recall -- adding more powerful forms of inference to the linking algorithm. Elaborated concept linking currently performs only a single form of inference, intersecting concept sets to infer that two

[9] This hierarchy is not specific to elaborated concept linking; it could be used in the same way with simple concept linking.

[10] We are using previously constructed ASK systems as our source of data.

[11] Indexers occasionally create flawed links when they strive too hard to increase the number of links attached to a source story which has few directly relevant target stories.

Type of Inference	Missed Links	Example
Taxonomic	5%	A story which discusses *Dustin* might be related to a story which discusses *educational software* because *Dustin* is a piece of *educational software.*
Encapsulated topics	5%	A story which discusses *educational software* might be related to a story which discusses *education* since the topic *educational software* encapsulates the topic *education.*
Alternatives	4%	A story which discusses *learning-by-doing* might be related to a story which discusses *memorization* since they are alternative ways of *learning.*
Planning/ causality	3%	A story which discusses *educational software* might be related to a story which discusses *learning* because *educational software* can cause *learning.*
Bridging	10%	A story which discusses *educational software* might bridge to a story which discusses *teaching.*

Table 2: Types of inference required to pair stories

stories are related. When pairing stories requires more complex inference, as it did in 57 of the missed links, we found that five additional types of inference, displayed in Table 2, were sufficient to account for the missed links.

The last category, bridging inference, merits further discussion. Most browsing links in ASK systems are based on questions users are likely to raise when they view a source story (such as "Why are social simulations valuable?" when reading the Dustin story). In contrast, bridging links introduce content that users might not have known to look for, but which might draw their interest when presented to them (such as "Why is it important to let students learn by doing?"). Users who wish to broadly sample a system's content may use bridging links to navigate between "clusters" of tightly related stories. However, bridging links are difficult to construct automatically because the relationship between the stories they join is often somewhat tenuous and therefore difficult to infer

POINT LINKING

The Motivation for Point Linking

It is instructive to compare elaborated concept linking with our initial method, narrative linking. Elaborated concept linking is easy to use, has a respectable recall rate, and is not limited to stories which revolve around intentionality. However, narrative linking offers significantly higher precision and thoroughness (since it can accurately assign link types and questions to links).

Point linking is our attempt to develop such a method. It addresses three shortcomings of narrative linking:

- Representational approach: Narrative linking attempted to capture the global structure of a story. Point linking uses local "snippets" of representation which attempt to capture just those aspects of a story which would cause an indexer to create a link involving it.
- Representational guidance: Narrative linking requires the indexer to build a custom taxonomy from scratch for each slot in its representational frame. Point linking provides predefined taxonomies for its slots. Only one of these taxonomies needs to be extended by the indexer.
- Domain coverage: Point linking can be used with any type of content, not just intentional chains.

One source of motivation for point linking comes from observing how experienced indexers perform manual question-based linking. When indexers argue about why two stories should be linked in a particular way, they often frame the argument in terms of terse statements about the contents of those stories. An indexer might say, for instance, "We should add an *alternatives* link because the first story states that 'necessity leads to innovation' and the second states that 'hard work leads to innovation.'" Statements like "necessity leads to invention" are exactly what we aim to capture in a representation of points. Why not ground a language used to support linking in the sorts of explanations that expert indexers use?

Another source of motivation for point linking comes from observing how good readers behave. When good readers read a text, they boil it down, abstracting its main points.[12] These points help readers to determine what questions they should ask themselves about a text and to integrate what they read with what they already know. If people use summarized points internally to attach new knowledge to pre-existing knowledge, why not use them in an external automated system which does the same?

The Architecture of a Point

We define a "point" of a story as a central piece of information someone might try to convey by telling the

[12] Others have stressed the importance of teaching novice readers to build summaries when reading (e.g., [17]).

story.[13] To illustrate, some example points from Engines for Education are "Simulations enable students to learn-by-doing," "The Creanimate program is based on the theory of case-based teaching", and "Multiple choice tests are bad." To represent points, we have developed a simple, fixed frame which contains five slots (Table 3). The goal of this frame is not to capture every nuance of the points authors make with stories. Rather it is to capture a limited amount of information (i.e., an amount which is not onerous for indexers to represent) which captures the broad sense of the author's intentions for use by an automated linker.

Slot Name	Legal Fillers	Example
Concept-1	(Any concept)	Simulations
Mode	Do, Should, Can	Do
Sense	Indeed, Not, Anti	Indeed[14]
Relation	(Predefined relation)	Enable
Concept-2	(Any concept)	Learning-By-Doing

Table 3. The point frame

The two concept slots indicate what topics an author addresses in a point. As in the other linking methods, indexers are expected to expand the set of legal concepts to reflect the universe of discourse in their particular ASK system. To provide guidance, the representation language for points provides a predefined concept hierarchy (identical to that provided for elaborated topic linking).

The mode, sense, and relation fields indicate what an author says about the topics in a point (i.e., how the concepts in a point relate to each other). We claim that there are only a small number of important ways in which speakers (and authors) relate concepts to each other. So, the language provides comprehensive, fixed vocabularies for the three fields which deal with how concepts interrelate. The example point demonstrates the strength of this approach. It seems reasonable that a language should contain predefined terms such as *Do*, *Indeed*, and *Enable*, but unreasonable to expect it to include *Simulations* and *Learning-By-Doing*.

The mode and sense fields allow an indexer to twist the meaning of a point, while the relation field provides the pivot around which the meaning may be twisted. The point language provides a predefined hierarchy of relations (Table 4). Unlike the concept hierarchy, this one is not expected to be routinely extended by indexers.[15] The relations it contains are intended to capture (sometimes at a high level) the important conceptual relationships which hold between concepts in the points people make. We have encoded the stories in Engines using the point language and have found the current hierarchy sufficient to capture the relationships required for the approximately 750 points which resulted.

- Have-Causal-Relation
  - Enable
  - Explain
  - Have-Result
  - Implement
  - Motivate
  - Explain
- Have-Class-Relation
  - Have-Classification-Element
  - Have-Example
  - Have-Subclass
- Have-Detail
  - Contain
  - Have-Attribute
  - Have-Context
- Have-Economic-Relation
  - Create
  - Possess
    - Know
- Have-Interpretation-Relation
  - Compare-With
    - Is-Better-Than
  - Have-Value-Judgment
- Have-Planning-Relation
  - Have-Goal
    - Have-Function
  - Have-Opportunity
  - Have-Warning
    - Have-Limitation
    - Have-Resolved-Warning
  - Use-Plan
    - Apply-Theory
    - Contain-Step
    - Have-Script
    - Use-As-Policy
  - Use-Role
  - Use-Thing
    - Borrow
    - Consume
- Have-Storytelling-Relation
  - Have-Coda
  - Have-Description
    - Have-Definition
  - Have-Illustration
  - Have-Teaser
- Have-Time-Relation
  - Have-History
  - Have-Prognosis
  - Precede

Table 4: The predefined relation hierarchy

Using Points to Build Browsing Links

A browsing link joins a story which raises a question to another which answers it. Point linking applies this progression not at the level of entire stories, but rather at the level of individual points -- it joins a point which raises a question to another which answers it. Since points are attached to stories, when point linking pairs two points, it also pairs the stories which make them.

To determine which points raise and answer questions, the method employs a library of "point association questions" (PAQs). Like the linking rules used in narrative linking, PAQs operate on structured representations of stories to

[13] Others have defined "point" differently. For example, [18]. proposed that the point of a statement was the impact that the speaker intended the statement to have on the listener.

[14] The sense slot allows an indexer to negate or invert a point. The filler *Indeed* leaves the meaning of a point unchanged. *Not* negates a point, meaning that the point is known to not hold true. *Anti* inverts a point, meaning that the opposite of a point holds true. For example, the point *[Simulations Do Anti Enable Learning-By-Doing]* should be interpreted as meaning that simulations *prevent* learning-by-doing.

[15] At least, it is not expected to be routinely expanded by indexers of ASK systems. It may be necessary to add new classes of relations to index material for other types of task-structured hypermedia systems.

Join the source point:	<Concept-1 Does Indeed Contain <Concept-2>]
To the target point:	<Concept-3> Does Indeed Contain <Concept-2>
Through the question:	"What else also contains a <Concept-2>?"
Using the link type:	Alternatives

Figure 3: An example point association question

infer links which correspond to a specialized sense of one of the CACs. More specifically each PAQ captures one type of question used in ASK system browsing links. Figure 3 shows an example.

To create links, the linker considers each point in a corpus of material as a potential source point, creating questions the point raises and then locating answers to them. For example, the example PAQ enables the program to create the link shown in Figure 2. One of the points the source story in this link makes is *Dustin Does Indeed Contain Social Simulation*. Since this point matches the PAQ's pattern for source points , the program instantiates the pattern for questions, thereby raising the *alternatives* question "What else also contains a social simulation?" To find target points, the program also instantiates the pattern for target points, creating the more specific pattern *<Concept-3> Does Indeed Contain Social Simulation.* The program then searches for target points which match the pattern, locating the point *George Does Indeed Contain Social Simulation*, which is one of the points made by the "George" story. Since this point answers the question, the linker draws a link between the "Dustin" and "George" stories, labeling it with he *alternatives* link type and the appropriate question.

We do not yet have empirical results from point linking. However, we expect that it will achieve a recall rate which approaches that of elaborated concept linking, though it will be slightly lower. The reason for the difference is that point linking as described here lacks one type of information available to elaborated concept linking: the set of concepts which a story mentions but does not elaborate (generally, only those concepts which are elaborated will be included in points). As a practical matter, this means that point linking will likely be less likely to infer context links which give general descriptions of peripheral concepts.[16]

We also expect that point linking will have a precision rate which approaches that of narrative linking because both methods use structured descriptions and explicit inference rules to privilege the most important links between stories. Elaborated concept linking has low precision because its representation provides scant information about the roles that concepts play in the stories that discuss them (and the method uses even less than is provided, since it does not currently consider *how* concepts are elaborated when creating links, only *that* they are). Narrative linking has high precision because its structured representation makes clear the roles concepts play in stories. We found, however, that most narrative linking rules tapped into only a few slots of the narrative frame. Hence, simpler structures should be able to achieve comparable results. Points are such simpler structures, designed to capture just those local fragments of representation which are important to make good links (including fragments that narrative frames could not capture because of their reliance on the intentional chain).

SUMMARY

We have approached the task of automating linking from a decidedly practical bent. Our approach has been require indexers to create just enough representation of stories to enable a linker to accomplish its task. In the process, we have learned the following lessons:

- Indexers have difficulty constructing global, structured representations. Hence we have devised linking methods which require simpler "snippets" of representation. We have also added predefined taxonomies which provide indexers with representational guidance.
- Different audiences of indexers require different linking methods. Specifically, novice indexers require methods with less complex representational demands than expert indexers.
- The central challenge a linking method faces is to filter out poor links without eliminating good ones. Hence, the primary reason to add knowledge to a linking system is to address this challenge.
- A straightforward linking method, simple concept linking, requires a minimum of representation and can suggest a large proportion of the hypermedia links that should be created. The critical shortcoming of this method is that it also suggests many poor links.
- Elaborated concept linking, a method which adds a minimum of representation to redress the shortcomings of concept linking, appears to suit the needs of novice indexers who require an easy-to-learn and easy-to-use method, but at the cost of poor precision.
- Point linking should suit the needs of professional indexers who can invest the effort to learn a more complex representation to achieve more thorough and accurate linking.

[16] Although point linking may fail to create links to such peripheral concepts, elaborated concept linking creates *too many*. Specifically, elaborated concept linking creates links to stories that elaborate them in inappropriately detailed ways.

ACKNOWLEDGMENTS

The distinction between concepts mentioned and concepts elaborated is due to Chris Wisdo. Thanks also to Chris for his helpful comments on this paper. We thank Carolyn Majkowski for assisting with the analysis of narrative linking as well for her continuing work in developing a taxonomy of generic questions for common types of topics. This work was supported by in part by ARPA (grant number # N00014-93-1-1212). The Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting. The Institute receives additional support from Ameritech and North West Water, Institute Partners, and from IBM.

REFERENCES

1. E. Conklin, *IEEE Computer* **2**, 17-41 (1987).

2. J. Conklin, M. L. Begeman, gIBIS: A hypertext tool for exploratory policy discussion, The ACM Conference on Computer-Supported Cooperative Work (CSCW'88) 1988), pp. 140-152.

3. D. C. Edelson, D. K. O'Neill, The CoVis Collaboratory Notebook: Supporting collaborative scientific inquiry, Proceedings of The 1994 National Educational Computing Conference 1994),

4. R. Bareiss, R. Osgood, Applying AI models to the design of exploratory hypermedia systems, The Proceedings of the Fifth ACM Conference on Hypertext (ACM Press, Seattle, WA, 1993), pp. 94-105.

5. R. C. Schank, *Cognitive Science* **I**, 421-441 (1977).

6. C. Cleary, R. Bareiss, A descriptive model of question asking during story acquisition interviews, Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society Atlanta, GA, 1994),

7. R. Osgood, Ph.D., Northwestern University (1994).

8. R. C. Schank, C. Cleary, . (Lawrence Erlbaum Associates, Hillsdale, NJ, 1995),

9. M. Bernstein, An apprentice that discovers hypertext links, Proceedings of the European Conference on Hypertext Versailles, France, 1990), pp. 212-223.

10. G. Salton, J. Allan, Selective text utilization and text traversal, The Proceedings of the Fifth ACM Conference on Hypertext (ACM Press, Seattle, WA, 1993), pp. 131-144.

11. D. T. Chang, HieNet: A user-centered approach for automatic link generation, The Proceedings of the Fifth ACM Conference on Hypertext (ACM Press, Seattle, WA, 1993), pp. 145-158.

12. J. G. Carbonell, in *Machine learning: An artificial intelligence approach*(Morgan Kaufmann, Los Altos, CA, 1986) pp. 371-392.

13. J. Nanard, M. Nanard, Should anchors be typed too? An experiment with MacWeb, The Proceedings of the Fifth ACM Conference on Hypertext (ACM Press, Seattle, WA, 1993), pp. 51-62.

14. R. Osgood, R. Bareiss, Automating index generation for constructing large-scale conversational hypermedia systems, Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI Press, Menlo Park, 1993), pp. 309-314.

15. R. C. Schank, R. P. Abelson, *Scripts, Plans, Goals, and Understanding* (Lawrence Erlbaum Associates, Hillsdale, NJ, 1977).

16. R. Spiro, J. Jehng, in *Cognition, Education, and Multimedia: Exploring Ideas in High Technology* D. Nix, R. Spiro, Eds. (Lawrence Erlbaum, Hillsdale, NJ, 1990).

17. A. Collins, J. S. Brown, S. E. Newman, in *Knowing, learning, and instruction: Essays in honor of Robert Glaser* L. B. Resnick, Eds. (Lawrence Erlbaum Associates, Hillsdale, NJ, 1983) pp. 453-494.

18. R. C. Schank, et al., *Cognitive Science* **6**, 255-275 (1982).