Finding and Linking Incidents in News

Ao Feng and James Allan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003, USA
{aofeng, allan}@cs.umass.edu

ABSTRACT

News reports are being produced and disseminated in overwhelming volume, making it difficult to keep up with the newest information. Most previous research in automatic news organization treated news topics as a flat list, ignoring the intrinsic connection among individual reports. We argue that more contextual information within and across the topics will benefit users in their news understanding process.

A news organization infrastructure, *incident threading*, is proposed in this article. All text snippets describing the occurrence of a real-world happening are combined into a news incident, and a network is composed of incidents that are interconnected by links in certain types. A limited vocabulary of connection types is defined and corresponding rules are established based upon the human experience of news understanding.

The incident threading system is implemented with two different algorithms. One starts from clustering of text passages and then creates links with pre-built rules. The other method defines a global score function over the whole collection and solves the optimization problem with simulated annealing. The former achieves higher accuracy in the identification of incidents and the latter generates better links, which is preferred since the links are more important for the formation of the incident network.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Clustering; H.3.4 [**Systems and Software**]: Information networks

General Terms

Algorithms, Design, Experimentation, Management, Measurement, Performance

Keywords

Automatic news organization, Incident threading, Threading rules, Global optimization, Simulated annealing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CIKM'07, November 6–8, 2007, Lisboa, Portugal.
Copyright 2007 ACM 978-1-59593-803-9/07/0011 ...$5.00.

1. INTRODUCTION

With the rapid development of modern technologies, the amount of information is increasing in an exponential manner [6]. Every day there is a huge amount of new information available to us, and a large portion of it is news. It comes from many different sources, including traditional media such as newspaper, radio and TV, and modern sources like the Web. Without proper organization of the overwhelming information, one can easily become lost because of its vast size. This phenomenon is called *information overload*.

It is not feasible for a user to go through all the information without any pre-processing, because the news a person can read is much less than the amount produced within the same time period. To help the user obtain the necessary information in the shortest time, a system should be designed that automatically processes news and converts it into a more user-efficient format.

The way news is organized in the system should match people's cognition process.

- Each user has his or her own information need. For example, a resident of New York City might be interested in a crime that happened in the City, but may not care if there is a conflict in Kosovo. So the system must be able to separate news from different topics.

- People remember what they already know, and are only interested in new messages rather than repetitions, even if they are described is a different way. That means the system should not provide duplicate information.

- People do not treat news events as isolated facts. Instead, they tend to compare new information to memory and insert it into the existing fact network, at a location next to the relevant pieces. It would be preferable if the system is able to link related events, because people are very likely to be interested in both (or neither).

Here we are proposing a model called *incident threading*. It identifies all text pieces that discuss the same real-world happening and assigns them to a news incident. Incidents are not treated as isolated entities, because there are certain types of links between the real-world events, and people have the ability to recognize them. In this model, causal, temporal, spatial or other kinds of connections are considered among the incidents, and their relations are explicitly marked with the specific type. With these links as edges, the

news forms an interconnected network of incidents showing how they are related.

The most important concepts in the model are *incident* and *incident network*. An incident is a news event that happens at a specific time (or in a time range), at a given geographical location, and involves one or more entities and some action. An incident can appear in various news reports, and even if the vocabularies do not look similar, it is always the same incident if they talk about the same thing. News is not static, as there are often new updates on a certain topic. The updates do not belong to the old incident, since the time and/or other factors have changed, but there is some internal connection that places them together. An incident network is a graph that shows these related incidents along with the links.

In Section 2, we will describe earlier research topics related to incident threading like TDT, event threading and discourse analysis. Section 3 shows the incident threading infrastructure and explains the main concepts. Section 4 introduces different implementations of the incident threading system, and the evaluation methods are in Section 5. Section 6 describes the details of the experiments. Section 7 shows that both the two-stage algorithm and simulated annealing achieve obvious improvement over the baseline, while the global optimization algorithm is preferred because of its performance in incident links. Section 8 concludes the article together with suggested future work.

2. PREVIOUS WORK

The idea of incident threading was motivated by existing research topics. *Topic Detection and Tracking* (TDT) monitors a news stream and places the information pieces into individual topics, while each topic includes all the news events closely related. It is ignored in TDT how a topic is established by the news events, and *event threading* tries to capture the internal structure of these topics. In addition to the effort of automatic news organization, *discourse analysis* studies the information flow in a press article with manual analysis. There are other news processing tasks, like novelty detection, news summarization and information filtering, which also aim at helping users in their news browsing. We now summarize these works and show why they have not provided an ideal framework for news organization.

2.1 TDT

TDT is a research program that focuses on event-based news organization. It breaks an incoming news stream into a list of topics, where each topic is "a set of news stories that are strongly related by some seminal real-world event" [1]. To simplify the problem, several assumptions were made in TDT:

- Topics do not overlap. That means each news story belongs to at most one topic. From our observation, it is not always true since parts of the same story are often about different events, and sometimes the boundary between similar topics is not very clear.
- Topics are independent. A topic is a complete object and any relation to others is ignored, which is obviously not the case in reality.
- Topics are indivisible. All the evaluation metrics are topic-based, and participants of TDT place all the efforts into making their topics closer to the truth (defined by examples). However, an important factor is ignored in this assumption. A topic is composed of a seminal event and other "related" events, but the TDT community did not go deeper into the topic structure.

TDT has become quite mature after eight annual evaluations (1997 to 2004). The concept "topic" has been well defined empirically, and building topics from a continuous news stream has achieved moderate accuracy. However, the way that TDT defines a topic seems inconsistent with user expectations, particularly by the assumptions listed above. Within incident threading, we remove or substantially reduce each of those assumptions.

2.2 Event threading

As an attempt to organize news in finer-grained units, *event threading* [5] tries to capture the news events within a TDT topic and the organization among those events. An event is defined as "something that happens at some specific time and place" [13], and events in the same topic are shown in a directed acyclic graph (DAG). An edge from event A to event B means that there is some dependency between them, either causal (A directly leads to the happening of B) or temporal (A precedes B in time). However, the causal and temporal relations are hard to distinguish, and a clear boundary is not set between them. Therefore, only the existence of dependency between two events is modeled, and no attempt is made to identify the relation type.

From the experiments in [5], acceptable accuracy can be achieved in event threading with simple algorithms and easy-to-extract features, but there are still many limitations in this model.

- Each topic is independent. In this work, topics are assumed to be independent to prevent linked events from spanning topic boundaries, thereby reducing the computational complexity of the task.
- Dependencies between events are binary. There can be many different types of dependencies, and the assignment of such labels is subject to the annotator. In this preliminary attempt, the relation was simplified to one of two values: related or not.
- A story is the smallest news unit. It assumes that all the text in a news story is coherent in content and discusses the same thing.

In order to make it clear how news is organized, we need a more complicated model that takes these into consideration. Incident threading bears many similarities to event threading, but we remove the topic independence assumption and incorporate multiple types of links (dependencies) between incidents.

2.3 Discourse analysis

One of the deficiencies of event threading is that it does not have the actual relation type, but how is the dependency decided between two events? If we randomly pick two events from a news topic and ask an annotator to give the description of their relation, we may end up with tens of different answers after trying 100 people. A limited vocabulary of relation types and a detailed description (if a formal definition cannot be provided) of each are necessary to avoid the possible confusion.

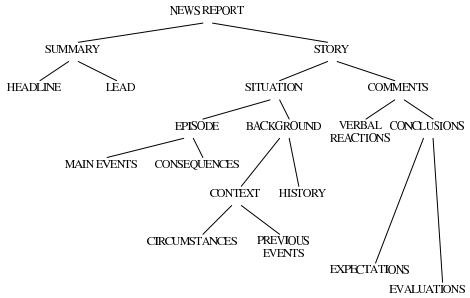

Figure 1: Hypothetical structure of a news schema

There has not been any previous attempt to do this in the information retrieval community, but discourse analysis in journalism deals with similar problems. Figure 1 (reproduced from a figure by Dijk [12]) shows the structure of a news schema in discourse analysis. Some of the units are valueless for us (like headline and lead), while others can be good candidates to describe the organization of news events.

It is worth pointing out that discourse analysis methods are usually applied by people instead of computer programs. We use the derived structure to select reasonable link types.

2.4 Others

Besides TDT and event threading, there are other research topics that also aim at automatic processing of news to help retrieve useful information.

TREC has a novelty track that ran for three years - 2002 through 2004 [10] - to deal with the redundancy problem. The conclusion from the track is that novelty detection is still a hard problem, mainly because of the limited information contained in each sentence.

News summarization is another way of reducing the reading workload of users. Newsblaster [4] clusters news stories into a hierarchical structure, where the units in the lowest level are similar to news events, and larger clusters correspond to topics. NewsInEssence [7] is another summarization system, and it supports user-formed clusters. News summarization systems reduce information length, but it is still the user's responsibility to keep track how the topic evolves over time.

News filtering systems [3] serve only information predicted to be interesting to users, and the key problem is to create and update the *user profile*. News filtering is based on the assumption of a relatively stable profile of interest and does not readily accommodate someone with ad-hoc or shifting interests.

All of those approaches address the problem of information overload, but none of them attacks the problem by creating a searchable and browsable network of incidents.

3. INCIDENT THREADING

From the description in Section 2, the previous research topics have addressed some problems in news organization, but each has its own deficiency and cannot provide an ideal solution for it. We believe that event threading, in particular, is on the right track, but it is still unnecessarily restrictive.

To have a clear view what is happening in a news stream, we may need to go below the topic level and analyze the internal structure of them. The individual news events should be created and organized into a fact network, where the edges show their actual relations instead of a yes/no judgment. Furthermore, events in different topics can also be related in some way (e.g., involving the same person, happening at the same place). By building such an event network, users can browse news events, see how the interesting topics appear, evolve and disappear in the news stream. They can also switch from one topic to another by following the links between them. This function is especially useful when new topics appear in the news stream, showing how they relate to other topics of interest.

The concept "event" is popular in information extraction (IE) where it has a different definition[1]. To avoid possible confusion of the concept, we name these news events *incidents*, and the graph that shows the relation among incidents as an *incident network*. The process of creating the incidents and generating networks is called *incident threading*. When we talk about *event*, *incident*, *fact* or *happening*, we refer to some incident that actually happened sometime in the real world and/or the union of all text passages that describe this happening.

3.1 Incident

Before we start talking about what incident threading is, or how we will implement a threading system, we need to define the basic concepts first.

1. News story: It is the basic unit in news distribution. Each story has a unique ID, a piece of text and source time, which marks when it was published. Some stories, mainly those from newswire, contain optional fields like title, headline or keywords.

2. Passage: Each story contains a text field, which is the union of multiple characters. A passage is the combination of consecutive characters that represents a complete description of some news event. One story can contain one or more passages.

3. Main characters (WHO): The most important named entities in a passage that show who or what is involved in the event described by the passage.

4. Time stamps (WHEN): We consider two types of time features. One is the publication time of the news, which is the same for all passages from one story. The other is the absolute or relative time point (or range) mentioned in the passage when the corresponding event happened. Sometimes the context is required to obtain the right time stamp, for descriptions like "on Wednesday", "this week", "when he was 18".

5. Location (WHERE): The geographical position where the event described in the passage happened. It is very

[1]An event in IE is an activity described by a sentence that involves zero or more entities. For example, "Israeli troops fought running gun battles with Palestinian civilians and security forces again today" describes an event, while "Israeli troops", "fought", "battles", "Palestinian civilians" and "security forces" are the text to extract. The event extraction task is usually limited to certain types of events (like conflicts) and focuses on accurate identification of their arguments. Different descriptions of the same semantic content are often handled separately.

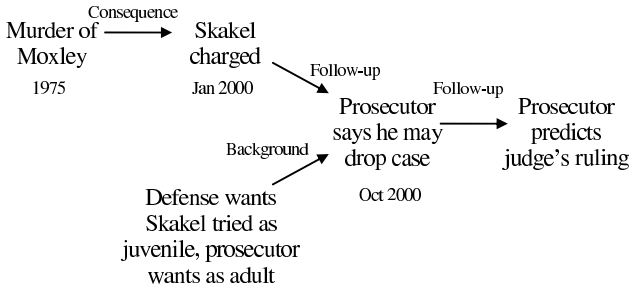

Figure 2: Sample incident network

common that the location information is unavailable in the passage, and inference from the context is required under that condition.

6. Action (WHAT): The key verb that describes the actual happening in the passage.

With the concepts above, we can define what an incident is. Suppose that we have a news collection of v passages $C = \{p_1, p_2, \ldots, p_v\}$. There are a total of t incidents $I = \{I_1, I_2, \ldots, I_t\}$ in this collection that satisfy:

$$\begin{aligned} \forall i \quad & I_i \neq \emptyset, I_i \in 2^C \\ \forall i \neq j \quad & I_i \cap I_j = \emptyset \\ \forall i \exists k \quad & p_i \in I_k \end{aligned} \tag{1}$$

The first constraint means that each incident is a non-empty set of passages (2^C is the power set of C, where each element of it is a possible subset of C), the second one specifies that the incidents do not overlap, and the last one says that each passage must belong to some incident.

Basically incidents are non-overlapping divisions of the passage collection. *An incident corresponds to a real-world occurrence that involves certain main characters, happening at specific time and location.* It is the union of passages that contain the same (or similar) features (e.g., who, when, where, what) and describe the same thing.

3.2 Incident network

In the earlier work of event threading [5], the connections between news events were established with binary links. It captured the intrinsic relations to some extent, but the yes/no relation was too simplified to show the news evolution accurately. Discourse analysis provides another framework that reflects the structure of a news report, and it can also be applied to modeling the relation of two incidents. Here we will borrow some concepts from it to build the incident network.

An incident network is one or more incidents that are connected by certain types of dependencies. Incidents in the network are arranged in a graph, and two incidents directly connected by an edge (directed or undirected) have some type of relation. Figure 2 shows an example of it. There are mainly three classes of connections, and we present them in decreasing order of anticipated difficulty.

The first class is called logical relations. Connections of this type specify that one incident is the necessary premise or inevitable result of another, as judged by a normal adult's experience. They are represented by directed edges in the incident network, and each edge goes from the logical premise to the result or consequence. Accurate identification of these types requires the ability to understand natural languages and in-depth analysis of human mind, which are very difficult to implement. We do not have any plan of semantic analysis for accurate logical relations. However, experiments in Section 6 suggest that simple rules with term features can also achieve acceptable accuracy for some types.

The second class of relations, named as progression, requires weaker links than the previous types. One incident may not necessarily lead to the other, but they involve the same main characters, happen at similar time and location. From the traditional TDT point of view, they discuss the same main topic and one follows the happening of the other. There is only one relation type in this class called "follow-up". Links in this class are shown as directed edges, pointing from the earlier incident to the later one.

The last class is called weak relations. It occurs when two incidents do not have a strong logical or progressive link between them, but contain some common factor(s), like involving the same person, happening at the same place. Dependencies in this class are represented by undirected edges in the incident network, because there is usually no priority or built-in order from the weak link of the overlapping feature.

The first two classes are strong relations that usually exist within news topics, and the last class mostly goes between topics and establishes a global incident network. It facilitates a user to navigate through the news, because the weak connections can lead the user from one interesting topic to another. It also helps to find all information that involves specific entities.

4. IMPLEMENTATION

In this section we will describe two ways to implement the incident threading system. One runs a clustering algorithm on a collection of passages to form incidents, then establishes links among them. The other considers the possible relations between two passages while both incidents and links are created together in a global optimization framework.

Clustering algorithms generate clusters in different granularities, and they can be news topics, subtopics or events. In TDT-2004, the traditional topic detection task was replaced by a hierarchical structure, and a hierarchical agglomerative clustering (HAC) algorithm with sampling achieved the highest performance in the evaluation [11]. Here we also use HAC to create incidents, but the links between them will be built with two different methods, incident similarity or pre-defined rules.

4.1 Baseline

The baseline method is very similar to the best "event threading" algorithm (*cos+TD+Simple-Thresholding* in [5]). However, time decay is not applied because it requires the duration of each TDT topic, which is not available in our experiments.

HAC is applied to a collection of n passages to generate the incidents. It starts with n singleton clusters where each cluster contains only one passage. In each round, the most similar cluster pair is merged, and the process goes on until the highest similarity falls below the preset clustering threshold. The similarity of two clusters is the average of all pair-wise passage similarities across the clusters.

Each passage is represented by its tf-idf term vector. There

are various ways to calculate term frequency (tf) and inverse document frequency (idf), and we use the formula in [2]. The tf component is,

$$\widehat{tf}_i = \frac{tf_i}{tf_i + 0.5 + 1.5 \times \frac{len_{pas}}{avg(len_{pas})}} \quad (2)$$

here tf_i is the frequency of term i, len_{pas} is the passage length, and $avg(len_{pas})$ is the average passage length in the whole collection. The idf component is,

$$idf_i = \frac{log\frac{n+0.5}{df_i}}{log(n+1.0)} \quad (3)$$

where df_i is the document frequency of term i, and n is the number of passages in the whole collection. The similarity of two vectors is calculated by the cosine of the angle between them,

$$cos(p_i, p_j) = \frac{\langle p_i, p_j \rangle}{\|p_i\| \times \|p_j\|} \quad (4)$$

After the HAC process stops, all incident pairs are compared. If the similarity between two incidents is over the link threshold, a link is created between them which points from the earlier incident to the later one. There is no type associated with the link.

4.2 Two-stage algorithm

The two-stage algorithm has similar procedure to the baseline, but additional features are introduced in the similarity calculation, and links are created by predefined rules instead of simple similarity.

In addition to the full text, there are other useful features in the content of a passage, including main characters, locations, time stamps, key verbs, etc. They are indexed as plain text in different fields of the passage, and can be viewed as various representations of the same description. With all these features, cosine similarity (Equation 4) of corresponding fields can be calculated between a passage pair, and the similarity of this pair is the weighted sum of similarities based on individual features,

$$Sim(p_i, p_j) = \sum_{k=1}^{l} w_k Sim_k(p_i, p_j) \quad (5)$$

where $Sim_k(p_i, p_j)$ is the similarity of p_i and p_j based on the k-th feature, and w_k is the weight associated with it. There are l features in total, and the weights of them add up to 1. The weights are empirically adjusted to achieve best performance, but the first feature (term vector of the full text) usually gets the most importance.

After the incidents are created, each passage is given a type label by a classifier (e.g. BoosTexter[9], which needs to be trained for individual types), and the type of an incident is determined by the labels of all passages in it. Next, some threading rules are applied to establish links among these incidents. In different scenarios, there can be various rules that apply to the individual circumstances. Table 1 shows the rules that will be used in our experiments. The fields in each rule are rule index, link type, type of incident 1, type of incident 2, additional requirements (sim: similarity higher than a given threshold, time: similarity over the threshold and incident 1 earlier than incident 2) and score upper bound (used later in Equation 16), respectively. Threading (link) rules will be described in more detail in Section 6.

Ind	Link	I1	I2	Req	Cap
1	Prediction	Prediction	General Damage	Sim	0.05
2	Consequence	General	Damage	Sim	0.05
3	Comment	General Damage	Comment	Sim	0.05
4	Preparation	Preparation	General	Sim	0.05
5	Follow-up	General	General	Time	0.05

Table 1: Threading rules in Science/Discovery

4.3 Global optimization

If two passages are randomly selected from a collection, there are two forces that interfere with each other and try to place them in their own designated location. One is the similarity between the passages, which tends to pull them together to merge them into the same incident. The other is their satisfaction of some rule that pushes them apart and places them into two connected incidents. These options are mutually exclusive, and each has some probability (or score) associated with it. For a passage pair, their relation can be in one of three possible states. They can be either in the same incident, connected by some relation, or not related at all. These relations can be encoded as: -1 (not related), 0 (in the same incident) or a positive integer (connected, with the value showing the link type).

When we expand the pair-wise competition to the whole collection, it becomes a global optimization problem. With a collection of n passages, an $n \times n$ relation matrix R can be established. Note that links are directional, so we encode R_{ij} and R_{ji} differently if they are positive: when there is a link of type r (see Table 1, $r = 1$ means a prediction link) going from incident i to j, $R_{ij} = 2r - 1$, and $R_{ji} = 2r$. Then we define a global score function on this relation matrix, where a larger score means better news organization.

$$S = \sum_{1 \le i,j \le n}^{i \ne j} score(i, j, R_{ij}) \quad (6)$$

In this equation,

$$score(i, j, R_{ij}) = \begin{cases} c & \text{if } R_{ij} = -1 \\ Sim(p_i, p_j) & \text{if } R_{ij} = 0 \\ Rule(p_i, p_j, R_{ij}) & \text{if } R_{ij} > 0 \end{cases} \quad (7)$$

Here c is a small constant assigned to unrelated passage pairs, $Sim(p_i, p_j)$ is the similarity of the passages calculated by Equation 5. $Rule(p_i, p_j, R_{ij})$ is a function that tells how well the passage pair fits the rule of relation R_{ij}, and its format differs by rule. Usually it is related to the incident types, passage similarity, and sometimes the time stamps of the passages.

The relation matrix R has n^2 parameters, but they are not completely independent. The relation between two passages should be symmetrical. When two passages are in the same incident, they must have the same relation to all other passages. The restrictions for the global optimization problem are,

$$\begin{aligned} \forall i,j,k \quad & R_{ij} = 0 && \Rightarrow R_{ik} = R_{jk} \\ \forall i,j \quad & R_{ij} \le 0 && \Rightarrow R_{ji} = R_{ij} \\ \forall i,j \quad & R_{ij} > 0 \wedge R_{ij} \equiv 1 \pmod 2 && \Rightarrow R_{ji} = R_{ij} + 1 \end{aligned} \quad (8)$$

With the restrictions in Equation 8, there is no explicit solution for the optimization problem of the global score function in Equation 6. Furthermore, it is too expensive to go over the whole solution space for a collection with reasonable size. Here we will use simulated annealing (SA) to search for the global maximum. The process of the SA algorithm is:

1. Initialize relation matrix R as all -1, except for the diagonal elements that are 0. Calculate initial score S. Set initial temperature T.
2. Record current best solution $RB = R$, $SB = S$.
3. while $(T > T_{min})$
   (a) Save current state $R_0 = R$.
   (b) Randomly select a passage i.
   (c) Select another passage j according to the distribution of R_{i*}.
   (d) Change the value of R_{ij}, update matrix R to satisfy the restrictions.
      i. $0 \rightarrow -1$: break a cluster into two
      ii. $0 \rightarrow +$: break a cluster into two, and select the relation that maximizes $Rule(p_i, p_j, R_{ij})$
      iii. $-1 \rightarrow 0$: merge two clusters
      iv. $-1 \rightarrow +$: build a link that maximizes $Rule(p_i, p_j, R_{ij})$
      v. $+ \rightarrow -1$: disconnect a link
      vi. $+ \rightarrow 0$: merge two clusters
   (e) Calculate new score SN.
   (f) If $(SN > SB)$, $RB = R$, $SB = SN$.
   (g) If $random() < e^{\frac{SN-S}{T}}$, keep the change of R. Otherwise, $R = R_0$.
   (h) $T = T \times constant$.
4. Return best solution RB.

5. EVALUATION

In order to evaluate how well the algorithms in Section 4 work, we need to compare their output to *ground truth*. Since there is no existing work that processes news in the same way, we do not have any collection that is annotated at the incident level. We hired some annotators to mark up part of the TDT collections. They were asked to identify passages that describe the same real-world happening and build links between related incidents with a limited type vocabulary. Section 6 provides more details.

There are different ways to represent the annotated data. We can generate a list of clusters (incidents) each including one or more passages, and a list of links among these clusters which indicates their relations. However, system-generated clusters are usually different from the annotated data, and it is a nontrivial task to establish the correct mapping between them. It also brings great difficulty to the evaluation of links. Another choice is to create the relation matrix of all passages, where elements in the matrix are encoded in the same way as in the global optimization framework.

With a relation matrix R from an algorithm and another one RT from the annotation, the clustering accuracy of incidents can be defined as,

$$\begin{aligned} Prec_{cluster} &= P(RT_{ij} = 0 | R_{ij} = 0) \\ Rec_{cluster} &= P(R_{ij} = 0 | RT_{ij} = 0) \\ F_{cluster} &= \frac{2 \times Prec_{cluster} \times Rec_{cluster}}{Prec_{cluster} + Rec_{cluster}} \end{aligned} \tag{9}$$

The probabilities are averaged over all passage pairs. Similarly, we can randomly select two different passages i and j, and the accuracy for binary links (ignoring the type) is,

$$\begin{aligned} Prec_{bin} &= P(RT_{ij} > 0 \wedge R_{ij} \equiv RT_{ij} \pmod 2 | R_{ij} > 0) \\ Rec_{bin} &= P(R_{ij} > 0 \wedge R_{ij} \equiv RT_{ij} \pmod 2 | RT_{ij} > 0) \\ F_{bin} &= \frac{2 \times Prec_{bin} \times Rec_{bin}}{Prec_{bin} + Rec_{bin}} \end{aligned} \tag{10}$$

When the link types are taken into consideration,

$$\begin{aligned} Prec_{link} &= P(R_{ij} = RT_{ij} | R_{ij} > 0) \\ Rec_{link} &= P(R_{ij} = RT_{ij} | RT_{ij} > 0) \\ F_{link} &= \frac{2 \times Prec_{link} \times Rec_{link}}{Prec_{link} + Rec_{link}} \end{aligned} \tag{11}$$

From the experiments, there is usually a tradeoff between the accuracy of clusters and links. When the performance of one gets higher, the other often decreases with it. A balance is required between these two for an overall evaluation measure. A perfect output should get a relation matrix identical to RT, and a mismatch between the corresponding elements of R and RT means an error. There are three types of values in the relation matrix, -1 (not related), 0 (cluster) and positive (link), so it is natural to compare the distribution of them in the two matrices (R and RT).

We start by defining a function that describes how well two relations match each other. If they have different signs, it is obviously a mistake. Otherwise, it should be correct (for 0 and -1) or possibly correct (for positive integers). An identical link requires both its existence and the right type, so we assign it a higher score if both match, but a link in the wrong type also gets partial score. Note that most elements in the relation matrix are -1's, so the sum over the whole matrix is easily overwhelmed by the type of non-related passage pairs. To get approximately equal contributions from all three types (25% from clusters, 25% from unrelated pairs and 50% from links), the number of elements in each class is counted in the truth matrix RT, and the matrix matching function is adjusted to model the distribution fairly.

$$match(x, y) = \begin{cases} 0 & \text{if } sgn(x) \neq sgn(y) \\ 2N_0/N_+ & \text{if } x = y > 0 \\ N_0/N_+ & \text{if } x > 0, y > 0, x \neq y \\ 1 & \text{if } x = y = 0 \\ N_0/N_- & \text{if } x = y = -1 \end{cases} \tag{12}$$

here $sgn(\cdot)$ is the sign function, which returns 1 if positive, 0 if zero and -1 if negative. N_+ is the number of positive elements in RT, N_0 is the number of 0's, and N_- is the number of -1's.

With the function in Equation 12, we can get an overall score that models how well R matches RT,

$$M(R, RT) = \frac{\sum_{i \neq j} match(R_{ij}, RT_{ij})}{\sum_{i \neq j} match(RT_{ij}, RT_{ij})} \tag{13}$$

The result ranges from 0 to 1; a perfect match returns 1. If a system outputs one big cluster that includes everything, R would be all 0's. $match(R_{ij}, RT_{ij})$ returns 1 for

the N_0 zero elements in RT, and 0 for others. On the other hand, $match(RT_{ij}, RT_{ij})$ gets N_0 1's, N_+ $2N_0/N_+$'s and N_- N_0/N_-'s. The matrix matching score for such a system is 0.25.

The matrix comparison evaluation is disadvantageous towards the baseline system, since it does not generate any relation type. This method also favors high accuracy in the link type, while a match in link type is given additional bonus over finding the correct link itself. We believe that it is the right assignment because the contextual information reflected in the link types is more important for getting the incident networks correct. Some relations are more important than others, but we are ignoring that factor to avoid arbitrary weight assignment.

6. EXPERIMENTS

The collection used in the experiments is part of TDT-3[2]. TDT topics come from different scenarios (rules of interpretation or ROIs used by the LDC annotators to classify topics). Each scenario has its own characteristic event organization. For example, legal cases usually have crimes, investigations, arrests, trials, etc, and the relations among them are generally fixed. Schank and Abelson [8] found similar phenomena in the understanding of human knowledge, and they created scripts for scenarios in real life (e.g., restaurant script[3]). Here we borrow the term "script" and generate one script for each ROI, which includes a list of rules for possible link types under that scenario.

Six topics were selected from the TDT-3 corpus that all deal with reports in science and discovery. We found it difficult for annotators to agree on passage boundaries, so for this first study we elected to treat each story as a passage (a simplification we are currently working to remove). Annotators were first asked to group stories into incidents so that all stories in an incident talk about the same event (as described in Section 3). They were then asked to assign each of those incidents to one of the following classes: *comment, prediction, damage, background, preparation*, or *general*.

- **Comment**: Verbal or written evaluation of some real-world happening.
- **Prediction**: An assumption that something will or will not happen in the future.
- **Damage**: Loss in any type caused by a real-world happening.
- **Background**: Additional information that helps people understand the context. Usually it is some general knowledge that is not associated with any time or location.
- **Preparation**: The process of making something ready for an event in the future.
- **General**: An event that does not fit in any of the categories described above.

[2]Available from the linguistic data consortium (LDC), catalog number LDC2001T58.

[3]The main steps in the restaurant script include: customer enters restaurant, customer finds seat, customer sits down, waiter/waitress gets menu, etc.

ROI-7: Science/discovery	
Topics	6
Total size	280
Topic sizes	52, 43, 158, 77, 2, 6
Language	English
Source	Newswire, broadcast
Incidents	30
N_-	32875
N_0	2583
N_+	3602

Table 2: Experiment corpus

As a final step, annotators linked appropriate pairs of incidents and selected a label for the link (thread) from: *prediction, consequence, comment, preparation*, or *follow-up*.

- **Prediction**: The first incident predicts the happening of the second one, and the second incident indicates the actual outcome.
- **Consequence**: The first incident directly causes the second to happen, and the latter describes some gain or loss.
- **Comment**: The second incident provides verbal or written commentary on the first one.
- **Preparation**: The first incident describes the process of making something ready for the second.
- **Follow-up**: The second incident happens after the first and they are closely related, but their relation does not satisfy any rule described above.

Some of the incident and link types are adapted from discourse analysis (see Figure 1), and the rules in this ROI are listed in Table 1. More information about the experiment corpus is in Table 2.

All three algorithms in Section 4 are implemented in our experiments. The stories from the six different topics form the collection (with duplicates removed). In the baseline and the two-stage algorithm, we opt to follow the pattern used by the annotators: first incidents are created with clustering, then links are established based on incident similarity or rules. Simulated annealing takes the stories as the basic units in the global optimization framework.

Heuristic information is used in many of our algorithms, functions and parameters, and much of the knowledge comes from our observation of the whole collection. Therefore, we are unable to leave out a strict "test" set without utilizing any information from it. All evaluation results reported are performance numbers for the whole experiment collection, and they only suggest how well we can do in this specific corpus. On the other hand, accuracy is not very sensitive with slight changes of most parameters, so these are good indications of actual performance.

In the two-stage algorithm, similarities based on different features are combined with Equation 5, and the weight assignment is,

$$(w_{term}, w_{who}, w_{where}, w_{when}, w_{what}) = (0.8, 0.1, 0.1, 0, 0) \quad (14)$$

Algorithm	Baseline	Two-stage	SA
$Prec_{cluster}$	0.330	0.390	0.248
$Rec_{cluster}$	0.540	0.547	0.455
$F_{cluster}$	0.410	0.455	0.321
$Prec_{bin}$	0.113	0.069	0.131
Rec_{bin}	0.282	0.360	0.402
F_{bin}	0.161	0.116*	0.198
$Prec_{link}$	0	0.049	0.118
Rec_{link}	0	0.254	0.361
F_{link}	0*	0.082*	0.178
$M(R, RT)$	0.402*	0.482	0.509

Table 3: Evaluation results of three different systems: baseline, two-stage and simulated annealing (micro-average)

Algorithm	Baseline	Two-stage	SA
$Prec_{cluster}$	0.548	0.535	0.588
$Rec_{cluster}$	0.795	0.924	0.632
$F_{cluster}$	0.611	0.654	0.588
$Prec_{bin}$	0.164	0.126	0.372
Rec_{bin}	0.109	0.048	0.447
F_{bin}	0.121	0.067*	0.396
$Prec_{link}$	0	0.066	0.156
Rec_{link}	0	0.030	0.217
F_{link}	0*	0.040*	0.177
$M(R, RT)$	0.418*	0.442	0.519

Table 4: Evaluation results of three different systems: baseline, two-stage and simulated annealing (macro-average)

The features are extracted by an automatic content extraction (ACE) system from New York University[4]. The last two features are not used since the extracted text is relatively short in comparison to others, and the corresponding similarity matrices are very sparse. After the incidents are created, majority voting is used to decide the type of the incident. A classifier is trained by BoosTexter to assign the label to each story, and a 3-fold cross-validation achieves 17% error rate.

The simulated annealing algorithm usually gets different results from various runs, so we start it 20 times and take one with the best result. The temperature starts at 100 and decreases 1% in each iteration. The process stops when the temperature is lower than 0.0001 or when the current state has not been changed in 50 steps (local maximum). The similarity of different features is combined in a different way from in the two-stage algorithm,

$$(w_{term}, w_{who}, w_{where}, w_{when}, w_{what}) = (0.9, 0.1, 0, 0, 0) \quad (15)$$

For most rules,

$$\begin{aligned} Rule(p_i, p_j, R_{ij}) &= sat(p_i \doteq R_{ij}^{I1}, p_j \doteq R_{ij}^{I2}) \\ &\quad \times min(Sim(p_i, p_j), R_{ij}^{cap}) \end{aligned} \quad (16)$$

where $sat(\cdot)$ is a function that assigns a weight according to the number of true predicates, $\doteq$ means that the passage is in the same type as the incident in the rule, and R_{ij}^{cap} (the last field in Table 1) is an upper bound of the similarity. Rule 5 in Table 1 also requires the time stamps of the incidents to be in the right order. The type labels of passages are assigned by the BoosTexter classifier.

Because of the randomness of SA, it usually takes more time for each run. While the baseline and two-stage algorithm can finish in under 10 seconds, SA usually takes about 2-10 minutes per run on the same server (about 2GHz single CPU, 2GB memory), depending on how fast it converges. Its speed will become the bottleneck for a large collection.

7. RESULTS

Table 3 shows the performance of the three different algorithms, with averages on a story-pair basis. Significance testing was done on a per-topic basis, and SA is better than the other two in link quality and overall score. With a one-tail t-test at 95% confidence level, SA achieves significant improvement over both in F_{link}, the two-stage algorithm in F_{bin} and the baseline in $M(R, RT)$ (numbers with * in the table). Table 4 lists the performance of the same systems, but evaluated on a topic-based averages. Although the numbers are different, the comparison between corresponding algorithms is very similar. Due to the limited size of our experiment collection, we are hesitant to claim that the results are more than suggestive, so the actual significance test P-values are not listed here.

From the comparison in Table 3 and 4, we can see that the traditional clustering algorithms achieve better performance for establishing incidents, while simulated annealing captures better links. When the similarity between two passages (or stories) is high, it is natural to assume that they are talking about the same thing and place them into the same incident. However, clustering algorithms do not consider the alternative choice - maybe they are similar simply because they mention the same topic. The main disadvantage of the two-stage algorithm is, once a cluster is formed, it becomes impossible to break it and model the relations among its different parts. On the other hand, SA considers both options at the same time, and tries to optimize the global incident network in a competitive process. We believe it is a better strategy for this application. However, the F-values for links are still low, especially for the clustering algorithms. We know that finding contextual links without semantic understanding of languages is a very difficult task, and the numbers look promising given the limited information used in the experiment.

Another observation is that the use of additional features improves clustering performance. Two-stage and the baseline use the same clustering algorithm, but the former gets higher accuracy (not significant) with named entities (who) and location information (where). Time stamps and key verbs did not show any help in our experiments, but they may still be useful if applied in the right way.

It is disappointing to see that two-stage gets lower accuracy in binary links, although it has the advantage of generating the actual link type. The high error rate of BoosTexter seems to be the main reason, as the training data are quite skewed (there are too many *general* passages and only a few for *comment*, *background*, etc.). We expect a lower error rate when the collection gets larger. Furthermore, a classi-

[4]Proteus, http://nlp.cs.nyu.edu/index.shtml

fier that outputs probabilities instead of deterministic labels will alleviate the impact of erroneously classified elements.

8. CONCLUSIONS

In this paper, we have presented a news organization infrastructure that is different from the traditional topic-based models. We believe that a flat list of news topics cannot provide users with enough information how a topic emerges, evolves and ends in the real world. However, with accurate identification of the content-coherent incidents and relations among them, the news organization is clearly displayed.

In addition to the framework, we propose a few algorithms to form the incident network. The two-stage method extends the clustering algorithm with scenario-specific rules, but the performance of finding links is greatly limited by the clustering accuracy. Based on the tension between possible relations of a passage pair, we also design a global optimization problem that takes all possible configurations into consideration. Experiments show that simulated annealing in the global optimization framework consistently provides more accurate links than the two-stage algorithm, at the cost of lower accuracy in clusters and higher computational complexity. For the application of incident threading, contextual information is more important, so global optimization is preferred over the traditional clustering algorithm.

The score function in Equation 7 reflects the competition between different relation types, but failure analysis shows that there are still many cases where the score cannot correctly match the truth relations. More analysis is necessary for better modeling of semantic information, where additional features and careful tuning of parameters are required.

Due to the difficulty in annotation, the experiments were based on whole news stories. From our observation, parts of a news story are often about different things, so passage-level news analysis will be more ideal for news organization. We are looking forward to building a passage-based incident threading system, and hopefully it will yield more accurate incidents and links.

The data collection in the experiments is small and limited to a single scenario. While we expect similar performance under other conditions, we need more data from multiple ROIs and create the corresponding rules for each of them.

The application of additional features extracted from the text proves useful in our experiments. However, the simple weighted sum may not be the ideal way to use them, and there are still some features that do not work well with this method. Further research is necessary to analyze the merit of those features.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the sponsor.

9. REFERENCES

[1] J. Allan, editor. *Topic Detection and Tracking: event-based information organization*. Kluwer Academic Publishers, 2002.

[2] M. Connell, A. Feng, G. Kumaran, H. Raghavan, C. Shah, and J. Allan. UMass at TDT 2004. In *Proceedings of TDT 2004*, 2004. www.nist.gov/speech/tests/tdt/tdt2004/papers/UMass-TDT2004-paper.pdf.

[3] F. Kilander. A brief comparison of news filtering software. Unpublished paper, 1995.

[4] K. R. McKeown, R. Barzilay, D. Evans, V. Hatzivassiloglou, J. L. Klavans, C. Sable, B. Schiffman, and S. Sigelman. Tracking and summarizing news on a daily basis with Columbia's Newsblaster. In *Proceedings of the Human Language Technology Conference*, 2002.

[5] R. Nallapati, A. Feng, F. Peng, and J. Allan. Event threading within news topics. In *Proceedings of ACM Thirteenth Conference on Information and Knowledge Management*, pages 446–453, 2004.

[6] D. E. O'Leary. The Internet, intranets, and the AI renaissance. *Computer*, 30(1):71–78, 1997.

[7] D. Radev, J. Otterbacher, A. Winkel, and S. Blair-Goldensohn. NewsInEssence: Summarizing online news topics. *Communications of the ACM*, 48(10):95–98, 2005.

[8] R. C. Schank and R. P. Abelson. *Scripts, Plans, Goals, and Understanding: an Inquiry into Human Knowledge Structure*. Lawrence Erlbaum Associates, 1977.

[9] R. E. Schapire and Y. Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[10] I. Soboroff. Overview of the TREC 2004 novelty track. In *The Thirteenth Text Retrieval Conference*. NIST, November 2004. http://trec.nist.gov/pubs/trec13/papers/NOVELTY.OVERVIEW.pdf.

[11] D. Trieschnigg and W. Kraaij. Scalable hierarchical topic detection: exploring a sample based approach. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 655 – 656, 2005.

[12] T. A. van Dijk. *News as Discourse*. Lawrence Erlbaum Associates, 1988.

[13] Y. Yang, J. Carbonell, R. Brown, T. Pierce, B. T. Archibald, and X. Liu. Learning approaches for detection and tracking news events. *IEEE Intelligent Systems Special Issue on Applications of Intelligent Information Retrieval*, 14(4):32–43, 1999.