From Topics to Narrative Documents

Management and Personalization of Topic Collections

Christine Müller

Department of Computer Science, Jacobs University, Bremen, Germany
c.mueller@jacobs-university.de

Abstract. The paper proposes a document planning approach that structures topic-oriented materials into user-specific, narrative documents. This is achieved by introducing narrative flows into topic collections and by identifying variant relations between topics. Technically, topic collections are modeled as graphs, where nodes correspond to topics and edges denote semantic dependencies, narrative flows, and variant relations between the topics. These graphs are traversed to produce narrative documents. To personalise the traversal, user contexts are considered and define the users' structure and content preferences. For illustration purposes the approach has been applied to a collection of learning resources.

1 Introduction

Modern web technologies have revolutionized the WWW and transformed it into a more social, user friendly, and flexible network. Users became media producers and web applications became more open and social, while at the same time improving their mutual integration. Thanks to their high usability in terms of content creation, software tools, such as wikis, blogs, forums, web annotations, and bookmarking tools, have acquired a mass of user-specific information that users can share among each other.

Most of this information is authored in a **topic-oriented** fashion: Users start by writing self-contained, independent text paragraphs (called **topics**). These can be interlinked with other topics as well as tagged or bookmarked. Other users can explore the resulting network of topics along the hyperlinks, tags, and bookmarks. However, in order to find, explore, and track information, users have to be able to filter and structure web content. The technique of sharing tags and bookmarks offers an interface for this. It allows systems to match bookmarks and to improve searching. However, users still have to dig through the tag clouds and have to filter relevant and useful information.

Essentially, users lack the coherent, consistent, and well-researched structure that conventional media like books and courses provide. This work claims that users could benefit from a service that assembles such documents from topic-oriented knowledge collections. For example, imagine that we could simply tell a computer to convert a number of Wikipedia articles into a personalized textbook, which satisfies our individual information needs and places information into a consistent story. Alternative, envisage that we could convert forum discussions such as [5] or developer networks like [10] into well-structured manuals.

This work aims at providing such kind of **document planning** services by structuring self-contained information units into *narrative* documents. These documents support users by providing a **narrative context** for the information presented. To assure the relevance and usefulness of the conveyed knowledge in these documents, they are tailored to a **user context** – a commonly known term that defines the users' information needs, preferences, background, etc.

2 Narrative vs. Topic-Oriented Paradigm

In document management we have always dealt with **narrative writings** like textbooks or novels. Respective documents usually include an introduction, a successive exploration of new ideas, reviews, and references [13]. The authoring of narrative writings follows a top-down approach, e.g., from a document, to chapters, to sections, to subsections, to visual document parts like examples or definitions, and, finally, to paragraphs. This nesting of information units forms a tree structure, henceforth referred to as **narrative structure** (short *structure*). To support a traditional document layout, the content of a document is linearized into a **narrative flow** by traversing the narrative structure from left to right[1]. The information units along the narrative flow are supported by preceding units and include multiple cross-references as well as narrative transitions (like "as we have seen above") to other units that improve the coherence of the text: All parts are neatly connected, the narrative flow guides the reader through the writing.

Nevertheless, though document-centered writing are very well suited to be read by humans, they are also limited to predefined structures and selections of material that do not adapt to a user context. Cross-references and transitions reduce the reusability of content and hamper the modularization of documents - two important prerequisites for the personalisation of documents.

The topic-oriented approach is based on the principles of reuse and modularization. It is followed by encyclopedias and has become particularly famous with the rise of modern web technologies like wikis and neatly all elearning systems that follow the *learning object paradigm* [14], an instance of the topic-oriented approach. Most document generation and personalization approaches focus on topic-oriented knowledge bases. A famous example are template-based generation approaches[2]: Templates are used to predefine structures and are filled with context-specific content to produce a concrete, human-readable instance. Unfortunately, most template-based approaches are restricted, in particular, the selection of appropriate content follows hard-coded routines and is based on predefined context parameters like communication channels, customer groups, etc. Also, since topics omit narrative cross-references and transitions, the resulting topic-oriented documents lack coherence.

Neither topic-oriented nor document-centered approach leads to a document management infrastructure, which supports modularization *and* coherence. To address this challenge, the author proposes to bridge the two paradigms and to combine aspects of both worlds. [11] explores one way: The topic-oriented principles of reuse and modularization are applied to the narrative world. A framework is proposed that supports

[1] Following [4], narrative structures are considered as ordered trees.

[2] See [11] for references and an analysis of other systems.

the modularization of narrative documents as well as the user-specific adaptation on all three document layers: the presentation, content, and structure layer [3].

This paper focuses on the other way: The introduction of narrative structures into topic-oriented knowledge bases to support the planning of *coherent* (i.e., narrative) documents. In addition, the topic-oriented material is enriched with variant relations between equivalent topics from which user-specific ones are selected during the document assembly. To illustrate and evaluate the proposed adaptation services, mathematics is used as test tube. In particular, the work is applied to the exercise collection of a theoretical computer science lecture.

The most important prerequisite for the proposed document planning approach is a representation of topics, narrative structures, and variant relations, which makes them comprehensible to a computer system. For the representation of topics we draw on XML markup languages, in particular, the mathematical format OMDoc [8]. For the representation of narrative structures and variant relations we use the XML encoding as proposed by [11].

3 Modeling Topic Collections as Topic Graphs

Having selected mathematics for the planning of documents from topic collections has turned out to be very beneficial. Mathematical knowledge is precise, highly-structured as well as extraordinarily interlinked and can thus be modelled easier than knowledge from other domains. Moreover, mathematical formats like OMDoc [8] thoroughly mark the **semantic structure** of mathematical knowledge and illustrate that the topic-oriented approach is very natural for mathematical knowledge. For example, OMDoc places mathematical topics (e.g., mathematical statements like lemmas, proofs, and examples) into larger structures that provide them with a mathematical context. These structures are referred to as **theories** and are linked via **theory morphisms** [12].

This modularization of mathematical knowledge is not only applied to theory objects but also to their constituents, i.e., statements like proofs, definitions, and examples. For example, OMDoc classifies such statements and marks the semantic dependencies between them. Drawing on XML technologies like XPath [4] and XPointer [6], any specifically marked aspect of the underlying representation format can be extracted. The algorithms proposed in this paper model mathematical topic collections as **topic graphs**, where nodes correspond to theories and statements and edges denote their semantic dependencies (i.e., theory morphisms between theory nodes and dependencies like 'illustrates' and 'proves' between statement nodes). Parent-child edges represent the nested structure of theory objects and their constituents in the topic graph.

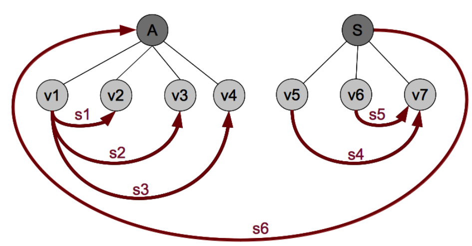

Fig. 1. An example topic graph

Figure 1 exemplifies a topic graph. The nodes A and S denote mathematical theories which are connected via a theory morphism: Theory A defines the algorithm for constructing a spanning tree and thus builds on theory S, which defines and exemplifies the concept 'spanning tree'. The theories A and S embed the nodes v_1 to v_7, which denote definitions and exercise interrelated via semantic dependencies. The definition v_1 is illustrated by the exercises v_2, v_3, and v_4. The exercise v_7 illustrates the two alternative definitions v_5 and v_6.

4 From Topic Graphs to Narrative Variant Graphs

As we learned before, topics are self-contained information units that omit cross-references and narrative transitions - two important aspects of narrative documents. In order to convert topic collections into narrative documents, topics graphs have to be enriched by narrative dependencies between the theories and their statements. The resulting graphs are called **narrative graphs**.

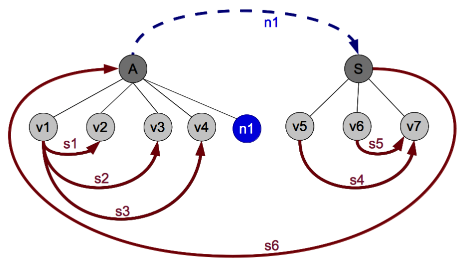

Fig. 2. An example narrative graph

Figure 2 illustrates the topic graph from Figure 1 extended by narrative flows. The nodes A and S are connected via the narrative edge `n1`, which denotes that in narrative terms the theory S builds on theory A. The theory A was enriched by a node with label `n1`, which represents the transitional text "We will now define the term spanning tree".

In order to support the generation of *user-specific* narrative documents, we need to enrich narrative graphs with variants [11]. This includes variant theories and statements (called **content variant**) as well as alternative narrative flows (called **narrative variants**). The resulting graphs are called **variant graphs**.

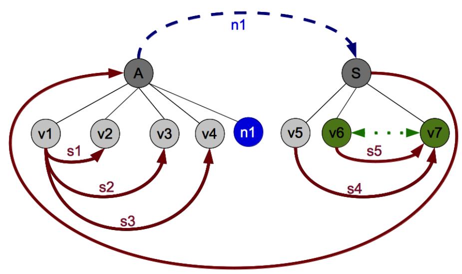

Fig. 3. An example variant graph

Figure 3 illustrates the extended narrative graph from Figure 2. A variation on the content level is marked: Definition v_6 and v_7 are variants. Definition v_6 originates from a textbook and definition v_7 was retrieved from Wikipedia. The variants allow an adaptation engine to select the most appropriate definition for a user depending on his content preferences.

5 Terminology

We adapt the terminology in [11] for planning documents from topic collections.

A **context parameter** is a key-value pair ($d = v$), where d denotes a context dimension and v its context value. A context parameter can define properties, e.g., ($\texttt{language} = \texttt{en}$) and relations, e.g., ($\texttt{more\_difficult\_than} = \texttt{exKruskal}$). Context dimensions are represented as mathematical symbols [2], context values denote mathematical symbols for property values and references for relation values [11].

A **context annotations** λ is represented as ordered set of context parameters.

An **adaptation context** Λ represents the ordered set of context parameter (or constraints). It defines a user's content and structure preferences. The order of a context parameter cp_s in Λ denotes its priority w for the adaptation (or **weight**), which is computed by subtracting the position pos of cp_s from the total number of context parameters in Λ, i.e., $w(cp_s) = size(\Lambda) - pos(cp_s)$.

A **topic** ι is an addressable, self-contained text paragraph of arbitrary size. Topics that include other topics (e.g., theories) are represented as **nested topics**: $\iota = (\langle \iota_1, ..., \iota_i \rangle, \lambda, \#)$. Plain texts (e.g., statements) are represented as **atomic topics**: $\iota = (\langle C_1, ..., C_i \rangle, \lambda, \#)$. A topic is represented as tupel: The first component is the *content* of a topic. It is represented as a sequence of topics (for nested topics) or a sequence of characters (for atomic topics). The topics $\iota_1, ..., \iota_i$ are also called the children of ι. The second component is the *context annotation*. It specifies the properties of the topic and its relation with other topics. The third component is a *unique identifier* $\# = \langle C_1, ..., C_i \rangle$, which is represented as sequence of characters [9].

A **topic graph** $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a simple, directed, labelled multigraph, where $\mathcal{V}$ is a set and $\mathcal{E}$ is a set of ordered pairs of elements from $\mathcal{V}$. The elements of $\mathcal{V}$ are called nodes and correspond to topics. The elements of $\mathcal{E}$ are called edges and represent semantic dependencies and parent-child relations between the topics. A graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ is called the **sub-graph** of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if $\mathcal{V}' \subseteq \mathcal{V}$ and $\mathcal{E}' \subseteq \mathcal{E}$, short $\mathcal{G}' \sqsubseteq \mathcal{G}$.

An **edge** $t = (\#, \tau, \lambda, \{\overrightarrow{\iota}_1, ..., \overrightarrow{\iota}_i\})$ is a tupel, where the first component is a *reference* $\#$ that identifies the corresponding semantic dependency, narrative flow or variant relation, the second component is the *edge type* τ, and the third component the *context annotation* λ of the edge. The fourth component is optional and represents a set of *transitional text* $\{\overrightarrow{\iota}_1, ..., \overrightarrow{\iota}_i\}$ that can be associated with a narrative flow.

A **narrative graph** is a topic graph extended with edges that represent narrative flows[3]. A **variant graph** is a narrative graph extended with edges representing variant relations between the nodes. A **narrative walk** $\mathcal{W}$ in $\mathcal{G}$ is a sequence $\langle v_1, ..., v_k \rangle$ of nodes of $\mathcal{G}$, such that $\mathcal{G}$ contains edges $e_l(v_i, v_{i+1})$ for all $i = 1, ...k$ with equal reference $\#$. A narrative walk is called **narrative path** P, if all nodes $v_1, ..., v_k$ are distinct. $\mathcal{N}$ denotes the **set of narrative walks** in $\mathcal{G}$.

6 From Graphs to Narrative Documents

To create narrative documents, variant graphs are traversed along the narrative edges between their nodes, while considering the parent-child edges between the nodes. This work proposes an iterative traversal that starts with the nested topics of the graph (representing mathematical theories) and then traverses their constituents (the mathematical statements of these theories). We thus first construct a graph of mathematical theories and use the respective narrative flow to create the coarse-grained components of a document, i.e., a sequence of section. In further iterations, the graphs of the theory constituents are traversed and used to create a narrative flow throw the content of each section[4]. The following pages specify the traversal algorithm, which are based on [11].

[3] Narrative flows are represented by one or many edges with equal reference.

[4] A nesting of sections and subsections is created by considering theories and sub-theories

Listing 1. Hybrid traversal for $(\mathcal{G}, \Lambda)$

```
let P = ⟨⟩

while not all nodes of 𝒱 in P do
  n = last node of P
  P′ = get narrative path for (n, P, 𝒢, Λ)

  if P′ is none then
    𝒱′ = 𝒱 \ P
    𝒢′ = (𝒱′, ℰ)
    P′ = get semantic path for 𝒢′
  fi

  append P′ to P
done
```

Listing 1 specifies the traversal algorithm. It takes as input the variant graph $\mathcal{G}$ and the adaptation context Λ. It returns the path P or `none`. As long as not all nodes in $\mathcal{V}$ are visited by P, the following steps are repeated. The algorithm first selects the last node n in P and calls the subroutine `get narrative path` in Listing 2. It returns the longest path P' from the set of narrative walks $\mathcal{N}$ in $\mathcal{G}$, preferably a path that starts with n. The path P' is appended to P. The append function omits all nodes at the beginning of P' that occur in this order at the end of P. For example, $P = \langle v_5, v_6, v_7 \rangle$ and $P' = \langle v_7, v_8 \rangle$ are merged to $P = \langle v_5, v_6, v_7, v_8 \rangle$[5].

Listing 2. Get narrative path for $(n, P, \mathcal{G}, \Lambda)$

```
let P′ = longest path ⟨v₁, ..., vⱼ, ..., vᵢ⟩ in 𝒩 where
  P′ starts with n and
  λ₁, ..., λₙ best match with Λ and
  (v₂, ..., vᵢ are not in P xor
   v₁, ..., vⱼ in P = ⟨u₁, ..., uₖ⟩ where j < k and ⟨v₁, ..., vⱼ⟩ equals
       ⟨uₖ₋ⱼ, ..., uₖ⟩)
done

if P′ is none then
  P′ = longest path ⟨v₁, ..., vⱼ, ..., vᵢ⟩ in 𝒩 where
    λ₁, ..., λₙ best match with Λ and
    (v₁, ..., vᵢ are not in P xor
     v₁, ..., vⱼ in P = ⟨u₁, ..., uₖ⟩ where j < k and ⟨v₁, ..., vⱼ⟩ equals
         ⟨uₖ₋ⱼ, ..., uₖ⟩)
  done
fi

return P′
```

[5] [11] proposes a hybrid traversal that also considers the semantic dependencies between the remaining nodes in $\mathcal{V}$ that are not connected via narrative edges. [11] also specifies a context-based sequencing of the graph as fallback that simply orders the nodes of $\mathcal{G}$ according to how well their context annotations match the adaptation context.

Listing 2 specifies the algorithm for finding a narrative path. It takes as input the node n, the path P, the graph $\mathcal{G}$, and the context annotation Λ. It outputs a path P'. The path is selected from $\mathcal{N}$, the set of narrative walks in $\mathcal{G}$. In a first step, the algorithm tries to select the longest path $P' = \langle v_1, ..., v_j, ..., v_i \rangle$ from $\mathcal{N}$. The path has to start with node n, the nodes context annotations $\lambda_1, ..., \lambda_n$ should best match with Λ (Listing 3), and either the nodes $v_2, ..., v_i$ must not be on P or, given the path $P = u_1, ..., u_k$, the sequence $\langle v_1, ..., v_j \rangle$ at the beginning of P' has to be equal to the sequence $\langle u_{k-j}, ..., u_k \rangle$ at the end of P. The latter condition can be removed to construct a walk. If no path can be selected, the algorithm tries to select a longest path from $\mathcal{N}$ with arbitrary start node v_1, where the context annotations $\lambda_1, ..., \lambda_n$ best match with Λ (Listing 3). The path or `none` is returned.

Listing 3. Compute match value for (P, Λ)

```
let w(P) = 0

forall v_i in P do
  w(λ_{v_i}) = 0

  forall cp_j in λ_{v_i} do
    if cp_k in Λ and cp_k satisfies cp_j then
      w(cp_j) = order of cp_k
    fi
    add w(cp_j) to w(λ_{v_i})
  done

  w(P) = w(P) + w(λ_{v_i})
done

return w(P) : size of P
```

The herein proposed approach supports authors to provide a variety of narrative flows between the nodes of a graph from which an appropriate alternative is selected and the appropriate transitional texts are displayed. To select an appropriate narrative flow, the context annotations of the nodes on a narrative path are matched with the adaptation context: the match value of a path is the average weight of its nodes. Listing 3 illustrates the matching, which adds up the weights of the nodes v_i on a path P. To compute a weight for a node, the context parameters cp_j in its context annotation λ_{v_i} are processed. All cp_j in λ_{v_i} that `satisfy` a context parameter cp_k in Λ are weighted with the order of cp_k [11]. The weight of a node v_i is computed by adding up the weights of the context parameters in λ_{v_i}. This weight is added to the weight of the path $w(P)$. After all weights have been added up, $w(P)$ is divided by the size of the path.

7 Conclusion

The paper proposes a document planning approach that structures topic-oriented materials into user-specific, narrative documents. This is achieved by introducing narrative flows into topic collections and by identifying variant relations between topics. Technically, topic collections are modeled as variant graphs, where nodes correspond to topics and edges denote semantic dependencies, narrative flows, and variant relations between the topics. These graphs are traversed to produce narrative documents. To personalise the traversal, user contexts are considered and define the users' structure and content preferences.

The proposed algorithms have been implemented in the *adaptor* library [1], which integrates JOMDOC [7] for the handling of OMDOC materials. The library has been integrated in the *panta rhei* system [11], which demonstrates the planning of a small corpus of learning resources. These learning resources are taken from a theoretical computer science course at Jacobs University as well as Wikipedia and are represented in the mathematical document format OMDOC. Further work has to apply the proposed algorithms to a large topic collection.

References

1. Adaptor – A Java Library for reordering OMDoc documents. Retrieved from https://trac.kwarc.info/panta-rhei/wiki/adaptor on Feb 28, 2010.
2. Stephen Buswell, Olga Caprotti, David P. Carlisle, Michael C. Dewar, Marc Gaetano, and Michael Kohlhase. The OPENMATH Standard, Version 2.0. Technical report, The Open Math Society, 2004.
3. Wendy Chisholm, Gregg Vanderheiden, and Ian Jacobs. Web Content Accessibility Guidelines 1.0. W3C recommendation, World Wide Web Consortium, May 1999.
4. James Clark and Steve DeRose. XML Path Language (XPath) Version 1.0. W3C recommendation, The World Wide Web Consortium, November 1999.
5. The CodeIgniter Forum. Retrieved from http://codeigniter.com/forums on June 1, 2010.
6. Paul Grosso, Eve Maler, Jonathan Marsh, and Norman Walsh. XPointer element() Scheme. W3C recommendation, World Wide Web Consortium, March 2003.
7. JOMDoc — a Java Library for OMDoc documents. Retrieved from http://jomdoc.omdoc.org on May 28, 2010.
8. Michael Kohlhase. OMDOC – *An open markup format for mathematical documents [Version 1.2]*, volume 4180 of *LNAI*. Springer, 2006.
9. Jonathan Marsh, Daniel Veillard, and Norman Walsh. `xml:id` Version 1.0. W3C recommendation, World Wide Web Consortium, September 2005.
10. Microsoft Developer Network. Retrieved from http://msdn.microsoft.com on June 1, 2010.
11. Christine Müller. *Adaptation of Mathematical Documents*. PhD thesis, Jacobs University Bremen, 2010.
12. Florian Rabe. *Representing Logics and Logic Translations*. PhD thesis, Jacobs University Bremen, 2008.
13. Norman Walsh. Topic-oriented authoring (2007, February 5). In *Norm's musings. Make of them what you will.* From http://norman.walsh.name/2007/02/05/painting, seen September 22, 2009.
14. IEEE Learning Technology Standards WG12. IEEE 1484.12.1.2002 Standard for Learning Object Metadata. Retrieved from http://ltsc.ieee.org/wg12/par1484-12-1.html on August 27, 2009.