

# Generation under Space Constraints

**Cécile Paris, Nathalie Colineau,**

**Andrew Lampert**

CSIRO – ICT Centre

Locked Bag 17

North Ryde, NSW 1670, Australia

FirstName.LastName@csiro.au

**Joan Giralt Duran**

Barcelona School of Informatics

Technical University of Catalonia

Barcelona, Spain

joangi@gmail.com

## Abstract

Reasoning about how much to generate when space is limited is a challenge for generation systems. This paper presents two algorithms that exploit the discourse structure to decide which content to drop when there are space restrictions, in the context of producing documents from pre-authored text fragments. We analyse the effectiveness of both algorithms and show that the second is near optimal.

## 1 Introduction

Many organisations employ content management systems to store information, typically at the paragraph level. The use of such systems enables the application of NLG techniques without the cost of acquiring a knowledge base or forming text from first principles. But it brings its own challenge: how to produce a coherent and well structured text satisfying specific space requirements when a system has no control over the text at the sentence level?

The ability to reason about space constraints becomes more pressing as the amount of available information increases and delivery channels become more diverse in terms of space requirements (e.g., web browsers, email, PDAs).

We, as humans, address this problem by shortening our sentences or restricting the content we include. We achieve the former by manipulating vocabulary and syntax. This requires careful attention to the text at sentence level and often does not reclaim significant amount of space.

We achieve the latter by dropping those pieces of content whose contribution to the communicative goal is most limited. This approach can reduce a text's length significantly but requires an understanding of the text's discourse structure.

In our application domain, we answer people's information needs by retrieving content from a repository of pre-authored text fragments and delivering that content via a variety of media (e.g., web, paper, email), each with their own space constraints. In this paper, we show how we exploit the discourse structure to determine what should be realised to best fit some specific space. In particular, we present two algorithms that perform this reasoning and analyse their comparative performance.

## 2 Related Work

NLG systems have exploited the discourse structure for a number of tasks – e.g., to generate appropriate cue phrases (e.g., Scott and de Souza, 1990) or reason about layout (e.g., Bateman et al., 2001). Our system uses the discourse structure to reason about how much to realise to fit a specific space. It produces one discourse tree that is then realised for different delivery channels, each with its own space requirements.

Like other systems (e.g., Moore and Paris, 1993), our system specifies the RST relations (Mann and Thompson, 1988) that hold between text spans during discourse planning. It then exploits the RST principle of nuclearity to decide what to realise. The intuition is that nuclei are important while satellites can be dropped. This intuition has been exploited in some systems to produce summaries (e.g., Sparck-Jones, 1993; Marcu, 1998). Our purpose is different, however. We do not aim to produce a summary but a text that fits into some space requirements,



Figure 1. A brochure generated by our system

potentially only slightly shortening a text. Our task brings new challenges, e.g., filling the space optimally and producing a balanced text.

Our system exploits the notion that some relations are more important than others. O'Donnell (1997) used this principle to produce documents of variable length. In his approach, sentence fragments were manually marked up with RST and the text was manipulated at or below the sentence level. In our work, we cannot manipulate text at sentence level nor manually mark up the documents to be shortened.

### 3 Reasoning about Space Constraints

We focus on applications in which the generated text is delivered through several channels. One such application is SciFly, which produces tailored brochures about our organisation. Given a query from a user (topic(s) of interest), the system consults a staff directory and a repository of text fragments to gather relevant information. The fragments, written by our marketing team, are self contained and comprised of one or two paragraphs. SciFly integrates all the relevant information into a coherent whole (see Figure 1) using the meta-data describing each fragment's content. A text fragment can be used with different rhetorical relations in different brochures. The system produces a 2-page paper brochure, a web output, and a PDF version of the paper brochure is emailed to the user with a summary in

the email body. All outputs are generated from the same discourse structure by our algorithm.

Our need to deliver the brochure via multiple channels led us to design algorithms that reason about the content to be expressed and the space available for each channel. The system follows a two-stage approach: during discourse planning, content and organisation are selected, and a discourse tree is built. The tree includes the top level communicative goal, intermediate goals and the rhetorical relations that exist between text spans, encoding both the purpose of each fragment and how they relate to each other. Then, at the presentation stage, the system reasons about this structure to decide what to realise when there is too much content for some channel.

We implemented and tested two algorithms for this reasoning. Both algorithms embody the principle of nuclearity and exploit the notion that some relations are more important than others. An importance value is assigned to relations based on their contribution to the communicative goal. Table 1 shows our assignments, which are based on judgments from our marketing staff.

To explain the algorithms, we represent the discourse tree using an abstract view, as shown in Figure 2. Each node is a communicative goal. White nodes indicate nuclei. Satellites are shaded in grey corresponding to the importance of the rhetorical relation linking them to the nucleus. The number inside each node is the approximate amount of content that node produces (in lines).

Shading	Discourse Relations	Importance
Black	Illustration, Background, Circumstance, Elaboration	Low
Dark Grey	Context, Motivation, Evidence, Summary, Justification,	Low-Medium
Light Grey	Preparation, Enablement	Medium-High
		High

Table 1. Importance score for some relations<sup>1</sup>

Each node is the root of a subtree (empty if the node is a leaf) which generates some content. In both algorithms, the system computes for each node an *approximation* of the space required for that content in number of lines (an approximation as it depends on style, line-wrapping and other formatting attributes in the final output). This is computed bottom-up in an iterative manner by looking at the retrieved content at each node.

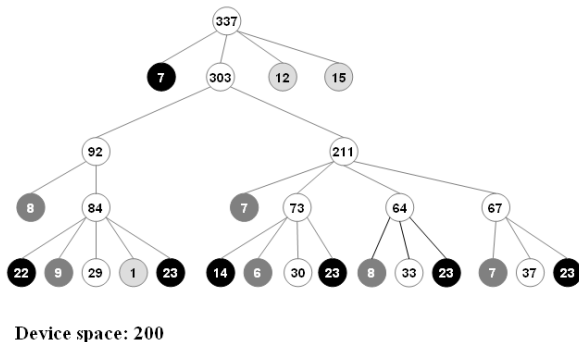


Figure 2. Discourse tree with space annotations

### 3.1 Simple Algorithm

The first algorithm is simple. It checks whether the top level node would result in too much content given the space requirements of the output channel (e.g., lines of content per page). If yes, the system traverses the tree, selects satellites with the lowest importance value and drops them with their sub-trees. The algorithm repeats this process until the total amount of content fits the required space. We deployed the SciFly system with this algorithm at a trade fair in 2005 and 2006 and measured the experience visitors had with the system. On average, people rated the system positively but noted that there was sometimes a lot of blank space in the brochures, when they felt that more information could have been included. This is because our simple algorithm drops many sub-trees at once, thus potentially deleting a lot of content in each step. This led us to our enhanced algorithm.

<sup>1</sup> In our system, we consider 5 levels of importance. We have merged levels here to avoid too many shades of grey.

### 3.2 Enhanced Algorithm

We redesigned the algorithm to take into account the depth of a node in addition to its rhetorical status. We assign each node a *weight*, computed by adding the weight of the node's parent and the *penalty score* of the rhetorical relation, which is (inversely) related to its importance score. Penalty scores range from 1 to 6, in increments of 1: A nucleus has a score of 1 to take the tree depth into account, high importance relations have a score of 2, and low importance relations have a score of 6. In a discourse tree, a child node is always *heavier* than its parent. The larger the weight, the less important the node is to the overall communicative goal. The system orders the nodes by their weight, and the heaviest nodes are dropped first. Thus, nodes deeper in the tree and linked by discourse relations with lower importance get removed first. Nodes are dropped one by one, until the top level node has an amount of content that satisfies the space constraint. This provides finer control over the amount of realised content and avoids the limitation of the first algorithm.

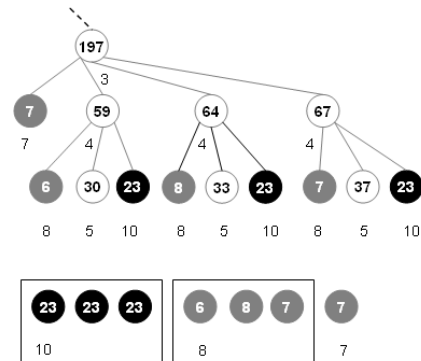


Figure 3. Ordered list of (satellite) cluster nodes

Sometimes, a discourse structure contains parallel sub-structures (e.g., bulleted points) that, if pruned unevenly, result in unbalanced text that seems odd. In such cases, the discourse structure typically contains several sub-trees with the same structure. In SciFly, such parallel structures occur when describing a list of projects. These are generated during discourse planning by a plan containing a *foreach* statement, e.g., (*foreach project in project-list (describe project)*). To address this situation, the system annotates all sub-structures issued from such a *foreach* statement. When constructing the ordered list of satellites, nodes at the same depth in the sub-structures are clustered together, as shown in Figure 3, taking into account their relationship to the nucleus. When dropping

nodes, the whole cluster is deleted together, rather than node by node. So, in Figure 3, the whole cluster of weight 10 is dropped first, then the cluster of weight 8, etc. This prevents one sub-structure from being pruned more than its sibling structures and ensures the resulting brochure is balanced.

## 4 Evaluation

We evaluated the algorithms to assess their comparative effectiveness, based on a test set of 1507 automatically generated brochures about randomly selected topics. We observed that 82.5% of the brochures generated with the enhanced algorithm filled over 96% of the available space (leaving at most 8 lines of empty space), compared to 29% of brochures generated with the simple algorithm. In addition, 96.5% of the brochures generated with the new algorithm filled at least 90% of the space, compared with 44.5% of brochures with the simple algorithm.

We also found that 75% of brochures included more content using the enhanced algorithm (an average of 32 additional lines), but 12% of the brochures contained less content. We examined the latter in detail and found that, for these cases, the difference was on average 4 lines, and that the reduction was due to our treatment of parallel discourse structures, thus representing a desirable loss of content to create balanced brochures.

We also performed a user evaluation to verify that the improvement in space usage had not decreased users' satisfaction. We asked users to compare pairs of brochures (simple algorithm vs. enhanced algorithm), indicating their preference if any. Seventeen users participated in the evaluation and were presented with seven pairs of brochures. To control any order effect, the pairs were randomly presented from user to user, and, in each pair, each brochure was randomly assigned a left-right configuration. Participants mostly preferred the brochures from the enhanced algorithm, or found the brochures equivalent, thus showing that our more effective use of space had not decreased users' satisfaction.

Overall, our results show that our enhanced algorithm is close to optimal in terms of conveying a message appropriately while filling up the space and producing a coherent and balanced text.

## 5 Conclusions

Reasoning about how much to generate when space is limited presents an important challenge

for generation systems. Most systems either control their generation process to avoid producing large amounts of text at the onset, or control the generation at the sentence level. In our application, we cannot resort to any of these approaches as we generate text reusing existing text fragments and need to produce one discourse tree with all the appropriate available content and then select what to realise to output on several delivery channels. To satisfy space constraints, we implemented and tested two algorithms that embody the notions of nuclearity and importance of information to decide which content to keep and which to withhold. Our approach produces documents that fill most of the available space while maintaining users' satisfaction.

## Acknowledgements

We thank M. Raji for her work on the user experiment, the members of our group and K. Vander Linden for their input, and everyone who participated in our evaluation.

## References

- John Bateman, T. Kamps, K. K. Reichenberger, K. and J. Klein. 2001. Constructive text, diagram and layout generation for information presentation: the DArt\_bio system. *Computational Linguistics*, 27 (3): 409–449.
- William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a functional theory of text organisation. *Text* 8(3):243–281.
- Daniel Marcu. 1998. To build text summaries of high quality, nuclearity is not sufficient. In *Working Notes of the AAAI-98 Spring Symposium on Intelligent Text Summarization*, Stanford, CA, 1–8.
- Johanna D. Moore and Cécile L. Paris. 1993. Planning Text for Advisory Dialogues: Capturing Intentional and Rhetorical Information. *Computational Linguistics*, 19 (4):651–694, Cambridge, MA.
- Donia R. Scott and Clarisse S. de Souza. 1990. Getting the message across in RST-based text generation. In Dale, Mellish & Zock (eds). *Current Research in NLG*. London: Academic Press. 119–128.
- Mick O'Donnell (1997). Variable Length On-Line Document Generation. *Proceedings of EWNLG*. Gerhard-Mercator University, Duisburg, Germany.
- Karen Spark Jones (1993). What might be in a summary? *Information Retrieval 93: Von der Modellierung zur Anwendung*. (Ed: Knorz, Krause and Womse-Hacker), Konstanz: Universitätsverlag Konstanz, 9–26.