

# GEA: a Complete, Modular System for Generating Evaluative Arguments

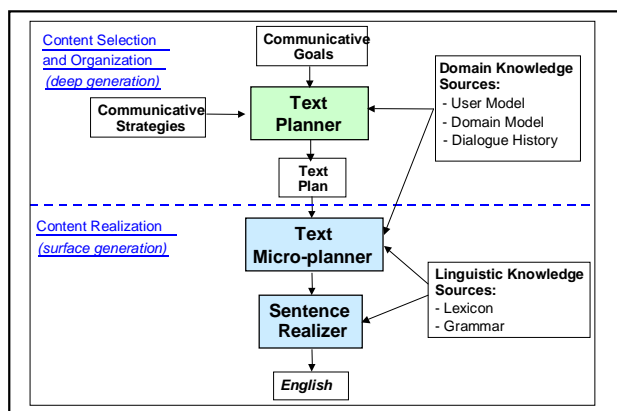
Giuseppe Carenini

Department of Computer Science  
University of British Columbia  
2366 Main Mall ,Vancouver, B.C. Canada V6T 1Z4  
carenini@cs.ubc.ca

**Abstract.** This paper presents a system for generating user tailored evaluative arguments, known as the Generator of Evaluative Arguments (GEA). GEA design is based on a pipelined architecture commonly used in natural language generation. After an overview description of GEA main components, we focus on how GEA performs the microplanning tasks. Details are provided by examining the generation of a sample argument.

## 1 Introduction

Evaluative arguments are communicative acts that attempt to advise or persuade the addressee that something is good (vs. bad) or right (vs. wrong). The ability to generate evaluative arguments is critical in many communicative settings involved in human-computer interaction. For instance, a system that serves as a student advisor may need to justify why a particular course is a good choice for its user. Or, in a different context, a real-estate software assistant may need to argue that one house is a terrible choice for its user. In the field of natural language generation (NLG), considerable research has been devoted to develop computational models for automatically generating user tailored evaluative arguments. Among others, [1-3] have investigated the process of selecting and structuring the argument content, while [4] developed a detailed model of how the selected content should be realised into natural language. However, a key limitation of previous research is that specific projects have tended to focus on only one aspect of the generation process, leaving the development of a comprehensive computational model as future work. In this paper, we present a preliminary attempt to develop such a model. By extending and integrating previous work, we have designed and implemented the Generator of Evaluative Arguments (GEA), a complete NLG system that covers all aspects of the generation process. GEA uses (as much as possible) domain-independent data structures and algorithms and encodes general principles on how evaluative arguments are to be generated. In the remainder of this paper, we first present a standard architecture for a generic NLG system. Then, we describe how the modules and tasks of this architecture have been instantiated in GEA to model the generation of evaluative arguments. After that, we

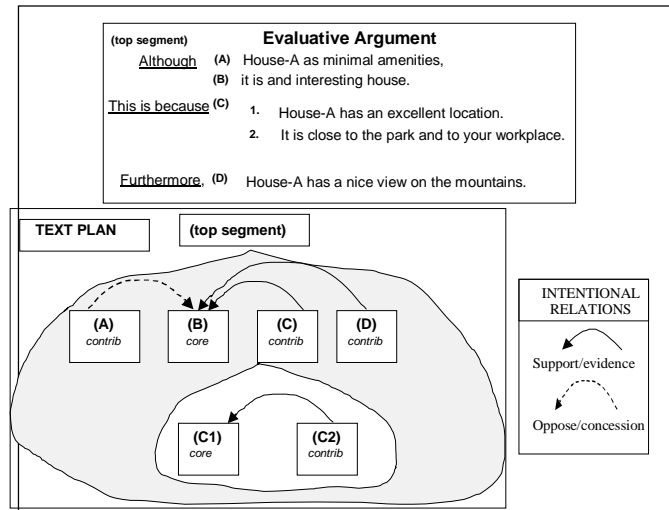


**Figure 1 NLG system pipeline architecture**

focus on the GEA text microplanner module. Details are provided by examining the generation of a sample user tailored evaluative argument in the real-estate domain.

## 2 Standard Architecture for a Generic NLG System

Text generation involves two fundamental tasks: a process that selects and organizes the content of the text (deep generation), and a process that expresses the selected content into natural language (surface generation). Most previous work in NLG makes the assumption that deep generation should strictly precede surface generation. The resulting pipeline architecture, which is adopted in this work, is shown in Figure 1. In this architecture language generation is modeled as a goal-driven communicative process. The initial input of deep generation is a set of communicative goals, typically of the form: make the hearer *believe* something, change the hearer *attitude* about something and make the hearer *intend to do* something. Then, a *Text Planner* selects and organizes content to achieve the communicative goals given as input (see [5]). In performing this task, the text planner applies a set of communicative strategies that specify for each communicative goal how it can be achieved by either posing further communicative sub-goals for the planner, or by performing a primitive communicative action. The application of these strategies typically relies on three domain knowledge sources (see Figure 1): a *domain model*, a *user model* and a *dialogue history*. The domain model is the source from which the content of the text is selected, while the user model and the dialogue history allow the planner to tailor the content and structure of the text to both features of the user and features of previous interaction. For instance, consider the communicative goal of increasing the user positive attitude towards an entity (e.g., a house). The text planner would select information about the house and related entities (e.g., the house's neighborhood) from a model of the real-estate domain. Furthermore, the text planner may select different information depending on the user, because different users might agree on the same evaluation for different reasons depending on their preferences. And finally, as an example of sensi-



**Figure 2** Sample text plan

tivity to the dialog history, the text planner may avoid repeating information that had been already presented to the hearer in a previous interaction.

The output of text planning is a *text plan*, a data structure that specifies: the rhetorical structure of the text, what propositions the text should convey and a partial order between those propositions. The rhetorical structure of the text subdivides the text into segments. Each segment is a portion of text that is intended to achieve a communicative goal. Segments are internally structured and consist of an element that most directly expresses the segment purpose and any number of constituents supporting that purpose.<sup>1</sup> The rhetorical structure also specifies for each supporting element how it relates to the main element, both from an intentional perspective, (i.e., how the supporting element is intended to support the main one), and from an informational perspective, (i.e., how its content relates to that of the main element). The text for a sample evaluative argument and its corresponding text plan are shown in Figure 2. The text plan specifies that the text consists of four subsegments. The third segment is itself composite and consist of two subsegments. Only intentional relations are shown because the informational ones are all obvious properties of the house.

Going back to the pipeline architecture shown in Figure 1, the generation of a text plan ends the process of *deep generation*. The second task involved in text generation, *surface generation*, comprises the two sub-processes of *text microplanning* and *sentence realization*. Notice that, while text planning primarily relies on domain knowledge and sentence realization primarily relies on linguistic knowledge, microplanning tasks consider the interactions between domain knowledge and linguistic knowledge [6]. The following three tasks belong to microplanning: (a) **Lexicalization** is the task

<sup>1</sup> The main element and the supporting elements are called differently in different discourse theories. In this paper we use core vs. contributors respectively. These elements can be composite segments themselves.

of selecting words and associated syntactic structures to express semantic information. Usually, lexicalization also includes the selection of cue phrases (e.g., “although”, “because”, “in fact”), which are words and phrases that mark the relationship between portions of text. Three basic types of lexicalization can be identified (see [6] for more details and examples). In *simple lexicalization*, a proto-phrase template is associated with each possible chunk of semantic information and an information chunk is lexicalized by instantiating its corresponding template. In *simple lexical choice*, several proto-phrase templates are associated with each proposition and its arguments. So, in addition to instantiating a template, we have the problem of selecting the most appropriate one. This selection is typically based on syntactic factors (e.g., the syntactic category in which the proposition has to be expressed), specific features of the particular information chunk and pragmatic factors (e.g., user knowledge and preferences). Finally, in *fine-grained lexicalization*, it is assumed that the chunks of information given as input are expressed in terms of abstract semantic primitives. And this requires additional more sophisticated processing. (b) **Aggregation** is the task of packaging semantic information into sentences. Three basic types of aggregation can be identified [6]. In *simple conjunction*, two or more informational elements are combined within a single sentence by using a connective such as *and*. For instance, two informational elements that could be realized independently as (a1)“House B-11 is far from a shopping area” and (a2)“House B-11 is far public transportation” can be combined and realized as the single sentence “(a1) and (a2)”. In *conjunction via shared participants*, two or more informational elements sharing argument positions with the same content are combined to produce a surface form where the shared content is realized only once. For instance, the two informational elements aggregated above in a *simple conjunction* could be combined in a *conjunction via shared participants* as “House B-11 is far from a shopping area and public transportation”. Finally, in *syntactic embedding*, an informational element that might have been realized as a separate major clause is instead realized as a constituent embedded into some other realized element. For instance, two informational elements that could be realized independently as “House B-11 offers a nice view” and “House B-11 offers a view on the river” can be combined and realized as “House B-11 offers a nice view on the river”. (c) The **generation of referring expressions** is the task of determining the semantic content of the noun phrases used to refer to the domain entities mentioned in the text plan. This task also includes determining when a pronoun is the most effective referring expression (i.e., **pronominalization** decision)<sup>2</sup>.

After microplanning (see Figure 1), the *Sentence Realizer* completes the generation process. It runs the output of the *Micro-Planner* through a computational grammar of English that produces English text. We do not discuss the realization process here, because GEA uses an off the shelf system as sentence realizer.

---

<sup>2</sup> No details on the task of generating referring expression proper are given here, because they are not needed to understand our system. Pronominalization is typically based on text segmentation and the related notion of local coherence. We describe our pronominalization algorithm in Section 4.

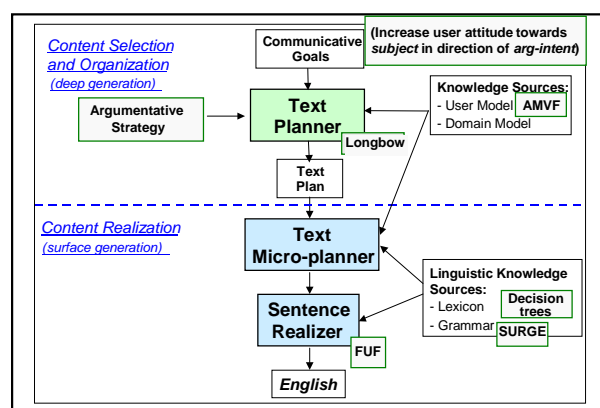


Figure 3 The GEA architecture

### 3 The Generator of Evaluative Arguments (GEA)

The design of GEA is based on principles from argumentation theory as well as on previous work in computational linguistics. GEA covers all aspects of generating user tailored evaluative arguments from selecting and organizing the content of the argument, to expressing the selected content into natural language. In this section, we describe the design and development of GEA by illustrating how its architecture specializes the standard architecture of a generic NLG system presented in the previous section (Figure 1). As shown in Figure 3, the input to the planning process is an abstract evaluative communicative goal expressing that the user attitude toward a subject should increase in the direction of the communicative intent. In GEA, the subject of the evaluation is an entity in the domain of interest (e.g., a house in the real-estate domain), while the argumentative intent is either positive or negative, with positive/negative meaning that the user should like/dislike the entity.

Given an abstract communicative goal, the Longbow text planner [7] selects and arranges the content of the argument by applying a set of communicative strategies that implement an *argumentation strategy* based on guidelines for content selection and organization from argumentation theory (e.g., [8]). The text planner decomposes abstract communicative goals into primitive ones. In parallel, it also decomposes communicative actions that achieve those goals and imposes appropriate ordering constraints among these actions. Two knowledge sources are involved in this process of goal and action decomposition (see Figure 3): (i) A *domain model* representing entities and their relationships in a specific domain. (ii) An additive multiattribute value function (*AMVF*), which is a complex model of the user's preferences [9].

An AMVF is a model of a person's values and preferences with respect to entities in a certain class. It comprises a *value tree* and a set of *component value functions*. A value tree is a decomposition of an entity value into a hierarchy of entity aspects (called objectives in decision theory), in which the leaves correspond to the entity primitive objectives (see left of Figure 4 for a simple value tree in the real estate domain). The arcs in the tree are weighted to represent the importance of an objective

with respect to its siblings (e.g., in Figure 4 *quality* for UserA is more than twice as important as *amenities* in determining the *house-value*). The sum of the weights at each level is always equal to 1. A component value function for a primitive objective expresses the preferability of each value for that objective as a number in the [0,1] interval, with the most preferable value mapped to 1, and the least preferable one to 0. For instance, in Figure 4 the *modern* value of the primitive objective *architectural-style* is the most preferred by UserA, and a *distance-from-park* of 1 mile has preferability  $(1 - (1/3.2 * 1))=0.69$ . Although for lack of space we cannot provide details here, given a user specific AMVF and an entity, GEA can compute precise quantitative measures that are critical in generating a user-tailored evaluative argument for that entity. First, it is possible to compute how valuable an entity is for that user. Second, GEA can compute how valuable any objective of the entity is for that user (see Figure 4 on the right for examples). Third, GEA can identify what objectives can be used as supporting or opposing evidence for the evaluation of their parent objective. Fourth, GEA can compute for each objective the strength of supporting (or opposing) evidence it can provide in determining the evaluation of its parent objective. In this way, our argumentation strategy can arrange evidence according to its strength and can generate concise arguments by only including sufficiently strong evidence. Details of the strategy and on the measure of evidence strength are presented in [10]. The argumentation strategy is implemented as a library of plan operators. Given an abstract evaluative communicative goal, the text planner applies the operator library and produces a text plan for an argument intended to achieve that goal.

Next, the text plan is passed to the GEA microplanner which performs aggregation, lexicalization and generates referring expressions. Aggregation, the packaging of semantic information into sentences, is performed according to the standard techniques summarized in Section 2. With respect to lexicalization, the GEA microplanner selects words to express evaluations by following an extension of previous work on realizing evaluative statements [4], whereas decisions about cue phrases (to express discourse relationships among text segments) are implemented as a decision tree based on features suggested in the literature (e.g., [11], [12]). The generation of referring expression in GEA is straightforward; an entity is always referred to by its proper noun. For pronominalization (deciding whether to use a pronoun or not to refer to an entity), simple rules based on centering theory [13] are applied. Finally, the output of text microplanning is unified by the GEA sentence realizer (FUF) with the Systemic Unification Realization Grammar of English (SURGE) [14].

## 4 The Generation of a Sample Argument

GEA is a complex computer application that integrates and extends several systems and formalisms. For illustration, in this section, we examine the generation of the sample evaluative argument shown in Figure 6. The argument is about a particular house for a particular user. Information about the house along with an AMVF preference model for the sample user are shown in Figure 4. For lack of space, we mainly focus here on the key tasks involved in how the GEA's microplanner processes the

text plan for the argument. Details on GEA's argumentation strategy for content selection and organization can be found in [10].

The generation of the argument is initiated by posing the communicative goal (*increased-attitude User-A House-2-33 +*) for the text planner to achieve. By applying the operator library implementing the argumentation strategy, the text planner selects and organizes the content of the argument. This process relies on the user's preference model and on the information about the house (shown in Figure 4). The selected content (i.e., a subset of the AMVF's objectives and their value for UserA) is organized in a text plan. As described in Section 2, a text plan specifies the rhetorical structure of the text, what propositions the argument should convey and a partial order between those propositions. Figure 5 shows the text plan generated by GEA for our example. The action decompositions and the relation of *evidence* and *concession* between them express the rhetorical structure. The leaves of the text plan express the propositions the argument should convey (e.g., *<Assert that the Quality of House-2-33 is for UserA 0.82>*). And the nodes of the text plan (i.e., the communicative actions) are ordered (e.g., the action *Assert-opposing-props* should be performed before *Assert-props-in-favor*). Notice that the text plan does not include the objective *Crime* (which is included in the argument). The reason is that the current implementation of the argumentation strategy only processes objectives of depth  $<3$  in the AMVF. The objective *Crime* is reintroduced in the argument by subsequent processing in an ad hoc fashion.

Once the process of content selection is completed, the text plan is passed to the GEA microplanner which performs the following tasks.

**Lexicalization proper** - The GEA microplanner performs *simple lexical choice* (see Section 2). It selects for each proposition in the text plan the most appropriate proto-phrase to express that proposition. First, the selection is based on the objective of the proposition and then on its value for the current user. For instance, in our sample argument, the proposition (*Location House-2-33 0.6*) according to the portion of the decision tree shown in Figure 7, is mapped to a proto-phrase which (with pronominalization) is realized as "*it has a reasonable location*", while the proposition (*Distance-shopping House-2-33 0.84*), according to the portion of the decision tree for the objective *Distance-shopping* (not detailed in the figure), is mapped to a proto-phrase which is realized as "*it offers easy access to the shops*". For lack of precise indications from linguistic theory, the numerical intervals that determine the final decisions are simply based on reasonable estimates.

**Aggregation** - In general, GEA performs both types of structural aggregation described in Section 2 (i.e., aggregation *via shared participants* and *by syntactic embedding*). To ensure argument coherence, aggregation is only attempted between objectives that are related to a claim by the same rhetorical relation. In our example, we have only one aggregation between the *Location* and *Neighborhood* objectives. The two propositions are aggregated by *syntactic embedding*. The aggregation strategy treats aggregation between *Location* and *Neighborhood* as a special case, because it combines two propositions that are not at the same level in the text plan (the evaluation of the neighborhood is evidence for the evaluation of the location, which in turn is evidence for the value of the house - see plan in Figure 5).

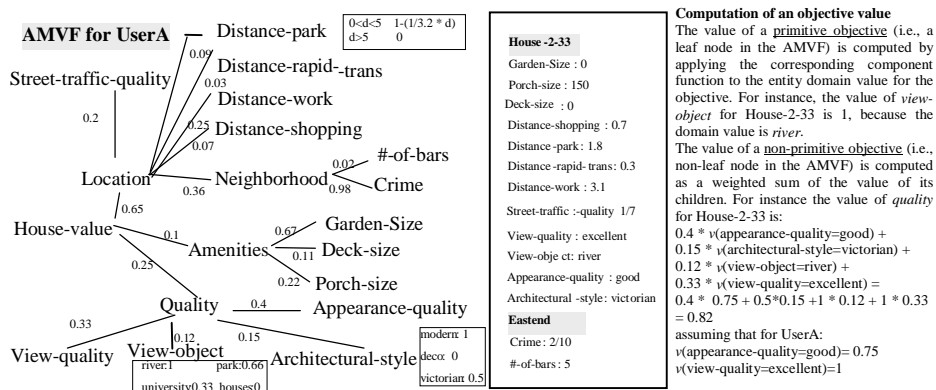


Figure 4 Preference model for UserA<sup>3</sup>, information about House2-33 and value computation

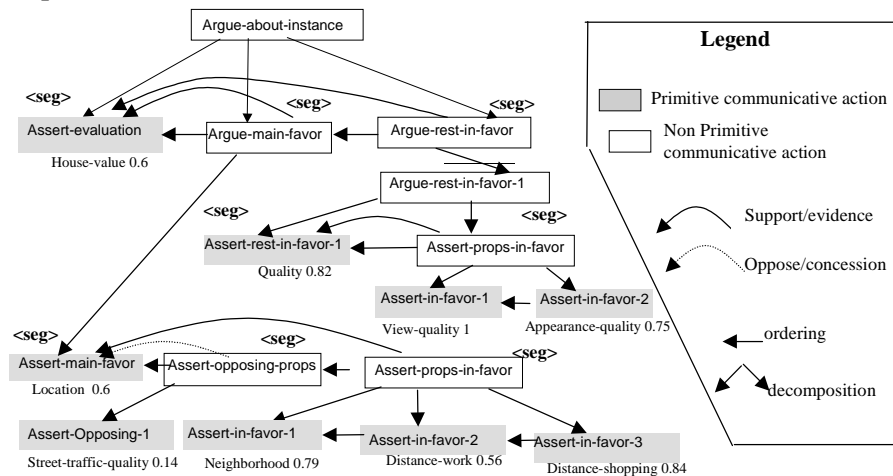


Figure 5 Text plan and segmentation structure

```

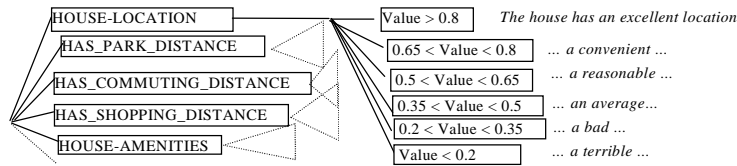
<<House 2-33 is an interesting house.>> b1<<In fact, it has a reasonable location in the safe
Eastend neighborhood.>> b2<<Even though the traffic is intense on 2nd street, >> b3<house 2-
33 is reasonably close to work. And also it offers an easy access to the shops.>>
b4<<Furthermore, the quality of house 2-33 is good.>> b5<House 2-33 offers an excellent
view. And also it looks beautiful.>>>
    
```

Figure 6 Evaluative argument about House-2-33, tailored to UserA

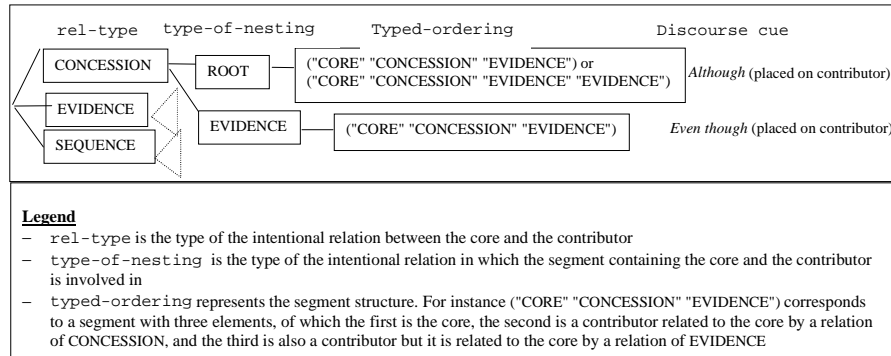
**Decision about cue phrases** are implemented as a decision tree taking into account relevant features suggested in the literature: (a) the intentional relationship between the core and the contributor (b) the whole segment structure in which core and contributor appear (with core and contributor positions within the segment), and (c) the

<sup>3</sup> For illustration, only three component value functions are shown.





**Figure 7** Decision tree for simple lexical choice in the real-estate domain



**Figure 8** Portion of decision tree for discourse cue selection

relationship in which the core and contributor segment itself is involved. For illustration, if the reader applies the portion of the decision tree shown in Figure 8 to the text plan for our example, s/he can verify why “Even though” was used to mark the only concession in our sample argument. For **pronominalization** we have devised simple rules based on centering theory (a theory of local coherence in discourse, see [13]): “In a discourse segment, successive references to the entity evaluated by the argument are realized as pronouns. In contrast, at the beginning of a new segment, the entity is referred to by a pronoun only if two conditions hold. The segment boundary is explicitly marked by a discourse cue and a pronoun has not been used to refer to that entity in the previous sentence”. Obviously, applying these rules requires a segmentation of the text given as input. As described in Section 2, the text plan expresses text segmentation: any core or contributor of an intentional rhetorical relation corresponds to a segment. If we apply this definition to our example, we obtain the segment structure shown in Figure 5 on the text plan and in Figure 6 on the corresponding text. The text contains five segment boundaries ( $b_1, \dots, b_5$  in Figure 6). For illustration, the pronominalization rule is applied to the segment boundary  $b_1$  as follows:  $b_1$  is explicitly marked by the “in fact” discourse cue and a pronoun has not been used in the sentence preceding  $b_1$  to refer to `House2-23`. Thus, a pronoun is used to refer to that entity in the following sentence.

## 5 Conclusions and Future Work

GEA is a fully-implemented, complete and modular NLG system for generating user tailored evaluative arguments. GEA implementation is mostly domain independent.

The system can be easily ported to new domains by simply specifying AMVF models for relevant entities and corresponding decision trees for lexicalization proper. We plan to extend GEA's coverage in at least two ways. First, we intend to enable GEA to generate more complex evaluative arguments (e.g., comparisons between entities). Secondly, we plan to apply GEA to larger AMVFs (i.e., depth  $\Rightarrow$  3). We expect additional techniques to be necessary to generate coherent text for these larger models. Finally, with respect to evaluation, we plan to continue testing GEA following the methodology described in [15] and successfully applied in [16].

## References

1. Morik, K., *User Models and Conversational Settings: Modeling the User's Wants*, in *User Models in Dialog Systems*, A.Kobsa and W.Wahlster, Eds. 1989, Springer-Verlag. p. 364-385.
2. Elzer, S., J. Chu-Carroll, and S. Carberry. *Recognizing and Utilizing User Preferences in Collaborative Consultation Dialogues*. in *Proceedings of Fourth International Conference of User Modeling*. 1994. Hyannis, MA.
3. Ardissono, L. and A. Goy. *Tailoring the Interaction with Users in Electronic Shops*. in *Proc. 7th Conference on User Modeling*. 1999. Banff, Canada: Springer-Verlag.
4. Elhadad, M., *Using argumentation in text generation*. *Journal of Pragmatics*, 1995. **24**: p. 189-220.
5. Moore, J.D., *Participating in Explanatory Dialogues: Interpreting and Responding to Questions in Context*. 1995, Cambridge, MA: MIT Press.
6. Reiter, E. and R. Dale, *Building Natural Language Generation Systems*. *Studies in Natural Language Processing*. 2000: Cambridge University Press.
7. Young, R.M. and J.D. Moore. *DPOCL: A Principled Approach to Discourse Planning*. in *Proceedings of the 7th Int. Workshop on Text Generation*. 1994. Montreal, Canada.
8. Mayberry, K.J. and R.E. Golden, *For Argument's Sake: A Guide to Writing Effective Arguments*. 2nd ed. 1996: Harper Collins, College Publisher.
9. Clemen, R.T., *Making Hard Decisions: an introduction to decision analysis*. 1996, Belmont, California: Duxbury Press.
10. Carenini, G. and J. Moore. *A Strategy for Generating Evaluative Arguments*. in *International Conference on Natural Language Generation*. 2000. Mitzpe Ramon, Israel. p.47-54
11. Knott, A. and R. Dale, *Choosing a set of coherence relations for text generation: a data-driven approach*, in *Trends in Natural Language Generation: an Artificial Intelligence Perspective* ( G. Adorni & M. Zock, eds. ). 1996, Springer-Verlag: Berlin. p. 47-67.
12. Eugenio, B.D., J. Moore, and M. Paolucci. *Learning Features that Predicts Cue Usage*. in *ACL97*. 1997. Madrid, Spain.
13. Grosz, B.J., A.K. Joshi, and S. Weinstein, *Centering: A Framework for Modelling the Local Coherence of Discourse*. *Computational Linguistics*, 1995. **21**(2): p. 203-226.
14. Elhadad, M. and J. Robin, *An overview of SURGE: a reusable comprehensive syntactic realization component*, . 1996, Dept of Math and CS, Ben Gurion Univ., Israel.
15. Carenini, G. *A Task-based Framework to Evaluate Evaluative Arguments*. in *International Conference on Natural Language Generation*. 2000. Mitzpe Ramon, Israel. p. 9-16
16. Carenini, G. and J. Moore. *An Empirical Study of the Influence of Argument Conciseness on Argument Effectiveness*. in *Annual Meeting of the Association for Computational Linguistics (ACL)*. 2000. Hong Kong, China. p. 150-157