Generating Varied Narrative Probability Exercises

Mariët Theune[1] **Roan Boer Rookhuiszen**[1] **Rieks op den Akker**[1] **Hanneke Geerlings**[2]
Department of Computer Science[1]
Department of Research Methodology, Measurement and Data Analysis[2]
University of Twente
Enschede, The Netherlands
m.theune@utwente.nl, a.r.boerrookhuiszen@alumnus.utwente.nl,
h.j.a.opdenakker@utwente.nl, h.geerlings@gw.utwente.nl

Abstract

This paper presents Genpex, a system for automatic generation of narrative probability exercises. Generation of exercises in Genpex is done in two steps. First, the system creates a specification of a solvable probability problem, based on input from the user (a researcher or test developer) who selects a specific question type and a narrative context for the problem. Then, a text expressing the probability problem is generated. The user can tune the generated text by setting the values of some linguistic variation parameters. By varying the mathematical content of the exercise, its narrative context and the linguistic parameter settings, many different exercises can be produced. Here we focus on the natural language generation part of Genpex. After describing how the system works, we briefly present our first evaluation results, and discuss some aspects requiring further investigation.

1 Introduction

Narrative exercises (also called word problems or story problems) are mathematical exercises embedded in a story or text. They are commonly used as test items, to assess or train a student's understanding of the underlying mathematical concepts. When solving a narrative exercise, the student is required to derive the underlying mathematical question from the story and to calculate the correct answer to this mathematical problem.

This paper presents Genpex, a system for generating narrative exercises expressing probability problems. Genpex was created in the context of an international project on item generation for testing student competencies in solving probability problems. Automatic item generation is an effective way of constructing many items with controlled difficulties, based on a set of predefined task parameters (Enright et al., 2002; Deane and Sheehan, 2003; Arendasy et al., 2006; Holling et al., 2009). The goal of our item generation project is to develop a model to support optimal problem and test construction. A large collection of narrative exercises is needed to test the developed models in field trials. All of these narrative exercises should be different, but the properties that define the difficulty of the exercise should be known. Genpex was designed to enable easy creation of new exercises meeting these requirements.

Figure 1 shows a narrative probability exercise generated by Genpex. The text of the exercise is in German, because the target group of our project are German high school students. The texts produced by Genpex are based on a set of example narrative exercises that were created earlier within the project (Zeuch, In preparation).

A property that sets Genpex apart from other narrative exercise generation systems is that it was specifically designed to support variation in the generated exercises. Unlike other systems, it not only changes the context of the narrative exercise (e.g., instead of bikes, the example exercise could also have been about hotel rooms with different properties) but it also varies the way the texts are formulated. Most existing systems for narrative exercise generation use fixed sentence templates to express mathematical content, which means that the same content is always expressed in the same way (Fa-

Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications, pages 20–29,
Portland, Oregon, 24 June 2011.

In einer großen Halle ist eine Mischung von Fahrrädern. Es gibt insgesamt 100 Fahrräder.	In a big hall there are a variety of bicycles. There are 100 bicycles in total.
Es gibt 30 grüne Fahrräder und es gibt 70 weiße. 40 Fahrräder sind Mountainbikes, 50 sind Rennräder und es gibt 10 Hollandräder. 70 Fahrräder sind billiger als 500 Euro und 30 Fahrräder teurer als 500 Euro. 41 Fahrräder sind billiger als 500 Euro und sind Rennräder.	There are 30 green bicycles and there are 70 white ones. 40 bicycles are mountain bikes, 50 are road bikes, and there are 10 Dutch bikes. 70 bicycles are less expensive than 500 Euros and 30 bicycles more expensive than 500 Euros. 41 bicycles are less expensive than 500 Euros and are road bikes.
Fahrradtyp und Preis sind abhängig voneinander und alle anderen Merkmale sind unabhängig voneinander.	Bicycle type and price are dependent on each other and all other properties are independent of each other.
Wie groß ist die Wahrscheinlichkeit, dass ein Fahrrad nicht sowohl ein Mountainrad als auch grün ist?	What is the probability that a bicycle is not both a mountain bike and green?
Wie groß ist die Wahrscheinlichkeit, dass ein Fahrrad entweder billiger als 500 Euro oder ein Rennrad ist?	What is the probability that a bicycle is either cheaper than 500 Euros or a road bike?

Figure 1: The text of an exercise generated by Genpex. (Left: German original, right: English translation.)

iron and Williamson, 2002; Arendasy et al., 2006; Holling et al., 2009). A system that uses a linguistically sophisticated approach, thus in principle allowing for similar text variations as Genpex, is ModelCreator (Deane and Sheehan, 2003; Higgins et al., 2005). However, this system focuses on semantic factors influencing the expression of events with different participants (e.g., different types of vehicles) rather than on generating linguistic variations.

Below, we first describe how a probability problem is constructed by Genpex, based on input by the user. Then we explain in some detail how the natural language generation (NLG) module of Genpex creates a text expressing the probability problem, focusing on the creation of variation in the generated texts. We end with a brief discussion of our first evaluation results and some pointers to future work.

2 Probability Problems

Figure 2 presents the probability problem underlying the narrative exercise of Figure 1. It specifies the context, the total number of entities (*numEntities*), and the distribution of (combinations of) attribute values over the entities. Number information may be suppressed so as not to give the answer away; this is done by inserting a question mark in the place of the number (e.g., *colour[green] = ?*). Explicitly listing such 'hidden' information in the probability problem ensures that all possible values of each attribute are mentioned in the text of the exercise. A basic assumption in creating the probability problems is that all entities have exactly one value for each attribute. For example, all bikes must have some colour, and they cannot have two colours at the same time.

In addition to the number statements, the probability problem also lists which pairs of attributes are dependent on each other. In the example, these are *type* and *price*. This means that if we look at the subset of bikes of a specific type, the probability that one of these bikes has a certain price is not the same as when we look at the entire collection of bikes (and vice versa). If a pair of attributes is not specified as being dependent, it is independent.

Q delineates the question part of the probability problem; we refer to the other parts (except *Context*) as 'statements'. All questions require the calculation of a probability. A question of the form *Q: P(A)* asks for the probability of **event** A, which can be described as "Someone randomly draws one entity out of a (sub)set of entities and this entity has property A". For example, the question could be to calculate the probability that a bike is black if we randomly pick one bike from the set of all bikes. We equate the probability of event A with the relative frequency of the set A of objects that satisfy property A, computed as $|A|/|U|$, where U is the set of all entities (that is, $|U|$ = *numEntities*). In general, the set we draw from is the entire set of entities, but this set can be limited by a conditional statement: the event $A|B$ can be described as "Someone randomly draws one entity with property A from a subset of entities that have property B". In this case, the

Context: bikes

numEntities: 100

colour[green] = 30
colour[white] = 70
type[mountainbike] = 40
type[sportsbike] = 50
type[hollandbike] = 10
price[<500] = 70
price[>500] = 30
price[<500] ∧ type[sportsbike] = 41

dependentAttributes: price & type

Q: P(¬(type[mountainbike] ∧ colour[green]))
Q: P(price[<500] ∨ type[sportsbike])

Figure 2: The probability problem underlying Figure 1.

probability $P(A|B)$ is computed as $|A \cap B|/|B|$. All events involve a single draw of exactly one entity.

Probability problems such as the one in Figure 2 are automatically created by Genpex; the only thing the user has to do is to select one or more question types (defining the difficulty of the exercise) and a context for the exercise. All available question types are of the form *P(A)* or *P(A|B)*, where A (but not B) can be a complex event, i.e., involving a conjunction or disjunction of properties. For example, *Q: P(A ∧ B)* asks for the probability that an entity has both property A and property B. Moreover, parts of a question can be negated.

Currently, Genpex can handle 25 different question types. Some restrictions we put on the available questions are the following. Each question involves at most two different attributes, to avoid complex dependencies. There are no recursive questions (e.g., double negations) and no conditional questions about independent attributes. Finally, we exclude questions that are likely to result in ambiguous language. For example, if we try to express the question *Q: P(¬(colour[white]) ∧ type[sportsbike])* in English, it will be something like "What is the probability that a bike is not white and a road bike?". Due to scope ambiguity of the negation, this sentence may be misinterpreted as "What is the probability that a bike is not white and also *not* a road bike?". The same ambiguity is found in the German sentence expressing this question.[1] Excluding these types of questions does not simplify the task for Genpex; the excluded questions are not more difficult to generate than the included ones. The main reason to exclude certain question types was to avoid creating exercises that might be unclear to the reader.

In addition to selecting one or more question types as input for Genpex, the user also selects a context for the exercise. As a resource, Genpex uses a repository of context files[2] with information concerning the entities that the exercise should be about ('bikes' in our example) and the properties they may have. Each attribute in the context file is linked to a lexical lemma for the word that expresses its relation to the entity (e.g., bikes *are* of a certain colour or type but *have* a certain price). Similarly, for each attribute, a list of possible attribute values and the words expressing them is provided. For example, the type attribute in the bikes context can have the values 'mountainbike', 'sportsbike', 'hollandbike' and 'seniorbike', respectively associated with the words "Mountainbike" (mountain bike), "Rennrad" (road bike), "Hollandrad" (Dutch bike) and "Seniorenrad" (senior bike). Other NLG-related information in the context files is discussed in Section 3. The context file also specifies world knowledge such as the range of *numEntities* (a context about rooms in a hotel will involve fewer entities than a context about books in a bookshop) and possible dependencies between attributes (in the bikes context, price is more likely to be dependent on type than on colour).

Taking the selected question type(s) and context as input, Genpex automatically constructs a probability problem. This involves selecting a number of attributes and values, depending on the question or questions that need to be answered, and creating a correct and complete **world**: an internal representation of the situation in which all entities are fully defined (all their properties are known), and there are no inconsistencies. A part of this world is reflected in the statements of the probability problem. Currently, all statements provide information that is

[1] Genpex does include a re-ordered, mathematically equivalent version of the same question: *Q: P(type[sportsbike] ∧ ¬(colour[white]))*. Because the generated questions follow the order of the attributes in the question specification, this version can be expressed without ambiguity as "What is the probability that a bike is a road bike and not white?"

[2] In the current Genpex prototype, five different contexts are available. The system comes with an editor for the creation of new context files.

required to solve the exercise; redundant information is not included. If the user manually edits the generated probability problem, Genpex reconstructs the world, and tries to solve the exercise using the information in the edited problem. The user is warned in case of inconsistencies or missing information. A warning is also issued if the edited problem contains properties for which no lexical information is available. See Boer Rookhuiszen (2011) for more details on how probability problems are constructed.

3 Language Generation

The NLG process of Genpex has two goals: generating a correct textual representation of a given probability problem, and enabling variation, so that multiple runs will result in different texts. The generated texts should be in grammatically correct German, and they must be unambiguous: the formulation of the text should not leave the reader uncertain about the underlying mathematical exercise.

An overview of the NLG component of Genpex is given in Figure 3. Its architecture reflects the language generation pipeline of Reiter and Dale (2000), with three modules: Document Planner, Microplanner and Surface Realizer. Information between the modules is exchanged in the form of a list of **sentence trees**, each defining the content and grammatical structure of a sentence. The Document Planner creates basic sentence trees. These are manipulated by the Microplanner to create variations. The microplanning stage can in principle be skipped, but that will result in very monotonous texts. Finally, the Surface Realizer applies the correct morphology to the sentence trees and creates the layout of the text. Below, we discuss each module in turn.

3.1 Document Planning: Creating Basic Sentence Structures

The input of the Document Planner is a probability problem, which defines the content and the structure of the narrative exercise. The output is a document plan: a structured list of sentence trees expressing the statements and questions in the probability problem. The document plan also includes an introduction: a simple 'canned' text specified in the context file. If multiple introduction texts are available, one is randomly selected.

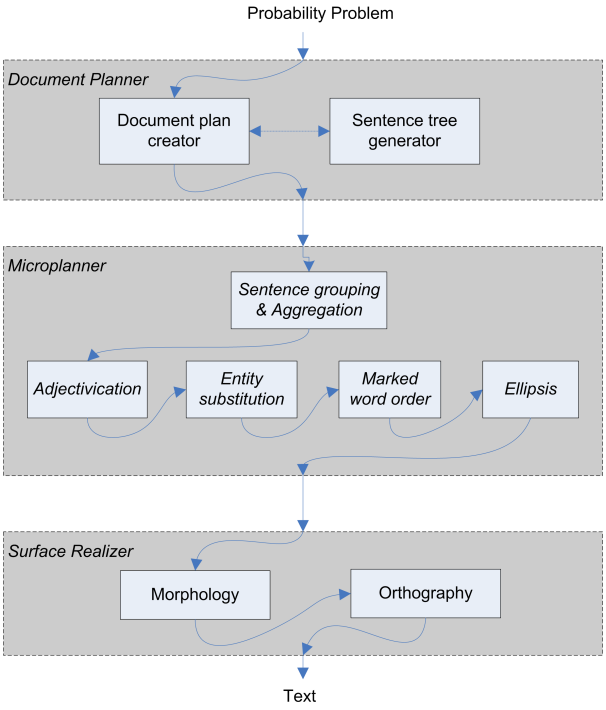

Figure 3: The NLG module of Genpex.

The sentences included in the document plan are all very simple, with the same basic structure. Take for example the statement *colour[white] = 70*. The Document Planner first creates a subject NP expressing the number of entities involved, e.g., "70 Fahrräder" (70 bicycles). Then it creates a VP expressing the relation and the attribute value, e.g., "sind weiß" (are white). The relevant words and their parts of speech are looked up in the context file. For the example statement, this process results in the following basic tree, shown in a simplified notation. Note that the words in the tree have not yet been inflected.

```
[s]
  [np grammaticalRole=su]
     [det grammaticalRole=num]70[/det]
     [noun grammaticalRole=hd]Fahrrad[/noun]
  [/np]
  [vp]
     [verb grammaticalRole=hd]sind[/verb]
     [adj grammaticalRole=predc]weiss[/adj]
  [/vp]
[/s]
```

All sentence trees for questions start with the phrase "Wie groß ist die Wahrscheinlichkeit dass" (What is the probability that), included as canned text in a tree node with syntactic category 'clause'.

This main clause is followed by an indefinite NP referring to the type of entities discussed in the exercise, e.g., "ein Fahrrad" (a bicycle). The structure of the rest of the sentence tree depends on the question type. Sentence tree templates are available for all possible question types. They can be used recursively: slots in the templates can be filled with an expression for an attribute value, or with one of the other templates.

Figure 4 shows the construction of a sentence tree for a fairly complex question of type *P(A ∨ B | ¬ C)*, using multiple question templates. For questions about conditional probabilities Genpex uses the slightly formal "vorausgesetzt" (given that), because simpler phrasings are likely to be ambiguous. For example, assume we want to ask the question *Q: P(type[mountainbike] | colour[green])*. A simple way to ask this question would be "Wie groß ist die Wahrscheinlichkeit dass ein grünes Fahrrad ein Mountainbike ist?" (What is the probability that a green bike is a mountain bike?). However, such a question could be mistakenly interpreted as asking for a joint probability: *Q: P(colour[green] ∧ type[mountainbike])*. For this reason, the more complex formulation is preferred.

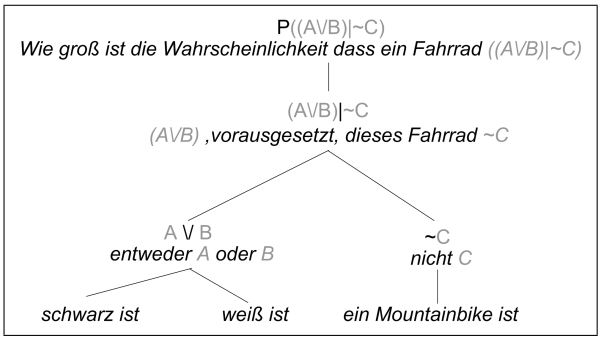

Figure 4: Construction of a question combining multiple templates. Translation, with brackets marking the template boundaries: "What is the probability that a bicycle [[is either black or white] given that this bicycle [is not a mountainbike]]?"

3.2 Microplanning: Creating Variation

The Microplanner modifies the sentence trees produced by the Document Planner by applying a number of **variation techniques**. These techniques place specific requirements on the sentences to which they can be applied, and therefore not every technique can be applied to all sentence trees.

When introducing variation in the narrative form of the exercise, it is important that variations of the same exercise should all have the same meaning and approximately the same difficulty. According to Deane and Sheehan (2003), it is possible to change the wording of a text without changing its difficulty. Reiter and Dale (2000) state that for example aggregating multiple sentences does not change the information they express, but improves the readability and fluency of the text. This is what we want to achieve: adding variation to the text without affecting its interpretation. Genpex therefore uses aggregation as well as a number of text variation techniques, assuming that they do not influence the meaning or difficulty of an exercise.

Below we discuss the operations applied to basic sentence trees in the Microplanner. They are only applied to sentences expressing statements, even though it would be practically possible to apply some of the variations to the questions too. Given that understanding the question is crucial for solving the exercise, and that varying the way the questions are asked might cause confusion, we chose to adhere to a fixed format for the questions, cf. Fairon and Williams (2002).

Aggregation. As a first step, the Microplanner applies aggregation: grouping multiple simple sentences and combining them into one complex sentence. This process leaves the original order of the sentences in the Document Plan intact. Sentences referring to different attributes are never grouped together, to avoid possible misinterpretations. For example, a complex sentence such as "70 bicycles are white and 40 bicycles are mountain bikes" might suggest that the 40 mountain bikes are different entities than the 70 white bikes, excluding the possibility of white mountain bikes. Since this is not the intended meaning, we avoid creating this kind of complex sentences. Sentences referring to the same attribute can be grouped together without risk, because there can never be any overlap between the sets of entities mentioned in these sentences (an entity cannot have multiple values for the same attribute).

Aggregation is performed on a maximum of three sentences to prevent the generation of overly large conjunctions. Groups of four basic sentences are ag-

gregated into two new complex sentences. This way we avoid creating unbalanced texts like example 1 below, preferring to generate sentences that are similar in both length and complexity, as in example 2.

1. 42 Fahrräder sind Mountainbikes, 168 Fahrräder sind Rennräder und 200 Fahrräder sind Hollandräder. 10 Fahrräder sind Seniorenräder.

   (42 bicycles are mountain bikes, 168 bicycles are road bikes, and 200 bicycles are Dutch bikes. 10 bicycles are senior bikes.)

2. 42 Fahrräder sind Mountainbikes und 168 Fahrräder sind Rennräder. 200 Fahrräder sind Hollandräder und 10 Fahrräder sind Seniorenräder.

   (42 bicycles are mountain bikes and 168 bicycles are road bikes. 200 bicycles are Dutch bikes and 10 bicycles are senior bikes.)

Aggregation in Genpex is not optional; it is always applied under the assumption that this will make the generated texts more coherent and pleasant to read. Moreover, it enables variation through ellipsis, as discussed later in this section. Variations in aggregation can be achieved by manually reordering the statements in the probability problem. This will lead to a different Document Plan and as a consequence, to different aggregations, within the restrictions stated above.

Adjectivication. The text variation technique we call 'adjectivication' changes the position and grammatical role of the adjective (if any) expressing the attribute value in a sentence. In basic sentence trees, attribute values expressed by adjectives are included as predicative complements in the VP. If we apply adjectivication to a sentence, the adjective is instead added as a modifier to the subject NP, and the original verb is removed. To make the sentence tree complete again, the words "Es gibt" (There are') are added in front. For example, the sentence "30 Fahrräder sind grün" (30 bicycles are green) will be changed to "Es gibt 30 grüne Fahrräder" (There are 30 green bikes). In German, adjectivication may cause the inflection of the adjective to change, because it gets a different grammatical role: when used as a modifier its inflection reflects the gender and case of the noun it modifies. This is taken care of by the Surface Realizer.

Entity substitution. In case an attribute value is expressed as a noun, e.g., "Rennrad" (road bike) the text variation technique we call 'entity substitution' can be applied. It involves replacing the noun that represents the entity in a basic sentence with the noun that represents the attribute value. As with adjectivication, the original verb is removed and instead "Es gibt" (There are) is added to the sentence. For example, entity substitution changes the basic sentence "50 Fahrräder sind Rennräder" (50 bicycles are road bikes) to "Es gibt 50 Rennräder" (There are 50 road bikes).

Marked word order. Another source of variation is topicalizing the phrase expressing the attribute value by moving it to the front of the sentence. Applying this variation technique changes the basic sentence "30 Fahrräder sind teurer als 500 Euro" (30 bicycles are more expensive than 500 Euros) to "Teurer als 500 Euro sind 30 Fahrräder" (More expensive than 500 Euros are 30 bicycles). Since using such a marked word order may come across as unnatural in a neutral discourse context, this type of variation should be applied with caution.

Ellipsis. This is the removal of duplicate words from sentences, which typically applies to aggregated sentences (Harbusch and Kempen, 2009). Genpex can apply different types of ellipsis, such as Gapping and Conjunction Reduction. Gapping is the removal of all except the first verb in an aggregated sentence. An example from Figure 1 is the sentence "70 Fahrräder sind billiger als 500 Euro und 30 Fahrräder teurer als 500 Euro" (70 bicycles are less expensive than 500 Euros and 30 bicycles more expensive than 500 Euros), where the verb "sind" (are) has been deleted from the second clause. (Forward) Conjunction Reduction deletes the subject of subsequent clauses if it is identical to the subject of the first clause. The following sentence is an example: "40 Fahrräder sind Mountainbikes und 50 sind Rennräder" (40 bicycles are mountain bikes and 50 are road bikes). It is possible to combine Gapping and Conjunction Reduction, e.g., "40 Fahrräder sind Mountainbikes und 50 Rennräder" (40 bicycles are mountain bikes and 50 road bikes).

Ellipsis is also possible in sentences with marked word order. For example, "Grün sind 30 Fahrräder und weiß sind 40 Fahrräder"(Green are 30 bicycles and white are 40 bicycles) could be reduced to "Grün sind 30 Fahrräder und weiß sind 40" (Green

are 30 bicycles and white are 40). However, in this sentence, the verb cannot be removed from the last clause. Genpex currently allows aggregated sentences in which some of the clauses have marked word order. In these cases, ellipsis is not applied, because it will most likely result in grammatically incorrect sentences. For example, in the sentence "30 Fahrräder sind grün und Weiß sind 40 Fahrräder" (30 bicycles are green and white are 40 bicycles) the system will not apply ellipsis.

For every sentence in the document plan, the system will check which of the variation techniques described above can be applied to it, by analyzing the structure of the sentence tree. If a technique is in principle applicable, the probability of it being actually applied depends on information in the context file, and on parameters set by the user through the GUI of Genpex. For every attribute in the context file, the author of the file can prevent Genpex from applying a specific technique by giving it a probability of 0, if it is never suitable in the case of this specific attribute. For example, applying marked word order to a sentence expressing the 'type' attribute in the bikes context would lead to odd sentences such as "Mountainbikes sind 40 Fahrräder" (Mountainbikes are 40 bicycles). Though grammatically correct, such sentences would be hard to interpret and therefore are best avoided.

During generation, the user can directly influence the probability that certain variations are applied through sliders in the GUI; see Figure 5. The probability holds for every sentence that satisfies the structural requirements of the variation technique in question, unless the technique is excluded based on the information in the context file, as explained above. After having set the variation probabilities, the user can click 'Update Text' to see the effect. The user can also choose to have the text automatically updated every time a slider is moved.

When the user saves a generated exercise, information about the variation techniques that have been applied is logged and saved together with the exercise. If further research shows that a certain variation technique has an unintended influence on exercise difficulty, it will be easy to exclude this technique from the creation of new exercises by setting its probability to 0 in the GUI.

3.3 Surface Realisation: the Final Polish

The main task of the Surface Realizer is to convert the sentence trees that have been manipulated by the Microplanner to actual text, applying correct morphology and orthography.

Information about German morphology is retrieved from a lexicon listing the possible word forms of each lemma in the context files. German has a rich inflectional system compared to English, with suffixes reflecting the gender, number and case of determiners, adjectives and nouns. Gender can be masculine, feminine or neuter, number is singular or plural, and case is nominative, accusative, dative or genitive. In the type of exercises currently generated by Genpex, all words are in nominative case. Number information for nouns and verbs is given in the sentence tree, while the inflection of determiners and adjectives in an NP depends on the properties of the noun. For the inflection of adjectives, Genpex also has to consider the determiner that is used before the adjective. In German, so-called 'strong inflection' has to be used after a cardinal number, 'weak inflection' after a definite determiner and 'mixed inflection' after an indefinite determiner. We currently use canoonet[3] as the source for German morphological information in Genpex.

Orthography is the process that converts the sentence trees to text. This is quite easy, because the word order is already defined by the tree structure. All values of the nodes in the tree can be joined together in a sentence in that order, separated by white spaces. The clauses in aggregated sentences are joined by a comma, except for the last conjunction where the word 'und' is used. A characteristic of German is that all nouns are capitalized. The Surface Realiser takes care of this, and also of the capitalization of the first word in each sentence, punctuation and the placement of paragraph boundaries. The generated texts are marked up with HTML for easy display in web browsers.

4 Evaluation

Potential users of Genpex (researchers working on test design) have been involved at different stages of development of the system, such as requirements specification and usability testing. Field trials with

[3] http://www.canoo.net/

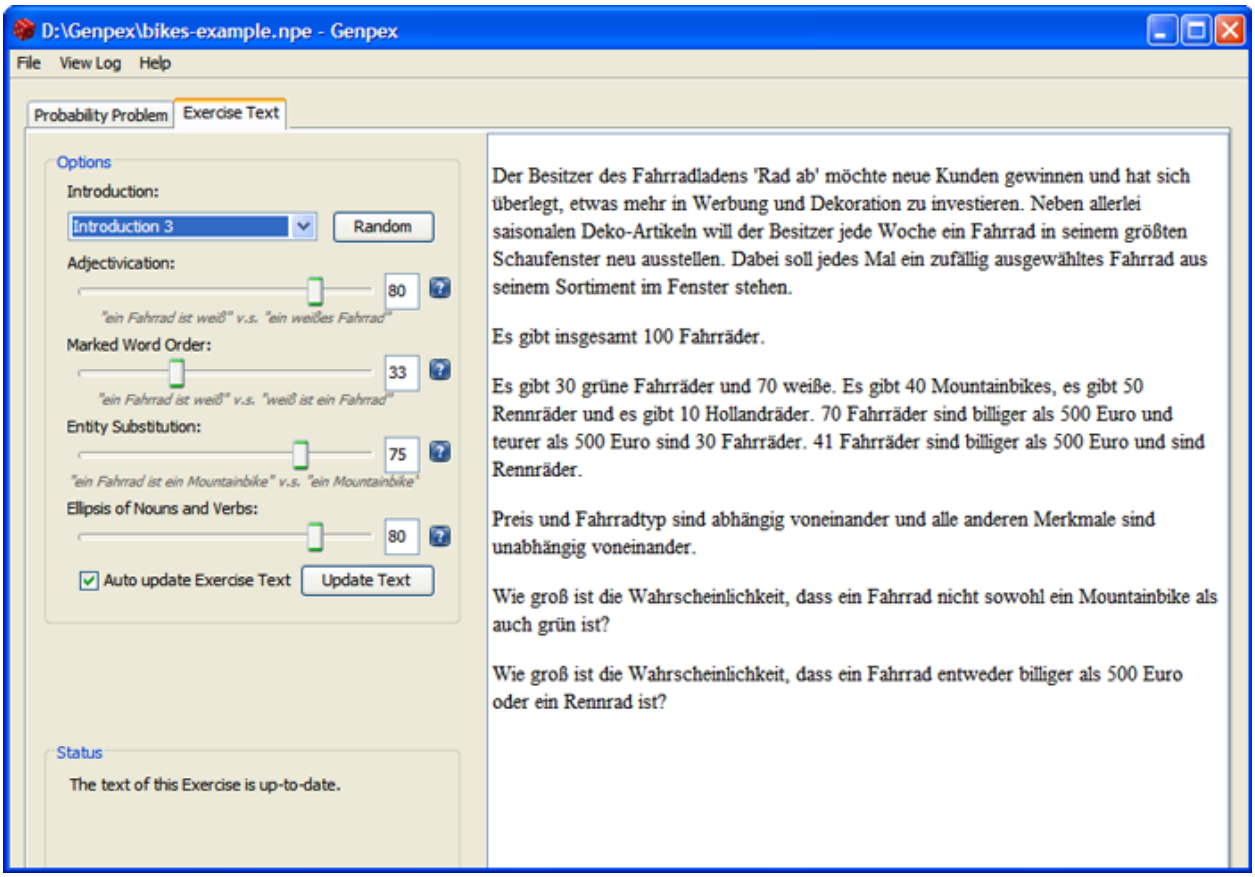

Figure 5: Screenshot of the GUI of Genpex, showing a variation of the narrative exercise in Figure 1. The introductory text was taken from Zeuch (In preparation).

students are future work, but we did carry out some preliminary, qualitative evaluations with a few native speakers of German (including one item generation expert) to test the grammaticality and understandability of the generated exercises. This revealed some small mistakes that have since been corrected, but also a few bigger problems with some of the variation techniques and other NLG aspects.

One of the things noted by the native speakers was that applying ellipsis sometimes leads to slightly unnatural sentences. The preferred type and degree of ellipsis is different for each type of sentence, but this is not taken into account by Genpex. As a consequence, the system frequently applies too much or too little ellipsis to the generated sentences, with less than ideal (though not ungrammatical) results. The existence of such preferred formulations is in line with the results of Cahill and Forst (2010), who carried out an experiment in which native speakers of German evaluated a number of alternative realisations of the same sentence. Their subjects accepted some variation in word order, but showed a clear preference for some of the alternatives.

Some of the generated question sentences also sounded a bit forced to the native speakers. For example, the question template for joint probabilities ($A \wedge B$) uses the formal phrasing "sowohl... als auch" (both ... and), whereas a simple "und" (and) would be the more natural choice for most questions. However, in some question contexts, in particular those involving negations, using the simpler formulation might lead to the kind of scope ambiguities mentioned in Section 2. Therefore, the choice was made to use "sowohl... als auch" in all cases, even in those where it is not strictly necessary. Similarly, questions asking for a conditional probability were found to be somewhat difficult to understand. For these questions, readability might be improved by using

two sentences to express them, along the lines of "Consider the set of bicycles that are not mountain bikes. What is the probability that one of those bicycles is either black or white?" as an alternative to the more complex formulation given in Figure 4.

The comments by the native speakers suggest that in some cases, Genpex goes too far in its "one size fits all" approach, and that we should try to add more flexibility to the NLG component, allowing it to make finer distinctions in the application of variation techniques to specific sentences and of question templates to specific question types.

5 Discussion

The texts currently being generated by Genpex are grammatical, but our native speakers reported that some sentences had to be studied carefully before it was possible to get the information needed to solve the problem. No actual misinterpretations occurred, but the increased reading time (as compared to more preferred formulations) may still increase the difficulty of the exercise. A thorough investigation into the effect of textual variations on item difficulty is therefore necessary. Genpex supports this type of research by enabling the systematic application of different variations, while logging all textual operations that have been applied and saving them together with the generated text. The underlying probability problem is saved together with the text as well, so all factors that certainly or potentially influence item difficulty are known. This makes it relatively easy to test the influence of those factors on the difficulty of the exercise, for example by carrying out the kind of statistical and cognitive analysis advocated by Graf et al. (2005).

The effect of the main parameters of the probability problems in Genpex (i.e., the type of question being asked) was already statistically analyzed by Holling et al. (2009) and Zeuch (In preparation). They used automatically generated items similar to the exercises generated by Genpex, except that their exercises did not have variations in wording apart from context-related ones. Also, the exercises used by Holling et al. (2009) mentioned probabilities instead of counts in the statements.

Once we know more about the effects of the textual variations, Genpex can be of great value to test developers, given that there exists a great need for large amounts of learning and assessment materials with a controlled level of difficulty (Enright et al., 2002; Fairon and Williamson, 2002; Deane and Sheehan, 2003; Arendasy et al., 2006; Holling et al., 2008; Holling et al., 2009). The initial development and testing of the system is a one-time investment, which we expect will pay off afterward when large amounts of test items can be created with little effort. In particular, we think Genpex can be very useful in combination with Computerized Adaptive Testing (CAT). The system could be used for on-the-fly generation of new items for each individual student, adapted to that student's skill level estimated from his or her previous answers. Because every student gets custom exercises, the risk of frequently used items becoming known among students is reduced, thus increasing test security.

In principle, given that the factors influencing item difficulty are known, generating difficult items is not more complicated than generating easy ones. However, as illustrated in Section 2, combining multiple difficulty factors such as negation and joint probability may lead to textual formulations that are ambiguous or hard to understand, and which – if not successfully prevented in advance – may need to be filtered out or corrected by hand. For that reason, Genpex seems most suitable for the generation of exercises up to a moderate level of complexity. Still, even if the need for hand-crafting will not be completely eliminated, reducing it to complex items that require particularly careful wording already represents a big gain in efficiency.

Acknowledgements

We thank our anonymous reviewers and everybody who helped testing Genpex. Special thanks go to our colleagues from the University of Münster for providing the reference exercises on which the Genpex output was modelled. This research was funded by the Deutsche Forschungsgemeinschaft (DFG), Schwerpunktprogramm "Kompetenzmodelle zur Erfassung individueller Lernergebnisse und zur Bilanzierung von Bildungsprozessen" (SPP 1293), project "Rule-based Item Generation of Algebra Word Problems Based upon Linear Logistic Test Models for Item Cloning and Optimal Design".

References

Martin Arendasy, Markus Sommer, Georg Gittler, and Andreas Hergovich. 2006. Automatic generation of quantitative reasoning items: A pilot study. *Journal of Individual Differences*, 27(1):2–14.

Roan Boer Rookhuiszen. 2011. Generation of German narrative probability excercises. Master's thesis, University of Twente.

Aiofe Cahill and Martin Forst. 2010. Human evaluation of a German surface realisation ranker. In E. Krahmer and M. Theune, editors, *Empirical Methods in Natural Language Generation*, volume 5790 of *Lecture Notes in Computer Science*, pages 201–221. Springer, Berlin / Heidelberg.

Paul Deane and Kathleen Sheehan. 2003. Automatic item generation via frame semantics: Natural language generation of math word problems. Educational Testing Service, Princeton. Paper presented at the Annual Meeting of the National Council on Measurement in Education (Chicago, IL, April 22-24, 2003).

Mary K. Enright, Mary Morley, and Kathleen M. Sheehan. 2002. Items by design: The impact of systematic feature variation on item statistical characteristics. *Applied Measurement in Education*, 15(1):49–74.

Cédrick Fairon and David M. Williamson. 2002. Automatic item text generation in educational assessment. In *Proceedings of TALN 2002*, pages 395–401.

Edith Aurora Graf, Stephen Peterson, Manfred Steffen, and René Lawless. 2005. Psychometric and cognitive analysis as a basis for the design and revision of quantitative item models. Technical Report RR-05-25, Educational Testing Service, Princeton.

Karin Harbusch and Gerard Kempen. 2009. Generating clausal coordinate ellipsis multilingually: A uniform approach based on postediting. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 138–145.

Derrick Higgins, Yoko Futagi, and Paul Deane. 2005. Multilingual generalization of the ModelCreator software for math item generation. Technical Report RR-05-02, Educational Testing Service, Princeton.

Heinz Holling, Helen Blank, Karoline Kuchenbäcker, and Jörg-Tobias Kuhn. 2008. Rule-based item design of statistical word problems: A review and first implementation. *Psychology Science Quarterly*, 50(3):363–378.

Heinz Holling, Jonas P. Bertling, and Nina Zeuch. 2009. Automatic item generation of probability word problems. *Studies In Educational Evaluation*, 35(2-3):71–76.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge, UK.

Nina Zeuch. In preparation. *Rule-based item construction: Analysis with and comparison of linear logistic test models and cognitive diagnostic models with two item types*. Ph.D. thesis, University of Münster.