

S I M O N F R A S E R U N I V E R S I T Y

Department of Linguistics

SANTY:

A Spell Checking Algorithm

for Treating Predictable Verb Inflection Mistakes

Made by Non-Native Writers of German

Term Paper for LING 807 – Computational Linguistics

Instructor: Dr. Paul McFetridge

Written by: Anne Rimrott

Spring Semester 2003

Contents

| | |
|--|----|
| List of Tables and Figures | ii |
| 1 Introduction..... | 1 |
| 2 Classification of Unknown Words and Analysis of Corpus | 2 |
| 3 An Overview of Spell Checkers | 4 |
| 4 SANTY | 6 |
| 4.1 Description of Program..... | 6 |
| 4.2 Challenges..... | 7 |
| 4.3 Evaluation of Program..... | 8 |
| 4.4 Shortcomings and Possible Future Developments | 10 |
| 4.5 Application and Interaction with Other Programs..... | 11 |
| 5 Conclusion | 12 |
| 6 References..... | 13 |
| 7 Appendix..... | 14 |
| 7.1 Illustration of SANTY's Algorithm | 14 |
| 7.2 Sample of Misspelled Verb Forms | 16 |

List of Tables and Figures

Tables:

| | |
|--|----|
| Table 1: Inflection of Weak Verbs | 2 |
| Table 2: Inflection of Strong Verbs | 3 |
| Table 3: Error Categories Based on Corpus | 4 |
| Table 4: Examples of Input-Output Pairs | 9 |
| Table 5: Demonstration of SANTY's Algorithm..... | 15 |
| Table 6: Sample of Misspelled Verb Forms | 16 |

Figures:

| | |
|---|---|
| Figure 1: Basic Architecture of SANTY | 7 |
|---|---|

1 Introduction

When it comes to inflecting German verbs, non-native writers often make predictable spelling mistakes that are not adequately corrected by most modern spell checkers. Many of these mistakes do not actually stem from an inability to employ German spelling rules, but are rather caused by applying various morphological rules to verbs where these rules do not apply. However, these incorrectly inflected verbs are unknown words to a regular text editor and are thus passed on to the spell checker for correction even though they would more appropriately be treated by a grammar checker. While a regular spell checker that employs approximate string matching techniques is able to provide appropriate spelling suggestions when the misspelled word closely resembles the spelling of the intended word (e.g. when writing "she gos" instead of "she goes"), it does not incorporate morphological knowledge and non-native intuitions into its algorithm and is thus at a loss when the misspelled word and the intended word are not orthographically related (e.g. when writing "he goed" instead of "he went"). SANTY, the prototype program presented in this paper, provides more adequate spelling suggestions to non-native speakers by incorporating morphological knowledge as well as knowledge of common types of learner mistakes into its spell checking algorithm. This new approach could eventually make it a valuable tool for non-native writers of German.

The paper is organized as follows. Chapter 2 will present a short overview of the classification of unknown words as well as a summary of authentic learner mistakes. In chapter 3, different spell checking algorithms will be discussed in relation to SANTY. In chapter 4, SANTY will be described and analyzed. The concluding fifth chapter will summarize the main points of this paper.

2 Classification of Unknown Words and Analysis of Corpus

Essentially, the problem a spell checker is faced with is the identification and classification of unknown words. With regards to the program presented in this paper, an unknown word can be defined as a word that is not listed in its dictionary. Several authors have proposed classifications of unknown words. Toole (2000: 173), for example, categorizes unknown words into proper names, abbreviations, numbers, misspellings, morphological variants of known words, and words missing from the dictionary. More specific to second language acquisition, both Tschichold et al. (1997: 8) and Johannessen et al. (2002) list morphological errors as a category when classifying spelling and grammar errors made by non-native speakers of a language. The subset of unknown words that are to be considered by the present program also relate to morphology: the mistakes are systematic misspellings of German present indicative or past participle verb forms that are caused by incorrectly applying German verb inflection rules.

Before a classification of these morphological mistakes is possible, a brief and very general overview of German verb morphology is necessary. With respect to inflection, verbs can be divided into weak, strong and irregular verbs. In the present indicative tense, weak verbs are inflected by adding a person-number suffix to the verb stem. The past participle is derived by adding the participial prefix *ge-* and the suffix *-t* to the stem. The following table illustrates this:

| | Singular | Plural |
|-----------------------------|------------------------------------|--------------|
| Infinitive: hören 'to hear' | 1 st person (ich) hör-e | (wir) hör-en |
| Past Participle: ge-hör-t | 2 nd person (du) hör-st | (ihr) hör-t |
| | 3 rd person (er) hör-t | (sie) hör-en |

Table 1: Inflection of Weak Verbs

Strong verbs basically take the same present tense suffixes, but many of them "change their stem vowels in the second and third persons singular of the present indicative [...] and

often in the past participle" (Conant 1991: 42). The past participle suffix for strong verbs is –en. Consider, for example, sprechen 'to speak':

| | | |
|---------------------------------|---------------------------------------|-----------------|
| | Singular | Plural |
| Infinitive: sprechen 'to speak' | 1 st person (ich) sprech-e | (wir) sprech-en |
| Past Participle: ge-sproch-en | 2 nd person (du) sprich-st | (ihr) sprech-t |
| | 3 rd person (er) sprich-t | (sie) sprech-en |

Table 2: Inflection of Strong Verbs

Görz (1988: 213) states that "[i]n most cases irregular verbs can be treated like regular strong verbs with the exception of some special forms". It is also relevant to the discussion below that in some cases the first and third person singular verb forms do not take a suffix at all (e.g. 'ich kann', 'er will').

Based on a corpus provided by Dr. Trude Heift (Director of the Language Learning Centre at Simon Fraser University) spelling mistakes made by non-native writers of German were analyzed and classified. Out of the various misspelled verb forms, mistakes that were caused by incorrectly applying German verb morphology were investigated closer. It became apparent that there are six main sources of systematic errors:

| Error Category | Description | Examples |
|---|--|--|
| Past Participle (PP) Errors | Deriving past participles by adding the suffix –en instead of –t and vice versa | *'gewohnen' for 'gewohnt' *'gefahrt' for 'gefahren' |
| Stem Vowel (SV) Errors | Omitting to change a stem vowel where it is necessary | *'helfst' for 'hilfst' [helfen] ¹ |
| | Changing a stem vowel where no SV change applies | *'bezählt' for 'bezahlt' [bezahlen] |
| | Correctly making a stem vowel change but with the wrong vowel | *'durf' for 'darf' [dürfen] |
| Errors relating to various kinds of Irregular Forms (IF errors) | Inflecting verbs that are irregular in some way as if they were completely regular | 'ich *dürfe' for 'ich darf' [dürfen] 'er *findt' for 'er findet' [finden] |
| Errors relating to 'ß' | writing 'ß' instead of 'ss' or vice versa | *'muß' for 'muss' |
| Errors relating to epenthesis | epenthesizing 'e' where it is not necessary or omitting to epenthesize it where it is required | *'findt' for 'findet' *'wohnet' for 'wohnt' |

¹ The infinitive form is provided in square brackets if it clarifies the discussion at hand.

| | | |
|-----------------------------|--|--|
| Errors relating to suffixes | Omitting a suffix where it is necessary or placing a suffix where it is not required (especially in the 1 st and 3 rd person singular) | 'ich *fahr' for 'ich fahre' 'ich *darfe' for 'ich darf' |
| Other Errors | Errors that are not captured by any of the categories above | 'konnte' for 'kann' '*lauften' for 'läuft' |

Table 3: Error Categories Based on Corpus

In addition to these categories, many misspelled words evidence a combination of these systematic errors. The form '*gefährht' (for 'gefahren'), for example, demonstrates problems with past participle formation as well as stem vowel application. SANTY is designed to detect and correct unknown words that are caused by mistakes belonging to the first three categories (i.e. PP, SV, and IF problems).

3 An Overview of Spell Checkers

After a short overview of possible designs for spell checkers and remarks on how SANTY can be classified, this chapter will briefly discuss the inadequacy of grammar and spell checkers for second language learners. It will be demonstrated that SANTY is a spell checker for non-native writers that tries to bridge the gap between spelling checkers and grammar checkers in general.

The following discussion is based on Berghel (1987). According to Berghel, spell checkers perform three basic operations: document normalization, spelling verification, and spelling correction. Document normalization entails "standardization of character encodings with regard to case, font, character set, etc.". Apart from standardizing case, SANTY does not normalize its input. The only input it can process effectively is written in standard ASCII characters (that means that, according to standard conventions, German *ä*, *ö*, *ü*, and *ß* are written as *ae*, *ou*, *ue*, and *ss* respectively).

With regard to spelling verification, the second operation mentioned by Berghel, there are two radically different approaches. In the *deterministic approach* (e.g. pattern matching

analyses), input words are checked against a dictionary. If they are not found in the dictionary they are regarded as misspellings. The *probabilistic approach* (e.g. n-gram analysis), on the other hand, does not attempt to match an inputted string with a lexical entry but rather employs algorithms to associate with each input word a "coefficient of peculiarity". Targets with a high coefficient of peculiarity are identified as possible misspellings. The approach used in SANTY is deterministic: an inputted word is treated as a misspelling if it is not found in the dictionary.

The last operation, spelling correction, can also take various forms. At one end of the scale, there are simple spelling correctors that will provide an "interactive facility which allows the user to choose a substitute from a list of alternative words". The words in this list are most commonly related to the misspelled word through a similar orthography (e.g. the MS WORD 2000 spell checker suggests 'goad, good, gored, geed, god, goes, geode, gold' as corrections of the misspelled word 'goed'). At the other end of the scale, the spelling correction "would involve the actual substitution of the correct spelling of the intended word for its misspelled counterpart". SANTY is a spell checker that provides very specific and limited spelling suggestions (usually one or two output words, never more than four) and is thus closer to the latter form of spelling correction. Since SANTY operates in a very limited environment with very specific assumptions about the input word (see chapter 4.4), it is possible to provide such specific feedback that almost corresponds to substituting the misspelled word for its correct counterpart.

Standard spell checkers and grammar checkers are usually not designed for non-native speakers (Jacobs & Rodgers 1999, Tschichold et al. 1997). When discussing standard spell checkers for German, Kese et al. (1992: 126) note the following:

[...] many more errors could be detected by a spelling corrector if it possessed at least some rudimentary linguistic knowledge. In the case of a word that takes irregular forms (like the German verb "laufen" [...]), a standard system seems to "know" the word and its forms for it is able to verify them, e.g., by simple lexicon lookup. Yet when confronted with a regular though false form of the very same word (e.g. with "laufte" as the 1st/3rd pers. sg. simple past ind. act. [...]), such a system normally fails to propose the corresponding irregular form ("lief") as a correction alternative.

Similarly, the MS WORD 2000 spell checker does not provide "went" as a correction alternative when the input word is "goed" (see above). These shortcomings are addressed by SANTY. In fact, SANTY would *only* output "lief" and "went" respectively in the two cases just mentioned. In theory, SANTY is meant to be a module for a standard spell checker. Together, the standard spell checker and SANTY would output more correction alternatives.

The problems SANTY addresses are related to both spelling and grammar: it is designed to correct spelling mistakes that are caused by an inadequate application of the grammar (i.e. morphology) of German. Ideally then, the misspelled words should be corrected by a grammar checker not a spell checker. In practice, this is not possible as standard grammar checkers are dependent on correctly spelled input that is passed on to them by a spell checker. Thus, SANTY addresses the spelling / grammar interface by being a spell checker that incorporates grammatical knowledge into its algorithm.

4 SANTY

4.1 Description of Program

As has already been mentioned, SANTY is a spell checking algorithm for treating predictable verb inflection mistakes by non-native writers of German. It uses morphological analyses to determine the intended verb inflection based on the wrongly inflected input word. The following chart illustrates the basic architecture of the program:

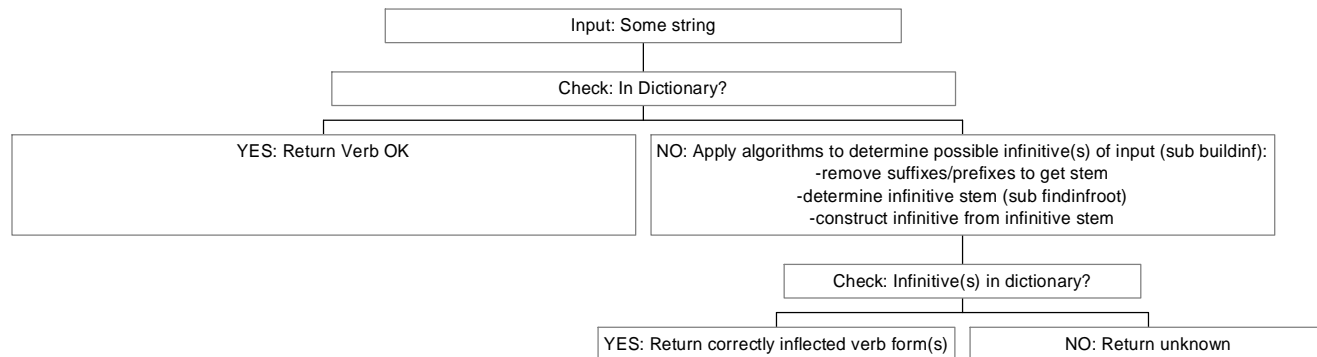


Figure 1: Basic Architecture of SANTY

How does this work in detail? Only an input string that is not found in the dictionary is passed on to the sub *buildinf* for morphological analysis. This sub uses regular expressions to remove potential suffixes and prefixes from the input word to determine possible input stems². As a verb can have more than one stem, these stems are then passed on to the sub *findinfroot* which returns the infinitive stem for each inputted stem it can find in a file of stem aliases³. Back in the *buildinf* sub, infinitives are created out of these infinitive stems. The infinitives are then passed on to the main program where they are looked up in a file that contains all verb forms and verb positions (see footnote 2). If the lookup is successful, the correctly inflected verb form(s) that corresponds to the inflectional form(s) of the input are returned⁴.

4.2 Challenges

Several challenges needed to be overcome by the program. In particular, the 3rd pers. sg. and 2nd pers. pl. suffix as well as one of the two participial suffixes have the same form: -t.

² The program saves the inflectional information it removed so that later on, the corresponding correct inflectional forms can be outputted. Each inflectional form is linked to a position (the 1st pers. sg. suffix – e occupies position 1, the 2nd pers. sg. suffix –st has position 2, etc., a past participle receives position 7).

³ The verb "wissen", for example, has the stems "weiß" and "wiss" in the present tense and "wuss" in the past participle. The file with stem aliases contains the infinitive stem of each verb along with present tense and past participle stems and non-existent stems that are frequently used by non-native speakers according to the corpus data. Based on the findings in the corpus, the patterns were generalized to add non-existent stems to verbs that were not present in the corpus. Most of the non-existent stems involve incorrect stem vowel changes (e.g. "*bezahl-" from "bezahlen").

⁴ The table in appendix 7.1 gives a detailed illustration of the program design.

The letter "t" is also the second part of the 2nd pers. sg. suffix –st. The –en suffix can also take on multiple meanings: it can be the 1st or 3rd pers. pl. suffix or the other participial ending. In addition, ge- at the beginning of a word can signify the participial prefix or it could just represent the first two letters of the verbal stem. SANTY addresses these problems of ambiguity by taking every possible meaning of the suffixes and prefixes into account. This is why in some cases, SANTY returns several spelling suggestions, each corresponding to a different meaning of prefixes and suffixes. When entering "gefaellen", for example, SANTY returns the following four suggestions:

| | | |
|-------------------------------|--------------------------|---------------------------|
| Spelling Suggestion: gefallen | Position: 7 ⁵ | From Infinitive: fallen |
| Spelling Suggestion: gefallen | Position: 4 | From Infinitive: gefallen |
| Spelling Suggestion: gefallen | Position: 6 | From Infinitive: gefallen |
| Spelling Suggestion: gefallen | Position: 7 | From Infinitive: gefallen |

And for "laueft", it returns:

| | | |
|-----------------------------|-------------|-------------------------|
| Spelling Suggestion: laeuft | Position: 3 | From Infinitive: laufen |
| Spelling Suggestion: lauft | Position: 5 | From Infinitive: laufen |

In unambiguous cases SANTY outputs just one spelling suggestion, consider the output for "duerfe":

| | | |
|---------------------------|-------------|--------------------------|
| Spelling Suggestion: darf | Position: 1 | From Infinitive: duerfen |
|---------------------------|-------------|--------------------------|

If SANTY received information from a syntactic parser ambiguity could be reduced drastically and the number of outputted spelling suggestions could be cut down to one in the majority of cases.

4.3 Evaluation of Program

Over 100 incorrectly inflected verb forms were selected from the corpus in order to find appropriate algorithms when designing SANTY. After completion of the program, SANTY was tested on these verb forms. SANTY performs with 100% accuracy when tested on

⁵ Position 1 = 1st pers. sg., position 2 = 2nd pers. sg., etc., position 7 = past participle.

input that evidences the three kinds of mistakes SANTY was designed to correct. That means the program has no trouble processing input that is misspelled due to incorrect application of stem vowel changes (SV errors), inflecting irregular verbs completely regular (IF errors), mixing up the past participle suffixes (PP errors) or a combination of SV, IF, and PP errors. However, as mentioned in chapter 2, there were also other types of mistakes that SANTY is not able to process.

The success of SANTY is naturally dependent on the size of the dictionary and on the quality of the file that contains the stem aliases. Currently, SANTY's dictionary contains over 60 verbs and the file with stem aliases lists infinitive, present tense, past participle, and non-existent stems (see footnote 3) for each of these verbs. For example, the file with stem aliases lists "hoer", the actual stem, and "hor", a non-existent stem that was used nine times (!) by non-native writers, as possible stems for "hören" ('to hear'). That way SANTY can offer "hoeren" (positions 4 and 6) as spelling suggestions for the misspelled input "horen".

SANTY can provide adequate spelling suggestions for 25 misspelled verb forms that occur more than once in the small sample corpus and many more verb forms that occur just once⁶. The table below shows some input-output pairs and the frequency of occurrence of the misspelled input word in the corpus.

| Input | Output | Frequency in Corpus | Infinitive/Gloss |
|--------------|------------------------|----------------------------|-------------------------|
| gewohnen | gewohnt (pos.: 7) | 11 | wohnen/live |
| bezaehlt | bezahlt (3, 5, 7) | 14 | bezahlen/pay |
| horen | hoeren (4, 6) | 9 | hören/hear |
| findt | findet (3, 5) | 8 | finden/find |
| kannt | kann (3) koennt (5) | 5 | können/can |

Table 4: Examples of Input-Output Pairs

The fact that many of these misspelled verb forms occur more than once in the small sample corpus indicates that the mistakes SANTY detects are common learner mistakes.

⁶ Some examples of misspelled verb forms can be found in appendix 7.2.

4.4 Shortcomings and Possible Future Developments

The success of SANTY is based on several assumptions. It is assumed that a part-of-speech-tagger has identified the unknown words that are passed on to SANTY as verbs, and, more specifically, that these misspelled verbs are believed to be misspellings of present tense or past participle verb forms. Moreover, SANTY is only useful when the misspellings can be traced back to one or more of the three error categories mentioned above (IF, SV, and PP errors). In all other cases, SANTY is not able to process input in a meaningful way. This indicates obvious shortcomings: SANTY cannot process misspelled verbs that are not in the present tense or past participle, let alone words that belong to different categories (e.g. nouns, adjectives). In later versions of the program, the algorithm could be expanded to cover the other error types listed in chapter 2 as well as other word categories.

In addition, there are some problems with efficiency. Because of the way the program is presently written, the file that contains all verb forms and verb positions lists all six present tense verb forms separately even though some of these forms are the same (e.g. "spielt" is the 3rd pers. sg. and the 2nd. pers. pl.). This multiple listing was done so that SANTY can access every verb position when doing the lookup at the end. Furthermore, all stems are listed in the stem alias file, even if there are no stem aliases. This is also due to the way the prototype is written. In a later version of SANTY, these issues could be handled in a more efficient way.

It also has to be pointed out that SANTY considers any string not present in its dictionary to be misspelled. However, sometimes non-native speakers inflect a verb in another tense or mood when trying to inflect it in the present tense indicative. For "mochte", for example, SANTY would suggest the present indicative 1st pers. sg. "möchte" even though the input word is a legitimate past indicative form of the verb "mögen" (to like). The

problem of detecting and correcting verb forms that are actually correct could be solved by either expanding the dictionary to include all possible forms of each verb or by combining SANTY with a sophisticated parser that recognizes the tense and mood a particular verb is supposed to be inflected in. Also, we could restrict the input by incorporating SANTY into a CALL⁷ program where the input is known to be present tense.

Another issue that could become problematic in later versions of SANTY is that many verbs are similar in spelling and can be easily confused by non-native writers. Consider, for example, part of the paradigm for "fallen" (to fall), "fällen" (to fell), "gefallen" (to please), and "befallen" (to overcome):

fallen: falle, fällst, gefallen, fiel
fällen: fälle, fällst, gefällt, fällte
gefallen: gefalle, gefälltst, gefallen, gefiel
befallen: befalle, befällst, befallen, befiel

The problem of verb similarity basically means that SANTY will output many different spelling suggestions for one input word. The intended spelling could be determined by a (not yet existing) sophisticated semantic parser or it could be easily ascertained in a restricted CALL environment.

4.5 Application and Interaction with Other Programs

It has been emphasized throughout this paper that SANTY is a spell checker for non-native speakers that was created by analyzing authentic learner input. It is thus obviously meant to be used by non-native writers of German.

There are two possible ways SANTY could interact with a larger program. It could either be used as a module for a standard spell checker in a standard word processor. Or SANTY could be incorporated into a CALL system where it could help provide learners with valuable feedback. Instead of just highlighting spelling errors, SANTY could offer

⁷ CALL = Computer-Assisted Language Learning.

grammatical information in the form of flash cards. It could point the learner to problems s/he has with stem vowel changes, past participles and irregularities of verbs in general.

5 Conclusion

This paper has introduced SANTY, a spell checker that incorporates grammatical knowledge into its spell checking algorithm and is designed to be used by non-native writers. Second language learners often make mistakes that follow predictable patterns. However, these mistakes are not detected by standard spell checkers. SANTY can detect three specific types of errors that learners frequently make in relation to present tense and past participle verb forms. A corpus of authentic misspelled verb forms was examined to arrive at the algorithms built into SANTY. Later on, SANTY was tested on these learner mistakes and was found to be 100% accurate in relation to these mistakes (but not others). SANTY could be used in a standard word processor or in a CALL environment to provide enhanced learner feedback. The functionality of the program can be significantly improved if it is used in connection with a part of speech tagger and a semantic parser. The program could also be extended to process other word categories such as nouns. Later versions of the program would try to resolve the problems and limitations of the current prototype version and could thus make SANTY a useful tool for non-native writers that would supply valuable feedback on spelling mistakes.

6 References

Berghel, H. L. "A Logical Framework for the Correction of Spelling Errors in Electronic Documents." Information Processing and Management: an International Journal 23.5 (1987): 474-494.

Conant, Jonathan B. Cochran's German Review Grammar. 4th ed. Englewood Cliffs: Prentice-Hall, 1991.

Görz, Günther and Dietrich Paulus. "A Finite State Approach to German Verb Morphology." Proceedings of COLING-88 (1988): 212-215.

Jacobs, Gabriel, and Catherine Rodgers. "Treacherous Allies: Foreign Language Grammar Checkers." CALICO Journal 16.4 (1999): 509-529.

Johannessen, Janne Bondi et al. "The Performance of a Grammar Checker with Deviant Language Input." Proceedings of the 19th International Conference on Computational Linguistics (2002): 1223-1227.

Kese, Ralf et al. "Extended Spelling Correction for German." Proceedings of the 3rd Conference on Applied Natural Language Processing (1992): 126-132.

Toole, Janine. "Categorizing Unknown Words: Using Decision Trees to Identify Names and Misspellings." Proceedings of the 6th Conference on Applied Natural Language Processing (2000): 173-179.

Tschichold, Cornelia et al. "Developing a New Grammar Checker for English as a Second Language." From Research to Commercial Applications: Making NLP Work in Practice. Proceedings of a Workshop Sponsored by the Association for Computational Linguistics. Eds. Burstein, J. and C. Leacock. Madrid: UNED, 1997. 7-12.

7 Appendix

7.1 Illustration of SANTY's Algorithm

The following table illustrates SANTY's algorithm.

| Input: | | machst | fraegst | gefahrt | gefaellen | bezaehlt |
|----------------------|---|----------------|---|---|---|--|
| Dictionary lookup | | found → end | not found → passed on to buildinf sub | not found → passed on to buildinf sub | not found → passed on to buildinf sub | not found → passed on to buildinf sub |
| sub buildinf | if loop #1 (for incorrectly inflected past participles where the infinitive also begins with ge- or be-) if input word matches <code>/^(ge be)(.[aeiou].+)(t en)\$/</code> take off -t or -en and keep the rest | | x | gefahrt → pass on to sub findinfroot | gefaell → pass on to sub findinfroot | bezaehl → pass on to sub findinfroot |
| sub findinfroot | Stem aliases. If stem is found return infinitive stem | | | x | gefall → pass back to sub buildinf | bezahl → pass back to sub buildinf |
| sub buildinf | build infinitive by adding -en to stem | | | | gefallen → pass on to main program | bezahlen → pass on to main program |
| Lookup in %irregverb | Look up infinitive and return inflected verb form that corresponds to the suffixes (and prefixes) that were taken off along with position and infinitive form | | | | Suggested spelling: gefallen, Position: 7, From infinitive: gefallen | Suggested spelling: bezaehl, Position: 7, From infinitive: bezahlen |
| sub buildinf | if loop #2 (for incorrectly inflected past participles where ge- is only the participial prefix but is not part of the infinitive stem) if <code>(\$verb =~ /[^]ge(.[aeiou].+)(t en)\$/)</code> take off ge- and -t or -en | | x | fahrt → pass on to sub findinfroot | faell → pass on to sub findinfroot | zaehl → pass on to sub findinfroot |

| | | | | | | |
|-------------------------|--|--|---|--|--|---|
| sub findinfroot | Stem aliases. If stem is found return infinitive stem | | | fahr → pass back to sub buildinf | fall → pass back to sub buildinf | zahl → pass back to sub buildinf |
| sub buildinf | build infinitive by adding – en to stem | | | fahren → pass on to main program | fallen → pass on to main program | zahlen → pass on to main program |
| Lookup in %irregverb | Look up infinitive and return inflected verb form that corresponds to the suffixes (and prefixes) that were taken off along with position and infinitive form | | | Suggested spelling: gefahren, Position: 7, From infinitive: fahren | Suggested spelling: gefallen, Position: 7, From infinitive: fallen | Suggested spelling: bezahlt, Position: 7, From infinitive: zahlen |
| sub buildinf | if loop #3 (for incorrectly inflected verbs in the present tense) if input word ends in one of the 6 suffixes take off suffix and keep the | | fraegs and fraeg → pass both on to sub findinfroot | gefahr → pass on to sub findinfroot | gefaell → pass on to sub findinfroot | bezaehl → pass on to sub findinfroot |
| sub findinfroot | Stem aliases. If stem is found return infinitive stem | | fraegs: not found fraeg: found, inf. stem: frag → pass back to sub buildinf | x | gefall → pass back to sub buildinf | bezahl → pass back to sub buildinf |
| sub buildinf | build infinitive by adding – en to stem | | fragen → pass on to main program | | gefallen → pass on to main program | bezahlen → pass on to main program |
| Lookup in %irregverb | Look up infinitive and return inflected verb form that corresponds to the suffixes (and prefixes) that were taken off along with position and infinitive form | | Suggested spelling: fragst, Position: 2, From infinitive: fragen | | Suggested spelling: gefallen, Positions: 4 and 6, From infinitive: gefallen | Suggested spelling: bezahlt, Positions: 3 and 5, From infinitive: bezahlen |

Table 5: Demonstration of SANTY'sAlgorithm

Infinitives of the example input words: "machen" (do), "fragen" (ask), "fahren" (drive),
"fallen" (fall) / "gefallen" (please), "bezahlen" (pay).

In summary, we get the following input-output pairs:

Input: machst

Output: HIT: "machst" is in dictionary. Program doesn't apply.

Input: fraegst

Output: Spelling Suggestion: fragst Position: 2 From Infinitive: fragen

Input: gefahrt

Output: Spelling Suggestion: gefahren Position: 7 From Infinitive: fahren

Input: gefaellen

Output: Spelling Suggestion: gefallen Position: 7 From Infinitive: fallen
 Spelling Suggestion: gefallen Position: 4 From Infinitive: gefallen
 Spelling Suggestion: gefallen Position: 6 From Infinitive: gefallen
 Spelling Suggestion: gefallen Position: 7 From Infinitive: gefallen

Input: bezaehlt

Output: Spelling Suggestion: bezahlt Position: 3 From Infinitive: bezahlen
 Spelling Suggestion: bezahlt Position: 5 From Infinitive: bezahlen
 Spelling Suggestion: bezahlt Position: 7 From Infinitive: bezahlen

7.2 Sample of Misspelled Verb Forms

The following table lists some misspelled verb forms that were detected in the corpus and that can be adequately processed by SANTY:

| Misspelled Input | Infinitive | Gloss | Misspelled Input | Infinitive | Gloss |
|------------------|------------|---------------|------------------|------------|------------|
| bezaehlt | bezahlen | pay | konnen | können | be able to |
| duerfe | dürfen | be allowed to | konnst | können | be able to |
| durfe | dürfen | be allowed to | koenne | können | be able to |
| gefahrt | fahren | drive | laueft | laufen | walk |
| faehre | fahren | drive | laeufen | laufen | walk |
| findt | finden | find | mochte | mögen | like |
| geholf | helfen | help | gesieht | sehen | see |
| gehelft | helfen | help | suecht | suchen | look for |

Table 6: Sample of Misspelled Verb Forms