Flexible Abbreviation Expansion

Gregg M. Stum, Patrick W. Demasco
Applied Science and Engineering Laboratories
University of Delaware / Alfred I. duPont Institute
Wilmington, Delaware USA

Abstract

This project is part of an augmentative and alternative communication project directed toward reducing the burdens imposed on both clinicians and users by traditional fixed abbreviation expanders. These burdens include the requirement to devise and memorize good unique abbreviations for the items that are to be abbreviated. This is achieved by the application of a set of formal abbreviation rules across the user's vocabulary in order to anticipate the abbreviations that the user will enter. A flexible expander is currently under development for the Microsoft Windows operating environment. Being independent of any particular application this expander offers consistent access to abbreviation expansion from within any context.

Background

Ever since [KV82], abbreviation expansion has been identified as an important technique to reduce the typing load of individuals for whom typing is their primary form of communication, or for whom typing is a significant physical task. It is important for these individuals to produce text as quickly as possible and with the least possible effort.

Abbreviation systems developed since then rely on the assignment of a specific unique abbreviation to each word that the user wishes to abbreviate. The user then enters this abbreviation and the system replaces it with the associated expansion. This approach is easy to implement and operates efficiently.

Unfortunately fixed abbreviation expansion imposes numerous unnecessary burdens. The user is required to memorize all of the abbreviations that the system recognizes. Someone must manually devise and enter any abbreviations not already in the system. The set of abbreviations is probably not optimized for the cognitive skills of the user.

In Word Compansion [DLM89] there is no predefined assignment of abbreviations to expansions. Rather, the system performs a number of transformations on an abbreviation in attempting to infer the intended expansion. In this way the user simply enters a reasonable abbreviation for the expansion they intend, rather than recall one that they had memorized.

Flexible Abbreviation Expansion is thus a hybrid of these two approaches. It uses knowledge of how people go about abbreviating in order to anticipate the most likely abbreviations that a user will enter for any given expansion. This allows it to operate as efficiently as traditional fixed expanders while bringing the power and flexibility of Word Compansion to the user.

Statement of the Problem

Current abbreviation expanders succeed primarily in decreasing a user's typing load. Ideally this also translates into increasing the rate at which the user produces text. While just this is of tremendous benefit, there is still room to significantly improve the user's typing capability.

The first improvement is to free the user from memorizing specific abbreviation assignments. Current systems do decrease the amount of memorization required of the user by utilizing mnemonic coding approaches to make the abbreviation assignments easier to remember. Typically these take the form of: <2> = 'to', <b> = 'be', <4>= 'for', and <bc> = 'because', and so forth. [V87][R91]. While such approaches work very well for the most common and frequently used items, the fact that they are still unique assignments means that once an abbreviation is used for one item, it is unavailable for any other item, no matter how good or appropriate it might be in that case as well.

This leads to the second improvement being covering the user's broad vocabulary needs. Generally expanders come with a set of default assignments for common vocabulary items such as function words, articles, and frequent verbs, nouns, and phrases. These do comprise the majority of the user's typing needs; they aren't its totality. There is a vast reservoir of subject-specific vocabulary that the user taps on demand. To abbreviate any of these items with a current expander, someone must have manually assigned an abbreviation to it. Since any of these items are used infrequently, it is not practical to make the effort in coming up with abbreviations for each of these that are both unique as well as easy for the user to remember.

A solution to this need illustrates the third improvement, tailoring the set of abbreviations to the individual user. To overcome the demands of manually making the abbreviation assignments, one could consider using sets of predefined abbreviations; one for 'chemistry', one for 'basketweaving', one for 'golf', and so one. Two problems with this solution are not being able to insure the uniqueness of any particular abbreviation, and that the abbreviation assignments do not in general reflect the preferences of its user.

Another improvement is making abbreviation expansion available to the user within any typing context in a consistent manner. Currently an expander is tied to a specific application. If a user has an expander for a speech aid it is separate and distinct from one they would use with a word processor. Not only must the user how to operate each of these, they must also remember a different set of abbreviations with each, or at least remember which and how the abbreviations differ.

Approach

Flexible Abbreviation Expansion makes all of these benefits available to the user. It achieves this by using a set of abbreviations for a given expansion, rather than a unique assignment. This set is not entered manually, rather it is generated automatically given the set of items that is to be abbreviated.

An abbreviation strategy directs the generation of the set of abbreviations associated with a vocabulary. The strategy consists of a set of abbreviation rules that are partitioned into subsets related to word length. That is, for a word of a specific length, there is a set of rules to be used in producing a set of abbreviations for that word. This set is ordered according to the user's preference in applying the rules. This arrangement is based on the observation that preferences vary among individuals based on word length [S91].

An abbreviation rule is a formal description of the relationship between an expansion and an abbreviation [S91]. A simple example is truncation, where the abbreviation is some number of the leftmost letters from the word. Another example is salient-letter, the first letter of each syllable in a multiple-syllable word.

Once the set of abbreviations is generated, they are entered into an abbreviation-expansion table. Each row of this table represents all of the expansions, for any particular abbreviation. These expansions are ordered by the preference for the rule used to produce the abbreviation. That is, the most preferred expansion is the one for which the abbreviation was produced by the most preferred rule. When two expansions have the same preference, they are arranged by increasing length. This represents the preference of having shorter abbreviations associated with shorter expansions.

The entire set of items that the user wishes to abbreviate is divided into separate vocabularies, where each is typically a set of perhaps as many two hundred related words. The user can then select a small number of these vocabularies as being active at any given time. This allows the expansion to focus on the particular subject about which the user is writing, rather than the as many as twenty thousand words that the user might know in general. Each such vocabulary has its own table. This is done for the convenience of the user and makes the process of expansion more efficient.

The user has a choice in how abbreviations are expanded. With one method the abbreviation is replaced with its most likely expansion. Should this not be the one that the user intended, they press the Space Bar (or any designated <abbreviate>key) again, and the next most likely expansion appears. The user repeats this, cycling through the entire list at will, until the desired expansion appears, at which point they resume typing with a non-<abbreviate> key. With the second method, should the user reject the most likely expansion, a list is presented from which the user can select the desired expansion directly. These expansion methods are actually not distinct: the user can press the Space Bar to scroll through the list, and can directly select the expansion in the first method. The difference between the two is essentially whether or not the list appears when the most likely expansion is rejected. This is provided as a feature for those users to whom the seeing the list is not of benefit.

One important observation about flexible expansion is that it does not preclude the traditional fixed expansion. The table produced for any vocabulary can be edited to add, change, or remove both abbreviations and expansions. In this way fixed abbreviation assignments can still be used by simply having only one possible expansion for an abbreviation. Thus the expander can provide the benefits of fixed expansion for the most frequently used items.

Implications

By removing the uniqueness constraint on the abbreviation-expansion relationship, the user no longer has specific assignments to memorize. Rather, the user can simply construct some reasonable abbreviation for the expansion item they wish to enter. With a suitable defined strategy the expander will have anticipated this abbreviation and can present the desired expansion. In the worst case the desired expansion would be one of the alternative expansions the system identified.

Using the strategy the system can tailor the set of abbreviations for a given set of items to any particular user. More importantly this set of abbreviations is automatically generated and does not have to be manually entered. Thus abbreviation expansion is made available across the user's vocabulary in a practical manner. Being independent of specific application, flexible abbreviation expansion gives the user consistent and transparent access across all applications. The user can make use of the same set of abbreviations in the same manner from within any application and gain the same benefit.

Discussion

A flexible abbreviation expander is being developed and tested for the Microsoft Windows environment. This typing aid gives users easy and consistent access to a wide variety of software ranging from popular off-the-shelf word-processing to specialized speech communication. In addition, this is one more tool in making workplace accommodation more complete.

Future enhancements of Flexible Abbreviation Expansion include incorporating adaptiveness to a particular user. Generally this means fine-tuning the operation of the = expander based on the selection patterns of the user. For example, with a given abbreviation the user may actually prefer an expansion that the strategy indicates as being the third most preferred. An adaptive expander would recognize this and override the strategy and reflect the user's preference with this abbreviation. In another case, the system could also observe more wide-scale tendencies and actually modify the Strategy itself. Of course, some users will be more sensitive to this fine-tuning than others. So the use and extent of adaptiveness would be entirely at the discretion and desire of each individual.

Acknowledgments

This work is supported by SBIR Grant Number RN91076101 from the Department of Education to GMS Systems for the development of the Windows application. Additional support is provided by Grant Number H133E80015 from the National Institute on Disability and Rehabilitation Research, and by the Nemours Foundation

References

[DLM89] Demasco, P., Lillard, M., and McCoy, K. (1989). Word Companion: allowing dynamic word abbreviations. *Proceedings of the RESNA 12th Annual Conference*,282- 283.

[KV82] Kelso, D.P., and Vanderheiden, G.C. (1982). Ten-Branch Abbreviation Expansion for Greater Efficiency in Augmentative Communication Systems. {RESNA}*5th Annual Conference on Rehabilitation Engineering*, 3.

[S91] Stum, G.M. (1991). Automatic Abbreviation Generation. Master’ s Thesis, U of Delaware.

[SDM91] Stum, GM., Demasco, P.W., and McCoy, K.F. (1991). Automatic Abbreviation Generation. *Proceedings of the RESNA 14th Annual Conference*, 97-99.

[VK87] Vanderheiden, G.C., and Kelso, D.P. (1987). Comparative Analysis of Fixed- Vocabulary Acceleration Techniques. AAC Augmentative and Alternative Communication, vol.3, no. 4, 196-206.

[R91] Ray, S. (1991). The Function of Function Words and Function Units in Augmentative Communication. *Proceedings of the RESNA 14th Annual Conference*, 94-96.

Contact

Gregg M. Stum
Applied Science and Engineering Laboratories
A. I. duPont Institute
PO Box 269 Wilmington, DE 19899
Email: stum@asel.udel.edu