GOODSPELL

Written by: Henry G. Baker, Ph.D.

In conjunction with Apple Computer Inc.

NOTICE

Apple Computer Inc. reserves the right to make improvements in the product described in this manual at any time and without notice.

DISCLAIMER OF ALL WARRANTIES AND LIABILITY

APPLE COMPUTER INC. AND SYNAPSE COMPUTER SERVICES MAKE NO WARRANTIES, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS MANUAL OR WITH RESPECT TO THE SOFTWARE DESCRIBED IN THIS MANUAL, ITS QUALITY, PERFORMANCE, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. APPLE COMPUTER INC. SOFTWARE IS SOLD OR LICENSED "AS IS." THE ENTIRE RISK AS TO ITS QUALITY AND PERFORMANCE IS WITH THE BUYER. SHOULD THE PROGRAMS PROVE DEFECTIVE FOLLOWING THEIR PURCHASE, THE BUYER (AND NOT APPLE COMPUTER INC., SYNAPSE COMPUTER SERVICES, THEIR DISTRIBUTOR, OR THEIR RETAILER) ASSUMES THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR, OR CORRECTION, AND ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES. IN NO EVENT WILL APPLE COMPUTER INC. OR SYNAPSE COMPUTER SERVICES BE LIABLE FOR DIRECT, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT IN THE SOFTWARE, EVEN IF APPLE COMPUTER INC. OR SYNAPSE COMPUTER SERVICES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010

Goodspell—A Spelling Checker for the Apple

Henry G. Baker, Jr., Ph.D.

Synapse Computer Services

You do a lot of word processing on your Apple. You don't spell or type as well as you would like. You want to avoid the embarrassment of misspelled letters and documents.

Goodspell is meant for you.

Goodspell quickly checks the spelling of your letters and documents quickly and gives you a log of possible misspellings. Its tireless eagle eyes perform more quickly and efficiently than a copy editor. Yet Goodspell won't put a hex on your wallet.

You will be able type more quickly, secure in the knowledge that most simple typographical errors will be caught by Goodspell.

Where Goodspell came from

Spell checking programs have been used for several years on large time-shar [illegible] to help find spelling errors in papers, manuals, [illegible] popular for word processing [illegible] spell-checking capability was

However, the slow speed and small size of the microprocessor did not seem to be compatible with the techniques used on the time-shared mainframe spell-checkers. For example, the DEC-20/60 is probably 100 times faster and has an address space 18 times larger than the 1 MHz 6502 used in the Apple II.

Nevertheless, Goodspell is a spell-checking program which runs on a 48K Apple II, has a 14,000 word dictionary, and can check files for spelling errors faster than 3000 words per minute (6 typewritten pages/minute). This is faster for the user than a time-shared spelling checker, and makes spell-checking very attractive on this small machine. (IBM claims only 1000 words per minute with their Displaywriter, which uses an 8086).

The Heuristic Spell-checking Principle

Most spell-checking programs work on the same principle--look

every word up in a dictionary, and if it is not there, assume that the word is misspelled and ask the user if it is correct. Since most words in a user's text are spelled correctly and are common words, they will be found in the dictionary. Therefore, only a handful of interactions are required even for a large text file. Since the computer can look up words faster than a human, and does not make mistakes, the spell-checker greatly improves the productivity and accuracy of copy-editing.

This simple spell-checking principle has many advantages. Its simple operation can be explained in a matter of minutes, and can be understood, unlike more complex, context-sensitive spell-checking methods. Thus, a non-computer scientist can easily grasp the spell-checker's features and drawbacks. For instance, a word in a text may be misspelled, but because it is in the dictionary--i.e. it is a correct word in some context--the spell-checker will assume that it is correct. However, even though this method will not catch all spelling errors, it still catches the vast majority of simple typing and spelling errors, leaving it to you for few obscure words in your text.

How to Operate the Goodspell Spell-Checker

Goodspell requires a 48K Apple II or Apple II+, a disk drive, a printer of some sort, and a text file created by the "Applewriter" text editor and justifier system.

The Goodspell program and dictionary is first loaded from the disk, and then the Applewriter text file is checked for spelling errors. Any suspected spelling errors are presented to the operator, together with the surrounding context, so that the operator can determine whether the word is spelled correctly or incorrectly. If the word is spelled incorrectly, it is logged on the printer for later correction. If the word is spelled correctly, it is entered into an "incremental dictionary" which lasts for the duration of this document only. Words in the incremental dictionary are considered to be spelled correctly, and do not cause additional interactions. Finally, multiple files can be quickly checked without reloading the dictionary.

The Goodspell Spell-Checker is extremely easy to operate.

1. Insert the Goodspell disk into drive 1 of a 48K Apple II or Apple II+ system. Boot this disk by turning the Apple off and then on (assuming an "autostart ROM").

2. Goodspell will ask you whether its printer address or slot number are correct. If they are, simply hit "RETURN". If your

printer slot or printer driver address is not correct, enter "N" or "NO", then "RETURN". Goodspell will then ask you for your printer slot or driver address in decimal. Enter either an integer slot number 1-7 or an address 768-65525. Goodspell will then remember this address and ask you if it is correct. If so, hit "RETURN" and continue. If not, hit "N" then "RETURN" and retry.

(To write printer drivers, see the appropriate appendix.)

(Note for Apple "COM CARD" users. The Apple COM CARD sometimes does not initialize itself properly when Goodspell tries to log the file name, and causes the Apple to either go into an infinite loop or "go off the deep end" and do strange things. Turn the Apple off and try booting again. If this does not work, hit "RESET" when asked for the printer address, and boot the disk using "PR#s", where "s" is the boot disk slot number. If this also fails, you must write a small printer driver program to properly initialize the COM CARD, and thereafter it should work without hassle.)

3. Goodspell will then print a copyright banner, and inform you that it is loading in the dictionary. This loading takes about 20 seconds.

4. Goodspell will then ask for the name of a file to check. At this time, replace the Goodspell disk with a disk on which the Applewriter file(s) you wish checked reside. Type the name of the Applewriter file (without the "TEXT."), and finish with a "RETURN". Goodspell will then proceed checking your Applewriter text file.

If your file is on a different disk drive or in a different slot number from the boot drive, simply append ",Dn" or ",Sn" (or both) to the file name, where "n" is the drive or slot number. (Corvus users note that ",Vnnn" is acceptable also). (Currently, "S$n", "D$n" and "V$n" are not acceptable.)

5. If you do not recall the exact name of the Applewriter file, simply respond to Goodspell with a "return". This will cause the "catalog" of the current disk to be shown, so that the proper file name can be ascertained.

6. Goodspell will go through your file sequentially, checking every word in its dictionary. When it comes across a word which is not in its dictionary, it will print 64 characters of context on the screen, followed by a line

'your word' NOT FOUND. ENTER (Y/N)?

If you answer "Y" (or just carriage return), Goodspell will assume that the word was a jargon word that was spelled correctly, and put the word into its "incremental" dictionary. This dictionary is an extension of Goodspell's builtin dictionary, and holds words peculiar to the current file being checked.

If you answer "N", Goodspell will assume that the word was a misspelling and log it on the printer (in upper and lower case to ease searching for it) so that it can be later corrected. It does not enter that word into the incremental dictionary.
(Some printer drivers will copy what is printed to the apple screen. However, since Goodspell logs words in both upper and lower case, the word may appear as garbage when printed on the Apple screen. this is due to the Apple's version of ASCII for its character generator. This garbage on the screen does not affect the proper operation of Goodspell.)

7. When the file has been completely checked, Goodspell will query the user for another filename to check.

8. To stop Goodspell, you must press RESET.

The Spell-Checking Dictionary

The spell-checking dictionary is the key to an efficient spell-checking program. The larger it is, the fewer interactions the program will have to have with the secretary. The faster it is, the more likely the secretary is to use it on every file.

How large should the dictionary be for efficient operation? Common sense would dictate that the dictionary be as large as possible. Common sense is almost correct, because a larger dictionary will be able to incorporate more and more jargon words such as legal or medical terminology. But common sense is also wrong in that the larger the dictionary, the more likely it is that a misspelled word will be found and considered correct.

Many time-shared spell-checking programs operate with dictionaries in the 30,000 to 60,000 word range. They tend to accumulate a large amount of jargon because a time-shared system must cater to a large variety of customers. However, most people's working vocabularies seldom exceed 15,000 words, hence a more personalized dictionary could be smaller and still be as effective as a 60,000 word dictionary that was not tuned to that person's needs.

Theoretical Results

"Zipf's law" tells us that the kth most probable word in a person's vocabulary is proportional to 1/k. That is, the most probable word in a person's vocabular is used 15,000 times as much as the least probable. Although Zipf's law is only approximately true, it can give us valuable advice on how to devise cost-effective spell-checking programs.

For a distribution that follows Zipf's law, the "information content" H is proportional to the logarithm of the number of words in the dictionary. Thus, the information content is only increased by 14% in going from 15,000 words to 60,000 words! Thus, a 15,000 word spell-checking dictionary can get 87.5% of the benefit of a 60,000 word dictionary at only 25% of the cost.

A secretary's time in checking the spelling of a document is the sum of the time it takes the computer to look up the words that it finds, plus the secretary's time to correct the words that are not found. Theory suggests that a 15,000 word dictionary will cause less than twice as many "correct, but not in dictionary" faults as a 60,000 word dictionary. Thus, if the computer time to look up words in the smaller dictionary is the same as the time to look up the words in the larger dictionary, we have produced a dictionary with better than 50% of the performance at only 25% of the cost.

If the computer dictionary lookup time for the smaller dictionary is much faster, it will be able to match the secretary's throughput on spell-checking with the larger dictionary, yet cost only 25% as much. This is what Goodspell has accomplished.

Dictionary Encoding

How much room does it take to store 15,000 words? Common sense tells us that the average word is 5 characters long. Therefore, with an added space to separate words, it should require 90,000 bytes to store the 15,000 words.

In actuality, the words in Goodspell's dictionary average 7.8 letters. (This is due to the fact that the 5 letter average is weighted by frequency of occurrence, while the 7.8 letter average is not. Hence longer words are correspondingly more frequent in a dictionary.) Using 8.8 characters per word stored would require 132,000 characters!

(We have ignored in this discussion how to encode the dictionary to also allow for efficient search).

We don't have room for 132,000 or even 90,000 characters on the Apple, so we had to do something else. Since there are only 26 letters and one separator, we could conceivably encode each character in 5 bits. This encoding would then allow the storage of our 15,000 words in only 82,500 bytes--certainly an improvement, but not nearly enough, because the Apple only has 49152 bytes of RAM, of which only about 36,000 bytes can actually be used for a program.

More complex codes would help somewhat. For example, a binary variable-length code based on character frequency would be able to encode our dictionary in close to 4 bits per character. Using this encoding, our dictionary would then require only 66,000 bytes--a savings of 22,500, but still not compact enough to fit in this small microcomputer.

We are led to consider how good the most compact encoding for our dictionary would be. It has been estimated that the information content of a character in English is only about 2 bits. Two bits per character would let us store our 15,000 words in 33,000 bytes, which would finally let us achieve our goal. However, achieving the 2 bit/character goal requires multiple character encoding, in which a single code word encodes a multi-letter sequence. A program for this encoding and decoding scheme would be complex and hopelessly slow.

The Goodspell Solution

The Goodspell dictionary fits 14,000 words in 33,690 bytes, thus achieving a density of 2.19 bits per character of dictionary--very close to our estimate of 2 bits per character for the minimum possible encoding of English.

Yet Goodspell checks files at up to 300 bytes per second--equivalent to 3000 words per minute (6 typewritten pages). Since most documents generated by word processing systems are 3 pages or less, they can be checked for correct spelling in less than one minute!

(This is to be compared with IBM's Displaywriter, which checks spelling at only 100 characters per second).

Goodspell is the solution to performance and quality in your office. Cn yu affurd to bee witot it?

Appendix A

Choice of words in main dictionary.

The 13,000 words in the main dictionary were chosen arbitrarily by Synapse to be a reasonable compromise between words which occur commonly, and "generally useful" words such as those which occur in this manual.

Thus, words such as "justifier" and "lookup" which are obviously jargon have been left out in favor of general purpose words such as "pursue", "questionable", etc.

To give you some idea as to the depth o the dictionary, Goodspell has been run on this manual itself, and the following words were not found in its main dictionary:

Goodspell Henry Bandley Cupertino misspellings eagle wallet typographical mainframe DEC MHz typewritten IBM Displaywriter Applewriter justifier autostart retry COM booting PR Dn Sn Corvus Vnnn jargon bastard filename Zipf kth logarithm lookup throughput separator cn yu affurd bee witot CF nonstandard signalled isn Jack.

(This run took 1 minute, 3 seconds.)

All the rest of the words in this manual are in the main dictionary!

Appendix B

What is a word?

Goodspell starts at the beginning of a file and reads every character in the file. The first alphabetic character (a-z or A-Z) is considered the start of the first word. The end of the first word is signalled by the first non-alphabetic character. Then non-alphabetic characters are skipped until the next alphabetic character is found. This is the beginning of the second word. Goodspell proceeds in this manner until the end of the file has been reached.

Thus, "isn't", "I've" and other contractions are considered to be two separate words for Goodspell's purposes.

(Since "case" (upper or lower case; i.e. "b" vs. "B") is ignored in looking up words in Goodspell's dictionary, Goodspell cannot check whether proper nouns such as "Jack" are always capitalized.)

Appendix C

Writing Printer Drivers

There is space between $300-$3cf allocated for a printer driver routine for driving nonstandard printers from Goodspell. This routine is called in the same way that Applewriter printer drivers are called. If by any chance your Applewriter printer driver does not work with Goodspell, please contact Synapse at the given address.

Appendix D

Setting Up The APPLE II System

This appendix includes a list of the equipment you will need to use the Goodspell Program on your Apple II. You do not need to read all the manuals but they should be on hand to answer questions that may arise in operating the equipment (eg. how to boot the diskette).

The Goodspell program is written in Applesoft BASIC and assembly language. To use it you will need the following equipment.

-an Apple II Plus with 48K bytes of RAM; or

-an Apple II with 48K bytes RAM and an Applesoft firmware card; or

-an Apple II with the Language System.

-an Applewriter program and text files.

Plus:

-an Apple Disk II with Controller (16 sector PROMs);

-a video monitor or television.

For reference you should have on hand a copy of the following manuals:

-This Manual;

-Apple II BASIC Programming Manual (Setting up the Apple II);

-DOS Manual (How to boot the diskettes).

SPECIAL NOTE

If you have purchased an Applewriter program and are operating with a 13- sector disk drive you will have to update your Apple Disk II controller cards to a 16-sector drive in order to use Goodspell.

You can update your Applewriter program disk and text files to 16-sectors by using the "MUFFIN" program on the DOS 3.3 Master diskette which is included in the DOS Update kit. Instructions for using the MUFFIN program can be found in the DOS 3.3 Users Manual which is also included in the update kit.