An Empirical Study of UNIX Command Abbreviation Schemes

Murthi Nanja and Curtis Cook
Department of Computer Science
Oregon State University
Corvallis, Oregon 97331

85-60-4

AN EMPIRICAL STUDY OF UNIX COMMAND ABBREVIATION SCHEMES

Murthi Nanja

Curtis Cook

Computer Science Department

Oregon State University

Corvallis, Oregon 97331

ABSTRACT

This study examined the **UNIX** command abbreviation schemes preferred by expert and novice UNIX programmers. The two parts of the conducted experiment were: subjective rating of UNIX command abbreviations for each of the six abbreviation categories (*acronym, combination, contraction, identity, synonym,* and *truncation*); subjects suggested descriptive command names for UNIX commands in each of the six abbreviation categories. The results suggest that experts rate UNIX command name abbreviations higher than novices and that experts and novices prefer different abbreviation schemes.

I. INTRODUCTION

UNIX is a relatively small, but elegant, flexible and powerful operating system. It is a complete programming system with many software tools for program development and software applications. It is widely used on minicomputers and seems destined to become a standard for microcomputers.

However, Norman[8] and others have criticized the cryptic and inconsistent UNIX command names that make it unfriendly, especially for casual users. In one study, Deleon, Harris, and Evens[3] showed that for a set of twelve UNIX commands, novice users made fewer errors using the UNIX command names than English names. But the UNIX name group used the on-line assistance nearly twice as much as the English name group.

An important aspect of human-computer communication is the ease with which the user inputs the desired commands. To allow for flexibility, the users should be able to enter commands in full or to abbreviate as desired. Therefore the main purpose for having abbreviations is to facilitate data entry or to reduce message length or to reduce entry errors. The three most common methods for creating abbreviations are truncation, contraction, and acronym. In truncation, the last few letters of a word are deleted; contraction involves deleting some specified set of letters (usually vowels) starting at the right end of the word; and the acronym scheme uses the first letter of each word of the command description. In addition abbreviations can be either fixed or variable length.

These abbreviation schemes have been compared in various studies [1,4,5,6,7]. Nawrocki [7] examined the errors generated when subjects interpreted abbreviations of single word constructed by truncation and contraction methods. He found no difference in the number of correct interpretations of word abbreviations for truncation method or the contraction method. But the interpretation of contraction led to more spelling errors and truncation produced more errors related to the endings or the tense of the words. Hodge and Pennington [5] empirically evaluated three abbreviation methods: truncation, contraction, and a combination of the two. They found that the contraction approach was preferred for short words, but truncation was used more frequently

for longer words. Ehrenreich and Porcu [4] conducted a series of rating, encoding, and decoding experiments which compared the benefits of truncation versus contraction and fixed versus variable length abbreviations. They found truncation to be the better abbreviation technique and no difference between fixed and variable length abbreviations.

In this paper we investigated the various abbreviation schemes used for UNIX command names. Chapter 2 gives an analysis of the lexical characteristics of UNIX command names, defines the six abbreviation scheme categories, and gives the distribution of UNIX command names for these categories. Chapter 3 describes the experiment in which UNIX experts and novices were given UNIX commands in each of the six abbreviation categories and their function descriptions. The subjects were required to rate the name according to its ease of use and recall and to suggest a preferred name for the command. An analysis of the results suggests that experts prefer the UNIX abbreviation more than novices and that experts and novices prefer different abbreviation schemes.

II. LEXICAL CHARACTERISTICS OF UNIX COMMAND NAMES

Lexical studies of any language such as natural language, programming language, and command language are based upon the symbols used. Common lexical measures are word counts, word frequency distributions, word length distributions, and character frequency distributions. In the following subsections, some of the lexical properties of UNIX are described.

For the purpose of this study 2.9BSD UNIX was used. It was assumed that, since different versions of UNIX are quite similar, that the choice of a particular version would have little effect upon the results. Only the command names and their corresponding function names listed in Section 1 in the UNIX Programmer's Manual [2] were used for the analysis. Commands listed in other sections of the manual were omitted from the analysis. The lexical properties considered include number of commands, length of command names, command name abbreviation schemes used, and frequency distribution of commands for abbreviation categories.

2.1. Number and Length of Command Names

The number and length of command names and related statistics are shown in Table 1.

Table 1. Command Length Statistics

Total number of commands	=	203
Shortest length of command	=	1
Longest length of command	=	10
Mean command length	=	4.3

The distribution of command length is shown in figure 1.

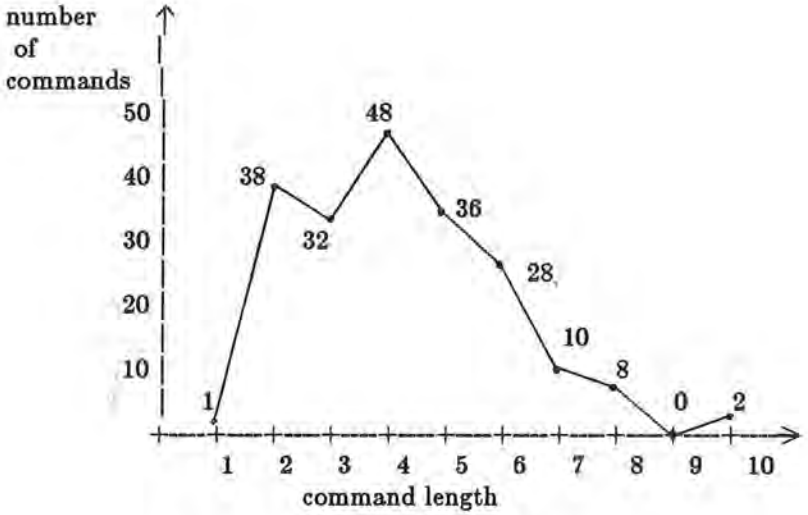

Figure 1. Command Length Distribution in UNIX

2.2. Abbreviation Schemes in UNIX

This section examines the different abbreviation schemes employed in forming UNIX command names. Most UNIX command names were derived in some way from the English word or phrase describing its function. No consistent abbreviation rule was used in forming UNIX command names.

We classified each UNIX command into one of seven abbreviation scheme categories based on its description. The categories are *acronym, identity, truncation, contraction, synonyms, combination scheme*, and *miscellaneous.*

The first category, acronym, is common in programming. Acronyms are formed from the concatenation of the first letter of the words in the description. People often use this type of abbreviation in every day life. Examples are: A.M. for 'ante meridiem', TBA for 'to be announced', TA for 'teaching assistant', MVP for 'most valuable player', etc. UNIX commands such as *cc, pwd, ps, su, du, od, wc, yacc, cb,* and *cd* fall under this category. For example, command names *pwd,* and *du* stand for 'print working directory', 'disk usage' respectively.

The identity category are those command names that are identical to its single English word function description. These command names are meaningful English words such as *echo, mail, sort, size, time, users, split, clear, count, fold,* and *rewind.*

The truncation category command names are formed by deleting the last few characters of its single English word function description. The UNIX command names, *ed, as, man* are derived from English words editor, assembler, manual respectively.

In contraction abbreviations, some specified set of letters (usually vowels) are deleted starting at the right end of a word and progressing to the left. However, the first letter of word is never deleted. Again, these abbreviations use a single English word chosen from the function description. Examples of UNIX command names formed by contraction technique are: *cmp* for compare, *rm* for remove, *mv* for move, *passwd* for password, *fmt* for format, etc.

The synonym category is similar to the identity category in the sense that both techniques use a single English word without any modification. However, in the case of synonyms, the word selected as command name is a synonym for the function description. Many UNIX command names viz. *apropos, sleep, clock, compact, crypt, look, units* were derived using this method.

The combination scheme is based on two or more English words from its function description; each word in the description is abbreviated using one of the abbreviation methods and then each abbreviated words concatenated. Combinations such as contraction followed by contraction, contraction followed by truncation, and identity followed by truncation are common in UNIX command names. Examples in this category are *chfn* (for change function), *chmod* (for change mode), *printenv* (for print environment), and *mkdir* (for make directory).

All command names whose derivations are unknown or do not fit into one of the previous categories were grouped into a miscellaneous category. The command names in this category include *tee, nice, troff, nroff,* and *awk.*

It is also possible to group these abbreviation categories into two different classifications based on whether command names are derived from one word or from a phrase. For this classification, the number of commands, and the percentage of commands in each abbreviation category are shown in Table 2. Interestingly about one-third of UNIX command names are not abbreviated(identity or synonym).

Table 2. Distribution of Commands in Abbreviation Categories

	number of commands	%
Single word	78	38.4
identity	36	17.7
truncation	21	10.4
contraction	21	10.4
Phrases	93	45.8
acronym	18	8.9
synonym	34	16.7
combination	41	20.2
Others		
miscellaneous	32	15.8

III. SUBJECTIVE RATING AND ENCODING EXPERIMENT

The experiment was performed to investigate the abbreviation preference of both novice and expert UNIX programmers. The preference was based on:

1. Subjective rating of each of the six abbreviation schemes according to ease of use and recall.
2. The preferred abbreviation scheme when given the freedom to choose any scheme.

This study differs from previous studies in several aspects. First, in the earlier studies, researchers concentrated on the truncation, contraction, and combinations of truncation and contraction whereas our study dealt with these plus three additional abbreviation schemes. Second,

we investigated abbreviations derived from command function descriptions (usually several words) rather than from single English words. Finally the study used the UNIX command language.

3.1 Method

3.1.1 Subjects

The subjects formed two groups, novices and experts. The novice group consisted of 21 Oregon State University undergraduates, who had completed almost all introductory courses in computer science program and who were enrolled in senior level operating system course which introduced the UNIX operating system and operating system concepts. All novices had good knowledge of at least one programming language, and **NOS** command language for **CDC CYBER 170/720** computer. The experts group consisted of 16 Oregon State University graduate students who had been using the UNIX operating system for more than a year. All subjects were unpaid volunteers.

3.1.2. Design

A two factor mixed design experiment was conducted to study UNIX command abbreviation schemes. The level of expertise was used as "*between*" factor and abbreviation schemes were used as "*within*" factor for this experiment. In other words, six different abbreviation techniques were used to form six conditions for subjects at each level of expertise. The two independent variables were abbreviation scheme and skill level. The dependent variables were rating score and type of encoding schemes preferred.

3.1.3 Materials

A set of UNIX command names served as a source of stimuli for this experiment. All command names used for this experiment were selected from the 203 command names and function descriptions in the UNIX Programmer's Manual [2] (2.9BSD UNIX version). First, the seven different abbreviation schemes (acronym, identity, truncation, contraction, synonym, combination, and miscellaneous) discussed in Section 3.2 were employed here to partition all UNIX commands

into seven different groups. However, the miscellaneous group of command was omitted from the experiment. Then from each group, ten command names and their function descriptions were arbitrarily selected. For an example, the command names and function descriptions for the acronym technique is shown in Table 3. Each list of 10 commands was on a separate page.

Table 3. Acronym Group Commands

No.	Command name	Command description
1	*wc*	word count
2	*du*	summarize disk usage
3	*pi*	Pascal interpreter code translator
4	*pwd*	print working directory
5	*cd*	change working directory
6	*cc*	C compiler
7	*su*	substitute user id temporarily
8	*tod*	print out a message germane to the current time of day
9	*ps*	process status
10	*dc*	desk calculator

Alongside each command name in each group was a rating scale ranging from 1 to 5. The digit 5 was labeled "*very easy*" and the digit 1 was labeled "*very hard*". This scale was used for rating commands on their ease of use and remembering. The very first page in the material was used to collect background information for each subject and instructions to subjects were on the second page.

3.1.4. Procedure

Subjects within a given level of expertise were assigned to six different abbreviation schemes, one at a time, in different order. Given the command name and its function description, the subjects circled a number on the rating scale to indicate the ease of use and remembering the command name They also were instructed to suggest a descriptive name for the command that consisted of at most 10 lower case letters. Subjects were given enough time to complete the experiment.

3.1.5. Results

The rating task data for both novices and experts is shown in Table 4. This table shows mean ratings broken down by abbreviation categories : acronym, combination, contraction, identity, synonym, and truncation for both novices and experts. The scores are the sum of the ratings for the 10 commands.

Table 4. Rating Score vs Abbreviation Schemes

Abbreviation scheme		Novices	Experts
Acronym	Mean	32.7	36.7
	Std dev	6.26	6.08
	N	21	16
Combination	Mean	35.3	36.1
	Std dev	5.36	6.05
	N	21	16
Contraction	Mean	34.5	38.4
	Std dev	5.33	5.42
	N	21	16
Identity	Mean	41.4	42.6
	Std dev	5.64	6.13
	N	21	16
Synonym	Mean	35.6	35.9
	Std dev	5.21	5.19
	N	21	16
Truncation	Mean	34.7	37.8
	Std dev	5.18	6.75
	N	21	16
Overall Score	Mean	35.68	38.94
	Std dev	6.04	7.07
	N	21	16

A one way analysis of variance performed on the data showed that for both levels of expertise, the rating score was significant for all abbreviation schemes. i.e. For novices, $F(5,100) = 10.98$, $p < .001$, and for experts, $F(5,75) = 4.16$, $p < .001$.

It is interesting to note, from Table 2, that experts rated all abbreviation schemes higher than novices. This difference was quite large for the acronym, contraction, and truncation abbreviation techniques, whereas this difference in rating score is almost negligible for the other three abbreviation schemes. The overall rating score for all abbreviations for experts was significantly higher than that of novices (38.94 vs 35.68), $t(35) = 1.81$, $p < .05$. This seems to reflect the

experts' familiarity with UNIX.

The rank-ordering of all abbreviation schemes on rating score obtained by both novices and experts is shown in Table 5.

Table 5. Rank-ordering of Abbreviation Schemes

Rank–ordering	Novices	Experts
1	Identity	Identity
2	Synonym	Contraction
3	Combination	Truncation
4	Truncation	Acronym
5	Contraction	Combination
6	Acronym	Synonym

Multiple comparisons test using F test (Scheffé method) was performed for all abbreviation schemes. Based on this test, the identity category was preferred over all other types of abbreviation schemes for novices, $p < 0.01$ for all comparisons. Also, novices preferred combination, and synonym schemes over acronym abbreviation technique, $p < 0.05$. However, experts rated identity category names significantly higher than acronym ($p < .05$), combination ($p < .05$), and synonym ($p < .01$). No other comparisons were significant for both novices and experts.

The encoding task data for both novices and experts is shown in Table 6. This data was obtained by classifying subjects' preferred abbreviation schemes into one of the following six abbreviation technique: acronym, combination, contraction, identity, synonym, and truncation. Each row corresponds to the subjects' preference for one of the abbreviation schemes. The scores in a row are the frequencies of subjects' preference. For example, in row1, when 21 novices were given 210 acronym abbreviation command names, for 46 commands they retained original command name, for 102 commands they preferred combination scheme, for 40 commands they preferred identity scheme, and for the remaining 22 commands they preferred synonym, truncation, and miscellaneous schemes as shown in the table.

Table 6. Abbreviation Preference of Novices and Experts

Novices Group

	acro	comb	cont	iden	syno	trun	total
acro	46	102	0	40	6	10	204
comb	0	196	0	8	2	1	207
cont	0	33	52	98	4	19	206
iden	0	35	4	164	5	1	209
syno	0	41	0	24	140	2	207
trun	0	30	6	57	13	101	207
total	46	437	62	391	170	134	1240*

* 20 commands fall under miscellaneous category

Experts Group

	acro	comb	cont	iden	syno	trun	total
acro	62	52	0	36	10	0	160
comb	0	146	0	6	8	0	160
cont	0	30	64	42	16	8	160
iden	0	14	2	134	10	0	160
syno	0	16	0	14	128	2	160
trun	0	16	6	24	14	100	160
total	62	274	72	256	186	130	960

Legend :

acro	--->	acronym
comb	--->	combination
cont	--->	contraction
iden	--->	identity
syno	--->	synonym
trun	--->	truncation

In order to analyse this data, all preferred command name abbreviation schemes different from the given abbreviation scheme were added together to form a single group, and then a Chi-Square test was performed between the total number commands in the given abbreviation group and the total number of commands in the aggregated group. The Chi-Square values indicate that the subjects' preference for command names depends on the type of abbreviation method used. i.e. For novices, $\mathbf{X}^2 = 130.70$, $p < .001$, and for experts, $\mathbf{X}^2 = 161.10$, $p < .001$.

The dominance-ranking of abbreviation schemes based on subjects' preference of command names is shown in Table 7. This table also shows the distribution of UNIX command names in six

different abbreviation categories. It is interesting to note that the dominance-ranking order is same for both novices and experts, and UNIX command repertoire.

However, the percentage of command names in each abbreviation category is different for novices, experts, and UNIX ($X^2 = 18.8$, $p < 0.01$ between UNIX and experts, $X^2 = 32.8$, $p < 0.001$ between UNIX and experts). The dominance-ranking was not significant between novices and experts. It is clear from the table that novices and experts used combination, identity, and synonym schemes for 80% and 74% of the given UNIX command names whereas these abbreviation schemes account for only for 55% of the total UNIX commands. A reasonable explanation for this difference is that it might be easiest to generate meaningful command names using combination, identity, and synonym schemes. Also, novices preferred truncation, contraction, acronym schemes for only 19% of the commands whereas experts the same abbreviation techniques for 28% of the UNIX commands. This may be because these three abbreviation schemes generate cryptic command names. In addition, as we expected, command names suggested by novices were longer than that of experts (mean command length = 5.0 for experts, mean command length = 5.3 for novices).

Table 7. Dominance–Ranking of Preferred Abbreviations

Dominance–Ranking	Novices	Experts	UNIX
1	combination (35.2)	combination (28.5)	comination (20.2)
2	identity (31.0)	identity (25.5)	identity (17.7)
3	synonym (13.5)	synonym (19.4)	synonym (16.7)
4	truncation (10.6)	truncation (13.5)	truncation (10.4)
5	contraction (4.9)	contraction (7.5)	contraction (10.4)
6	acronym (3.7)	acronym (6.5)	acronym (8.9)
7	miscellaneous (1.4)	miscellaneous (0.0)	miscellaneous (15.8)

* Enclosed value is percentage of preferred command names in each abbreviation category. For UNIX this number is the percentage of commands in the abbreviated category.

IV. CONCLUSION

This paper has presented an initial study of UNIX command name abbreviation schemes. It reported the results of an experiment that attempted to determine the command name abbreviation scheme preference of expert and novice UNIX users. The following conclusions are based on the performance of the limited number of subjects (21 novices and 16 experts) who participated in the experiment.

1. With respect to ease of use and recall, experts rate UNIX command names higher than novices. This probably reflects the UNIX experience of the experts.

2. Both experts and novices rate the identity abbreviation scheme highest, but almost totally reverse the rank ordering of the other five abbreviation schemes. Novices rank the two meaningful word schemes (identity and synonym) first and second and combination third, while experts rate identity first, synonym last, and combination next to last. Novices seem to prefer schemes that provide the most help in recall. Experts seem to prefer more cryptic abbreviations because they need less of a cue in remembering a name. In fact out of 21 novices only two of them suggested shorter abbreviations for five command name descriptions.

Experts and novices disagreed on their rating of truncation and contraction, the most studied abbreviation schemes. The experts slightly preferred contraction over truncation which agrees with most published studies [4, 6].

3. For commands over all abbreviation categories, given the freedom to suggest a preferred name, both experts and novices had the same order of abbreviation preference. This order also coincided with the frequency distribution of the 203 UNIX commands in the abbreviation categories. Combination, identity, and synonym were the most preferred schemes. Since almost all function descriptions are several words, this seems to indicate that users prefer a name that either includes part of each word in the command function description or is a meaningful English word. The other three schemes (truncation, contraction, and acronym) are derived from a single word.

Norman [8] feels that a major problem with UNIX is the lack of a consistent abbreviation policy for command names. Developing such a policy appears to be a considerable problem, since

our results suggest that experts and novices prefer different abbreviation schemes.

The most basic conclusion from this study is that novices and experts prefer quite different UNIX command name abbreviations. Because of the limited number of subjects, this conclusion will warrant additional investigation. If this is indeed the case, then the next stage is to determine which scheme or schemes should be used in user interface design.

REFERENCES

[1] Barrett, J.A., and Grems, M. *Abbreviating words systematically.* Commun. ACM **3**, 15 (May 1960), 323-324.

[2] *Berkeley UNIX Programmer's Manual.* PDP-11 Version 2.9, 7th edition, Volume **1**, July 1983.

[3] De leon, L., Harris, W.G., and Evens, M. *Is there really trouble with UNIX?* CHI'83 Proceedings, December 1983, 125-129.

[4] Ehrenreich, S.L., and Porcu Theodora. *Abbreviations for automated systems : Teaching operators the rules. Human factors in computer systems.* Edited by Thomas, J.C., and Schneider, M.L. Ablex Publishing Corporation, 1984.

[5] Hodge, M.H., and Pennington, F.M. *Some studies of word abbreviation behavior.* J. Exper. Psych. **98**, 2 (May 1973), 350-361.

[6] Moses, F.L., and Potash, L.M. *Assessment of abbreviation methods for automated tactical systems.* U.S. Army Research Institute for the Behavioral and Social Sciences, Alexandria, VA, Technical Report **398** (NTIS: AD A077 840), August 1979.

[7] Nawrocki, L.H. *Word abbreviations in man-computer communication systems.* (ARI Working Paper MF 79-034). U.S. Army Research Institute, Alexandria, VA, 1979.

[8] Norman, D.A. *The trouble with UNIX.* Datamation, November 1981, 139-150.