PREFOR - The FORtran PRE-compiler for all source forms

Kalman Žiha
University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture
Ivana Luèiæa 5, 10000 Zagreb, Croatia
kalman.ziha@fsb.hr

Abstract: A pre-compiler for different levels of Fortran compilers such as 77 or 90 is provided in order to enable interchangeable source forms, such as fix form, tab form and free form in a condensed manner. The pre-compiler concept in general can support different programming facilities such as the source code statistics, the re-labelling or re-sequencing of statements, the embedded comments, the blank string compression, the programme documentation etc.

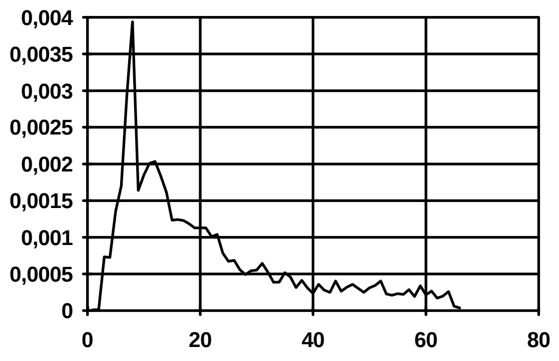
Keywords: Fortran, Fortran 77, Fortran 90, compilation, pre-compilation, source code.

1. Introduction

The FORTRAN programming language has a long and remarkable history. Many other programming languages have been developing in different application fields. Some of the languages offered various user-conveniences. Nevertheless, FORTRAN is still in wide use, especially by the scientific and engineering communities. Meanwhile, different innovations in the computational and data processing abilities of FORTRAN have been implemented. The improved computational efficiency and new functions of FORTRAN, contribute to its competitiveness against other languages. However, there were no changes in the basic FORTRAN source statement format until the implementation of the FORTRAN 90. The original fixed form of the FORTRAN, pertinent to earlier levels, with a single statement placed in a line, was quite practical in the era of punched cards and it remained unchanged till the implementation of the free input form in FORTRAN 90. The motivation behind the development of the free format FORTRAN statement is partly due to the inefficient usage of the FORTRAN line, and partly due to the observation, that a skilled FORTRAN programmer may find it quite impractical to start a new line for every new statement, even if it is quite a short one. The pattern for such a free form input had already been demonstrated in the programming language known as PL/I which is supported by IBM.
This paper present the FORtran PRE-compiler, the PREFOR, which provides for more comfortable Fortran programming at different levels. The relating standards for Fortran programming are American National Standard FORTRAN 90 (ANSI X3.198-1992) (same as International Standards Organisation ISO/IEC 1539:1991) and American National Standard FORTRAN 77 (ANSI X3.9-1978).

2. The usage of a line in FORTRAN programming

The usage of FORTRAN lines in different user-written FORTRAN programmes in engineering and scientific fields, has been analysed. Altogether 111166 unchanged original fix form FORTRAN lines of source programmes for optimisation, finite element method, different numerical models, mathematical and statistical libraries etc., have been considered. There were 17% labelled statements, 12% continuation lines and 28% comment lines. 1694167 characters were scanned in statement fields and 37657 characters in label fields. The results are presented in histograms (Fig. 1.), giving the number of characters in a 66 character long statement field and the number of digits in 5 character long label field. The results did not indicate much variation from programme to programme.



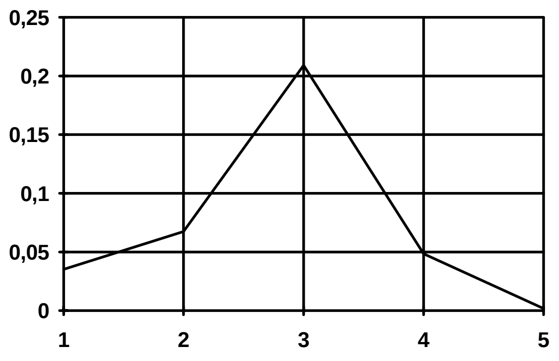


Fig. 1. Histograms representing the number of characters in statement field and in label field

The histogram (Fig.1.), indicates that the utilisation of the common FORTRAN line is quite poor. The average statement length is 21 characters, of 66 available positions, with a standard deviation of 15 characters (c.o.v.=0.71). The maximum frequency is detected for 8 character in a statement field exceeding the next highest frequency about twice. The average label length is 3 digits, out of 5 available positions, with a standard deviation of 1 digit (c.o.v.=0.33).

3. Basic conventions for writing programmes using PREFOR

According to FORTRAN 90, a program unit can be written either in fix form or in a free form, which is designated with, for example source program file name extension "f "and "for" for a fixed form or "f90" for a free form. Additional PREFOR input format features in the paper which do not necessarily comply with FORTRAN 90 rules, will be printed in *italics.*

Despite FORTRAN 90, the PREFOR source code is able to combine both free and fixed forms.

In a free source form statements are not limited to specific positions on a source line, and a line can contain from 0 to 132 characters. More than one statement (or a partial statement) can appear on a single source line if a statement separator is placed between the statements. For a free source form the general FORTRAN 90 conventions are applied. Blank characters are significant in a free source form. They must not appear as lexical tokens, except within a character context. Blank characters must be used to separate names, constants, or labels from adjacent keywords, names, constants or labels.
In fixed source forms there are restrictions on where a statement can appear within a line. A source line is in general 80 characters in length and has columns divided into fields for statement labels, continuation indicators, statement text and sequence numbers as follows:

- columns 1-5 are provided for the statement label,
- column 6 is reserved for the continuation mark which is optional,
- columns 7-72 are provided for the statement
- columns 73-80 are available for optional sequence number.

Source code usable for all forms is subject to the following restrictions:

- Blanks - treated as significant,
- Statement labels - place in column position 1 through 5,
- Statements - start in column position 7,

- Comment indicators - use only !, place anywhere except in column position 6,
- Continuation indicator - use only &, place in column position 73 of the initial line and in column 6 of each continuation line.

All source forms allow lowercase characters to be used as an alternative to uppercase characters.

A single PREFOR line can contain any number of statement fields, consisting of a statement label (optional, used when it is necessary) and the statement itself. A statement label must be one to five decimal digits long; blanks and leading zeros are ignored. PREFOR fields are delimited by PREFOR separators.

4. The use of the PREFOR

There are two ways of using the PREFOR pre-compiler:

1.1. The free form FORTRAN 90 source code can be subjected to pre-compilation.
1.2. The mixed form PREFOR source code can be subjected to pre-compilation.

The result of the pre-compilation in both cases is the FORTRAN symbolic code in fix form, following the basic FORTRAN conventions for fix forms, or following the restrictions for source code usable in all forms. In both cases, FORTRAN compilers at all levels are applicable to further compilation.

2. The existing FORTRAN programmes written in a fixed source form can be subjected to automatic condensation using PREFOR features in order to compress the fix form FORTRAN symbolic codes by the insertion of appropriate separators and by placing more than one statement in a line according to FORTRAN 90 free form conventions. The result of condensation is provided for the FORTRAN 90 compilation. It can also be pre-compiled in reverse, back to initial fix form, or to the source code usable to all forms.

In all cases, reference lists and source statistics can be made available and different programming utilities can be implemented.

PREFOR does not change the FORTRAN conventions. Any originally written FORTRAN code in a fix form remains unchanged after PREFOR pre-compilation. A free form source code complies with the FORTRAN 90 input conventions.

4.1. The PREFOR separators

The PREFOR uses four basic symbols: the blank, the statement separator, the continuation mark and the comment indicators, (unless they appear in a Holerith or character constant, or within a comment line). The PREFOR symbols can be set by the user. The default symbols are presented as follows.

The statement field separator which complies with FORTRAN 90 standard is semicolon (;). A semicolon delimits the statement field at its aft end. Consecutive semicolons (with or without intervening blanks) are considered to be one semicolon. If a semicolon is the last character on a line, or the last character before a comment, it is ignored. If there is no PREFOR continuation mark in the PREFOR line, the start of the line and the end of the line are considered as implied delimiters. A statement field can consist of the statement label and of the statement itself.

Additionally, PREFOR allows for the optional statement label separator colon (:) within the statement field not provided by FORTRAN 90, which delimits the statements label at its aft end from the statement itself, alternatively to the blank or string of blanks.

The ampersand character (&) indicates a continuation line (unless it appears in a Holerith or character constant, or within a comment line). The continuation line is the first noncomment line following the ampersand. If the first nonblank character on the next noncomment line is the ampersand, the statement continues with the character following the ampersand. If a lexical token must be continued, the first nonblank character on the next noncomment line must be the ampersand, followed immediately by the rest of the token. The ampersand within a line, if it is not the first or the last nonblank character in a line or in a comment, is ignored.
A comment indicator can precede the first statement of a program unit and appear anywhere within a program unit. All blank line is also a comment line. Comments have no effect on the interpretation of a program unit. A comment line is a line with any of the characters !, C or * at the beginning of the source line. Comment lines cannot be continued. If the comment indicator ! appears within a source line, the comment extends to the end of the line.

In addition, the embedded comments within a PREFOR line, not provided in FORTRAN 90, are delimited by ! in front of, and by ! or by the statement field separator ; behind the comment. Embedded comments can be placed anywhere within a PREFOR line.

4.2. The pre-compilation rules

- If there are no PREFOR symbols in a line, and if it is not a comment line, it is checked if it is a fix form FORTRAN line or a free form FORMAT line.
- The fix form FORTRAN lines remain unchanged under the PREFOR pre-compilation. The compression of blank strings can be performed on user's request.
- If PREFOR symbols are encountered in a line, the statements between the separators are unpacked and considered each as a free form source input of a statement field. If no special PREFOR features such as label separators or embedded comments are encountered, the line is treated as a FORTRAN 90 free form line.
- A free form statement field is first checked if there is a label, either by scanning for the label separator (;), or checking for a continuous string of up to five digits delimited by blank(s). If there is a statement label in the leading position, the next field is considered as the statement itself. If there is no label, the field is also considered as the statement itself.

The PREFOR pre-compilation consists in rewriting a mixed form of free form source file into a fix form source file, or into a source file with source code usable for all forms:

- The statement label encountered in a free form is placed in the first six columns of the FORTRAN line, starting with the first character of the label in the specified column of the FORTRAN line (from 1 to 5, default value is one, but it can be set by the user).
- The sixth column is reserved for an optional continuation mark. In a source code usable for all forms, the continuation mark is always ampersand (&) and has to be placed in column position 73 on the continued line and in column 6 of the continuation line. Each free form statement is placed in a single FORTRAN line, starting from the specified column position, (from 7 to 73, usually the seventh column if not set otherwise. The condensation of multiple embedded blanks down to only one blank can be activated on request.

- The embedded comment can be either ignored or placed at the user's request in a single FORTRAN comment line preceding the statement line, with the FORTRAN continuation mark "C" in the first column of the line.
- PREFOR can on request, automatically assign statement numbers placed in columns 73-80.

4.3. The PREFOR reference lists and source code statistics

The reference list relates the PREFOR line numbers with the FORTRAN line numbers in pre-compilation. The first number corresponds to the PREFOR line number within the PREFOR file. A colon is placed after the PREFOR line number and before the FORTRAN line specification. The line specification or a range of line numbers corresponding to the FORTRAN line within the FORTRAN file is obtained by pre-compilation.
The source code statistics can be applied to formal data such as the number of lines, the number of characters etc., as well as to logical data, e.g. number of loops, number of I/O statements and the appropriate I/O variables etc.

4.4. The setting of the PREFOR defaults

For the user's convenience, some of the PREFOR features can be set optionally.
The following default settings are provided:

:	the label field delimiter
;	the statement field delimiter
&	the continuation mark
!	the comment delimiter
132	the free form line length
80	the fix form line length
1	the starting column of a label (from 1 to 5) in a fix form line
7	the starting column of a statement in a fix form line
yes	condensation of embedded blanks within a statement
yes	ignore embedded comments in PREFOR line
no	add a line sequence number in columns 73-80.

4.5. The automatic condensation of existing ordinary FORTRAN code

The automatic condensation of the existing ordinary fix form FORTRAN source code to the PREFOR code should take place on request. The automatic condensation can be performed by using the PREFOR convention in reverse. This PREFOR feature is still under development.

5. Examples

Let us consider a simple FORTRAN program written in a fix form:
```
L     123456789.......  Columns
1           DIMENSION K(20)
2           DATA N/20/
3           DO  1 I =1,N
4     1     K(I)=I**2
5           END
```
The above FORTRAN program can be rewritten from the fix form into only one PREFOR line:

```
L      123456789......  Columns
1      DIMENSION K(20);DATA N/20/;DO 1 I=1,N;1:K(I)=I**2;END
```

The reference list for this example is quite simple: 1:1,5
Note that if the symbol : is replaced by blank, code complies with FORTRAN 90 free form.
The same program can be written also in a form whit the embedded comment, as shown:

```
L      123456789......  Columns
1      DIMENSION K(20);DATA N/20/ ! Organisational statements
2      DO 1 I =1,N;!Calculation!1:K(I)=I**2;END
```

The appropriate PREFOR reference list is as follows: 1: 1,2 2: 3,5

In the next example, the use of a continuation mark is illustrated:

```
L      123456789......  Columns
1      DIMENSION K(20);DATA N/20/
2      DO 1 I =1,N;!calculation!1:K(I)=&
3      &I**2;!Continuation!END
```

The appropriate PREFOR reference list is as shown: 1: 1,2 2: 3,4 3: 4,5

6. Conclusion

PREFOR has been developed earlier in a similar form as presented, for the author's own fun and personal use. The basic PREFOR pattern has followed the PL/I programming language source format. The PREFOR pre-compiler itself is prepared by using standard FORTRAN compilers of any level and their advanced string manipulation functions. The PREFOR is initially developed under UNIX operation system, but it can be easily implemented to any other platform.
PREFOR's pre-processing abilities provide an easy way to the use of the FORTRAN 90 advanced free form source input for the users of the lower level FORTRAN compilers based on a fix form source input. Alternatively, the pre-compilation concept can be applied in condensation of the fix form code to a FORTRAN 90 free form.
In addition, PREFOR can provide certain user's conveniences such as the mixture of different input forms in a single programme unit. The statement label separator, specific to PREFOR, potentially allows the application of symbolic labels instead of pure numeric labels. Fully embedded comments in a line can be attached to any statement field and can provide further conveniences of source code description. The pre-compilation concept can provide different statistical information on a formal and logical level of the source code as well as creating additional programme documentation.

References

1. DEC Fortran, Language Reference Manual, 1992,
   Order Number AA-PU45A-TK, Maynard, Massachusets.
2. DEC Fortran 90, Language Reference Manual, 1994,
   Order Number AA-Q66SA-TK, Maynard, Massachusets.
3. PL/I for OS/2 Marketing Broshure, IBM Form Number GC26-8154, 1994.