Comma Separated Values (CSV) Processing

Eugene Ying

2015-07-15

CSV History

- Started in 1967 when IBM OS/360 Fortran supported CSV in addition to fixed-column-aligned data (e.g. punched cards).

- It is now supported by almost all spreadsheets and database management systems.

A Non-APL Solution

Microsoft Excel is quite good at processing CSV file as its input. Thus many APL programmers use Microsoft Excel to import a CSV file, and then export it as an aligned text file for an APL function to process.

However, shouldn't APL, a more powerful tool, be able to process a CSV file directly and more efficiently?

The Big Hurdles

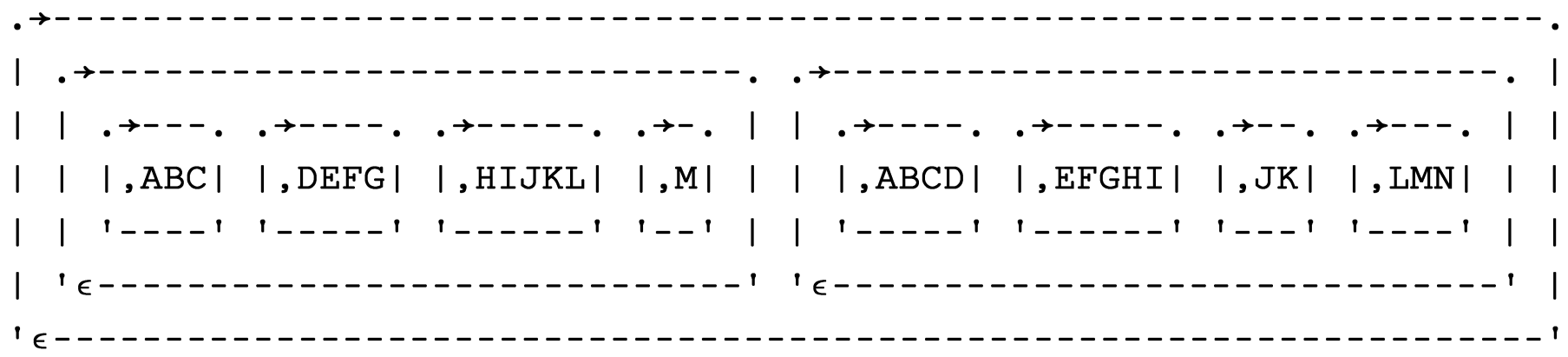
- Suppose we have a very large CSV file.
- We do not know what is the maximum number of fields in each record.
- And we do not know what is the maximum width of each field.
- Do we need to pre-process the CSV file (which can be quite time consuming) in order to find out these 2 unknown variables?

CSV Processing Part 1

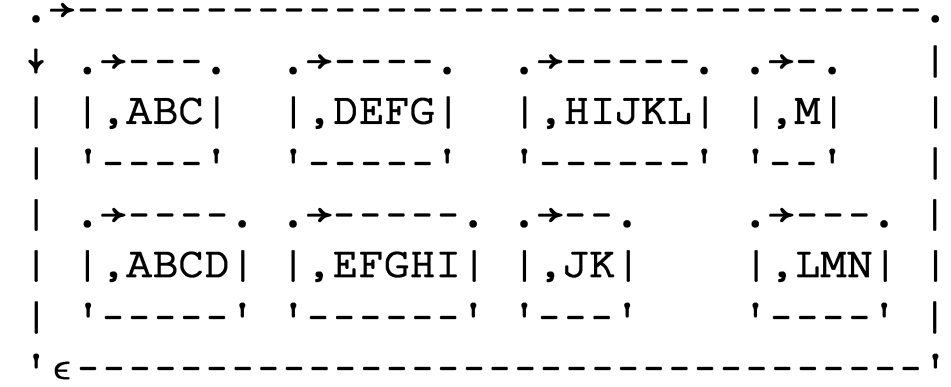
```
DISPLAY CSV
.→----------------.
↓ABC,DEFG,HIJKL,M |
|ABCD,EFGHI,JK,LMN|
'-----------------'
SEP←','
DISPLAY CSV1←(''⍴SEP),CSV
.→-----------------.
↓,ABC,DEFG,HIJKL,M |
|,ABCD,EFGHI,JK,LMN|
'------------------'
DISPLAY CSV2←{(+/∨\' '≠⍵)↑¨↓⍵}CSV1
.→-----------------------------------------.
| .→----------------. .→-----------------. |
| |,ABC,DEFG,HIJKL,M| |,ABCD,EFGHI,JK,LMN| |
| '-----------------' '------------------' |
'∊-----------------------------------------'
```

CSV Processing Part 2

```
DISPLAY CSV3←SEP{⎕ML←0 ⋄ ,(⍵∊¨⊂⍺)⊂¨⍵}CSV2
```

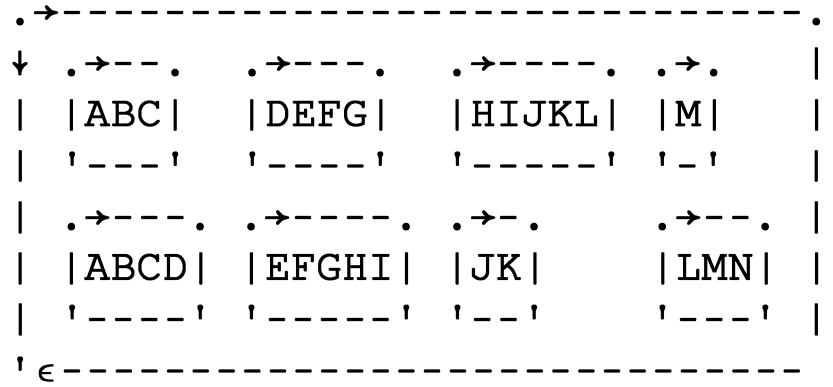



```
DISPLAY CSV4←{⎕ML←0 ⋄ ↑⍵}CSV3
```

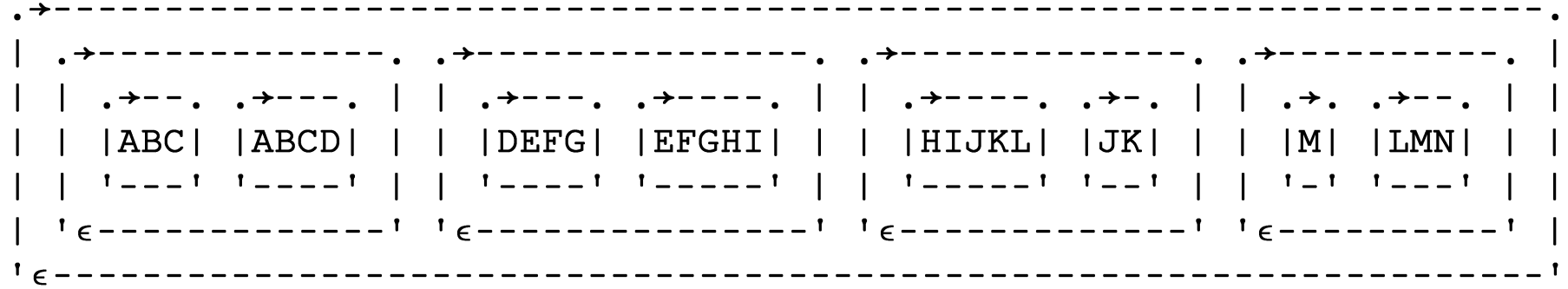


CSV Processing Part 3

```
DISPLAY CSV5←1↓¨CSV4
```




```
DISPLAY CSV6←⊂[1]CSV5
```



CSV Processing Part 4

```
DISPLAY CSV7←{⎕ML←0 ⋄ ↑¨⍵}CSV6
.→-----------------------------.
| .→---. .→----. .→----. .→--. |
| ↓ABC | ↓DEFG | ↓HIJKL| ↓M  | |
| |ABCD| |EFGHI| |JK   | |LMN| |
| '----' '-----' '-----' '---' |
'∈-----------------------------'
```

The 4 fields of the CSV matrix are now transformed into 4 items of the CSV7 nested vector. Each item is left justified and with width automatically determined.

CSV with Blanks as Separators

Many CSV files use blanks, instead of commas, as separators, and that can create a problem for the previous algorithm.

Problem of CSV with Blanks as Separators, Part 1

```
DISPLAY CSV
.→----------------.
↓ABC DEFG  HIJKL M|
|ABCD EFGHI JK LMN|
'-----------------'
SEP←' '
DISPLAY CSV1←(''⍴SEP),CSV
.→-----------------.
↓ ABC DEFG  HIJKL M|
| ABCD EFGHI JK LMN|
'------------------'
DISPLAY CSV2←{(+/∨\' '≠⊖⍵)↑¨↓⍵}CSV1
.→-----------------------------------------.
| .→-----------------. .→-----------------. |
| | ABC DEFG  HIJKL M| | ABCD EFGHI JK LMN| |
| '------------------' '------------------' |
'∈-----------------------------------------'
```

2 blanks

Problem of CSV with Blanks as Separators, Part 2

```
DISPLAY CSV3←SEP{⍵ML←0 ⋄ ,(⍵∊¨⊂⍺)⊂¨⍵}CSV2
.→-------------------------------------------------------------------------.
| .→-------------------------------. .→-----------------------------. |
| | .→---. .→----. .→. .→-----. .→-. | | .→----. .→-----. .→--. .→---. | |
| | | ABC| | DEFG| | | | HIJKL| | M| | | | ABCD| | EFGHI| | JK| | LMN| | |
| | '----' '-----' '-' '------' '--' | | '-----' '------' '---' '----' | |
| '∊-------------------------------' '∊-----------------------------' |
'∊-------------------------------------------------------------------------'
```

When blanks are used as separators, 2 consecutive blanks create an empty field, 3 consecutive blanks create 2 empty fields, and so on. The person who created the CSV file or the programmer who processes the CSV file may not be able to detect them visually. The output then becomes misaligned.

How to Skip Empty Fields

Is there a simple solution to skip processing empty fields?

The answer is Yes.
Use □ML←3 partition,
instead of □ML←0 partition.

CSV with Empty Field Skipped, Part 1

```
 DISPLAY CSV
.→----------------.
↓ABC DEFG  HIJKL M|
|ABCD EFGHI JK LMN|
'------------------'
SEP←' '
DISPLAY CSV2←{(+/∨\' '≠⊖⍵)↑¨↓⍵}CSV
.→---------------------------------------.
| .→----------------. .→----------------. |
| |ABC DEFG  HIJKL M| |ABCD EFGHI JK LMN| |
| '----------------' '----------------' |
'∊---------------------------------------'
DISPLAY CSV3←SEP{⎕ML←3 ⋄ ,(~¨⍵∊¨⊂⍺)⊂¨⍵}CSV2
.→-----------------------------------------------------.
| .→------------------------. .→------------------------. |
| | .→--. .→---. .→----. .→. | | .→---. .→----. .→-. .→--. | |
| | |ABC| |DEFG| |HIJKL| |M| | | |ABCD| |EFGHI| |JK| |LMN| | |
| | '---' '----' '-----' '-' | | '----' '-----' '--' '---' | |
| '∊------------------------' '∊------------------------' |
'∊-----------------------------------------------------'
```

CSV with Empty Field Skipped, Part 2

```
 DISPLAY CSV4←{⎕ML←0 ⋄ ↑⍵}CSV3
.→-----------------------------.
↓ .→--.  .→---.  .→----. .→.    |
| |ABC|  |DEFG|  |HIJKL| |M|    |
| '---'  '----'  '-----' '-'    |
| .→---. .→----. .→-.    .→--. |
| |ABCD| |EFGHI| |JK|    |LMN| |
| '----' '-----' '--'    '---' |
'∈-----------------------------'

DISPLAY CSV6←⊂[1]CSV4
.→-------------------------------------------------------------.
| .→-----------. .→-------------. .→-----------. .→----------. |
| | .→--. .→---. | | .→---. .→----. | | .→----. .→-. | | .→. .→--. | |
| | |ABC| |ABCD| | | |DEFG| |EFGHI| | | |HIJKL| |JK| | | |M| |LMN| | |
| | '---' '----' | | '----' '-----' | | '-----' '--' | | '-' '---' | |
| '∈-----------' '∈-------------' '∈-----------' '∈----------' |
'∈-------------------------------------------------------------'
```

CSV with Empty Field Skipped, Part 3

```
DISPLAY CSV7←{⎕ML←0 ⋄ ↑¨⍵}CSV6
.→-----------------------------.
| .→---. .→----. .→----. .→--. |
| ↓ABC | ↓DEFG | ↓HIJKL| ↓M  | |
| |ABCD| |EFGHI| |JK   | |LMN| |
| '----' '-----' '-----' '---' |
'∈-----------------------------'
```

The 4 fields of the CSV matrix are now transformed into 4 items of CSV7 nested vector. Each item is left justified and with width automatically determined. *All empty fields are discarded*.

Miscellaneous Details

If the separator is part of an input string, it should be preceded by an escape character.

Before you process a CSV file with escape characters, replace every escape + separator character by a special character that you never use. After you process the CSV file, replace the special characters by the separator characters.

Conclusion

To process a CSV file, you manipulate depth 1, depth 2, and depth 3 arrays.

If you want to keep the empty fields, use the ⎕ML←0 partition primitive function.

If you want to drop the empty fields (especially when blanks are used as separators), use the ⎕ML←3 partition primitive function.

By combining these 2 algorithms into a single function, you have a general purpose APL CSV processing function.

Your general purpose CSV function should specify the separator characters and escape character as part of the input argument.