AWKompiled

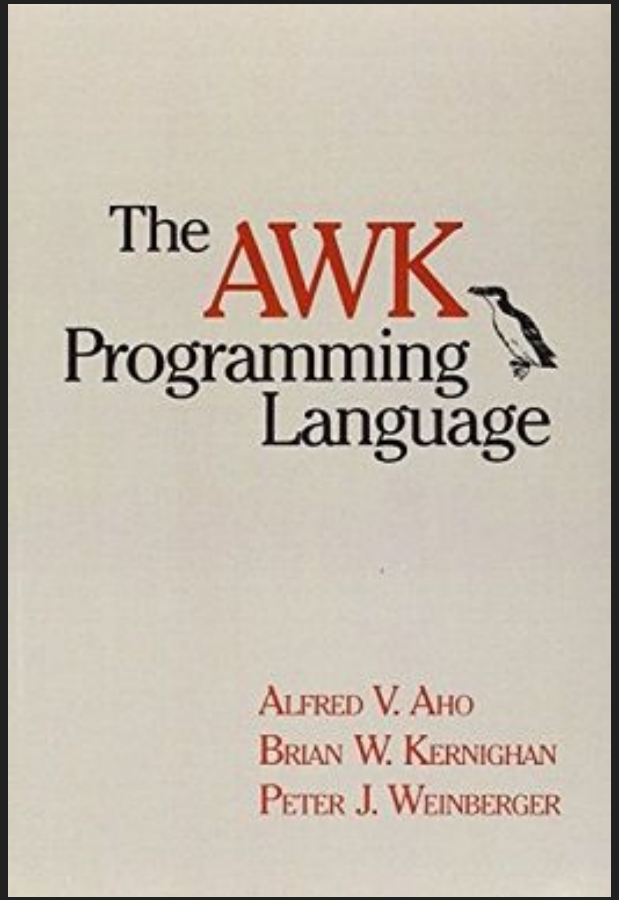
Native Floripa 2022

INTELITRADER

Alfred Aho

Peter **W**einberger

Brian **K**ernighan



```
pattern { action }
```

{ action }

pattern

[Output] Field Separator

Number of Fields

Number of Records

[Output] Record Separator

```
{
  if( cond ) {
    code;
  }
  for( start; step; cond ) {
    code
  }
}
```

> “Awk also maintains a delicate balance between being a line-oriented utility like grep and a full programming language.”
– Andy Oram

```
BEGIN { print "File\tOwner"}

{ print $9 "\t" $3}

END { print " - DONE -" }
```

```
#!/bin/sh
awk '
BEGIN { print "File\tOwner" }
{ print $9 "\t" $3}
END { print " - DONE -" }
'
```

How much is too much?

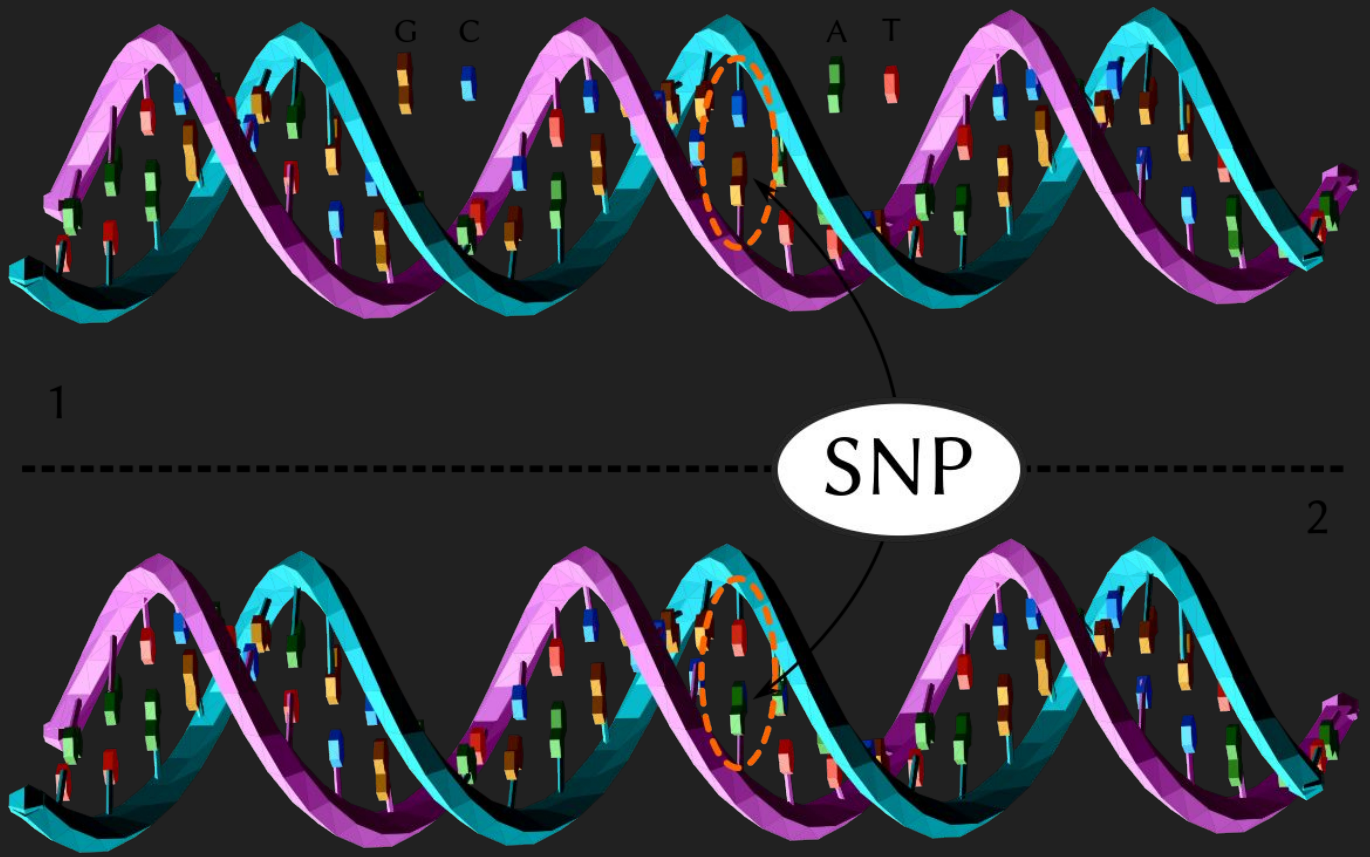
1 GB?

50 GB?

500 GB?

Using AWK and R to parse **25TB** of DNA – Nick Strayer

There were ~2.5 million SNPS and ~60 thousand people



Amazon Athena

```
select * from intensityData
limit 10;

select * from intensityData
  where snp = 'rs123456';
```

> “Eight minutes and 4+ terabytes of data queried later I had my results. (...) If we ever wanted to run a model over all the data we better be ready to wait roughly 38 years and pay $50 million. Clearly this wasn’t going to work.” – Nick Strayer

There's no **cheap way** to parse 25tb of data at once.

Sorting is hard, especially when **data is distributed**.

Never, ever, try and make 2.5 million partitions. (**cost: $1k+ USD**)

Don't sleep on the basics. Someone probably **solved your problem** in the 80s.

Gnu parallel is magic and everyone should use it.

DNA Solution

```
yp1234,577,1,3

yp5678,577,3,5

Yp9012,132,8,9

...
```

DNA Solution

```
awk -F, '{ print > $2 ".csv" }' file.csv
```

```
yp1234,577,1,3
yp5678,577,3,5
Yp9012,132,8,9
```

GNU Parallel

```
parallel --block 100M --pipe \
awk -F '\t'
'{
  print $1,...,$30 \
    chunked/{#}_chr \
    $15\".csv\"
}'
```

```
END { print “tada!” }
```

> “The Enlightened Ones say that....

You should never use **C** if you can do it with a **script**;

You should never use a **script** if you can do it with **awk**;

Never use **awk** if you can do it with **sed**;

Never use **sed** if you can do it with **grep**.”

yes 'SomeSampleText SomeOtherText 33 1970 YetAnotherText 777 abc 1 AndSomeMore' | head -1000000 > bigsample.txt

time gawk 'BEGIN {a = 0;} {if (**$5 == "YetAnotherText"**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {a = 0;} {if (**$0 ~ /YetAnotherText/**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {a = 0;} **/YetAnotherText/** {a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {a = 0;} {if (**NF == 9**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {a = 0;} {if (**$1 == "SomeSampleText"**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {a = 0;} {if (**$9 == "AndSomeMore"**) a ++;} END {print "a: " a;}' bigsample.txt

code	time
$5 == "YetAnotherText"	0m5.857s
$0 ~ /YetAnotherText/	0m5.252s
/YetAnotherText/	0m5.190s
NF == 9	0m5.441s
$1 == "SomeSampleText"	0m5.084s
$9 == "AndSomeMore"	0m5.711s

yes "<SomeSampleText:SomeOtherText=33>1970<YetAnotherText:777=abc>1<AndSomeMore>" | head -1000000 > bigsample.txt

time gawk 'BEGIN {**FS = "<|:|=";**} {if (**$5 == "YetAnotherText"**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {FS = "<|:|=";} {if (**$0 ~ /YetAnotherText/**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {FS = "<|:|=";} **/YetAnotherText/** {a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {FS = "<|:|=";} {if (**NF == 8**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {FS = "<|:|=";} {if (**$2 == "SomeSampleText"**) a ++;} END {print "a: " a;}' bigsample.txt

time gawk 'BEGIN {FS = "<|:|=";} {if (**$8 == "AndSomeMore>"**) a ++;} END {print "a: " a;}' bigsample.txt

code	time
$5 == "YetAnotherText"	0m8.287s
$0 ~ /YetAnotherText/	0m5.530s
/YetAnotherText/	0m5.362s
NF == 8	0m8.457s
$2 == "SomeSampleText"	0m6.519s
$8 == "AndSomeMore>"	0m8.490s

time grep -c YetAnotherText bigsample.txt

0m0.471s

yes 'a b c d' | head -12000000 > bigsample.txt

time gawk '{if(**NF==5**)print("a")}' bigsample.txt

time gawk '{if(**$4=="Hahaha"**)print("a")}' bigsample.txt

time gawk '{if(**$1=="Hahaha"**)print("a")}' bigsample.txt

time gawk '**/Hahaha/**{if(**$4=="Hahaha"**)print("a")}' bigsample.txt

code	time
NF==5	0m1.305s
$4=="Hahaha"	0m1.249s
$1=="Hahaha"	0m0.997s
/Hahaha/ && $4=="Hahaha"	0m0.932s

It seems that parsing and splitting into fields is faster when there is **one simple delimiter**, instead of several delimiters.

*Getting NF can be slow because the line has to be parsed and fields calculated, and more so if there are **several delimiters**.*

$N is faster than $M where N < M

/pattern/ in $0 is faster than $N == "pattern" (especially if N is not in the beginning)

```
<xn:SubNetwork id="ROOT_1">
    <xn:SubNetwork id="ROOT_2">
        <xn:attributes>
        ...
        </xn:attributes>
```

xml sample 300k times bigger (5,4 million lines, 160MB)

time gawk 'BEGIN{**FS="[<:=]"**}**NF>=4**{a++}END{print a+0}' bigsample.txt

time gawk 'BEGIN{**FS="<|:|="**}**NF>=4**{a++}END{print a+0}' bigsample.txt

time gawk 'BEGIN{**FS="<|:|="**}**NF>=4&&/:SubNetwork/**{a++}END{print a+0}' bigsample.txt

time gawk 'BEGIN{**FS=":SubNetwork"**}**NF>=2**{a++}END{print a+0}' bigsample.txt

time gawk '**/:SubNetwork/**{a++}END{print a}' bigsample.txt

code	time
FS="[<:=] && NF>=4	0m20.256s
FS="<|:|= && NF>=4	0m19.938s
FS="<|:|= && NF>=4 && /:SubNetwork/	0m20.569s
FS=":SubNetwork" && NF>=2	0m17.190s
/:SubNetwork/	0m12.903s

Debug

[GNM]AWK -D >?

Profiling

[GNM]AWK -p awkprof.out

MAWK

Same code, different **parser**.

Some issues with **bugs** and nonconformant GAWK/NAWK behavior.

Fast as hell, though (**q.e.d. ?**)

gawk	mawk
0m32.588s	0m7.327s
0m42.878s	0m14.434s
0m4.359s	0m1.490s
1m29.436s	0m22.915s

> “Show that if you use your :SubNetwork as field separator, it's the fastest.”

Regex as filter is much faster than run the action.

Mawk is fast as hell. (q.e.d.)

References

- https://ferd.ca/awk-in-20-minutes.html
- https://www.grymoire.com/Unix/Awk.html
- https://www.gnu.org/software/gawk/manual/gawk.html
- https://livefreeordichotomize.com/2019/06/04/using_awk_and_r_to_parse_25tb/
- https://stackoverflow.com/questions/43513975/awk-gawk-performance
- https://invisible-island.net/mawk/
- https://www.gnu.org/software/parallel/

INTELITRADER