Easy parallelization in R (using the foreach package)

Dan Turner (turnerdan dot com)

For a live (Zoom) workshop on March 14th, 2022.

Welcome to this R workshop about loops loops loops!

Before we can begin, please make sure that you can run the code chunk below.

Code chunks are areas of "R Markdown Notebooks" that allow us to execute our scripts. You can use the "Run" commands at the top of the code editor pane to run all or parts of chunks, or use the Cmd + Enter shortcut to Run any highlighted code, or the current line.

```
# Set the root to point to wherever /howtoforeach/ is on your computer
setwd( "~/Git/howtoforeach" )  # Change

## Load required packages

# For workshop related tasks
library( tictoc )     # for obtaining accurate code run time
require( lme4 )       # we will bootstrap a few models; only using a few functions so saving memory by
library( beepr )      # sometimes I like an auditory notification when long processes finish

# For making foreach run correctly
library( tidyverse )  # for data processing and better syntax
library( foreach )    # for parallel loop procedures across cores
library( parallel )   # for setting up your machine for multi-core processing
library( doParallel ) # for setting up your machine for multi-core processing (also)
```

We will do some additional setup specific to `foreach` when that package is introduced.

Part 1: Getting started

To loop or not to loop

Beginning coders often start by writing looping scripts because they highlight the power of using computers to automate tasks. In general, "for" loops are used for tasks with a clear end point, and "while" loops are used for tasks with indefinite ends points.

For example, if you want to repeat a sequence for every row in a dataframe, you would use `for()`, but if you want to repeat a sequence until a condition is met, you would use a `while()` loop, which runs while the argument of `while()` remains `TRUE`.

You may have received advice to "not write loops", but there is nothing inherently incorrect about writing loops; it's legitimate code! This advice comes from the fact that some people believe that slow code is bad code, but I disagree.

Sometimes, writing a loop is easier because it follows a sequential flow that humans are used to reasoning through. When R starts breaking up calculations across cores, rewriting code into faster languages, and other performance-enhancing strategies, sometimes it's difficult to understand what is really happening at the same level.

Also, sequential processing is inherent to many computational tasks, such as text processing, repeated sampling (bootstrapping), and web scraping. So, rather than preaching that nobody should write another loop, I will show you how to write loops that are more computationally efficient.

What's the solution?

Let's assume we have to write our code as a loop for some reason, but we also want the code to execute as quickly as possible. The main reason why loops tend to run slow is due to their sequential nature–the computer has to wait to process Item 100 until it can process Item 99–and every processing delay piles up.

Our solution will be to split up the single sequence into multiple chunks, so Items 1 through 33, 34 through 67, and 68 through 100 will be processed in parallel, cutting execution time up to 66%.

"I need the absolute fastest code!"

I highly recommend the `tidyverse` collection of R packages. If you can substitute a loop for a tidyverse statement, it will be orders of magnitude faster in most cases. It is also faster to write, and easier to understand, which can be just as important as having a working script.

If you don't mind learning different syntax, you can also look at `data.table` which is generally faster than `tidyverse`, but is limited to rectangular (column-based) data.

"Vectorized" is a catch-all term for these super-fast packages that operate at multiple points of the data at the same time. Sometimes vectorized code is also run in parallel–it depends on the package and what the specific process is.

Part 2: The humble loop

Loading data files with a loop

In the next code chunk, you'll see the humble loop loading our text data files, and a vectorized version for comparison. We will time our operations using the package 'tictoc'.

Notice in this example how much more work goes into writing a loop, but it's clear and easy to modify. For example, I can write a progress message into the loop, but not into its vectorized equivalent.

Also, see how I have to create a container for the looped data; otherwise each loop would overwrite the previous one, and we would not be able to access the results of each iteration.

```
##############################
# Sequential (looping) import #
##############################

# List the files in the /data/ directory to build the filepaths
filepaths = paste0( getwd(), "/data/", list.files( path = "./data/", pattern = ".csv"))

# Let's take a peek at the first file, file.1
file.1 = read_csv( filepaths[[1]], col_types = cols())  # TIP: col_types = cols() suppresses warnings

# Empty version of file.1 (so I don't need to specify colnames later)
reviews.seq = file.1[FALSE,]

# Looping the list of files (slow and lots of lines)
tic("Looping version")  # start timer
  # I am demonstrating a 'for' loop for now, but there are also 'while' loops
  for( file in seq_along( filepaths ) ) {
    print( paste("Loading file", file, "out of", paste0(length(filepaths)) ) )  # try this with vectori
    the.file = read_csv( filepaths[[ file ]], col_types = cols() )  # read the data into the.file
    reviews.seq = rbind( reviews.seq, the.file)  # add it the.file to the bottom of reviews.seq
  }
```

```
## [1] "Loading file 1 out of 3"
## [1] "Loading file 2 out of 3"
## [1] "Loading file 3 out of 3"
```

```
toc()  # end timer
```

```
## Looping version: 0.28 sec elapsed
```

For comparison, all of the above is accomplished in 1 line here:

```
######################
# Vectorized import #
######################

tic("Vectorized version")  # start timer

# With the plyr package, we can apply read_csv directly to the filepaths list
reviews = plyr::ldply( .data = filepaths, .fun = read_csv, col_types = cols() )

toc()  # end coder timer
```

```
## Vectorized version: 0.258 sec elapsed
```

```
#  Hopefully the two objects are of the same size.
ifelse( identical( dim(reviews), dim(reviews.seq) ),
        "PASS: Results match in size", "FAIL: Results do not match in size")
```

```
## [1] "PASS: Results match in size"
```

We will conclude our comparisons of looping versus vectorized code here... if you want speed, avoid sequential operations and instead use functions that automatically vectorize the computation, such as `tidyverse` or `data.table`.

Choosing a computational environment

Before we transition into executing parallel code, let's talk about another important factor in performance– how much processing power is available? If you need more capacity than a single computer can provide, then you may want to run the code on a purpose-made computational environment.

For example, Northwestern University maintains a research computing cluster called Quest with 11,800 cores, which are optimized for running code for its students and faculty. Basic Access is free. A non-free option is Amazon AWS, which will allow you to (glossing over details) create a virtual computer for you to run code on.

To state the obvious, if your code will be running for a few days, it's more convenient to have it run on a non-essential non-personal computer!

More info: https://www.it.northwestern.edu/research/user-services/quest/overview.html

---

Part 3: The parallel loop

Text processing example

Text is typically slow to process, for a variety of reasons, and the way we work with the data during text analysis is often sequential. This makes text processing a good candidate for use of parallel loops.

For this example, we will use the data we loaded, which consists of ~50k job reviews from Amazon employees, scraped from indeed.com. Expect this chunk to take extra time to process.

```
# For each row, let's extract the state and count the length of the review details
{ # start timing
tic("Sequential text processing")

for ( row in 1:nrow( reviews ) ) {

  # Using a regular expression to extract pairs of capitalized letters
  reviews$state[row] = str_extract( reviews$Work_Location[row],"\\b[A-Z]{2}")

  # nchar() counts the characters
  reviews$nchar[row] = nchar( reviews$Review_Details[row] )

}

toc() } # end timing
```

```
## Sequential text processing: 39.031 sec elapsed
```

Now let's split the loop over our cores, using foreach

First we have to set up the computer to run our loops in parallel. Out of the box, `foreach` does not run in parallel; it requires the `doParallel` package in order to use multiple cores. This only has to be done once per R session, and usually I do this immediately after loading libraries.

```
# Returns the number of cores that R can see
n.cores = detectCores()

# I like to keep one core in reserve. Otherwise, if the R session crashes, it can make your whole compu
cl = makeCluster( n.cores - 1 )

# Initiate an R process on the earmarked cores
registerDoParallel( cl )
```

Now that we have N-1 cores ready to receive commands from R, we will take the previous loop and divide it across the available cores. The following chunk calls `foreach()` to create a loop of row indexes which we will use to process each review in two ways: 1. We will attempt to extract the state abbreviation code (e.g. IL, CA, NY) from the review. 2. We will count the number of characters in the review. The results will be passed to `rbind` to create an extended version of the original data.

```
{  # begin timing
tic("Parallel text processing")

# Same functions as our sequential loop, but using foreach() and %dopar%
reviews.processed = foreach( row = 1:nrow( reviews ),
         .combine = "rbind",
         .packages = c("stringr")) %dopar% {  # Packages need to be loaded for each core

          # Extract state code (2 capital letters)
          the.state = str_extract( reviews$Work_Location[row],"\\b[A-Z]{2}")

          # Extract review length
          the.nchar = nchar( reviews$Review_Details[row] )

          # I'm essentially appending 2 columns to the original data by extending each row
          return( c( reviews[row,], the.state, the.nchar ) )

}  # end of foreach loop

toc() } # end timing chunk
```

```
## Parallel text processing: 38.529 sec elapsed
```

```
# Adding pretty column names to our extended data
reviews.processed = as.data.frame( reviews.processed )
colnames( reviews.processed ) = c(colnames( reviews ), # starting point
                              "state.code", # I could also left join
                              "review.len") # but this is fast and clean

reviews = reviews.processed
```

Benchmarks

Dan's 2012 MacBook (no Zoom): 45s

Data chunking example

In the previous example, we looped each row by creating an iterator out of `reviews`. This is not a great use of parallel processing though, because there is little to gain when splitting 48k rows across a few cores. It's usually more efficient to break up your data into bigger chunks, for example, by state. You always want to parallelize the slowest part of the process, if possible.

For example, I usually use foreach() when web scraping because the slowest part of the process is making a connection with the remote server, so I have each core processing a different URL usually. I make sure I only ever have to call a URL once, and once I have made a connection, I used vectorized code to extract the target data, which minimizes the time per page.

```
# Let's get a list of all the (confirmed) states that appear in the $state field. We confirm states by
state.list = as.data.frame( table( unlist( reviews$state ) ) )
the.states = intersect( state.abb, reviews$state )

{  # beginning of timer
tic("Parallel loop of all states")

# Parallel loop of the states
reviews.bystate = foreach( state = seq_along( the.states ),
         .combine = "rbind",
         .packages = c("tictoc")) %dopar% {  # let's do it in base R to avoid calling packages here

           # Collect the rows for this state
           the.state = unlist( the.states[state] )
           state.rows = reviews[ which( reviews$state.code == the.state ), ]

           # Let's time each state, out of curiosity
           tic(paste("State:", the.state))

           # Sum the nchar column
           state.sum = sum ( as.integer( state.rows$review.len ) )

           # Let's get the average rating too, while we're at it
           state.avg = round( mean( as.integer( state.rows$Rating ) ) )

           # Returning a vector on each loop, which I rbind into a matrix (fast!)
           return( c(the.state, state.sum, state.avg, toc()$toc) )

}  # end reviews.bystate
toc() }  # end of timer
```

```
## Parallel loop of all states: 2.278 sec elapsed
```

```r
# Turn the matrix into a df w/ pretty colnames
reviews.bystate = as.data.frame( reviews.bystate )
colnames( reviews.bystate ) = c( "state.code", "review.len.sum", "rating.mean", "processing time")
```

It's much faster to process state-by-state versus row-by-row, but why?

Why is it faster to loop state-by-state?

The 47 states represented in the data will be equally divided across multiple cores on my computer. For example, since I am allowing R to utilize 3 cores, each processor only has to handle about 15 states, rather than a single core handling all 47 one at a time.

Maybe Core #1 processed Michigan and Florida and Core #2 processed Illinois and California. We have a lot of control over this process, but by default foreach() is optimized to balance performance with being a general purpose tool.

Using foreach() is a little bit like dividing your data into thirds and having two colleagues process them for you while you run your share. At the end of the day, you'll have 3 sets of results which, when combined, are identical to if you ran the process sequentially.

Coding Challenge A

The first challenge involves repairing the `$state` variable, which you can tell by looking at `state.list` contains many errors. Here's some meta-code that shows the basic process I will guide you through:

Split Place_Of_Work_1 to get state

Loop city,state strings to extract state abbreviation `foreach(city_state_string)`

Validate extracted data using built-in list, state.abb `if(state in state.abb) str_extract(state)`

```r
# write_rds( reviews, "./data/snapshot_a.rds")  in coding_challenges you will load this file
```

Part 4: Going deeper into `foreach()`

Now that we've covered the basic setup and syntax of `foreach()`, we will talk more about its parameters and use cases.

Combining results

Besides being able to use multiple iterators to generate loops, `foreach` is also flexible in how it formats its output. Depending on how much memory a

Concatenate `.combine = 'c'`

I use this method when I want the result to be a list, or list of lists.

Column/row bind `.combine = 'cbind'`/`.combine = 'rbind'`

I use this method when I want the result to be a dataframe, tibble, or matrix.

Basic operators `.combine = '+'/.combine = '*'`

These methods make sense for bootstrapping and simulating data.

Custom function `.combine = 'cfun'`

```
# Adapted from the foreach documentation
comb.c     = foreach(i=icount(4), .combine='c') %dopar% exp(i)
comb.cbind = foreach(icount(4), .combine='cbind') %dopar% rnorm(4)
comb.add   = foreach(icount(4), .combine='+') %dopar% rnorm(4)

# TIP: You can include conditional logic to foreach calls using `%:%` and `when()` ... somewhat like Py

# As long as `i` is between 1-5, wrote the corresponding letter to the console
foreach( i = icount( 32 ) ) %:% when( i <= 5 ) %dopar% letters[i]
```

```
## [[1]]
## [1] "a"
##
## [[2]]
## [1] "b"
##
## [[3]]
## [1] "c"
##
## [[4]]
## [1] "d"
##
## [[5]]
## [1] "e"
```

Nested `foreach` loops

A common text analysis problem is computing the similarity score between different strings. There are a lot of scores we could use, but to keep things simple, we simply take the proportion of words in common.

Loops can contain other loops, which can be convenient for checking all combinations of variables. In our toy problem, we are comparing vectors of words, so the first thing we need to do is create a dataset consisting of these vectors.

```
# Create a list of vectors, one for each review, containing a vector of words
grid_words = unlist( reviews$Place_Of_Work_1, reviews$Place_Of_Work_2 ) %>%
  tolower() %>%
  str_extract_all( pattern = '[a-z]+' )

# Print the head below
head( grid_words )
```

```
## [[1]]
## [1] "amazon"   "shopper"  "current"  "employee" "dedham"   "ma"       "december"
```

```
##
## [[2]]
## [1] "warehouse" "team"      "member"    "former"    "employee"  "richmond"
## [7] "va"        "december"
##
## [[3]]
##  [1] "warehouse"  "order"      "picker"     "driver"     "both"
##  [6] "inbound"    "and"        "outbound"   "capacities" "former"
## [11] "employee"   "charleston" "tn"         "december"
##
## [[4]]
##  [1] "sortation" "associate" "at"        "delivery"  "station"   "former"
##  [7] "employee"  "austin"    "tx"        "december"
##
## [[5]]
## [1] "warehouse"   "worker"      "former"      "employee"    "pendergrass"
## [6] "ga"          "december"
##
## [[6]]
## [1] "amazon"    "warehouse" "associate" "current"   "employee"  "brooklyn"
## [7] "park"      "mn"        "december"
```

The easiest way to exhaustively compare two vectors is by using `expand.grid()` which finds all combinations. For example:

```
expand.grid( state.abb, state.abb )
```

```
##      Var1 Var2
## 1      AL   AL
## 2      AK   AL
## 3      AZ   AL
## 4      AR   AL
## 5      CA   AL
## 6      CO   AL
## 7      CT   AL
## 8      DE   AL
## 9      FL   AL
## 10     GA   AL
## 11     HI   AL
## 12     ID   AL
## 13     IL   AL
## 14     IN   AL
## 15     IA   AL
## 16     KS   AL
## 17     KY   AL
## 18     LA   AL
## 19     ME   AL
## 20     MD   AL
## 21     MA   AL
## 22     MI   AL
## 23     MN   AL
## 24     MS   AL
## 25     MO   AL
```

```
## 26     MT   AL
## 27     NE   AL
## 28     NV   AL
## 29     NH   AL
## 30     NJ   AL
## 31     NM   AL
## 32     NY   AL
## 33     NC   AL
## 34     ND   AL
## 35     OH   AL
## 36     OK   AL
## 37     OR   AL
## 38     PA   AL
## 39     RI   AL
## 40     SC   AL
## 41     SD   AL
## 42     TN   AL
## 43     TX   AL
## 44     UT   AL
## 45     VT   AL
## 46     VA   AL
## 47     WA   AL
## 48     WV   AL
## 49     WI   AL
## 50     WY   AL
## 51     AL   AK
## 52     AK   AK
## 53     AZ   AK
## 54     AR   AK
## 55     CA   AK
## 56     CO   AK
## 57     CT   AK
## 58     DE   AK
## 59     FL   AK
## 60     GA   AK
## 61     HI   AK
## 62     ID   AK
## 63     IL   AK
## 64     IN   AK
## 65     IA   AK
## 66     KS   AK
## 67     KY   AK
## 68     LA   AK
## 69     ME   AK
## 70     MD   AK
## 71     MA   AK
## 72     MI   AK
## 73     MN   AK
## 74     MS   AK
## 75     MO   AK
## 76     MT   AK
## 77     NE   AK
## 78     NV   AK
## 79     NH   AK
```

```
## 80     NJ   AK
## 81     NM   AK
## 82     NY   AK
## 83     NC   AK
## 84     ND   AK
## 85     OH   AK
## 86     OK   AK
## 87     OR   AK
## 88     PA   AK
## 89     RI   AK
## 90     SC   AK
## 91     SD   AK
## 92     TN   AK
## 93     TX   AK
## 94     UT   AK
## 95     VT   AK
## 96     VA   AK
## 97     WA   AK
## 98     WV   AK
## 99     WI   AK
## 100    WY   AK
## 101    AL   AZ
## 102    AK   AZ
## 103    AZ   AZ
## 104    AR   AZ
## 105    CA   AZ
## 106    CO   AZ
## 107    CT   AZ
## 108    DE   AZ
## 109    FL   AZ
## 110    GA   AZ
## 111    HI   AZ
## 112    ID   AZ
## 113    IL   AZ
## 114    IN   AZ
## 115    IA   AZ
## 116    KS   AZ
## 117    KY   AZ
## 118    LA   AZ
## 119    ME   AZ
## 120    MD   AZ
## 121    MA   AZ
## 122    MI   AZ
## 123    MN   AZ
## 124    MS   AZ
## 125    MO   AZ
## 126    MT   AZ
## 127    NE   AZ
## 128    NV   AZ
## 129    NH   AZ
## 130    NJ   AZ
## 131    NM   AZ
## 132    NY   AZ
## 133    NC   AZ
```

```
## 134    ND   AZ
## 135    OH   AZ
## 136    OK   AZ
## 137    OR   AZ
## 138    PA   AZ
## 139    RI   AZ
## 140    SC   AZ
## 141    SD   AZ
## 142    TN   AZ
## 143    TX   AZ
## 144    UT   AZ
## 145    VT   AZ
## 146    VA   AZ
## 147    WA   AZ
## 148    WV   AZ
## 149    WI   AZ
## 150    WY   AZ
## 151    AL   AR
## 152    AK   AR
## 153    AZ   AR
## 154    AR   AR
## 155    CA   AR
## 156    CO   AR
## 157    CT   AR
## 158    DE   AR
## 159    FL   AR
## 160    GA   AR
## 161    HI   AR
## 162    ID   AR
## 163    IL   AR
## 164    IN   AR
## 165    IA   AR
## 166    KS   AR
## 167    KY   AR
## 168    LA   AR
## 169    ME   AR
## 170    MD   AR
## 171    MA   AR
## 172    MI   AR
## 173    MN   AR
## 174    MS   AR
## 175    MO   AR
## 176    MT   AR
## 177    NE   AR
## 178    NV   AR
## 179    NH   AR
## 180    NJ   AR
## 181    NM   AR
## 182    NY   AR
## 183    NC   AR
## 184    ND   AR
## 185    OH   AR
## 186    OK   AR
## 187    OR   AR
```

```
## 188    PA   AR
## 189    RI   AR
## 190    SC   AR
## 191    SD   AR
## 192    TN   AR
## 193    TX   AR
## 194    UT   AR
## 195    VT   AR
## 196    VA   AR
## 197    WA   AR
## 198    WV   AR
## 199    WI   AR
## 200    WY   AR
## 201    AL   CA
## 202    AK   CA
## 203    AZ   CA
## 204    AR   CA
## 205    CA   CA
## 206    CO   CA
## 207    CT   CA
## 208    DE   CA
## 209    FL   CA
## 210    GA   CA
## 211    HI   CA
## 212    ID   CA
## 213    IL   CA
## 214    IN   CA
## 215    IA   CA
## 216    KS   CA
## 217    KY   CA
## 218    LA   CA
## 219    ME   CA
## 220    MD   CA
## 221    MA   CA
## 222    MI   CA
## 223    MN   CA
## 224    MS   CA
## 225    MO   CA
## 226    MT   CA
## 227    NE   CA
## 228    NV   CA
## 229    NH   CA
## 230    NJ   CA
## 231    NM   CA
## 232    NY   CA
## 233    NC   CA
## 234    ND   CA
## 235    OH   CA
## 236    OK   CA
## 237    OR   CA
## 238    PA   CA
## 239    RI   CA
## 240    SC   CA
## 241    SD   CA
```

```
## 242    TN   CA
## 243    TX   CA
## 244    UT   CA
## 245    VT   CA
## 246    VA   CA
## 247    WA   CA
## 248    WV   CA
## 249    WI   CA
## 250    WY   CA
## 251    AL   CO
## 252    AK   CO
## 253    AZ   CO
## 254    AR   CO
## 255    CA   CO
## 256    CO   CO
## 257    CT   CO
## 258    DE   CO
## 259    FL   CO
## 260    GA   CO
## 261    HI   CO
## 262    ID   CO
## 263    IL   CO
## 264    IN   CO
## 265    IA   CO
## 266    KS   CO
## 267    KY   CO
## 268    LA   CO
## 269    ME   CO
## 270    MD   CO
## 271    MA   CO
## 272    MI   CO
## 273    MN   CO
## 274    MS   CO
## 275    MO   CO
## 276    MT   CO
## 277    NE   CO
## 278    NV   CO
## 279    NH   CO
## 280    NJ   CO
## 281    NM   CO
## 282    NY   CO
## 283    NC   CO
## 284    ND   CO
## 285    OH   CO
## 286    OK   CO
## 287    OR   CO
## 288    PA   CO
## 289    RI   CO
## 290    SC   CO
## 291    SD   CO
## 292    TN   CO
## 293    TX   CO
## 294    UT   CO
## 295    VT   CO
```

```
## 296    VA   CO
## 297    WA   CO
## 298    WV   CO
## 299    WI   CO
## 300    WY   CO
## 301    AL   CT
## 302    AK   CT
## 303    AZ   CT
## 304    AR   CT
## 305    CA   CT
## 306    CO   CT
## 307    CT   CT
## 308    DE   CT
## 309    FL   CT
## 310    GA   CT
## 311    HI   CT
## 312    ID   CT
## 313    IL   CT
## 314    IN   CT
## 315    IA   CT
## 316    KS   CT
## 317    KY   CT
## 318    LA   CT
## 319    ME   CT
## 320    MD   CT
## 321    MA   CT
## 322    MI   CT
## 323    MN   CT
## 324    MS   CT
## 325    MO   CT
## 326    MT   CT
## 327    NE   CT
## 328    NV   CT
## 329    NH   CT
## 330    NJ   CT
## 331    NM   CT
## 332    NY   CT
## 333    NC   CT
## 334    ND   CT
## 335    OH   CT
## 336    OK   CT
## 337    OR   CT
## 338    PA   CT
## 339    RI   CT
## 340    SC   CT
## 341    SD   CT
## 342    TN   CT
## 343    TX   CT
## 344    UT   CT
## 345    VT   CT
## 346    VA   CT
## 347    WA   CT
## 348    WV   CT
## 349    WI   CT
```

```
## 350    WY   CT
## 351    AL   DE
## 352    AK   DE
## 353    AZ   DE
## 354    AR   DE
## 355    CA   DE
## 356    CO   DE
## 357    CT   DE
## 358    DE   DE
## 359    FL   DE
## 360    GA   DE
## 361    HI   DE
## 362    ID   DE
## 363    IL   DE
## 364    IN   DE
## 365    IA   DE
## 366    KS   DE
## 367    KY   DE
## 368    LA   DE
## 369    ME   DE
## 370    MD   DE
## 371    MA   DE
## 372    MI   DE
## 373    MN   DE
## 374    MS   DE
## 375    MO   DE
## 376    MT   DE
## 377    NE   DE
## 378    NV   DE
## 379    NH   DE
## 380    NJ   DE
## 381    NM   DE
## 382    NY   DE
## 383    NC   DE
## 384    ND   DE
## 385    OH   DE
## 386    OK   DE
## 387    OR   DE
## 388    PA   DE
## 389    RI   DE
## 390    SC   DE
## 391    SD   DE
## 392    TN   DE
## 393    TX   DE
## 394    UT   DE
## 395    VT   DE
## 396    VA   DE
## 397    WA   DE
## 398    WV   DE
## 399    WI   DE
## 400    WY   DE
## 401    AL   FL
## 402    AK   FL
## 403    AZ   FL
```

```
## 404    AR   FL
## 405    CA   FL
## 406    CO   FL
## 407    CT   FL
## 408    DE   FL
## 409    FL   FL
## 410    GA   FL
## 411    HI   FL
## 412    ID   FL
## 413    IL   FL
## 414    IN   FL
## 415    IA   FL
## 416    KS   FL
## 417    KY   FL
## 418    LA   FL
## 419    ME   FL
## 420    MD   FL
## 421    MA   FL
## 422    MI   FL
## 423    MN   FL
## 424    MS   FL
## 425    MO   FL
## 426    MT   FL
## 427    NE   FL
## 428    NV   FL
## 429    NH   FL
## 430    NJ   FL
## 431    NM   FL
## 432    NY   FL
## 433    NC   FL
## 434    ND   FL
## 435    OH   FL
## 436    OK   FL
## 437    OR   FL
## 438    PA   FL
## 439    RI   FL
## 440    SC   FL
## 441    SD   FL
## 442    TN   FL
## 443    TX   FL
## 444    UT   FL
## 445    VT   FL
## 446    VA   FL
## 447    WA   FL
## 448    WV   FL
## 449    WI   FL
## 450    WY   FL
## 451    AL   GA
## 452    AK   GA
## 453    AZ   GA
## 454    AR   GA
## 455    CA   GA
## 456    CO   GA
## 457    CT   GA
```

```
## 458    DE   GA
## 459    FL   GA
## 460    GA   GA
## 461    HI   GA
## 462    ID   GA
## 463    IL   GA
## 464    IN   GA
## 465    IA   GA
## 466    KS   GA
## 467    KY   GA
## 468    LA   GA
## 469    ME   GA
## 470    MD   GA
## 471    MA   GA
## 472    MI   GA
## 473    MN   GA
## 474    MS   GA
## 475    MO   GA
## 476    MT   GA
## 477    NE   GA
## 478    NV   GA
## 479    NH   GA
## 480    NJ   GA
## 481    NM   GA
## 482    NY   GA
## 483    NC   GA
## 484    ND   GA
## 485    OH   GA
## 486    OK   GA
## 487    OR   GA
## 488    PA   GA
## 489    RI   GA
## 490    SC   GA
## 491    SD   GA
## 492    TN   GA
## 493    TX   GA
## 494    UT   GA
## 495    VT   GA
## 496    VA   GA
## 497    WA   GA
## 498    WV   GA
## 499    WI   GA
## 500    WY   GA
## 501    AL   HI
## 502    AK   HI
## 503    AZ   HI
## 504    AR   HI
## 505    CA   HI
## 506    CO   HI
## 507    CT   HI
## 508    DE   HI
## 509    FL   HI
## 510    GA   HI
## 511    HI   HI
```

```
## 512    ID   HI
## 513    IL   HI
## 514    IN   HI
## 515    IA   HI
## 516    KS   HI
## 517    KY   HI
## 518    LA   HI
## 519    ME   HI
## 520    MD   HI
## 521    MA   HI
## 522    MI   HI
## 523    MN   HI
## 524    MS   HI
## 525    MO   HI
## 526    MT   HI
## 527    NE   HI
## 528    NV   HI
## 529    NH   HI
## 530    NJ   HI
## 531    NM   HI
## 532    NY   HI
## 533    NC   HI
## 534    ND   HI
## 535    OH   HI
## 536    OK   HI
## 537    OR   HI
## 538    PA   HI
## 539    RI   HI
## 540    SC   HI
## 541    SD   HI
## 542    TN   HI
## 543    TX   HI
## 544    UT   HI
## 545    VT   HI
## 546    VA   HI
## 547    WA   HI
## 548    WV   HI
## 549    WI   HI
## 550    WY   HI
## 551    AL   ID
## 552    AK   ID
## 553    AZ   ID
## 554    AR   ID
## 555    CA   ID
## 556    CO   ID
## 557    CT   ID
## 558    DE   ID
## 559    FL   ID
## 560    GA   ID
## 561    HI   ID
## 562    ID   ID
## 563    IL   ID
## 564    IN   ID
## 565    IA   ID
```

```
## 566    KS   ID
## 567    KY   ID
## 568    LA   ID
## 569    ME   ID
## 570    MD   ID
## 571    MA   ID
## 572    MI   ID
## 573    MN   ID
## 574    MS   ID
## 575    MO   ID
## 576    MT   ID
## 577    NE   ID
## 578    NV   ID
## 579    NH   ID
## 580    NJ   ID
## 581    NM   ID
## 582    NY   ID
## 583    NC   ID
## 584    ND   ID
## 585    OH   ID
## 586    OK   ID
## 587    OR   ID
## 588    PA   ID
## 589    RI   ID
## 590    SC   ID
## 591    SD   ID
## 592    TN   ID
## 593    TX   ID
## 594    UT   ID
## 595    VT   ID
## 596    VA   ID
## 597    WA   ID
## 598    WV   ID
## 599    WI   ID
## 600    WY   ID
## 601    AL   IL
## 602    AK   IL
## 603    AZ   IL
## 604    AR   IL
## 605    CA   IL
## 606    CO   IL
## 607    CT   IL
## 608    DE   IL
## 609    FL   IL
## 610    GA   IL
## 611    HI   IL
## 612    ID   IL
## 613    IL   IL
## 614    IN   IL
## 615    IA   IL
## 616    KS   IL
## 617    KY   IL
## 618    LA   IL
## 619    ME   IL
```

```
## 620    MD   IL
## 621    MA   IL
## 622    MI   IL
## 623    MN   IL
## 624    MS   IL
## 625    MO   IL
## 626    MT   IL
## 627    NE   IL
## 628    NV   IL
## 629    NH   IL
## 630    NJ   IL
## 631    NM   IL
## 632    NY   IL
## 633    NC   IL
## 634    ND   IL
## 635    OH   IL
## 636    OK   IL
## 637    OR   IL
## 638    PA   IL
## 639    RI   IL
## 640    SC   IL
## 641    SD   IL
## 642    TN   IL
## 643    TX   IL
## 644    UT   IL
## 645    VT   IL
## 646    VA   IL
## 647    WA   IL
## 648    WV   IL
## 649    WI   IL
## 650    WY   IL
## 651    AL   IN
## 652    AK   IN
## 653    AZ   IN
## 654    AR   IN
## 655    CA   IN
## 656    CO   IN
## 657    CT   IN
## 658    DE   IN
## 659    FL   IN
## 660    GA   IN
## 661    HI   IN
## 662    ID   IN
## 663    IL   IN
## 664    IN   IN
## 665    IA   IN
## 666    KS   IN
## 667    KY   IN
## 668    LA   IN
## 669    ME   IN
## 670    MD   IN
## 671    MA   IN
## 672    MI   IN
## 673    MN   IN
```

```
## 674    MS   IN
## 675    MO   IN
## 676    MT   IN
## 677    NE   IN
## 678    NV   IN
## 679    NH   IN
## 680    NJ   IN
## 681    NM   IN
## 682    NY   IN
## 683    NC   IN
## 684    ND   IN
## 685    OH   IN
## 686    OK   IN
## 687    OR   IN
## 688    PA   IN
## 689    RI   IN
## 690    SC   IN
## 691    SD   IN
## 692    TN   IN
## 693    TX   IN
## 694    UT   IN
## 695    VT   IN
## 696    VA   IN
## 697    WA   IN
## 698    WV   IN
## 699    WI   IN
## 700    WY   IN
## 701    AL   IA
## 702    AK   IA
## 703    AZ   IA
## 704    AR   IA
## 705    CA   IA
## 706    CO   IA
## 707    CT   IA
## 708    DE   IA
## 709    FL   IA
## 710    GA   IA
## 711    HI   IA
## 712    ID   IA
## 713    IL   IA
## 714    IN   IA
## 715    IA   IA
## 716    KS   IA
## 717    KY   IA
## 718    LA   IA
## 719    ME   IA
## 720    MD   IA
## 721    MA   IA
## 722    MI   IA
## 723    MN   IA
## 724    MS   IA
## 725    MO   IA
## 726    MT   IA
## 727    NE   IA
```

```
## 728    NV   IA
## 729    NH   IA
## 730    NJ   IA
## 731    NM   IA
## 732    NY   IA
## 733    NC   IA
## 734    ND   IA
## 735    OH   IA
## 736    OK   IA
## 737    OR   IA
## 738    PA   IA
## 739    RI   IA
## 740    SC   IA
## 741    SD   IA
## 742    TN   IA
## 743    TX   IA
## 744    UT   IA
## 745    VT   IA
## 746    VA   IA
## 747    WA   IA
## 748    WV   IA
## 749    WI   IA
## 750    WY   IA
## 751    AL   KS
## 752    AK   KS
## 753    AZ   KS
## 754    AR   KS
## 755    CA   KS
## 756    CO   KS
## 757    CT   KS
## 758    DE   KS
## 759    FL   KS
## 760    GA   KS
## 761    HI   KS
## 762    ID   KS
## 763    IL   KS
## 764    IN   KS
## 765    IA   KS
## 766    KS   KS
## 767    KY   KS
## 768    LA   KS
## 769    ME   KS
## 770    MD   KS
## 771    MA   KS
## 772    MI   KS
## 773    MN   KS
## 774    MS   KS
## 775    MO   KS
## 776    MT   KS
## 777    NE   KS
## 778    NV   KS
## 779    NH   KS
## 780    NJ   KS
## 781    NM   KS
```

```
## 782    NY   KS
## 783    NC   KS
## 784    ND   KS
## 785    OH   KS
## 786    OK   KS
## 787    OR   KS
## 788    PA   KS
## 789    RI   KS
## 790    SC   KS
## 791    SD   KS
## 792    TN   KS
## 793    TX   KS
## 794    UT   KS
## 795    VT   KS
## 796    VA   KS
## 797    WA   KS
## 798    WV   KS
## 799    WI   KS
## 800    WY   KS
## 801    AL   KY
## 802    AK   KY
## 803    AZ   KY
## 804    AR   KY
## 805    CA   KY
## 806    CO   KY
## 807    CT   KY
## 808    DE   KY
## 809    FL   KY
## 810    GA   KY
## 811    HI   KY
## 812    ID   KY
## 813    IL   KY
## 814    IN   KY
## 815    IA   KY
## 816    KS   KY
## 817    KY   KY
## 818    LA   KY
## 819    ME   KY
## 820    MD   KY
## 821    MA   KY
## 822    MI   KY
## 823    MN   KY
## 824    MS   KY
## 825    MO   KY
## 826    MT   KY
## 827    NE   KY
## 828    NV   KY
## 829    NH   KY
## 830    NJ   KY
## 831    NM   KY
## 832    NY   KY
## 833    NC   KY
## 834    ND   KY
## 835    OH   KY
```

```
## 836    OK   KY
## 837    OR   KY
## 838    PA   KY
## 839    RI   KY
## 840    SC   KY
## 841    SD   KY
## 842    TN   KY
## 843    TX   KY
## 844    UT   KY
## 845    VT   KY
## 846    VA   KY
## 847    WA   KY
## 848    WV   KY
## 849    WI   KY
## 850    WY   KY
## 851    AL   LA
## 852    AK   LA
## 853    AZ   LA
## 854    AR   LA
## 855    CA   LA
## 856    CO   LA
## 857    CT   LA
## 858    DE   LA
## 859    FL   LA
## 860    GA   LA
## 861    HI   LA
## 862    ID   LA
## 863    IL   LA
## 864    IN   LA
## 865    IA   LA
## 866    KS   LA
## 867    KY   LA
## 868    LA   LA
## 869    ME   LA
## 870    MD   LA
## 871    MA   LA
## 872    MI   LA
## 873    MN   LA
## 874    MS   LA
## 875    MO   LA
## 876    MT   LA
## 877    NE   LA
## 878    NV   LA
## 879    NH   LA
## 880    NJ   LA
## 881    NM   LA
## 882    NY   LA
## 883    NC   LA
## 884    ND   LA
## 885    OH   LA
## 886    OK   LA
## 887    OR   LA
## 888    PA   LA
## 889    RI   LA
```

```
## 890    SC   LA
## 891    SD   LA
## 892    TN   LA
## 893    TX   LA
## 894    UT   LA
## 895    VT   LA
## 896    VA   LA
## 897    WA   LA
## 898    WV   LA
## 899    WI   LA
## 900    WY   LA
## 901    AL   ME
## 902    AK   ME
## 903    AZ   ME
## 904    AR   ME
## 905    CA   ME
## 906    CO   ME
## 907    CT   ME
## 908    DE   ME
## 909    FL   ME
## 910    GA   ME
## 911    HI   ME
## 912    ID   ME
## 913    IL   ME
## 914    IN   ME
## 915    IA   ME
## 916    KS   ME
## 917    KY   ME
## 918    LA   ME
## 919    ME   ME
## 920    MD   ME
## 921    MA   ME
## 922    MI   ME
## 923    MN   ME
## 924    MS   ME
## 925    MO   ME
## 926    MT   ME
## 927    NE   ME
## 928    NV   ME
## 929    NH   ME
## 930    NJ   ME
## 931    NM   ME
## 932    NY   ME
## 933    NC   ME
## 934    ND   ME
## 935    OH   ME
## 936    OK   ME
## 937    OR   ME
## 938    PA   ME
## 939    RI   ME
## 940    SC   ME
## 941    SD   ME
## 942    TN   ME
## 943    TX   ME
```

```
## 944    UT   ME
## 945    VT   ME
## 946    VA   ME
## 947    WA   ME
## 948    WV   ME
## 949    WI   ME
## 950    WY   ME
## 951    AL   MD
## 952    AK   MD
## 953    AZ   MD
## 954    AR   MD
## 955    CA   MD
## 956    CO   MD
## 957    CT   MD
## 958    DE   MD
## 959    FL   MD
## 960    GA   MD
## 961    HI   MD
## 962    ID   MD
## 963    IL   MD
## 964    IN   MD
## 965    IA   MD
## 966    KS   MD
## 967    KY   MD
## 968    LA   MD
## 969    ME   MD
## 970    MD   MD
## 971    MA   MD
## 972    MI   MD
## 973    MN   MD
## 974    MS   MD
## 975    MO   MD
## 976    MT   MD
## 977    NE   MD
## 978    NV   MD
## 979    NH   MD
## 980    NJ   MD
## 981    NM   MD
## 982    NY   MD
## 983    NC   MD
## 984    ND   MD
## 985    OH   MD
## 986    OK   MD
## 987    OR   MD
## 988    PA   MD
## 989    RI   MD
## 990    SC   MD
## 991    SD   MD
## 992    TN   MD
## 993    TX   MD
## 994    UT   MD
## 995    VT   MD
## 996    VA   MD
## 997    WA   MD
```

```
## 998    WV   MD
## 999    WI   MD
## 1000   WY   MD
## 1001   AL   MA
## 1002   AK   MA
## 1003   AZ   MA
## 1004   AR   MA
## 1005   CA   MA
## 1006   CO   MA
## 1007   CT   MA
## 1008   DE   MA
## 1009   FL   MA
## 1010   GA   MA
## 1011   HI   MA
## 1012   ID   MA
## 1013   IL   MA
## 1014   IN   MA
## 1015   IA   MA
## 1016   KS   MA
## 1017   KY   MA
## 1018   LA   MA
## 1019   ME   MA
## 1020   MD   MA
## 1021   MA   MA
## 1022   MI   MA
## 1023   MN   MA
## 1024   MS   MA
## 1025   MO   MA
## 1026   MT   MA
## 1027   NE   MA
## 1028   NV   MA
## 1029   NH   MA
## 1030   NJ   MA
## 1031   NM   MA
## 1032   NY   MA
## 1033   NC   MA
## 1034   ND   MA
## 1035   OH   MA
## 1036   OK   MA
## 1037   OR   MA
## 1038   PA   MA
## 1039   RI   MA
## 1040   SC   MA
## 1041   SD   MA
## 1042   TN   MA
## 1043   TX   MA
## 1044   UT   MA
## 1045   VT   MA
## 1046   VA   MA
## 1047   WA   MA
## 1048   WV   MA
## 1049   WI   MA
## 1050   WY   MA
## 1051   AL   MI
```

```
## 1052   AK   MI
## 1053   AZ   MI
## 1054   AR   MI
## 1055   CA   MI
## 1056   CO   MI
## 1057   CT   MI
## 1058   DE   MI
## 1059   FL   MI
## 1060   GA   MI
## 1061   HI   MI
## 1062   ID   MI
## 1063   IL   MI
## 1064   IN   MI
## 1065   IA   MI
## 1066   KS   MI
## 1067   KY   MI
## 1068   LA   MI
## 1069   ME   MI
## 1070   MD   MI
## 1071   MA   MI
## 1072   MI   MI
## 1073   MN   MI
## 1074   MS   MI
## 1075   MO   MI
## 1076   MT   MI
## 1077   NE   MI
## 1078   NV   MI
## 1079   NH   MI
## 1080   NJ   MI
## 1081   NM   MI
## 1082   NY   MI
## 1083   NC   MI
## 1084   ND   MI
## 1085   OH   MI
## 1086   OK   MI
## 1087   OR   MI
## 1088   PA   MI
## 1089   RI   MI
## 1090   SC   MI
## 1091   SD   MI
## 1092   TN   MI
## 1093   TX   MI
## 1094   UT   MI
## 1095   VT   MI
## 1096   VA   MI
## 1097   WA   MI
## 1098   WV   MI
## 1099   WI   MI
## 1100   WY   MI
## 1101   AL   MN
## 1102   AK   MN
## 1103   AZ   MN
## 1104   AR   MN
## 1105   CA   MN
```

```
## 1106   CO   MN
## 1107   CT   MN
## 1108   DE   MN
## 1109   FL   MN
## 1110   GA   MN
## 1111   HI   MN
## 1112   ID   MN
## 1113   IL   MN
## 1114   IN   MN
## 1115   IA   MN
## 1116   KS   MN
## 1117   KY   MN
## 1118   LA   MN
## 1119   ME   MN
## 1120   MD   MN
## 1121   MA   MN
## 1122   MI   MN
## 1123   MN   MN
## 1124   MS   MN
## 1125   MO   MN
## 1126   MT   MN
## 1127   NE   MN
## 1128   NV   MN
## 1129   NH   MN
## 1130   NJ   MN
## 1131   NM   MN
## 1132   NY   MN
## 1133   NC   MN
## 1134   ND   MN
## 1135   OH   MN
## 1136   OK   MN
## 1137   OR   MN
## 1138   PA   MN
## 1139   RI   MN
## 1140   SC   MN
## 1141   SD   MN
## 1142   TN   MN
## 1143   TX   MN
## 1144   UT   MN
## 1145   VT   MN
## 1146   VA   MN
## 1147   WA   MN
## 1148   WV   MN
## 1149   WI   MN
## 1150   WY   MN
## 1151   AL   MS
## 1152   AK   MS
## 1153   AZ   MS
## 1154   AR   MS
## 1155   CA   MS
## 1156   CO   MS
## 1157   CT   MS
## 1158   DE   MS
## 1159   FL   MS
```

```
## 1160   GA   MS
## 1161   HI   MS
## 1162   ID   MS
## 1163   IL   MS
## 1164   IN   MS
## 1165   IA   MS
## 1166   KS   MS
## 1167   KY   MS
## 1168   LA   MS
## 1169   ME   MS
## 1170   MD   MS
## 1171   MA   MS
## 1172   MI   MS
## 1173   MN   MS
## 1174   MS   MS
## 1175   MO   MS
## 1176   MT   MS
## 1177   NE   MS
## 1178   NV   MS
## 1179   NH   MS
## 1180   NJ   MS
## 1181   NM   MS
## 1182   NY   MS
## 1183   NC   MS
## 1184   ND   MS
## 1185   OH   MS
## 1186   OK   MS
## 1187   OR   MS
## 1188   PA   MS
## 1189   RI   MS
## 1190   SC   MS
## 1191   SD   MS
## 1192   TN   MS
## 1193   TX   MS
## 1194   UT   MS
## 1195   VT   MS
## 1196   VA   MS
## 1197   WA   MS
## 1198   WV   MS
## 1199   WI   MS
## 1200   WY   MS
## 1201   AL   MO
## 1202   AK   MO
## 1203   AZ   MO
## 1204   AR   MO
## 1205   CA   MO
## 1206   CO   MO
## 1207   CT   MO
## 1208   DE   MO
## 1209   FL   MO
## 1210   GA   MO
## 1211   HI   MO
## 1212   ID   MO
## 1213   IL   MO
```

```
## 1214   IN   MO
## 1215   IA   MO
## 1216   KS   MO
## 1217   KY   MO
## 1218   LA   MO
## 1219   ME   MO
## 1220   MD   MO
## 1221   MA   MO
## 1222   MI   MO
## 1223   MN   MO
## 1224   MS   MO
## 1225   MO   MO
## 1226   MT   MO
## 1227   NE   MO
## 1228   NV   MO
## 1229   NH   MO
## 1230   NJ   MO
## 1231   NM   MO
## 1232   NY   MO
## 1233   NC   MO
## 1234   ND   MO
## 1235   OH   MO
## 1236   OK   MO
## 1237   OR   MO
## 1238   PA   MO
## 1239   RI   MO
## 1240   SC   MO
## 1241   SD   MO
## 1242   TN   MO
## 1243   TX   MO
## 1244   UT   MO
## 1245   VT   MO
## 1246   VA   MO
## 1247   WA   MO
## 1248   WV   MO
## 1249   WI   MO
## 1250   WY   MO
## 1251   AL   MT
## 1252   AK   MT
## 1253   AZ   MT
## 1254   AR   MT
## 1255   CA   MT
## 1256   CO   MT
## 1257   CT   MT
## 1258   DE   MT
## 1259   FL   MT
## 1260   GA   MT
## 1261   HI   MT
## 1262   ID   MT
## 1263   IL   MT
## 1264   IN   MT
## 1265   IA   MT
## 1266   KS   MT
## 1267   KY   MT
```

```
## 1268   LA   MT
## 1269   ME   MT
## 1270   MD   MT
## 1271   MA   MT
## 1272   MI   MT
## 1273   MN   MT
## 1274   MS   MT
## 1275   MO   MT
## 1276   MT   MT
## 1277   NE   MT
## 1278   NV   MT
## 1279   NH   MT
## 1280   NJ   MT
## 1281   NM   MT
## 1282   NY   MT
## 1283   NC   MT
## 1284   ND   MT
## 1285   OH   MT
## 1286   OK   MT
## 1287   OR   MT
## 1288   PA   MT
## 1289   RI   MT
## 1290   SC   MT
## 1291   SD   MT
## 1292   TN   MT
## 1293   TX   MT
## 1294   UT   MT
## 1295   VT   MT
## 1296   VA   MT
## 1297   WA   MT
## 1298   WV   MT
## 1299   WI   MT
## 1300   WY   MT
## 1301   AL   NE
## 1302   AK   NE
## 1303   AZ   NE
## 1304   AR   NE
## 1305   CA   NE
## 1306   CO   NE
## 1307   CT   NE
## 1308   DE   NE
## 1309   FL   NE
## 1310   GA   NE
## 1311   HI   NE
## 1312   ID   NE
## 1313   IL   NE
## 1314   IN   NE
## 1315   IA   NE
## 1316   KS   NE
## 1317   KY   NE
## 1318   LA   NE
## 1319   ME   NE
## 1320   MD   NE
## 1321   MA   NE
```

```
## 1322   MI   NE
## 1323   MN   NE
## 1324   MS   NE
## 1325   MO   NE
## 1326   MT   NE
## 1327   NE   NE
## 1328   NV   NE
## 1329   NH   NE
## 1330   NJ   NE
## 1331   NM   NE
## 1332   NY   NE
## 1333   NC   NE
## 1334   ND   NE
## 1335   OH   NE
## 1336   OK   NE
## 1337   OR   NE
## 1338   PA   NE
## 1339   RI   NE
## 1340   SC   NE
## 1341   SD   NE
## 1342   TN   NE
## 1343   TX   NE
## 1344   UT   NE
## 1345   VT   NE
## 1346   VA   NE
## 1347   WA   NE
## 1348   WV   NE
## 1349   WI   NE
## 1350   WY   NE
## 1351   AL   NV
## 1352   AK   NV
## 1353   AZ   NV
## 1354   AR   NV
## 1355   CA   NV
## 1356   CO   NV
## 1357   CT   NV
## 1358   DE   NV
## 1359   FL   NV
## 1360   GA   NV
## 1361   HI   NV
## 1362   ID   NV
## 1363   IL   NV
## 1364   IN   NV
## 1365   IA   NV
## 1366   KS   NV
## 1367   KY   NV
## 1368   LA   NV
## 1369   ME   NV
## 1370   MD   NV
## 1371   MA   NV
## 1372   MI   NV
## 1373   MN   NV
## 1374   MS   NV
## 1375   MO   NV
```

```
## 1376   MT   NV
## 1377   NE   NV
## 1378   NV   NV
## 1379   NH   NV
## 1380   NJ   NV
## 1381   NM   NV
## 1382   NY   NV
## 1383   NC   NV
## 1384   ND   NV
## 1385   OH   NV
## 1386   OK   NV
## 1387   OR   NV
## 1388   PA   NV
## 1389   RI   NV
## 1390   SC   NV
## 1391   SD   NV
## 1392   TN   NV
## 1393   TX   NV
## 1394   UT   NV
## 1395   VT   NV
## 1396   VA   NV
## 1397   WA   NV
## 1398   WV   NV
## 1399   WI   NV
## 1400   WY   NV
## 1401   AL   NH
## 1402   AK   NH
## 1403   AZ   NH
## 1404   AR   NH
## 1405   CA   NH
## 1406   CO   NH
## 1407   CT   NH
## 1408   DE   NH
## 1409   FL   NH
## 1410   GA   NH
## 1411   HI   NH
## 1412   ID   NH
## 1413   IL   NH
## 1414   IN   NH
## 1415   IA   NH
## 1416   KS   NH
## 1417   KY   NH
## 1418   LA   NH
## 1419   ME   NH
## 1420   MD   NH
## 1421   MA   NH
## 1422   MI   NH
## 1423   MN   NH
## 1424   MS   NH
## 1425   MO   NH
## 1426   MT   NH
## 1427   NE   NH
## 1428   NV   NH
## 1429   NH   NH
```

```
## 1430   NJ   NH
## 1431   NM   NH
## 1432   NY   NH
## 1433   NC   NH
## 1434   ND   NH
## 1435   OH   NH
## 1436   OK   NH
## 1437   OR   NH
## 1438   PA   NH
## 1439   RI   NH
## 1440   SC   NH
## 1441   SD   NH
## 1442   TN   NH
## 1443   TX   NH
## 1444   UT   NH
## 1445   VT   NH
## 1446   VA   NH
## 1447   WA   NH
## 1448   WV   NH
## 1449   WI   NH
## 1450   WY   NH
## 1451   AL   NJ
## 1452   AK   NJ
## 1453   AZ   NJ
## 1454   AR   NJ
## 1455   CA   NJ
## 1456   CO   NJ
## 1457   CT   NJ
## 1458   DE   NJ
## 1459   FL   NJ
## 1460   GA   NJ
## 1461   HI   NJ
## 1462   ID   NJ
## 1463   IL   NJ
## 1464   IN   NJ
## 1465   IA   NJ
## 1466   KS   NJ
## 1467   KY   NJ
## 1468   LA   NJ
## 1469   ME   NJ
## 1470   MD   NJ
## 1471   MA   NJ
## 1472   MI   NJ
## 1473   MN   NJ
## 1474   MS   NJ
## 1475   MO   NJ
## 1476   MT   NJ
## 1477   NE   NJ
## 1478   NV   NJ
## 1479   NH   NJ
## 1480   NJ   NJ
## 1481   NM   NJ
## 1482   NY   NJ
## 1483   NC   NJ
```

```
## 1484   ND   NJ
## 1485   OH   NJ
## 1486   OK   NJ
## 1487   OR   NJ
## 1488   PA   NJ
## 1489   RI   NJ
## 1490   SC   NJ
## 1491   SD   NJ
## 1492   TN   NJ
## 1493   TX   NJ
## 1494   UT   NJ
## 1495   VT   NJ
## 1496   VA   NJ
## 1497   WA   NJ
## 1498   WV   NJ
## 1499   WI   NJ
## 1500   WY   NJ
## 1501   AL   NM
## 1502   AK   NM
## 1503   AZ   NM
## 1504   AR   NM
## 1505   CA   NM
## 1506   CO   NM
## 1507   CT   NM
## 1508   DE   NM
## 1509   FL   NM
## 1510   GA   NM
## 1511   HI   NM
## 1512   ID   NM
## 1513   IL   NM
## 1514   IN   NM
## 1515   IA   NM
## 1516   KS   NM
## 1517   KY   NM
## 1518   LA   NM
## 1519   ME   NM
## 1520   MD   NM
## 1521   MA   NM
## 1522   MI   NM
## 1523   MN   NM
## 1524   MS   NM
## 1525   MO   NM
## 1526   MT   NM
## 1527   NE   NM
## 1528   NV   NM
## 1529   NH   NM
## 1530   NJ   NM
## 1531   NM   NM
## 1532   NY   NM
## 1533   NC   NM
## 1534   ND   NM
## 1535   OH   NM
## 1536   OK   NM
## 1537   OR   NM
```

```
## 1538   PA   NM
## 1539   RI   NM
## 1540   SC   NM
## 1541   SD   NM
## 1542   TN   NM
## 1543   TX   NM
## 1544   UT   NM
## 1545   VT   NM
## 1546   VA   NM
## 1547   WA   NM
## 1548   WV   NM
## 1549   WI   NM
## 1550   WY   NM
## 1551   AL   NY
## 1552   AK   NY
## 1553   AZ   NY
## 1554   AR   NY
## 1555   CA   NY
## 1556   CO   NY
## 1557   CT   NY
## 1558   DE   NY
## 1559   FL   NY
## 1560   GA   NY
## 1561   HI   NY
## 1562   ID   NY
## 1563   IL   NY
## 1564   IN   NY
## 1565   IA   NY
## 1566   KS   NY
## 1567   KY   NY
## 1568   LA   NY
## 1569   ME   NY
## 1570   MD   NY
## 1571   MA   NY
## 1572   MI   NY
## 1573   MN   NY
## 1574   MS   NY
## 1575   MO   NY
## 1576   MT   NY
## 1577   NE   NY
## 1578   NV   NY
## 1579   NH   NY
## 1580   NJ   NY
## 1581   NM   NY
## 1582   NY   NY
## 1583   NC   NY
## 1584   ND   NY
## 1585   OH   NY
## 1586   OK   NY
## 1587   OR   NY
## 1588   PA   NY
## 1589   RI   NY
## 1590   SC   NY
## 1591   SD   NY
```

```
## 1592   TN   NY
## 1593   TX   NY
## 1594   UT   NY
## 1595   VT   NY
## 1596   VA   NY
## 1597   WA   NY
## 1598   WV   NY
## 1599   WI   NY
## 1600   WY   NY
## 1601   AL   NC
## 1602   AK   NC
## 1603   AZ   NC
## 1604   AR   NC
## 1605   CA   NC
## 1606   CO   NC
## 1607   CT   NC
## 1608   DE   NC
## 1609   FL   NC
## 1610   GA   NC
## 1611   HI   NC
## 1612   ID   NC
## 1613   IL   NC
## 1614   IN   NC
## 1615   IA   NC
## 1616   KS   NC
## 1617   KY   NC
## 1618   LA   NC
## 1619   ME   NC
## 1620   MD   NC
## 1621   MA   NC
## 1622   MI   NC
## 1623   MN   NC
## 1624   MS   NC
## 1625   MO   NC
## 1626   MT   NC
## 1627   NE   NC
## 1628   NV   NC
## 1629   NH   NC
## 1630   NJ   NC
## 1631   NM   NC
## 1632   NY   NC
## 1633   NC   NC
## 1634   ND   NC
## 1635   OH   NC
## 1636   OK   NC
## 1637   OR   NC
## 1638   PA   NC
## 1639   RI   NC
## 1640   SC   NC
## 1641   SD   NC
## 1642   TN   NC
## 1643   TX   NC
## 1644   UT   NC
## 1645   VT   NC
```

```
## 1646   VA   NC
## 1647   WA   NC
## 1648   WV   NC
## 1649   WI   NC
## 1650   WY   NC
## 1651   AL   ND
## 1652   AK   ND
## 1653   AZ   ND
## 1654   AR   ND
## 1655   CA   ND
## 1656   CO   ND
## 1657   CT   ND
## 1658   DE   ND
## 1659   FL   ND
## 1660   GA   ND
## 1661   HI   ND
## 1662   ID   ND
## 1663   IL   ND
## 1664   IN   ND
## 1665   IA   ND
## 1666   KS   ND
## 1667   KY   ND
## 1668   LA   ND
## 1669   ME   ND
## 1670   MD   ND
## 1671   MA   ND
## 1672   MI   ND
## 1673   MN   ND
## 1674   MS   ND
## 1675   MO   ND
## 1676   MT   ND
## 1677   NE   ND
## 1678   NV   ND
## 1679   NH   ND
## 1680   NJ   ND
## 1681   NM   ND
## 1682   NY   ND
## 1683   NC   ND
## 1684   ND   ND
## 1685   OH   ND
## 1686   OK   ND
## 1687   OR   ND
## 1688   PA   ND
## 1689   RI   ND
## 1690   SC   ND
## 1691   SD   ND
## 1692   TN   ND
## 1693   TX   ND
## 1694   UT   ND
## 1695   VT   ND
## 1696   VA   ND
## 1697   WA   ND
## 1698   WV   ND
## 1699   WI   ND
```

```
## 1700   WY   ND
## 1701   AL   OH
## 1702   AK   OH
## 1703   AZ   OH
## 1704   AR   OH
## 1705   CA   OH
## 1706   CO   OH
## 1707   CT   OH
## 1708   DE   OH
## 1709   FL   OH
## 1710   GA   OH
## 1711   HI   OH
## 1712   ID   OH
## 1713   IL   OH
## 1714   IN   OH
## 1715   IA   OH
## 1716   KS   OH
## 1717   KY   OH
## 1718   LA   OH
## 1719   ME   OH
## 1720   MD   OH
## 1721   MA   OH
## 1722   MI   OH
## 1723   MN   OH
## 1724   MS   OH
## 1725   MO   OH
## 1726   MT   OH
## 1727   NE   OH
## 1728   NV   OH
## 1729   NH   OH
## 1730   NJ   OH
## 1731   NM   OH
## 1732   NY   OH
## 1733   NC   OH
## 1734   ND   OH
## 1735   OH   OH
## 1736   OK   OH
## 1737   OR   OH
## 1738   PA   OH
## 1739   RI   OH
## 1740   SC   OH
## 1741   SD   OH
## 1742   TN   OH
## 1743   TX   OH
## 1744   UT   OH
## 1745   VT   OH
## 1746   VA   OH
## 1747   WA   OH
## 1748   WV   OH
## 1749   WI   OH
## 1750   WY   OH
## 1751   AL   OK
## 1752   AK   OK
## 1753   AZ   OK
```

```
## 1754   AR   OK
## 1755   CA   OK
## 1756   CO   OK
## 1757   CT   OK
## 1758   DE   OK
## 1759   FL   OK
## 1760   GA   OK
## 1761   HI   OK
## 1762   ID   OK
## 1763   IL   OK
## 1764   IN   OK
## 1765   IA   OK
## 1766   KS   OK
## 1767   KY   OK
## 1768   LA   OK
## 1769   ME   OK
## 1770   MD   OK
## 1771   MA   OK
## 1772   MI   OK
## 1773   MN   OK
## 1774   MS   OK
## 1775   MO   OK
## 1776   MT   OK
## 1777   NE   OK
## 1778   NV   OK
## 1779   NH   OK
## 1780   NJ   OK
## 1781   NM   OK
## 1782   NY   OK
## 1783   NC   OK
## 1784   ND   OK
## 1785   OH   OK
## 1786   OK   OK
## 1787   OR   OK
## 1788   PA   OK
## 1789   RI   OK
## 1790   SC   OK
## 1791   SD   OK
## 1792   TN   OK
## 1793   TX   OK
## 1794   UT   OK
## 1795   VT   OK
## 1796   VA   OK
## 1797   WA   OK
## 1798   WV   OK
## 1799   WI   OK
## 1800   WY   OK
## 1801   AL   OR
## 1802   AK   OR
## 1803   AZ   OR
## 1804   AR   OR
## 1805   CA   OR
## 1806   CO   OR
## 1807   CT   OR
```

```
## 1808   DE   OR
## 1809   FL   OR
## 1810   GA   OR
## 1811   HI   OR
## 1812   ID   OR
## 1813   IL   OR
## 1814   IN   OR
## 1815   IA   OR
## 1816   KS   OR
## 1817   KY   OR
## 1818   LA   OR
## 1819   ME   OR
## 1820   MD   OR
## 1821   MA   OR
## 1822   MI   OR
## 1823   MN   OR
## 1824   MS   OR
## 1825   MO   OR
## 1826   MT   OR
## 1827   NE   OR
## 1828   NV   OR
## 1829   NH   OR
## 1830   NJ   OR
## 1831   NM   OR
## 1832   NY   OR
## 1833   NC   OR
## 1834   ND   OR
## 1835   OH   OR
## 1836   OK   OR
## 1837   OR   OR
## 1838   PA   OR
## 1839   RI   OR
## 1840   SC   OR
## 1841   SD   OR
## 1842   TN   OR
## 1843   TX   OR
## 1844   UT   OR
## 1845   VT   OR
## 1846   VA   OR
## 1847   WA   OR
## 1848   WV   OR
## 1849   WI   OR
## 1850   WY   OR
## 1851   AL   PA
## 1852   AK   PA
## 1853   AZ   PA
## 1854   AR   PA
## 1855   CA   PA
## 1856   CO   PA
## 1857   CT   PA
## 1858   DE   PA
## 1859   FL   PA
## 1860   GA   PA
## 1861   HI   PA
```

```
## 1862   ID   PA
## 1863   IL   PA
## 1864   IN   PA
## 1865   IA   PA
## 1866   KS   PA
## 1867   KY   PA
## 1868   LA   PA
## 1869   ME   PA
## 1870   MD   PA
## 1871   MA   PA
## 1872   MI   PA
## 1873   MN   PA
## 1874   MS   PA
## 1875   MO   PA
## 1876   MT   PA
## 1877   NE   PA
## 1878   NV   PA
## 1879   NH   PA
## 1880   NJ   PA
## 1881   NM   PA
## 1882   NY   PA
## 1883   NC   PA
## 1884   ND   PA
## 1885   OH   PA
## 1886   OK   PA
## 1887   OR   PA
## 1888   PA   PA
## 1889   RI   PA
## 1890   SC   PA
## 1891   SD   PA
## 1892   TN   PA
## 1893   TX   PA
## 1894   UT   PA
## 1895   VT   PA
## 1896   VA   PA
## 1897   WA   PA
## 1898   WV   PA
## 1899   WI   PA
## 1900   WY   PA
## 1901   AL   RI
## 1902   AK   RI
## 1903   AZ   RI
## 1904   AR   RI
## 1905   CA   RI
## 1906   CO   RI
## 1907   CT   RI
## 1908   DE   RI
## 1909   FL   RI
## 1910   GA   RI
## 1911   HI   RI
## 1912   ID   RI
## 1913   IL   RI
## 1914   IN   RI
## 1915   IA   RI
```

```
## 1916   KS   RI
## 1917   KY   RI
## 1918   LA   RI
## 1919   ME   RI
## 1920   MD   RI
## 1921   MA   RI
## 1922   MI   RI
## 1923   MN   RI
## 1924   MS   RI
## 1925   MO   RI
## 1926   MT   RI
## 1927   NE   RI
## 1928   NV   RI
## 1929   NH   RI
## 1930   NJ   RI
## 1931   NM   RI
## 1932   NY   RI
## 1933   NC   RI
## 1934   ND   RI
## 1935   OH   RI
## 1936   OK   RI
## 1937   OR   RI
## 1938   PA   RI
## 1939   RI   RI
## 1940   SC   RI
## 1941   SD   RI
## 1942   TN   RI
## 1943   TX   RI
## 1944   UT   RI
## 1945   VT   RI
## 1946   VA   RI
## 1947   WA   RI
## 1948   WV   RI
## 1949   WI   RI
## 1950   WY   RI
## 1951   AL   SC
## 1952   AK   SC
## 1953   AZ   SC
## 1954   AR   SC
## 1955   CA   SC
## 1956   CO   SC
## 1957   CT   SC
## 1958   DE   SC
## 1959   FL   SC
## 1960   GA   SC
## 1961   HI   SC
## 1962   ID   SC
## 1963   IL   SC
## 1964   IN   SC
## 1965   IA   SC
## 1966   KS   SC
## 1967   KY   SC
## 1968   LA   SC
## 1969   ME   SC
```

```
## 1970   MD   SC
## 1971   MA   SC
## 1972   MI   SC
## 1973   MN   SC
## 1974   MS   SC
## 1975   MO   SC
## 1976   MT   SC
## 1977   NE   SC
## 1978   NV   SC
## 1979   NH   SC
## 1980   NJ   SC
## 1981   NM   SC
## 1982   NY   SC
## 1983   NC   SC
## 1984   ND   SC
## 1985   OH   SC
## 1986   OK   SC
## 1987   OR   SC
## 1988   PA   SC
## 1989   RI   SC
## 1990   SC   SC
## 1991   SD   SC
## 1992   TN   SC
## 1993   TX   SC
## 1994   UT   SC
## 1995   VT   SC
## 1996   VA   SC
## 1997   WA   SC
## 1998   WV   SC
## 1999   WI   SC
## 2000   WY   SC
## 2001   AL   SD
## 2002   AK   SD
## 2003   AZ   SD
## 2004   AR   SD
## 2005   CA   SD
## 2006   CO   SD
## 2007   CT   SD
## 2008   DE   SD
## 2009   FL   SD
## 2010   GA   SD
## 2011   HI   SD
## 2012   ID   SD
## 2013   IL   SD
## 2014   IN   SD
## 2015   IA   SD
## 2016   KS   SD
## 2017   KY   SD
## 2018   LA   SD
## 2019   ME   SD
## 2020   MD   SD
## 2021   MA   SD
## 2022   MI   SD
## 2023   MN   SD
```

```
## 2024   MS   SD
## 2025   MO   SD
## 2026   MT   SD
## 2027   NE   SD
## 2028   NV   SD
## 2029   NH   SD
## 2030   NJ   SD
## 2031   NM   SD
## 2032   NY   SD
## 2033   NC   SD
## 2034   ND   SD
## 2035   OH   SD
## 2036   OK   SD
## 2037   OR   SD
## 2038   PA   SD
## 2039   RI   SD
## 2040   SC   SD
## 2041   SD   SD
## 2042   TN   SD
## 2043   TX   SD
## 2044   UT   SD
## 2045   VT   SD
## 2046   VA   SD
## 2047   WA   SD
## 2048   WV   SD
## 2049   WI   SD
## 2050   WY   SD
## 2051   AL   TN
## 2052   AK   TN
## 2053   AZ   TN
## 2054   AR   TN
## 2055   CA   TN
## 2056   CO   TN
## 2057   CT   TN
## 2058   DE   TN
## 2059   FL   TN
## 2060   GA   TN
## 2061   HI   TN
## 2062   ID   TN
## 2063   IL   TN
## 2064   IN   TN
## 2065   IA   TN
## 2066   KS   TN
## 2067   KY   TN
## 2068   LA   TN
## 2069   ME   TN
## 2070   MD   TN
## 2071   MA   TN
## 2072   MI   TN
## 2073   MN   TN
## 2074   MS   TN
## 2075   MO   TN
## 2076   MT   TN
## 2077   NE   TN
```

```
## 2078   NV   TN
## 2079   NH   TN
## 2080   NJ   TN
## 2081   NM   TN
## 2082   NY   TN
## 2083   NC   TN
## 2084   ND   TN
## 2085   OH   TN
## 2086   OK   TN
## 2087   OR   TN
## 2088   PA   TN
## 2089   RI   TN
## 2090   SC   TN
## 2091   SD   TN
## 2092   TN   TN
## 2093   TX   TN
## 2094   UT   TN
## 2095   VT   TN
## 2096   VA   TN
## 2097   WA   TN
## 2098   WV   TN
## 2099   WI   TN
## 2100   WY   TN
## 2101   AL   TX
## 2102   AK   TX
## 2103   AZ   TX
## 2104   AR   TX
## 2105   CA   TX
## 2106   CO   TX
## 2107   CT   TX
## 2108   DE   TX
## 2109   FL   TX
## 2110   GA   TX
## 2111   HI   TX
## 2112   ID   TX
## 2113   IL   TX
## 2114   IN   TX
## 2115   IA   TX
## 2116   KS   TX
## 2117   KY   TX
## 2118   LA   TX
## 2119   ME   TX
## 2120   MD   TX
## 2121   MA   TX
## 2122   MI   TX
## 2123   MN   TX
## 2124   MS   TX
## 2125   MO   TX
## 2126   MT   TX
## 2127   NE   TX
## 2128   NV   TX
## 2129   NH   TX
## 2130   NJ   TX
## 2131   NM   TX
```

```
## 2132   NY   TX
## 2133   NC   TX
## 2134   ND   TX
## 2135   OH   TX
## 2136   OK   TX
## 2137   OR   TX
## 2138   PA   TX
## 2139   RI   TX
## 2140   SC   TX
## 2141   SD   TX
## 2142   TN   TX
## 2143   TX   TX
## 2144   UT   TX
## 2145   VT   TX
## 2146   VA   TX
## 2147   WA   TX
## 2148   WV   TX
## 2149   WI   TX
## 2150   WY   TX
## 2151   AL   UT
## 2152   AK   UT
## 2153   AZ   UT
## 2154   AR   UT
## 2155   CA   UT
## 2156   CO   UT
## 2157   CT   UT
## 2158   DE   UT
## 2159   FL   UT
## 2160   GA   UT
## 2161   HI   UT
## 2162   ID   UT
## 2163   IL   UT
## 2164   IN   UT
## 2165   IA   UT
## 2166   KS   UT
## 2167   KY   UT
## 2168   LA   UT
## 2169   ME   UT
## 2170   MD   UT
## 2171   MA   UT
## 2172   MI   UT
## 2173   MN   UT
## 2174   MS   UT
## 2175   MO   UT
## 2176   MT   UT
## 2177   NE   UT
## 2178   NV   UT
## 2179   NH   UT
## 2180   NJ   UT
## 2181   NM   UT
## 2182   NY   UT
## 2183   NC   UT
## 2184   ND   UT
## 2185   OH   UT
```

```
## 2186   OK   UT
## 2187   OR   UT
## 2188   PA   UT
## 2189   RI   UT
## 2190   SC   UT
## 2191   SD   UT
## 2192   TN   UT
## 2193   TX   UT
## 2194   UT   UT
## 2195   VT   UT
## 2196   VA   UT
## 2197   WA   UT
## 2198   WV   UT
## 2199   WI   UT
## 2200   WY   UT
## 2201   AL   VT
## 2202   AK   VT
## 2203   AZ   VT
## 2204   AR   VT
## 2205   CA   VT
## 2206   CO   VT
## 2207   CT   VT
## 2208   DE   VT
## 2209   FL   VT
## 2210   GA   VT
## 2211   HI   VT
## 2212   ID   VT
## 2213   IL   VT
## 2214   IN   VT
## 2215   IA   VT
## 2216   KS   VT
## 2217   KY   VT
## 2218   LA   VT
## 2219   ME   VT
## 2220   MD   VT
## 2221   MA   VT
## 2222   MI   VT
## 2223   MN   VT
## 2224   MS   VT
## 2225   MO   VT
## 2226   MT   VT
## 2227   NE   VT
## 2228   NV   VT
## 2229   NH   VT
## 2230   NJ   VT
## 2231   NM   VT
## 2232   NY   VT
## 2233   NC   VT
## 2234   ND   VT
## 2235   OH   VT
## 2236   OK   VT
## 2237   OR   VT
## 2238   PA   VT
## 2239   RI   VT
```

```
## 2240   SC   VT
## 2241   SD   VT
## 2242   TN   VT
## 2243   TX   VT
## 2244   UT   VT
## 2245   VT   VT
## 2246   VA   VT
## 2247   WA   VT
## 2248   WV   VT
## 2249   WI   VT
## 2250   WY   VT
## 2251   AL   VA
## 2252   AK   VA
## 2253   AZ   VA
## 2254   AR   VA
## 2255   CA   VA
## 2256   CO   VA
## 2257   CT   VA
## 2258   DE   VA
## 2259   FL   VA
## 2260   GA   VA
## 2261   HI   VA
## 2262   ID   VA
## 2263   IL   VA
## 2264   IN   VA
## 2265   IA   VA
## 2266   KS   VA
## 2267   KY   VA
## 2268   LA   VA
## 2269   ME   VA
## 2270   MD   VA
## 2271   MA   VA
## 2272   MI   VA
## 2273   MN   VA
## 2274   MS   VA
## 2275   MO   VA
## 2276   MT   VA
## 2277   NE   VA
## 2278   NV   VA
## 2279   NH   VA
## 2280   NJ   VA
## 2281   NM   VA
## 2282   NY   VA
## 2283   NC   VA
## 2284   ND   VA
## 2285   OH   VA
## 2286   OK   VA
## 2287   OR   VA
## 2288   PA   VA
## 2289   RI   VA
## 2290   SC   VA
## 2291   SD   VA
## 2292   TN   VA
## 2293   TX   VA
```

```
## 2294   UT   VA
## 2295   VT   VA
## 2296   VA   VA
## 2297   WA   VA
## 2298   WV   VA
## 2299   WI   VA
## 2300   WY   VA
## 2301   AL   WA
## 2302   AK   WA
## 2303   AZ   WA
## 2304   AR   WA
## 2305   CA   WA
## 2306   CO   WA
## 2307   CT   WA
## 2308   DE   WA
## 2309   FL   WA
## 2310   GA   WA
## 2311   HI   WA
## 2312   ID   WA
## 2313   IL   WA
## 2314   IN   WA
## 2315   IA   WA
## 2316   KS   WA
## 2317   KY   WA
## 2318   LA   WA
## 2319   ME   WA
## 2320   MD   WA
## 2321   MA   WA
## 2322   MI   WA
## 2323   MN   WA
## 2324   MS   WA
## 2325   MO   WA
## 2326   MT   WA
## 2327   NE   WA
## 2328   NV   WA
## 2329   NH   WA
## 2330   NJ   WA
## 2331   NM   WA
## 2332   NY   WA
## 2333   NC   WA
## 2334   ND   WA
## 2335   OH   WA
## 2336   OK   WA
## 2337   OR   WA
## 2338   PA   WA
## 2339   RI   WA
## 2340   SC   WA
## 2341   SD   WA
## 2342   TN   WA
## 2343   TX   WA
## 2344   UT   WA
## 2345   VT   WA
## 2346   VA   WA
## 2347   WA   WA
```

```
## 2348   WV   WA
## 2349   WI   WA
## 2350   WY   WA
## 2351   AL   WV
## 2352   AK   WV
## 2353   AZ   WV
## 2354   AR   WV
## 2355   CA   WV
## 2356   CO   WV
## 2357   CT   WV
## 2358   DE   WV
## 2359   FL   WV
## 2360   GA   WV
## 2361   HI   WV
## 2362   ID   WV
## 2363   IL   WV
## 2364   IN   WV
## 2365   IA   WV
## 2366   KS   WV
## 2367   KY   WV
## 2368   LA   WV
## 2369   ME   WV
## 2370   MD   WV
## 2371   MA   WV
## 2372   MI   WV
## 2373   MN   WV
## 2374   MS   WV
## 2375   MO   WV
## 2376   MT   WV
## 2377   NE   WV
## 2378   NV   WV
## 2379   NH   WV
## 2380   NJ   WV
## 2381   NM   WV
## 2382   NY   WV
## 2383   NC   WV
## 2384   ND   WV
## 2385   OH   WV
## 2386   OK   WV
## 2387   OR   WV
## 2388   PA   WV
## 2389   RI   WV
## 2390   SC   WV
## 2391   SD   WV
## 2392   TN   WV
## 2393   TX   WV
## 2394   UT   WV
## 2395   VT   WV
## 2396   VA   WV
## 2397   WA   WV
## 2398   WV   WV
## 2399   WI   WV
## 2400   WY   WV
## 2401   AL   WI
```

```
## 2402   AK   WI
## 2403   AZ   WI
## 2404   AR   WI
## 2405   CA   WI
## 2406   CO   WI
## 2407   CT   WI
## 2408   DE   WI
## 2409   FL   WI
## 2410   GA   WI
## 2411   HI   WI
## 2412   ID   WI
## 2413   IL   WI
## 2414   IN   WI
## 2415   IA   WI
## 2416   KS   WI
## 2417   KY   WI
## 2418   LA   WI
## 2419   ME   WI
## 2420   MD   WI
## 2421   MA   WI
## 2422   MI   WI
## 2423   MN   WI
## 2424   MS   WI
## 2425   MO   WI
## 2426   MT   WI
## 2427   NE   WI
## 2428   NV   WI
## 2429   NH   WI
## 2430   NJ   WI
## 2431   NM   WI
## 2432   NY   WI
## 2433   NC   WI
## 2434   ND   WI
## 2435   OH   WI
## 2436   OK   WI
## 2437   OR   WI
## 2438   PA   WI
## 2439   RI   WI
## 2440   SC   WI
## 2441   SD   WI
## 2442   TN   WI
## 2443   TX   WI
## 2444   UT   WI
## 2445   VT   WI
## 2446   VA   WI
## 2447   WA   WI
## 2448   WV   WI
## 2449   WI   WI
## 2450   WY   WI
## 2451   AL   WY
## 2452   AK   WY
## 2453   AZ   WY
## 2454   AR   WY
## 2455   CA   WY
```

```
## 2456   CO   WY
## 2457   CT   WY
## 2458   DE   WY
## 2459   FL   WY
## 2460   GA   WY
## 2461   HI   WY
## 2462   ID   WY
## 2463   IL   WY
## 2464   IN   WY
## 2465   IA   WY
## 2466   KS   WY
## 2467   KY   WY
## 2468   LA   WY
## 2469   ME   WY
## 2470   MD   WY
## 2471   MA   WY
## 2472   MI   WY
## 2473   MN   WY
## 2474   MS   WY
## 2475   MO   WY
## 2476   MT   WY
## 2477   NE   WY
## 2478   NV   WY
## 2479   NH   WY
## 2480   NJ   WY
## 2481   NM   WY
## 2482   NY   WY
## 2483   NC   WY
## 2484   ND   WY
## 2485   OH   WY
## 2486   OK   WY
## 2487   OR   WY
## 2488   PA   WY
## 2489   RI   WY
## 2490   SC   WY
## 2491   SD   WY
## 2492   TN   WY
## 2493   TX   WY
## 2494   UT   WY
## 2495   VT   WY
## 2496   VA   WY
## 2497   WA   WY
## 2498   WV   WY
## 2499   WI   WY
## 2500   WY   WY
```

But this will not work for grid_words, because it's too large

```
## NULL
```

This seems to be an example of an operation better accomplished through loops, since it's computationally intensive and we don't have the resources to run a vectorized version of this script.

Since loops break up the computation, it's possible to run long processes (with interruptions, although I'm not showing that).

```
# For the purposes of this workshop, you may want to down-sample before this operation (start small)
loop_words = sample( grid_words, 100 )

# Parallel loop of loops to compare
{tic("Exhaustive similarity comparisons")  # timing starts
reviews.compared =
  foreach( loop.outer = seq_along( loop_words ), .combine='rbind' ) %:%
    foreach( loop.inner = seq_along( loop_words ), .combine='c', .packages = 'dplyr' ) %do% {

      # Count words total and words in common, using set theory (fast)
      words.inner     = length( loop_words[ loop.inner ][[1]] )
      words.total     = length( union( loop_words[ loop.outer ][[1]], loop_words[ loop.inner ][[1]] ))
      words.in.common = length( intersect( loop_words[ loop.outer ][[1]], loop_words[ loop.inner ][[1]]

      # Calculate the.score
      # the.score = words.in.common / words.total words.inner
      the.score = words.in.common / words.inner

      # Return the output
      return( the.score )
    }
toc()}  # timing ends
```

```
## Exhaustive similarity comparisons: 6.158 sec elapsed
```

```
# DING
# beep()
```

Benchmarks

10. . . . .0.1s 100. . . 3.3s 500. . . 73.7s 1k. . . 321s 2k. . . 1252s

---

Coding Challenge B

The second challenge involves finding the mean `$Rating` of each city/state combination. Create an outer loop that lists the cities for which there are reviews, then create an inner loop that calculates the mean `$Rating` on each iteration. Here's some meta-code to get you started thinking about this problem:

Outer Loop `foreach(state) %:% city_list = unique cities in state`

Inner Loop `foreach(city) %dorpar% calculate the mean review$Rating`

Good luck!

```
write_rds( reviews, "./data/snapshot_b.rds")  # in coding_challenges you will load this file
```

Bootstrapping

So far the examples have focused on `for` loops iterating over lists and dataframes, but this is not the whole story. For many purposes, you might want to use a `while` loop, which terminates when/while certain conditions are being met.

Another common use for loops is bootstrapping, which involves running a process for a specified number of times. For example, we can run the same model 1,000 times over different subsets of the same data and compare the results...

```
# Let's time our bootstrap loop
{
tic("Bootstrapping a linear model n times")

bootstrap = foreach( icount( 1000 ),  # how many loops?
                     .combine  = rbind,  # each run of the model will generate 1 row
                     .packages = c ("dplyr", "lme4")) %dopar% {  # parallel mode
                     #.packages = c ("dplyr", "lme4")) %do% {  # non-parallel mode

    # Sample some random rows to make things run faster
    sample = reviews %>% sample_n(nrow(reviews) * 0.33, replace = FALSE)

    # Fit the model predicting sentiment based on length and rating
    model = lm( data    = sample,
                formula = unlist(Rating) ~ unlist(nchar) )

    # Extract model coefficients
    coefs = coefficients( model )

    # Output
    return( coefs )
  }

toc()
}
```

```
## Bootstrapping a linear model n times: 42.816 sec elapsed
```

```
# We can get the mean of each coefficient as follows
colMeans( bootstrap )
```

```
##   (Intercept) unlist(nchar)
##   3.477841008  -0.002841132
```

The parallel version typically runs 33% faster for me.

Coding Challenge C

For this challenge, you will be bootstrapping a model of review sentiment, based on a quick text analysis that results in each review being assigned a score based on how positive or negative, according to an unsupervised

algorithm. We will be trying to predict the sentiment based on the rating and review length. Here's some meta-code to get you thinking about this problem:

Sentiment data generation `sentimentr( review_text )`

Loop a linear model of sentiment `lm( sentiment ~ rating + length )`

Abstract over the results `colMeans(bootstrapped_results)`

Part 6: Final thoughts

I don't often write loops, but when I do, they run in parallel.

To get the most out of parallel loops, you will need to have access to as many cores (processors) as possible. Consider using remote computing resources, such as Quest (Northwestern). For the best performance, use vectorized code whenever possible, such as the functions provided in `tidyverse` and `data.table`, rather than loops.

That said, many processes are easy to write as loops because the code runs sequentially, which is easy to conceptualize. Rewriting loops using vectorized functions essentially gives you more power as a researcher, because it unlocks the computing power you already have. More importantly, *it saves your time!*

Happy looping looping looping!

One last step

Best practice would be to close the script with stopping the cluster...

```
# This severs the connection between R and the 'extra' cores we were leveraging.
stopCluster( cl )
```