

Figure 1. Overview of potential variant creation.

FIG. 1

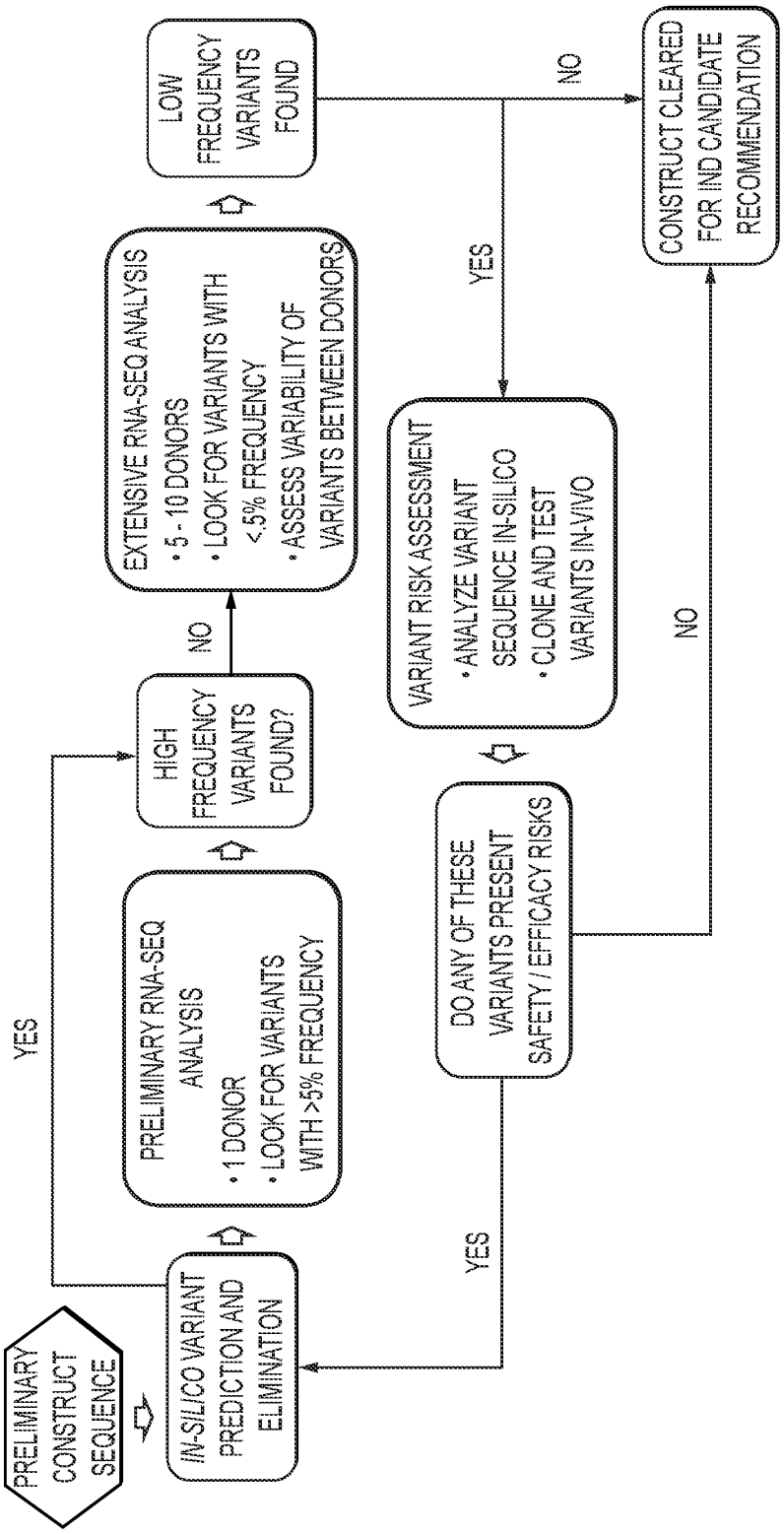

PRELIMINARY CONSTRUCT SEQUENCE
IN-SILICO VARIANT PREDICTION AND ELIMINATION
PRELIMINARY RNA-SEQ ANALYSIS
• 1 DONOR
• LOOK FOR VARIANTS WITH >5% FREQUENCY
HIGH FREQUENCY VARIANTS FOUND?
YES
NO
EXTENSIVE RNA-SEQ ANALYSIS
• 5 - 10 DONORS
• LOOK FOR VARIANTS WITH <5% FREQUENCY
• ASSESS VARIABILITY OF VARIANTS BETWEEN DONORS
LOW FREQUENCY VARIANTS FOUND
DO ANY OF THESE VARIANTS PRESENT SAFETY / EFFICACY RISKS
YES
NO
VARIANT RISK ASSESSMENT
• ANALYZE VARIANT SEQUENCE IN-SILICO
• CLONE AND TEST VARIANTS IN-VIVO
YES
NO
CONSTRUCT CLEARED FOR IND CANDIDATE RECOMMENDATION

FIG. 2

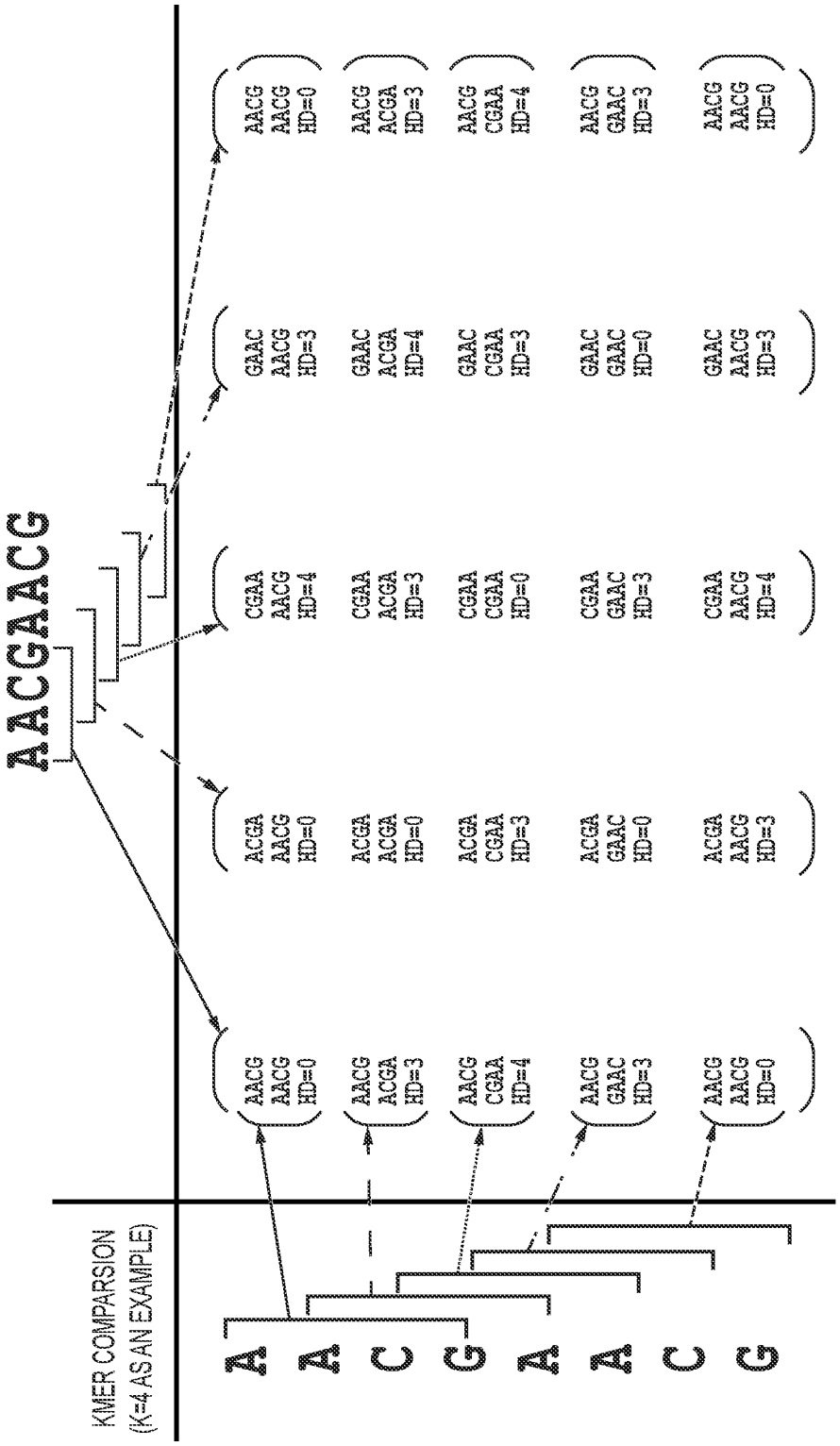

AACGAACG
KMER COMPARSION
(K=4 AS AN EXAMPLE)
A A C G A A C G
AACG AACG HD=0
AACG ACGA HD=3
AACG CGAA HD=4
AACG GAAC HD=3
AACG AACG HD=0
ACGA AACG HD=0
ACGA ACGA HD=3
ACGA CGAA HD=3
ACGA GAAC HD=0
ACGA AACG HD=3
CGAA AACG HD=4
CGAA ACGA HD=3
CGAA CGAA HD=0
CGAA GAAC HD=3
CGAA AACG HD=4
GAAC AACG HD=3
GAAC ACGA HD=4
GAAC CGAA HD=3
GAAC GAAC HD=0
GAAC AACG HD=3
AACG AACG HD=0
AACG ACGA HD=3
AACG CGAA HD=4
AACG GAAC HD=3
AACG AACG HD=0

FIG. 3

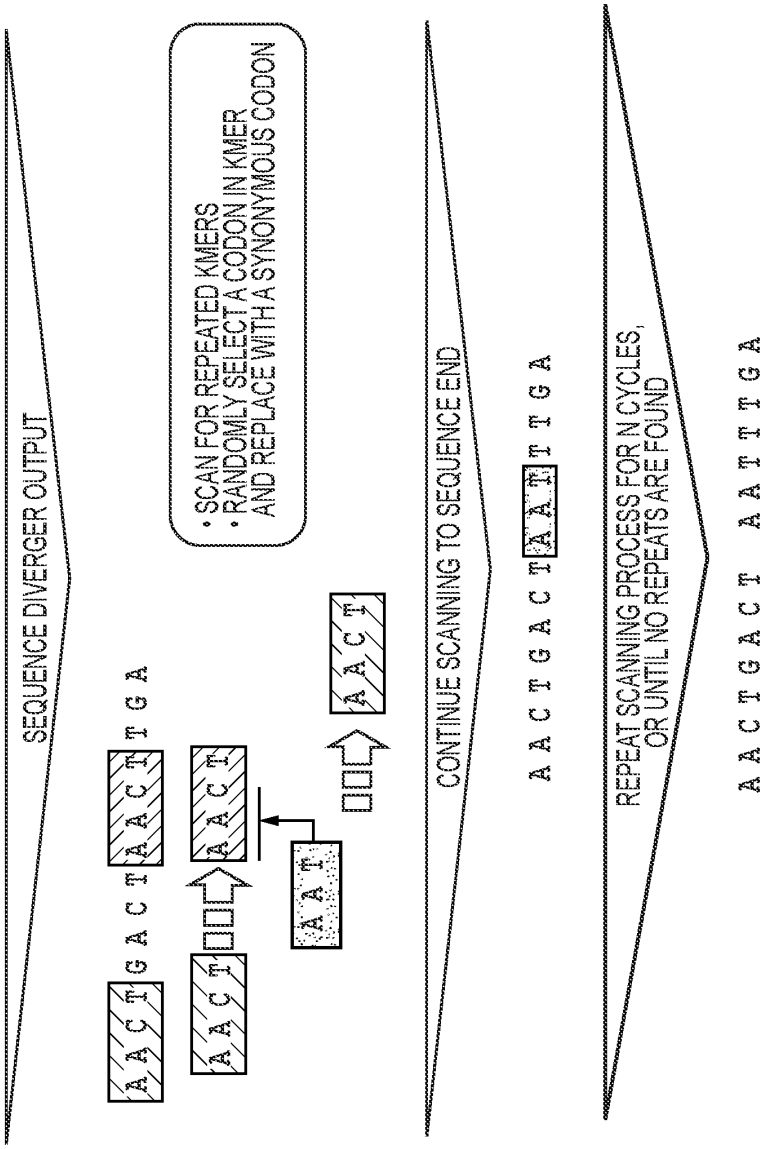

SEQUENCE DIVERGER OUTPUT
AACTGACTAACTTGA
AACT
AACT
AAT
AACT
• SCAN FOR REPEATED KMERS
• RANDOMLY SELECT A CODON IN KMER AND REPLACE WITH A SYNONYMOUS CODON
CONTINUE SCANNING TO SEQUENCE END
AACTGACTAATTGA
REPEAT SCANNING PROCESS FOR N CYCLES, OR UNTIL NO REPEATS ARE FOUND
AACTGACT AATTGA

FIG. 4

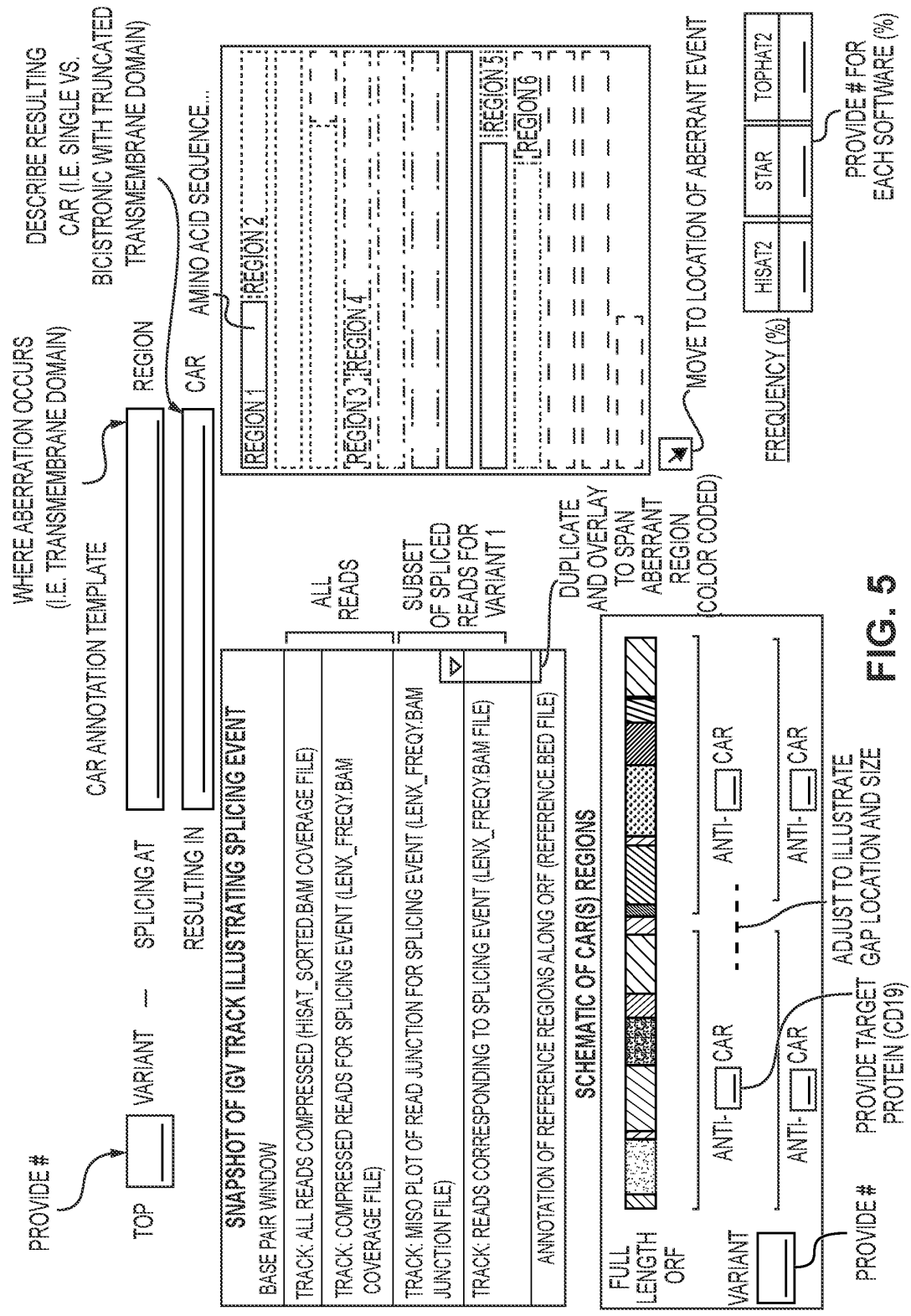

PROVIDE #
TOP
VARIANT
SPLICING AT
RESULTING IN
CAR ANNOTATION TEMPLATE
SPECIFY CAR REGION WHERE ABERRATION OCCURS (I.E. TRANSMEMBRANE DOMAIN)
REGION
CAR
AMINO ACID SEQUENCE...
DESCRIBE RESULTING CAR (I.E. SINGLE VS. BICISTRONIC WITH TRUNCATED TRANSMEMBRANE DOMAIN)
REGION 1
REGION 2
REGION 3
REGION 4
REGION 5
REGION 6
SNAPSHOT OF IGV TRACK ILLUSTRATING SPLICING EVENT
BASE PAIR WINDOW
TRACK: ALL READS COMPRESSED (HISAT_SORTED.BAM COVERAGE FILE)
TRACK: COMPRESSED READS FOR SPLICING EVENT (LENX_FREQY.BAM COVERAGE FILE)
TRACK: MISO PLOT OF READ JUNCTION FOR SPLICING EVENT (LENX_FREQY.BAM JUNCTION FILE)
TRACK: READS CORRESPONDING TO SPLICING EVENT (LENX_FREQY.BAM FILE)
ANNOTATION OF REFERENCE REGIONS ALONG ORF (REFERENCE.BED FILE)
ALL READS
SUBSET OF SPLICED READS FOR VARIANT 1
DUPLICATE AND OVERLAY TO SPAN ABERRANT REGION (COLOR CODED)
MOVE TO LOCATION OF ABERRANT EVENT
SCHEMATIC OF CAR(S) REGIONS
FULL LENGTH ORF
VARIANT
ANTI- CAR
ANTI- CAR
ANTI- CAR
ANTI- CAR
PROVIDE #
PROVIDE TARGET PROTEIN (CD19)
ADJUST TO ILLUSTRATE GAP LOCATION AND SIZE
FREQUENCY (%)
HISAT2
STAR
TOPHAT2
PROVIDE # FOR EACH SOFTWARE (%)

FIG. 5

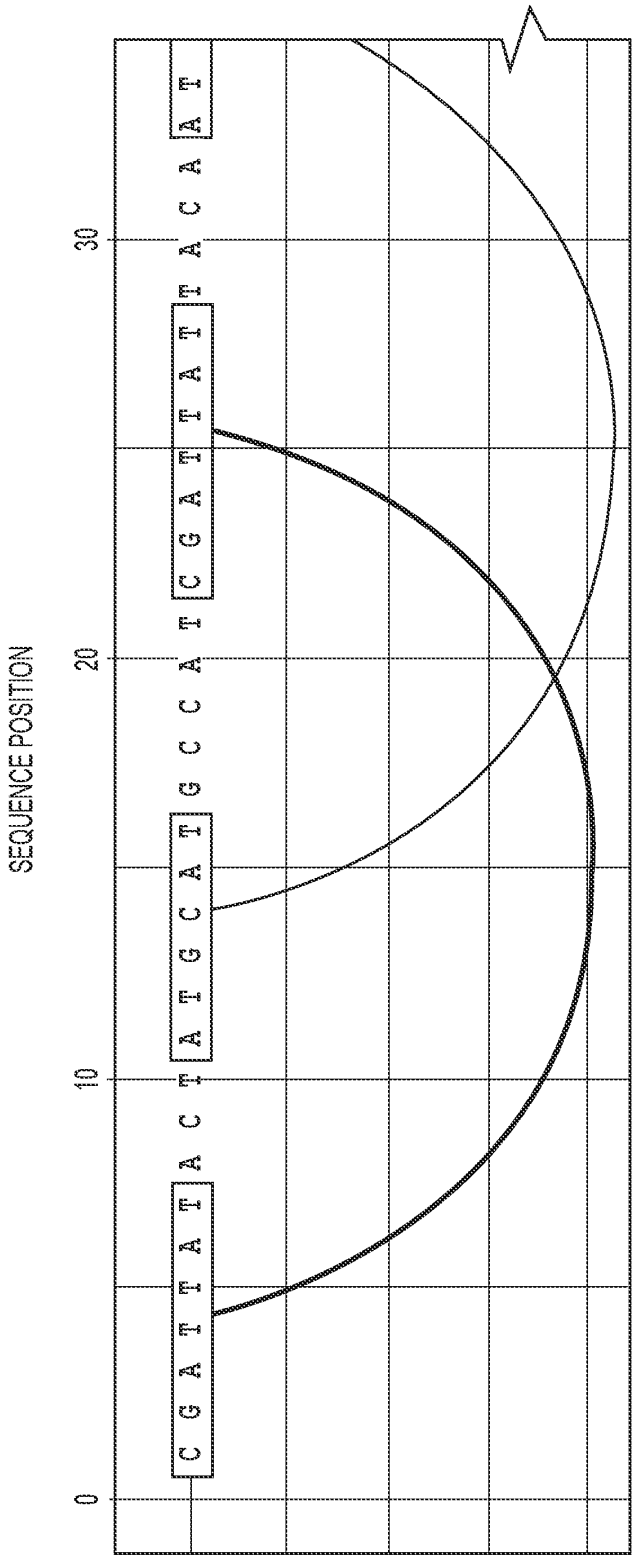

SEQUENCE POSITION
0
10
20
30
C G A T T A T A C T A T G C A T G C C A T C G A T T A T A C A A T

FIG. 6

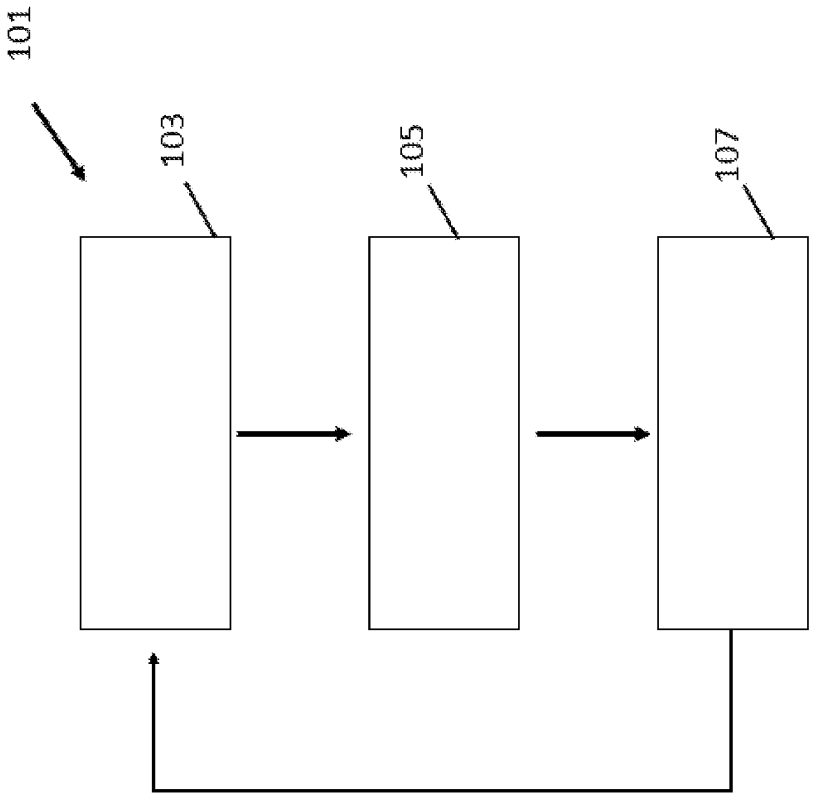


FIG. 7A

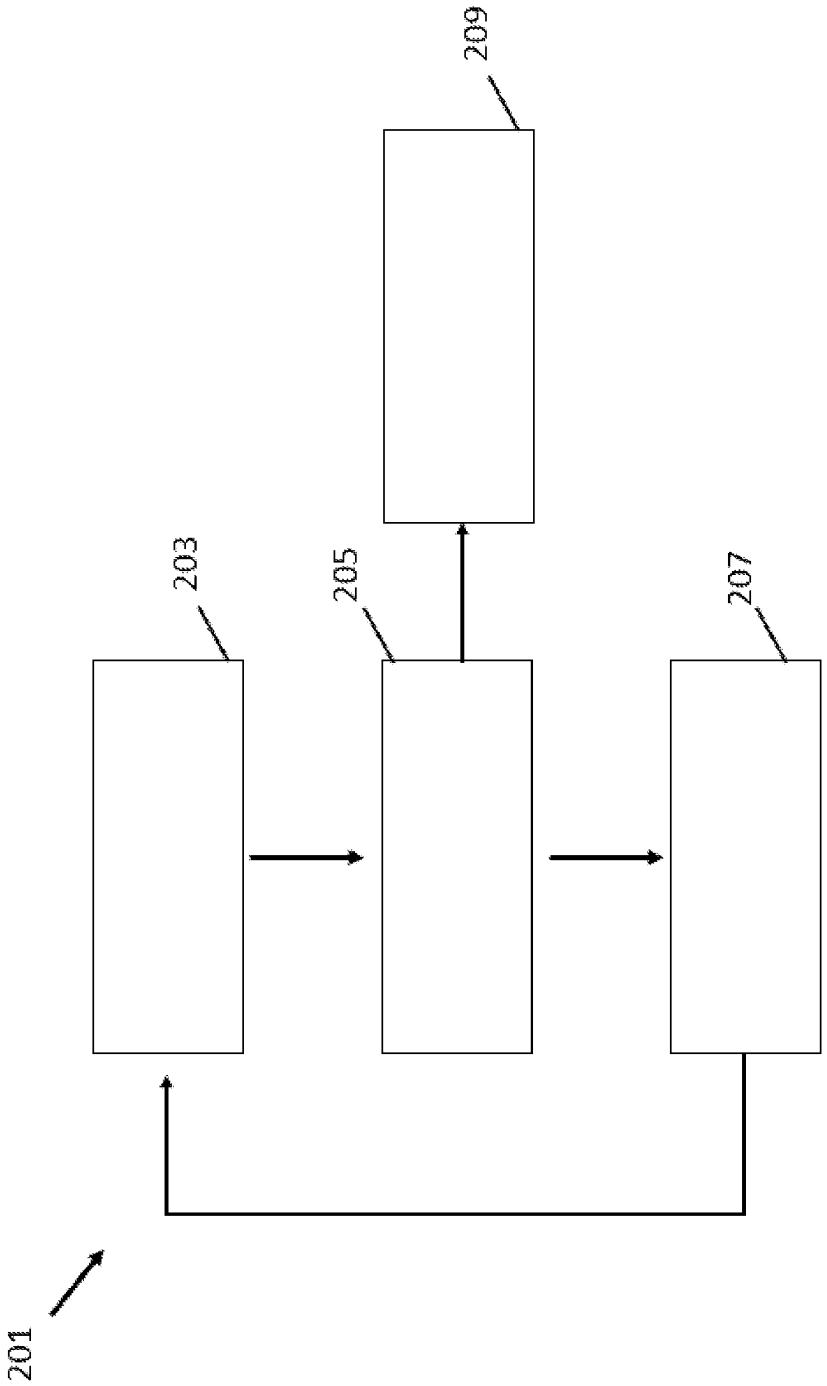
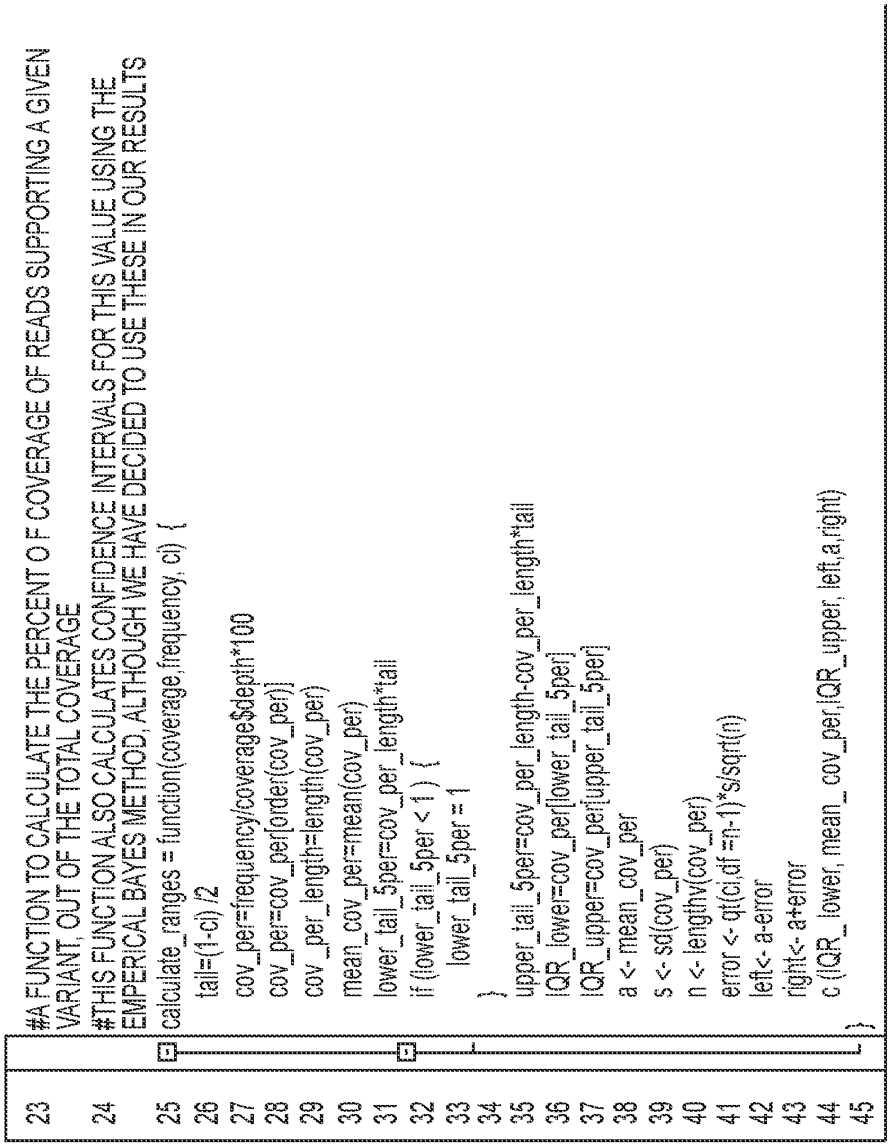


FIG. 7B



```
#A FUNCTION TO CALCULATE THE PERCENT O F COVERAGE OF READS SUPPORTING A GIVEN VARIANT, OUT OF THE TOTAL COVERAGE
#THIS FUNCTION ALSO CALCULATES CONFIDENCE INTERVALS FOR THIS VALUE USING THE EMPERICAL BAYES METHOD, ALTHOUGH WE HAVE DECIDED TO USE THESE IN OUR RESULTS
calculate_ranges = function(coverage,frequency, ci) {
    tail=(1-ci) /2
    cov_per=frequency/coverage$depth*100
    cov_per=cov_per[order(cov_per)]
    cov_per_length=length(cov_per)
    mean_cov_per=mean(cov_per)
    lower_tail_5per=cov_per_length*tail
    if (lower_tail_5per < 1 ) {
        lower_tail_5per = 1
    }
    upper_tail_5per=cov_per_length-cov_per_length*tail
    IQR_lower=cov_per[lower_tail_5per]
    IQR_upper=cov_per[upper_tail_5per]
    a <- mean_cov_per
    s <- sd(cov_per)
    n <- lengthv(cov_per)
    error <- qt(ci,df=n-1)*s/sqrt(n)
    left<- a-error
    right<- a+error
    c (IQR_ lower, mean_ cov_per,IQR_upper, left,a,right)
}
```

FIG. 8A

```
#A FUNCTION TO GET THE COUNTS AND PERCENT COVERAGE OF EACH VARIANT FOUND IN A SAMPLE
#THIS FUNCTION CAN ALSO MAKE BAM FILES OF READS SUPPORTING EACH VARIANT WITH THE MAKEGAPBAMS PARAMETER
#THIS PARAMETER IS OPTIONAL BECUASE MAKING THESE FILES BECOMES COMPUTATIONALLY INFEASIBLE WITH WITH LARGE NUMBERS OF READS
process_ sub_task = function(subTask, bamFlanks, readFlanks, dataDirectory, mainOutDirectory, makeGapBams, samTools, orfStart, orfStop
  , interval) {
  setwd (dataDirectory)
  bamFile = paste0(bamFlanks[1], subTask, bamFlanks[2])
  readFile = paste0(readFlanks[1], subTask, readFlanks[2])
  if (file.info(readFile)$size >0) {
    outDirectory = paste0(mainOutDirectory, subTask, "/" )
    system (p aste0("mkdir " , paste0(mainOutDirectory, subTask, "/" )))
    readTable = read.csv(readFile, stringsAsFactors = FALSE)
    readTable = readTable[!is.na(readTable$gapped_ translated),]
    readTable = readTable[unlist (lapply (readTable$V2, function(x) !any (samFlags(x) [c ("READ_UNMAPPED" , "NOT_PRIMARY_ALIGNMENT" ,
      "READ FAILS_ VENDOR_QC" , "DUPLICATE_READ" ))])),]
    if (nrow (readTable) ! = 0) {
      freqTable = table (readTable$gapped_translated)
      freqTable = freqTable[order (freqTable, decreasing= TRUE), drop = FALSE]
      freqTable = as.data.frame(cbind(Variant = rownames (freqTable), Freq = as.numeric(freqTable)), stringsAsFactors = FALSE)
      if (makeGapBams) {
        freqTable$BAM_ File = make_ gap_ bams(readTable , freqTable$Variant, bamFile, outDirectory, samTools)
      }
      system (paste0 (samTools , "view -h " , bamFile, " | awk ' {if($0 ~ /^@/ | | $6 ~ /S/) {print $0}}' | samtools view -Sb - >" ,
      outDirectory, "no_clip_ bam.bam" ))
      coverage = system (paste0 (samTools , "depth -a -d 0 " , outDirectory, "no clip_ bam.bam" ), intern= TRUE)
      coverage=read . table( text=coverage, sep=" \t " )
      names (coverage) = c ( "construct" , "position" , "depth" )
      ranges_ ORF = t(apply (freqTa ble, 1 , function (x) calculate_ ranges(coverage[coverage$position >= orfStart & coverage$position <=
      orfStop,], as.numeric (x [ "Freq" ]) , interval)))
```

FIG. 8B

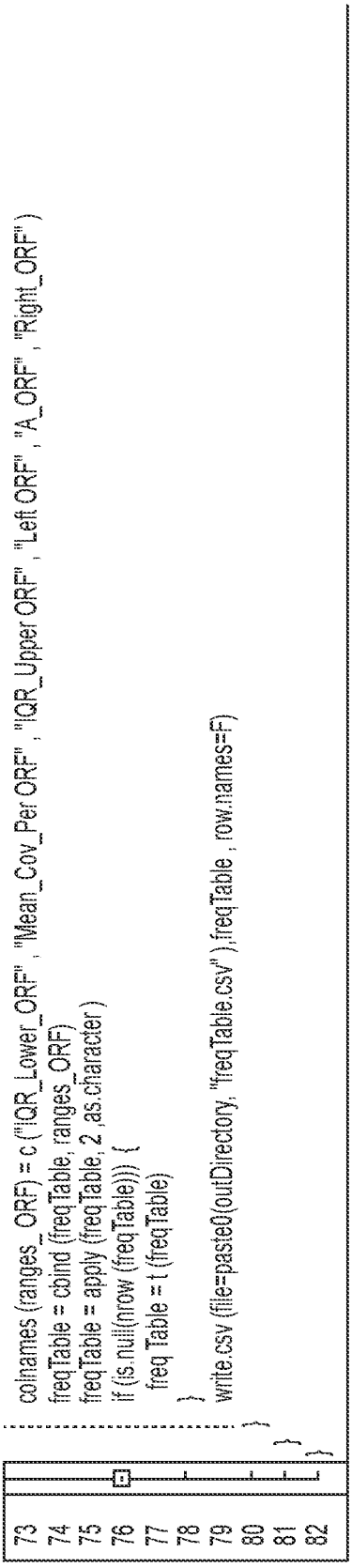
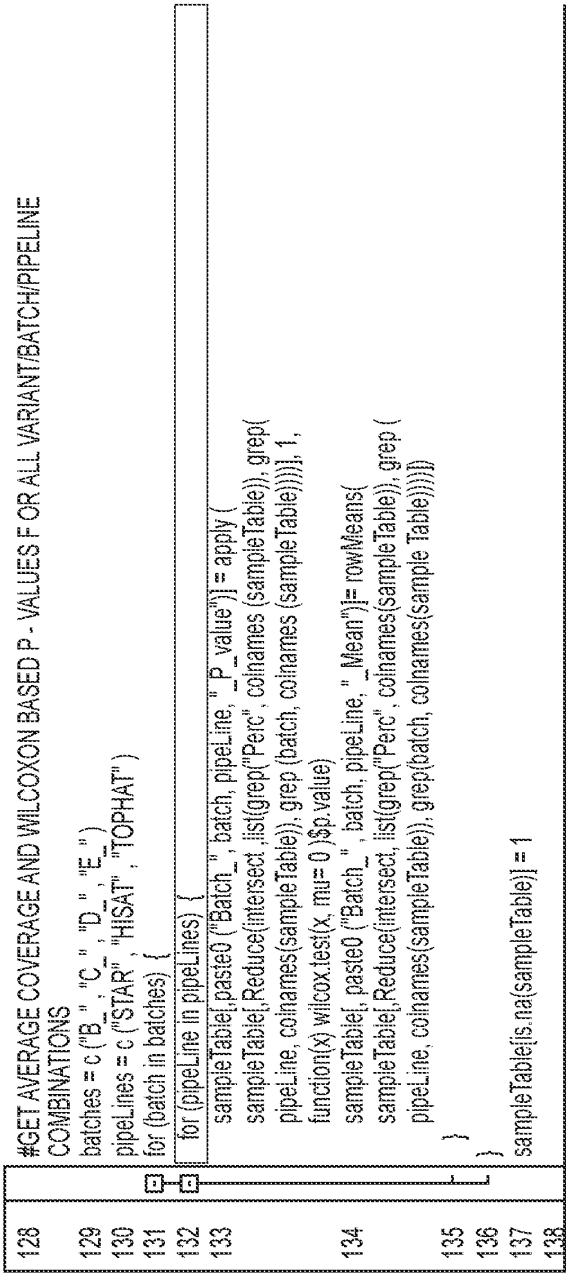

colnames (ranges_ORF) = c ("IQR_Lower_ORF", "Mean_Cov_Per ORF", "IQR_Upper ORF", "Left ORF", "A_ORF", "Right_ORF")
freqTable = cbind (freqTable, ranges_ORF)
freqTable = apply (freqTable, 2 ,as.character)
if (is.null(nrow (freqTable))) {
freq Table = t (freqTable)
}
write.csv (file=paste0(outDirectory, "freqTable.csv"),freqTable , row.names=F)
}
}
}

FIG. 8B
(CONT.)



```
#GET AVERAGE COVERAGE AND WILCOXON BASED P - VALUES F OR ALL VARIANT/BATCH/PIPELINE
COMBINATIONS
batches = c ("B_" , "C_" , "D_" , "E_" )
pipeLines = c ("STAR" , "HISAT" , "TOPHAT" )
for (batch in batches) {
  for (pipeLine in pipeLines) {
    sampleTable[,paste0 ("Batch_" , batch, pipeLine, "_P_value")] = apply (
    sampleTable[,Reduce(intersect ,list(grep("Perc" , colnames (sampleTable)), grep(
    pipeLine, colnames(sampleTable)), grep (batch, colnames (sampleTable))))], 1,
    function(x) wilcox.test(x, mu= 0 )$p.value)
    sampleTable[, paste0 ("Batch_" , batch, pipeLine, "_Mean")]= rowMeans(
    sampleTable[,Reduce(intersect, list(grep("Perc" , colnames(sampleTable)), grep (
    pipeLine, colnames(sampleTable)), grep(batch, colnames(sample Table)))))]
  }
}
sampleTable[is.na(sampleTable)] = 1
```

FIG. 8C