**\*\*\*程序说明\*\*\***

\*\*\*源程序中包含三个程序脚本，分别为：

\*\*\*demo\_run.sh

\*\*\*step1.MockReadsGenerator.R

\*\*\*step2.MakeFqFormat.sh

其中step1.MockReadsGenerator.R和step2.MakeFqFormat.sh分别是软件第一、二步运行的主程序。使用时可对demo\_run.sh示例程序进行输入文件和参数的修改，然后执行程序demo\_run.sh即可。

demo\_run.sh

#!/usr/bin/sh

## This is a demo for Software MockReadsGenerator

## Written by XiaoHan

## 2023-09-20

echo "start at `date`" &&

/jdfsbjcas1/ST\_BJ/P21H28400N0232/xiaohan2/Software/miniconda/envs/R4.2/bin/Rscript step1.MockReadsGenerator.R \

--Mode BaseRandom \

--Templates ./Demo/01-input/BaseRandom/i1.refseq.txt \

--SiteNums 300 \

--ReadLength 100 \

--NegativeRate 0.1 \

--BarcodeLens 8 \

--Outdir Demo/02-output/BaseRandom &&

/jdfsbjcas1/ST\_BJ/P21H28400N0232/xiaohan2/Software/miniconda/envs/R4.2/bin/Rscript step1.MockReadsGenerator.R \

--Mode TemplateRandom \

--Templates ./Demo/01-input/TemplateRandom/i1.RefSeqs.xlsx \

--SiteNums 500 \

--ReadLength 150 \

--Outdir Demo/02-output/TemplateRandom &&

bash step2.MakeFqFormat.sh Demo/02-output/BaseRandom/o1.BaseRandom\_demo.fq Demo/02-output/BaseRandom/o1.BaseRandom.fq &&

bash step2.MakeFqFormat.sh Demo/02-output/TemplateRandom/o1.TemplateRandom\_demo.fq Demo/02-output/TemplateRandom/o1.TemplateRandom.fq &&

echo "end at `date`" &&

echo "Still water run deep" 1>&2 &&

echo "Still water run deep" > demo.sign

step1.MockReadsGenerator.R

########################################################################################

# Author: xiaohan2@genomics.cn

# Date: 2023-09-20 12:00:00

# LastEditors: xiaohan2@genomics.cn

# LastEditTime: 2023-09-20 12:00:00

# Description: 生成抓取N8 barcode的参数测试集

# Mock dataset type I:

# 目的：滑动生成覆盖某个序列区间的mock reads（序列区间碱基随机生成）

# 输入：模板参考序列、序列区间长度、阴性对照reads率

# 输出：一组覆盖指定区间序列的mock reads，指定区间序列随机

# Mock dataset type II:

# 目的：滑动生成覆盖某个序列区间的mock reads（序列区间碱基通过指定）

# 输入：一组模板参考序列、序列区间长度

# 输出：一组覆盖指定区间序列的mock reads，指定区间序列从参考模板选取

# Test dataset1: 从barcode侧翼位点滑动的91个位点的100bp reads,

# 其中每个相同位点300条reads，270条含有随机barcode,30条不含随机barcode

# Test dataset2: 从39条参考序列区间中随机挑选一条，滑动149个位点的150bp reads，

# 其中每个相同位点500条reads

# Copyright 2023 by xiaohan2 All Rights Reserved.

########################################################################################

### >>> 0. 参数传递

#-------------------------------------------------------------------------------

library(optparse)

library(openxlsx)

library(stringr)

library(dplyr)

# 描述参数

option\_list <- list(

make\_option(c("--Mode"), type = "character", default = FALSE,

action = "store", help = "Mode: BaseRandom or TemplateRandom"

),

make\_option(c("--Templates"), type = "character", default = FALSE,

action = "store", help = "MotherSeq templates file path"

),

make\_option(c("--SiteNums"), type = "integer", default = FALSE,

action = "store", help = "The number of single site mock reads"

),

make\_option(c("--ReadLength"), type = "integer", default = FALSE,

action = "store", help = "The length of mock reads"

),

make\_option(c("--NegativeRate"), type = "double", default = FALSE,

action = "store", help = "ONLY for BaseRandom mode: the number of single site negative mock reads"

),

make\_option(c("--BarcodeLens"), type = "integer", default = FALSE,

action = "store", help = "ONLY for BaseRandom mode: the length of barcodes"

),

make\_option(c("--Outdir"), type = "character", default = FALSE,

action = "store", help = "Result directory"

),

make\_option(c("--Try"), type = "logical", default = FALSE,

action = "store\_TRUE", help = "This is demo and have a try!"

)

)

# 解析参数

args = parse\_args(OptionParser(option\_list = option\_list, usage = "This script is a test for arguments!"))

### >>> Main <<<

#-------------------------------------------------------------------------------

### Part 1: BaseRandom

if(args$Mode=="BaseRandom"){

if (args$Try) { # 程序测试使用

outdir <- "Demo/02-output/BaseRandom"

if(!dir.exists(outdir)){dir.create(outdir,recursive = T)}

# 模版序列需替换位置用N表示

mother\_seq <- "cctggagacctccgcgccccgcaacctccccttctacgagcggctcggcttcaccgtcaccgccgacgtcgaggtgcccgaaggaccgcgcacctggtgcatgacccgCAAGCCCGGTGCCTGATGCAGGCATATCAATAAGCGGAGGANNNNNNNNCGATATCTCGAGGGTACCTTTAAGACCAATGACTTACAAGGCAGCTGTAGATCTTAGCCACTTTTTAAAAGAAAAGGGGGGACTGGAAGGGCTAATTCACTCCCAACGAAGATAAGATCTGCTTTTTGCTTGTACTGGGTCTCTCTGGTTAGACCAGATCTGAGCCTGG"

# mother\_seq <- read.table("Demo/01-input/BaseRandom/i1.refseq.txt",header=F)$V1

n\_barcode\_len <- 8 # Barcode长度

n\_site\_reads <- 300 # 单个位点总reads数

neg\_rate <- 0.1 # 单个位点阴性对照数

read\_lens <- 100 # 单个reads长度

barcode.tag <- paste(rep("N",n\_barcode\_len),collapse="") # 模版序列中的N

} else {

outdir <- args$Outdir

if(!dir.exists(outdir)){dir.create(outdir,recursive = T)}

# 模版序列需替换位置用N表示

mother\_seq <- read.table(args$Templates,header=F)$V1 # 模板序列

n\_barcode\_len <- args$BarcodeLens # Barcode长度

n\_site\_reads <- args$SiteNums # 单个位点总reads数

neg\_rate <- args$NegativeRate # 单个位点阴性对照数

read\_lens <- args$ReadLength # 单个reads长度

barcode.tag <- paste(rep("N",n\_barcode\_len),collapse="") # 模版序列中的N

}

### >>> 1. 确定100bp目标序列(分带/不带barcode)

#-------------------------------------------------------------------------------

# 确定Barcode起始位点与终末位点

N\_index\_begin <- gregexpr("N",mother\_seq)[[1]][1]

N\_index\_end <- gregexpr("N",mother\_seq)[[1]][n\_barcode\_len]

# 滑动起始终止位点

start\_site\_index <- N\_index\_end + 1 - (read\_lens - 1)

end\_site\_index <- N\_index\_begin - 1

### >>> 2. 生成随机字串

#-------------------------------------------------------------------------------

N\_random\_barcode <- function(N\_len,N\_reads,sed=666){

set.seed(sed)

for (i in 1:N\_len) {

if (i == 1) {

out <- sample(c("A","G","C","T"),N\_reads,replace = T)

} else{

out <- paste0(out, sample(c("A","G","C","T"),N\_reads,replace = T))

}

}

return(out)

}

### >>> 3. 位点字符串提取

#-------------------------------------------------------------------------------

site\_seq <- function(mother\_str,start\_site,n\_len\_reads,n\_len\_barcode,n\_site\_reads,neg\_rate){

### 提取目标区段序列

pos\_seq <- substr(mother\_str,start\_site,start\_site + n\_len\_reads - 1)

neg\_seq <- gsub("N","",substr(mother\_str,start\_site - n\_len\_barcode,start\_site + n\_len\_reads - 1)) # 阴性对照reads与阳性reads终止位点相同，起始位点平移一个barcode长度

### Barcode在序列中的位置

barcode\_start\_site <- unlist(gregexpr(barcode.tag,pos\_seq,fixed = T))

barcode\_end\_site <- barcode\_start\_site + n\_len\_barcode - 1

### 计算两组reads数目

pos\_num <- n\_site\_reads \* (1 - neg\_rate)

neg\_num <- n\_site\_reads \* neg\_rate

### 根据pos reads数产生相应数目的随机barcode

sub\_pos\_randomBar <- N\_random\_barcode(N\_len = n\_len\_barcode,N\_reads = pos\_num,sed = start\_site)

### 重复目标区段

pos\_seq\_n <- rep(pos\_seq,pos\_num)

neg\_seq\_n <- rep(neg\_seq,neg\_num)

bar\_start\_site\_n <- rep(barcode\_start\_site,pos\_num)

bar\_end\_site\_n <- rep(barcode\_end\_site,pos\_num)

### 替换上随机barcode

for (i in 1:pos\_num) {

pos\_seq\_n[i] <- gsub(barcode.tag,sub\_pos\_randomBar[i],pos\_seq\_n[i])

}

### 结果整合

out.list <- list()

barcode.seq <- c(sub\_pos\_randomBar,rep(NA,neg\_num))

barcode.start.index <- c(bar\_start\_site\_n,rep(NA,neg\_num))

barcode.end.index <- c(bar\_end\_site\_n,rep(NA,neg\_num))

total\_seq <- c(pos\_seq\_n,neg\_seq\_n)

out.list <- list(total\_seq,barcode.seq,barcode.start.index,barcode.end.index)

return(out.list)

}

for (i in start\_site\_index:end\_site\_index) {

tmp <- site\_seq(mother\_str = mother\_seq,

n\_len\_reads = read\_lens,

n\_site\_reads = n\_site\_reads,

start\_site = i,

n\_len\_barcode = n\_barcode\_len,

neg\_rate = neg\_rate)

seq <- tmp[[1]]

barcode.seq <- tmp[[2]]

barcode.start.index <- tmp[[3]]

barcode.end.index <- tmp[[4]]

if(i == start\_site\_index){

out.seq <- seq

out.barcode.seq <- barcode.seq

out.barcode.start.site <- barcode.start.index

out.barcode.end.site <- barcode.end.index

} else{

out.seq <- c(out.seq,seq)

out.barcode.seq <- c(out.barcode.seq,barcode.seq)

out.barcode.start.site <- c(out.barcode.start.site,barcode.start.index)

out.barcode.end.site <- c(out.barcode.end.site,barcode.end.index)

}

}

### >>> 4. 输出为fq文件

#-------------------------------------------------------------------------------

total\_reads\_num <- (end\_site\_index - start\_site\_index + 1) \* n\_site\_reads

reads\_id <- str\_pad(1:total\_reads\_num,width = 7,side = "left",pad = "0")

reads\_id <- paste0("@RI:",reads\_id)

barcode\_id <- paste0("BC:",gsub(">","",out.barcode.seq))

start\_id <- paste0("SS:",out.barcode.start.site)

end\_id <- paste0("ES:",out.barcode.end.site)

## 整理fastq文件

info.merge <- paste(reads\_id,barcode\_id,start\_id,end\_id,sep = "|||")

skeleton.fq <- data.frame(info=info.merge,seq=out.seq)

skeleton.fq <- skeleton.fq %>% mutate(strand="+",score=paste0(rep("F",read\_lens),collapse = ""))

## 整理metadata文件

metadata <- data.frame(seqname=info.merge, seq=out.seq, barcode=out.barcode.seq,

barcode\_start\_site=out.barcode.start.site, barcode\_end\_site=out.barcode.end.site)

metadata <- metadata %>% mutate(is\_empty=case\_when(grepl("NA",metadata$seqname)~"WithoutBarcode",

TRUE~"WithBarcode"))

## 输出序列

write.table(skeleton.fq,paste0(outdir,"/o1.BaseRandom\_demo.fq"),sep = "\t",quote = F,col.names = F,row.names = F)

write.xlsx(metadata,paste0(outdir,"/o2.BaseRandom\_metadata.xlsx"),colNames=T,rowNames=F,keepNA=T)

### Part 2: TemplateRandom

}else if (args$Mode=="TemplateRandom"){

if (args$Try) { # 程序测试使用

### >>> 1. 配置文件

###-----------------------------------------------------------------------------

outdir <- "Demo/02-output/TemplateRandom"

if(!dir.exists(outdir)){dir.create(outdir,recursive = T)}

source.dt <- read.xlsx("Demo/01-input/TemplateRandom/i1.RefSeqs.xlsx",sheet = 1,startRow = 1)

n\_site\_reads <- 500 # 单个位点总reads数

read\_lens <- 150 # 单个reads长度

# 滑动起始终止位点

start\_site\_index <- 1

end\_site\_index <- 149

} else{

### >>> 1. 配置文件

###-----------------------------------------------------------------------------

outdir <- args$Outdir

if(!dir.exists(outdir)){dir.create(outdir,recursive = T)}

source.dt <- read.xlsx(args$Templates,sheet = 1,startRow = 1)

n\_site\_reads <- args$SiteNums

read\_lens <- args$ReadLength

start\_site\_index <- 1

end\_site\_index <- read\_lens - 1

}

### >>> 2. 位点字符串提取函数

#-------------------------------------------------------------------------------

### 对于每个位点，500条reads，每条reads长度150bp，随机从39种参考序列中抽取

site\_seq\_random <- function(meta,start\_site,n\_len\_reads,n\_site\_reads,seeds=666){

### 原始信息提取

pos\_num <- nrow(meta)

pos\_str <- meta[,2]

pos\_seqname <- meta[,1]

set.seed(seed = seeds)

seed.num <- sample(1:pos\_num,n\_site\_reads,replace = T)

### 提取目标区段序列

pos\_seq <- substr(pos\_str[seed.num],start\_site,start\_site + n\_len\_reads - 1)

pos\_seqname\_record <- pos\_seqname[seed.num]

### 结果整合

total\_seq <- pos\_seq

total\_seqname <- pos\_seqname\_record

site\_start <- rep(start\_site,n\_site\_reads)

site\_end <- rep(start\_site + n\_len\_reads - 1,n\_site\_reads)

out.list <- list(total\_seqname,total\_seq,site\_start,site\_end)

return(out.list)

}

### >>> 3. 滑动生成数据

#-------------------------------------------------------------------------------

for (i in start\_site\_index:end\_site\_index) {

tmp <- site\_seq\_random(meta = source.dt[,1:2],

start\_site = i,

n\_len\_reads = read\_lens,

n\_site\_reads = n\_site\_reads,

seeds = i)

seqname <- tmp[[1]]

seq <- tmp[[2]]

start.index <- tmp[[3]]

end.index <- tmp[[4]]

if(i == start\_site\_index){

out.seqname <- seqname

out.seq <- seq

out.start.index <- start.index

out.end.index <- end.index

} else{

out.seqname <- c(out.seqname,seqname)

out.seq <- c(out.seq,seq)

out.start.index <- c(out.start.index,start.index)

out.end.index <- c(out.end.index,end.index)

}

}

### >>> 4. 输出为fq文件

#-------------------------------------------------------------------------------

total\_reads\_num <- (end\_site\_index - start\_site\_index + 1) \* n\_site\_reads

reads\_id <- str\_pad(1:total\_reads\_num,width = 7,side = "left",pad = "0")

reads\_id <- paste0("@RI:",reads\_id)

barcode\_id <- paste0("BC:",gsub(">","",out.seqname))

start\_id <- paste0("SS:",out.start.index)

end\_id <- paste0("ES:",out.end.index)

info.merge <- paste(reads\_id,barcode\_id,start\_id,end\_id,sep = "|||")

skeleton.fq <- data.frame(info=info.merge,seq=out.seq)

skeleton.fq <- skeleton.fq %>% mutate(strand="+",score=paste0(rep("F",read\_lens),collapse = ""))

metadata <- data.frame(seqname=info.merge,seq=out.seq,start\_site=out.start.index,end\_site=out.end.index)

metadata <- metadata %>% mutate(is\_empty=case\_when(grepl("EmptyVector",metadata$seqname)~"empty",

TRUE~"Non\_empty"))

write.table(skeleton.fq,paste0(outdir,"/o1.TemplateRandom\_demo.fq"),

sep = "\t",quote = F,col.names = F,row.names = F)

write.xlsx(metadata,paste0(outdir,"/o2.TemplateRandom\_metadata.xlsx"),colNames=T,rowNames=F,keepNA=T)

}

step2.MakeFqFormat.sh

#!/usr/bin/sh

infile=$1

outfile=$2

sed 's/\t/\n/g' $infile > $outfile && rm -rf ${infile}