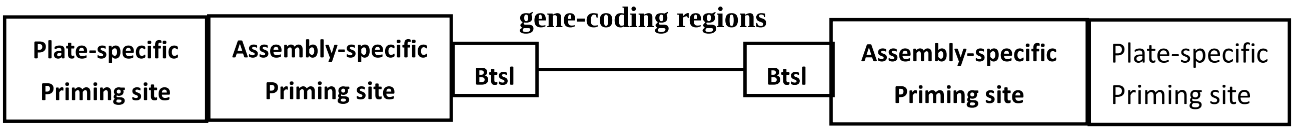
Description of this tool :

This tool deconstructs a given set of genes to generate the sequences of oligonucleotides for synthesis on a chip.The purpose of this script is computationally designing the DNA chip for amplifying the oligonucleotide subpools, and assembling 500- to 800-bp constructs.
The major stages of synthesis pipeline are computational design, chip synthesis, serial PCRs that isolate the oligonucleotides necessary to build each construct, and assembly of the constructs. The key principle is that well-designed primers can amplify a desired subset of oligonucleotides and, thereby, dilute the undesired DNA to the point where it does not interfere with the downstream gene assembly reaction (the major of scripts been developed by Nikolai Eroshenko et al..;2009 who form Harvard School of Engineering and Applied Sciences, Cambridge, Massachusetts).We have automated these design steps with Biopython scripts (Cock et al., 2009)

Internal operation:

Each construct to be built must be split up into short overlapping fragments. Each frag-ment must, in turn, be flanked by the assembly- and plate-specific subpool priming sequences, as well as restriction sites for removing the priming sequences(Figure)


gene-coding regions
Plate-specific Priming site
Assembly-specific Priming site
BtsI
BtsI
Assembly-specific Priming site
Plate-specific Priming site

The gene-coding regions of the oligonucleotides within each assembly subpool partially overlap, allowing them to be assembled into the full-length construct using a high-fidelity polymerase. The gene-coding region is flanked by BtsI cut sites that permit enzymatic removal of the subpool-specific priming sites. The gene-coding region is also flanked by a pair of assembly-specific priming sites, which are shared by all the oligonucleotides within a particular assembly subpool. The assembly-specific priming sites are, in turn, flanked by a pair of plate-specific priming sites common to all the oligonucleotides within a particular plate-specific subpool. So there have two module in the script: one is ols pool generation script. Anther is primer-design-script.You can see basic information in publication and in the supplemental materials:Scalable gene synthesis by selective amplification of DNA pools from high-fidelity microchips Sriram Kosuri, Nikolai Eroshenko, Emily LeProust, Michael Super, Jeffrey Way, Jin Billy Li & George Church Nature Biotechnology (2010) 28:1295 doi:10.1038/nbt.1716 News & Views, Cover Art

To run the script separately :
a) Place input files in the input-seqs directory

b) Edit the configuration file plate-based-assembly-from-ols-pool-config

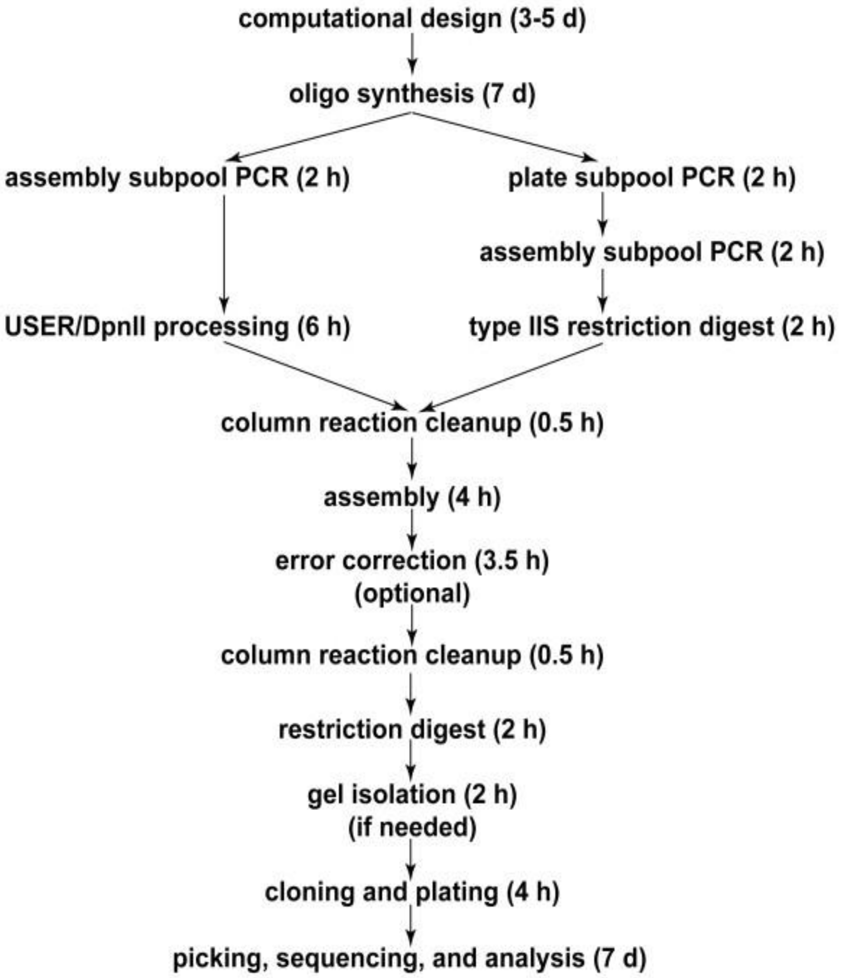
There is one configuration entry per file. Make sure that you set:
initialPlateNum(current entry) = initialPlateNum(previous entry) + (number of plates used up by sequences in previous entry)

c) cd to script directory in terminal and type “python gasp.py <configuration file >”

d) pay attention to outputs in the form of:

oligo files (oligo sequences generated corresponding to each input file),
primer files (primer list corresponding to each input file),
report files (build seq by build seq delineation of primer sets to use, for each input file)

Workflow for gene synthesis from high-fidelity DNA microchips :


computational design (3-5 d)
oligo synthesis (7 d)
assembly subpool PCR (2 h)
plate subpool PCR (2 h)
assembly subpool PCR (2 h)
USER/DpnII processing (6 h)
type IIS restriction digest (2 h)
column reaction cleanup (0.5 h)
assembly (4 h)
error correction (3.5 h)
(optional)
column reaction cleanup (0.5 h)
restriction digest (2 h)
gel isolation (2 h)
(if needed)
cloning and plating (4 h)
picking, sequencing, and analysis (7 d)

Shown here are the major steps and approximate timings of the entire gene(1G) synthesis process. The branch point reflects the choice of whether USER/DpnII processing (left branch after oligo synthesis) or type IIS enzymatic processing (right branch) are used for removing the amplification sites. The process outlines the final optimized form of the optimized protocols. The times given in parentheses are estimates that account for the time involved in both setting up and running the reactions.

Script 2: Get_configFile.pl

The function of this script is help user create configuration file which get Parameters form UI and then run auto.

Example:

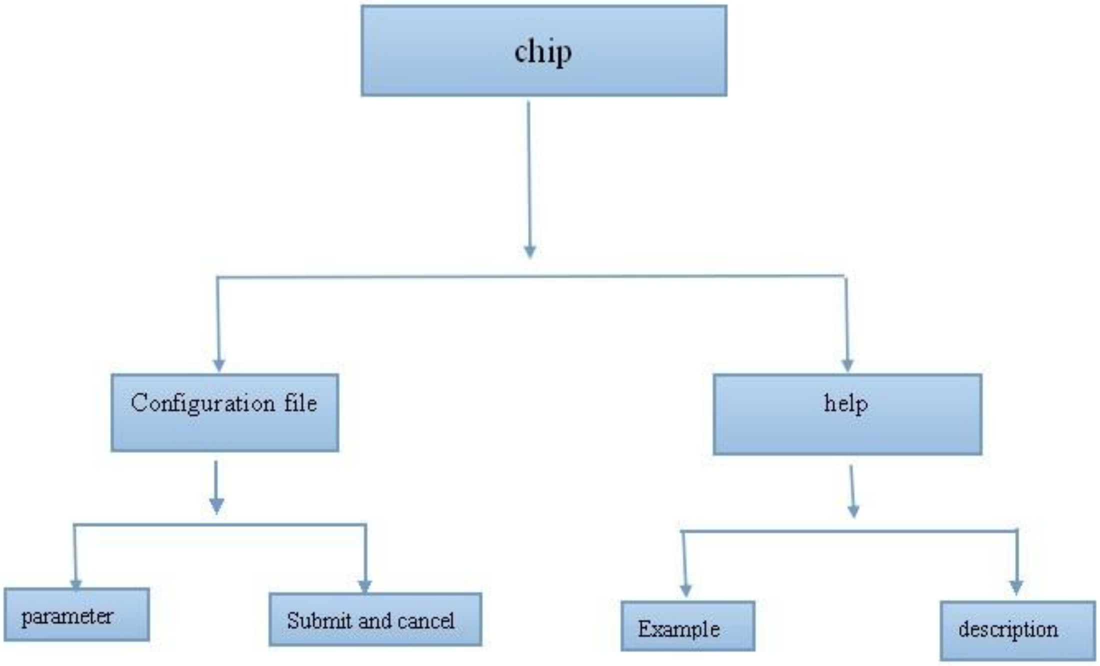
Usage : perl Get_configFile.pl <option> <output file name>

```
Perl Get_configFile.pl    -a 1 -b True -c 20 -d 2 -f 11 -i 4 -l input-seqs/yeast_chr1_3_16.all_bb.fasta
-n 4 -u 100 -o test
Or : perl Get_configFile.pl -l input-seqs/yeast_chr1_3_16.all_bb.fasta -n 4 -u 100 -o test
```

Parameters

		default
a	plate position index of primer set when using fixed primer set	0
b	true if all seqs in file get same primer set	false
c	Leeway in junction position that is allowed in searching for acceptable overlaps	10
d	plate # of primer set when using fixed primer set	1
f	length of oligo can be + or - this #	10
h	Print help information	
i	96-well plates holding assembled constructs are numbered starting with this number (never set to 1)	3
l	the location of a fasta file containing seqs for the desired constructs	
n	Reject a decomposition if it contains a self-dimer with free energy in kcal/mol below this number	3
u	oligoSizeMax	200
o	The name of project	

UI design



In our project, we want to help user design a new genomic according to the three modules we private and then use this script to help us to design the oligonucleotide and the priming sequence. So after the users have designed the genomic what they need, the next step is choose this script in our software’s UI.

Dependencies

UnaFold and BioPython are required to run the Python scripts in this package.

Please read followings for installation before run script !!!

Prerequisites: *UNAFold*and *Biopython.*

1、UnaFold
http://dinamelt.rit.albany.edu/download.php
2、Biopython：
http://biopython.org/DIST/docs/install/Installation.html

Help page:
The description of software:
GASP:Gene Assembly by Subpool PCR This set of scripts designs oligonucleotides that can be used to synthesize genes from high-complexity DNA pools

Parameter description
The parameters, which are described in detail below, may have to be further adjusted if the DNA will be processed using methods that deviate from the workflow described here.

InitialPlaneNum: 96-well plates of assemblies will be numbered sequentially initiating at this value. This should never be set to 1, as plate #1 is reserved for construction primers.

avgoverlapsize: Each construct will be broken up into assembly oligos that will be fused using a polymerase. The fusion reaction requires priming through overlaps between neighboring oligos. This setting specifies the mean length of the overlap region.

deltaGThresholdForOverlaps: Rejects any overlaps with a secondary structure that has a hybridization free energy less than the value specified (in units of kcal/mol).

selfDimerThreshold: Rejects assembly oligos that have any self-dimerization configura-tions with a hybridization free energy less than the value specified (arbitrary units).

lengthleeway: Sets allowable variation in the length of the overlap regions.

positionleeway: Sets allowable variation in the assembly oligo junction position. Increas-ing this value results in a less constrained search space, but increases the computation time and increases variation in synthesized oligonucleotides’ lengths.

oligoSizeMax: The maximum oligo size that will be designed. This includes the full-length oligos that include the coding region, the restriction enzyme processing site, and the assembly-specific and plate-specific priming sites. This value should typically be constrained by the commercial synthesis platform used. Note that many of the oligos will be shorter than this maximum value.

seqsToAvoidInOverlapRegion: Specifies positions to be avoided in the overlap between neighboring assembly oligos. This should usually be left blank, but can be used in specialized applications, such as constructing proteins with known repeated regions.

Example :

EXPLANATION OF CONFIG FILE:

```
{
    "initialPlateNum": 4, # 96-well plates holding assembled constructs are numbered starting with this number
    "buildSequencesFile": "input-seqs/yeast_chr1_3_16.all_bb.fasta", # the location of a fasta file containing seqs for the desired constructs
    "primerOutputFile": "output-files/primer-output.txt", # the location of a txt file which will contain primer sequence outputs
    "oligoOutputFile": "output-files/oligo-output.fasta", # the location of a fasta file which will contain oligo sequence outputs for the OLS pool
    "RESpacing": [ # list of offsets of enzyme cut sites from the end of the corresponding enzyme recognition sites, with enzymes ordered as in REVector
        5,
        2,
        5,
        4
    ],
    "REVector": [ # list of restriction enzymes that gasp will search through if SearchForRe is set to "True"
        "BsaI",
        "BtsI",
        "BsmBI",
        "BspQI"
    ],
    "SearchForRE": "True", # set this to "False" if you wish to control the exact restriction site which is used, specified by REToUse. Examples: usually set to "False" for DNA origami scaffold sequences, usually set to "True" for synthetic genes for in-vivo use.
    "REToUse": "", # Restriction enzyme to use if SearchForRE is set to "False", e.g., "BsaI". Leave blank if SearchForRE is set to "True". Usually set to "BsaI" for Shih lab DNA
                    origami scaffold sequences.
    "forwardPrimersLibraryFile": "primer-library/forward_finalprimers.fasta", # Fixed library of ~ 3000 orthogonal forward primers
    "reversePrimersLibraryFile": "primer-library/reverse_finalprimers.fasta", # Fixed library
```

of ~ 3000 orthogonal reverse primers

"avgoverlapsize": 20, # Average length of overlap region between adjacent oligos

"deltaGThresholdForOverlaps": -3, # Overlap is rejected if its hybridization free energy in kcal/mol is below this number

"selfDimersThreshold": 3, # Reject a decomposition if it contains a self-dimer with free energy in kcal/mol below this number

"insertionSizeToKillRESite": 2, # Leave this set to 2 for now

"lengthleeway": 10, # Leeway in oligo length that is allowed in searching for acceptable overlaps

"overlaptemps": [# Overlap regions must have a melting temperature in this range

55,

65

],

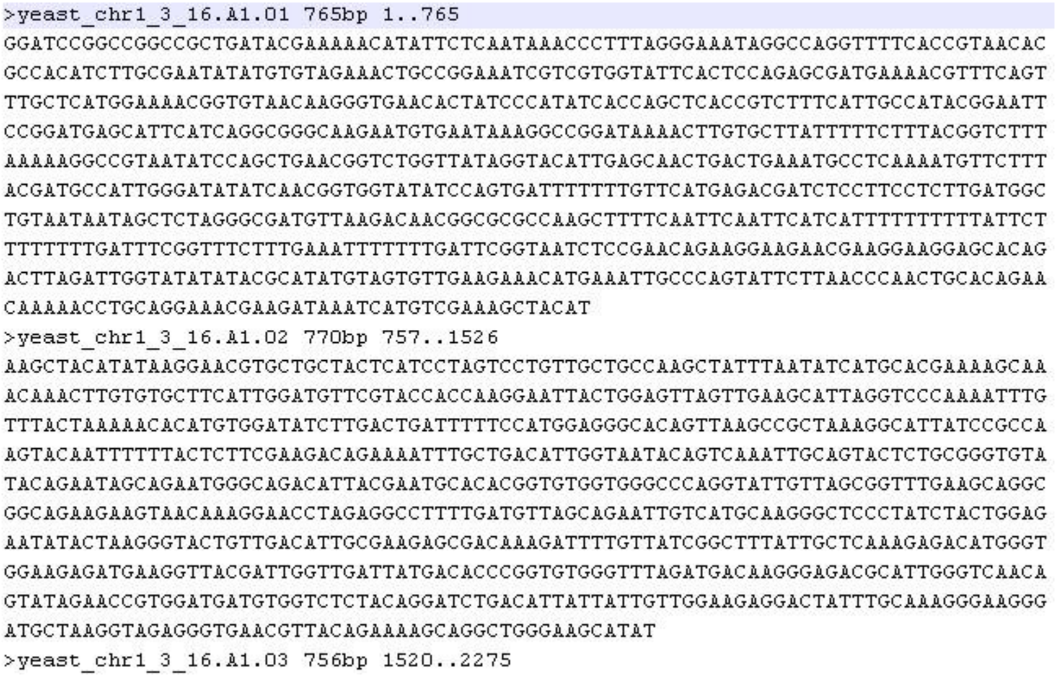
"positionleeway": 10 # Leeway in junction position that is allowed in searching for acceptable overlaps

}

Result:

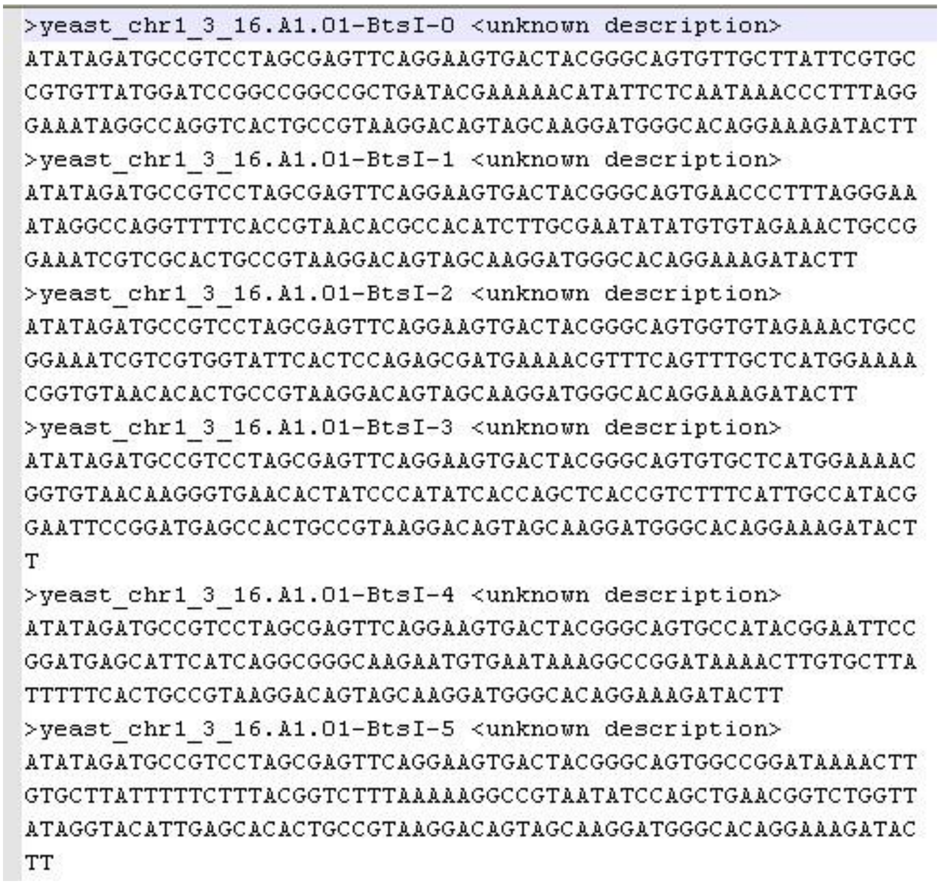
The first one will contain a report that contains: (1) The sequences to be synthesized on the DNA chip in FASTA format; (2) The plate-specific, position-specific, and construction primers needed to build the set of assemblies; (3) The plate-specific, position-specific, and construction primers that correlate with each individual assembly. The second e-mail will contain a FASTA file that contains the sequences that should be synthesized on the DNA chip.

Appendix:



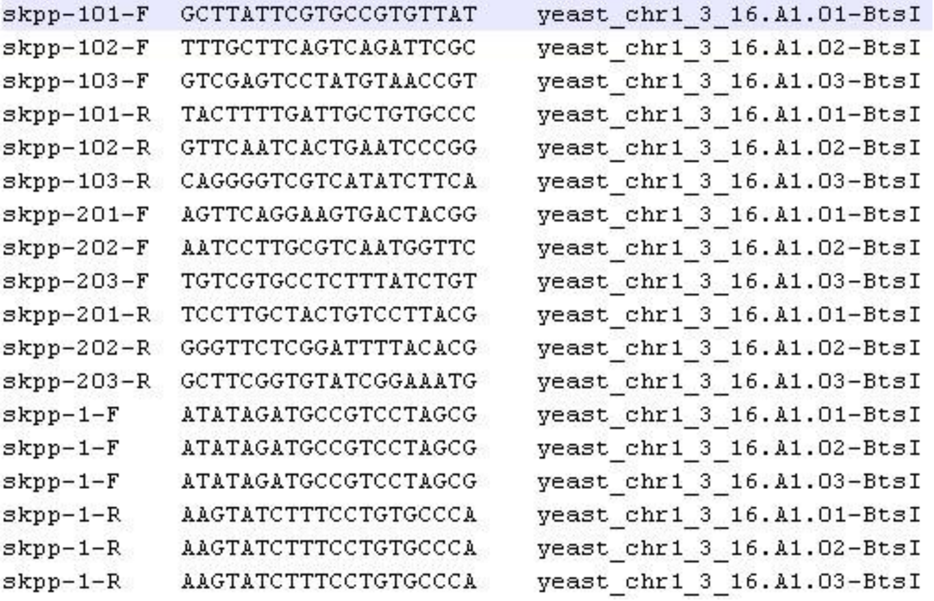
```
>yeast_chr1_3_16.A1.01 765bp 1..765
GGATCCGGCCGGCCGCTGATACGAAAAACATATTCTCAATAAACCCTTTAGGGAAATAGGCCAGGTTTTCACCGTAACAC
GCCACATCTTGCGAATATATGTGTAGAAACTGCCGGAAATCGTCGTGGTATTCACTCCAGAGCGATGAAAACGTTTCAGT
TTGCTCATGGAAAACGGTGTAACAAGGGTGAACACTATCCCATATCACCAGCTCACCGTCTTTCATTGCCATACGGAATT
CCGGATGAGCATTCATCAGGCGGGCAAGAATGTGAATAAAGGCCGGATAAAACTTGTGCTTATTTTTCTTTACGGTCTTT
AAAAAGGCCGTAATATCCAGCTGAACGGTCTGGTTATAGGTACATTGAGCAACTGACTGAAATGCCTCAAAATGTTCTTT
ACGATGCCATTGGGATATATCAACGGTGGTATATCCAGTGATTTTTTTGTTCATGAGACGATCTCCTTCCTCTTGATGGC
TGTAATAATAGCTCTAGGGCGATGTTAAGACAACGGCGCGCCAAGCTTTTCAATTCAATTCATCATTTTTTTTTTATTCT
TTTTTTTGATTTCGGTTTCTTTGAAATTTTTTTGATTCGGTAATCTCCGAACAGAAGGAAGAACGAAGGAAGGAGCACAG
ACTTAGATTGGTATATATACGCATATGTAGTGTTGAAGAAACATGAAATTGCCCAGTATTCTTAACCCAACTGCACAGAA
CAAAAACCTGCAGGAAACGAAGATAAATCATGTCGAAAGCTACAT
>yeast_chr1_3_16.A1.02 770bp 757..1526
AAGCTACATATAAGGAACGTGCTGCTACTCATCCTAGTCCTGTTGCTGCCAAGCTATTTAATATCATGCACGAAAAGCAA
ACAAACTTGTGTGCTTCATTGGATGTTCGTACCACCAAGGAATTACTGGAGTTAGTTGAAGCATTAGGTCCCAAAATTTG
TTTACTAAAAACACATGTGGATATCTTGACTGATTTTTCCATGGAGGGCACAGTTAAGCCGCTAAAGGCATTATCCGCCA
AGTACAATTTTTTACTCTTCGAAGACAGAAAATTTGCTGACATTGGTAATACAGTCAAATTGCAGTACTCTGCGGGTGTA
TACAGAATAGCAGAATGGGCAGACATTACGAATGCACACGGTGTGGTGGGCCCAGGTATTGTTAGCGGTTTGAAGCAGGC
GGCAGAAGAAGTAACAAAGGAACCTAGAGGCCTTTTGATGTTAGCAGAATTGTCATGCAAGGGCTCCCTATCTACTGGAG
AATATACTAAGGGTACTGTTGACATTGCGAAGAGCGACAAAGATTTTGTTATCGGCTTTATTGCTCAAAGAGACATGGGT
GGAAGAGATGAAGGTTACGATTGGTTGATTATGACACCCGGTGTGGGTTTAGATGACAAGGGAGACGCATTGGGTCAACA
GTATAGAACCGTGGATGATGTGGTCTCTACAGGATCTGACATTATTATTGTTGGAAGAGGACTATTTGCAAAGGGAAGGG
ATGCTAAGGTAGAGGGTGAACGTTACAGAAAAGCAGGCTGGGAAGCATAT
>yeast_chr1_3_16.A1.03 756bp 1520..2275
```

Figure 1. Shown here is the format of input file in this program



```
>yeast_chr1_3_16.A1.01-BtsI-0 <unknown description>
ATATAGATGCCGTCCTAGCGAGTTCAGGAAGTGACTACGGGCAGTGTTGCTTATTCGTGC
CGTGTTATGGATCCGGCCGGCCGCTGATACGAAAAACATATTCTCAATAAACCCTTTAGG
GAAATAGGCCAGGTCACTGCCGTAAGGACAGTAGCAAGGATGGGCACAGGAAAGATACTT
>yeast_chr1_3_16.A1.01-BtsI-1 <unknown description>
ATATAGATGCCGTCCTAGCGAGTTCAGGAAGTGACTACGGGCAGTGAACCCTTTAGGGAA
ATAGGCCAGGTTTTCACCGTAACACGCCACATCTTGCGAATATATGTGTAGAAACTGCCG
GAAATCGTCGCACTGCCGTAAGGACAGTAGCAAGGATGGGCACAGGAAAGATACTT
>yeast_chr1_3_16.A1.01-BtsI-2 <unknown description>
ATATAGATGCCGTCCTAGCGAGTTCAGGAAGTGACTACGGGCAGTGGTGTAGAAACTGCC
GGAAATCGTCGTGGTATTCACTCCAGAGCGATGAAAACGTTTCAGTTTGCTCATGGAAAA
CGGTGTAACACACTGCCGTAAGGACAGTAGCAAGGATGGGCACAGGAAAGATACTT
>yeast_chr1_3_16.A1.01-BtsI-3 <unknown description>
ATATAGATGCCGTCCTAGCGAGTTCAGGAAGTGACTACGGGCAGTGTGCTCATGGAAAAC
GGTGTAACAAGGGTGAACACTATCCCATATCACCAGCTCACCGTCTTTCATTGCCATACG
GAATTCCGGATGAGCCACTGCCGTAAGGACAGTAGCAAGGATGGGCACAGGAAAGATACT
T
>yeast_chr1_3_16.A1.01-BtsI-4 <unknown description>
ATATAGATGCCGTCCTAGCGAGTTCAGGAAGTGACTACGGGCAGTGCCATACGGAATTCC
GGATGAGCATTCATCAGGCGGGCAAGAATGTGAATAAAGGCCGGATAAAACTTGTGCTTA
TTTTTCACTGCCGTAAGGACAGTAGCAAGGATGGGCACAGGAAAGATACTT
>yeast_chr1_3_16.A1.01-BtsI-5 <unknown description>
ATATAGATGCCGTCCTAGCGAGTTCAGGAAGTGACTACGGGCAGTGGCCGGATAAAACTT
GTGCTTATTTTTCTTTACGGTCTTTAAAAAGGCCGTAATATCCAGCTGAACGGTCTGGTT
ATAGGTACATTGAGCACACTGCCGTAAGGACAGTAGCAAGGATGGGCACAGGAAAGATAC
TT
```

Figure 2. SHORTENED_yeast_chr1_3_16all_bb-oligo-output.fasta



```
skpp-101-F   GCTTATTCGTGCCGTGTTAT    yeast_chr1_3_16.A1.01-BtsI
skpp-102-F   TTTGCTTCAGTCAGATTCGC    yeast_chr1_3_16.A1.02-BtsI
skpp-103-F   GTCGAGTCCTATGTAACCGT    yeast_chr1_3_16.A1.03-BtsI
skpp-101-R   TACTTTTGATTGCTGTGCCC    yeast_chr1_3_16.A1.01-BtsI
skpp-102-R   GTTCAATCACTGAATCCCGG    yeast_chr1_3_16.A1.02-BtsI
skpp-103-R   CAGGGGTCGTCATATCTTCA    yeast_chr1_3_16.A1.03-BtsI
skpp-201-F   AGTTCAGGAAGTGACTACGG    yeast_chr1_3_16.A1.01-BtsI
skpp-202-F   AATCCTTGCGTCAATGGTTC    yeast_chr1_3_16.A1.02-BtsI
skpp-203-F   TGTCGTGCCTCTTTATCTGT    yeast_chr1_3_16.A1.03-BtsI
skpp-201-R   TCCTTGCTACTGTCCTTACG    yeast_chr1_3_16.A1.01-BtsI
skpp-202-R   GGGTTCTCGGATTTTACACG    yeast_chr1_3_16.A1.02-BtsI
skpp-203-R   GCTTCGGTGTATCGGAAATG    yeast_chr1_3_16.A1.03-BtsI
skpp-1-F     ATATAGATGCCGTCCTAGCG    yeast_chr1_3_16.A1.01-BtsI
skpp-1-F     ATATAGATGCCGTCCTAGCG    yeast_chr1_3_16.A1.02-BtsI
skpp-1-F     ATATAGATGCCGTCCTAGCG    yeast_chr1_3_16.A1.03-BtsI
skpp-1-R     AAGTATCTTTCCTGTGCCCA    yeast_chr1_3_16.A1.01-BtsI
skpp-1-R     AAGTATCTTTCCTGTGCCCA    yeast_chr1_3_16.A1.02-BtsI
skpp-1-R     AAGTATCTTTCCTGTGCCCA    yeast_chr1_3_16.A1.03-BtsI
```

Figure 3.SHORTENED_yeast_chr1_3_16all_bb-primer-output.txt