Pan-Genomes Analysis Pipeline (PGAP)

Yongbing Zhao

Update at April 13, 2016

Any suggestion or questions for PGAP, please mail to pgap@big.ac.cn

1. What's PGAP?

PGAP is a Pan-Genome Analysis Pipeline and it has integrated several useful analysis methods in genome research into one Perl script, which would make bacterial genome research easier.

2. What could PGAP do?

PGAP has five program sections, there are as following:

1) Orthologs clusters identification among multiple genomes
2) Pan-genome analysis for given strains
3) Functional genes variation and SNP calling among given strains
4) Evolutionary analysis based on pan-genome and SNP data
5) Orthologs clusters function analysis

3. How to install PGAP?

PGAP is developed with Perl script, so there is no necessary to compile PGAP. Because several extra program would be invoked in PGAP, please install and set the extra program path in PGAP.pl source code prior to running PGAP. Here is the list for the extra programs:

1) blastall and formatdb in BLAST (2.2.12 or higher).
   BLAST is available from ftp://ftp.ncbi.nih.gov/blast/executables/release/
2) mafft.
   mafft is available from http://mafft.cbrc.jp/alignment/software/
3) dnaml, dnadist, neighbor, seqboot, consense in PHYLIP (version 3.69)
   PHYLIP package is available from http://evolution.gs.washington.edu/phylip.html
4) mcl
   mcl is available from http://micans.org/mcl/

4. Input data

Three type files are required for running the whole pipeline and they are protein sequences, and their corresponding nucleotide sequences and annotation files.

PGAP recognizes these three type files by their special extension names, ***.pep*** for protein sequences, ***.nuc*** for nucleotide sequences and ***.function*** for annotation file. For each strain, these three type files should have the same prefix, which we call **nickname**. All input files are named as ***nickname.pep***, ***nickname.nuc*** and ***nickname.function.*** For different strains, the nicknames should be different.

Take *Salmonella enterica* subsp. enterica serovar Typhi str. CT18 for example, we could name its three type files "CT18.pep, CT18.nuc, CT18.function" or "NC_003198.pep, NC_003198.nuc, NC_003198.function" or others like these.

As for gene name, 16758994, STY0117, orf_0112 are valid. But the gene name should be unique in all strains. For the same strain, the gene names should keep consistent in all the three files.

In protein sequences files (***.pep*** file), the sequence should be in FASTA format. The data format are as follows:

```
>16758994
MNRISTTTITTITITTGNGAG
>16762656
MLYKGCLMKSDVQLNLRAKESQRALIDAAAEILHKSRTDFILETACRAAENVILDRRVFNFNDEQ
YEEFINLLDAPVADDPVIEKLLARKPQWDV
```

In the nucleotide sequences files (***.nuc*** file), the sequence should be in FASTA format. The data are as follows:

```
>16758994
ATGAACCGCATCAGCACCACCACCATTACCACCATCACCATTACCACAGGTAACGGTGCGGGCTG
A
>16762656
ATGCTATACAAGGGGTGTCTCATGAAATCAGATGTTCAACTTAACCTTAGAGCTAAGGAGTCGCA
GCGGGCGCTCATTGATGCAGCTGCGGAAATCCTTCACAAGTCACGTACAGATTTCATTCTGGAAA
CGGCCTGCCGGGCTGCCGAGAATGTGATCCTTGACCGCCGTGTATTTAACTTTAACGATGAGCAA
TATGAGGAGTTCATCAATCTGCTTGATGCACCGGTCGCAGATGATCCCGTTATCGAAAAACTGCT
GGCAAGGAAACCTCAGTGGGACGTGTAA
```

In the annotation files (***.function*** file) , there would be three columns, the first column is the gene name, the second is COG classification and the last column is function description. If the gene has no COG classification information, put "-" on the corresponding position. These three columns are **separated by <tab>** and there is **NO header** in this table. The data are as follows:

```
16758994    -          thr operon leader peptide
16762656    COG4453S    hypothetical protein
```

If the bacterial genomic data were downloaded from NCBI several scripts were provided in PGAP package to convert the NCBI data to the above required formats. The usage of these scripts would be introduced in the later paragraphs.

5. What does PGAP output?

All output result files are named with a digital prefix (according to the order listed in what PGAP could do?). For example, the result from orthologs clusters identification will begin with "1", so on and so forth.

1) Error message

0.error.message: if some problem was found before performing substantial analysis, the PGAP program would crash and all error messages would be reported in this file.

2) Orthologs clusters identification among multiple genomes

1.Orthologs_Cluster.txt: cluster list detail, if some strain has no genes in the cluster, mark with "-".

1.Gene_Distribution_By_Conservation.txt: gene number in each strains by clusters conservation.

3) Pan-genome analysis for given strains

2.PanGenome.Profile.txt: pan-genome and core genome function

2.PanGenome.Data.txt: the temporary data used for fitting pan-genome function.

4) Genome variation and SNP calling among given strains

3.CDS.variation.txt: variation detail in CDS region among all given strains.

There are seven columns (see the demonstrated figure).

1»	7»	213.1»	I,V»	A,G»	GGGGAGG»nonsynonymous
1»	7»	225.3»	V»	A,T,G»	GTTGATT»synonymous
1»	7»	232.1»	S,A»	T,G»	GGGGTGG»nonsynonymous
1»	7»	234.3»	G»	T,C»	TCCCCCC»synonymous
2»	7»	1»	-,M»	-»	---M-M-»InDel
2»	7»	2»	-,T»	-»	---T-T-»InDel
2»	7»	3»	-,E»	-»	---E-E-»InDel

The 1st column is the Cluster ID, which is consistent with the ID in 1.Orthologs_Cluster.txt.

The 2nd column is the cluster conservation of current cluster.

The 3rd column is the variation position, which counts according to the alignment result of protein sequences in this cluster. For indel events, the position is an integer. For synonymous mutation and nonsynonymous mutation, the position is a floating number, in which the integer part marks the position of amino acid in the alignment result of protein sequences, while the decimal part mark the position of codon. For example: 213.1 mean there are variation on the 213th amino acid in the alignment result of protein sequences and the variation location on the 1st codon. 225.3 means that the variation location on the 3rd codon of the 225th amino acid.

The 4th column shows the amino acid types on current position.

The 5th column shows the nucleotide types on current position, For Indel, only "-" will be given.

The 6th column shows all gene nucleotide profile in current position (for indel, amino acid will be listed). The order of nucleotide/amino acid is consistent with the gene order in current cluster in 1.Orthologs_Cluster.txt.

The 7th column shows the variation type (Indel, synonymous and synonymous).

3.CDS.variation.for.evolution.txt: the temporary DNA sequence file in phylip format. This file records the variation in the core CDS region. Another difference between 3.Core.CDS.variation.txt and 3.CDS.variation.txt is that, if there is a variation in some amino acid, the corresponding three nucleotides are output in this file.

3.CDS.variation.analysis.txt: is the summary result for 3.CDS.variation.txt.

5) Evolutionary analysis based on pan-genome and SNP

In the result of this part, all phylogenetic trees calculated by phylip are output in the .tree file, user could use visualization software to view the phylogenetic trees (Tips: TreeView could be freely used to output the figure you prefer. It could be downloaded from http://darwin.zoology.gla.ac.uk/~rpage/treeviewx).

6) Orthologs clusters function analysis

5.Orthologs_Cluster_Function.txt: COG classification for each gene cluster

5.Orthologs_Whole_Cluster_COG_Distribution.txt: COG enrichment for all clusters

5.Orthologs_Core_Cluster_COG_Distribution.txt: COG enrichment for core gene cluster

5.Orthologs_Dispensable_Cluster_COG_Distribution.txt: COG enrichment for dispensable gene cluster

5.Orthologs_specifc_Cluster_COG_Distribution.txt: COG enrichment for strain specific gene cluster

6. Data format conversion

To make data preparation easily, several scripts have been provided to converting NCBI downloaded data:

1) For NCBI new format data from ftp://ftp.ncbi.nlm.nih.gov/genomes/all, `Converter_NCBINewFormatData.pl` could be used to prepare the input data. Here is the capture of the parameters.

```
Usage:   perl Converter_NCBINewFormatData.pl [options]

Options:

   --inprefix   String     The prefix of the input files, such as GCF_000007545.1_ASM754v1
                           If two or more strains were provided, please join their prefixs with "+"
                           Such as GCF_000007545.1_ASM754v1+GCF_000008105.1_ASM810v1+GCF_000711315.1_ASM71131v1

   --indir      String     The directory of those input files

   --outprefix  String     The prefix for the output files.
                           If a value "ty2" was provided, the output files would be: ty2.nuc, ty2.pep, and ty2.function
                           If two or more strains were provided, please join their prefixs with "+"
                           If the prefix was not provided, the assembly value would be used as the prefix, such as GCF_000007545

   --outdir     String     The directory for the output files

Note:
   1. Before converting data with this script, please prepare *feature_table.txt, *genomic.fna and *protein.faa files for each strain.

   2. This script was designed for NCBI new format data only. If part of your data is in the old format, please use the Converter_finished.pl
 or Converter_draft.pl script to convert the data.
```

First decompress *feature_table.txt.gz, *genomic.fna.gz and *protein.faa.gz files in the same folder. If you have multiple strains 'genomic data are in new format, you could also put all the decompressed data in the same folder and then convert them with one command.

Convert One by one:

```
perl Converter_NCBINewFormatData.pl -inprefix
GCF_000007545.1_ASM754v1 -indir input_directory -outprefix ty2
-outdir output_directory
```

Convert all at once:

```
perl Converter_NCBINewFormatData.pl -inprefix
GCF_000007545.1_ASM754v1+GCF_000008105.1_ASM810v1+GCF_000711315
.1_ASM71131v1 -indir input_directory -outprefix
name1+name2+name3 -outdir output_directory
```

2) For NCBI old RefSeq data (completed genome sequences) from ftp://ftp.ncbi.nlm.nih.gov/genomes/archive/old_refseq/Bacteria/ (this address may be different from previous ftp address), *Converter_finished.pl* could be used to prepare the input data. and Here is the capture of the parameters.

```
Usage:   perl Converter_finished.pl [options]

Options:

  -S String    Input the strains nickname,
               If 2 or more, join them with '+',
               For example: CT18+NC_011834+SPA
  -I String    Input file directory
  -O String    Output file directory
```

First, put all strains' .ptt, .ffn and .faa files in the same folder. In the finished folder of the testdata directory, there are some test data from five E. coli strains: AC_000091.faa, AC_000091.ffn, AC_000091.ptt and so on. For these data, we could convert them one by one, or convert them together.

Convert One by one:

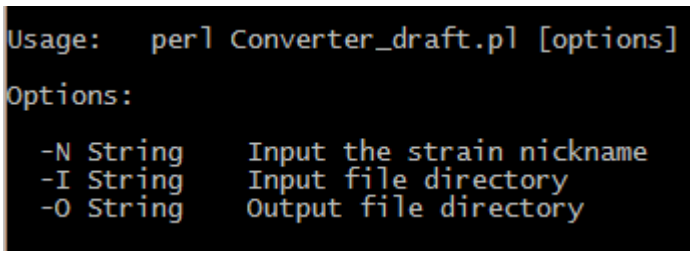
Take AC_000091 for example, we could convert AC_000091 data with the command like this:

```
perl Converter_finished.pl -S AC_000091 -I input_drectory -O output_directory
```

Convert all at once:

```
perl Converter_finsihed.pl -S AC_000091+NC_000913+NC_004431 -I input_drectory -O output_directory
```

3) For NCBI old RefSeq data (draft genome sequences) from ftp://ftp.ncbi.nlm.nih.gov/genomes/archive/old_refseq/Bacteria_DRAFT/ (this address may be different from previous ftp address), *Converter_draft.pl* could be used to prepare the input data. and Here is the capture of the parameters.



Decompress each strain's .ffa.tgz, .ffn.tgz, and .ptt.tgz file in an individual folder. Then use the *Converter_draft.pl* script to convert each strain's data separately.

```
perl Converter_draft.pl -N nickname -I input_drectory -O output_directory
```

In the above command, different strain should be given a different nickname.

7. Tutorials for PGAP

Step 1 Preparing input data

Prepare input data with provided scripts for NCBI data or prepare user own data with DIY script.

Step 2 Preparing the third part programs and program path setting

Four extra programs (BLAST, MAFFT, PHYLIP and MCL) are required prior to running PGAP. These programs could be easily installed according to their README files and manuals. After these programs are installed, open the PGAP.pl file and replace each program's path in the script.

Step 3 Running PGAP

For PAGP, there are five functional modules, cluster analysis of functional genes, pan-genome profile analysis, genetic variation analysis of functional genes, species evolution analysis and function enrichment analysis of gene clusters. The following commands could trigger all these analysis modules with the default parameters.

```
perl PGAP.pl –strains nickname1+nickname2+...+nicknameN –input
input_directory –output output_directory --cluster –pangenome --
variation --evolution --function –method MP
```

Or

```
perl PGAP.pl –strains nickname1+nickname2+...+nicknameN –input
input_directory –output output_directory --cluster –pangenome --
variation --evolution --function –method GF
```

--cluster --pangenome --variation --evolution and --function denote whether to run cluster analysis of functional genes, pan-genome profile analysis, genetic variation analysis of functional genes, species evolution analysis and function enrichment analysis of gene clusters respectively. Theoretically, the entire five modules could be run alone, when the required input data are available. For the first time to analysis a batch of new strains, *--cluster* is required.

--thread could be used to accelerate the process in cluster analysis of functional genes, when *--cluster* is added.

--evalue, *--score*, *--identity*, *--coverage* (or *--local* and *--global*) could be used for adjust the cutoff for homologs identification. *– coverage* is valid when *--method GF* is used, while *--local* and *--global* is valid when *--method MP* is used.

--bootstrap could be used for adjust the boorstrap times when *--evolution* is used