README

1 General

The metatool YACOP [1] generates a prediction, which is based on the output of existing gene finding programs like Glimmer or Orpheus. In present YACOP supports the combination of Critica105b[1] [5] (with wublast[2]), Glimmer2.02 [3] or Glimmer2.10[3] [2] (with RBSfinder[4] [4]), Orpheus[5] [6] (with dps[6] [7]) and ZCurve1.0 [8]. However, it's possible to extend YACOP and integrate additional tools (see section 8, p.8), for its source code is avaliable. The current version of YACOP is realised in Perl (v.5.6.1). Its implemented and tested under RedHat Linux v2.4.18.
YACOP expects input in FASTA (starting with script `combinate`) or multiple-FASTA format (starting with script `combinate_multi`). Some of the methods used internally rely on nucleotide, codon, or oligomere frequencies that have to be derived from the input therefore the minimal length of the input sequence is 50 kB. When started with option `concat`, even shorter contigs can be analysed, if sufficient fragments of the sequence are available (given as multiple-FASTA).
The starter script `combinate` automatically triggers the gene finding programs and evaluation of their predictions. That prerequisites the correct installation of the tools (Critica, Glimmer etc.). In case of problems concerning the individual gene finding programs, please consult the respective readme-files. The invocation of ZCurve, which is developed in FORTRAN, cannot be integrated for the binarys are only available for win32 systems. Anyhow, the output can be generated under windows and then be posted manually during invocation to the skript `combinate` (not possible for `combinate_multi`). A C++-version of ZCurve is in process, and may soon be integrated.
The paths directing to the tools should be set in the central `ini`-file of YACOP (see section 4, p. 3).

[1] http://geta.life.uiuc.edu/ gary/programs/CRITICA/
[2] http://blast.wustl.edu/
[3] http://www.tigr.org/software/glimmer/
[4] http://www.tigr.org/software/
[5] http://pedant.gsf.de/orpheus/
[6] ftp://cs.mtu.edu/pub/huang/

2 System requirements

Most gene prediction tools are implemented in Perl, C or C++. Therefore, these programming languages should be available on your system. If you don't use default paths or default compilers, the tools as well as the sources of YACOP should be edited individually.
Some gene prediction tools (Critica, Orpheus), used by YACOP rely on database searches and statisitcal analysis, for that it is not recommendable to install it on an normal pc. Unless you would like to use YACOP to combine the output of Glimmer and ZCurve, which are fast and require far less capacity.
The following programming languages have to be available on you system:

1. Perl5.6.1[7] or higher (should be installed in `/usr/bin/perl`)
2. BioPerl.3[8] or higher
3. C and C++ (default compiler gcc) needed for Glimmer, Orpheus, dps and Critica

YACOP can be used on multi-processor systems running PBS (Portable Batch System) as well as on a single processor system. In this case, programs are executed sequentially. For the adaptation of the YACOP source code to different system architectures, see section 7, p. 7.

3 Additional scripts

Some additional conditions have to be fulfilled: The scripts `check_qstat`, `gl2rbs`, `separate` and `toUp` need to be available in your `PATH`. You can either copy the scripts to your `bin` directory or set respective links there. Another possibility is to integrate the whole YACOP home-directory into your `PATH`. If you would like to use the scripts manually, have a look at the comments.

- `gl2rbs` is needed for conversation of Glimmer output to input of RBS-finder.
- `toUp` converts lowecase sequences to Critica-compatible uppercase version. If you start YACOP with `combinate_multi` invocation of `toUp` occurs automatically, unlike during startup of `combinate`, where you should trigger the script manually (if your sequence is composed of lowercase characters). The uppercase sequence is written to the file FASTA-path_`uc` (after usage this file will be removed).

[7] http://www.perl.com/
[8] http://bioperl.org/

- `check_qstat` is only needed if you use YACOP on multi-processor machines. It checks out current state of the subprocesses.

- `seperate` will be invoken in `-concat` mode. It seperates the output of `*.tbl` or `*.sum_out` to the subsets of the concatenated contigs. The individual results will be written to `~/data/longtime/contigs_tbl/_nr.NAME` or `~/data/longtime/contigs_sum/_nr.NAME`. The suffix `NAME` corresponds to the contig name. The invocation of `separate` is triggered by `combinate`. By default, only the mode-dependent results in `*.tbl` will be seperated. If you like to use `seperate` on the whole results (mode- and length independent), which are stored in `*.sum_out`, you should comment respective code in.

4 Running YACOP

There are two sets of parameters to be considered.

a) Parameters during invocation of YACOP
The script `combinate` should be called with a statement like:

```
combinate gene.fasta ini-file PREFIX [-gr -c -o -gb gene.gbk
-z zcurve.pred -v -concat]
```

The parameters for `combinate_multi` are alike:

```
combinate_multi multi_fasta ini-file PREFIX [-c -o -gr -gb
file.gbk -v -concat]
```

Note, that you should type '`perl ~/YACOP/combinate ..`' if you didn't add the YACOP home-directory to your `PATH` variable.
The filename of the FASTA-file must be given with the **absolute path**. The parameters `-gr`, `-c` and `-o` tell YACOP which tools to use. The parameter `-c` activates invocation of Critica, -gr activates Glimmer plus RBSfinder and `-o` triggers the invocation of Orpheus in combination with dps. With the parameter `-z` a file containing ZCurve output can be posted to the skript (not possible for `combinate_multi`).
The term `PREFIX` is synonym for a prefix naming your organism. It is used to compose names of result-files and may be formatted according to the regular expression `/^R([A-Z]{2,3}/`. The parameter -gb must be set together with the path to a GenBank-file. This option allows to check the performance of YACOP and the integrated tools, by comparing the annotation deposited at GenBank with the output generated by the programs (see section 6, p. 6. The Parameter -v is for verbose mode (writing information of `ini`-file to `properties.inf`). YACOP copies the resulting output to a directory `~/data/longtime` (located at the current home directory), see section 6, p. 5.

b) The `ini`-file (example is given in `YACOP/GenePred.ini`) contains all parameters required by the tools, the path to their home directories and the parameters needed by the Perl program `meta_pred.pl`. All parameters (plus a short description of their function) can be set by using this central repository.
The parameter `mode` defines how YACOP merges the individual predictions and how to select the gene starts. Possible settings are:

- `crit_orp_gl` combines the predictions of Critica with the predictions made by both Glimmer and Orpheus. For the prediction of the startcodons the output of Critica gets the highest priority. If an orf is not predicted by Critica, but by Orpheus and Glimmer, the Orpheus output is favoured.

- `crit_zc_gl` outputs Critica predictions with the intersection of ZCurve and Glimmer, Critica starts will be prefered, elsewise the start position predicted by ZCurve is taken into account.

- `zc_crit_gl` combines Critca predictions with the intersection of ZCurve and Glimmer predictions. Starts of ZCurve are favoured where possible, elsewise Critica starts are taken.

- `crit_zc_gl_orp` combines the predictions of Critica with the intersection of the predictions of Glimmer, ZCurve and Orpheus. Start coordinates are taken from Critica output, otherwise for predictions not found in Critica output the prediction of ZCurve is taken.

- `zc_crit_gl_orp` combines the predictions of Critica with the intersection of the predictions of Glimmer, ZCurve and Orpheus. For the prediction of startcodons, in this mode the ZCurve output is favoured. If a predicted orf is not found in the ZCurve output, the Critica start coordinate is taken.

- `zc_gl` outputs intersection of predictions of ZCurve and Glimmer with ZCurve start.

In order to implement additional modes, the subroutine `eval_start_mode(HASH)` has to be modified (see section 7, p. 7).
Additionally a parameter `-minlength` can be set in the `ini`-file. This causes the tool to reject all predictions shorter than given number (denoted in BP). This limitation obtaines for all predictions except such made by Critica.

5 Input format

The input format should be of type FASTA or multiple-FASTA. The script `combinate` handles the triggering of the individual tools. The script `combinate_multi` splits multiple-FASTA input into several FASTA-files. Each

of these temporarily generated files will then be processed by an instance of `combinate`, created by `combinate_multi`. If started in `concat` mode `combinate_multi` concatenates the sequence fragments in given multiple-FASTA file and posts it as one sequence to `combinate`. All predicted coding regions traversing a concatenation-site are removed from the result and stored to file `meta_pred.msg` in resp. output directory.
Example for the FASTA format:

```
>AP000398_Buchnera_sp._APS_complete_genome.
TTATCCACAGATTTGTTCTTTACTAATAATAATAGTAATTATTATTTTTTATTTTTTTTATTTTTTT
GAATTTAAACCTTAAAGAAAAGAAAAAGATCTTTTTTTTAAGATATTATGTTTTTAAGATTAACATG
TGTTATCTTGAATAAAATATTAATACTATTTGAATATTTTTAAATTTTTAAAAGGTTTTTATATGTT
```

`combinate` accepts only sequences composed of uppercase characters. This is due to limitations imposed by some of the tools that do not work for lower-case sequences. `combinate_multi` does not have this limitation. The script `toUp` is called before passing sequence files to `combinate`. `toUp` converts the input sequence to an uppercase version. The annotation of a contig should start with > and should not exceed one line. Note that some of the tools used for prediction do not accept sequence ids which are too long or contain white spaces.

6 Output formats

YACOP stores results in the directory `~/data/longtime`. Where `longtime` corresponds to the time of invocation of YACOP in long format. If you are not contend with this, you can change the output path in `combinate` (see variable `output_dir`, line 27).
Each prediction of YACOP consists of three basic output files:

a) The file `PREFIX.sum_out` contains all predictions made by the tools integrated in YACOP. The file can be regarded as a table. Each line describes one predicted orf. The landmarks used to organize the output are the stopcodons occurring in the sequence. The first column contains the positions of the stopcodons (sorted in ascending order), given by the first nucleotide of the codon. The second column lists the reading frame of the gene (comp). The following columns contain the output of the different tools. Each start coordinate is defined as the first nucleotide of startcodon.
Start positions given for Glimmer are those generated by RBSfinder (Glimmer/RBSfinder). RBSfinder alters the initial predicted start of Glimmer by shifting them up or downstream if the results differ. The number, given in column 3 is the offset, RBSfinder has introduced for the start position. In brackets the amount of the shift is denoted. A positive number indicates a downstream shift, a negative correspondingly an upstream shift. The output for Critica is the position of the startcodons and a p-value, rating the

quality of the prediction. For Orpheus, only the predicted start position is given. In evaluation mode, i.e. the name of a GenBank file is given during startup, as shown below, the start position and the annotation as deposited in the file are added to the output. If ZCurve output was given during invocation, additionally its predicted start and a score (like Critica-score, rating the prediction of ZCurve) will be denoted in-between Orpheus output and GenBank output (not shown).
Example for a file of type `PREFIX.sum_out`:

```
2081 F     164 ( 33)    197 6.82e-147    164  197 glucose inhibited divis
2771 R    2924 (  -)      -         -      -  -
3100 F    2278 (  -)   2278  3.08e-20   2278  2278 ATP synthase A chain
3376 F    3139 (  -)   3139  2.27e-07      -  3139 ATP synthase C chain
3980 F    3497 (  -)   3497  8.36e-16   3497  3497 ATP synthase B chain
```

stop comp | Glimmer/RBSf. | Critica | Orpheus | GenBank

b) The file `PREFIX.tbl` contains a gene table. The genes compiled in this file are extracted from the sets of genes predicted by the individual tools. The arrangement of genes is depending on the mode, which has to be set in the `ini`-file. The default mode (as given in example `ini`-file, see section 4, p. 3) is `crit_orp_gl`. For each predicted coding region, one line of output is generated. The first column contains the id of the gene. The id results from the given PREFIX and a number incremented automatically. The id is formatted according to the regular expression `/^R([A-Z]{2,3})\d{5,6}/`. In addition, the name of the contig and the coordinates of the gene are added.
NOTE: 3'-end is given exclusive stopcodon, 5'-end is inclusive startcodon. An example for the format of PREFIX.tbl:

```
RBU000001 AP000398_Buchnera_sp._APS_complete_genome._197_2080
RBU000002 AP000398_Buchnera_sp._APS_complete_genome._2278_3099
RBU000003 AP000398_Buchnera_sp._APS_complete_genome._3139_3375
RBU000004 AP000398_Buchnera_sp._APS_complete_genome._3497_3979
```

c) The file `PREFIX.fasta` consists of the amino acid sequences (inclusive startcodon), annotated with the id according to the entries deposited in the tbl-file.

```
>RBU000001
MFNLRNFDVIVVGAGHAGTEAAMASSRMGCKTLLLTQKISDLGALSCNPAIGGIGKSHLVKEIDAL
GGMMAKAIDYSGIQFRILNSSKGPAVRSTRAQADKILYHETVKKILKKQNNLLILEAEVKDLIFKN
YSVVGVLTQNEINFYSRSVVLAAGTFLGGKIHIGLKSYSAGRIGDKSAIDLSVRLRELSLRVNRLK
TGTPPRIDINTVNFNNLLIQNSDTPVPVFSFMGNVSHHPKQIPCYLTHTNEKTHEIIRKNLDKSPI
YTGFLKGLGPRYCPSIEDKIVRFPDRKSHQVFLEPEGLSSIKVYPNGISTSLPIEVQEQIVASIKG
LEKSKIIRPGYAIEYDFFDPKDLNLTLESKLIKGLFFAGQINGTTGYEEAASQGLLAGLNAALSSK
NTEGWFPRRDQAYLGVLIDDLTTQGTEEPYRMFTSRAEYRLSLREDNADLRLTEIGRKLGLVNDSR
```

```
WIRYNQKVLNIQTEMNRLKKNKISPISPDADILKKLYNINLIKEISMSELLKRPQIRYQDLQSLES
FRTGIVDLEAIGQIENEIKYAGYIKRQSEEIERHLKNENTFLSSIYDYNKIRGLSSEVVKKLNDYK
PISIGQASRISGITPAAISILLIHLKKESYKHTL
```

In the output directory you will find several more files. Most of these are output files of the tools, Critica, Glimmer and Orpheus (see table). You will also find a file `meta_pred.msg`. This file contains errormessages an predicted genes, that were removed from the resultset in `concat` mode. If you started YACOP in `verbose` mode (option `-v`) the file `properties.inf` will be added. This file contains the parameter composition of this invocation of YACOP (set in the `.ini` file). The files named `sh_clongtime.sh` and `sh_olongtime.sh` are the invocation scripts of Critica and Orpheus if YACOP is startet on multi-processor mashines. In this case the files `sh_*longtime.o` contain all messages printed to STDOUT and `sh_*longtime.e` all messages printed to STDERR by Critica or Orpheus.

Glimmer		
	`long-orfs`	`gl_PREFIX.orfs`, `gl_PREFIX.msg_lo`
	`extract`	`gl_PREFIX.extract`
	`build-icm`	`gl_PREFIX.model`
	glimmer2	`gl_PREFIX.out`, `gl_PREFIX.msg_gl`
RBSfinder	`gl2rbs`	`PREFIX_tmp.gl2rbs`
		`gl_PREFIX.rbs_out gl_PREFIX.rbs_msg`
Critica		
	blast-contigs	`PREFIX.blast`
	make-blastpairs	`PREFIX.blast.pairs`
	scanblastpairs	`PREFIX.triplets`
	iterate-critica	`PREFIX`nr.cds (nr ≡ iteration)
Orpheus		
	`dps`	`orph_PREFIX.dps`
	`orpheus2` (`-wcu`)	`orph_PREFIX.nuc_usage`
	`orpheus2` (`sure` oder `rcu`)	`NAME.orfaln`, `NAME.orfnuc`, `NAME.orfprot`
	`starter`	`orph_PREFIX.weights`

7 Modification of the source code

Adaptation to multi- or single-processor machines YACOP can be startet with PBS systems by using the command qsub(1B). The script `combinate` calls the starter-objects (inheriting from `Nudge.pm`), which creates scripts to submit Critica and Orpheus invocation via qsub. Glimmer is started as a non-batch-process. For the usage of qsub, additionally the script `check_qstat` is provided. During invocation of `check_qstat` the name of a qsub-script has to be given as a command line parameter (this is done automatically by `combinate`. It is used to control the execution of the respective subprocess. If you are not using a bash (borne-again shell) on your system, the scripts `check_qstat` and the variable `Nudge::BASH_CALL`

should be modified. The objects extending Nudge.pm also provide subroutines to start the sub-programs sequentially. The code for sequential starts is commented out, this could easily be altered.

Altering output-format If you require other formats you should change one of the writing routines or create a new one and add the function-call to the main program of `meta_pred.pl`. If you do so, note that the object QuickResult only contains the predictions respective to current mode and minlength, meanwhile ResultSet contains all.
Current output and related routines are:

- `output_all(ResultSet)` calling `write_all(HASH)`
- `write_table(QuickResults)` calling `eval_start_mode(HASH)`
- `write_multiple_fasta(QuickResults)` calling `eval_start_mode(HASH)`

additionally, predictions which are discarded in concat mode are written to STDERR in routine `trim_results(ResultSet)`. In `combinate` error messages of `meta_pred.pl` are redirected to the file `meta_pred.msg`.

8 Integration of other prediction tools

First you have to add the invocation of your tool to the skript `combinate`. For that you should add parameters, home directory etc. to the `ini`-file. For the invocation itself you should write an object extending `Nudge.pm` comparable to `GlimmerNudge.pm` and add the path and the call of respective subroutines to combinate.
For parsing the output of your tool, a parser according to `GlimmerParser.pm` should be implemented an integrated to `meta_pred.pl`. Furthermore you should add evaluation of the arguments to routine `eval_args()`.
For the invocation of `meta_pred.pl`, the name of the parameter invoking your tool (like `arg{MYTOOL} = "-X"`) and the result file created by it (like `orf_check_comm{MYTOOL}=" ".arg{MYTOOL}." ". my_tool->get_result-output()`) should be added to `combinate`.

The modifications of `meta_pred.pl`:

1. flags adaption of subroutine `eval_args()`
   (e.g. `F_GLIMMER=combinate::arg{GLIMMER}`)
2. modi adaption of subroutine `eval_start_mode(HASH)`
   (e.g. `M_ZC_GL`)

3. adaption of subroutine `eval_start_mode` for the most performant prediction of start coordinates.

If the full merged output of all tools is needed, you should also modify subroutine `output_all(ResultSet)` and `write_all(HASH)`.

References

[1] Tech M., Merkl R. 2003. Yacop - Enhanced Gene Prediction Obtained by a Combination of Existing Methodes. *In Silico Biology* in press.

[2] Salzberg S., Delcher A., Kasif S., White O. 1998. Microbial gene identification using interpolated Markov models. *Nucleic Acids Res.* **26**:544-548.

[3] Delcher A., Harmon D., Kasif S., White O., Salzberg S. 1999. Improved microbial gene identification with GLIMMER. *Nucleic Acids Res.* **27**:4636-4641.

[4] Suzek B.E., Ermolaeva M.D., Schreiber M., Salzberg S. 2001. A probabilistic method for identifying startcodons in bacterial genomes. *Bioinformatics* **17**:1123-1130.

[5] Badger J., Olsen G. 1999. CRITICA: Coding Region Identification Tool Invoking Comparative Analysis. *Mol. Biol. Evol.* **16(4)**:512-524.

[6] Frishman D., Mironov A., Mewes H.-W., Gelfand M. 1998. Combining diverse evidence for gene recognition in completely sequenced bacterial genomes. *Nucleic Acids Res.* **26**:2941-2947.

[7] Huang X. 1996. *Micorb. Compar. Genomics* **1**: 281-291.

[8] Guo F.-B., Hou H.-Y., Zhang C.-T. 2000. ZCURVE: a new system for recognizing protein-coding genes in bacterial and archaeal genomes. *Nucleic Acides Res.* **31**: 1780-1789.