AutoCoEv – a high-throughput *in silico* pipeline for revealing novel protein-protein interactions

Petar B. Petrov[1,2*], Vid Šuštar[1], Luqman O. Awoniyi[1,2], Martti Tolvanen[3], and Pieta K. Mattila[1,2*]

[1] Institute of Biomedicine and MediCity Research Laboratories, University of Turku, Turku, Finland
[2] Turku Bioscience, University of Turku and Åbo Akademi University, Turku, Finland
[3] Department of Future Technologies, University of Turku, Turku, Finland

* Corresponding authors. PP: petar.petrov@utu.fi; PM: pieta.mattila@utu.fi

Keywords: coevolution, correlation, protein, interaction, automation

ABSTRACT

Protein-protein interactions govern the cellular decision-making and various functions. Thus, identifying new interactions between proteins can significantly facilitate our understanding on the mechanistic principles of protein functions. Coevolution between proteins is a sign of functional communication and as such provides a powerful approach to search for novel molecular partners. However, evolutionary analyses of large arrays of proteins, *in silico*, is a highly time-consuming effort, which has prevented the usage of this method beyond protein pairs or narrow protein families, for example. Here, we developed AutoCoEv, a user-friendly computational pipeline for the search of coevolution between hundreds and even thousands of proteins. Driving over 10 individual programs, with CAPS2 as a key software for coevolution detection, AutoCoEv achieves seamless automation and parallelization of the workflow. In addition, we provide a patch to CAPS2 source code to improve its statistical output, allowing for multiple comparisons correction.

We apply the method to inspect coevolution among 297 individual proteins identified to be in close proximity to the B cell receptor (BCR) before and after receptor activation. We successfully detected coevolutionary relations between the proteins, predicting novel partners and clusters of interacting molecules. We conclude that AutoCoEv can be used to predict protein interaction from large datasets of hundreds, and with aid of super-computing recourses even thousands of proteins in a time and cost-efficient manner.

Introduction

The biological function of proteins is carried out through associations with various molecules, the majority of which being, in fact, other proteins. The specific interplay between their amino acids, reflects the underlying molecular mechanisms of protein association, activity and regulation. Investigating interactions not only elucidates the functional role of proteins, but can reveal the existence of yet unknown protein partners. Screening for novel interactions is of high importance for deciphering the complexity of protein networks, determinant for the functional organization of cells. It has been suggested that relations between proteins can be extrapolated from the evolutionary history of their genes, *in silico* [1,2]. Such computational approaches, however, would demand a high degree of automation when used with large datasets, an issue that we successfully addressed in this work.

Evolution of proteins is influenced by structural and functional constraints between amino acids, enforcing their changes in a concerted manner. Detecting *intra-* or *inter-* molecular coevolution is regarded as a sign for correlation between residues within the same protein, or between sites belonging to different partners, respectively [3]. Various computational methods for prediction have been described, among which are BIS2 [4], ContactMap [5], DCA [6], Evcouplings [7], MISTIC [8] and CAPS2 [9]. Many searches for inter-protein coevolution have been confined to a relatively small number of partners, where an existing correlation has been essentially anticipated [10–14]. With no automation, extending the work to even tens of proteins presents a challenge, and becomes virtually impossible for larger numbers of proteins.

Here, we developed an automated and user-friendly computational pipeline, called AutoCoEv, for the large-scale screening for protein interactions. In the center of the workflow is CAPS2 (Coevolution Analysis using Protein Sequences 2) software, that compares the evolutionary rates at two sites in the form of their correlated variance [9]. By driving over 10 additional programs,

AutoCoEv achieves high level of automation, flexibility and processes parallelization, enabling the analysis of hundreds and even thousands or proteins.

Implementation

The preparation pipeline for most coevolutionary analyses has a relatively simple concept. Typically, for each protein of interest, a multiple sequence alignment (MSA) of its orthologues needs to be produced, optionally combined with a phylogenetic tree. The process requires the correct identification of orthologues, retrieval of the sequences, filtering, and various format conversions.

Command line interface, configuration and input

AutoCoEv is written in BASH, and offers a simple menu-driven command line interface (CLI), where the individual steps are enumerated (Figure 1). Options for the programs that AutoCoEv drives, as well as different filterings are configured in a single file, which is well commented and described in details in the manual distributed with the script. Once configuration options have been set, a user can simply go through the consecutive steps and conduct the analysis automatically.

Upon start, AutoCoEv offers to download its required databases from OrthoDB [15] and run initial preparations, such as FASTA database indexing. As an input, the script requires a list of proteins with their UniProt identifiers [16] and a list of species with taxonomic codes. Optionally, a phylogenetic tree may be provided from an external source, such as TimeTree [17], to be used as a guide (see later).

Identification of orthologues

For each protein of interest, AutoCoEv consults OrthoDB searching for orthologues in the species from the user-provided list. The script matches the UniProt ID of each protein to its OrthoDB ID, then extracts its unique orthologues group (OG) ID at a certain level of organisms (e.g. *Eukaryota*,

Metazoa, *Vertebrata*, *Tetrapoda*, *Mammalia*). This level, or *node*, is specified by the user and would depend on the species for which orthologues are searched (next step). The script will report proteins with missing or duplicated entries at OrthoDB, as well as species where orthologues of a protein are not found. The script will also report proteins with the same OG ID at the specified level, that are a result of gene duplication (see Manual for details).

Once these searches are done, AutoCoEv prepares a list of homologues for each protein, found in the species of interest (Figure 2 A). Due to alternative splicing or gene duplication, there may be several homologues identifiers per species for the same protein. Therefore, the script determines the correct orthologue by a reciprocal BLAST [18] run as follows. The amino acid sequence (from the reference organism, e.g. mouse) of each protein is downloaded from UniProt and prepared as a local BLAST database. Homologous sequences from each species are then retrieved and blasted against the UniProt sequence from the reference organism (e.g. mouse). For each species, the hit with best score is considered as the closest to the correct orthologues. Their sequences are collected, with the option to omit the ones that do not pass a certain identity and coverage (gaps) threshold, specified by the user. Now, each protein has a collection of automatically curated orthologous sequences, ready for alignment.

Multiple sequence alignment and quality filtering

AutoCOEV offers a choice of three widely-used and accurate programs for the creation of multiple sequence alignments (MSA): MAFFT [20], MUSCLE [21] and PRANK [22] (Figure 2B). Different MAFFT aliases are supported, while for PRANK an external phylogenetic tree (e.g. obtained from TimeTree) can be specified as a guide. The script then offers to run Gblocks [23] on the generated MSA, to filter out regions that are poorly aligned, too divergent or otherwise unreliable. Eliminating columns of low quality aims to improve the accuracy of results, while the shorter MSA may significantly speed up the consequent analyses.

Phylogenetic trees

CAPS2 will generate trees automatically at runtime from the MSAs of the analysis, if no trees were specified. We have patched the program, so the generated trees are exported in the output (see Materials and Methods, "Patch for verbose CAPS output"), enabling inspection and assessment.

Trees can also be calculated by another program (Figure 2C), which may be desirable to improve the sensitivity of the analysis. For the purpose, AutoCoEv drives PhyML (Phylogenetic estimation using Maximum Likelihood) [24] to calculate trees from each MSA (using either the full-length alignments or the ones processed by Gblocks). An external tree (e.g. obtained from TimeTree) can be specified as a guide, and users are provided with the option to automatically root the produced trees. In our experience, CAPS2 runs are more stable when rooted trees are used. Conversion between sequence formats is done by Squizz, while trees processing, such as trimming of branches, as well as, rooting are done by TreeBeST [25].

Detection of inter-protein coevolution by CAPS

Before the actual CAPS2 run, AutoCoEv takes several preparation steps of the orthologues MSA and (optionally) trees, created for each protein. The script determines all unique pairwise combinations between the proteins, creating an individual folder dedicated to each pair. As an example, 10 proteins will produce 45 combinations, while the 297 proteins of our dataset yield 43,956 pairs. For each combination, before being placed in the respective folder, the MSA (and trees) of the two proteins are compared to determine the species they have in common (Figure 2D). The sequences of the "unshared" species are removed from the MSA by SeqKit [26] and the trees (if used) are trimmed by TreeBeST, accordingly. In our experience, the presence of too many species not shared by the two proteins deteriorates the stability of CAPS. The number of species affects the specificity of the coevolution detection [9], therefore users can specify a minimum threshold "shared species" for a protein pair.

When all is set, AutoCoEv runs CAPS2 via GNU/Parallel (see next), spawning multiple individual instance of the program, each operating in a separate protein pair folder (Figure 2E and 3).

Parallelization

The computational time required for the analysis presents a major bottleneck, as many programs lack CPU multi-threading. Utilization of multiple processor cores is a necessity at several steps through this workflow, the most critical one being CAPS2 run itself. For CAPS2, and other software, we overcome these shortcomings by executing the program via GNU/Parallel [19]. We achieve the simultaneous run of multiple, single-core jobs, dramatically speeding up the time of computation (Figure 3).

Results processing and assessment

After CAPS runs are complete in all protein pair folders, AutoCoEv processes the results in several steps of filtering, sorting and assessment (for details, see Manual). For each protein pair, where coevolution was detected, the correlated amino acids are inspected in regards of their column within the MSA. Those MSA columns with alignment gaps above a certain threshold (e.g. more than 20% gaps) are omitted, in order to improve the specificity of the analysis.

By default, CAPS2 calculates a correlation cutoff threshold, specific for each individual protein pair, based on a number of simulated sequences, serving as null data (CAPS manual, p8). The correlation values of residues from one protein pair cannot be directly compared to those from a different protein pair. For a large-scale analysis, as the one presented here, ranking the results of multiple comparisons is very important. Therefore, we prepared a patch for CAPS source code (see Materials and Methods, “Patch for verbose CAPS output”), which makes the program extract the p-value of each correlation (*closest upper-bound* p-value, see Supplementary Materials).

The script then extracts the correlated residues with best p-values per protein pair, as well as calculates a mean value from the total number of correlations. In the final step (Figure 2 F), the script writes a XML (Extensible Markup Language) file of the resulting network, ready for visualization and additional analyses by Cytoskape [27].

Application

We tested AutoCoEv on a set of 297 mouse proteins identified by APEX2 proximity biotinylation proteomics to be located at the lipid raft membrane domains in B cells (Awoniyi et al, 2020, bioRxiv). A list of 37 tetrapod species was prepared and an external phylogenetic tree was obtained from TimeTree knowledge-base. For each protein, orthologues sequences with homology greater than 35% to mouse, having less than 25% gaps in the BLAST alignment were considered. MSA were created by MAFFT with alias "mafft-linsi" and processed by Gblocks to eliminate low quality regions. Trees were then calculated by PhyML from the filtered MSA, using the TimeTree tree as a guide and produced trees were rooted by TreeBeST. Before running CAPS2, a total of 43,956 unique pairs were produced, with minimum overlapping species set to 20.

AutoCoEv successfully completed the CAPS2 jobs in the individual protein pairs folders. After the run was complete, the script processed the results, excluding pairs from MSA columns with more than 20% gaps across species. Using Cytoskape, we limited the results to those with p-values below 0.00001, achieving a very stringent cutoff. The produced network (Figure 4) illustrates numerous connections between proteins, having several distinct proteins as interactions "hubs".

Discussion

In this paper, we present AutoCoEv: an interactive script, aimed at the large-scale prediction of inter-protein coevolution. The variety of options that can be easily set in the configuration file, allow for numerous adjustments, providing a great level of flexibility in the workflow.

The automated batch identification of orthologues, grants the seamless processing of even thousands of proteins, when high computing power is available. The choice of MSA software largely depends on the sequences being aligned, therefore our script drives three of the most widely used MSA programs [28], and we do not exclude incorporation of additional methods. At the moment, AutoCoEv drives PhyML, offering a reliable means for trees calculation, and we are planning to implement wrappers for at least RAxML [29] and MrBayes [30].

By default, inter-molecular analysis in CAPS2 was designed for just a handful of proteins, or only a pair, as illustrated by the web interface (http://caps.tcd.ie/caps/). Lack of multi-threading is a major obstacle in the analysis of large datasets and here we have successfully overcome this issue. By creating in advance all possible pairwise combinations, then running multiple instances of the program simultaneously, we take full advantage of modern multi-core processor architecture. With the increasing numbers of processor cores in the recent years, it is possible to analyse hundreds of proteins even on a common modern PC. For example, the CAPS processing of our 297 proteins dataset took around two days on a workstation with an Intel Xeon W-2135 (6 cores and 12 threads). For protein numbers reaching thousand or more, access to super-computing resources is required to keep the analysis run times low.

Since correlation values derived by CAPS cannot be directly compared between the protein pairs, we opted to extract the actual p-value for each correlation. Comparable p-values also allow the usage of additional statistical tests, such as Bonferroni correction, and we believe the statistical power of the analyses got improved by this addition.

Materials and methods

Data availability

The script is under open liencese and is freely available for download from the GitHub repository of our group (https://github.com/mattilalab/petrov-et-al-2020) together with the data used in this work. For details, please see Manual on GitHub.

Script environment and required software

AutoCoEv is written in BASH and its development was done on a Slackware 14.2 (http://www.slackware.com/) GNU/Linux system. Software tools and their dependencies, not included in the base system, were installed from the scripts available at SlackBuilds.org (https://slackbuilds.org/). These are: BLAST+ (2.10.1), CAPS (2.0), EMBOSS (6.6.0), Exonerate (2.4.0), Gblocks (0.91b), MAFFT (7.453), MUSCLE (3.8.1551), PRANK (170427), Parallel (20200522), PhyML (3.3.20190909), SeqKit (0.12.1), squizz (0.99d) and TreeBeST with Ensembl modifications (1.9.2_git347fa82). Compiling CAPS requires Bio++ (release 1.9) libraries (https://github.com/BioPP): bpp-utils (1.5.0), bpp-seq (1.7.0), bpp-numcalc (1.8.0) and bpp-phyl (1.9.0, patched). A virtual machine image with all requirements pre-installed is also available upon request. See Supplementary Materials for details and dependencies.

Databases

When searching for orthologues, AutoCoEv takes advantage of the data from OrthoDB (https://www.orthodb.org/). The following databases are used (10.1): odb10v1_all_fasta.tab.gz, odb10v1_gene_xrefs.tab.gz, odb10v1_OG2genes.tab.gz. The script also communicates with UniProt (https://www.uniprot.org/) to download the latest sequence of each protein of interest from its reference organism. In case an external (i.e. not calculated from MSA) phylogenetic tree was needed, the TimeTree knowledgebase was used (http://www.timetree.org/).

Patch for verbose CAPS output

Our patch introduces two modifications to CAPS, making the program produce more verbose output. First, the program will use *TreeTemplateTools* from bpp-phyl to export CAPS generated trees by *treeToParenthesis*, if none were supplied by the user at run-time. Second, after coevolution between sites has been determined, CAPS searches back within *totaltemp* vector of null simulated data, for the p-value corresponding closest to the estimated correlation value. See Supplementary Materials for detailed explanation with excerpts of code.

Proximity biotinylation

For details, see Awoniyi *et al* 2020, a preceding study from our group, published in parallel with this work. Briefly, lysates of B cells stimulated with 10µg/mL antibody against BCR were collected after 0 min, 5 min, 10 min and 15 min time points. The APEX2 system was used to induce biotinylation of proteins within 20 nm range in close proximity to the BCR. Samples were subjected to streptavidin affinity purification followed by mass spectrometry analysis. MaxQuant (1.6.17.0) was used for database search and after differential enrichment analysis with NormalyzerDE (1.6.0), a list of 297 proteins identified with high confidence was prepared.

Author contributions

PP developed the script and wrote the manuscript. VŠ conceived the original idea of the automated pipeline and contributed to trouble-shooting. LA provided the data for analyses, generated the network analysis of the results and developed a statistical evaluation of the results. MT contributed to trouble-shooting and brought ht expertise in the theoretical basis. PM contributed to the study design, trouble-shooting, and manuscript preparation.

References

1. Baussand J, Carbone A. A Combinatorial Approach to Detect Coevolved Amino Acid Networks in Protein Families of Variable Divergence. PLoS Comput Biol 2009; 5:

2. Kuriyan J. Allostery and coupled sequence variation in nuclear hormone receptors. Cell 2004; 116:354–356

3. Fares MA, McNally D. CAPS: coevolution analysis using protein sequences. Bioinformatics 2006; 22:2821–2822

4. Oteri F, Nadalin F, Champeimont R, et al. BIS2Analyzer: a server for co-evolution analysis of conserved protein families. Nucleic Acids Research 2017; 45:W307–W314

5. Wang S, Sun S, Li Z, et al. Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model. PLOS Computational Biology 2017; 13:e1005324

6. Morcos F, Pagnani A, Lunt B, et al. Direct-coupling analysis of residue coevolution captures native contacts across many protein families. Proc. Natl. Acad. Sci. U.S.A. 2011; 108:E1293-1301

7. Hopf TA, Green AG, Schubert B, et al. The EVcouplings Python framework for coevolutionary sequence analysis. Bioinformatics 2019; 35:1582–1584

8. Simonetti FL, Teppa E, Chernomoretz A, et al. MISTIC: Mutual information server to infer coevolution. Nucleic Acids Res. 2013; 41:W8-14

9. Fares MA, Travers SAA. A Novel Method for Detecting Intramolecular Coevolution: Adding a Further Dimension to Selective Constraints Analyses. Genetics 2006; 173:9–23

10. Travers SAA, Fares MA. Functional Coevolutionary Networks of the Hsp70–Hop–Hsp90 System Revealed through Computational Analyses. Molecular Biology and Evolution 2007; 24:1032–1044

11. Huang Y, Temperley ND, Ren L, et al. Molecular evolution of the vertebrate TLR1 gene family - a complex history of gene duplication, gene conversion, positive selection and co-evolution. BMC Evol Biol 2011; 11:149

12. Ruiz-González MX, Fares MA. Coevolution analyses illuminate the dependencies between amino acid sites in the chaperonin system GroES-L. BMC Evol Biol 2013; 13:156

13. Petrov P, Syrjänen R, Smith J, et al. Characterization of the Avian Trojan Gene Family Reveals Contrasting Evolutionary Constraints. PLoS ONE 2015; 10:e0121672

14. Champeimont R, Laine E, Hu S-W, et al. Coevolution analysis of Hepatitis C virus genome to identify the structural and functional dependency network of viral proteins. Sci Rep 2016; 6:26401

15. Kriventseva EV, Kuznetsov D, Tegenfeldt F, et al. OrthoDB v10: sampling the diversity of animal, plant, fungal, protist, bacterial and viral genomes for evolutionary and functional annotations of orthologs. Nucleic Acids Res 2019; 47:D807–D811

16. Consortium TU. UniProt: a worldwide hub of protein knowledge. Nucleic Acids Res 2019; 47:D506–D515

17. Kumar S, Stecher G, Suleski M, et al. TimeTree: A Resource for Timelines, Timetrees, and Divergence Times. Mol. Biol. Evol. 2017; 34:1812–1819

18. Camacho C, Coulouris G, Avagyan V, et al. BLAST+: architecture and applications. BMC Bioinformatics 2009; 10:421

19. Tange O. GNU Parallel 2018. 2018;

20. Katoh K, Standley DM. MAFFT Multiple Sequence Alignment Software Version 7: Improvements in Performance and Usability. Mol Biol Evol 2013; 30:772–780

21. Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. 2004; 32:1792–1797

22. Löytynoja A. Phylogeny-aware alignment with PRANK. Methods Mol. Biol. 2014; 1079:155–170

23. Castresana J. Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. Mol. Biol. Evol. 2000; 17:540–552

24. Guindon S, Dufayard J-F, Lefort V, et al. New Algorithms and Methods to Estimate Maximum-Likelihood Phylogenies: Assessing the Performance of PhyML 3.0. Syst Biol 2010; 59:307–321

25. Vilella AJ, Severin J, Ureta-Vidal A, et al. EnsemblCompara GeneTrees: Complete, duplication-aware phylogenetic trees in vertebrates. Genome Res 2009; 19:327–335

26. Shen W, Le S, Li Y, et al. SeqKit: A Cross-Platform and Ultrafast Toolkit for FASTA/Q File Manipulation. PLOS ONE 2016; 11:e0163962

27. Shannon P, Markiel A, Ozier O, et al. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. Genome Res 2003; 13:2498–2504

28. Thompson JD, Linard B, Lecompte O, et al. A Comprehensive Benchmark Study of Multiple Sequence Alignment Methods: Current Challenges and Future Perspectives. PLOS ONE 2011; 6:e18093

29. Stamatakis A. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. Bioinformatics 2014; 30:1312–1313

30. Ronquist F, Teslenko M, van der Mark P, et al. MrBayes 3.2: efficient Bayesian phylogenetic inference and model choice across a large model space. Syst. Biol. 2012; 61:539–542

Figure legends

Figure 1. AutoCoEv offers a simple and intuitive menu. *A) Preparation menu.* In steps 1-4, the script offers to automatically retrieve and prepare the needed databases. This includes downloading, extracting, fasta indexing and "trimming" genes and orthologous groups databases (e.g for genes of the reference "mouse", and orthologous groups at node "metazoa"). Step 5 processes the external species tree, substituting species names with their corresponding taxid, needed for the later steps of the workflow. *B) Main menu.* By carrying out sequentially the steps, the user can complete the whole process of correlation identification. Steps 1-3 deal with homologous sequences retrieval; steps 4-6 carry the identification of correct orthologues; steps 7-9 create the MSA and phylogenetic trees for orthologues; steps 10 and 11 run parallelized CAPS for each unique protein pair combination; steps 12-14 process the results and generate XML; step 15 simply exits the script.

Figure 2. AutoCoEv seamlessly connects the numerous steps of the pipeline. *A) Orthologues identification procedure.* Reading the user-provided lists of proteins of interest and species to be searched, the script communicates between databases to extract genes (ODB id) and orthologous groups (OG id) identifiers. Homologous sequences are then blasted against the UniProt sequences from the reference organism (e.g. mouse) in order to prepare a FASTA list of correct orthologues. *B) Alignments and proicessing.* Orthologues are then aligned by selected method (mafft, muscle or prank) and optionally processed by Gblocks, excluding regions of low quality. *C) Phylogenetic trees generation.* PhyML calculates trees from the MSA generated in the previous step, optionally using the external tree as a guide. The produced trees can be optionally rooted automatically by TreeBeST. *D) Create all unique protein pairs.* In preparation for the actual CAPS run, the script calls SeqKit and TreeBeST to ensure that only species that both paired proteins have in common are present in their MSA and trees. Since number of species affects the specificity of CAPS, the user can specify a threshold of the minimum permitted number of common species (e.g. 20). Each pair folder has two subfolders: for MSA and trees. *E) Running CAPS for each protein pair.* The script executes CAPS in each protein pair folder in a parallelized fashion via GNU/Parallel (see Figure 3 for details). *F) Results processing.* The output in each pairs folder is inspected, filtering columns by alignment quality and p-values of the correlation. Finally, the results are parsed into an XML file ready for the network analysis by Cytoskape.

Figure 3. Parallelization achieves great speed improvement of processes. AutoCoEv calls GNU/Parallel in order to utilize miltiple CPU cores on many steps. For a 6-core CPU, Parallel will spawn 6 simultaneous instances of CAPS, each operating in an individual protein pair folder, each running on a single core. When complete, another instance of CAPS for another folder will be spawned, taking advantage of the free core. If a core has two threads, then each process will occupy a single thread, effectively executing 12 processes at a time for a 6-core, 12-thread CPU.

Figure 4. Coevolutionary network. A total of 297 proteins were screened for coevolution and a subnetwork of those correlations with p-values < 0.00001 was selected. Colour code denotes p-value, while line thickness corresponds to the bootstrap value of the correlation – the proportion of the phylogeny where correlation was observed by CAPS.

Figure 1

A

```
Do initial preparations:
1) Download databases       4) Trim databases
2) Extract databases        5) Convert species to taxid?
3) Index FASTA database     6) [DONE AND CONTINUE]
```

B

```
Select a step:
 1) Pair UniProt <-> OrthoDB <-> OGuniqueID
 2) Prepare orthologues list (level: 32523)
 3) Get FASTA sequences of all orthologues
 4) Download sequences from UniProt (organism: 10090)
 5) Prepare BLAST database from UniProt sequences
 6) BLAST orthologues against UniProt sequence (10090, detailed: yes)
 7) Get FASTA sequences of the best hits (identity: 35.000; gaps: 25)
 8) [MSA] Create MSA with selected method (mafft-linsi)
 9) [TRE] Prepare trees (phyml, mafft-linsi, gblocks, exguide, rooted)
10) [RUN] Create all unique pairs (increment by 100)
11) [RUN] CAPS run (alpha: 0.01, mafft-linsi, gblocks, phyml)
12) [RES] Inspect CAPS results
13) [RES] Generate columns stats
14) [XML] Process CAPS results
15) [Exit script]
Your choice:
```

Figure 2

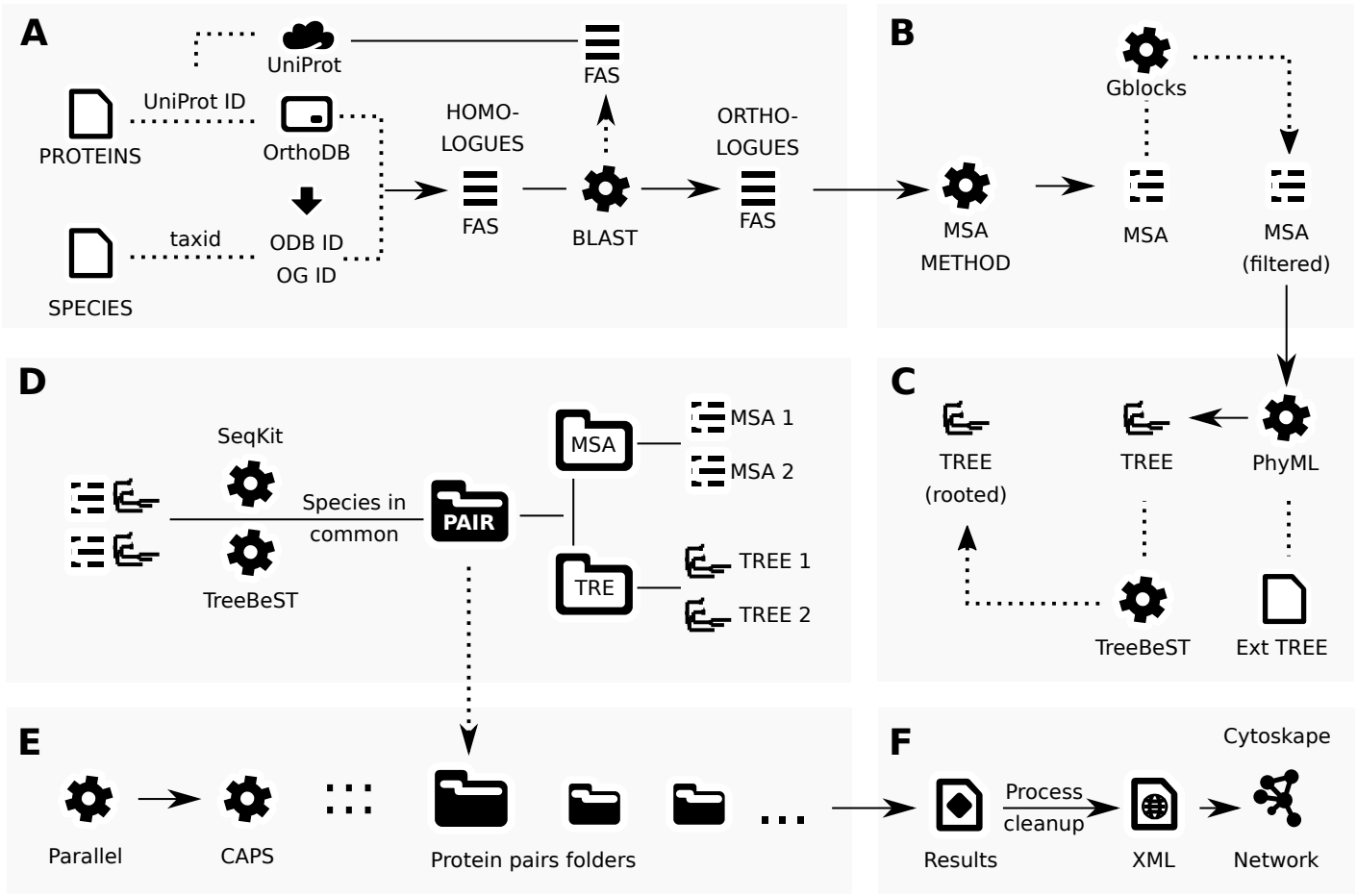

Figure 3

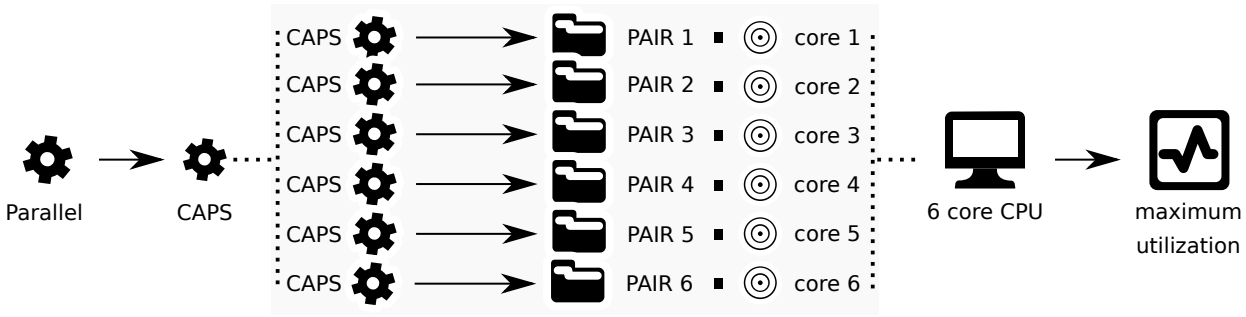

Figure 4