Ultrafast Clustering and Ranking of Macromolecular Structures Using uQlust

User Manual

Ver. 2.0

Rafal Adamczak & Jarek Meller, Cincinnati & Torun, 2014/2015/2016

Released as of July 2016; Available at http://github.com/uQlust (ver. 2.0)

Table of Contents

Section 1: What is uQlust?

uQlust is a software package for ultrafast ranking and clustering of macromolecular structures, including proteins and RNAs. uQlust combines versatile structural profiles of both proteins and nucleic acids with linear time algorithm for comparison of all pairs of models using 1D-Jury (Adamczak and Meller, 2011), and profile hashing for efficient and low memory footprint clustering of macromolecular structures, including hierarchical clustering. While reducing dramatically the computation time and memory requirements with respect to existing methods, uQlust yields comparable accuracies in protein and RNA clustering and model quality assessment.

Even though uQlust has been primarily developed for large scale structure prediction and molecular simulations for both proteins and RNA, in which case all models pertain to a particular macromolecule and are assumed to be of the same length, uQlust can also be used in conjunction with an arbitrary profile, such as the FragBag structural motif frequency profile. In latter case, protein structures of arbitrary length are first projected into a motif frequency vector (profile), thus enabling the use of 1d-jury and profile hashing-based clustering approaches implemented in uQlust.

The source code for uQlust was written in C# by Rafal Adamczak, based on a joined work with Jarek Meller (see references below). The code is easily portable between different operating systems, and the system independent pre-compiled executables can be run as long as .NET (Windows) or Mono (Linux) are installed (see also the next section). The source code, and a number of utilities, benchmarks and examples are available as part of the GitHub uQlust package at http://github.com/uQlust.

References:

R. Adamczak & J. Meller, *uQlust: Combining Profile Hashing with Linear-time Ranking for Efficient Clustering and Analysis of Big Macromolecular Data*, to be published
R. Adamczak, J. Pillardy, B.K. Vallat, J. Meller; *Fast Geometric Consensus Approach for Protein Model Quality Assessment.* Journal of Computational Biology 18(12):1807-18 (2011); PMID: 21244273 (for now, please cite the above 1D-Jury paper)

Section 2: Profiles and Profile Hashing in uQlust

This section contains a quick introduction to the profiles and profile hashing-based clustering heuristics available in uQlust. For more details see the last section of this manual and the manuscript.

The approach adopted in uQlust simplifies structure-to-structure comparison, ranking and clustering by projecting 3D coordinates into a suitable 1D structural profile that assigns each residue to a distinct state, e.g. exposed vs. buried. As shown in (Adamczak and Meller, 2011), using profile pre-processing one can implicitly compare all pairs of models and rank them using geometric consensus with a linear time complexity algorithm, referred to as 1D-Jury. While this approach has been developed primarily for fast assessment of similarity between all pairs of alternative 3D structures of the same macromolecule, it can be generalized to arbitrary structures (of any length) as long as they can be efficiently projected into fixed length structural profiles, such as fragment frequency profiles used by FragBag (Budowski-Tal et al., 2010).

Furthermore, structural profiles can be combined with profile hashing to enable efficient and low memory footprint hierarchical clustering. The main idea is to use structural profiles to define hashing keys and collate profiles/structures with the same keys into micro-clusters that are subsequently either tuned (with some level of profile coarse graining and further projections/filters) to obtain certain number of clusters and data coverage or aggregated hierarchically using the Hamming, cosine or other applicable distance measure. As a result, uQlust enables an approximate hierarchical clustering approach that achieves effective linear scaling for large data sets of macromolecular structures represented as 1D profiles (although it is not strictly linear in this case).

Building on these algorithmic engines, the following efficient clustering heuristics are implemented in uQlust:

- uQlust:Hash(K,F) that aggregates data into K clusters (comprising F% of data) by simply tuning-up the granularity of hashing keys;
- uQlust:Rpart(K,F) that uses a 1D-Jury reference-based partitioning of data while changing the radius of clustering to achieve K target cluster with F% of data;
- uQlust:Tree hierarchical clustering that proceeds to aggregate the initial hashing (or reference-based) micro-clusters using Hamming, cosine (or RMSD) distance.

The above heuristics can be combined with any of the profiles listed below. For each profile, its type (as defined by the macromolecule it applies to, i.e. either protein or RNA), the source of state assignment (^RNA-SS-TA is not computed internally and requires the DSSR utility to provide secondary structure and torsional angle state assignment for RNAs), the number of states and the size (length) of the profile are reported.

Profile Name	Type	Source	Number of States	Size
SS-SA	Prot	uQlust:DSSP	N_{SS} * N_{SA}	N_{res}
CA(SS)-NC(SA)	Prot	uQlust	N_{PSS} * N_{Cont}	N_{res}
CA-CM	Prot	uQlust	2	$N_{res}(N_{res}+1)/2$
FragBag	Prot	uQlust:FragBag	Max frequency	N_{frag}
RNA-SS-LW	RNA	uQlust:RNAview	N_{SS} * N_{LW}	N_{base}
RNA-SS-TA^	RNA	DSSR	N_{SS} * N_{TA}	N_{base}
RNA-P-CM	RNA	uQlust	2	$N_{base}(N_{base}+1)/2$
RNA-FragBag	RNA	uQlust	Max frequency	N_{frag}
User generated	Prot/RNA	User defined	User defined	User defined

Each of the above listed profiles, can be used for either model assessment using 1D-Jury (denoted as uQlust:1D-ProfileName), or explicit clustering with profile hashing, using hash keys generated with a profile of choice to provide an initial 'slicing' of data. Depending on the application, profiles may be tuned and refined by the user by changing the level of coarse-graining (of residue states), combining different profiles, setting weights to emphasize the importance of certain states (e.g., secondary structure vs. solvent accessibility), or by changing the fragment library to be used in the case of FragBag type fragment frequency type profiles. Predefined workflows allow one to start with a default setup, where an appropriate profile and clustering parameters are selected for the problem at hand. See sections 5 and 6 for examples.

Section 3: System Requirements

Windows:

1. Windows 64 bit system
2. .NET v.3.5 or higher (if .NET is not installed, it should install automatically)

Linux:

1. Mono v.3.8 or higher

2. boost library (typically included in Linux distributions)

NOTE: Since uQlust is written in C#, it is compiled to a bytecode (CIL) which is contained within the uQlust.exe file. Thus, the main uQlust module does not need re-compiling. However, two additional modules derived using C/C++ distributions of DSSP and RNAview utilities (kindly made available by their authors – see references) do need to be compiled into libDSSP.so and libRNA.so files. In order to create these two files, just run the 'make' command in the main distribution directory (referred to as uQlustDistr). The boost library is used to compile the DSSP module and create libDSSP.so. Check if the files libDSSP.so and libRNA.so have been created in the main uQlust directory, which contains the uQlust.exe file.

Section 4: Executing uQlust

Windows

Interactive (GUI) version:

```
uQlust.exe
```

Terminal (BATCH) version:

```
uQlustTerminal.exe -f configuration_file
```

Linux

Interactive (GUI) version:

```
mono   uQlust.exe
```

Terminal (BATCH) version:

```
mono uQlustTerminal.exe -f configuration_file
```

NOTE: Examples of configuration files with pre-defined workflows are included in the **workFlows** subdirectory of the uQlustDistr directory (see also BATCH mode section below).

NOTE: When running uQlust GUI interface in the advanced mode for the first time, one needs to set a number of options for uQlust that define the input file location and type, the type of task to be performed (e.g., ranking vs. clustering), 1D-profile, methods to be used etc. We strongly suggest exploring the ‘easy mode’ with a number of pre-defined workflows and well tested parameters before using the ‘advanced mode’.

Section 5: Quick Start Using Easy Mode

The GUI interface provides two options referred to as ‘easy’ and ‘advanced’ mode, respectively. The first dialog window allows one to switch between the modes, while subsequent dialog windows allow one to navigate and select pre-defined workflows in the case of the ‘easy’ mode. The use of the latter is strongly recommended for first time uQlust users. While it limits the number of options, it enables testing typical ranking or clustering applications using a number of illustrative data sets and pre-loaded recommended parameters for each workflow.

The user can select the type profile (1D or residue-based vs. fragment-based vs. user defined), the type of macromolecule (protein vs. RNA), and the type of analysis (clustering with either Hash/Rpart or Tree methods vs. ranking with 1D-jury method). Some parameters can be adjusted even in the ‘easy mode’, e.g., when applicable the user can select between Hamming or cosine or RMSD distance measures. For each of the examples, data directories (pointing data sets included in the distribution) and default parameters are preloaded. The following example workflows are included:

- Clustering of protein structures with a fragment-based (FragBag) profile and either uQlust:Hash, uQlust:Rpart or uQlust:Tree: villin headpiece structures obtained by using MD simulations starting from three distinct conformations are used (thus, the expected result is three well defined clusters in this case);
- Ranking of protein structures using DSSP-based 1D profile (SS-SA) and 1D-jury method: models for one of the TASSER benchmark targets are used (thus, structures closer to native are expected to obtain higher scores);
- Clustering of RNA structures with fragment-based (RNA-FragBag) profile and either uQlust:Hash, uQlust:Rpart or uQlust:Tree: 5S, 16 S and 23S ribosomal RNAs (of different length) are used (thus, the expected result is three well defined clusters in this case);
- Ranking of RNA structures using RNAview-based 1D profile (RNA-SS-LW) and 1D-jury method: models for one of the FARNA benchmark targets are used (thus, structures closer to native are expected to obtain higher scores).

These pre-defined workflows can also be executed in the batch mode (the corresponding configuration files are included in the workFlows directory), or refined using 'advanced' mode (see next section).

Section 6: Examples of Typical Workflows

In order to simplify (and illustrate) the use of uQlust for typical ranking or clustering application, a number of pre-defined workflows and the corresponding data sets are included in the workFlows directory. Several specific examples of such workflows are described below. In each case, in addition to configuration files and command line options for the batch mode, the corresponding steps and options for the uQlust GUI interface (using 'advanced mode') are also provided to guide running these workflows interactively.

NOTE: In order to use a different (than pre-specified illustrative) dataset, the workflow configuration file must be edited to modify the "Path to data directory#" variable/string. Of course, various parameters can also be adjusted as needed.

Ranking Protein Models Using 1Djury

Batch mode:

```
mono uQlustTerminal.exe -f workFlows/protein/uQlust_config_file_1djuryRanking.txt
```

GUI:

- If current data directory is empty press '***Add***' button, and select data/proteins/1abv
- Select '***Ranking***' option in the main menu, and press radio button '***1Djury***'
- Press '***Setup profiles***' button to open the profile editor, and press 'load profile' (second to last icon); select '***SS3_SA9_jury.profiles***' as the profile type, and press "Save" button;
- Press '***Run***' button, specify the name of the process (any string), and press 'OK' button.

NOTE: The results are ready when the computation time is displayed in the 'Clustering results' panel. Activate (if not active) the results line for a job by pressing left mouse button, and then press right mouse button and select 'Show Results' item.

Ranking RNA Models Using 1Djury

Batch mode:

```
mono uQlustTerminal.exe -f workFlows/rna/uQlust_config_file_RNA_1djuryRanking.txt
```

GUI: Follow the steps for ranking protein models above, while selecting RNA as the type of macromolecule, and RNA-SS-TA.profiles as the profile type.

NOTE: In this case, the option -m RNA is not needed for the batch mode because a profile file (which has been generated already using DSSR) is used directly, and there is no need to process (or use) PDB files at this point anymore.

Reference-based Clustering of Protein Structures

Batch mode:

```
mono uQlustTerminal.exe -f workFlows/protein/uQlust_config_file_Rpart.txt
```

GUI:

- If current data directory is empty press '***Add***' button, and select data/proteins/1abv
- Select '***Clustering***' item in the main menu, and then 'Hash cluster'
- Select '***Rpart***' and specify the target number of clusters, K, and percent of data in these clusters, F
- Select '***1Djury***' in 'Find consensus states' to define the type of reference vectors
- Press '***Setup profiles***' button to open the profile editor and press 'load profile' (second to last icon), select '***SS3_SA9.profiles***' in the profiles directory, and press "Save" button
- Press '***Run***' button, specify the name of the process (any string), and press 'OK' button.

Hashing-based Clustering of Protein Structures

Batch mode:

```
mono uQlustTerminal.exe -f workFlows/protein/uQlust_config_file_Hash.txt
```

GUI:

- If current data directory is empty press '***Add***' button, and select data/proteins/1abv
- Select '***Clustering***' item in the main menu, and then 'Hash cluster'
- Select '***Hash***' choose the method for hash key pruning ('Entropy' or 'Meta columns'), and specify the target number of clusters, K, and percent of data in these clusters, F
- Select '***1Djury***' in 'Find consensus states' to define the type of reference vector for hashing
- Press '***Setup profiles***' button to open the profile editor and press 'load profile' (second to last icon), select '***SS3_SA9.profiles***' in the profiles directory, and press "Save" button
- Press '***Run***' button, specify the name of the process (any string), and press 'OK' button.

Fast Hierarchical Clustering of Protein Structures

Batch mode:

```
mono uQlustTerminal.exe -f workFlows/protein/uQlust_config_file_Tree.txt
```

GUI:

- If current data directory is empty press '***Add***' button, and select data/proteins/1abv
- Select '***Clustering***' item in the main menu, and then select '***uQlust:Tree***' radio button
- Select '***1Djury***' in 'Find consensus states' to define the reference vector for hashing
- Press '***Setup profiles***' button to open the profile editor and press 'load profile' (second to last icon), select '***SS3_SA9.profiles***' in the profiles directory
- Select RMSD as the distance measure for aggregation of micro-clusters, and press "Save" button
- Press '***Run***' button, specify the name of the process (any string), and press 'OK' button.

K-means Clustering of RNA Structures

Batch mode:

```
mono uQlustTerminal.exe -m RNA -f workFlows/rna/uQlust_config_file_RNA_Kmeans.txt
```

GUI:

NOTE: Other examples of configuration files with pre-defined workflows are included in the **workFlows** subdirectory of the uQlustDistr directory.

Section 7: Defining Workflows Using Advanced Mode in GUI

The following section describes step by step how to define workflows and set up relevant options when using uQlust graphical interface (executed by running uQlust.exe) in the 'advanced mode' (which can be selected in the first uQlust GUI dialog window).

STEP 1: DEFINE INPUT DATA

In order to run any ranking or clustering method using uQlust, one needs to define the source of structural data to be used, in particular the subdirectory that contains structures for analysis. The 'Add' button in the main form, which opens automatically when executing uQlust GUI, allows the user to select the source of structures/models. Such selected directory should appear in the current data directory list.

Adding Input Directories

One may add multiple directories using 'Add' button; each of these directories will be analyzed separately by running the method selected in the next step (ranking or clustering of some type) on structures from each of those directories separately. One may also remove directory from the list by using the ‘Remove’ button to remove selected, or all directories by pressing the ‘Remove All’ button.

Defining a List of Input Directories with Structure Data

There is also possibility to add a list of directories that are defined in an external, user generated file by using the radio button “File with list of the directories”. Each directory in the file should be listed in a separate line, e.g.,

```
H:\casp10\T0650
H:\casp10\T0652
H:\casp10\T0653
```

Defining Other General Options

Use the submenu ‘Settings’ in the main menu to define the following items (if applicable):

- The type (extension) of files with input structures to be analyzed: from the directory specified only files with this extension will be read by uQlust;
- The name of the directory where profile files to be generated (see STEP 2) will be stored, and, if exist, will be read, so there is no need to generate profiles again;
- Define the **maximum number of CPU cores** that will be used during calculations;

STEP 2: DEFINE MACROMOLECULE TYPE

Use the submenu ‘Settings’ in the main menu to define the type of macromolecule (uQlust macromolecule mode) to be analyzed:

- Define uQlust macromolecule mode by selecting ‘Protein’ or ‘RNA’ in the ‘Settings’ menu.

NOTE: Some structural profiles and options are only available in Protein mode, which is much better developed in the current version of the program. For example, in the case of the protein mode it is possible to use only alpha carbon atoms, or all atom models.

NOTE: As an alternative to the RNA DSSR-based profile currently available in uQlust, one can use an arbitrary, user defined profile that applies to RNA structure analysis (see STEP 4 below).

STEP 3: DEFINE ANALYSIS TYPE

Ranking methods

1. ‘***1DJury***’: the algorithm ranks structures based on the similarity between their profiles. It calculates similarity of particular structure to all other available structures, as in 3D-jury

algorithm. The difference is that profiles with smart preprocessing are used here, reducing the complexity of the algorithm to O(n), where n is the number of structures;

2. '***3DJury***': the algorithm ranks structures based on their 3D superposition-based distance with O(n^2) time complexity (for the comparison of all pairs of models). NOTE: this can be very slow, and not significantly more accurate than 1D-jury; to be used primarily for evaluation purposes on small data sets;
3. '***Sift***': in this algorithm, score for each of the structure is calculated based on the analysis of packing of amino acid residues within the first and second contact shell in terms of radial distribution function; this is very useful when assessing if protein models are physically plausible, and can be used for ranking as well.

Clustering methods

1) **'Hash clustering':** available options in this group include the profile hashing-based '***uQlust:Hash***' (uQlust:Hash(K,F)) and reference-based '***uQlust:Rpart***' (uQlust:Rpart(K,F)) heuristics;
2) **'uQlust:Tree'**: using '***uQlust:Tree***' (or approximate hierarchical clustering starting from hash key-based initial slicing of data and subsequent aggregation and tree building) for large data sets;
3) **'Hierarchical clustering'**: available options include '***Agglomerative***' (traditional full hierarchical clustering) with either Hamming distance or RMSD, and '***Fast top-down***' clustering using 2-means to split the data, and **'Davies-Bouldin top-down'** approach that uses repeated runs of K-means to define an optimal split of the data at each level;
4) **'K-means'**: either classical K-means with a random start, or enhanced K-means with 1D-jury centroids (referred to as '***uQlust:K-means***').

NOTE: While uQlust:Tree is very fast and is meant to be applied to large data sets, traditional hierarchical approaches that are provided for direct comparison may turn out to be very slow, and use a lot of memory – use with caution.

NOTE: In the traditional, full hierarchical clustering (referred to as **'Agglomerative')** one can select any of the typical linkage criteria: single, average or complete linkage. Moreover, one can select among several distance measures, including 'Rmsd' and 'Maxsub' for proteins/RNAs (using their 3D structures), or 'Hamming' and 'Cosine' for any profile.

NOTE: For each of the clustering methods, centroids/reference structures at each level may be found be averaging structures that belong to the cluster with the selected distance measure, or by using 1d-jury (recommended). In the latter case, one needs to check "Use 1d-jury to find reference structure" checkbox and specify profile that will be used by the 1d-jury method.

Define Clustering Target

For profile and reference-based clustering methods, referred to as uQlust:Hash(K,F) and uQlust:Rpart(K,F), respectively, the main parameters K and F must be selected to define the target clustering structure, in analogy to K-means (although the actual clustering algorithms are very different – see Appendix).

In the main menu, choose ‘Clustering’ item, and then ‘Hash cluster’. Select ‘Hash’ or ‘Rpart’, depending on the clustering approach to be used, and specify the target number of clusters, K, and the fraction (percent) of data to be contained in those K target clusters, F.

NOTE: The choice of K and F should be carefully assessed, just like the choice of K for K-means. In fact, one of the goals of uQlust is to make it possible to run repeatedly fast clustering to enable assessing the quality of results with different parameters and effectively estimate the optimal number of clusters K.

Distance Measures

Several standard distance measures can be used for either traditional K-means or hierarchical clustering, and either 3D structure or profile based aggregation of initial micro-clusters in uQlust:Tree. The following options can be selected in uQlust GUI (or specified in batch configuration files):

- ‘***Rmsd***’ – root means square deviation of atomic positions to be used for 3D structure-to-structure comparison and distance computation; there are two options (radio buttons)
    - ‘All atoms’ – calculate RMSD using all available atoms
    - ‘Only CA’ – calculate RMSD using only C_alpha atoms
- ‘***Maxsub***’ – only C_alpha atoms are used in this case; the MaxSub 3D structural similarity scores are converted into a distance measure;
- ‘***Hamming***’ – Hamming distance to be used for profile-based clustering (especially for binary profiles, such as CA-CM);
- ‘***Cosine***’ – Cosine distance to be used for profile-based clustering (especially for fragment-based profiles).

STEP 4: DEFINE STRUCTURAL PROFILES TO BE USED

Structural profiles are used in uQlust to generate a suitable 1D projection of 3D structure, with the goal of achieving efficiency while minimizing the loss of accuracy. Each of the profiles invokes some definition of structural states. The number of states depends on the profile used. For example, amino acid residues in a protein may be assigned to some discrete secondary structure and/or solvent accessibility states. In the case of secondary structures, one can use three states (helix, beta strand or coil), or 8 states, as defined using the DSSP utility, for instance. The actual selection and tuning-up of structural profiles can be done using a profile editor, which is available as part of the GUI interface, as described in the next section.

Predefined Profiles in uQlust

- Arbitrary, user defined profiles – profiles that are generated by the user and loaded into uQlust, while defining all the states to appear in the profile, and their transition table in the ‘Profile manager’
- SS – secondary structure profile generated by using the built-in DSSP module; there are 8 states initially that can further be reduced using ‘Profile manager’
- SA – solvent accessibility also generated by the DSSP module, there are up to 10 states corresponding to 10% RSA intervals, from 0 to 100% RSA, as defined by the DSSP utility
- Contact – contact-based profile that is based on the number of neighbors in the first contact shell (distance between side chains geometric centers smaller than 8.5A); there are 10 states, corresponding to the number of neighbors – all residues with more than 9 neighbors are assigned state ‘9’.
- Contact CA-CM – the same as Contact but using C_alpha atoms only
- SS CA – approximate secondary structure profile with states defined based on the distance between C alpha atoms; the distance is calculated between i-th and i+2 (dist2) residues, as well as i-th and i+4 (dist4) residues; there are 9 states including state U (unphysical).
- ContactMap – a binary contact map is used to represent the structure of the protein, the threshold for contacts vs. non-with contact set to 8.5A, and the sequence distance between residues set to 12; the actual profile is defined by concatenating the rows of the upper corner of the contact map into a vector of length n*(n+1)/2; since such defined profile for a large protein could be very long, contact map profiles are preprocessed to decrease the amount of needed storage space; specifically, positions with only zeros in the whole ensemble of sparse profiles are removed, with the implication that all contact map profiles must be generated again when a new structure is added to the ensemble (after profiles where generated); in order to force regenerating profiles, one has to delete the current profile file, or change directory where the profiles are stored.

NOTE: For a more complete description of structural profiles implemented in uQlust, please see the Appendix.

Section 8: Profile Manager

Both, arbitrary (outside) and predefined (internal) profiles can be defined and/or further adjusted by the user in the graphical profile editor available in uQlust GUI (called ‘Profile manager’).

Operations on Profiles

The following operations on profiles are available in profile manager:

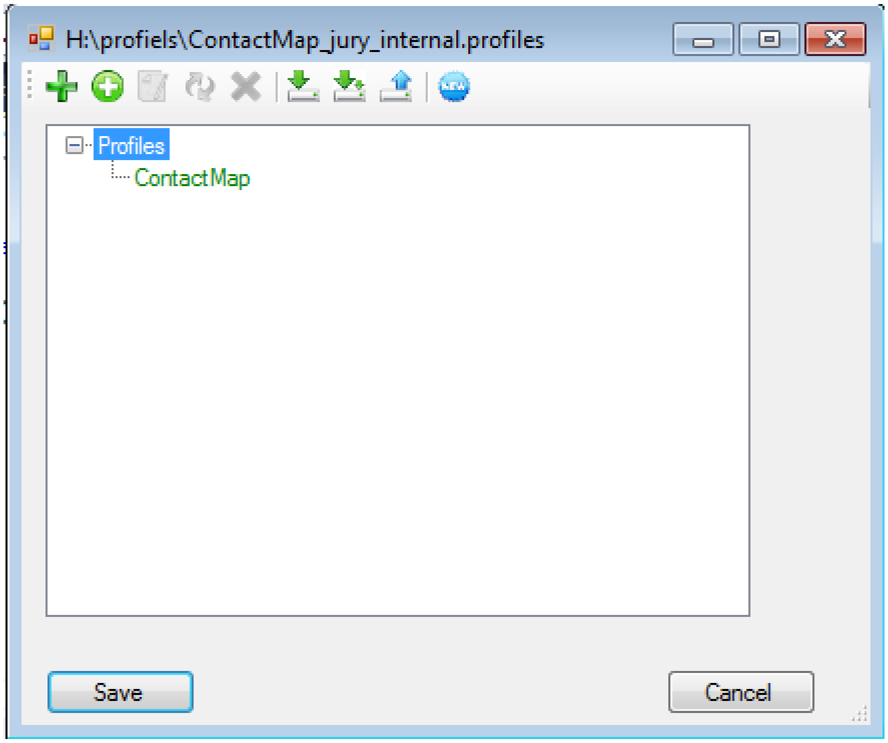
• Edit profile to change state definition or transition table	📝
• Change the status of selected profiles from active to non-active and vice versa. Active profiles are depicted in green color, whereas non-	🔄

active in red. Only active profiles are used to represent structures	
• Remove profile from the list	✖
• Save as, save, load	⤓ ⤓ ⤒
• Remove all profiles from the list	NEW
• Add a profile that will be generated by some external program	✚
Operation	**Icon**

NOTE: To make any change in an existing profile, it has to be selected first.

Selecting Profiles

The profile manager can be used to further configure profiles to be used for ranking or clustering. The figure below shows a window of profile manager with ContactMap profile added to the profiles list. To add additional profiles to the list, press the ⊕green ‘round plus’ button when ‘Profiles’ node is selected. If the ContactMap node is selected, then the new profile will be created as its child, which may be useful if some other profile is required to create the new one.



Tuning-up Predefined Profiles

Profile editor allows one to change (reduce) the number of states in the profile by combing some of the states into one state, and to define weights for each type of transition between the states, reflecting the implied similarity between the states and effective penalty for not being in the same state. For example, for secondary structure states H, E and C, states H and E can be considered as the most opposite states, while C could be regarded as somewhat similar to both H and E. Consequently, a difference between H and E should be penalized much more strongly, compared to a transition between H and C. Such flexibility allows one to define profiles with some implicit notion of similarity between the states, while still providing a linear complexity solution to the ranking problem (see also Adamczak and Meller, 2011). For example, in order to calculate the hamming distance between two secondary structures profiles

1: C C C H H H C C C
2: H C C E H H C C C

One can consider that the difference between H-E is more important than C-H, it is possible to add weight for the transition H-E higher then C-H.

Combining Profiles

'Profile manager' can also be used to define a new profile by combining a number of already defined profiles, including the 'primitive' internal profiles available in uQlust. For example protein structure can be represented by secondary structure and solvent accessibility (0 – fully buried state, 9 – fully exposed state) profiles.

```
C  C  C  H  H  H  C  C  C
9  9  9  3  3  3  7  7  7
```

All selected profiles will be combined into one profile that combines together states at each position into joined states. Here, the final SS-SA profile takes the following form:

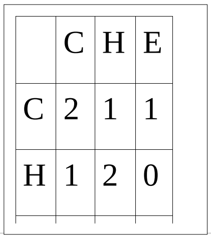
C9 C9 C9 H3 H3 H3 C7 C7 C7

When combining profiles, their corresponding transition tables will also be merged. There are two types of merging methods: one for similarity based ranking or clustering (used by 1djury), and the other for distance based clustering. In the former case, table merging is made by multiplying scores, whereas in the latter case by adding distances.

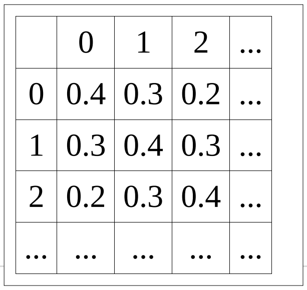
Similarity transition tables

SS transition table SA transition table Combined table

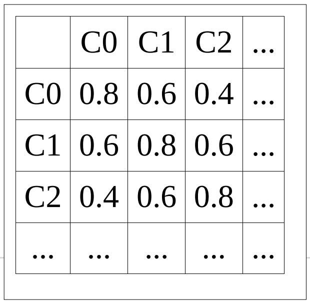
Similarity ('jury') transition tables:



	C	H	E
C	2	1	1
H	1	2	0

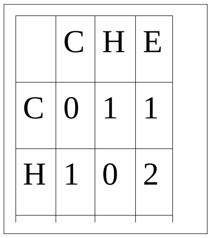


	0	1	2	...
0	0.4	0.3	0.2	...
1	0.3	0.4	0.3	...
2	0.2	0.3	0.4	...
...	...	...	...	...

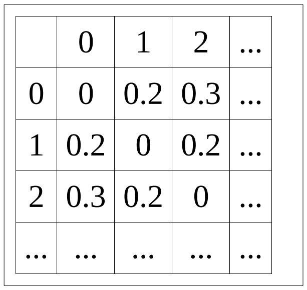


	C0	C1	C2	...
C0	0.8	0.6	0.4	...
C1	0.6	0.8	0.6	...
C2	0.4	0.6	0.8	...
...	...	...	...	...

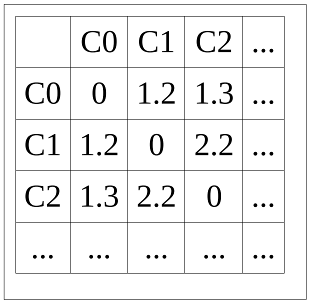
Distance transition tables:



	C	H	E
C	0	1	1
H	1	0	2



	0	1	2	...
0	0	0.2	0.3	...
1	0.2	0	0.2	...
2	0.3	0.2	0	...
...	...	...	...	...



	C0	C1	C2	...
C0	0	1.2	1.3	...
C1	1.2	0	2.2	...
C2	1.3	2.2	0	...
...	...	...	...	...

Defining Arbitrary Profiles

In the case of user defined profiles, the user needs to generate them using a suitable external application. For example, DSSR generated assignment of torsional states for RNA can be converted into a set of discrete states defined by the user based on some notion of the resolution of these states (and appropriate projection into discrete states for continuous variables) that would be required to capture distinct conformations. These states (and in particular symbols for each state) must be consistent with the internal definition of the corresponding user defined profile in uQlust. The states for external, user built profiles can be defined using profile editor (see below). Obviously, the output from an external application must be processed to generate the corresponding profile files (e.g., one file per RNA structure) with state symbols that are consistent with the internal profile definition, so that the profile can be then loaded into uQlust.

External Profile Files

External profiles must be stored in a file with the following format:

>name of structure

name_of_profile profile definition of profile

Example 1:

```
>Run8_2_627.pdb

ContactMap profile 0 0 0 0 1 0 1 0 1 0 1 1 1 1 1 1 0 0 0 0 1 1 0

>Run0_1_191.pdb

ContactMap profile 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Example 2:

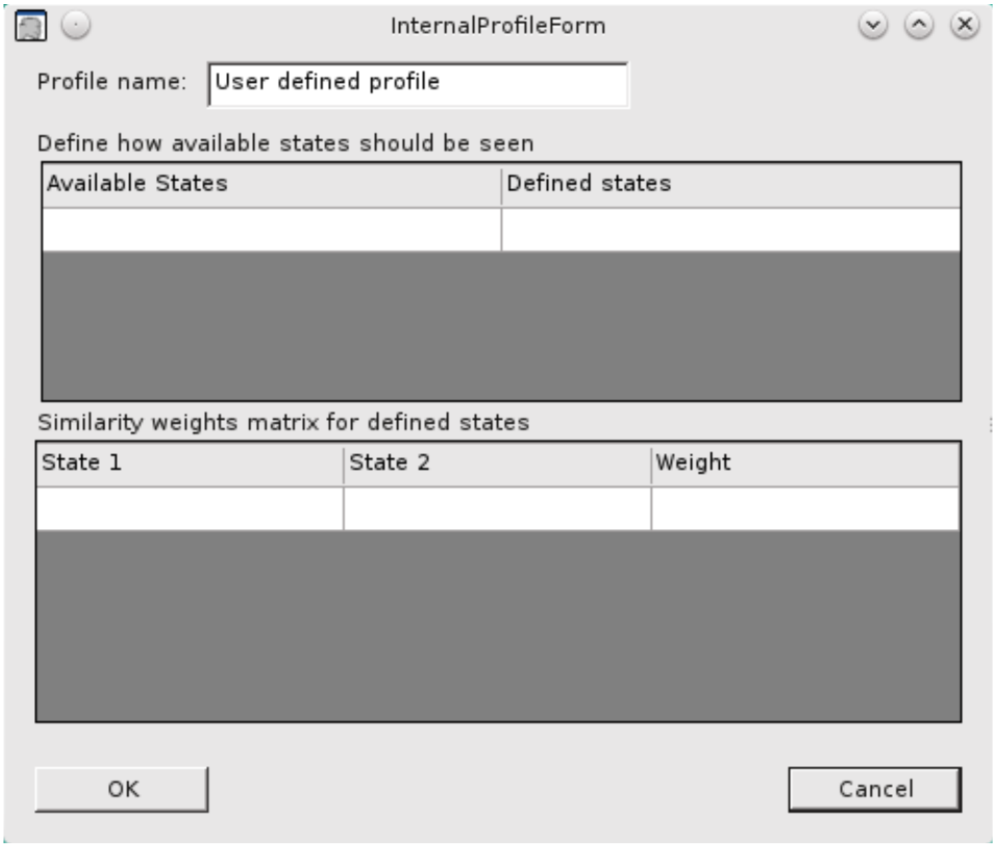
```
>d4579.pdb
```

```
SS profile C C H H H H H - T T S S S S H H H H T T T S

SA profile 9 8 4 6 7 4 0 4 6 2 1 5 6 6 1 1 1 8 6 4 9 7

>d11939.pdb

SS profile C C C - H H H H H T T S S S S S S S G G G S

SA profile 9 9 8 7 6 0 2 6 2 1 4 9 3 8 3 0 6 8 2 7 2 0
```

The external profiles must be aligned, and thus each profile must be of the same length. The gaps in the alignment are represented by character '-', as can be seen in Example 2. Moreover, for each of the structures, one can define more than one profile, as shown in Example 2, as long as the names of these profiles are different. In order to load an external profile one has to select '***File with aligned profiles***' radio button in the main panel.

Defining New Profiles Using Profile Editor

In order to define a new profile, go to the profile editor (by using setup profile button in any clustering or ranking method). Press icon '***New profile***' (🆕), then press icon '***Add internal profile***' (➕) and select '***User defined profile***'. Internal profile form should appear on the screen.

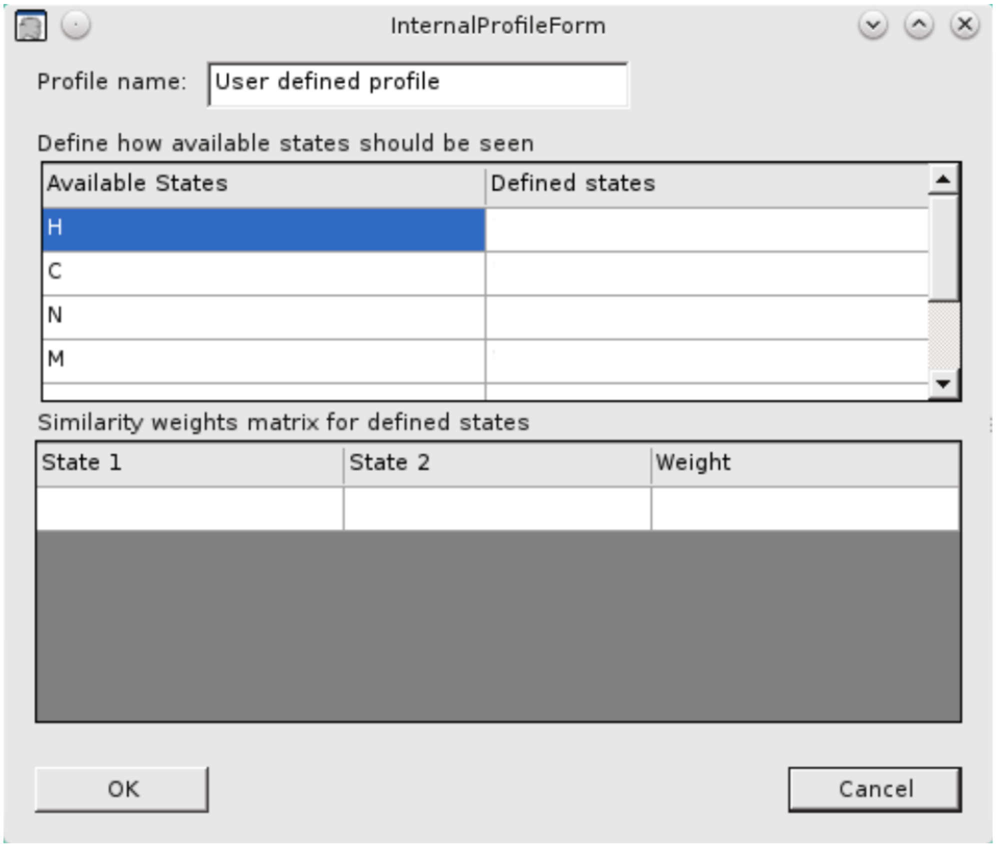


InternalProfileForm

Profile name: User defined profile

Define how available states should be seen
Available States
Defined states
Similarity weights matrix for defined states
State 1
State 2
Weight
OK
Cancel

In the profile name field, specify the name of the profile (this name will be used by profile editor - it is not the name of the profile file). The next step is to define states that may appear in the profile. All possible states in a profile must be specified in the left panel of the section '***Define how available states should be seen***'. For example, in order to define a state denoted as 'H', click on the empty row of '***Available states***' column and press H and then return. State H has been defined, in the same way one may defined other states, e.g., C, N, M.



InternalProfileForm

Profile name: User defined profile

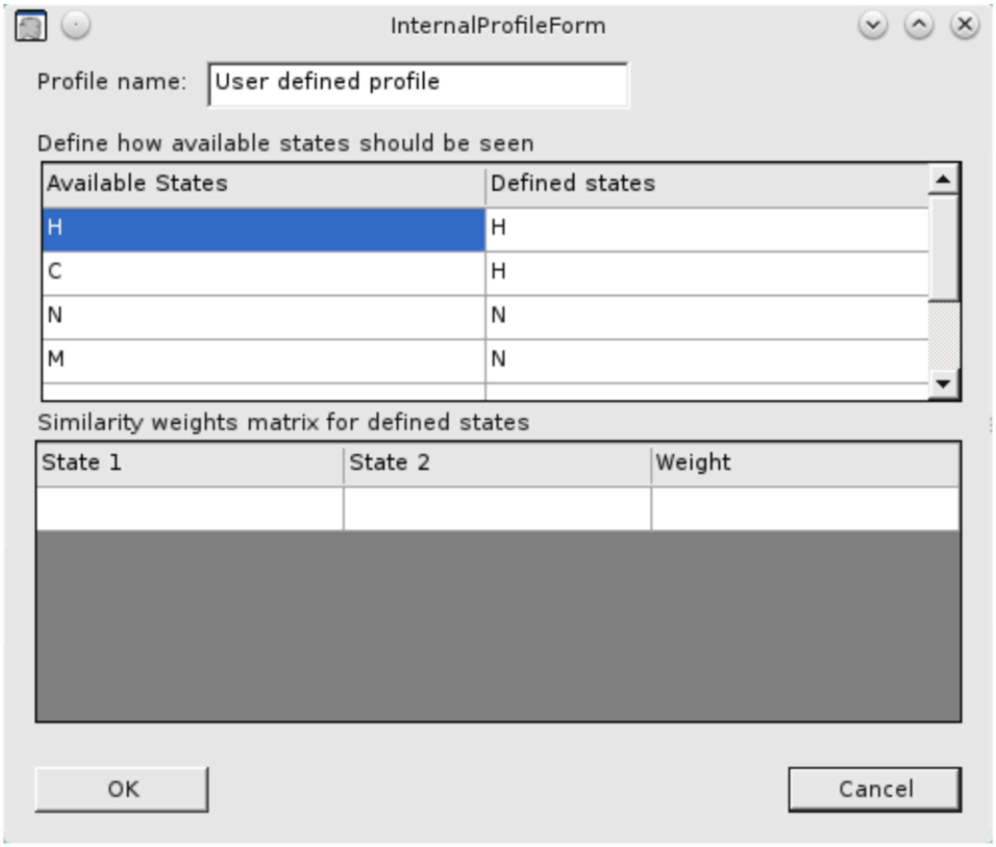
Define how available states should be seen

Available States	Defined states
H	
C	
N	
M	

Similarity weights matrix for defined states

State 1	State 2	Weight

On the right panel called '***Defined states'***, the actual definition of each state listed on the left side must be provided. In the simplest case, left panel and right panels might be identical, but in the case one would like to merge some states, it can be done in 'Defined states' panel. In the example below, H and C states are defined as H and N and M as N.



Profile name: User defined profile

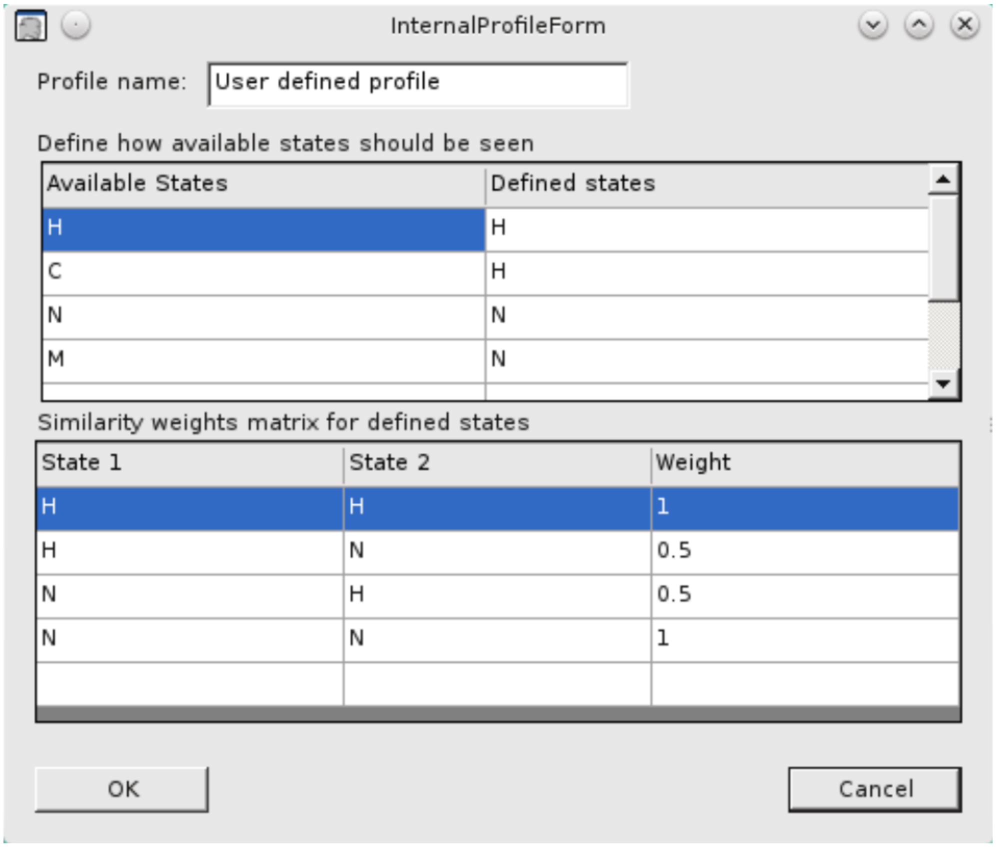
Define how available states should be seen

Available States	Defined states
H	H
C	H
N	N
M	N

Similarity weights matrix for defined states

State 1	State 2	Weight

When all states are defined, a ‘Similarity weight matrix’, or ‘Distance weight matrix’ in the case of profiles used for clustering in conjunction with the Hamming distance measure, must be provided. For each state, in our example (H,N), similarity (or distance) to all other states must be defined. If for some combination of states, their similarity is not provided, ten it is assumed to be 0.



InternalProfileForm

Profile name: User defined profile

Define how available states should be seen

Available States	Defined states
H	H
C	H
N	N
M	N

Similarity weights matrix for defined states

State 1	State 2	Weight
H	H	1
H	N	0.5
N	H	0.5
N	N	1

When all is finished, press OK to save the profile.

The profile is defined now, and it can be used for aligned profiles that are stored in a file with the format described in the previous section (assuming that all state symbols are consistent). If some other states are used in the profile file, the file will not be loaded, and an error message will be generated.

Section 9: Profile Hashing

In order to perform clustering using profile hashing, a structural profile and a method to define a reference (or consenus) profile must be properly defined. The latter is used to generate hash keys that are used to represent each structure in terms of a binary profile, and as a basis for initial slicing of the data into micro-clusters with the same value of the hashing function, given its granularity (pruning and smoothing of the profiles and the corresponding hash keys).

Reference Profile

The reference profile may be defined by a consensus state at each position of the profile, or by using 1d-jury ranking to define a 1D-jury centroid as a reference vector (recommended).

Subsequently, each profile is transformed into a hash key based on consensus profile as follows: each position in the profile is compared with the corresponding position in the consensus profile, if these are the same states, then 0 is put into the key (otherwise 1 is added).

Consensus states	CCCCCCCCEEEEHEEE
Current profile	CCCCHCCCHHEEHEEH
Key	0000100011001001

The hash keys built in this way are used in the hash table as a basis for fast aggregation of data into micro-clusters: if the keys are the same, the structures are put in the same cluster (see also Appendix).

Hash Key Pruning and Regularization

Hash keys defined for a profile of choice, as described above, can be shortened (pruned) or regularized (smoothed). In uQlust these two options are used to effectively provide additional aggregation with the goal of achieving either the target number of micro-clusters in uQlust:Tree, or the target number of clusters in uQlust:Hash(K,F) and uQlust:Rpart(K,F).

Profile Regularization

For **regularization**, the size of the window, the type of the profile (which can be different from the profile for keys generation), and the threshold distance options must be provided. Symbols with red color represent current regularization window:

Current key	0000100011001001

The distance between the consensus (reference) and current profiles within the window of 7 positions considered here is 1. Hence, if the threshold distance is set to <=1, then the corresponding 'smoothed' key takes the following form:

After regularization	0000000011001001

Profile Pruning

For **pruning**, one can effectively combine/merge clusters by removing key positions (columns) that are deemed least informative, with the goal of achieving the required number of target clusters in uQlust:Hash(K,F), or to change the granularity of the resulting micro-clusters in uQlust:Tree. Two heuristics are available to prune hash keys:

- ***Entropy*** – a number of hash key positions with the lowest entropy (with some definition of what constitutes states that are regarded as sufficiently similar, or 'close enough', to consider them as identical for the calculation of entropy) are removed from the key to achieve the targeted number of clusters K through the resulting collapse/aggregation of 'slices' of data with the same value of the hashing function;
- ***Meta columns*** – this method decreases the length of the key by 3 by analyzing three position at a time (starting from the first column in the profile) to identify those triples that represent local fragments in the same (e.g., secondary structure) state, as

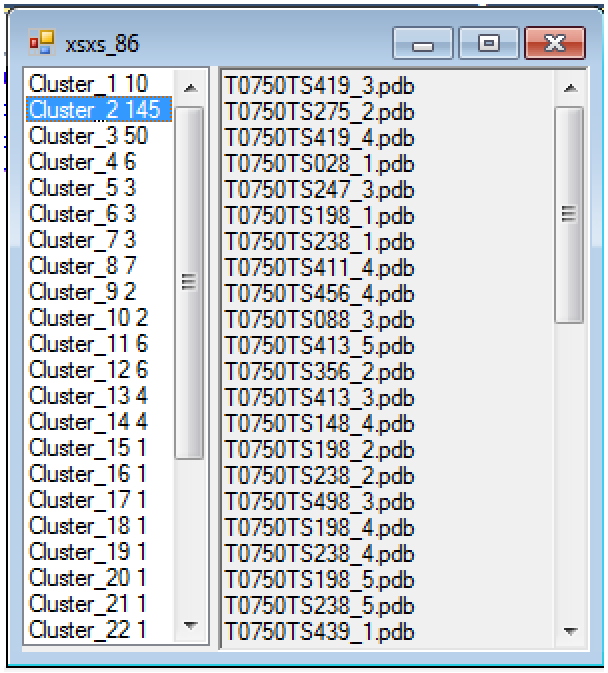
opposed to those that correspond to transitions from one state to another (e.g., from alpha to coil); specifically, for i-th position in the key, i+1 and i+2 are considered as well, and based on their states new key is built: if those 3 positions have the same state then 0 will be added to the 'meta column' key (otherwise 1 will be added).

Section 10: Analysis and Visualization of Results Using GUI

Results are available in the section 'Clustering results' panel. When clustering is finished, the font color is switched to green, otherwise it is red. One must click LMB on the row with results to be visualized. Once the row is selected, click RMB to select 'Show Results' option from the context menu. Several visualization methods are available, as described below.

Text List View

This option is available for non-hierarchical clustering methods.



- Cluster_1 10
- Cluster_2 145
- Cluster_3 50
- Cluster_4 6
- Cluster_5 3
- Cluster_6 3
- Cluster_7 3
- Cluster_8 7
- Cluster_9 2
- Cluster_10 2
- Cluster_11 6
- Cluster_12 6
- Cluster_13 4
- Cluster_14 4
- Cluster_15 1
- Cluster_16 1
- Cluster_17 1
- Cluster_18 1
- Cluster_19 1
- Cluster_20 1
- Cluster_21 1
- Cluster_22 1

```
T0750TS419_3.pdb
T0750TS275_2.pdb
T0750TS419_4.pdb
T0750TS028_1.pdb
T0750TS247_3.pdb
T0750TS198_1.pdb
T0750TS238_1.pdb
T0750TS411_4.pdb
T0750TS456_4.pdb
T0750TS088_3.pdb
T0750TS413_5.pdb
T0750TS356_2.pdb
T0750TS413_3.pdb
T0750TS148_4.pdb
T0750TS198_2.pdb
T0750TS238_2.pdb
T0750TS498_3.pdb
T0750TS198_4.pdb
T0750TS238_4.pdb
T0750TS198_5.pdb
T0750TS238_5.pdb
T0750TS439_1.pdb
```

Left panel of the window above contains a list of clusters that consist of the label "Cluster", its number, and the numbers of structures in the cluster. Right panel contains the list of structures in the currently selected cluster.

Traveling Salesman View

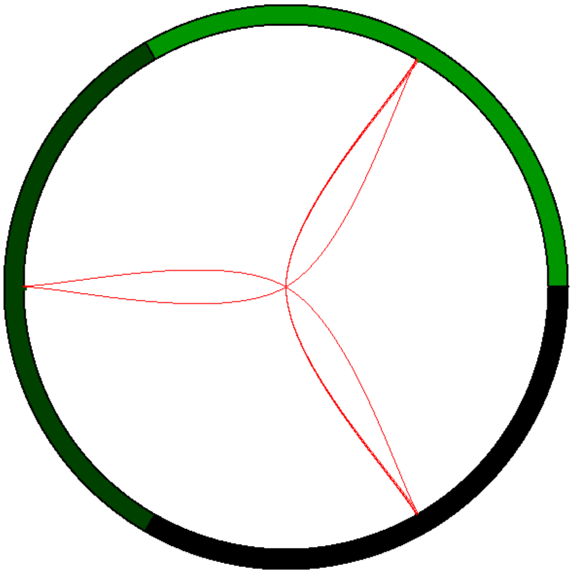
This view is available for non-hierarchical clustering methods. It allows one to visualize transitions between clusters, assuming that the sequential order with which these clusters are 'visited', e.g., in the course of MD simulations, is informative. In this view, clusters of conformations correspond to 'cities' in the traveling salesman problem, although the purpose here is to map out the 'trajectory', rather than solving any optimization problem. A file that defines the order of structures is needed (e.g., using the order of MD frames). When this method is chosen, the 'open file' dialog will appear to enable selecting a proper with following format:

```
structure_name rank

…
```

For example:

```
Run0_0.pdb time0

Run0_1.pdb time1

Run0_2.pdb time2

Run0_3.pdb time3

Run0_4.pdb time4

Run0_5.pdb time5

Run0_6.pdb time6

Run0_7.pdb time7
```

If the first three structures belong to cluster_1, the next two structures to cluster_2, and last three to cluster_3, one will observe transition from cluster_1 to cluster_2 and from cluster_2 to cluster_3. Picture below shows results for 3 clusters. Clusters are positioned on a circle, the bigger the cluster the more space on the circle it takes. Transitions between clusters are denoted by red lines. One may click LMB on the cluster to see only transitions from and into this cluster.



Sunburst View

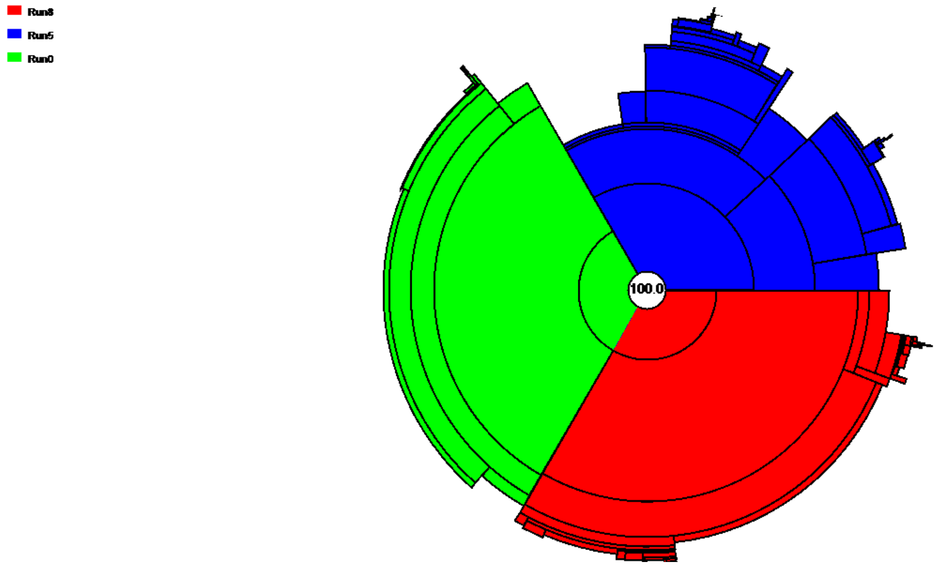
This method is available for hierarchical clustering methods only. It uses a file with label (class) definition, based on some prior knowledge, so that unsupervised clustering can be superimposed with some external class labels. The format of this file is as follows:

```
structure_name label
```

For example:

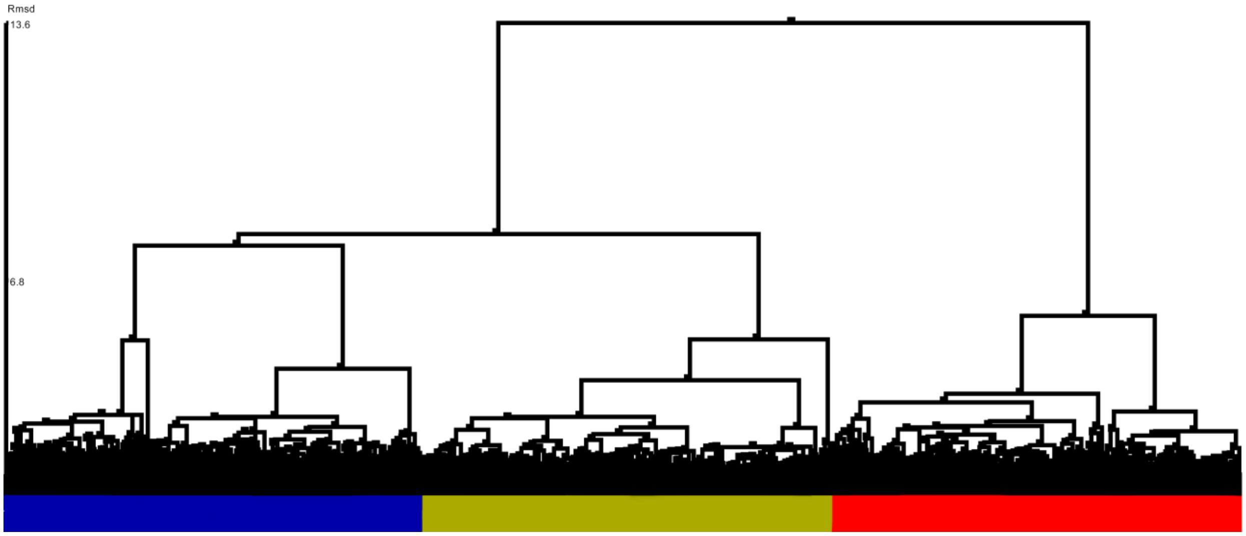
```
Run0_0.pdb MarkovState0

Run0_100.pdb MarkovState0

Run0_101.pdb MarkovState1

Run0_102.pdb MarkovState1

Run0_103.pdb MarkovState2

Run0_104.pdb MarkovState2
```

The hierarchical clustering in this view is represented by nested circular blocks, denoted by black lines within a wheel. The bigger the cluster the bigger part of the wheel it takes. The distances from the middle (white) circle represent the distances of the representatives of the clusters from each other. The number in the middle shows the percentage of the available data currently shown, one may click LMB on any cluster to see more details, or click RMB to return to the previous view. One may also change the definition of colors for each class by clicking LMB on the region where color and the label appears (on the picture left upper corner).



Dendrogram View

This method is available for hierarchical clustering methods only. The black squares on the dendrogram denote the points where one may use either LMB or RMB click. The LMB click will show the text list with structures that belongs to the cluster. The RMB click allows one to zoom in and out, so that only clusters below the selected point will be shown. To return to the previews picture one may click RMB, pointing mouse outside of any square region. As in the Wheel method, external labels can be superimposed with the unsupervised clustering.



APPENDIX: FORMAL DEFINITION OF MAIN METHODS AND PROFILES

Profile Hashing

Profile hashing is used in conjunction with 1D-Jury as a basis for low memory footprint and ultrafast clustering heuristics available in uQlust, including approximate hierarchical clustering. In uQlust, binary hash keys are generated with a 1D profile of choice by comparing each profile with a reference profile that obtains the maximum 1D-Jury score (which can be computed in linear time).

Clustering Heuristics

The first heuristic briefly described here is a profile hashing-based clustering that is referred to as **uQlust:Hash(K,F)**, where K defines the number of target clusters, and F denotes the fraction of data that should be contained within those K clusters. Hash(K,F) starts by initial data 'slicing' into micro-clusters with the same value of the hashing function. Subsequent agglomeration into K clusters (comprising F% of data) is obtained by simply changing the granularity of hash keys, which is achieved by removing sufficient number of least informative profile hash key positions.

Another heuristic in uQclust is a form of reference-based partitioning, which is referred to as **uQlust:Rpart(K,F)**. As before, this new heuristic relies on the initial identification of 1D-jury 'centroid' for the entire data set, as a suitable reference conformation. While Rpart also represents all profiles in terms of binary hash keys, the subsequent partitioning of data proceeds very differently. Namely, Rpart identifies recursively macro-clusters centered on a reference profile by adjusting radius of clustering to achieve certain K clusters comprising F% of data.

Finally, in the case of approximate hierarchical clustering, which is referred to as **uQlust:Tree**, the first step is analogous to that used for Rpart(K,F) or Hash(K,F), except that a large K is used to induce a large number of small clusters, and F is set to 100% to include all data. In the next step, a 1D-jury centroid is computed for each micro-cluster, and from this level traditional average distance agglomerative (bottom-up) hierarchical clustering with either Hamming distance (for arbitrary profiles), or RMSD (only for proteins or RNAs) is applied.

Definition of Structural Profiles

An arbitrary, residue level or fragment-based profile generated by using an external application can be used for ranking or clustering in conjunction with Hamming or cosine distance-based ranking and clustering approaches.

Internally computed protein profiles (starting from a set of all-atom or reduced PDB or DCD files) include:

i) SS-SA or secondary structure (SS) – solvent accessibility (SA) profile, which assigns each amino acid residue to one of up to $N_{SS} = 8$ secondary structures (by default $N_{SS} = 3$), and one of up to $N_{SA} = 10$ solvent accessibility states (by default $N_{SA} = 2$ with the threshold of 20% relative solvent accessibility (RSA) separating 'buried' from 'exposed' states); the DSSP utility (Touw et al., 2015) is implemented internally to provide the definition of SS and RSA;

ii) CA(SS)-NC(SA) or approximate distance dependent secondary structure (SS) – solvent accessibility (SA) profile, which can be used for $C_{\square}$ models, and assigns pseudo-secondary structure states based on distances (in Ang) between C_α atoms (CA); Specifically, a combination of $d4 = d(C_{\alpha,i},C_{\alpha,i+4})$ and $d2 = d(C_{\alpha,i},C_{\alpha,i+2})$ is used to define the states: intervals (4.0,6.0) and (6.0,8.0) are considered for d2, and intervals (4.0,7.0), (7.0,9.0), (9.0,11.0) and (11.0,13.0) are considered for d4, such that $4.0 < d2 < 6.0$ and $4.0 < d4 < 7.0$ defines a pseudo-helix, while $6.0 < d2 < 8.0$ coupled with $7.0 < d4 < 9.0$ defines a pseudo-strand, and six additional pseudo-secondary structure states are defined as other combinations of d2 and d3 intervals, resulting in the total of 9 states (together with 'OTHER' state, hence $N_{PSS} = 9$); solvent accessibility is approximated by the number of contacts (NC) within 8.5 Ang radius around $C_{\alpha,i}$ (which is capped at 10, hence $N_{Cont} = 10$, although further coarse graining is possible);

iii) CA-CM (contact map), which is also applicable to both atomistic and reduced models, and consists of the top triangle of the binary contact map, where $d(C_{\alpha,i},C_{\alpha,j}) < 8.5$ Ang.

In analogy to protein profiles, 1D RNA profiles for ranking and clustering (of equal length RNA models) are built either using backbone phosphorus atom contact map (denoted as RNA-P-CM) where $d(P_i,P_j) < 15.5$ Ang, $|i-j| > 11$, or by considering a combination of secondary structure and base pairing states generated by using RNAview. Namely, a simplified secondary structure assignment (stem vs. loop, $N_{SS} = 2$) is combined with a coarse-grained Leontis and Westhof (LW) classification of base-pairs as one 15 different types based on nucleotide pairs (AT vs. GC), glycosidic bond orientation (cis vs. trans), interacting edges (Watson-Crick, Hoogsteen, Sugar Edge, and their frequently observed combinations plus ‘Other’ state), resulting in 30 distinct states (denoted as RNA-SS-LW). In addition, a pre-defined workflow (denoted as RNA-SS-TA) is available to simplify using DSSR utility as a source of state assignment for RNAs. RNA-SS-TA combines simple secondary structure state assignment (N_{SS} = 2) with distinct torsional angle states (N_{TA} = 5), defined as combinations of DSSR epsilon-zeta BI and BII backbone states with chi syn- and anti- states (plus ‘other’ state). The resulting 10 distinct states that can be further split based on base-pair type assignment, similar to that used for RNA-SS-LW.

Another type of profiles available in uQlust takes a form of a structural motif/fragment frequency profile to represent arbitrary structures (of any length). For proteins, uQlust uses the FragBag library of 400 backbone fragments of length 11 residues, while its custom developed RNA-FragBag counterpart is used for RNAs. RNA-FragBag consists of 92 representative coarse-grained 5-mer backbone (phosphorus atom) RNA fragments, derived from the RNA05 set of RNA structures. Such derived fragment were subsequently clustered by using uQlust:K-means and RMSD.

References:

Adamczak, R. and Meller, J. (2011) *Fast geometric consensus approach for modeling quality assessment*, J Comp Biol 18(12): 1807-1818

Budowski-Tal, I. et al. (2010) *FragBag, an accurate representation of protein structure, retrieves structural neighbors from the entire PDB quickly and accurately*, Proc. Natl. Acad. Sci. USA 107 (8): 3481-3486

Coutsias, E. A. et al. (2004) *Using quaternions to calculate RMSD*, J. Comput. Chem. 25: 1849–1857

Das, R., Baker, D. (2007) *Automated de novo prediction of native-like RNA tertiary structures*, Proc. Natl. Acad. Sci. U.S.A.(104): 14664–14669

Elmer, S. et al. (2005) *Foldamer dynamics expressed via Markov state models. II. State space decomposition*, J. of Chem. Phys. 123 (11), 114903

Ginalski, K. et al. (2003) *3D-Jury: a simple approach to improve protein structure predictions*, Bioinformatics 19: 1015-1018

Jamroz, M. et al. (2013) *CABS-flex: server for fast simulation of protein structure fluctuations*, Nucl. Acids Res. 41: W427-W431

Lu, X. and Olson, W. K. (2003) *3DNA: a software package for the analysis, rebuilding and visualization of three-dimensional nucleic acid structures*, Nucleic Acids Res. 31(17): 5108-21

Nugent, T. et al. (2014) *Evaluation of predictions in the CASP10 model refinement category*, Proteins (82) 98–111

Siew, N. et al. (2000) *MaxSub: an automated measure for the assessment of protein structure prediction quality*, Bioinformatics 16, 776-785

Touw, W. G. et al. (2015) *A series of PDB related databases for everyday needs*, Nucleic Acids Research 43(Database issue): D364-D368

Wu, S. et al. (2007) *Ab initio modeling of small proteins by iterative TASSER simulations*, BMC Biology (5): 17