1 Locomotif Introduction

In recent years many new noncoding RNAs have been found and many more are believed to exist. These are active molecules with a function of their own that depends both on their structure and some specific sequence motifs. Then, there are classical RNA motifs responsible for regulatory control mechanisms. Using the fact that these motifs are composed of typical structure parts such as stems, bulge loops or single stranded regions, we chose to develop a graphical editor for defining RNA motifs. The motifs should then be translated into a description that allows to search them. Having the ADP framework ready for RNA folding with the thermodynamic algebras already implemented, it is not difficult to imagine using this approach for finding RNA motifs. Instead of a general RNA folding grammar, particular grammars for every motif must be generated, but based on the same thermodynamic algebras.

1.1 General Concept

The approach of visual definition of RNA motifs and subsequent generation of search programs is based on two attributes of RNA motifs:
First, they are tree-like structures with a root (the start of the underlying sequence, termed the "5' end") and corresponding substructures. This aspect facilitates both the storage of structures in the graphical editor as well as the translation into tree-like XML documents. And second, RNA motifs are composed of a limited set of "building blocks", namely stems, bulges, internal loops, hairpin loops, multiloops (multifurcations) and single stranded regions. This allows us to construct any motif iteratively by placing the required building blocks next to each other.

1.2 System Architecture

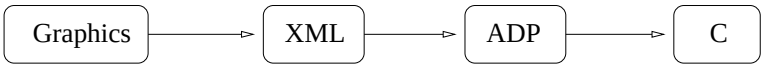

Figure 1: The graphical programming system is composed of 4 layers from graphics via XML and ADP code to an executable C-program.

The graphical programming system consists of four stages illustrated in Figure 1. The main part of the system is the graphical editor for molecular RNA motifs. It is implemented in Java, utilising Java Graphics2D for the visual components. The editor serves as the interface through which the user visually defines an RNA motif. The motif is then translated into an XML representation containing all biologically relevant information. Subsequently, the XML code is translated into a declarative ADP program for the motif. This translation is also implemented in Java. Finally, the ADP grammar is compiled to an executable C-program, the motif matcher, by the ADP compiler. Using the generated motif matcher, new ocurrences of the RNA motif can be located in input RNA/DNA sequences.

The four-tier composition of the programming system is devised to allow for an integration within a client-server architecture as shown in Figure 2.

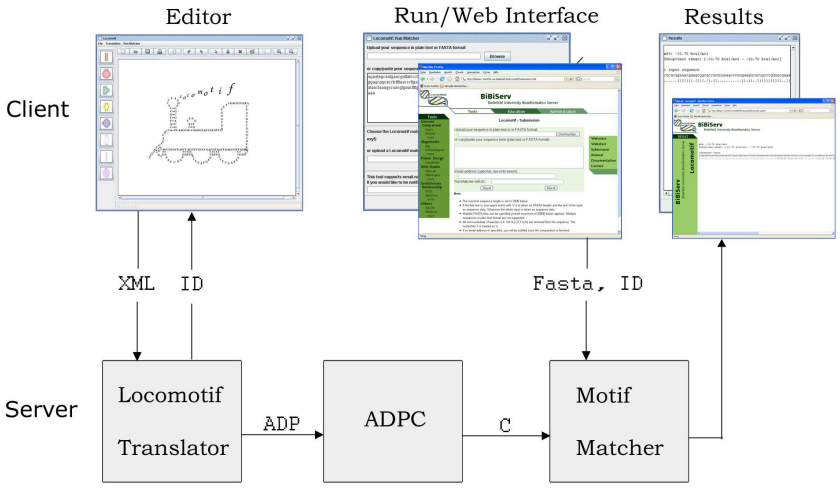

Figure 2: Overall structure of the graphical programming system. Rectangles are programs, ovals are data formats.

The graphical editor (Locomotif) is accessible on our webserver, BiBiServ[1], as an application through Java Web Start. Running under any common browser and operating system, Java Web Start automatically checks for updates of the application and downloads all needed resources without requiring any installation procedures from the user.

The XML representation of the designed RNA motifs serves as a transport layer between Client (Graphical Editor) and Server. Using an XML schema, we can verify on the server side that only valid information is sent from the client to our server. This eliminates the risk of subsequent compilation errors and reduces any issues regarding the security of information content sent to the server.

A unique identifier (ID) is returned to the client through which it can access the generated search tool for a certain time period (currently 10 days). The ID is stored during the current use of the programming system and can also be saved in a file. Optionally, the user can choose to have this information sent to an email address together with a link to a submission web page.

The next step on the server side is the translation of the information stored in the XML document into declarative ADP code and the subsequent compilation to executable C-code.

Once compilation of the matcher is completed, the search program can be run through the editor where the user can input RNA sequences directly or upload FASTA-files together with the obtained ID. Alternatively, the matcher can be accessed via the submission web page. The search tool is found according to the unique ID and applied onto the given sequence. The result is then presented in an extra window of the editor or on a web page. A third option is to use the ID on the download web page to obtain the C-sources of the matcher together

[1]Bielefeld Bioinformatics Server, http://bibiserv.techfak.uni-bielefeld.de/{locomotif}

with a Makefile to run the search program on one's own system.

2 Running the Locomotif editor: Java Webstart

The Locomotif editor is available on our webserver, BiBiServ[2], as a Java Web-Start application. Based on Java WebStart, the Locomotif editor can simply be started by clicking on the Locomotif Icon. To use WebStart technology for the Locomotif system, a Java Runtime Environment (JRE) based on at least J2SE 5.0 must be installed on your system. Usually, Java will already be available on your system. Otherwise, please refer to http://java.sun.com/javase/downloads/index.jsp. In most cases though, simply clicking the Locomotif icon should open the Locomotif editor. Then, Java Web Start automatically checks for updates of the application and downloads all needed resources without requiring any installation procedures from the user.

3 Designing a motif

In Locomotif, the design of an RNA motif is a stepwise process. Building blocks of structural elements are placed next to each other to construct the overall backbone of the motif. Then, detailed information on these structural elements can be added via graphical interfaces. At all times, the user can edit the existing structure by delete or detach operations or other forms of interaction. A detailed introduction to each phase of motif design is given in the next paragraphs.

3.1 Building Blocks

On the left side of the Locomotif editor are buttons for the 9 different types of building blocks as shown in Figure 3. The first 5 from top to bottom include typical structural elements such as stems/stacking regions, hairpin loops, bulge and internal loops and multiloops. Then, there are two kinds of compound building blocks representing either a combination of stems, bulges and internal loops (the ClosedStruct) or a combination of all 5 structural building blocks (the ClosedEnd). Finally, there is a building block for simple pseudoknots and for single strand connections or extensions. Sometimes, not all buttons are active if the use of some building blocks (e.g. single strands) is not possible at the moment.

3.1.1 Handling

Generally, a building block is chosen by clicking on the appropriate button. Then, the chosen block is drawn next to the mouse cursor as soon as it moves within the drawing surface. You can place the building block anywhere on that surface by doing another mouseclick. Then, another building block can be added to the existing structure by moving it in proximity of an available structure end. It jumps to the correct position and is fixed with another mouseclick. Alternatively, you can choose to begin a new motif part by placing the building block in another remote location of the drawing surface. If you are not satisfied

[2]Bielefeld Bioinformatics Server, http://bibiserv.techfak.uni-bielefeld.de/{locomotif}

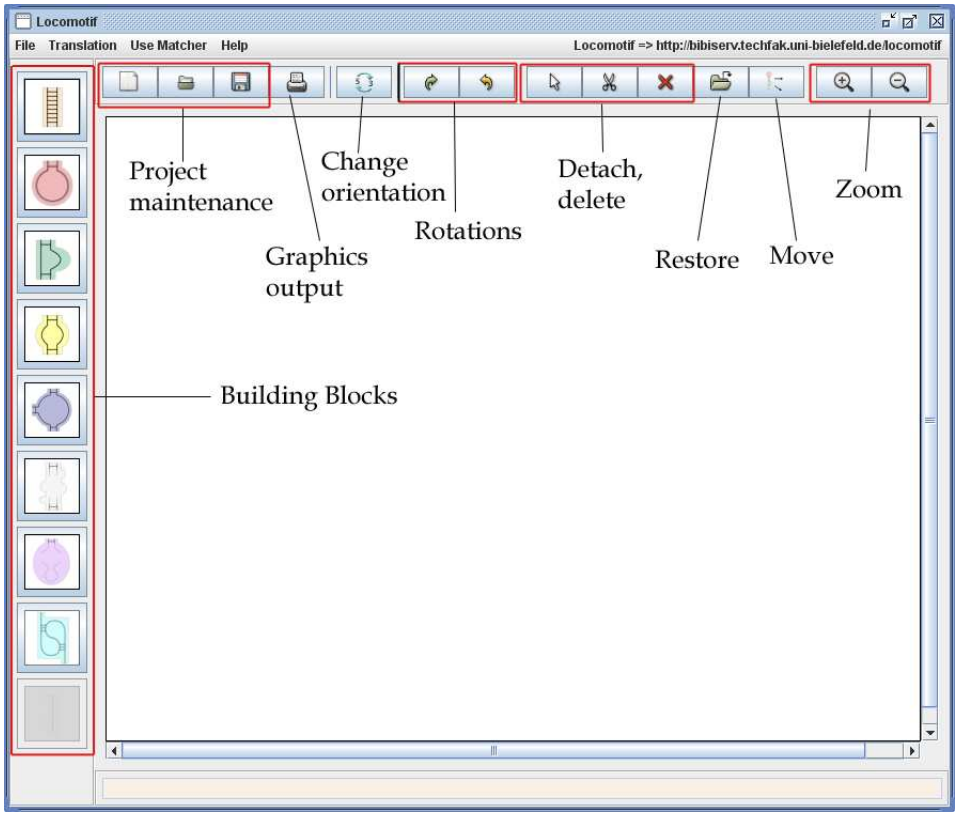

Figure 3: Screenshot of the Locomotif editor with annotated button functions.

with your choice of building block, you can always remove it from the mouse cursor by doing a right click with the mouse.

Single Strands. To use a single strand, move it to one open end and fix it by doing a mouseclick. Then, a connecting line will be drawn between the motif part and the mouse cursor and you can choose the final location of the other end. Either, click somewhere within the drawing surface to extend the motif part or move the cursor in proximity of another motif end to form a connection by another mouseclick. A simple rectangle is used to display the single strand building block. If both ends connected by the single strand are on the same or similar horizontal axis, it can be difficult to see or access the single strand building block. Therefore, it is a good idea to place motif ends on different horizontal axes.

3.2 Building Block Restrictions

While the Locomotif building blocks were designed with the intention to enable addition of any building block to any other, some restrictions are in place.

• You cannot add a hairpin loop to a motif part which has only the one remaining open end. Since we do not allow for circular structures, we must prevent this addition to ensure the correctness of the underlying motif. The same thing holds for the ClosedEnd compound building block.

• You cannot attach two compound building blocks to each other because the same structure would then be defined redundantly. This would cause ambigious results with the same hit being reported numerous times.

• Single strands are intended to act as motif part extensions or connections. Therefore, they can only be added to completed structure parts and then be used to extend these or connect them to other parts. You cannot extend both ends of a motif part, if there are other parts remaining, since that would prevent connections within the overall motif structure. Adding single strands to two sides of a helical element always creates incorrect structures in the building block concept of RNA motifs: Using two single strands and a hairpin loop instead of an internal loop and a hairpin loop violates the design concepts and would lead to incorrect search programs using the existing translation rules. So, in order to use single strands, you must first complete the design of the entire motif with only the connection gaps remaining and then fill these by connecting them with single strands.

3.3 Motif Refinements

For each building block, annotations can be added to refine the motif. A graphical interface is opened by double-clicking on a building block where you can input the information. This includes size restrictions effective upon the building block (local size), as well as those effective upon the substructure rooted at the building block (global size restrictions). Additionally, sequence motifs can be specified for loop regions and basepairs. Any stored textual input can be removed by deleting it and pressing apply or ok.

Here is a list of additional refinements specific for certain types of building blocks:

Stems Either a basepair string in the format "GC;AU;NN;CG" or a motif for one of the strands can be specified. The sequence start always coincides with the start of the stem building block (it can be moved inwards using N bases in front of the motif). If the motif is to occur after e.g. 3-4 bases, two neighboring stem building blocks must be used. The first having an approximate size of 3-4 basepairs and the second containing the sequence motif. Another option for the stem is to allow small loop interruptions of 1-2 bases in which case no sequence motif can be defined. These restrictions arise from the structure of the underlying grammars describing the motif for the search program.

Bulge and Internal loops For the bulge loop, the location of the loop can be chosen via the interface and the effect is displayed in the drawing surface immediately. For internal loops, size and sequence information can be given for each of the loop regions.

Multiloops The multiloop interface is based on an interactive graphics that allows the user to select the individual loop segments or basepairs by clicking on them. Once highlighted in this manner, for each loop segment, a size restriction can be given. Also, for each loop segment or basepair, sequence information can be stored. Finally, in another interactive graphics, the number and location of the exits can be chosen by clicking on the exit rectangles. Occupied exits are indicated by grey dots whereas open ends are indicated by red dots. Exits with red dots can be removed by clicking on the rectangle and new exits can be added at rectangles without a dot. A minimum number of 3 exits is required.

Pseudoknots For each of the 2 stems, size restrictions can be given. For the 3 loops, either a size restriction can be given or you can choose to allow internal folding of the loop. In that case, interior substructures within that part of the pseudoknot are allowed, yet you cannot demand a particular secondary structure.

Single Strands For a connecting single strand, you can choose to allow free optional folding of the single strand into another motif part.

Hairpin Loops and ClosedEnds Obviously, for these building blocks ending a motif part, global size and local size are the same and thus you cannot input global size information.

ClosedStruct For a ClosedStruct on the other hand, it is not possible to demand a particular local size for the building block due to restrictions arising from the underlying ADP motif grammars.

3.4 Interactions

Several types of interaction are possible to facilitate design of motif structures. These can be accessed via the buttons on top of the drawing surface as depicted in Figure 3. To the left are buttons for project maintenance. You can create a new project, open an existing one or save the current one. The print button is used for generating graphical output of the motif design. Either a standard graphics format such as jpeg or png can be produced or a svg-file can be obtained that can be converted to other publication quality formats such as ps.

Then, there are several buttons influencing the graphics. You can change the orientation of the structure, i.e. exchange the 5' and the 3' end. Doing this, you will be asked if the stored sequence information shall be reversed as well. Next, you can use the two arrow buttons to rotate the entire structure including all motif parts by 45 degrees. Then, you can select a building block and use one of the following buttons to detach or delete it. Or you can use the detach/delete button and choose the building block by clicking on it. The scissors indicate the detach operation in which case, the chosen building block with all the refinements stored for it will be reattached to the mouse cursor. It can then be placed somewhere else on the screen. The cross represents a delete operation in which case the building block is removed from the project. You can also restore the project to the previously stored version. Finally, you can

move the entire structure around or zoom in and out of the view to obtain a better overview on the overall motif design.

3.4.1 Restrictions: Single Strands

Again, some restrictions are in place to prevent the accidental creation of incorrect secondary structures. This includes both structures which are just temporarily out of order while editing the motif and those which present a wrong overall result of motif design. While it is desirable to prevent occurences of the second case, it can be easier for the user to go from a temporarily wrong motif to a correct final one. Nevertheless, we do not allow even temporarily incorrect motif designs to prevent the second case from happening.

The most obvious restriction needed is again the way single strands must be handled. If you wish to edit some part of the motif by removing a building block, you must first remove all single strands from the entire structure. You cannot simply detach them, but must delete them. In all cases where a building block directly neighboring a single strand is to be removed, this makes very good sense. The single strand does not have an active open end and thus, no building blocks can be attached to it. Giving an active open end to the single strand itself could easily lead to the creation of knotted structures that do not fulfill the overall 5'-3' orientation anymore.

Yet, there are those cases where a building block remotely located from any single strand is to be replaced by another or simply removed from the structure. Like, e.g. detaching a hairpin loop and placing it somewhere else for a while. Then, removing the neighboring bulge and using an internal loop instead. And finally, reattaching the previous hairpin loop. Such an editing operation does not endanger the correctness of the final motif and any intermediate step does represent a potentially correct secondary structure. Yet, we are not allowed to do this if any other part of the motif contains a single strand. A potential solution would be to freeze the remaining motif design and to restrict editing to the site in question so that no changes can be made to the rest of the structure and most importantly, no single strands may be added to any open end at that time. After all editing operations are done and a closed motif is recreated, the rest of the motif could be defrozen and thus incorrect motif designs are prevented. This remains future work as of now.

3.5 Approaching motif design

Given the single strand restrictions and the overall interplay of building blocks, there are several strategies on how to approach motif design successfully. We prefer constructing a motif in a top-down fashion, dividing it into the different motif parts connected by single strands on the top level (if present). Each motif part is then built individually by placing the appropriate building blocks next to each other. Another good way to go about is to build the structure iteratively from 5' to 3' end with gaps for any potential single strand connections. A third strategy is to identify the most prominent motif areas, such as a large multiloop e.g., and commence the motif definition in those areas. When several related motifs are to be designed, parts described first may be stored and re-used. Only at the very end, single strands are used to connect the different motif parts or to extend the 5' or 3' end.

4 Obtaining Motif Code

Since Locomotif is based on a step-wise translation from visual motif design to an executable search program, several levels of code are involved in this process. You are able to access all levels of code from the Locomotif editor or via a download site. First, the graphics is translated to XML, then further to ADP and finally to C sources with compilation to a C executable. Additionally, at the bottom of the editor window, you can also view a Shape representation of the motif which is an abbreviated form of the dot-bracket strings commonly used for representing RNA secondary structures. And, you can access the motif information via the menu "File→Export" which opens a frame displaying all annotations made for the individual building blocks in the order from 5' to 3' end. Please recall though, that you do not need to read or use any of these motif code forms. You can simply design the motif, send the information to the server as described in the next section and run the program without worrying about language codes.

XML The XML level is mainly used as a means of transportation from the client editor to the server where all further translations take place. It contains all relevant biological information, but completely abstracts from the graphical layout. Therefore, it is not possible to reconstruct the graphics from the XML file. Yet, you can also store a ".rna" project within the editor which contains all the information necessary to load a visual project within the editor.

To obtain the XML file, you need to go to the menu "File", then choose "Export" and further choose "XML". There is little you can do with the XML file unless you were to write a script that directly accesses the webservice functions of the BiBiServ which allow you to communicate with the server. Then, you could edit the XML file and send the new version to the server for translation. Yet, it must pass the requirements of the XML schema and it is probably much easier to simply edit the motif via the Locomotif editor.

Potentially, you could use the XML code generated via Locomotif as a basis for the translation to other languages describing RNA motif search grammars (e.g. RNAMotif). Currently, we do not work on any such translations, but you are welcome to do so, if you wish.

ADP You can obtain the ADP code the same way as the XML code, using the menu "File→Export→ADP". This code is produced via local translations which are based on the same Java functions used on our server. After saving it, you are free to edit the ADP code (for references on writing ADP programs, please visit http://bibiserv.techfak.uni-bielefeld.de/adp/). Using the ADP compiler you can produce your own C-program based on the edited ADP code. This gives you a lot of freedom as long as you have some functional programming skills. Using your own ADP compiler renders you completely independent from our server if you desire so. In contrast to the C code, the ADP translations are based on very readable code that is easy to edit if you have the necessary programming skills.

C You cannot access the C code directly from the Locomotif editor as it is produced only on the server side by the ADP compiler. Yet, you can obtain

the code from the server using the ID you get once sending the XML code to the server (please refer to the next sections for details on the client-server communication protocols). The downloaded C sources are hard to read, since variables are not named in a sensible way, but rather enumerated. Nevertheless, the ADP code is contained as comments within the C sources. This way, you can easily find a certain part of the motif grammer within the C code by searching for the corresponding ADP program line. Anyone with sufficient C programming skills should be able to alter the C code if desired. Overall, we believe it to be easier to write code on the ADP level instead of the C level, but the choice remains yours in the end.

5 Generating a Motif Matcher

The generation of an executable search program for the motif you designed is an automatic process that requires no further input from you. The prerequisite for matcher generation is a finished motif design, i.e. a structure that has only the 5' and the 3' as remaining open ends. Either in a double-stranded exit of a building block or in one or two single strands extending the structure.

Motifhead Then, you need to access the socalled motifhead which is indicated by a red circle around the 5' end. Clicking on the circle opens an interface where you must choose a name for the project and you must choose the search type. You can either do a local search in which case the best occurence of the motif within a (potentially longer) sequence is found. Or you can do a global search in which case the entire given sequence is folded according to the motif definition. Optionally, you can specify an overall minimum and/or maximum size for the motif in case of the local search (it makes no sense to restrict the size for the global search as the entire sequence is folded anyway). Actually, it is very useful to give a maximum size whenever the expected motif is smaller than the given target sequence. Restricting the size restricts the foldings to the size we expect instead of aiming to fold much longer parts of the sequence to find the optimum.

Send Information Once a project name is given and the search type is chosen, you can send the necessary XML information to the server by using the menu "Translation" and "Send XML". You will be asked if you want to specifiy an email address in which case you will be informed about successful matcher generation (or any problems) via email. In the email you will receive an ID referring to your matcher and also a link to the online submission and download sites as described in the next section. This relieves you from any need to further use the Locomotif editor. Alternatively, you can also leave the space blank and wait for the matcher generation results. Matcher compilation usually takes only some seconds. In case of very large motifs it can go up to 1 or 2 minutes. Any longer time is usually an indication of an error (i.e. the compiler going into an endless loop) which should not occur. You will be informed upon successful compilation in a small window which also contains the Bibiserv ID referring to the matcher. You can choose to store the ID in a file or you can also copy it by highlighting it with the mouse cursor (and then paste it wherever you need it). It can be useful to store the ID, if you plan to use the matcher for a longer time period. Otherwise, you will loose the ID as soon as you close the Locomotif

editor. The ID is used to access your matcher on our server. If you provided an email address, the ID will also be sent to you in an email together with links to the online submission and download sites.

6 Running a matcher

Once you obtained a matcher ID, you are ready to use the generated search program. Basically, there are 3 ways how to work with the matcher. You can start a search either directly from the Locomotif editor or from the online submission site. Or, you can download the matcher source code (C) in order to compile and run it on your own system.

6.1 via Locomotif

The quickest way is to use the Locomotif editor for a search. This is recommended while still working on improving your matcher by testing it on rather short target sequences. You must use the menu "Use Matcher" and then choose "Run Now". A new window will open where you can select the matcher ID in a drop down menu. All IDs you received during the current use of the Locomotif editor will be available. Alternatively, you can also upload an ID from a file. Sequences can be provided directly in plain or fasta format or the can be uploaded from file. You cannot use multiple plain sequences, but multiple fasta files are fine. Again, you can provide an email adress so that you will be notified once the search terminates. You will receive a link to a webpage where you can access the results. If you do not give an email adress, you must leave the Locomotif editor window open until the search phase is finished. Otherwise, there is no way to still access the results. Therefore, you should either use an email adress or restrict searches to short target sequences as searches on longer sequences may take some time. (Please expect a runtime of $O(n^3)$ (or $O(n^4)$ if using pseudoknots) even though smaller bounds are achievable depending on the motif definition and/or the use of the window functionality via the Download option).

6.2 via Webpage

You can also search independently from the Locomotif editor via the submission site at http://bibiserv.techfak.uni-bielefeld.de/locomotif/submission.html. There you can also provide sequences directly or from file and you must input a Locomotif Matcher ID. You should copy the ID either in a file or directly within the Locomotif editor and paste it in the appropriate field. You can also open the webpage directly from the Locomotif editor using the menu "Use Matcher→Submit" and pressing "OK". The results will be presented in the web browser which must remain open until finished or again, you can give an email address and receive a link to the result site via mail.

6.3 via Download

Finally, you can also run the matcher on your own system by downloading all the required sources. The download site is located at http://bibiserv.techfak.uni-bielefeld.de/locomotif/download.html (also accessible via the menu "Use Matcher

→Download"). You must give a Locomotif Matcher ID and press "Download". A webpage will open where you can store the C file containing your motif matcher. You are also provided with a tar file containing the necessary libraries. In a common Unix or Linux environment following the information on the webpage should result in successful compilation of an executable matcher program (untarring the tar file, copying and changing to the required directory and running make). Then, you can run the matcher by invoking the program "Tdm". It offers an interactive command line interface; view all options via "?". Similar to the searches on the server, you must provide sequences in plain or fasta format and you can also use a few additional parameters. For instance, instead of just calculating the best hit, you can obtain suboptimal hits like the best 10% or all hits within a certain energy range from the optimum. You can use a window function to scan very large sequences.

7 Result presentation

Results are presented in form of Dot-Bracket strings where any unpaired base is shown as a dot and a basepair is shown by an opening and corresponding closing bracket. You will see the target sequence and below the dot-bracket string usually starting and ending with a large number of dots. The first bracket indicates the start of the motif (unless as single strand extends the 5' end) and the last bracket indicates the end (unless 3' end is extended). You also get a minimum free energy value for the hit. In case of a multiple fasta file, a result is given for each sequence in the file. If no hit can be found, you will receive an empty result file.

RNAMovies In the newest version of Locomotif, we also offer output visualization via the program RNAMovies[3] developed by Robert Giegerich and Dirk Evers. For each target sequence, you can open an RNAMovies window, via the button "Visualize" and a drop-down selection, which displays all the results found for that sequence. In case of several results, you can skip through them using "Previous" or "Next", and you can also watch them in form of an animated movie. (The program is intended to visualize RNA secondary structure spaces. For displaying independent hits of a motif within a target sequence, the animation is not particularly useful, but still available.) Figure 4 shows an RNAMovies visualization of a search for oxyS executed with the Locomotif system. Several image output formats can be generated from the RNAMovies visualization including publication quality ones. If using the webpage to submit sequences, a direct upload to RNAMovies is not possible. Yet, you can still use the program based on the Locomotif output which has to be transformed slightly to the input files required by RNAMovies. For more information, please refer to the manual of RNAMovies[3].

RNAplot For another convenient display of the dot-bracket strings, we recommend using the program rnaplot from the Vienna package[4]. You can simply save the Locomotif output (fasta header, sequence and dot-bracket string) or

[3] http://bibiserv.techfak.uni-bielefeld.de/rnamovies
[4] http://www.tbi.univie.ac.at/RNA/

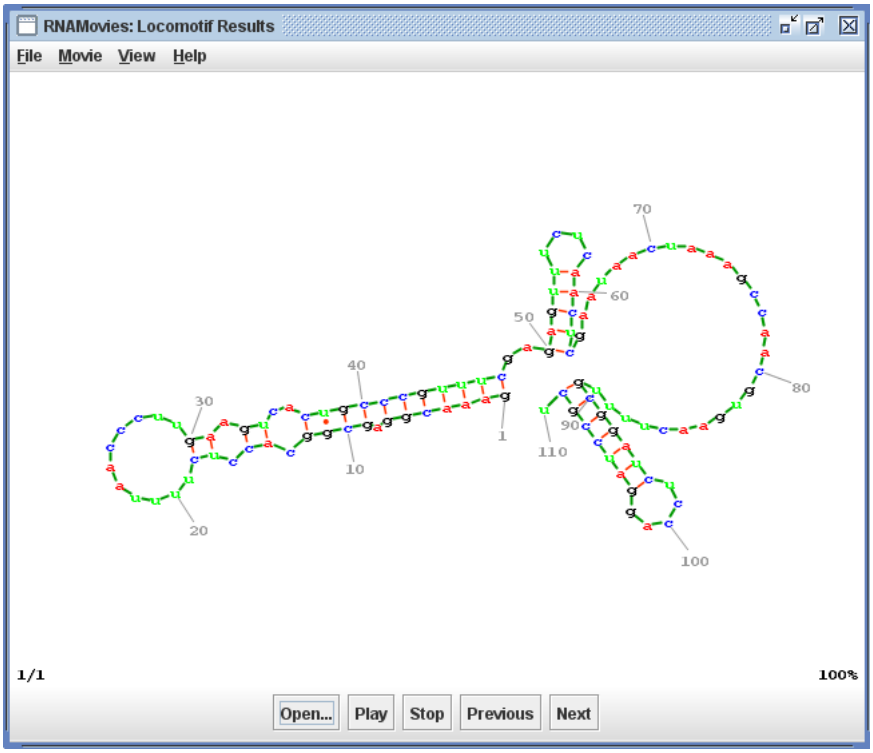

Figure 4: Screenshot of the integrated visualization done with RNAMovies.

copy and paste it, and as a result, you will receive an actual 2d structure of the hit as a .ps file.