Web Service for Finding Ribosomal Frameshifts*

Yanga Byun, Sanghoon Moon, and Kyungsook Han**

School of Computer Science and Engineering, Inha University, Inchon 402-751, Korea
quska@inhaian.net, jiap@inhaian.net, khan@inha.ac.kr

Abstract. Recent advances in biomedical research have generated a huge amount of data and software to deal with the data. Many biomedical systems use heterogeneous data structures and incompatible software components, so integration of software components into a system can be hindered by incompatibilities between the components of the system. This paper presents an XML web service and web application program for predicting ribosomal frameshifts from genomic sequences. Experimental results show that the web service of the program is useful for resolving many problems with incompatible software components as well as for predicting frameshifts of diverse types. The web service is available at http://wilab.inha.ac.kr/fsfinder2.

1 Introduction

A large-scale bioinformatics system often consists of several application programs dealing with a large volume of raw or derived data. For example, a data analysis program generates new data that may be modeled and integrated by other programs. However, the programs may not be interoperable due to the differences in data formats or running platforms. As a result, developing a bioinformatics system with a few application programs requires extra work to make the components compatible. In fact the difficulty in bioinformatics study comes more from the heterogeneity of the programs and data than from the quantity of the data.

Web service resolves some of these problems by exchanging messages between different applications developed by various programming languages. The basic process of web service is exchanging Simple Object Access Protocol (SOAP) messages described by XML (eXtensible Markup Language). When the server of web services receives the SOAP request with the parameters for calling the method of web service, the server returns the SOAP message in response to the method. An application program can use web services developed in different languages, whose results in turn can be a request message for another web service.

Recently various bioinformatics systems supporting web services have been developed [1]. European Bioinformatics Institute (EBI) provides several web services such as Dbfetch for biological database, ClustalW for multiple alignments of DNA and protein sequences, Fasta for nucleotide comparison [2]. Databases of DDBJ [3] and KEGG [4] give web service access. A tool named Taverna was developed for the bioinformatics workflows with several relevant biological web services [5].

* This work was supported by the Korea Science and Engineering Foundation (KOSEF) under grant R01-2003-000-10461-0.

** Corresponding author.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part I, LNCS 3991, pp. 284 – 291, 2006.

In previous work we developed a program called FSFinder (Frameshift Signal Finder) for predicting frameshifting [6]. Frameshifting is a change in reading frames by one or more nucleotides at a specific mRNA signal [7]. Frameshifts are classified into different types depending on the number of nucleotides shifted and the shifting direction. The most common type is a -1 frameshift, in which the ribosome slips a single nucleotide in the upstream direction. +1 frameshifts are much less common than -1 frameshifts, but have been observed in diverse organisms [8]. FSFinder is written in Microsoft C# and is executable on Windows systems only. To remove these limitations and to handle frameshifts of general type, we developed a web service and web application called FSFinder2. Users can predict frameshift sites of any type online from any web browser and operating system. By providing web service, FSFinder2 is more usable and compatible than the previous version of the program.

2 Systems and Method

Our system gives three ways to access: standalone application, web application and web service. The standalone application runs on Windows system only. The web application and service can be used with any web browser on any system. The web service is different from other web services in which the client simply makes a request based on the input format of the server and watches the web page returned in response to the request. On the other hand, the web service of our system is not only available through the web page but also enables to exchange soap messages in the XML format. This means that the user can use a web service without using the web page when he knows the input XML schema, output XML schema and the address of the web service. The user can also modify or reuse the reply message in the XML format to make it suitable to his system. The rest of this section describes the messages supported by the web service.

2.1 The Request SOAP Message

The ‘FSFinder_run’ message is the request message that includes the information of the input sequence and one or more user-defined models. Fig. 1A shows the elements of the request message. The ‘components’ element, which is a subelement of ‘FSFinder_Model’ can include any combination of four components: pattern type, secondary structure type, signal type and counter type [6]. The attribute ‘spacer’ is the space from the previous components. This attribute is needed from the second component. One number means fixed spacer and the range of numbers by ‘nuber1~number2’ means flexible spacer in the range.

Nucleotide characters (A, G, C, T or U, R, Y, M, K, S, W, H, B, V, D and N) and comma (,) are used to represent nucleotides. Comma marks the position of codons. It is optional and no comma represents any codon. The pattern component represents a pattern of nucleotides like the slippery site of the -1 frameshift model. Pattern type requires pattern characters, the match and exception nucleotide characters that represent the pattern characters. Repeated pattern characters are represented once. It finds the match sequence pattern with match characters without exception characters. The secondary structure component defines the size of pseudoknot parts in Fig. 2. The signal type can include any number of nucleotide signals. The counter type defines a count and the counter equation.

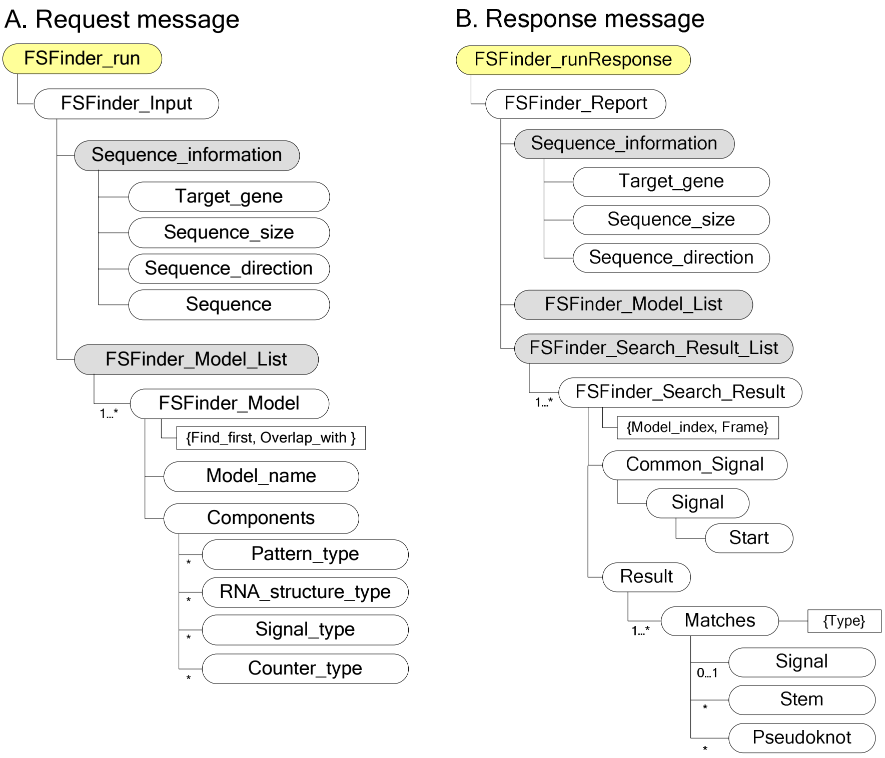


Fig. 1. The request message (left) and the response message (right) of the FSFinder2 web service. (A) The root element of the request message is 'FSFinder_run'. It requires one 'FSFinder_Input' element. 'Sequenece_information' and 'FSFinder_Model_List' are required as subelements of 'FSFinder_input'. 'Sequence_information' represents the input sequence. 'FSFinder_Model_List' includes one or more user-defined models. The element for the user-defined model is named 'FSFinder_Model'. This element requires overlapping open reading frames and the position of the component to be found first. (B) The root element of the response message is 'FSFinder_runResponse' and it has one child element named 'FSFinder_Report'. The child elements of 'FSFinder_Report' are 'Sequence_information', 'FSFinder_Model_list' and 'FSFinder_Search_Result_List'. 'Sequence_information' in the response message does not include the input sequence.

2.2 The Response SOAP Message

The response message includes the search results and the user-defined models specified in the request message. The structure of the response message is shown in Fig. 1B. The 'FSFinder_Search_Result_List' element has one or more 'FSFinder_Search_Result' elements. 'Model_index' and 'Frame' are the attributes of the element. 'Model_index' shows the index of the model from which the result came. 'Frame' is the frame where the search started. The search result includes one 'Common_Signal' and one 'Result' elements. The 'Result' element has 'Matches' elements in the same number and order as

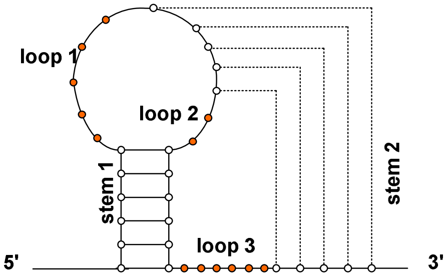


Fig. 2. Five parts of a pseudoknot structure. The size of each part can be defined by the subelements of 'RNA_Structure_type'.

specified in the components of the model. The 'Matches' element includes one signal or any number of stems and pseudoknots. 'Common_Signal' is the sequence fragment that has more than one match.

2.3 Web Application

Web application helps the user use the web service easily with a web browser (Fig. 3). Since the web application server sends an HTML document only in response to user request, all the user needs is a web browser that can read an HTML document. The web application server handles these requests with active server page (ASP). An ASP document generates an XML document and this XML document is shown as an HTML page with the current XSLT (the Extensible Stylesheet Language

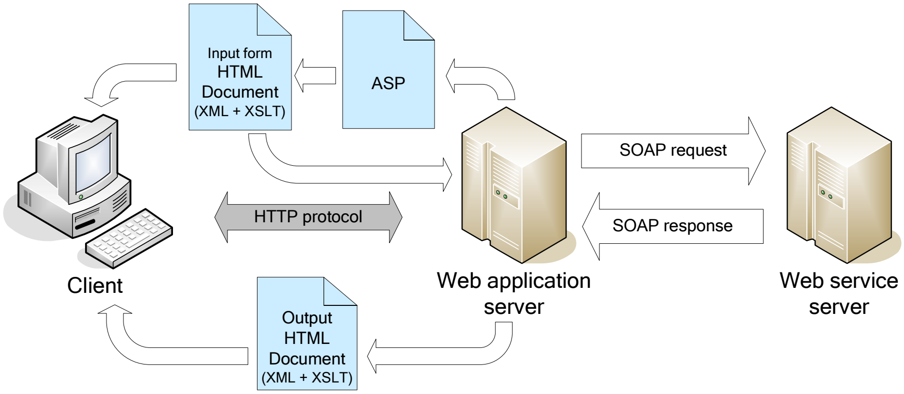


Fig. 3. The processes involved in the web application. The client connects to the web application server with HTML document using HTTP protocol. The web application server makes the request SOAP message and sends it. When the web service server sends back the result of the request, the web application server makes an XML document for the response SOAP message and returns the XML document in the current style sheet.

Transformations; http://www.w3.org/2002/ws/) style sheet. When the user finishes editing the request for the web service with previous process, the web application server sends the request to the web service server. The web service server calls 'FSFinder_run' method with the request, returns the result and sends back the SOAP response. Similarly, the ASP document makes an XML document and sends the output as the style of XSLT document for an output.

3 Implementations

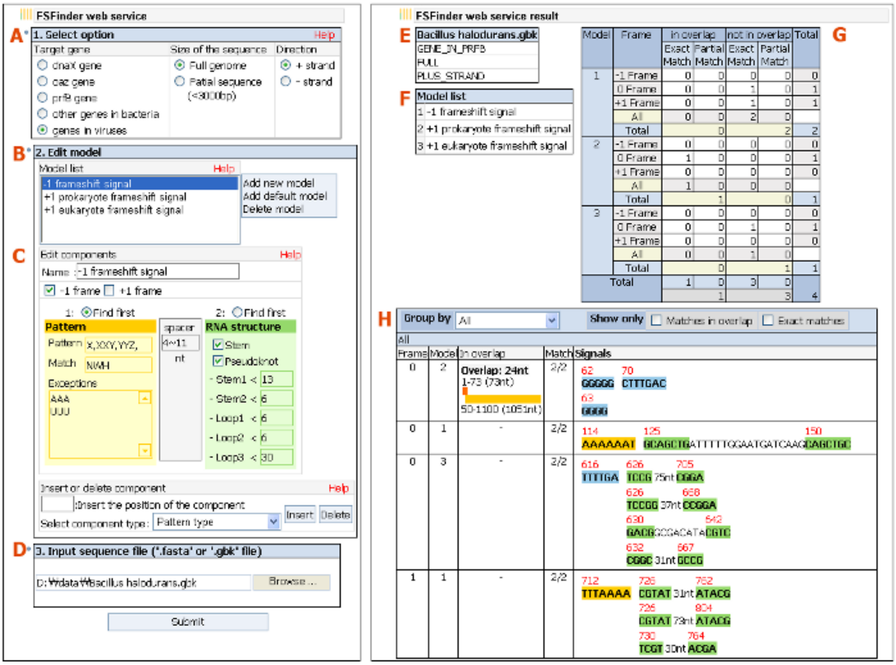
FSFinder2 was implemented using XML, XSLT, ASP and JavaScript. If the user sends a query to the server after defining a new model, the computation is performed on the server side and the results are sent back to the user. Consequently FSFinder2 is independent of the operating systems and web browsers of his/her computer.

The XML schema of FSFinder2 described in the previous section can be downloaded at the web site. In the 'FSFinder in web' page, we provide the developer's guide containing information such as Web Services Description Language (WSDL), SOAP message and XML schema. Web service can be used only with the WSDL explanations. The results from any tools (applications, databases and so on) can be the input of this web service with making XML message as the form needed by this system. The output can also be input of any tools. Therefore, FSFinder2 guarantees high reusability and solves the problems from the differences of the implementation languages.

Three types of frameshift are considered as basic frameshifts, and their models are predefined: the most common -1 frameshifts, +1 frameshifts of the RF2 type, and +1 frameshift of the type found in the ornithine decarboxylase antizyme (ODC antizyme). The models for these frameshifts consist of a Shine-Dalgarno sequence, frameshift site, spacer and downstream secondary structure.

FSFinder2 extends the three basic models used in FSFinder to incorporate user-defined models. For the upstream Shine-Dalgarno sequence, FSFinder2 considers AGGA, GGGG, GGAG, AGGG and GGGA as well as classical Shine-Dalgarno sequences such as GGAGG and GGGGG. For the slippery site of the +1 frameshift, the sequence CUU URA C, where R is a purine (that is, either adenine or guanine), is considered, and for the downstream structure, H-type pseudoknots as well as stem-loops are considered. Users can define not only the basic models but also their own models with a combination of the four types of components witch was defined in our previous work [6]: Pattern type, Secondary structure, Signal type and Counter type.

Predicting frameshift sites using FSFinder2 consists of 3 steps: select option, edit model and input sequence file (Fig. 4A-D). In the select option, the user specifies the gene type, and its sequence and direction (Fig. 4A). In the select option, the user specifies the target gene type, whether the input DNA sequence is a complete genome or partial sequence, and its sense (+ or -). For the target gene type, the user can choose one of dnaX gene, oaz gene, prfB gene, other genes in bacteria and genes in viruses. Because it requires different methods for fining overlapping regions of ORFs, bacterial needs to be distinguished form viral genes.



FSFinder web service

1. Select option

Target gene	Size of the sequence	Direction
○ dnaX gene	● Full genome	● + strand
○ caz gene	○ Patial sequence (<3000bp)	○ - strand
○ prfB gene		
○ other genes in bacteria		
● genes in viruses		

2. Edit model

Model list

-1 frameshift signal
+1 prokaryote frameshift signal
+1 eukaryote frameshift signal

Add new model
Add default model
Delete model

Edit components

Name : -1 frameshift signal

☑ -1 frame ☐ +1 frame

1: ● Find first

Pattern
Pattern X,XXY,YYZ,
Match NWH
Exceptions
AAA
UUU

spacer
4~11
nt

2: ○ Find first

RNA structure
☑ Stem
☑ Pseudoknot
- Stem1 < 13
- Stem2 < 6
- Loop1 < 6
- Loop2 < 6
- Loop3 < 30

Insert or delete component

:Insert the position of the component

Select component type : Pattern type

3. Input sequence file ('.fasta' or '.gbk' file)

D:₩data₩Bacillus halodurans.gbk

FSFinder web service result

Bacillus halodurans.gbk
GENE_IN_PRFB
FULL
PLUS_STRAND

Model list	
1	-1 frameshift signal
2	+1 prokaryote frameshift signal
3	+1 eukaryote frameshift signal

Model	Frame	in overlap		not in overlap		Total
		Exact Match	Partial Match	Exact Match	Partial Match	
1	-1 Frame	0	0	0	0	0
	0 Frame	0	0	1	0	1
	+1 Frame	0	0	1	0	1
	All	0	0	2	0	
	Total	0		2		2
2	-1 Frame	0	0	0	0	0
	0 Frame	1	0	0	0	1
	+1 Frame	0	0	0	0	0
	All	1	0	0	0	
	Total	1		0		1
3	-1 Frame	0	0	0	0	0
	0 Frame	0	0	1	0	1
	+1 Frame	0	0	0	0	0
	All	0	0	1	0	
	Total	0		1		1
Total		1	0	3	0	
		1		3		4

Group by All ▾ Show only ☐ Matches in overlap ☐ Exact matches

All

Frame	Model	In overlap	Match	Signals
0	2	Overlap: 24nt 1-73 (73nt) 50-1100 (1051nt)	2/2	62 70 GGGGG CTTTGAC 63 GGGG
0	1	-	2/2	114 125 150 AAAAAAT GCAGCTGATTTTTGGAATGATCAAGCAGCTGC
0	3	-	2/2	616 626 705 TTTTGA TCCG 75nt CGGA 626 668 TCCGG 37nt CCGGA 630 642 GACGGCGACATACGTC 632 667 CGGC 31nt GCCG
1	1	-	2/2	712 726 762 TTTAAAA CGTAT 31nt ATACG 726 804 CGTAT 73nt ATACG 730 764 TCGT 30nt ACGA

Fig. 4. The input page (left) and result page (right) of FSFinder2. Left: (A) The select option lets the user choose the type of genes expressed via frameshifts, the size of the sequence and its direction. (B, C) The user can define a new model by specifying its components and their locations. (D) The user selects the input sequence file. Right: (E) This box gives the file name of the input sequence, target gene, sequence size and direction. (F) This shows potential frameshift sites found by FSFinder2 for each model in the model list. (G) The results are separated into exact matches and partial matches in each of the overlapping and non-overlapping regions. (H) The results are grouped into model types, frames containing the frameshift sites and the overlapping regions of ORFs. FSFinder2 displays the locations and lengths of the overlapping ORFs. The match column shows the number of matched components out of the total components in the model. The signal column presents matched components and sequences. The matched parts in the signal are highlighted in the color defined by the user. Yellow, green, sky blue and red designate pattern type, RNA structure, signal type and counter type, respectively. The red numbers above the sequence designate the position of the component with respect to the first nucleotide of the sequence.

In the edit model, the user can define a new model of frameshift (Fig. 4B). After choosing 'Add new model' in the model list, the user defines the components of the new model (Fig. 4C). The check button of the -1 and +1 frames determines the type of frameshift. -1 frame is for finding -1 frameshift and +1 frame is for +1 frameshift. The component selected as 'find first' is searched earlier than any other component of the model. Below the 'find first' button, there are parameters of the component that the user can specify. After choosing the first component to search for, user can add a new component to the model. Each component can be changed before running the program, and spacers of arbitrary length are inserted between the components. As the last step, users choose the input sequence file (Fig. 4D). Two kinds of file formats can be handled by FSFinder2: Genbank file format and Fasta file format.

The right part of Fig. 4 gives an example of output. Information such as file name, size of sequence, direction and models are shown in Fig. 4E-F. Candidate frameshift sites are found based on the models in the model list (Fig. 4G). The exact matches indicate the number of frameshift sites for which all components conform to the defined model in the overlapping regions of ORFs, and the partial matches indicate the number of frameshift sites for which not all components of the model conform to the model. 0, +1 and -1 frames indicate that FSFinder2 finds genes from the first nucleotide, second nucleotide and third nucleotide of the input sequence, respectively.

Results are grouped into frames, model types, and overlapping and non-overlapping regions of ORFs (Fig. 4H). Users also have the option to see only the matches in the overlapping regions and correct matches. In the overlap column, FSFinder2 displays the overlapping genes and their lengths, which is quite useful when examining the frameshift sites identified. For instance, the prfB gene of E. coli, which encodes release factor 2 (RF2), always consists of a short ORF overlapped with a long ORF. Frameshift sites not conforming to this pattern can be filtered out.

4 Conclusion

We developed FSFinder2 to predict frameshift sites with a user-defined model. The web service and web application of FSFinder2 were implemented using XML, XSLT and JavaScript. If the user sends a query to the FSFinder2 server after setting parameters or defining a new model, all the computations are done on the server side. After computation, the server sends the results to the user. Consequently FSFinder2 is independent of the operating system or web browser of his/her computer. The web service provided by FSFinder2 is slightly different from most existing web applications in which a client makes a request based on the input format and watches the web page for a reply via the web browser. The FSFinder2 server and client exchange SOAP messages in the XML format according to the properties of the web service. If the user knows the input XML schema, output XML schema and address of the web service, the user can use the web service without using the web page. Since the reply message is sent in the XML format, the user can modify it to suit his/her system.

Experimental results of testing FSFinder2 on ~190 genomic and partial DNA sequences showed that it predicted frameshift sites efficiently and with greater sensitivity and specificity than other programs, because it focused on the overlapping regions of ORFs and prioritized candidate signals (For -1 frameshifts, sensitivity was 0.88 and specificity 0.97; for +1 frameshifts, sensitivity was 0.91 and specificity 0.94) [6, 9-11]. FSFinder2 has been successfully used to find unknown frameshift sites in the Shewanella genome. We believe FSFinder2 is the first program, guaranteed high reusability, capable of predicting frameshift signals of general type and that it is very useful for analyzing programmed frameshift signals in complete genome sequences. The web service of the FSFinder would be useful as not only the function of FSFinder2 itself but also intermediate of the cooperation with the other web services.

References

1. Stein, L.: Creating a bioinformatics nation. Nature, 417 (2002), 119–120
2. Pillai, S., Silventoinen,V., Kallio, K., Senger, M., Sobhany, S., Tate, J., Velankar, S., Golovin, A., Henrick, K., Rice, P., Stoehr, P., Lopez, R.: SOAP-based services provided by the European Bioinformatics Institute. Nucleic Acids Research, 33 (2005), 25-28
3. Miyazaki, S., Sugawara, H., Gojobori, T., Tateno, Y.: DNA Data Bank of Japan (DDBJ) in XML. Nucleic Acids Research, 31 (2003), 13-16
4. Kawashima, S., Katayama, T., Sato, Y. Kanehisa, M.: KEGG API: A Web Service Using SOAP /WSDL to Access the KEGG System. Genome Informatics, 14 (2003), 673-674
5. Oinn, T., Addis, M,. Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics, 20 (2004), 3045-3054
6. Sanghoon, M., Yanga, B., Hong-Jin, K., Sunjoo, J., Kyungsook, H.: Predicting genes expressed via -1 and +1 frameshifts. Nucleic Acids Research, 32 (2004) 4884-4892
7. Baranov, P.V., Gesteland, R.F., Atkins, J.F.: Recoding: translational bifurca-tions in gene expression. Gene, 286 (2002), 187-201
8. Farabaugh, P.J.: Programmed translational frameshifting. Ann. Rev. Genetics, 30 (1996), 507-528
9. Hammell, A.B., Taylor, R.C., Peltz, S.W., Dinman, J.D.: Identification of putative programmed -1 ribosomal frameshift signals in large DNA databases. Genome Research, 9 (1999), 417-427
10. Bekaert, M., Bidou, L., Denise, A., Duchateau-Nguyen, G., Forest, J., Froidevaux, C., Hatin, Rousset, J., Termier, M.: Towards a computational model for -1 eukaryotic frameshifting sites. Bioinformatics, 19 (2003), 327-335
11. Shah, A.A. Giddings, M.C., Parvaz, J.B., Gesteland, R.F., Atkins, J.F., Ivanov, I.P.: Computational identification of putative programmed translational frameshift sites. Bioinformatics, 18 (2002), 1046-1053