

FramePlus: A sensitive algorithm for aligning DNA to protein sequences

Eran Halperin, Simchon Faigler and Raveh Gill-More*

Compugen Ltd., 72 Pinchas Rosen Street, Tel Aviv, Israel 69512

Abstract

Motivation: Automated annotation of ESTs is becoming increasingly important as EST databases continue to grow rapidly. A common approach to annotation is to align the gene fragments against well-documented databases of protein sequences. The sensitivity of the alignment algorithm is key to the success of such methods.

Results: This paper introduces a new algorithm, *FramePlus*, for DNA-protein sequence alignment. The *SCOP* database was used to develop a general framework for testing the sensitivity of such alignment algorithms when searching large databases. Using this framework, the performance of *FramePlus* was found to be somewhat better than other algorithms in the presence of moderate and high rates of frameshift errors, and comparable to *TranslatedSearch* in the absence of sequencing errors.

Availability: The source code for *FramePlus* and the testing datasets are freely available at <ftp.compugen.co.il/pub/research>.

Contact: raveh@compugen.co.il

Keywords: dynamic programming, sequence alignment, algorithm benchmarking

1 Introduction

A database search can potentially identify the function of a previously uncharacterized biological sequence and raise its importance as a possible drug target or research candidate. As a result, sequence alignment algorithms are among the most important annotation tools available. The sensitivity of a search algorithm, however, can have a crucial effect on the quality of the annotation; different al-

gorithms will find (and miss) different potential homologues under different circumstances.

A special case of sequence alignment algorithms are those that compare nucleic acid sequences to protein sequences; such algorithms will be referred to as *frame* algorithms. A frame algorithm aligns the nucleic acid sequence to the protein sequence after translating the former using the genetic code, possibly taking account of insertions, deletions, and substitutions.

Frame algorithms are particularly useful for annotating Expressed Sequence Tags (ESTs). ESTs are fragments of messenger RNA, which are typically sequenced in a single-pass, possibly low-quality, manner [Adams et al., 1993]. Huge databases of ESTs exist [Boguski et al., 1993]. Comparing ESTs to high-quality protein databases such as SwissProt [Bairoch and Boeckmann, 1994] is one of the most promising approaches for discovering novel genes, although this is a non-trivial process [Altschul et al., 1994, Burke et al., 1998].

The first frame algorithm developed was *Translated Search* [Peltola et al., 1986]. This program uses the approach known as *six-frame translation*, and is based directly on the Smith-Waterman algorithm [Smith and Waterman, 1981]. In recent years, heuristic database search packages have incorporated frame algorithms as well. Well-known heuristic algorithms include *BlastX* [Altschul et al., 1990] and *FastX* [Pearson et al., 1997]. Other related directions of research include [Hein, 1994, Pederson et al., 1998, Birney et al., 1996].

Early frame algorithms were not very tolerant of errors, especially frame-shift errors (insertions and deletions). More recent frame algorithms, such as *FrameSearch* [Edelman et al., 1995], *FastY* [Zhang et al., 1997], *LAP* [Huang and Zhang, 1996], and *Grail* [Guan and Uberbacher, 1996], explicitly model

*To whom correspondence should be addressed

transitions between frames in order to recognize and handle frameshift errors. Some of these algorithms also model introns and other biological phenomena. Generalizing *FramePlus* to accommodate introns in a similar way is straightforward. Introns are not a major problem for EST annotation, so such extensions are not considered in the present work.

This paper introduces a new frame algorithm called *FramePlus*. *FramePlus* is an extension of *FrameSearch* in which differences resulting from sequencing errors are modeled separately from differences resulting from evolution. Modeling these different biological phenomena separately should improve the sensitivity of database searches.

In the present work, this conjecture is tested using an adaptation of the method of [Brenner et al., 1998]. The method compares the sensitivity of search algorithms using the *SCOP* database of structurally classified proteins [Murzin et al., 1995]. This provides a general framework for benchmarking, which was used to compare *FramePlus* and several other frame algorithms.

The results suggest that under realistic conditions, *FramePlus* is significantly more sensitive than all the other tested algorithms. In cases of low sequence identity, *FramePlus* finds as many as 13% more true positives than any other tested algorithm.

The tradeoff is speed; on a general purpose computer, *FramePlus* is much slower than heuristic algorithms such as *BlastX* and *TFastY*. When compared to other dynamic-programming frame algorithms, *FramePlus* takes 50% longer than *FrameSearch* and twice as long as *Translated Search*. However, on special purpose hardware, dynamic programming algorithms can be accelerated by up to 3 orders of magnitude, which makes their performance competitive with heuristic algorithms run in software. *FramePlus* and *FrameSearch* have been implemented this way.

2 Materials and Methods

This section begins with a description of the model that allows *FramePlus* to achieve such sensitive database search results. This is followed by an explanation of the testing procedure, including search parameters, the test dataset, and benchmarking criteria.

2.1 The Model

Figure 1 shows the *Frameplus* model for comparing sequences. The model includes four states and 13 transitions, with actions occurring on the transitions. Each transition is labeled by a pair of numbers; the first is the number of nucleotides emitted by the transition, the second is the corresponding number for amino acids. Each transition yields a score, possibly negative, that results from the alignment or indel added by the transition. Every alignment between a DNA and a protein sequence corresponds to a path through the model. The score of the alignment is the sum of the scores of the transitions along the alignment's path.

The most extensively used transition is the one from Match to Match. It emits 3 DNA bases and one amino acid and adds to the score a contribution based on the compatibility of the codon and the amino acid. A deletion of a DNA base is modeled by moving from Match to Delete and then back using the transition labeled (2,1). Similarly, an insertion of a DNA base is modeled by moving from Match to Insert and back.

The Insert-3 state is the key difference between the *FramePlus* and *FrameSearch* models. This state models insertions of codons in the DNA sequence, rather than insertions of individual nucleotides. These are two different phenomena, the former caused by evolution, the latter caused by sequencing errors. Adding this state allows the different kinds of insertions to be scored differently. For example, one might penalize an insertion of three bases less severely than an insertion of two bases, since the first phenomenon may be explained as a single evolutionary event, but the second must have been caused by two events (either two nucleotide insertions or a codon insertion and a nucleotide deletion).

2.2 Parameters

Transition scores are supplied as parameters. A transition out of the Match state represents alignment of a codon and an amino acid. These scores can be positive or negative, depending on the specific nucleotides and amino acid; the BLOSUM62 matrix [Henikoff and Henikoff, 1992] is used by default.

All other transitions contribute negative scores that are independent of the data. Opening a gap

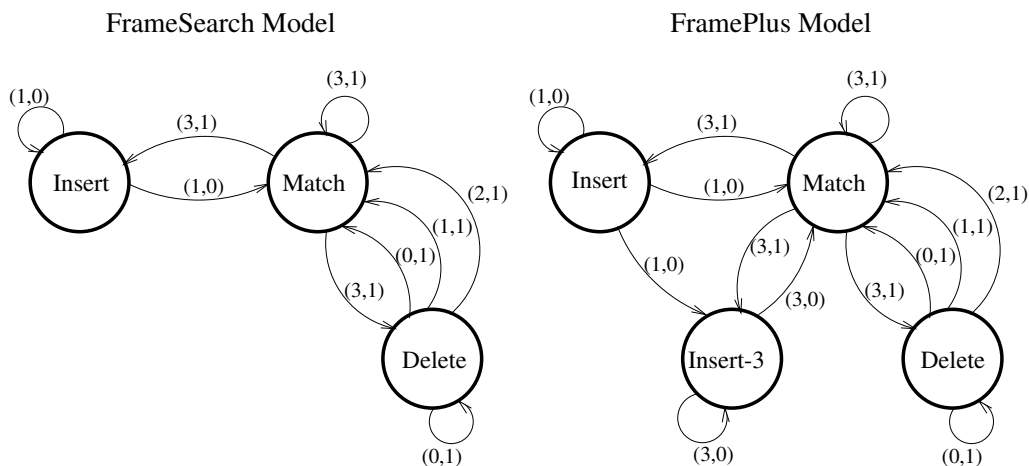


Figure 1: FrameSearch and FramePlus models. The models can be thought of as finite state automata which emit x nucleotides and y amino acids each time a transition labeled (x, y) is used. Transitions scores are model parameters.

is typically more expensive than extending it. The default gap open parameter for transitions out of the Insert-3 and Delete states is 10.0. For transitions out of the Insert state the default is 7.0. The default gap extension parameter for transitions out of the Insert-3 and Delete states is 0.5 and for transitions out of the Insert state it is 6.0.

In these tests, all algorithms were run with their default parameters. E-values were used as scores, except that P-values were used for *BlastX*.

2.3 The *SCOP* Database

The more realistic model used by *FramePlus* should, at least in theory, make it a more sensitive search tool than *FrameSearch*. In order to see if practice follows theory, the ability of *FramePlus* and several other frame algorithms to identify distant homologs was tested.

The protein domains in *SCOP* [Murzin et al., 1995] are arranged hierarchically according to information about their structure and function. Each domain is annotated with information about its class, fold, super-family and family. For purposes of the testing performed in this work, two proteins are considered homologous if their annotation is identical in all four fields.¹

¹Testing using just the first three fields, i.e., using super-family instead of family, yields results that are remarkably similar (data not shown).

The dataset used for testing is a subset of *SCOP* containing only domains which are not closely related to each other; no two sequences have a significant alignment with more than 40% identity. This dataset can be found at http://scop.mrc-lmb.cam.ac.uk/scop/pdbd/pdb40d_1.37. The limited similarity amplified the need for sensitivity in the alignment algorithms.

2.4 Finding the Genes

The testing consists of comparing each member of a protein dataset against a DNA database. The following procedure was used to create the two datasets with known relationships between the proteins and the genes.

- Use the *SCOP* domains to search *SwissProt* and find at most a single protein for each domain.
- Use the *SwissProt* annotation to identify the name of the gene associated with each protein.
- Extract genes from GenBank by name.
- Align each domain with its corresponding gene using *BlastX*.²

²*BlastX* was used instead of alignment software developed at Compugen to avoid possibly biasing the results because of code shared with *FramePlus*.

- Remove domain/gene pairs in which the aligned region contains fewer than 50 amino acids or less than 95% identity.
- Create datasets of the aligned regions of nucleotides and amino acids.
- Use the quality measurements described in the next subsection to rate each algorithm's sensitivity.

This procedure generated 757 domain/gene pairs, with an average domain length of 190 amino acid residues. The alignments exhibited 100% identity in most cases. Among all possible pairs of proteins, 902 pairs contained two proteins from the same family according the *SCOP* classification. The largest family of homologous proteins in this set included 8 proteins.

2.5 Error simulation

In order to make the tests more realistic, errors were introduced into the DNA sequences to simulate sequencing errors. The DNA fragments were modified by introducing independent insertions, deletions and misreads. One test used realistic error rate estimates of 0.45% mismatches, 0.63% insertions, and 0.24% deletions.³ A second test used very high error rates of 1% mismatches, 1% insertions, and 1% deletions.

2.6 Algorithm Benchmarking

The following test procedure was carried out for each algorithm considered. The output from each test was a set of pairs of putative homologs, ordered by alignment score. As described below, each set was divided into true and false pairs and the results from different algorithms were compared.

- Compare every sequence of the protein dataset to every sequence of the DNA dataset and record the alignment score for each pair.
- Remove self pairs (i.e., the original 757 gene/protein pairs).
- Sort the pairs in descending order by alignment score.
- Mark each pair as a true positive or a false positive, based on the family indicated in the *SCOP* annotation, as described earlier.

³These estimates are based on unpublished data from clustering and assembly of ESTs in the Compugen LEADS system.

An intuitive way to compare algorithms is to plot the number of false pairs as a function of the position in the sorted list of scores, as shown in Figure 2. The graph passes through the point (x, y) if exactly y false pairs are found among the x highest-scoring pairs in the sorted list.

The graph for an ideal algorithm, which always gives true pairs better scores than false pairs, would run along the x-axis (i.e., take value zero) from the origin through an x-value equal to the number of true pairs. After this it will be linear with a slope of unity. Similarly, the worst algorithm, which always gives true pairs worse scores than false pairs, will rise linearly with a slope of unity starting at the origin.

The graphs for all algorithms will be bounded by these two lines. Some examples appear in Figure 2. Using this approach allows inter-algorithm comparison without the need to select a threshold for accepting pairs. This approach also allows comparison of two scoring threshold schemes, such as E-values and alignment scores, with a single algorithm.

2.7 Quantifying Quality

The graphs described in the previous section are informative, but quantitative measures are easier to use for some purposes. The following three measures were used:

- ROC_{50} (receiver operating characteristic, 50) [Gribskov and Robinson, 1996]: Identify a scoring threshold with exactly 50 false pairs above it, plot the number of true pairs vs. false pairs to that position in the list, calculate the area below the graph, and normalize it to lie in the range $[0, 1]$ by dividing the measured area by the maximum possible area. Higher ROC_{50} values are preferred.
- EN (equivalence number) [Pearson, 1995]: For each position in the sorted list of pairs, determine the numbers of false positives and false negatives that would occur if that position corresponded to the threshold. Identify the unique position at which these values are equal. The number of false positives at this

position is the equivalence number. Alternatively, this is the number of false positives among the top-scoring **tp** pairs, where **tp** is the total number of true pairs. Lower EN values are preferred.

- *MER* (minimal error ratio) [Duda and Hart, 1973]: Determine the numbers of false positives and false negatives for different thresholds, as for the EN calculation. Choose as the threshold the position for which the sum of these two numbers is minimal; this sum is the MER. Lower MER values are preferred.

3 Results

The test included the programs *BlastX* (version 2.0), *TFastY* (version 3.0), *LAP*, *Translated Search*, *FramePlus*, and *FrameSearch* (Compugen GenCore version 4.5, available from the EBI at <http://www2.ebi.ac.uk/cgi-bin/genweb/admin/login.cgi>). All algorithms were tested using their default parameters, sorted by e-scores (p-scores for *BlastX*, raw scores for *LAP*). Their general behavior is illustrated in Figure 2 and the details of the critical region are highlighted in Figure 3.

The figures and the following table show that, although all algorithms are far from ideal, *FramePlus* is significantly more sensitive than the other algorithms according to all measures. For example, the last column illustrates that among the 902 highest-scoring pairs (the number of true pairs in the data), *FramePlus* finds 361 true positives and *LAP* finds only 320 (13% less). Other algorithms find even fewer.

<i>Algorithm</i>	<i>ROC₅₀</i>	<i>MER</i>	<i>EN</i>	tp – <i>EN</i>
<i>BlastX</i>	0.2317	699	628	274
<i>Trans. Search</i>	0.2291	702	590	312
<i>FrameSearch</i>	0.2331	697	607	295
<i>TFastY</i>	0.2711	658	586	316
<i>LAP</i>	0.2627	672	582	320
<i>FramePlus</i>	0.2971	639	541	361

FramePlus also outperforms the other algorithms in the case of very high error rates, though the performance difference is smaller. The results are summarized in the following table and in Figure 4. The difference in the final column has dropped to 10%. *TFastY* and *LAP* seem to perform better at high

error rates; this may indicate that their default parameters were optimized for low quality data.

<i>Algorithm</i>	<i>ROC₅₀</i>	<i>MER</i>	<i>EN</i>	tp – <i>EN</i>
<i>BlastX</i>	0.1620	764	684	218
<i>Trans. Search</i>	0.1666	768	667	235
<i>FrameSearch</i>	0.1871	744	651	251
<i>TFastY</i>	0.2229	710	639	263
<i>LAP</i>	0.2219	713	628	274
<i>FramePlus</i>	0.2270	707	611	291

The same test was run on data with no simulated errors. As this dataset does not contain frameshifts, fault tolerant algorithms should not add sensitivity. As is evident in Figure 5, this is indeed the case.

This benchmarking framework can be used for other purposes as well. The authors have used it to evaluate various statistical scoring schemes and the effect of alternative comparison matrices and gap penalties. The framework can be used to optimize such parameters for specific circumstances. In the case of realistic error-rates, change of all user controlled parameters was attempted, one at a time, in all tested algorithms. No significant improvements were recorded (data not shown). Note that the default parameters for *FramePlus* were chosen before the implementation of the benchmark, and were not influenced by it.

4 Implementation

FramePlus has been implemented in several ways. All are straightforward uses of known dynamic programming methods.

Dynamite [Birney and Durbin, 1997] and OneModel [Nachman et al., 1998] (<http://www.compugen.co.il/news/poster0398.html>) allow simple specification of dynamic programming models. They automatically generate code to run the model on special-purpose hardware accelerators. The resulting source code is efficient and contains scoring and alignment routines and linear memory alignment. Running *FramePlus* with a 300 bp query sequence against *SwissProt* on a Compugen BioXL/P-2 (entry level machine) takes 8.5 seconds.

FramePlus was also implemented as a standalone C program. The source code is freely available at <ftp.compugen.co.il/pub/research>. This implementation, when run on a 500 MHz Digital Per-

sonal Workstation, achieved a speed of approximately 1 million matrix cells per second. More precisely, comparing the first protein in the benchmark dataset (153 residues) to the entire DNA dataset (757 sequences, 426,968 base-pairs) takes 66.4 seconds using *FramePlus*, compared to 76.2 seconds for *LAP*, and 0.81 seconds for *TfastY*.

5 Discussion

This paper has presented *FramePlus*, a new algorithm for aligning DNA and protein sequences. It extends *FrameSearch* by differentiating between changes resulting from evolution and those resulting from sequencing errors. Testing with various levels of errors in the data show that *FramePlus* is consistently more sensitive than other frame algorithms.

FramePlus is easy to implement and has been accelerated on special-purpose hardware, so CPU power is not a bottleneck in using it. A typical database search of an EST vs. SwissProt using an accelerated version of *FramePlus* takes less than 10 seconds, which is competitive with heuristic frame algorithms such as *BlastX* and *TFastY*.

These considerations make *FramePlus* an important tool for anybody working with sequence alignment and sequence database searching.

The general framework used here for evaluating frame algorithms can also be used to compare other kinds of sequence alignment algorithms. It can also be used to investigate the effect of various parameters within a single algorithm.

6 Acknowledgments

We would like to thank Tim Hubbard for suggesting the method of algorithm comparison, and Ewan Birney and Hershel Safer for their advice. We would also like to thank the referees for many useful comments.

Adams, M., Kerlavage, A., C., F., and Venter, J. (1993). 3,400 new expressed sequence tags identify diversity of transcripts in human brain. *Nat. Genet.*, **4**:256–267.

Altschul, S., Boguski, M., Gish, W., and Wootton, J. (1994). Issues in searching molecular sequence databases. *Nat. Genet.*, **6**:119–29.

Altschul, S., Gish, W., Miller, W., Myers, E., and

Lipman, D. (1990). Basic local alignment search tool. *J. Mol. Biol.*, **215**:403–410.

Bairoch, A. and Boeckmann, B. (1994). The swiss-prot protein sequence data bank: Current status. *Nucl. Acid. Res.*, **22**:3578–3580.

Birney, E. and Durbin, R. (1997). Dynamite: A flexible code generating language for dynamic programming methods used in sequence comparison. In *Fifth International Conference on Intelligent Systems for Molecular Biology*, pages 56–64. AAAI Press.

Birney, E., Thompson, J., and Gibson, T. (1996). Pairwise and searchwise: finding the optimal alignment in a simultaneous comparison of a protein profile against all dna translation frames. *Nucl. Acids Res.*, **24**:2730–2739.

Boguski, M., T.M., L., and Tolstoshev, C. (1993). dbest—database for "expressed sequence tags". *Nat. Genet.*, **4**:332–333.

Brenner, S., Chothia, C., and Hubbard, T. (1998). Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl. Acad. Sci.*, **95**:6073–6078.

Burke, J., Wang, H., Hide, W., and Davison, D. (1998). Alternative gene form discovery and candidate gene selection from gene indexing projects. *Genome Res.*, **8**:276–290.

Duda, R. and Hart, P. (1973). *Pattern classification and scene analysis*. John Wiley and Sons.

Edelman, I., Faigler, S., Mintz, E., Natan, A., and Devereux, J. (1995). A rigorous alignment program for searching protein databases with nucleic acid queries. Poster, Genome Sequence and analysis Conference, Hilton-Head, Florida.

Gribskov, M. and Robinson, N. (1996). The use of receiver operating characteristic (roc) analysis to evaluate sequence matching. *Comput. Chem.*, **20**:25–34.

Guan, X. and Uberbacher, E. (1996). Alignments of dna and protein sequences containing frameshift errors. *CABIOS*, **12**:31–40.

Hein, J. (1994). An algorithm combining dna and protein alignment. *J. of Theor. Biol.*, **167**:169–174.

Henikoff, S. and Henikoff, J. (1992). Amino acid substitution matrices from protein blocks. *PNAS*, **89**:10915–10919.

Huang, X. and Zhang, J. (1996). Methods for comparing a dna sequence with a protein sequence. *CABIOS*, **12**:497–506.

Murzin, A., Brenner, S., Hubbard, T., and Chothia, C. (1995). scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, **247**:536–540.

Nachman, I., Amitai, M., Birney, E., Ventura, B., Shmulevich, D., Axel, I., and Faigler, S. (1998). Creating and accelerating new hmms through Compugen’s OneModel environment. RECOMB ’98 poster.

Pearson, W. (1995). Comparison of methods for searching protein sequence databases. *Protein Sci.*, **4**:1145–1160.

Pearson, W., Wood, T., Zhang, Z., and Miller, W. (1997). Comparison of dna sequences with protein sequences. *Genomics*, **46**:24–36.

Pederson, C., Lyngso, R., and Hein, J. (1998). *Comparison of coding DNA Combinatorial Pattern Matching*, volume 1448 of *Lecture Notes in Computer Science*, pages 153–173. Springer-Verlag.

Peltola, H., Soderlund, H., and Ukkonen, E. (1986). Algorithms for the search of amino acid patterns in nucleic acid sequences. *Nucl. Acid. Res.*, **14**:99–107.

Smith, T. and Waterman, M. (1981). Identification of common molecular subsequences. *J. Mol. Biol.*, **147**:195–197.

Zhang, Z., Pearson, W., and Miller, W. (1997). Aligning a dna sequence with a protein sequence. *J. Comp. Biol.*, **4**:339–349.

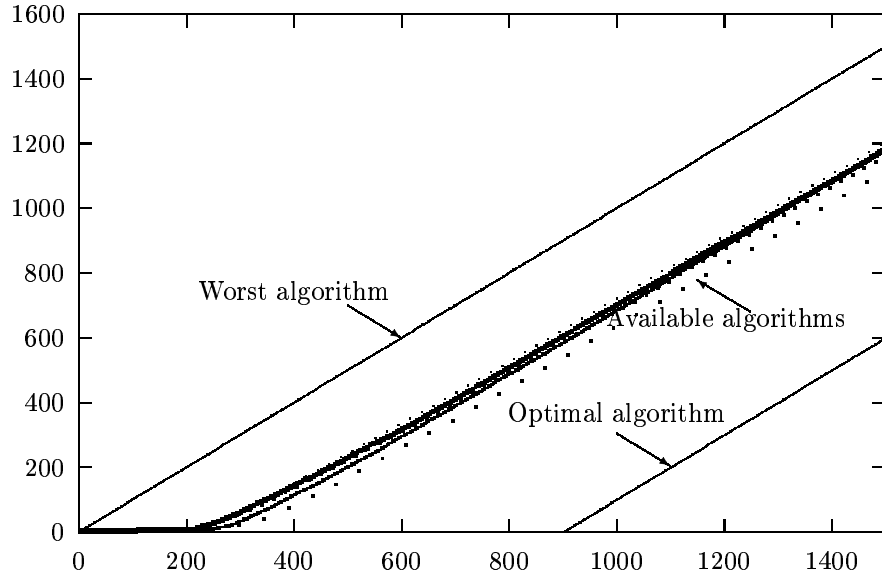


Figure 2: Performance of various frame algorithms, including theoretical best and worst algorithms, in the case of low sequence identity and realistic error rates. Each line shows the number of incorrectly reported homologs as a function of the number of reported pairs for a single algorithm. Current frame algorithms detect approximately $\frac{1}{3}$ of the structurally related pairs of genes with less than 40% sequence identity.

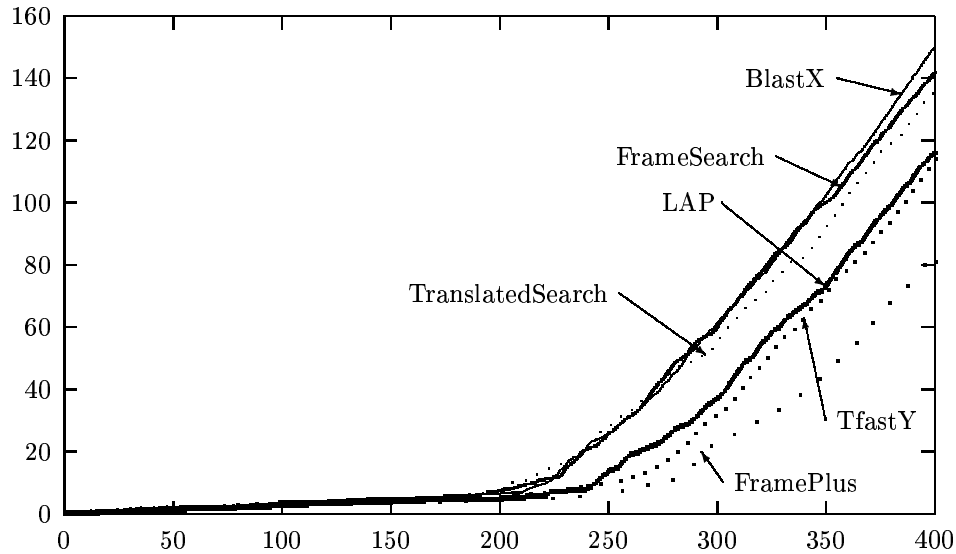


Figure 3: Closeup view of the critical region of Figure 2. *FramePlus* is the most sensitive, followed by *TFastY* and *LAP*. The other algorithms are significantly less sensitive.

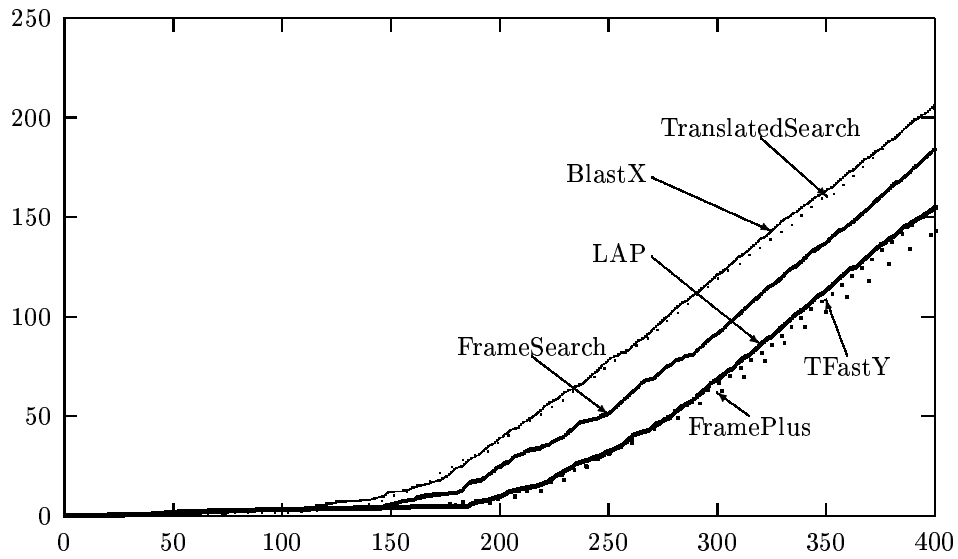


Figure 4: The graph corresponding to Figure 2 for the case of 1% mismatches, 1% insertions and 1% deletions. These values may be achieved by extremely low quality ESTs.

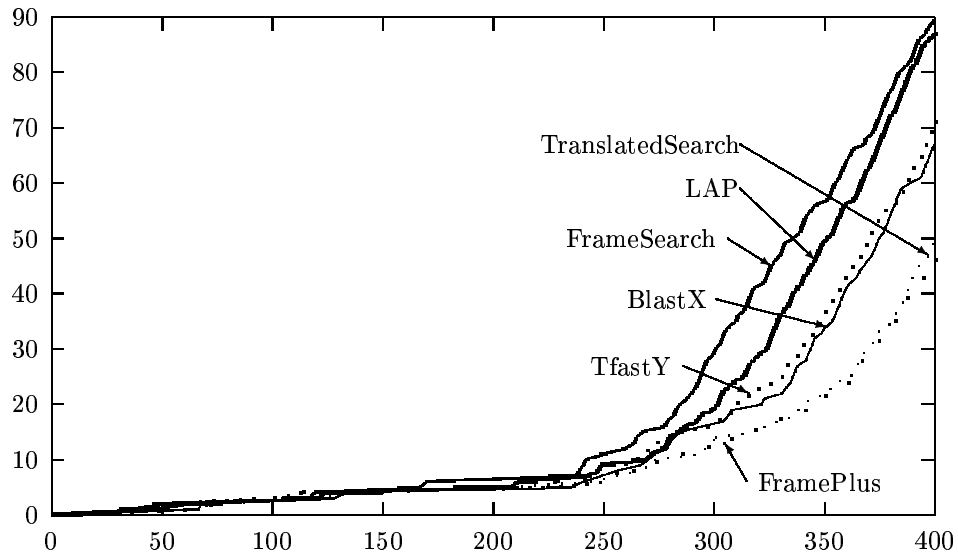


Figure 5: The graph corresponding to Figure 2 for the case of no simulated errors. In this situation, *Translated Search* performs as well as *FramePlus*.