Efficient Algorithms and Data Structures in Support of DNA Mapping and Sequence Analysis

Progress report for Grant DE-FG03-90ER60999 February 1991 -
February 1992

D. Gusfield, E.L. Lawler, Balasubramanian, W.I. Chang, A. Cobbs, J. Kecioglu, M. Kriguer, J. Lampe, D. Naor,

1 UC Berkeley

Activities at UC Berkeley supported by DOE grant DE-FG03-90ER60999
February 1991- February 1992

William I-Wei Chang received his PhD in May, 1991 and took a postdoctoral fellowship at Cold Spring Laboratory with Tom Marr. Chang's thesis, entitled "Approximate Pattern Matching and Biological Applications," reports on the sublinear time approximate string matching algorithm described in the previous progress report. The thesis describes an adaptation of the algorithm to the *codon matching*: finding coding regions in the DNA for a given protein. The thesis also contains a chapter on a new and exceptionally fast dynamic programming algorithm for approximate string matching developed by Chang and Jordan Lampe, an undergraduate student employed as a programmer under the grant.

Chang was not able to adapt our linear and sublinear expected time algorithms to high error thresholds, i.e., thresholds approaching the level of statistical insignificance. (Although the algorithms do work properly for high thresholds, running time may increase to unacceptably high levels.)

Chang and Lawler's paper on "Approximate String Matching in Sublinear Expected Time" has been revised and submitted to *Algorithmica* for a special issue on string algorithms and applications. Chang and Jordan Lampe will present their paper on very fast dynamic programming algorithms for approximate string matching at the Conference on Pattern Matching to be held at the University of Arizona in late April.

In addition to Chang, Tandy Warnow received her PhD in May, 1991 with a thesis entitled "Combinatorial Algorithms for Construction of Phylogenetic Trees". Tandy was not supported as a research assistant under this grant. However, she was associated with our group and she and William

count as the first students to graduate with PhDs in computational biology from Berkeley.

Two graduate students, Marc Krieger and Archie Cobbs, joined the project this past year as research assistants and are beginning to do research on fragment and sequence assembly algorithms.

This Spring semester we have been offering a Seminar in Computational Biology, with incidental support from the grant for some speakers' travel expenses. This seminar has been quite well attended and has done much to stimulate interest in computational biology as a research field within Computer Science, Mathematics and Statistics. To date, the speakers have been as follows:

- Jan 23 Gene Lawler, Computer Science, UC Berkeley,
"Algorithmic problems in computational biology"
- Jan 30 John Kececiloglu, Computer Science, UC Davis,
"Exact and approximation algorithms for
DNA sequence reconstruction"
- Feb 6 Dan Gusfield, Computer Science, UC Davis,
"Approximations for multiple sequence alignments"
- Feb 13 Gene Myers, Computer Science, U Arizona
"Similarity searchers over biosequences"
- Feb 20 Vince Sarich, Anthropology, UC Berkeley,
"Clocks and phylogenies in language"
- Feb 27 Tom Ray, Sante Fe Institute,
"Evolution and ecology of digital organisms"
- Mar 5 Tandy Warnow, U Southern California
"Determining evolutionary history from molecular data"
- Mar 12 Richard Karp, Computer Science, UC Berkeley,
"Some approaches to the contig assembly problem"
- Mar 19 Charles Sibley, San Francisco State U,
"Phylogenetic reconstruction from DNA comparison"

2 UC Davis

Activities at UC Davis supported by DOE grant DE-FG03-90ER60999 February 1991 - February 1992

Below we discuss several distinct research efforts during the past year. Several of these efforts continue work begun during the first year of the grant.

1. PARAL and XPARAL: a software package to efficiently and optimally align sequences using parameterized match, mismatch, indel and gap weights, D. Gusfield, K. Balasubramanian, J. Bronder, D. Mayfield, D. Naor and P. Stelling.

In the first year of the project we began developing a software package for parametric sequence analysis called PARAL. It is described in the next paragraph. The progress on PARAL since last years report is described after that.

In the central problem of aligning two sequences (strings) there is considerable disagreement about how to weight (score) matches, mismatches, insertions, deletions, and gaps. PARAL allows the user to avoid specifying exact weights by (provably efficiently) computing optimal alignments as a *function of variable weights*. For example, consider the problem of aligning two strings to maximize ($\#$ of matches - α $\#$ of mismatches - β $\#$ indels), where α and β are variables. In one mode, the user inputs a choice for α and β , PARAL computes an optimal alignment A for that choice, and then determines and displays the region P in α, β space such that A is an optimal alignment for *any* point in P , and for *no* points outside of P . That space can be proved to be a *convex polygon*. In a more complex mode, PARAL completely partitions the α, β space into such convex polygons. The central use for PARAL is to study the sensitivity of the problem to choices made for the weights, and to give a small set of alignments such that for any choice of α, β , one of the alignments is optimal, and can be found quickly.

In the last year we have worked on two extensions to PARAL. First we have developed a very user-friendly interface to the system written for X-windows and in particular for Motif. With this interface the user interacts with menus and windows to indicate the type of analysis to do, and to see both the polygon decomposition and the associated alignments. The user can explore the solutions in the decomposition or localize the computation for greater efficiency. The new system is called XPARAL. We expect to widely distribute the system, and so a highly robust and friendly user interface is very important. Second, we have devised a faster approach to finding the complete decomposition. This approach reduces the number of fixed-parameter alignments that have to be computed, and has the potential to significantly increase the running time of the method. We are presently incorporating these ideas in the system.

2. Efficient algorithms for multiple sequence alignment with guaranteed error bounds, Dan Gusfield,

In the first year of the grant we worked on multiple sequence alignment. An introduction to that work is described in the next paragraph, and progress since the last report is described after that.

Multiple string (sequence) alignment is a difficult problem of great value in computational biology. There are two rather different ways to measure the goodness of a multiple alignment, depending on whether one wants to find conserved subpatterns in the set of strings, or to build a tree deriving the strings from a common ancestor. For both measures, no efficient algorithm is known for optimally solving the problem for any but very small cases. A common approach in computer science to computationally hard problems is to develop fast (heuristic) algorithms whose maximum possible deviation from the optimal solution can be *proven* to be bounded by a small multiplicative factor. For the multiple string alignment problem (under either measure), no bounded error methods have been reported. In this work we provided the first such bounds. We developed two computationally efficient multiple alignment methods (one for each objective function) whose value is guaranteed never to be more than twice the value of the optimal multiple alignment. In the case of one of the objective functions, the guaranteed bounds are even smaller when the number of strings is small (1.33 for three strings of any length), and in the case of the other objective function, the method also yields a non-obvious *lower bound* on the value of the optimal solution.

In the last year we have continued work on multiple sequence alignment in both theoretical and empirical directions. In the theoretical direction we extended the above results to randomized algorithms. A randomized algorithm is one which makes some random decisions during its computation rather than being fully specified. The purpose of a randomized algorithm is to increase program speed at the cost of some certainty about the goodness of the output. The goal of theoretical work is to precisely specify the form of that trade-off. The non-randomized heuristic reported above requires the computation of k^2 pairwise string alignments. Assuming that all strings are of length n , the time is then $O(n^2k^2)$. In the randomized version of the algorithms, the time is reduced to $O(n^2pk)$ with the following results:

Theorem 2.1 *Using only time $O(n^2pk)$ the randomized algorithm will produce an alignment which, for any r , will be within a factor of $2 + 1/(r - 1)$ of the optimal alignment with probability at least $1 - [(r - 1)/r]^p$.*

An alternate way to say this is the following:

Theorem 2.2 *For any $r > 1$, define $e(r)$ to be the expected number times you must run the randomized algorithm (where each run takes time $O(n^2k)$) before the value of best resulting alignment is within a factor of $2 + 1/(r - 1)$ of the optimal alignment. Then $e(r) \leq r$.*

For example, $e(r)$ is at most two for an error bound of 3, and $e(r)$ is at most ten for a bound of 2.1112. Note that $e(r)$ is independent of k and of the lengths of the strings. What these two theorems say is that one can significantly speed up the heuristic while preserving most of its value as a guaranteed approximation method.

On the empirical side, we ran the method on several sets of biological sequences and rated the value of the alignments against a weak, but easily computed, lower bound on the optimal value. We used the method to align 19 Homeoboxes from different species. In this case the heuristic found a solution that was within 2% of the lower bound, hence within 2% of the optimal. The randomized version of the algorithm was also tested, and it achieved an alignment within 5% of the lower bound. At the other end, we ran experiments with strings that were not so similar as homeoboxes. In that case the algorithm always produced a solution that was within 16% of the lower bound, and hence within 16% of the optimal. However, in cases where the strings are quite dissimilar, we believe that the lower bound used is fairly far from the optimal, hence the deviation of the algorithm produced alignment and the optimal one is believed to be well below 16%.

3. An efficient algorithm for the all-pairs suffix-prefix problem, Gusfield, Landau, and Scheiber.

This is completely new work begun since the last report.

For a pair of strings (S_1, S_2) , define the suffix-prefix match of (S_1, S_2) to be the longest suffix of string S_1 that matches a prefix of string S_2 . The following problem occurs in several problems concerning sets of strings: Given a collection of strings $S_1, S_2, \dots, S_k$ of total length m , find the suffix-prefix match for each of the k^2 ordered pairs of strings. We developed an algorithm that solves the problem in $O(m + k^2)$ time. That is, the time to find all k^2 desired suffix-prefix matches runs in time proportional to just the length of the input, plus the length of the output. This is certainly time optimal and although the algorithm is simple, it is surprising that such a fast solution is possible. The problem arises, for example, in the first phase of the recent Blum et. al. approximation result for the shortest common superstring problem, which has been taken as a model for sequence assembly.

4. On the Number of Solutions to the Probed Partial Digestion Problem, Lee Newberg (UCB) and Dalit Naor (UCD).

In last years report we discussed results on the number of solutions to the probed partial digestion problem. Here we introduce that problem and discuss progress on that problem since the last report.

The **Probed Partial Digestion** (PPD) mapping scheme is used to generate physical maps of large DNA strands. Initially, the DNA strand is partially digested with a restriction enzyme (RE), and a collection of all DNA fragments between *any* two cutting sites is generated (the fragments may contain cutting sites in the middle). In the second stage a labeled probe, which attaches to the DNA between two RE cutting sites, is hybridized to the partially digested DNA, and the sizes of fragments to which the probe hybridizes are measured. The objective is to reconstruct the linear order of the RE cutting sites from the multiset of measured lengths.

The computational problem of reconstructing the linear order is a difficult one. One of the difficulties is the multiplicity of solutions; i.e., in many cases, more than one underlying linear ordering is consistent with a multiset of measured lengths, although in reality there is only one true linear order for the RE cutting sites. Since, without additional information, there is no way to distinguish between the true linear order and any other consistent order, the objective is to reconstruct *all* possible solutions and leave it to the biologists to decide which solution is most likely to be the right one.

It was earlier shown that PPD is equivalent to the following multiset-addition problem: Let X and Y denote sets of nonnegative integers. Let S be a multiset of nonnegative integers containing exactly one zero. Given S find all unordered pairs $\{X, Y\}$ such that

$$S = X + Y = \{x + y : x \in X, y \in Y\}.$$

It then remains to find how many solutions this multiset-addition problem, and therefore the PPD problem, might have for a given input multiset S . In our recent work in the last year we further show that this question is related to a number-theoretic function $H(N)$: $H(N)$ is the number of ways to factor an integer N into the product of integers, each greater than one, where the order of factors is significant.

If $\#ppd(S)$ is the number of solutions for a multiset S that contains exactly one zero, and $\#PPD(N) = \sup\{\#ppd(S) : |S| = N\}$, then it turns

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

out that $H(N)$ is a lower bound on $\#PPD(N)$. We show that

$$\limsup_{N \rightarrow \infty} \frac{\#PPD(N)}{N^t} = \infty$$

for all $t < \zeta^{-1}(2)$ where $\zeta(t)$ is the Riemann Zeta Function and $\zeta^{-1}(2) \approx 1.73$; alternatively, that for worst case data sets, the probed partial digest problem with an input of N fragment lengths will have $\Omega(N^t)$ solutions for any $t < \zeta^{-1}(2)$.

More specifically, If S_N is the set $\{0, 1, \dots, N-1\}$, then for every value of $t < \zeta^{-1}(2)$ there must exist an infinite sequence $\{N_1, N_2, \dots\}$ satisfying

$$\lim_{i \rightarrow \infty} \frac{\#ppd(S_{N_i})}{N_i^t} = \infty.$$

We do not have an algorithm to *construct* such a sequence for arbitrary values of t . However, we can construct some sequences. We give a series of examples to such sequences. The examples are:

For $N_i = 2^i$, $\#ppd(S_{N_i}) = N_i/2 = 2^{i-1}$.

For $N_i = 3 \cdot 2^i$, $\#ppd(S_{N_i}) = \frac{N_i \lg(4N_i/3)}{6} = (i+2)2^{i-1}$.

For any t satisfying $\frac{6^t}{2} - 3^t - 2^t + 1 < 0$ (i.e. $t < 1.4352791$) there exists infinitely many values of N of the form $N = 2^i \cdot 3^j$ satisfying $\#ppd(S_N) = \Omega(N^t)$.

Conclusions - As this is only a lower bound, it still remains to find an upper bound on this number. Also, it is still left to be determined how bad this phenomenon is in the average case. The PPD experiment is still viable, if there is only one solution (or very few solutions) on average. multiset-subtraction problem for any multiset S of size N .

3 Papers appeared or written since the last report:

1. Parametric Optimization of sequence alignment, Gusfield, Balasubramanian and Naor.

Appeared in the Proceedings of the Third Annual SIAM/ACM Symposium on Discrete Algorithms, January 1992. It is also submitted for journal publication. This paper details the mathematical results proved about parametric sequence alignment.

2. Efficient algorithms for multiple string alignment with guaranteed error bounds. D. Gusfield.

To appear in the Bulletin on Mathematical Biology.

3. XPARAL: A software package for parametric sequence alignment with variable match, mismatch, indel, and gap weights, D. Gusfield, K. Balasubramanian, J. Brondor, D. Mayfield, D. Naor and P. Stelling.

This paper describes the program XPARAL, the ideas behind it, and how to use it.

4. Efficient solution to the all-pairs suffix-prefix problem, Gusfield, Landau, Scheiber.

To appear in the proceedings SEQUENCES 91, June, 1991. Also to appear in Information Processing Letters.

5. A Lower Bound on the Number of Solutions to the Probed Partial Digest Problem, D. Naor and L. Newberg.

Submitted to the Bulletin of Mathematical Biology.

4 Research for the next year

4.1 Proposed research for Dan Gusfield, et al

1. XPARAL: implementing the speed ups.

In the next year we will continue the development of XPARAL with the objective of making it extremely portable, reliable, and fast. The work to date has shown that the basic ideas behind the method work and can be correctly implemented. We have also put in a user interface that makes the package easy to use. Our emphasis now is to improve the speed of the package by implementing the new ideas that we have developed in the past year, and also to increase its portability to many different machines. This effort is justified by our belief that XPARAL can be a very important tool in the hands of a wide range of users.

2. All pairs common substrings.

The following problem is solved as the the first step of a fragment assembly method discussed by Soderlund et.al. in "Stochastic solutions for genomic assembly problems". Given a set of k strings, find for each of the $\binom{k}{2}$

pairs of strings, the longest substring common to both strings. The fastest way presently known to solve this problem requires $O(km)$ time, where m is the length of all the strings. This problem is similar to the all-pairs suffix-prefix problem discussed above, where a surprisingly fast solution, $O(m)$ time, was obtained. However, the technique used in the suffix-prefix problem does not readily extend to this problem, and we are presently working on the problem.

3. All pairs alignment.

Continuing in the direction discussed in topic 2, the more general problem is the following: If we have k strings and want to optimal alignment between each pair of strings, can we compute this significantly faster than by considering each pair separately. The question directly addresses the issue of transitivity in alignment. That is, if we know how strings A and B align (or more exactly we know their entire alignment dynamic programming table) and we know how strings B and C align, can we use that information to speed up the alignment of strings A and C . Although the statement of this problem is abstracted from any specific application, it is a fundamental problem, and an affirmative answer to this question would have wide applications and implications in many applied sequence analysis problems.

4. Shortest common superstring - speedup and models.

The shortest common superstring has been taken as a working model for the sequence reconstruction problem of determining the correct full sequence of many overlapping smaller sequences. The problem of finding the smallest superstring is NP-hard, and hence unsolved. However, there are some heuristics for the problem with bounded worst case deviation from the shortest solution. At present, the best method gives a deviation with a guaranteed bound of three. We propose two things. First to further reduce the bound on the guaranteed deviation in the shortest superstring model, and second to develop more appropriate reconstruction models, and solutions to these models. For example, in the superstring model, every input string must appear somewhere as a substring of the superstring. However, there is no requirement that every character on the superstring be part of more than one input string. This is reasonable if nothing is known about the input strings. But when the input strings come from shotgun sequencing experiments, one expects that most points on the reconstructed string will be part of several of the input strings. Hence we would like to explicitly incorporate such constraints into the problem statement, so that the solution is more likely to be meaningful.

5. Alignment problems with convex gap costs.

In most biological models of alignment, a gap (a consecutive set of spaces) is given a score that is different from the simple sum of scores of the spaces making up the gap. When a gap is given a score that is constant, independent of its length, or is a linear function of its length, then the best alignment can be computed as quickly as without scoring gaps. However, the time needed to solve the problem increases when gaps are scored as an arbitrary function of length, although when the score is a convex (or concave) function of length, an important speedup is possible. There are two important remaining problems with this speedup for convex gap scores. First, existing methods require, in worst case, more space than do solutions for the case when gaps are not scored. Second, with convex gap costs, the speedup only works for global alignment, not for the much more important case of local alignment. We have begun looking at these two problems, will continue in this direction in the next year.

6. Multiple sequence alignment.

There are many problems in this general area. The first is to examine more scoring methods than were examined in the results described in paragraph 2 of section 2. In that work, a multiple alignment was scored by the sum-of-pairs method, and by the tree-alignment method. For both of these scoring systems, the heuristics we developed find a multiple alignment that is no more than twice the score of the optimal one. But other scoring systems need to be examined. One important one is the consensus scoring system: Given an alignment, the most frequent character in each column is taken as the consensus character (provided it appears above a certain threshold frequency). Then, each column is scored in the alignment by summing the distance (taken from a distance matrix) of each character in the column to the consensus character for that column. The alignment is scored by summing the column scores. We would like a fast heuristic method whose score is always within a fixed deviation of the optimal

Other scoring schemes and proposed research on multiple sequence alignment appear in the next subsection.

4.2 Proposed research for John Kececioğlu

John Kececioğlu is partially supported under the present DOE grant. In the next year he will concentrate on the following three topics, concerning multiple sequence alignment and physical mapping, as well as his participation

in parts of the preceeding proposed topics.

7. Maximum weight multitrace

Recently, [Ke91] formulated a new multiple sequence alignment problem, the *maximum weight multitrace problem*. The input to the problem is a collection of sequences, together with a set of pairs of characters from the sequences. Each pair represents the constraint that the two characters appear in the same column of the alignment, and each pair can have an associated weight. The output is a subset of the pairs of maximum total weight that can be simultaneously met by a multiple sequence alignment.

This formulation has several nice features. Biologists often form multiple alignments by merging pairwise alignments. The maximum weight multitrace problem formalizes this process, in that the pairs of characters for the input may come from any set of pairwise alignments. Interestingly, character pairs can also come from *local similarities*, i.e. optimal alignments between portions of the sequences. For such an input, maximum weight multitrace provides a rigorous way to construct multiple sequence alignments from local similarities.

[Ke91] presents a fast approximation algorithm for maximum weight trace that finds a solution guaranteed to have weight within a factor of $2/k$ of the maximum, where k is the number of sequences in the input. We propose to investigate exact algorithms for maximum weight multitrace, since the additional structure imposed by the input suggests we may be able to compute an optimal solution while filling in significantly less than all $O(n^k)$ entries in the dynamic programming matrix.

8. Minimal mutation trees

One of the oldest multiple sequence alignment problems is the *minimal mutation tree problem* examined in [Sa75]. The input is a collection of sequences, together with a tree whose leaves are labeled by the sequences. The problem is to determine ancestral sequences for the internal nodes so as to minimize the total number of mutations between sequences that are adjacent in the tree. Such a solution induces a natural multiple sequence alignment over the input sequences.

The algorithm of [Sa75] takes $O(n^k)$ time for k sequences of length n . We propose to apply a technique implemented in [LAK89] to reduce the running time when the input sequences are well-clustered by the tree. At heart this technique computes lower bounds on the cost of an extension of a partial alignment, and uses these bounds to eliminate paths through the dynamic programming matrix.

9. Ordering a clone library

One of the most basic problems in the human genome project is that of creating an *ordered clone library*. Each clone in the library represents a fragment of DNA from a common DNA molecule. The data obtained for a clone is usually the set of DNA probes that hybridize to it, or the set of fragment lengths from a complete digest with a restriction enzyme. Given the data, the object is to determine the order of the clones along the molecule. In one formulation, each clone is a set from an alphabet Σ , and one seeks a shortest sequence S over Σ such that every clone set is equal to the set of characters in some substring of S . Sequence S implicitly specifies an ordering of the clones; in this formulation the shortest sequence gives the best ordering.

Recently, [Ka92] has shown that the problem of evaluating an ordering, i.e. determining the length of a shortest sequence in which the clones appear in a specified order, can be solved efficiently when the data meets some assumptions. These assumptions are (1) that no clone set is a subset of another clone set, and (2) that the data is error-free. The assumptions are not always met in practice. Even if in the molecule no clone fragment is contained in another, it is possible for one clone set to be contained in another, especially with probe hybridization data; throwing out clones with contained sets ignores information that may be important for recovering the correct order. In addition, with fragment length data, the lengths are not known precisely, but only to within some experimental error; considering lengths to match when they agree within some percentage does not remove the problem, since the resulting match relation is no longer transitive.

We propose to extend the work of Karp, using ideas developed in [Kec91], to develop a method tailored for fragment length data that works even in the presence of containments and inexact lengths.

4.3 Proposed research for E. L. Lawler, et al

10. PQ/decomposition trees for clone ordering

We have obtained a number of results on sequence and fragment assembly which we intend to improve upon in the next few months. In particular, we have adapted techniques from machine scheduling theory to the problem of ordering clones tested for unique probes or STSs, and with ordering constraints resulting from in situ hybridization experiments. In the terminology of scheduling theory, the probes are 'tasks' to be sequenced, the probe

data provide 'contiguity' constraints on the tasks, and in situ hybridization experiments provide 'precedence' constraints. All such constraints can be represented in a data structure which combines the attributes of a 'PQ' tree and a digraphic decomposition tree.

We are currently working on techniques for dealing with inconsistencies in data, and in using the PQ/decomposition trees to find either maximum likelihood or most parsimonious solutions to data error problems. We plan to write computer code for manipulating the PQ/decomposition trees, which can then be tested on actual data.

11. Traveling salesman models for fragment assembly

Some versions of the fragment assembly problem can be formulated as something close to a traveling salesman problem or TSP. Recently, our colleague Richard Karp has formulated one version of this problem in which oligonucleotide are used to probe fragments for which, by assumption, no fragment is properly contained in another. This formulation of the problem makes it particularly attractive to try applying standard TSP solution methods, e.g., 2-opting and 3-opting, to the resulting instance of the TSP. However, because the problem being solved is not exactly a TSP, it is not so easy to recompute the cost function after a local improvement is made. We will adapt, program and test several of the well known TSP heuristics to apply to this problem.

12. Multiple string alignment

We are plan to consider extensions of the multiple sequence alignment results of Gusfield and Pevzner to "local" multiple sequence alignment: finding relatively short substrings of multiple sequences for which good alignments exist. This involves a careful formulation of the alignment objective function in order to make this feasible. We wish to make it feasible to incorporate convex gap penalties into the model.

References

- [LAK89] LIPMAN, DAVID J., STEPHEN F. ALTSCHUL and JOHN D. KECIOGLU (1989). "A tool for multiple sequence alignment," *Proceedings of the National Academy of Science USA* 86 (1989) 4412-4415.
- [Kar92] KARP, RICHARD (1992). Personal communication.

- [Kec91] KECECIOGLU, JOHN (1991). *Exact and Approximation Algorithms for DNA Sequence Reconstruction*, Ph.D. dissertation, Technical Report 91-26 (December 1991), Department of Computer Science, The University of Arizona, Tucson, Arizona, 85721.
- [San75] SANKOFF, DAVID (1975). "Minimal mutation trees of sequences," *SIAM Journal on Applied Mathematics* 28 (1975) 35-42.

END

**DATE
FILMED
5/13/92**