MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0384541 8

ornl

ORNL/TM-12764

OAK RIDGE NATIONAL LABORATORY

MARTIN MARIETTA

A Multiple Divide-and-Conquer (MDC) Algorithm for Optimal Alignments in Linear Space

X. Guan
E. C. Uberbacher

OAK RIDGE NATIONAL LABORATORY
CENTRAL RESEARCH LIBRARY
CIRCULATION SECTION
4500N ROOM 175
LIBRARY LOAN COPY
DO NOT TRANSFER TO ANOTHER PERSON
If you wish someone else to see this report, send in name with report and the library will arrange a loan.

MANAGED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831; prices available from (615) 576-8401, FTS 626-8401.

Available to the public from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161.

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 a 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.

ORNL/TM-12764

Engineering Physics and Mathematics Division

A MULTIPLE DIVIDE-AND-CONQUER (MDC) ALGORITHM FOR OPTIMAL ALIGNMENTS IN LINEAR SPACE

X. Guan and E. C. Uberbacher

DATE PUBLISHED - June 1994

Research supported by the Office of Health and Environmental Research, U.S. Department of Energy, and the Laboratory Directed Research and Development Programs

Prepared by the
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831
Managed by
MARTIN MARIETTA ENERGY SYSTEMS, INC.
for the
U.S. DEPARTMENT OF ENERGY
under Contract No. DE-AC05-84OR21400

MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0384541 8

CONTENTS

	Page No.
ABSTRACT	v
1. Introduction	1
2. Dynamic Programming Algorithms	1
3. A New Linear Space Alignment Algorithm	3
4. Analysis and Results	4
5. Conclusions	6
Acknowledgments	7
References	8

ABSTRACT

This paper describes an algorithm for optimal sequence alignments in linear space. A new multiple divide-and-conquer technique is presented that leads to a linear space alignment algorithm which improves upon an existing algorithm by Myers and Miller.

1 Introduction

Dynamic programming algorithms are often used to find the similarities of sequences as well as to deliver the actual alignment of two sequences. Two kinds of alignments are used to compare sequences: local alignments and global alignments. The local alignments attempt to locate conserved regions, while the global alignments identify overall relationship between two sequences.

While dynamic programming algorithms are relatively time consuming, the space required is often the limiting factor when aligning long sequences. A linear space algorithm for computing maximal common subsequences, proposed by Hirschberg [1], was applied by Myers and Miller [2] to deliver optimal alignments in linear space. We have improved the Myers and Miller algorithm by introducing a multiple divide and conquer technique that reduces the algorithm's running time while maintaining its linear space property.

In the following sections, we use brackets [] to represent a matrix and parentheses () to represent a single entry in the matrix: $D[m,n]$ represents a matrix D, while $D(i,j)$ represents a single entry in the matrix D at the ith row and the jth column.

2 Dynamic Programming Algorithms

We first give a dynamic programming algorithm for global sequence alignments. Given two sequences $A = a_1a_2...a_m$ and $B = b_1b_2...b_n$, a cost, w, for k insertions or deletions is defined as $w(k) = (u * k + v)$, where $u \geq 0$ and $v \geq 0$. The minimum cost of aligning the two sequences is given in [3]:

$$D(i,j) = min \begin{cases} D(i-1,j-1) + c(b_i, a_j), \\ P(i,j), \\ Q(i,j). \end{cases} \qquad 0 < i \leq m, 0 < j \leq n$$

where

$$P(i,j) = min \begin{cases} D(i-1,j) + w(1), \\ P(i-1,j) + u. \end{cases}$$

$$Q(i,j) = min \begin{cases} D(i,j-1) + w(1), \\ Q(i,j-1) + u. \end{cases}$$

$$P(0,k) = Q(k,0) = D(k,0) = D(0,k) = w(k), \forall k > 0$$

and

$$c(b_i, a_j) = \begin{cases} \geq 0 & if \;\; b_i \neq a_j \\ \leq 0 & if \;\; b_i = a_j \end{cases}$$

The value $D(m, n)$ represents the minimum cost of aligning the two sequences.

To deliver the optimal alignment (that corresponds to the minimal cost), a traceback procedure is performed to find the actual alignment. A straightforward implementation of the traceback procedure requires $O(nm)$ space, thus limiting it to only short sequence alignments.

If only the minimum cost is needed and not the alignment, linear space is enough to do the calculation. As the value in position (i, j) depends only on values in three other positions, i.e., $(i, j-1)$, $(i-1, j-1)$ and $(i-1, j)$, one can do the calculation in a row by row fashion using only linear space (see [2] for details on the linear space cost only algorithm). This algorithm will be referred to later as the linear-space-cost-only algorithm.

The Hirschberg algorithm, applied by Myers and Miller to sequence alignments [2], is a divide-and-conquer algorithm designed to compute the alignment using linear space. The idea is to divide the first sequence into two halves A_1A_2 and then to locate the optimal point in the second sequence (which divides the second sequence into two parts B_1B_2) such that the optimal alignment is the concatenation of the optimal alignment of A_1 and B_1 and the optimal alignment of A_2 and B_2 (see Figure 1).

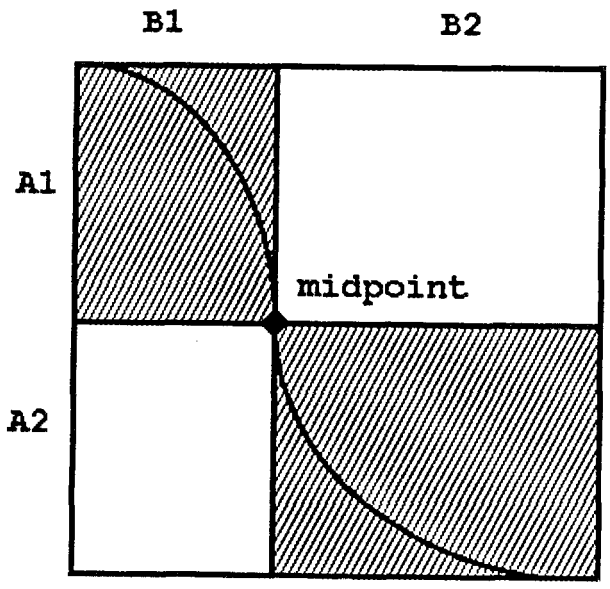


Figure 1. Divide and conquer.

The optimal alignment of two sequences can be thought of as a path in the cost matrix that starts at $D(0,0)$ and ends at $D(m,n)$. This path must cross the middle row (*midrow*) $i^* = \frac{m}{2}$. The problem is to find the point (*midpoint*) where the optimal path crosses the *midrow*.

Myers and Miller's algorithm uses the linear-space-cost-only algorithm to find the *midpoint*. First it calculates

- $DF(j)$ = the minimum cost of aligning $a_1a_2...a_{i^*}$ and $b_1b_2...b_j$
- $DR(j)$ = the minimum cost of aligning $a_{i^*+1}...a_{n-1}a_m$ and $b_{j+1}b_{j+2}...b_n$

where $0 \leq j \leq n$. Then it calculates

$$j^* = min_{j\in[0,n]}\{DF(j) + DR(n-j)\}.$$

(i^*, j^*) is the optimal *midpoint* (for a proof see [1]).

The optimal alignment is obtained by recursively finding the optimal alignment of the two subsequences before the *midpoint* and the optimal alignment of the two subsequences after the *midpoint*.

The actual algorithm is a bit more involved as it considers the types of the operations (replacement, delete, etc.) before and after the *midpoint* on the optimal path, and divides the problem accordingly.

3 A New Linear Space Alignment Algorithm

We first introduce a new technique to find the *midpoint*. We define a matrix $CROSS[m,n]$: $CROSS(i,j)$, when $i \geq \frac{m}{2}$, contains the column of the cost matrix where the optimal path that starts at $D(0,0)$ and ends at $D(i,j)$ crosses the *midrow*. $CROSS(i,j)$ is undefined for $i < \frac{m}{2}$. When both $D[m,n]$ and $CROSS[m,n]$ have been calculated, $D(m,j)$ contains the minimum cost of aligning $A = a_1a_2...a_m$ and $B = b_1b_2...b_j$, $0 \leq j \leq n$, and $CROSS(m,j)$ contains the column where the optimal path that starts at $(0,0)$ and ends at (m,j) crosses the *midrow*. In particular, $CROSS(m,n)$ contains the *midpoint*.

The calculation of $CROSS[m,n]$ is straightforward. Suppose we are calculating $D(i,j)$. The last operation in Figure 2 refers to the operation (i.e., a replacement, a deletion, or a insertion) that leads to $D(i,j)$. The path in Figure 2 refers to the optimal path that starts at $D(0,0)$ and ends at $D(i,j)$.

- If $(i = i^* + 1)$ then
  - $CROSS(i,j) = j$ if the last operation is a deletion,
  - $CROSS(i,j) = j - 1$ if the last operation is a replacement,
  - $CROSS(i,j) = CROSS(i,j-1)$ if the last operation is an insertion.
- else pass on the crossing-point information
  - $CROSS(i,j) = CROSS(i-1,j)$ if the last operation is a deletion,
  - $CROSS(i,j) = CROSS(i-1,j-1)$ if the last operation is a replacement,
  - $CROSS(i,j) = CROSS(i,j-1)$ if the last operation is an insertion.

Figure 2. Calculation of $CROSS(i,j)$ at $D(i,j)$.

Like $D[i,j]$, $CROSS[m,n]$ can be calculated easily in linear space. All we need is the value $CROSS(m,n)$. The cost is the added operation and the extra n space. We call the linear space cost only alignment algorithm with the added matrix $CROSS[m,n]$ the modified-linear-space-cost-only alignment algorithm.

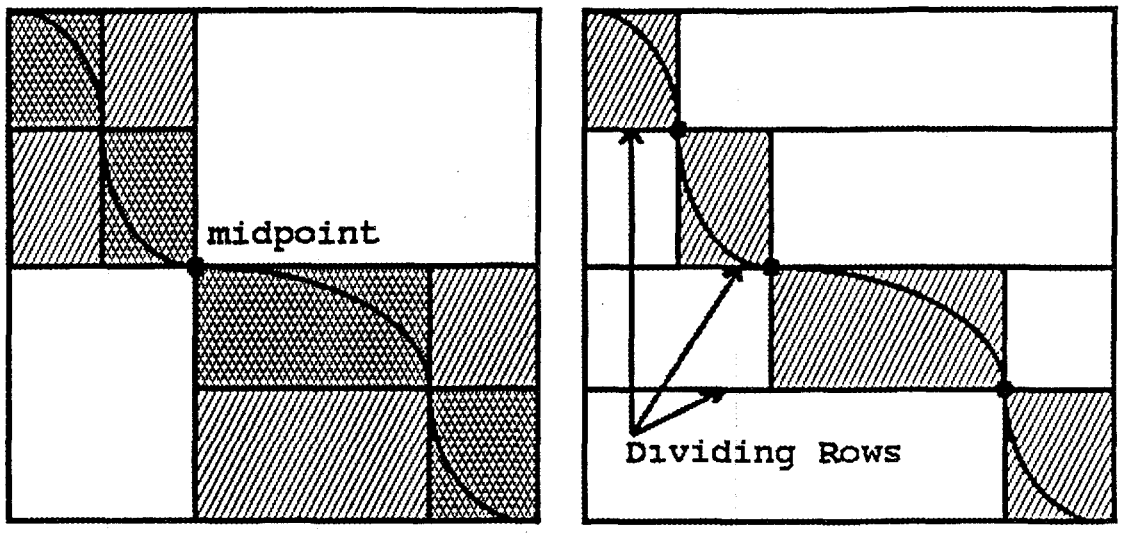
A natural extension of the above divide-and-conquer technique is to use more than one *dividing row*. The calculations of where the optimal path crosses the *dividing rows* are similar to that for *midrow*. Extra kn space is need to store the crossing information for k *dividing rows* .

The main MDC algorithm is outlined as follows: given two sequences,

- apply the modified-linear-space-cost-only alignment algorithm to find and save the crossing information at these k *dividing rows*,
- recursively divide each sequence into k subsequences according to the crossing information at these k *dividing rows* until the alignment is trivial to do,
- the optimal alignment is the concatenation of the optimal alignments of these k pairs of subsequences.

4 Analysis and Results

Let T_1 be the time of linear-space-cost-only algorithm. The total area of the calculation of the linear-space-cost-only algorithm is the cost matrix D. Using the Myers and Miller algorithm, after one recursion, the problem is divided into two subproblems whose total area is half of the cost matrix. In turn these two subproblems are divided into four subproblems whose total area is one fourth of the cost matrix, and so on (see Figure 3(a)).



(a) The Myers and Miller algorithm. (b) The MDC algorithm.

Figure 3. Divide and conquer.

So the approximate time T_2 for the Myers and Miller algorithm is

$$T_2 = T_1 * (1 + \frac{1}{2} + \frac{1}{4} + \cdots) \approx 2T_1$$

If four *dividing rows* are used in the MDC algorithm, each recursion divides a problem (or a subproblem) into four subproblems (see Figure 3(b)).

So the time of our algorithm T_3 is approximately

$$T_3 = T_1 * (1 + \frac{1}{4} + \frac{1}{4^2} + \cdots) \approx \frac{4}{3}T_1$$

In general, let d be the number of *dividing rows.* A problem is recursively divided into $s = d + 1$ subproblems, so T_3 is

$$T_3 \approx \frac{s}{s-1}T_1$$

This is only an estimate since the cost of the added operation is not taken into account. But as demonstrated below, the gain as a result of the introduction of the multiple divide-and-conquer technique is much more than the cost, resulting in substantial saving in the execution time.

To test the method, we used a simplified version, where $w(k) = u*k$. In the following table, MDC4, MDC8, and MDC16 refer to the MDC algorithm when $s = 4$, 8, or 16 respectively, and seq_len is the length of each of the two sequences. The algorithms were tested on a SUN SPARC 10 Workstation. We used a straightforward implementation of the algorithm. A refined version that reduces the cost of the newly introduced operation in the linear-space-cost-only algorithm will lead to a faster algorithm.

seq_len	Myers	MDC4	MDC8	MDC16
3500	0:34	0:29	0:25	0:23
7000	2:41	1:55	1:41	1:35
14000	9:13	7:45	7:07	6:21
28000	37:05	31:40	27:05	26:08

Table 2. Times (in minutes:seconds) of the linear space alignment algorithms

5 Conclusions

Sequence alignments using dynamic programming algorithms are demanding in both time and space, so efficient sequence alignment algorithms have been an active topic in computational biology. Here we have presented a multiple divide-and-conquer technique which improves the linear space alignment algorithm proposed by Myers and Miller. Still, aligning long sequences takes considerable time, so approximate algorithms which use prescreening methods such as dot matrix and k-tuple look tables, may be used together with this optimal dynamic programming algorithm to produce even more efficient systems.

6 Acknowledgments

This Research was supported by the Office of Health and Environmental Research, U.S. Department of Energy under Contract No. DE-AC05-84OR21400 with Martin Marietta Energy System, Inc.

References

[1] D. S. Hirschberg, "A Linear Space Algorithm for Computing Maximal Common Subsequences," *Communications of the ACM,* vol. 18, 1975, 341-343.

[2] E. W. Myers and W. Miller, "Optimal Alignments in Linear Space," *CABIOS*, vol. 4, 1988, 11-17.

[3] O. Gotoh, "An Improved algorithm for matching biological sequences," *Journal of Molecular Biology*, vol. 162, 1982, 705-708.

INTERNAL DISTRIBUTION

1. B. R. Appleton
2. M. Beckerman
3. J. R. Einstein
4. C. W. Glover
5. W. C. Grimmell
6-10. X. Guan
11. J. P. Jones
12. H. E. Knee
13-17. R. C. Mann
18. S. Matis
19-20. R. J. Mural
21. E. M. Oblow
22. C. E. Oliver
23. S. Petrov
24. N. S. V. Rao
25. D. B. Reister
26. M. B. Shah
27-31. E. C. Uberbacher
32. R. C. Ward
33. X. Ying
34. EPMD Reports Office
35-36. Laboratory Records Department
37. Laboratory Records, ORNL-RC
38. Document Reference Section
39. Central Research Library
40. ORNL Patent Office

EXTERNAL DISTRIBUTION

41. Office of Assistant Manager for Energy Research and Development, Oak Ridge Operations, U.S. Department of Energy, P.O. Box 2008, Oak Ridge, TN 37831
42. Jim Decker, Director, Office of Energy Research, Dept. of Energy, Washington, DC 20585
43. David Smith, Health Effects & Life Sciences Research Division, Office of Health & Environmental Research, Dept. of Energy, Washington, DC 20585
44. B. J. Barnhart, Health Effects and Life Sciences Research Division, Office of Health and Environmental Research, Dept. of Energy, Washington, DC 20585
43. John Wooley, Health Effects and Life Sciences Research Division, Office of Health and Environmental Research, Dept. of Energy, Washington, DC 20585
44. Mark S. Boguski, National Institutes of Health NCBI/NLM, 8600 Rockville Pike, Bethesda, MD 20894
45. M. D. Zorn, Lawrence Berkeley Laboratory, MS 50B-3216, 1 Cyclotron Road, Berkeley, CA 94720
46. J. Fickett, Los Alamos National Laboratory, P.O. Box 1663, Los Alamos, NM 87545
47. Christian Burks, Los Alamos National Laboratory, Theorectical Biol. & Biophysics Grp., T-10, MS K710, Los Alamos, NM 87545
48. A. Jamie Cuticchia, John Hopkins Hosp./Welch Med. Library, 1830 East Monument Street, Baltimore, MD 21205-2100
49. E. Branscomb, Human Genome Center, Biomedical Sciences Division, Lawrence Livermore National Laboratory, Livermore, CA 94550
50. A. Lapedes, Center for Human Genome Studies, Los Alamos National Laboratory, P.O. Box 1663, Los Alamos, NM 87545
51. T. Hunkapiller, Division of Biology, California Institute of Technology, Pasadena, CA 91125
52. Radoje Drmanac, Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439

53. Chris Fields, The Institute For Genomic Research, 932 Clopper Road, Gaithersburg, MD 20878
54. Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831
55. Raymond F. Gesteland, University of Utah, Howard Hughes Medical Institute, 6160 Eccles Genetics Building, Salt Lake City, UT 84112
56. Steven Henikoff, Fred Hutchison Cancer Research Ctr., HHMI-FHCRC, 1124 Columbia Street, Seattle, WA 98104
57. Tim Hunkapiller, California Institute of Technology, 139-74, Pasadena, CA 91125
58. Thomas G. Marr, Cold Spring Harbor Laboratory, P.O. Box 100, Cold Spring Harbor, NY 11724
59. Sylvia J. Spengler, Lawrence Berkeley Laboratory, Human Genome Center, MS 1-213, 1 Cyclotron Road, Berkeley, CA 94720
60. Marvin Stodolsky, U.S. Department of Energy, Office of Health & Environmental Res., ER-72 GTN, Washington, DC 20545-0001
61. Jude Shavlik, Computer Science, University of Wisconsin, 1210 W. Dayton Street, Madison, WI 53706