A Fast Look-Up Algorithm for Detecting Repetitive DNA Sequences

X. Guan and E. C. Uberbacher
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6364

Repetitive sequences are abundant in human DNA sequences. By one estimate, from 30% to 50% human genome consists of repeats of one form or another (Benson and Waterman, 1994). In this paper, we focus on repetitive sequences in which short words are repeated many times, referred to as tandem repeats or microsatellites in the literature. We describe a linear time algorithm that can scan DNA sequences rapidly for tandem repeats. Recently, a new look-up technique, which uses indices calculated from non-contiguous, overlapping tuples, was employed in the FLASH program (Califano and Rigoutsos) for sequence homology search and is used here to recognize tandem repeats.

Our algorithm generates indices for each position of a sequence and uses these indices to identify tandem repeats. For each position, we look at a fixed window at that position, and select all possible ordered k elements from this window. These k elements are required to contain the first element of this window. For an example, suppose the sequence segment that falls within this window is TGTAT, and we want to generate all ordered 3 element combinations that contain the first element. There are 6 of them: TGT, TGA, TGT, TTA, TTT, and TAT. These k-tuples are non-contiguous (they are not necessarily formed with contiguous elements of the original sequence) and overlapping. Let $a_1 a_2 ... a_k$ be such a k-tuple and each a_i can take s different values. An index is calculated from this k-tuple:

$$I = \sum_{i=1}^{k} \lambda(a_i) * s^{i-1}$$

where λ is a function that maps a_i to an integer in $[0, s-1]$. Indices calculated from non-contiguous k-tuples allow insertions and deletions between matching segments, and overlapping k-tuples provide rich indices for each position in a sequence.

The new indexing technique is used here to recognize tandem repeats. we scan the sequence from left to right and calculate indices for each position as above. For each position i, we look for a position $j < i$ that shares the most indices with i (Two positions are said to share an index if this index is calculated from the same non-contiguous k-tuple for these two positions). We say that position i has one vote for position j. If there is a tandem repeat in the sequence, the first position of the first pattern in the tandem repeat will get the most votes, and the last position that votes for the first pattern marks the end of the tandem repeat.

Space is not as much a problem here as it is in FLASH. Since we are looking for tandem repeats of short patterns, for each position, we only need to look to the left of the position limited distance (look-backward-distance), and the table size is at most $\binom{w-1}{k-1} k^s$ ($\binom{n}{m}$ is a binomial coefficient, i.e., the number of ways to choose m ordered positions from n positions). For the parameters that we are using in our present system, window size $w = 5$, tuple size $k = 3$, $s = 4$, the table size is 486. Let f be the look-backward-distance. Our algorithm identifies tandem repeats with pattern sizes from 1 to f in one pass. In a comparison, a fixed pattern size is a required input parameter to the method by Benson and Waterman (1994). We scan the sequence only once, and for each position, only a constant number of indices are calculated and looked up in the table. As a result, our algorithm is a linear time algorithm.

To illustrate the use of the algorithm, we have applied it to the Human Tissue Plasminogen Activator Gene (GenBank Release 89, accession number K03021). The parameters we used were $w = 5$, $k = 3$. Two annotated tandem repeats, a $(RY)^n$ (RY repeated n times) run (7170-7225) and a $TGATAGA$ tandem repeat region (23888-24458), were identified. Two other un-annotated tandem repeats, a $(AC)^n$ run followed by a $(TC)^n$ run (16910-16961) and a poly(A) segment (17107-17132), were also identified by our algorithm. These four tandem repeats were reported by the Pythia server (Milosavljevic and Jurka, 1993). Regions identified by our algorithm but not reported by Pythia include several poly(A) segments (1015-1047, 26467-26486, 29083-29104, 19160-19178) and an interesting tandem repeat region (21562-21597):

AAATAATAATAATAAATAAATAATAAATAAATAAAT.

Pythia also uses a linear time algorithm, but in practice, our algorithm is 5-6 times faster (note, we compare our algorithm with only that portion of the Pythia server which identifies simple repeat regions).

The algorithm was implemented in C language, and has been incorporated into the GENQUEST (Guan *et al.*, 1993) and GRAIL Internet email and client-server systems. To access GENQUEST, send an email message to Q@ORNL.GOV.

References

Califano,A. and Rigoutsos, I. "FLASH: A Fast Look-up Algorithm for String Homology," *CABIOS.* To appear.

Benson, G. and Waterman, M.S. (1994) "A Method for Fast Database Search for all k-nucleotide Repeats," *Proc., Proc., The 2nd International Conference on Intelligent Systems for Molecular Biology*, AAAI Press.

Guan,X. Mural,R. Petrov,S. and Uberbacher,E.C. (1993) "A Sensitive Sequence Comparison Server for DNA and PROTEINS," *Proc., Genome Sequencing and Analysis Conference V*, Hilton Head Island, South Carolina.

Milosavljevic,A. and Jurka,J. (1993) "Discovering Simple DNA Sequences by the Algorithmic Significance Method," *CABIOS,* vol. 9, 407-411.