Nucleic Acids Research, 2003, Vol. 31, No. 23 ***6935–6941***
DOI: 10.1093/nar/gkg886

Mastering seeds for genomic size nucleotide BLAST searches

Valer Gotea[1], Vamsi Veeramachaneni[1] and Wojciech Makałowski[1,2,*]

[1]Institute of Molecular Evolutionary Genetics and Department of Biology and [2]Department of Computer Science and Engineering, The Pennsylvania State University, 514 Mueller Lab, University Park, PA 16802, USA

Received August 9, 2003; Revised September 10, 2003; Accepted October 8, 2003

ABSTRACT

One of the most common activities in bioinformatics is the search for similar sequences. These searches are usually carried out with the help of programs from the NCBI BLAST family. As the majority of searches are routinely performed with default parameters, a question that should be addressed is how reliable the results obtained using the default parameter values are, i.e. what fraction of potential matches have been retrieved by these searches. Our primary focus is on the initial hit parameter, also known as the seed or word, used by the NCBI BLASTn, MegaBLAST and other similar programs in searches for similar nucleotide sequences. We show that the use of default values for the initial hit parameter can have a big negative impact on the proportion of potentially similar sequences that are retrieved. We also show how the hit probability of different seeds varies with the minimum length and similarity of sequences desired to be retrieved and describe methods that help in determining appropriate seeds. The experimental results described in this paper illustrate situations in which these methods are most applicable and also show the relationship between the various BLAST parameters.

INTRODUCTION

Research in molecular biology and evolution has changed dramatically since the creation, in the early 1980s, of nucleotide and protein sequence databases. The extraordinary expansion of the Internet in the last decade has played a significant part in this transformation by allowing researchers easy access to these databases. Various programs have been developed and improved over the years for the purpose of finding similar sequences in sequence databases, an important discovery technique that has increasingly become one of the most valuable tools in biomedical research. These programs include FASTA (1), SIM (2), BLAST (3), WU-BLAST (http://blast.wustl.edu), Gapped BLAST and PSI-BLAST (4), BLAST 2 Sequences (5), MegaBLAST (6), MUMmer (7), QUASAR (8), REPuter (9), SENSEI (10), BLAT (11), PatternHunter (12) and BLASTZ (13).

The most frequently used, by far, is the NCBI BLAST family package, with up to 100 000 queries per day submitted through the NCBI web site, in spite of the emergence of new software like PatternHunter and BLASTZ. PatternHunter was reported to be able to run on desktop computers even with large data sets, due to low memory usage and shorter running times (12), but because it is a commercial package, it is not readily available to most users. BLASTZ, on the other hand, is optimized for finding a single best, orthologous match for the query sequence (13), of great help in aligning entire genomes (14). However, it dynamically masks regions participating in multiple hits, making it unsuitable for other applications, such as finding all duplicated segments within genomes.

As the Smith-Waterman algorithm is too slow for searching large databases, algorithms from the BLAST family use a much faster heuristic approach. They first find short contiguous or patterned (starting with MegaBLAST 2.2.4, but not in BLASTn) matches, also known as seeds or words, which are then extended into alignments. The first step is crucial for the quality of the search, since the number of extended alignments directly depends on the number of initial seeds. The seed size used by BLAST can be controlled to some extent by the user. If a small enough seed size is chosen, all potential alignments can be retrieved, but the running time could be very long. If, on the other hand, a large seed size is specified, the running time can be reduced, but the chance of missing alignments of interest increases.

In our work, we focus on the influence of seed size on genomic size DNA searches, while its influence on protein searches was analyzed by Korf (15) and some aspects of searching for protein coding regions in nucleotide databases were the focus of work by Anderson and Brass (16). Genomic size DNA searches are often carried out to detect orthologous regions between two different genomes or to detect segmental duplications within a genome. In such searches, the objective is to find all alignments that have a certain minimum length and similarity. However, the default seed values used by programs in the BLAST family are not appropriate for some frequently needed alignment length–similarity combinations (e.g. 1 kb length, 80% identity). We show that the use of default seeds in these cases can result in more than a quarter of alignments of interest not being retrieved. We describe

*To whom correspondence should be addressed. Tel: +1 814 865 5025; Fax: +1 814 865 9366; Email: wojtek@psu.edu

The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors

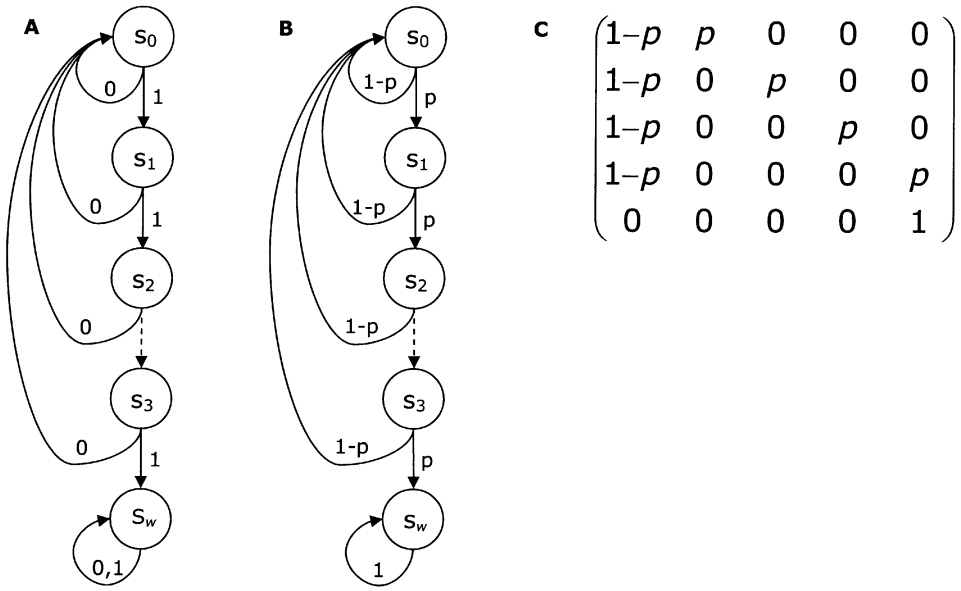

Figure 1. (**A**) Graphical representation of the deterministic finite automaton (DFA) that recognizes a contiguous seed of size w. (**B**) The zero order Markov chain associated with the DFA. Transition probabilities are given by frequencies of 0 $(1 - p)$ and 1 (p). Once the final state (s_w) is reached, the automaton remains in that state with probability 1. (**C**) The matrix representation of the Markov chain is used to compute the exact hit probability of a seed, which is given by the $(0,w)$ element of the matrix raised to the nth power.

methods by which researchers can determine optimal seeds, seeds that are long with the condition that they seed all alignments of interest. We suggest a simple enhancement to BLAST that allows such optimal seeds to be computed at run time and show how this enhancement can be implemented. We show that the spaced seeds used by MegaBLAST are sub-optimal and provide optimal spaced seeds. Finally, we describe experimental results that illustrate the relationship between the various BLAST parameters and describe when the model described in this paper is applicable.

MATERIALS AND METHODS

The underlying assumption in BLAST is that any alignment of interest will contain regions in which there are no mismatches between the query and subject sequences. These regions, called words in the BLAST nomenclature, are noted and extended into longer alignments only if their length is greater than (or equal to) the seed size parameter ('–W' for BLASTn and MegaBLAST). For example, the following alignment with identity 70% will be ignored by BLAST if a seed size greater than 4 is specified.

```
GTACTGCCGC
 ||||  |||  → 0111101110
TTACTCCCGT
```

Since all matches are treated equally in the comparison of nucleotide sequences, we can also view the alignment as a binary string in which ones represent matches and zeros represent mismatches. Consequently, we say that an alignment can be seeded by a seed of length w if w consecutive ones are present within the binary representation of the alignment.

At the genomic scale, due to the high percentage of non-coding sequence, mismatches can be assumed to be randomly distributed in the alignments of interest. Thus, a target alignment of length n and similarity p can be regarded as a binary string of n independent Bernoulli trials with $P(1) = p$ and $P(0) = 1 - p$. The probability of such a string containing w consecutive ones is called the hit probability of a seed of length w (12). Note that a trivial seed of length one will always have a hit probability of 1. However, since search time increases exponentially for smaller seeds, our goal, given the minimum length and similarity of the sequences to be retrieved, is to find the seed of maximum length that has a hit probability of 1 or very close to 1. For a contiguous seed of length w it is easy to see that (i) the hit probability increases with alignment length since a longer alignment is more likely to contain at least w consecutive matches and (ii) the hit probability increases with alignment similarity since an alignment with more matches is more likely to contain at least w consecutive matches. Therefore, if a seed is optimal for the retrieval of alignments of a particular length and similarity, it will also find all other longer alignments with higher similarities.

The exact hit probability of a seed for a particular alignment length and similarity pair can be computed using the deterministic finite automaton (DFA) approach (17). Figure 1A shows the $(w + 1)$ state automaton capable of detecting the presence of w consecutive ones in a binary alignment string. Every state s_i corresponds to i consecutive ones being found. By associating probability p with transitions labeled 1 (matches) and probability $(1 - p)$ with transitions labeled 0 (mismatches), we can derive the zero order Markov chain (Fig. 1B) and its associated $(w + 1) \times (w + 1)$ transition probability matrix M (Fig. 1C; matrix indices are 0 through w). The $[0,w]$ element of matrix M^n represents the probability of reaching state s_w after n alignment positions have been examined, and is the exact hit probability of the seed.

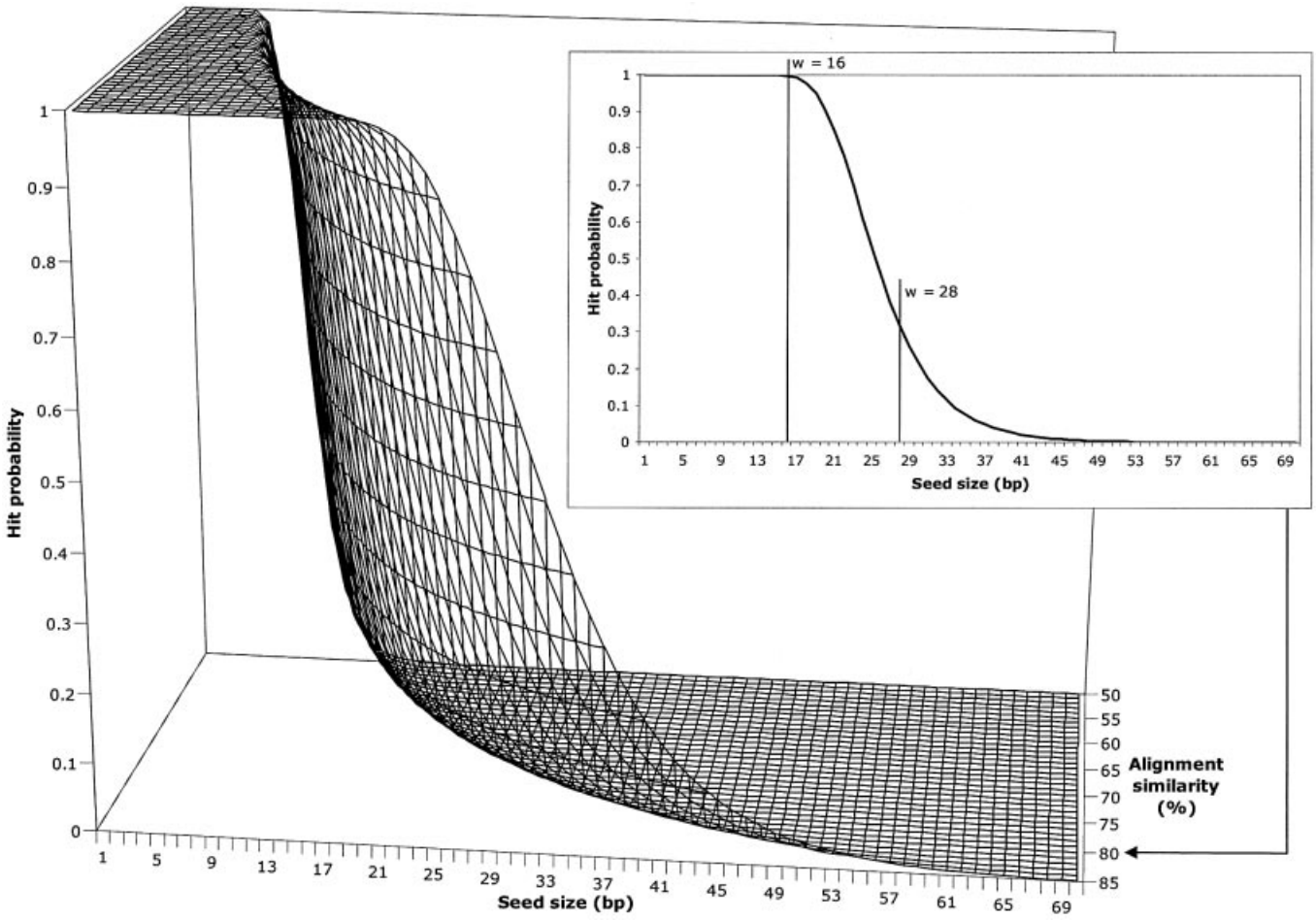

Figure 2. Variation of contiguous seed hit probability with alignment similarity (the interval between 50 and 85% similarity for an alignment of 1000 bp is shown). The section illustrates the hit probability for the case of an alignment similarity of 80%. The default MegaBLAST seed of size 28 is clearly not advisable in this case, as its hit probability is only 0.3176. A seed size of 16 would be more suitable, since its hit probability is 0.9979.

Our C implementation of the DFA method runs extremely fast for any practical seed size (source code available upon request). The computational time is of the order of $(\log_2 n) \times (w + 1)^3$, where n and w are the alignment length and seed length, respectively. The speed rapidly decreases for seed sizes larger than 100, but practical seed sizes are usually smaller than this value. The memory requirement is proportional to $(w + 1)^2$ and therefore is not a constraint.

To test the actual influence of the seed size on the number of hits retrieved, we used the 64 bit version of MegaBLAST v2.2.5 to search for duplicated segments at least 1000 bp long and at least 80% identical. We compared the sequence of human chromosome 13 with the whole genome (lower case letter masked version of November 2002 assembly from Golden Path). Soft masking (MegaBLAST options '–F m –U T') was used to avoid alignments consisting entirely of masked sequence. However, these options allow alignments to be extended into masked regions, because it is conceivable that segmental duplications include segments of repetitive elements. Also, MegaBLAST options '–p 80 –e 1' were used to eliminate some hits that were not of interest. The search was carried out with seed sizes of 16, 20, 24, 28, 32, 36 and 40. Alignments meeting our length and similarity criteria were extracted by post-processing the tabular output. The platform used was a Sun Fire V880 computer with 32 Gb of memory running Solaris 5.8.

Starting with version 2.2.4, the NCBI program MegaBLAST makes available a limited number of discontiguous or spaced seeds (http://www.ncbi.nlm.nih.gov/blast/discontiguous.html). A spaced seed consists of a set of matching positions (ones) with interspersed zeros that indicate positions where mismatches are allowed to occur. For example, the seed with template 110011 can start any alignment that contains one of the substrings 110011, 110111, 111011 or 111111. The weight (w) of a spaced seed is the number of matching positions (ones) in its template and the span (t) is the length of the template.

The exact hit probability of a spaced seed cannot be computed by using the DFA approach described earlier because of the overlapping positions that spaced seeds can inspect [fig. 1 in Buhler *et al.* (18)]. We, therefore, implemented the dynamic programming algorithm for exact hit probabilities of spaced seeds described by Keich *et al.* (http://www-cse.ucsd.edu/~ukeich/papers/seeds_a.pdf) in Perl. The running time of this program is proportional to $t \times 2^{(t-w)} \times n$ and it becomes prohibitive for long seeds ($t > 20$) with small

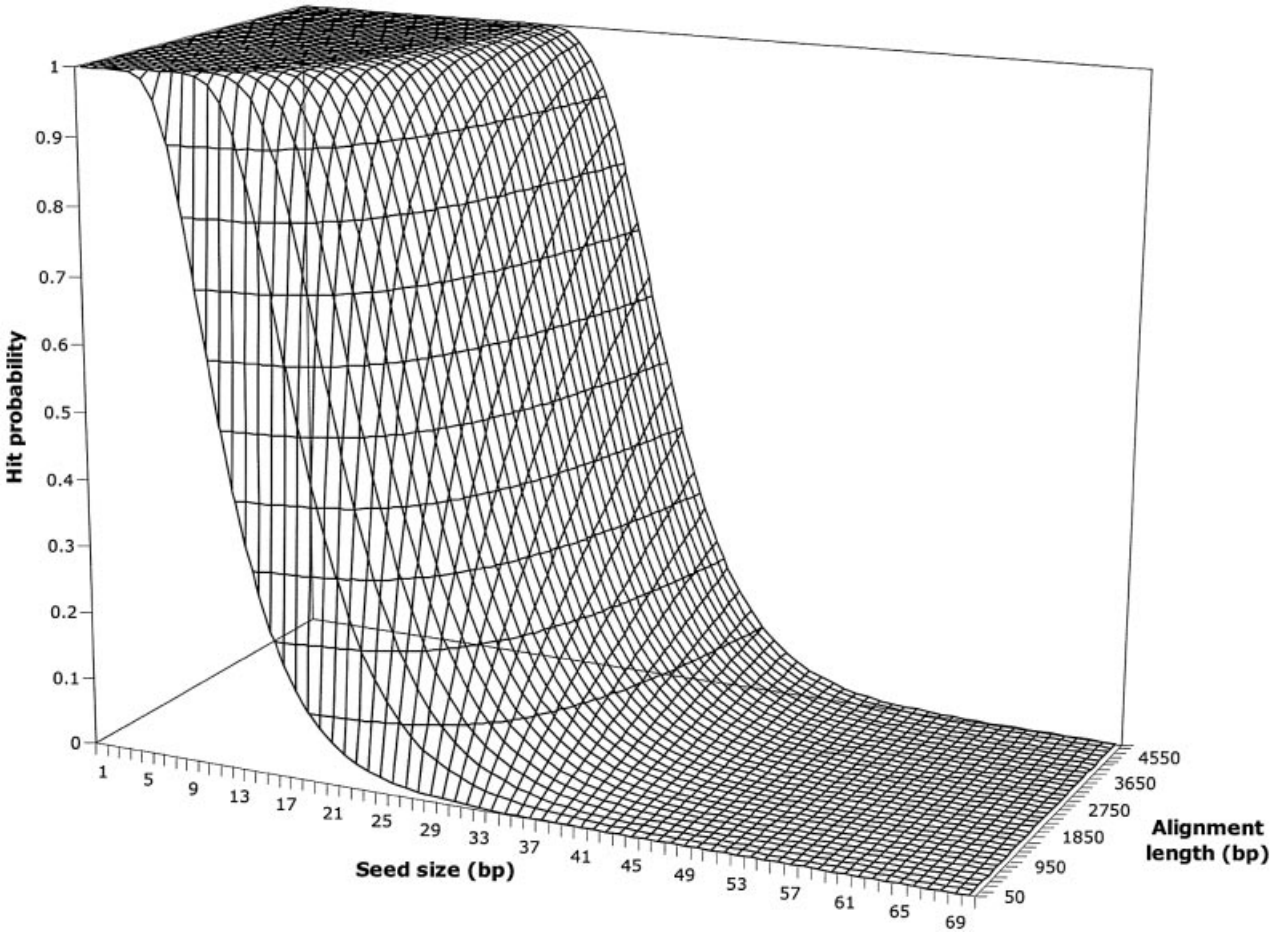

Figure 3. Variation of contiguous seed hit probability with alignment size (distributions for alignment sizes between 50 and 5000 bp for alignments of 80% similarity are shown).

weights. We used this program to compute the exact hit probabilities of the spaced seeds provided by the NCBI MegaBLAST and also to find the optimal template for each weight–span combination made available in MegaBLAST.

RESULTS

The hit probability of a seed depends on the length and similarity of the alignments that are desired. Different aspects of the relationship between these four entities are shown in Figures 2 and 3. Figure 2 shows that hit probability increases sharply with alignment similarity. On the other hand, Figure 3 shows that alignment length has a big influence only when it is small (a few hundred base pairs).

We also compared the results of the MegaBLAST searches conducted with different contiguous seeds. We verified that alignments found by using a particular seed are also detected using any smaller seed. For each alignment obtained by using word size 16, we used the length and similarity of the alignment to compute the probability that the alignment would be detected by a larger seed. Based on these computations, we calculated the expected number of hits that would be produced by using larger word sizes. The actual and expected numbers of hits are shown in Figure 4.

The hit probability of spaced seeds varies with alignment length and similarity in a manner similar to that of contiguous seeds. In general, a spaced seed always has a higher hit probability than the corresponding contiguous seed with the same weight. A more detailed analysis and comparison of contiguous and spaced seeds is presented in Ma *et al*. (12).

MegaBLAST (v2.2.4 onwards) provides two types of spaced seeds of different weights and spans: one designed for coding regions, where a mismatch is found most frequently at the third codon position, and the other designed for non-coding regions, where mismatches are assumed to occur randomly. We computed the exact hit probabilities for each of these seeds. For each weight–span combination, the MegaBLAST seed designed for non-coding regions had a higher hit probability than the seed designed for coding regions. This is to be expected since our hit probability model assumes that mismatches occur at random. However, the MegaBLAST seeds for non-coding regions are not optimal, i.e. in each case we were able to find spaced seeds with the same weight and span that had higher hit probabilities. The seeds used by MegaBLAST and the optimal seeds along with

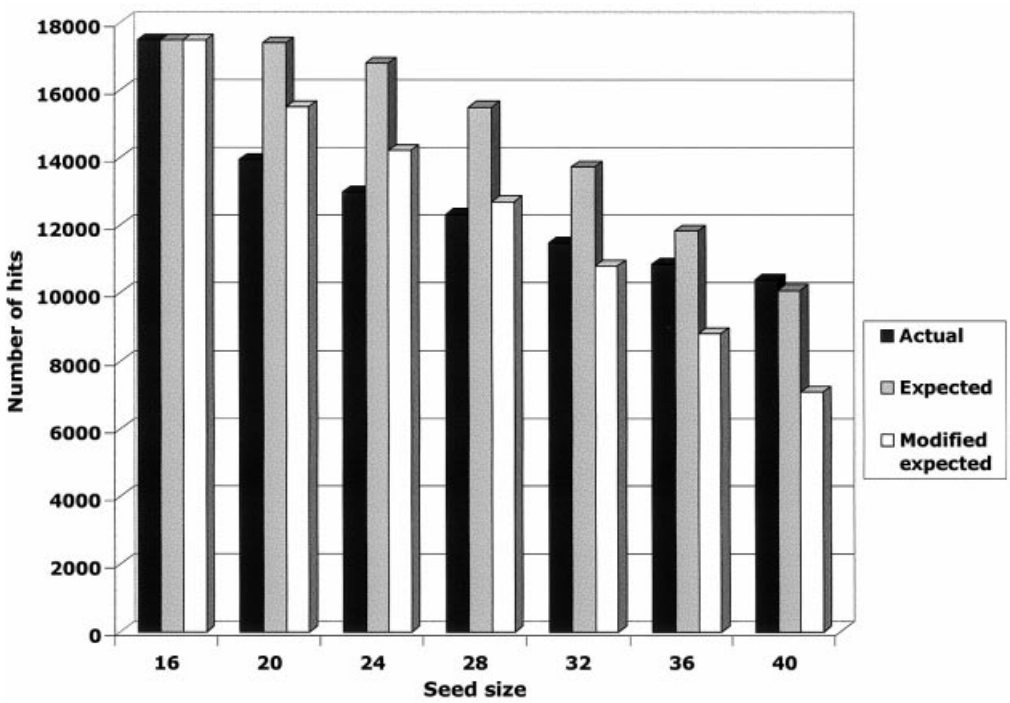

Figure 4. The actual number of hits that are generated by running MegaBLAST and the number of hits expected by computing theoretical hit probabilities are shown.

Table 1. The spaced seeds used by MegaBLAST are sub-optimal in that there exist other seeds with the same weight and span with higher hit probabilities

MegaBLAST parameters	MegaBLAST/optimal seed	Hit probability for alignment lengths and similarities			
		64 bp		200 bp	
		0.7	0.8	0.7	0.8
–W 11 –t 16 –N 1	1110010110110111	0.4612	0.8766	0.8964	0.9995
Optimal	1110111010010111	0.4655	0.8805	0.8995	0.9995
–W 12 –t 16 –N 1	1110110110110111	0.3391	0.7888	0.7809	0.9964
Optimal	1101110101101111	0.3493	0.8030	0.7937	0.9972
–W 11 –t 18 –N 1	11101001011001011	0.4625	0.8780	0.9029	0.9996
Optimal	11101100101001011	0.4671	0.8821	0.9062	0.9996
–W 12 –t 18 –N 1	11101011001011011	0.3531	0.8074	0.8062	0.9977
Optimal	11101100101101011	0.3564	0.8121	0.8104	0.9979
–W 11 –t 21 –N 1	11101001010001001011	0.4547	0.8729	0.9066	0.9996
Optimal	11100101100100101011	0.4604	0.8789	0.9110	0.9997
–W 12 –t 21 –N 1	11101001011001001011	0.3444	0.7985	0.8091	0.9978
Optimal	11101010010011010011	0.3525	0.8098	0.8189	0.9983

We show the templates of the seeds used by MegaBLAST, the templates of the optimal seeds and their hit probabilities for some alignment length–similarity combinations. As expected, the difference is most noticeable for small alignments with low similarity.

the hit probabilities for a few alignment length–similarity pairs are shown in Table 1.

DISCUSSION

MegaBLAST needs powerful hardware to handle long queries and databases, and an optimal seed can be critical in some situations. We successfully completed searches with chromosomes X (151 567 156 bp), Y (50 360 226 bp) and 8 (146 305 119 bp) through 22 (47 848 585 bp) sequences as individual query sequences against the whole human genome database (3 050 630 827 bp) on a Linux cluster with 1 Gb of memory. For chromosomes 1 (244 258 774 bp) through 7 (157 432 593 bp) we only succeeded on a Sun Fire V880 computer with 32 Gb of memory running Solaris 5.8. We used the 64 bit version of MegaBLAST 2.2.5, which was able to allocate more than 4 Gb of memory (the natural limit of 32-bit based computers). On machines with lower memory, an alternative is to split the query into smaller overlapping pieces [overlaps of 10 kb are considered sufficient (13)], run searches independently and assemble the final results.

As MegaBLAST memory requirements depend on the seed size (the bigger the seed, the less memory and time required), it is critically important to find an optimal seed, especially when the hardware available has limited resources and the search time is limited. However, the optimal seed depends to a

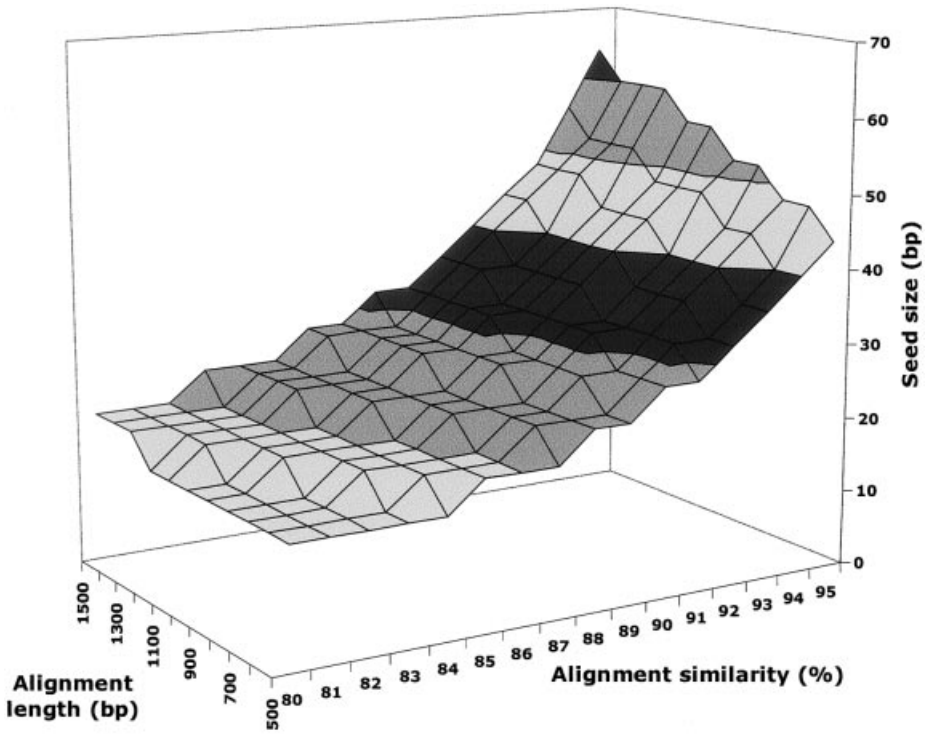

Figure 5. The seed size shown is the largest seed with hit probability >0.95 for that particular alignment length and similarity. Thus, using seeds recommended by this figure theoretically ensures that 95% of all alignments of interest will be seeded.

Table 2. Recommended program and option choices for genomic size searches, given a small number of target alignment lengths and similarities

Target alignment similarity (%)	Target alignment length	Recommended program	Recommended options	Seed hit probability
95	1000	MegaBLAST	–W 44	0.999
	10 000	MegaBLAST	–W 84	0.999
90	1000	MegaBLAST	–W 28	0.997
	10 000	MegaBLAST	–W 44	0.999
80	1000	MegaBLAST	–W 16	0.997
	10 000	MegaBLAST	–W 24	0.999
70	1000	MegaBLAST	–W 12 –t 21 –N 1	0.999
	10 000	MegaBLAST	–W 16	0.999
60	1000	BLASTn	–W 8	0.999
	10 000	MegaBLAST	–W 12 –t 21 –N 1	0.999
50	1000	BLASTn	–W 7	0.982
	10 000	BLASTn	–W 9	0.999

Note that seed size can be considerably increased for long and highly similar alignments.

large extent on the length and similarity of the desired alignments. For instance, if the goal is to retrieve all alignments that are at least 1000 bp long and at least 80% identical then a seed of size 16 with hit probability 0.997 would be more appropriate than the default seed of size 28 with hit probability of only 0.3176. Thus, once the alignment length and similarity criteria are set, one can use the programs described in Materials and Methods to compute the optimal seed size.

Figure 5 provides recommended seeds for a variety of alignment length–similarity combinations. Note that we only show seed sizes that are multiples of 4 since current versions of MegaBLAST round the user-specified seed size to the closest multiple of 4. The seed size at any mesh point in the figure is the largest seed with hit probability >0.95 for that particular alignment length–similarity pair. Thus, using seeds according to this figure ensures (theoretically) that at least 95% of all desired alignments will be seeded. Figures for other hit probabilities can be constructed in a similar manner. In addition to Figure 5, we compiled a list of recommendations for a small number of scenarios (desired alignment length and similarity), presented in Table 2. These can both be used as guidelines for designing appropriate seeds for genomic size searches. From the end user perspective, it would be easier and more helpful if BLAST could simply look up the appropriate seed size from the tabular data represented by such figures given the desired alignment length and similarity as online parameters.

It is important to remember that MegaBLAST is optimized for finding alignments with high similarity. When used with word size 16 (and higher) it can be up to 10 times faster than the more traditional BLASTn program. The minimum word

size permitted by MegaBLAST is 8. However, it is our experience that with seed sizes smaller than 16 its speed is inferior to that of BLASTn for large scale comparisons (W.Makałowski, unpublished data).

In terms of spaced seeds, MegaBLAST provides only a limited choice at present. Our comparison of the sequence of human chromosome 13 with the entire human genome could not be carried out with seeds with weight less than 16 due to memory constraints (more than 25 Gb). We, therefore, could not evaluate the performance of MegaBLAST spaced seeds (all of which have weight at most 12) in this particular case. However, experiments with other data sets suggest that, hit probabilities being approximately equal, it is better, both in terms of speed and memory usage, to use the contiguous seed of size 16 rather than the longest and heaviest spaced seed ('–W 12 –t 21 –N 1'). Also, as mentioned in Results, the spaced seeds used by MegaBLAST are sub-optimal in the sense that there are other seeds with the same weight and span with higher hit probabilities. While the difference is not significant for alignments longer than 500 bp or for alignments with similarity greater than 80%, it might still be better to replace the current templates with those shown in Table 1.

While using hit probabilities as the basis for selecting seeds, one should be mindful of the underlying assumption of the hit probability model, namely that mismatches occur at random. Overlooking this assumption can lead to results which are, at first sight, surprising. For instance, we used MegaBLAST to compare the sequence of the *Fugu* genome against itself to detect segmental duplications. We conducted the search with several different seed sizes but the number of the alignments of length 1000 bp and identity 80% remained more or less constant. This is in large part due to the compact nature of the *Fugu* genome (1/10 of the human genome size) which contains mainly coding regions. Interspersed repeats cover only 2.7% of the genome, as compared to 35–50% or more in mammals, including human (19,20). Thus, the mismatches that occur in alignments are non-random and the theoretical hit probabilities do not apply.

One should also realize that the behavior of BLAST is controlled not by the word size alone; the choice of scoring parameters, the treatment of gaps and the filtering options that are selected can all affect the results. This fact is reflected in our experiments involving the comparison of human chromosome 13 to the entire human genome. We looked into the reason behind the disparity between the expected and actual number of hits shown in Figure 4. One of the main reasons for the discrepancy is the use of the soft masking approach. In nearly 2000 of the 17 480 hits found with seed size 16, the query sequence was entirely lower case with a small stretch (<20 bp in length) of upper case letters. Since our MegaBLAST filtering options prevent these alignments from being seeded when word size 20 is used, we computed a 'modified expected' number of hits (Fig. 4) for word size 20, not taking into consideration those alignments. 'Modified expected' values for the other word sizes (24–40) were calculated in a similar manner and they are a good approximation of the actual number of hits for seed sizes less than 32. When larger seed sizes are used, a higher percentage of alignments that are detected have a higher number of gaps, and this accounts for the increasing disparity between the expected and actual number of hits. For instance, the median number of gaps in the alignments found with word size 40 is three times the number of gaps in the alignments found by using word size 16. Since our hit probability model assumes that all alignments are ungapped, it under-predicts the number of hits for large word sizes.

The above case analysis shows the relationship between the various parameters used by BLAST. In this paper, our main focus has been the seed parameter, since it is the starting point for any alignment. However, as we have shown in our experimental results, researchers should be aware of the assumptions underlying the hit probability model and take into consideration the relationship between the different parameters when designing seeds for their searches.

REFERENCES

1. Lipman,D.J. and Pearson,W.R. (1985) Rapid and sensitive protein similarity searches. *Science*, **227**, 1435–1441.
2. Huang,X. and Miller,W. (1991) A time-efficient, linear-space similarity algorithm. *Adv. Appl. Math.*, **12**, 337–357.
3. Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.
4. Altschul,S.F., Madden,T.L., Schaffer,A.A., Zhang,J., Zhang,Z., Miller,W. and Lipman,D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.
5. Tatusova,T.A. and Madden,T.L. (1999) BLAST 2 Sequences, a new tool for comparing protein and nucleotide sequences. *FEMS Microbiol. Lett.*, **174**, 247–250.
6. Zhang,Z., Schwartz,S., Wagner,L. and Miller,W. (2000) A greedy algorithm for aligning DNA sequences. *J. Comput. Biol.*, **7**, 203–214.
7. Delcher,A.L., Kasif,S., Fleischmann,R.D., Peterson,J., White,O. and Salzberg,S.L. (1999) Alignment of whole genomes. *Nucleic Acids Res.*, **27**, 2369–2376.
8. Burkhardt,S., Crauser,A., Lenhof,H.-P., Rivals,E., Ferragina,P. and Vingron,M. (1999) Q-gram based database searching using a suffix array. Paper presented at the Third Annual International Conference on Computational Molecular Biology, Lyon, France.
9. Kurtz,S. and Schleiermacher,C. (1999) REPuter: fast computation of maximal repeats in complete genomes. *Bioinformatics*, **15**, 426–427.
10. States,D.J. and Agarwal,P. (1996) Compact encoding strategies for DNA sequence similarity search. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, **4**, 211–217.
11. Kent,W.J. (2002) BLAT—the BLAST-like alignment tool. *Genome Res.*, **12**, 656–664.
12. Ma,B., Tromp,J. and Li,M. (2002) PatternHunter: faster and more sensitive homology search. *Bioinformatics*, **18**, 440–445.
13. Schwartz,S., Kent,W.J., Smit,A., Zhang,Z., Baertsch,R., Hardisson,R.C., Haussler,D. and Miller,W. (2003) Human-mouse alignments with BLASTZ. *Genome Res.*, **13**, 103–107.
14. Waterston,R.H., Lindblad-Toh,K., Birney,E., Rogers,J., Abril,J.F., Agarwal,P., Agarwala,R., Ainscough,R., Alexandersson,M., An,P. *et al.* (2002) Initial sequencing and comparative analysis of the mouse genome. *Nature*, **420**, 520–562.
15. Korf,I. (2003) Serial BLAST searching. *Bioinformatics*, **19**, 1492–1496.
16. Anderson,I. and Brass,A. (1998) Searching DNA databases for similarities to DNA sequences: when is a match significant? *Bioinformatics*, **14**, 349–356.
17. Nicodeme,P., Salvy,B. and Flajolet,P. (1999) Motif statistics. In *Proceedings of the 7th Annual European Symposium on Algorithms*, Prague, Czech Republic.
18. Buhler,J., Keich,U. and Sun,Y. (2003) Designing seeds for similarity search in genomic DNA. Paper presented at the Seventh Annual International Conference on Research in Computational Molecular Biology, Berlin, Germany.
19. Aparicio,S., Chapman,J., Stupka,E., Putnam,N., Chia,J.M., Dehal,P., Christoffels,A., Rash,S., Hoon,S., Smit,A. *et al.* (2002) Whole-genome shotgun assembly and analysis of the genome of *Fugu rubripes*. *Science*, **297**, 1301–1310.
20. Makałowski,W. (2001) The human genome structure and organization. *Acta Biochim. Pol.*, **48**, 587–598.