BIOINFORMATICS APPLICATIONS NOTE Vol. 27 no. 17 2011, pages 2433–2434
doi:10.1093/bioinformatics/btr368

Sequence analysis

Advance Access publication June 20, 2011

SpEED: fast computation of sensitive spaced seeds

Lucian Ilie[1,*], Silvana Ilie[2] and Anahita Mansouri Bigvand[1]

[1]Department of Computer Science, University of Western Ontario, London, ON N6A 5B7 and [2]Department of Mathematics, Ryerson University, Toronto, ON M5B 2K3, Canada

Associate Editor: John Quackenbush

ABSTRACT

Summary: Multiple spaced seeds represent the current state-of-the-art for similarity search in bioinformatics, with applications in various areas such as sequence alignment, read mapping, oligonucleotide design, etc. We present SpEED, a software program that computes highly sensitive multiple spaced seeds. SpEED can be several orders of magnitude faster and computes better seeds than the existing leading software programs.

Availability: The source code of SpEED is freely available at www.csd.uwo.ca/~ilie/SpEED/

Contact: ilie@csd.uwo.ca

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on April 6, 2011; revised on June 9, 2011; accepted on June 10, 2011

1 INTRODUCTION

Approximate string searching is at the core of many fundamental algorithms in bioinformatics. Since the quadratic-time dynamic programming algorithm of Smith–Waterman is too slow, heuristic methods have to be used. The most popular of those is implemented in BLAST (Altschul *et al.*, 1990) and uses the hit-and-extend approach: 11 consecutive matches are considered a hit and the neighborhood is investigated further for potentially local similarity. Such a match is called a *seed*. It has been noticed by Califano and Rigoutsos (1993) that requiring the matches to be non-consecutive increases the chance of finding similarities, and Ma *et al.* (2002) introduced the idea of optimal *spaced* seeds, that is, seeds where the matches are distributed so as to maximize the sensitivity (i.e. probability to find a local similarity). Multiple spaced seeds (Li *et al.*, 2004) go much further and approach perfect sensitivity. Since then multiple spaced seeds have been used in many software programs in a variety of applications, such as sequence alignment (Li *et al.*, 2004; Ma *et al.*, 2002; Noe and Kucherov, 2005) read mapping (David *et al.*, 2011; Homer *et al.*, 2009), oligonucleotide design (Feng and Tillier, 2007), to name a few.

It is therefore very important to compute seeds with very high sensitivity. The relevant problems are hard (Ma and Li, 2007) and all heuristic algorithms for computing seeds require exponential time with the exception of the one of Ilie and Ilie (2007). Our goal in this note is to engineer this algorithm into an efficient software for computing multiple spaced seeds. The new program, SpEED, has two execution modes, fast and best. We have compared SpEED with the two leading software programs, Mandala (Buhler *et al.*, 2005) and Iedera (Kucherov *et al.*, 2006) for a variety of seeds used in practice. SpEED-best computes always the best seeds with SpEED-fast coming often in the second place, whereas SpEED-fast is three to five orders of magnitude faster than all the other ones.

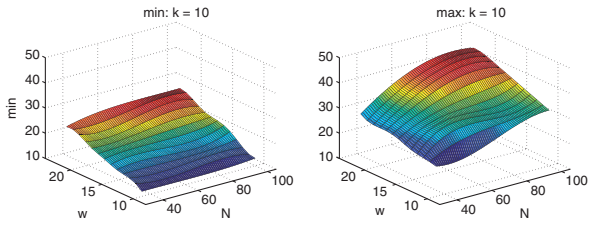

Fig. 1. Min and max values ($k=10$) by Matlab's cubic spline interpolation.

2 METHODS

A *spaced seed* s is a string over the alphabet $\{1,*\}$, where 1 is a match and $*$ a don't care. The number of 1's in s is called the *weight* of s. A *multiple spaced seed* is a set of seeds $S=\{s_1,s_2,\ldots,s_k\}$. In the Bernoulli model (Li *et al.*, 2004), an alignment is represented as a (random) sequence R of 1's and 0's (matches and mismatches) where the probability p of a match is called *similarity*. The length of R will be denoted by N. A seed s *hits* R if aligning s with R at some position causes each 1 in s to be aligned with a 1 in R. A multiple seed S *hits* R if there exist $s\in S$ so that s hits R. The *sensitivity* of S is the probability that S hits R. It depends on S, p, and N. A dynamic programming (exponential) algorithm is given by Li *et al.* (2004) that computes the sensitivity of a given (multiple) seed.

Finding optimal seeds by exhaustive search is feasible only for single seeds. The only polynomial-time heuristic algorithm (Ilie and Ilie, 2007) uses *overlap complexity*, a polynomial-time computable measure which is very well correlated with sensitivity. To avoid testing all possible seeds, the algorithm successively improves a fixed starting seed based solely on overlap complexity.

The algorithm of Ilie and Ilie (2007) has the drawback of requiring the lengths of the seeds as input parameters. We have addressed this issue in SpEED by computing a number of good length sets and then interpolating those in order to produce good lengths for a wide range of parameters. To make the preprocessing feasible, we have developed an algorithm that computes all seed lengths from the minimum and maximum ones. (See Supplementary Material for details.) These values for $k=10$ are shown in Figure 1. SpEED-first is the multiple seed computed using these lengths. SpEED-best is the most sensitive seed obtained after a number of restarts with random lengths and seeds between the given min and max.

3 RESULTS

We have compared SpEED with the two leading software programs, Mandala (Buhler *et al.*, 2005) and Iedera (Kucherov *et al.*, 2006)

*To whom correspondence should be addressed.

Table 1. Sensitivity comparison (darker colour indicates higher sensitivity)

W	p	Original seeds	Mandala	Iedera	SpEED	
					First	Best
SHRiMP: 4 seeds ($N=50$)						
	0.75	89.6113	90.6608	90.6802	90.6835	90.9098
10	0.80	97.3159	97.7316	97.7586	97.7436	97.8337
	0.85	99.6613	99.7283	99.7437	99.7414	99.7569
Time (s)			261	2706	**0.06**	1300
	0.75	81.6772	83.0512	83.2413	83.1190	83.3793
11	0.80	94.1141	94.7845	94.9350	94.8619	94.9861
	0.85	99.0145	99.1929	99.2189	99.2079	99.2431
Time (s)			714	5355	**0.07**	1658
	0.80	89.3037	90.2580	90.3934	90.3786	90.5750
12	0.85	97.7253	98.0786	98.0781	98.1023	98.1589
	0.90	99.8330	99.8633	99.8773	99.8777	99.8821
Time (s)			2092	10 476	**0.1**	2772
	0.85	84.0995	84.3838	84.5795	84.5476	84.8212
16	0.90	97.1676	97.3023	97.2806	97.3299	97.4321
	0.95	99.9260	99.9287	99.9331	99.9338	99.9388
Time (s)			10 218	10 220	**0.3**	2279
	0.85	71.1961	72.1954	72.1695	72.8024	73.1664
18	0.90	92.5652	93.0855	93.0442	93.5595	93.7120
	0.95	99.6299	99.6603	99.6690	99.7372	99.7500
Time (s)			1604	5432	**0.6**	31 374
PatternHunter II: 16 seeds ($N=64$)						
	0.70	92.4114	92.3811	92.0708	92.9759	93.2526
11	0.75	98.4289	98.4320	98.3391	98.5971	98.6882
	0.80	99.8449	99.8448	99.8366	99.8675	99.8820
Time (s)			37 806	12 326	**13.1**	44 651
BFAST: 10 seeds ($N=50$)						
	0.85	58.6907	—	60.1535	60.3534	60.8127
22	0.90	87.3359	—	87.9894	88.3817	88.5969
	0.95	99.2249	—	99.2196	99.3524	99.3659
Time (s)			>1 day	62 683	**18.8**	29 298

Seeds were computed by Mandala, Iedera and SpEED (fast and best) with the same parameters as those of SHRiMP, PatternHunter II and BFAST. The number of iterations was: 100 000 for Iedera and 10 for Mandala (both according to the authors' instructions) and 5000 for SpEED-best. The maximum time allowed for all programs was 1 day; we give the times until the best seed was obtained.

for computing seeds with some of the most challenging parameters used in practice: SHRiMP (David *et al.*, 2011), PatternHunterII (Li *et al.*, 2004) and BFAST (Homer *et al.*, 2009). Table 1 gives the sensitivities, all of which were computed using the algorithm of Li *et al.* (2004). SpEED-best is the best in all cases considered with SpEED-fast coming often in second place. SpEED-first (times in bold) is three to five orders of magnitude faster than all the others, since it is polynomial-time. The improvement of the original seeds is significant. (A 1% improvement in sensitivity implies that, for 100x coverage of the human genome, an additional 3 billion nucleotides could be mapped by using the better seed.)

An experimental evaluation of the SpEED seeds is given in Table 2 by comparison with the optimal for three cases, where the parameters have been chosen such that exhaustive search is feasible and the sensitivities are in the low 80 s to maximize the difference to the optimal. The SpEED-best sensitivities are very close or even the same as the optimal.

Table 2. Comparison between the sensitivity of the SpEED seeds and the optimal ones for three sets of parameters

k	w	N	p	$\ell_1,\ldots,\ell_k$	Optimal seeds	SpEED	
						First	Best
2	14	35	0.88	17, 21	0.829 910	0.827 472	0.828 856
3	10	35	0.78	12, 14, 16	0.818 325	0.813 690	0.818 325
4	6	35	0.60	8, 9, 10, 11	0.850 258	0.845 523	0.848 790

4 CONCLUSION AND FUTURE RESEARCH

SpEED-first is suitable for on-the-fly computation of seeds, whereas SpEED-best computes the best seeds, apparently very close to the optimal. Further development includes modifying the overlap complexity so that seeds of different lengths can be compared as well as addressing models different than Bernoulli, such as Markov.

ACKNOWLEDGEMENTS

We would like to thank an anonymous referee for pointing out a case when our software failed to work.

Funding: Natural Sciences and Engineering Research Council of Canada (NSERC to S.I. and L.I.).

Conflict of Interest: none declared.

REFERENCES

Altschul,S.F. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Buhler,J. *et al.* (2005) Designing seeds for similarity search in genomic DNA. *J. Comput. Syst. Sci.*, **70**, 342–363.

Califano,A. and Rigoutsos,I. (1993) FLASH: fast look-up algorithm for string homology. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Rec.*, pp. 353–359.

David,M. *et al.* (2011) SHRiMP2: sensitive yet practical short read mapping. *Bioinformatics*, **27**, 1011–1012.

Feng,S. and Tillier,E. (2007) A fast and flexible approach to oligonucleotide probe design for genomes and gene families. *Bioinformatics*, **23**, 1195–1202.

Homer,N. *et al.* (2009) BFAST: An alignment tool for large scale genome resequencing. *PLoS ONE*, **4**, e7767.

Ilie,L. and Ilie,S. (2007) Multiple spaced seeds for homology search. *Bioinformatics*, **23**, 2969–2977.

Kucherov,G. *et al.* (2006) A unifying framework for seed sensitivity and its application to subset seeds. *J. Bioinform. Comput. Biol.*, **4**, 553–569.

Li,M. *et al.* (2004) Pattern-HunterII: highly sensitive and fast homology search. *J. Bioinform. Comput. Biol.*, **2**, 417–440.

Ma,B. and Li,M. (2007) On the complexity of the spaced seeds. *J. Comput. Syst. Sci.*, **73**, 1024–1034.

Ma,B. *et al.* (2002) PatternHunter: faster and more sensitive homology search. *Bioinformatics*, **18**, 440–445.

Noé,L. and Kucherov,G. (2005) YASS: enhancing the sensitivity of DNA similarity search. *Nucleic Acids Res.*, **33**, W540–W543.