Theoretical Computer Science 230 (2000) 39–48

Theoretical Computer Science

www.elsevier.com/locate/tcs

Topology of strings: Median string is NP-complete

C. de la Higuera[a,*], F. Casacuberta[b,1]

[a] *Département d'Informatique Fondamentale (DIF), LIRMM, 161 rue Ada, 34392 Montpellier Cedex 5, France*

[b] *Departemento Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, 46071 Valencia, Spain*

Received April 1996; revised March 1997

Communicated by M. Crochemore

Abstract

Given a set of strings, the problem of finding a string that minimises its distance to the set is directly related with problems frequently encountered in areas involving Pattern recognition or computational biology. Based on the Levenshtein (or edit) distance, different definitions of distances between a string and a set of strings can be adopted. In particular, if this definition is the sum of the distances to each string of the set, the string that minimises this distance is the (generalised) median string. Finding this string corresponds in speech recognition to giving a model for a set of acoustic sequences, and in computational biology to constructing an optimal evolutionary tree when the given phylogeny is a star. Only efficient algorithms are known for finding approximate solutions. The results in this paper are combinatorial and negative. We prove that computing the median string corresponds to a NP-complete decision problems, thus proving that this problem is NP-hard. © 2000 Elsevier Science B.V. All rights reserved

Keywords: NP-complete problems; String-to-string correction; Levenshtein distance; generalised) Median; Multiple sequence alignment

1. Introduction

A well-known distance between strings (or sequences) is the Levenshtein distance (or edit distance) which corresponds to the smallest number of deletions, insertions and substitutions needed to transform one string (or sequence) into another. This very natural distance can be adapted to the case where some operations are more likely than others, and is thus given by a score scheme which is a matrix indexed by the symbols in the alphabet, to which is added a special symbol indicating "no symbol". We will

* Corresponding author. E-mail: cdlh@lirmm.fr.

[1] Work partially supported by the Spanish CICYT under grant TIC95-0884-C04-01.

work throughout this paper with the easiest score scheme: every operation has cost 1: this does not allow for generality (some schemes can be "easier"), but does entail that our results are independent of the score scheme in the sense that we will not allow the score scheme to be modified unnaturally to obtain negative results. This distance has been thoroughly studied (see [2] for instance) and is used in many cases because, in contrast with other natural distances between strings it is polynomially computable: Wagner and Fisher [15] give a dynamic programming algorithm that computes this distance in $O(|u| \cdot |v|)$ time, where u and v are the two strings involved. A survey of the different algorithms computing the Levenshtein distance can be found in [11, 13, 14].

The topological question we study in this paper is the existence and the computation of a (generalised) median, or some sort of a centre string for a given set of strings. This one can be a string that minimises the sum of the distances to each string of the set, which we will call the *median string*, but other topological measures are possible, by adding weights to the strings or looking for a string that minimises the maximum of the distances instead of the sum. One of these approximations corresponds to the search of a *set median*, that is when the search is constrained to the given input set. This constrained problem can be solved in polynomial time, and has been addressed in operational research for networks [8].

In pattern recognition, a number of techniques have been developed which are based on searching a string or a small set of strings that adequately model a (large) set of garblled strings or patterns. Examples of such techniques are the cluster methods for syntactic patterns (for the set median) [5] and (generalised) median strings [9]. In speech recognition, heuristics are currently used to obtain approximations to the median string as a model of some lexical or sub-lexical unit from a set of acoustic sequences corresponding to utterances of such units [1, 9].

In biology, the comparison of molecules, and more recently of genetic sequences has issued an important literature concerned with the problem of multiple alignment [4, 10, 12]. In a broad sense, multiple alignment can be defined as to find a set of patterns that, with some distance restrictions, appear in the same order in all the sequences of a given set. One variant of the problem is known as "multiple tree alignment with a given phylogeny" and can be defined as follows:

Given a set X of n sequences over some alphabet Σ and an unlabelled tree (the phylogeny) with n leaves, an evolutionary tree is a labelled tree built on the phylogeny, where the labels of the leaves are exactly the sequences in set X, and the labels of the other nodes are sequences over the alphabet Σ. The cost of an evolutionary tree is the sum of all the edit distances between the labels of pairs of nodes joined by a vertex. An optimal evolutionary tree is one that minimises the cost. One classical phylogeny is the star: a tree with $n+1$ nodes, n of them being leaves. Clearly, finding an optimal evolutionary tree when the given phylogeny is a star is exactly the same problem as finding the median string. In [16] it is proved that finding an optimal evolutionary tree is NP-hard, when the given phylogeny is a binary tree or a star, but the score scheme the authors use is problem dependent and does not even satisfy triangular inequality. Other results are proven in [17]: if the phylogeny has bounded degree then although the problem is NP-hard, it does admit a polynomial time approximation scheme (PTAS).

A third related problem deals with indexing a set of sequences in a database. Fast retrieval programs are required and to do this it is proposed to index the database by special reference sequences [6]. Retrieval then consists in comparing the new query sequence with the sequences in the index, computing the distance, and only searching the database when the distance is smaller than some bound. Ideally, the index sequence should be the centre string, i.e. the string that minimises the maximum of distances from each string to the index string.

In this paper we investigate, under a combinatorial point of view, the problem of finding the median of a set of strings, and prove that the associated decision problem is NP-complete, thus making the search problem intractable. A similar result is given for centre strings.

The rest of the paper is organised as follows: in the next section basic definitions are reviewed. In Section 3 we first deal with a purely technical result, then give the appropriate reductions for the median and centre string problems.

2. Preliminaries

Let $[n]$ denote the set $\{i \in \mathbb{N} \mid 1 \leqslant i \leqslant n\}$. Thus $[0]$ denotes the empty set. An alphabet is a non-empty set of symbols. Although an alphabet of size one can present interesting problems in some cases, this is not so for elaborate distances over sequences. So, in the sequel, we will always suppose that the alphabet (usually denoted Σ) contains at least two symbols. For a given alphabet Σ, the set of all finite-length strings of symbols from Σ is denoted Σ^*. The empty string is denoted λ. For a string w, $|w|$ denotes the length of w. Given two strings u and v over Σ, u is a subsequence of v if and only if u can be obtained from v through the deletion of certain occurrences of symbols.

Given two strings w and w' in Σ^*, w rewrites into w' in one step if one of the following correction rules holds:

- $w = uxv,\ w' = uv$ and $u, v \in \Sigma^*,\ x \in \Sigma$ (single-symbol deletion).
- $w = uv,\ w' = uxv$ and $u, v \in \Sigma^*,\ x \in \Sigma$ (single-symbol insertion).
- $w = uxv,\ w' = uyv$ and $u, v \in \Sigma^*,\ x, y \in \Sigma,\ x \neq y$ (single-symbol substitution).

We will consider the reflexive and transitive closure of this derivation, and denote $w \xrightarrow{k} w'$ if and only if w rewrites into w' by k operations of single-symbol deletion, single-symbol insertion and single-symbol substitution.

Definitions

- Given 2 strings w and w', the *Levenshtein distance* between w and w' denoted $d(w, w')$ is the smallest k such that $w \xrightarrow{k} w'$.
- The *global distance* between a string w and a finite set of strings W over an alphabet Σ, $(D(w, W))$ is

$$D(w, W) = \sum_{w' \in W} d(w, w').$$

- Given a finite set of strings W over an alphabet Σ, a string m over Σ is a *median* of W if and only if given any string w over Σ, $D(m,W) \leqslant D(w,W)$.

Example. $d(abaa,aab)=2$. *abaa* rewrites into *aab* via (for instance) a deletion of the *b* and a substitution of the last *a* by a *b*.
Let $W=\{aab,bb,abab\}$ and $w=ab$. Then $D(w,W)=4$, since $d(w,aab)=1$, $d(w,bb)$ $=1$ and $d(w,abab)=2$. The string *abb* is a median of W. Notice that the median is not unique, *aab* and *abab* are alternative medians.

If the string minimising the global distance is constrained to belong to W, it is known as a set median. As the computation of the distance between any two strings of the set W can take place in polynomial time, finding the set median is a relatively easy problem. This is also studied in operational research [8]. Kohonen [9] proposes the following heuristic to approximate the median string: compute the set median, and by local corrections (computing each time $D(w,W)$) try to improve it. In [3] another approximation is proposed based on a greedy strategy to build incrementally a solution. Other heuristics have been proposed but it remains unclear whether the problem admits a PTAS in [16] Wang and Jiang prove that the answer is negative when the score scheme is a parameter of the problem. In their proof they use a score scheme that does not satisfy triangular inequality.

We end this section with an easy lemma that will be used in the sequel.

Lemma 1. *Given two strings u and v, if $|u| \geqslant |v|$ then $d(u,v) \geqslant |u|-|v|$.*

3. Intractability results

3.1. Related NP-complete problems

The purpose of this subsection is purely technical: to establish that the problem we will note $LCS0$ is NP-complete. This problem can be seen as just a subcase of the more general and well-known “Longest Common Subsequence” problem, but shall be used in the sequel as generic problem for the reductions of Median String.

Longest Common Subsequence

Instance: n strings $w_1,\ldots,w_n$ over an alphabet Σ, a positive constant $k \leqslant n$.

Question: Does there exist a string w with $|w| \geqslant k$ which is a subsequence of each w_i?

This problem is known to be NP-complete if the alphabet Σ contains at least two symbols, however, it is solvable in polynomial time for any fixed k or for fixed n [7]. We can reduce longest common subsequence to the following problem:

LCS0

Instance: A positive constant k, n strings $w_1,\ldots,w_n$ over an alphabet Σ, all of length $2k$.

Question: Does there exist a string w with $|w| \geqslant k$ which is a subsequence of each w_i?

Proposition 1. *LCS*0 *is* NP-*complete.*

Proof. We will write "w is a subsequence of W" for "w is a subsequence of each w_i in W'" Let $\langle k, W=\{w_1,\dots,w_n\} w_i \in \Sigma^*\rangle$ be an instance of Longest Common Subsequence. Without loss of generality, we can suppose $n>1$.

Let $l=Max\{|w_i|: 1\leqslant i\leqslant n\}$. There are two cases to consider:

- $\boldsymbol{l<2k}$**.** Then consider the set of n new strings $W'=\{w_1',\dots,w_n'\}$ over the alphabet $\Sigma\cup\{\alpha,\beta\}$ where α and β are 2 different symbols that do not belong to Σ; all the w_i' are of length $2k$ and obtained as follows:

$$w_i' = w_i\alpha^{2k-|w_i|} \quad \text{if } i \text{ is even,}$$

$$w_i' = w_i\beta^{2k-|w_i|} \quad \text{if } i \text{ is odd.}$$

Suppose now that *LCSO* holds for W'. Thus, there exists a string w, of size at least k, that is a subsequence of each w_i'.

Obviously, w contains no occurrence of α or β (since $n>1$) and so w is also a subsequence of W. Also $|w|\geqslant k$.

Conversely, if W admits a subsequence of length at least k, then it is a subsequence of W'.

- $\boldsymbol{l\geqslant 2k}$**.** Then consider the set $W'=\{w_1',\dots,w_n'\}$ of n new strings over $\Sigma\cup\{\alpha\}$ (with $\alpha\notin\Sigma$), all of length $2(l-k)$ obtained as follows:

$$w_i' = w_i\alpha^{2(l-k)-|w_i|}.$$

Suppose again that *LCS*0 holds for W'. Let w by any solution of *LCS*0; by definition we have $|w|\geqslant l-k$. But w has at most $l-2k$ occurrences of symbol α since there exists a w_i' containing only $l-2k$ occurrences of α symbols. Thus, there exists a string (with no occurrence of α) of length at least $l-k-(l-2k)=k$ which is a subsequence of W' and as it contains no α, also a subsequence of length at least k of W.

Conversely, if W admits a subsequence w of length at least k, $w\alpha^{l-2k}$ is a subsequence of W' of length at least $l-k$. As the hypothesis is $l\geqslant 2k$, $|w\alpha^{2l-k}|\geqslant k$. Finally, *LCS*0 is a special case of Longest Common Subsequence, so it belongs to NP. □

3.2. Computing the median string is NP-hard

The problem of computing the median string is at least as difficult as the associated decision problem:

Median String

Instance: A set W of n strings $w_1,\dots,w_n$ over an alphabet Σ, a constant K.

Question: Does there exist in Σ^* a string w for which $D(w,W)\leqslant K$?

Indeed, if one can find a string w that minimises the distance with respect to a given set of strings, then comparing this distance with K, one gets the answer to the decision problem.

Theorem 1. *Median String is* NP-*complete.*

Proof. We first give the construction of the reduction i.e. the encoding of instances of $LCS0$ into instances of Median String.

Construction. Let $W=\{w_1,\ldots,w_n\}$ be an instance of $LCS0$. All these strings are of length $2k$. We construct the following instance of Median String:

$\phi(W)=\{w_1,\ldots,w_n,\alpha_1,\ldots,\alpha_{n-1}\}$ where the α_i are $n-1$ different symbols that do not belong to Σ. We take as distance parameter $k=(2n-1)k$.

The reduction holds if the following is correct:

Fact 1. *Given a set W of n strings $w_1,\ldots,w_n$, all of length $2k$, with $k>0$, there exists a string w in Σ^* for $\phi(W)$ such that $D(w,\phi(W))\leqslant K$ if and only if there exists a subsequence of length at least k of $w_1,\ldots,w_n$.*

Proof (*Fact* 1). ($\Leftarrow$) To prove this we need the following straightforward lemma:

Lemma 2. *Given two strings u and v, if u and v have no symbol in common, then $d(u,v)=\max(|u|,|v|)$.*

(*continuing with the proof of Fact* 1)

Now, let w be a subsequence of length at least k of $w_1,\ldots,w_n$. Then consider $j=|w|-k$. Since w is a subsequence of each w_i, $d(w,w_i)=|w_i|-|w|=2k-(j+k)=k-j$. Now $\forall i\in[n-1]$, $d(w,\alpha_i)=|w|=k+j$ (by Lemma 2). Thus, $D(w,\phi(W))=n(k-j)+(n-1)(k+j)=(2n-1)k-j\leqslant K$. This proves Fact 1. $\square$

($\Rightarrow$) We again need some more technical results:

Lemma 3. *Given a set W of n strings $w_1,\ldots,w_n$, all of length $2k$, with $k>0$ and $n>1$, and a string w, if $D(w,\phi(w))\leqslant K$, then there exists a string w' containing no occurrence of any α_i, such that $D(w',\phi(W))\leqslant K$.*

Proof (*Lemma* 3). If w contains no occurrence of any α symbol, we are done. If not, consider w' obtained by erasing all α symbols from w.

Thus, for each w_i in W we have $d(w_i,w')\leqslant d(w_i,w)$; for each α_j we have $d(\alpha_j,w')=|w'|$ and if α_j appears in w, $d(\alpha_j,w)=|w|-1$, if not $d(\alpha_j,w)=|w|$, therefore $d(\alpha_j,w)\geqslant|w|-1$.

Thus, adding up:

$$\begin{aligned}D(w',\phi(W)) &\leqslant (w,\phi(W))-(n-1)(|w|-1)+(n-1)|w'|\\ &=D(w,\phi(W))-(n-1)(|w|-|w'|-1).\end{aligned}$$

As $n\geqslant1$ and $|w|-|w'|\geqslant1$, $D(w',\phi(W))\leqslant D(w,\phi(W))\leqslant K$. $\square$

This lemma will be used in the sequel as follows: if a string w meets the bound $(D(w,\phi(W))\leqslant K)$, w' obtained by erasing the α symbols from w also meets the bound. We will denote this transformation by *erase* α.

Lemma 4. *Given a set W of n strings $w_1,\dots,w_n$, all of length $2k$, with $k>0$ and $n>1$, and a string w, if $D(w,\phi(W))\leqslant K)$, then $|erase\,\alpha(W)|\geqslant k$.*

Proof (*Lemma* 4). By Lemma 3 we can transform w into $w' = erase\,\alpha(w)$ that contains no α symbols. Suppose $|w'|<k$. Thus $|w'| = k-c$ (c being a non-null positive constant, and obviously $k\geqslant c$).
There are two cases to consider:

$$\text{If } k>c \qquad \forall i\in[n] \quad d(w',w_i)\geqslant k+c \qquad \text{(by Lemma 1)},$$

$$\forall i\in[n-1], \quad d(w',\alpha_i)=k-c \quad \text{(by Lemma 2)}.$$

Thus, $D(w',\phi(W))\geqslant n(k+c)+(n-1)(k-c)=(2n-1)k+c=K+c$; as $D(w',\phi(W))\leqslant K$, c is necessarily 0 and $|w|=k$.
If $k=c$, $D(w',\phi(W)) \geqslant 2nk+(n-1)=K+n+k-1$; as $D(w',\phi(W))\leqslant K$, $n+k\leqslant 1$, and n or k equal to 0. □

Lemma 5. *Given a set W of n strings $w_1,\dots,w_n$, all of length $2k$, with $k>0$, a string w such that $D(w,\phi(W))\leqslant K$, then to rewrite all the elements of W into $w' = erase\,\alpha(w)$ and respect the bound $D(w',\phi(W)) \leqslant K$ at most a total of c single-symbol insertions and single-symbol substitutions is needed, where $|erase\,\alpha(w)| = k+c$ (c being a non-null positive constant).*

Proof (*Lemma* 5). From Lemma 3, we can transform w into $w' = erase\,\alpha(w)$ which does not contain any occurrence of any α_i and respects $D(w',\phi(W))\leqslant K$.
We have

$$D(w',\phi(W)) = \sum_{i\in[n-1]} d(w',\alpha_i) + \sum_{i\in[n]} d(w',w_i).$$

To rewrite each α_i into w', $k+c$ operations (all insertions and substitutions) are needed, by Lemma 2. Hence, a total of $(n-1)(k+c)$ operations.
For each w_i (being of length $2k$) to rewrite into w' at least $k-c$ deletions are required. Hence, a total of $n(k-c)$ operations.
Thus, as there is a maximum total of $K=(2n-1)k$ operations, are left at most: $(2n-1)k-(n-1)(k+c)-n(k-c)=c$ operations for the insertions and substitutions needed to rewrite all the w_i into w', when respecting the bound $D(w',\phi(W)) \leqslant K$. □

We can now conclude the proof of Fact 1: by Lemma 3 transform w into $w' = erase\,\alpha(w)$. If $D(w',\phi(W))\leqslant K$, then, by Lemma 4 we have $|w'|=k+c$, so (by Lemma 5) w' contains at most c occurrences of symbols obtained from some w_i

in W by single-symbol insertions and single-symbol substitutions. If these symbols are erased from w' yielding w'' we have:

(1) $|w''| \geqslant k$,

(2) $\forall i \in [n] w''$ is obtained from each w_i through deletions only.

Hence, w'' is a solution of $LCS0$ for W.

It follows that Median String is NP-hard. It is also obviously in NP as given a string w, one can check in polynomial time that $D(w, W) \leqslant K$.

3.3. *Computing the centre string is NP-hard*

We now consider the problem of the centre string. In topology this corresponds to finding the centre of the smallest ball (i.e. the one with the smallest radius) containing all the elements of a given set. In the case of DNA and protein sequence data a current problem is that of the organisation of large databases [6], where the centre string is ideal for indexing. Formally:

Definition. The *maximal distance* between a string w and a finite set of strings W over an alphabet $\Sigma, (\Delta(w, W))$ is

$$\Delta(w, W) = \underset{w' \in W}{\text{Max}}\ d(w, w').$$

Given a finite set of strings W over an alphabet Σ, a string s over Σ is a *centre* of W if and only if given any string w over Σ, $\Delta(s, W) \leqslant \Delta(w, W)$.

Example. Let $W = \{aab, bb, abab\}$ and $w = ab$. Then $\Delta(w, W) = 2$.

The string bab is a centre of W. The centre is not unique, abb is an alternative centre.

We can now define the associated decision problem:

Centre String

Instance: A set W of n strings $w_1, \ldots, w_n$ over an alphabet Σ, a constant k.

Question: Does there exist in Σ^*, a string s for which $\Delta(s, W) \leqslant k$?

Theorem 2. *Centre String is NP-complete.*

Proof. Given a string s one can compute in polynomial time all the distances $d(s, w)$ (for all w in W). Hence, one can check whether each of these distances is at most k. Thus, Centre String is in NP.

We now reduce $LCS0$ to Centre String as follows: given a set W of n strings all of size $2k$, add λ to W giving W'.

Suppose the answer to $LCS0$ on instance W is yes, then there exists some subsequence s of size at least k of all strings in W, and by deleting $|s| - k$ letters from s we have a string s' of size exactly k also subsequence of all strings in W.

We now have $\Delta(s', W') = k$, and $d(\lambda, s') = k$, so $\Delta(s', W') = k$ and the answer to Centre String is positive.

Conversely, if s is a centre string for W', with $d(s,w) \leqslant k$ for each w of length $2k$ and with $d(s,\lambda) \leqslant k$, then by Lemma 1 $|s| = k$. But in that case each w' must be obtained from s through single-symbol insertions and s is necessarily a subsequence of all strings in W. $\square$

4. Conclusions

We have, nevertheless, proved in this paper that for different classical topological definitions of median and centre, finding such a string for a given set of strings leads to solving some NP-complete problem. Questions remain open: Wang and Jiang's result [16] that the problem does not admit a PTAS is clearly stronger, but requires an unnatural score scheme that does not satisfy triangular inequality. On the other hand, in the proof of our Theorem 1 the alphabet is required not to be constant. Thus, both results depend on some auxiliary parameter (the distance or the size of the alphabet). It would certainly be of interest to prove the result of NP-completeness for Median String, with the standard edit distance and a constant alphabet (ideally of size 2). Finally, a general question arises: does there exist a non-trivial distance on the set of strings such that both computing the distance between two strings and computing the median of a set of strings can be done in polynomial time (and space)?

Acknowledgements

The authors are grateful to Enrique Vidal for his suggestions on the manuscript. They also wish to thank an anonymous referee for pointing out to the relevant literature concerning this problem in Computational Biology.

References

[1] P. Aibar, M.J. Castro, F. Casacuberta, E. Vidal, Multiple template modelling of sublexical units, in: P. Laface (Ed.), Speech Recognition and Understanding Recent Advances, Trends and Applications, NATO ASI Series, Springer, Berlin, 1991, pp. 519–524.

[2] A.V. Aho, Algorithms for finding patterns in strings, Handbook of Theoretical Computer Science, Elsevier, Amsterdam, 1990, pp. 290–300.

[3] F. Casacuberta, M.D. Antonio, A greedy algorithm for computing approximate median strings, in: Proc. VII Spanish Symp. on Pattern Recognition and Image Analysis, Bellaterra, 1997, to appear.

[4] S.C. Chan, A.K.C. Wong, D.K.Y. Chiu, A survey of multiple sequence comparison methods, Bull. Math. Biol. 54 (4) (1992) 563–598.

[5] K.S. Fu, Syntactic Pattern Recognition and Applications, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[6] S. Ganguly, J. Leichter, M. Noodewier, Fast search methods for biological sequence databases, LCSR-Tech. Report 217, 1993.

[7] M.R. Garey, D.S. Johnson, Computers and Intractability: a Guide to the Theory of NP-Completeness, W.H. Freeman, San Francisco, 1979.

[8] P. Hansen, M. Labbé, J.-F. Thisse, From the median to the generalized center, Oper. Res. 25(1) (1991) 73–86.
[9] T. Kohonen, Median strings, Pattern Recognition Lett. 3 (1985) 309–313.
[10] J.B. Kruskal, D. Sankoff, An anthology of algorithms and concepts for sequence comparison, in: D. Sankoff, J.B. Kruskal (Eds.), Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison, Addison-Wesley, Reading, MA, 1983.
[11] L. Miclet, Méthodes Structurelles pour la Reconnaissance des Formes, Eyrolles, Paris, 1984.
[12] M-F. Sagot, A. Viari, H. Soldano, Multiple sequence comparison: a peptide matching approach, in: 6th Symp. on Comb. Pattern Matching, Helsinki, 1995.
[13] G.A. Stephen, String Searching Algorithms, Lecture Notes Series on Computing, vol. 3, World Scientific, Singapore, 1994.
[14] E. Ukkonen, Finding approximate patterns in strings, J. Algorithms, 6 (1985) 132–137.
[15] R. Wagner, M. Fisher, The string-to-string correction problem, J. ACM 21 (1974) 168–178.
[16] L. Wang, T. Jiang, On the complexity of multiple sequence alignment, J. Computat. Biol. 1 (4) (1994) 337–348.
[17] L. Wang, T. Jiang, E.L. Lawler, Approximation algorithms for tree alignment with a given phylogeny, Algorithmica 16 (3) (1996) 302–315.