Kernel K-Means for Categorical Data

Julia Couto

James Madison University,
Harrisonburg VA 22807, USA
coutoji@jmu.edu

Abstract. Clustering categorical data is an important and challenging data analysis task. In this paper, we explore the use of kernel K-means to cluster categorical data. We propose a new kernel function based on Hamming distance to embed categorical data in a constructed feature space where the clustering is conducted. We experimentally evaluated the quality of the solutions produced by kernel K-means on real datasets. Results indicated the feasibility of kernel K-means using our proposed kernel function to discover clusters embedded in categorical data.

1 Introduction

Clustering is an important data analysis task aimed to partition data into groups such that objects in the same group are similar among themselves while objects in different clusters are different. Most of the clustering algorithms found in the literature seek to cluster numerical data. Typically, numerical clustering algorithms rely on a distance metric, such as Euclidean distance or Minkowski distance, to measure the dissimilarity among objects. Categorical data, data whose attributes are discrete and unordered, lack a natural metric to assess the dissimilarity among categorical objects [5]. As a consequence, clustering categorical data is a difficult and challenging problem. The discovery of natural groups embedded in categorical datasets is a relevant issue in several fields such as psychology and bioinformatics.

In recent years, several clustering algorithms for categorical data have been proposed [1,2,5,9,20]. In this paper, we present a novel approach for clustering categorical data by means of kernel methods. Kernel methods [17] focus on the application of standard machine learning algorithms to data embedded into an inner product feature space through kernel functions. In applying this approach, we propose to embed categorical objects into an inner product feature space of large dimensionality where the clustering is conducted. The embedding of the categorical data into the feature space is expected to exploit the intrinsic correlations of the groups in the data [17]. The inner product in the new space defines a distance metric among the embeddings of the objects. The computation of the inner product of the embedding of any pair of objects is performed through kernel functions. A standard clustering algorithm that relies on a distance metric can be applied to discover the clusters of the categorical data in the feature space. To the best of our knowledge, there are no other categorical clustering

A.F. Famili et al. (Eds.): IDA 2005, LNCS 3646, pp. 46–56, 2005.

algorithms that follow this approach. In this paper, we formulate a novel kernel function for categorical data based on Hamming distance to embed the data into a constructed inner product feature space. Due to its simplicity, we have chosen the popular clustering algorithm K-means as the clustering algorithm to carry out the discovering of the clusters in a feature space.

This paper is organized as follows. Section 2 briefly describes some related work in clustering algorithms for categorical data. Section 3 formulates a new kernel function based on Hamming distance to compare categorical objects and describes the family of diffusion kernel functions for categorical data. Section 4 describes kernel K-means, an extension of K-means for clustering data in a feature space. Section 5 compares and discusses the quality of the clustering produced by kernel K-means using the kernel functions discussed in Section 3 with respect to other clustering algorithms for categorical data.

2 Related Work

K-modes [11] is an extension of the K-means clustering algorithm for categorical data. K-modes uses the modes of the objects grouped in the same cluster as its representative. The algorithm minimizes the dissimilarity of the objects in a cluster with respect to its mode.

STIRR [7] is an iterative method based on non-linear dynamic systems on multiple instances of weighted hypergraphs (known as basins). Each attribute value is represented as a weighted vertex. Two vertices are connected when the attribute values they represent co-occur at least once in the dataset. The weights are propagated on each hypergraph until the configuration of weights in the main basin converges to a fixed point.

ROCK [9] is an agglomerative hierarchical clustering algorithm for categorical data that uses the concept of links between objects. A link between two categorical objects is defined as the number of common neighbors. Two objects are neighbors when their Jaccard coefficient exceeds a certain threshold θ defined by the user. ROCK proceeds in an agglomerative fashion to maximize its criterion function. The choice of threshold θ is critical to the quality of the clusters found by ROCK and seems to be dataset dependent. ROCK does not exploit correlations among attributes and it does not deal with noise or missing values in a dataset.

CACTUS [5] is a combinatorial search based algorithm that uses intra-attribute and inter-attribute summary information to discover clusters of attribute values. A cluster is defined as a maximal set of strongly connected attribute values. A set of attribute values are strongly connected if the number of tuples in the dataset containing the attribute values exceeds their expected co-occurrence by a user-defined threshold under the attribute independence assumption. CACTUS uses the intra-attribute and inter-attribute summaries to compute all the cluster-projections on each attribute. Then, CACTUS heuristically constructs a set of candidate clusters by combining cluster-projections to ensure that the attribute values in the candidate clusters are pairwise strongly connected. At last, CACTUS discards those candidate clusters whose attribute values are not strongly connected.

CLICK [20] is a graph approach for clustering categorical data sets that characterizes a cluster as a maximal k-partite clique. In CLICK, each attribute value is a vertex in a k-partite graph, two vertices in the k-partite graph are linked by an edge when they belong to different attributes and are strongly connected [5]. CLICK uses a heuristic approach to detect all the maximal k-partite strongly connected.

COOLCAT [2] and LIMBO [1] use information theory to discover clusters in categorical data. COOLCAT is an incremental partition clustering algorithm to minimize the expected entropy. COOLCAT uses a sample to identify k categorical objects with maximum pairwise entropy. Afterward, COOLCAT incrementally places each object in a cluster that achieves the minimum expected entropy. COOLCAT assumes the independence of the attributes to compute the entropy of each cluster. LIMBO is a scalable two-stage clustering algorithm for large categorical datasets based on the agglomerative information bottleneck algorithm (AIB) [18]. LIMBO starts partitioning the dataset into a set of initial clusters in such a way that the loss of information is minimized. Then, LIMBO applies AIB to the initial clusters until it obtains the desired number of clusters.

More recently, some clustering kernel methods have been proposed [3,6,17,21]. Ben et al.[3] formulate the clustering problem as a convex optimization problem that finds the smallest enclosing sphere of the embedding of the data in a feature space. The preimages of the smallest enclosing sphere define the contours of the clusters in the data. In [6], the author proposes an iterative procedure similar to expectation maximization to discover clusters in a feature space in such a way that the intra-cluster distance is minimized. Objects whose embeddings belong to the same cluster in the feature space are clustered together. A kernel clustering scheme based on K-means for large datasets is proposed in [21].

3 Kernel Functions for Categorical Data

Kernel methods applies standard learning machine algorithms that rely on distance metrics or inner products to data embedded into a feature space using kernel functions. The embedding of the data into a feature space is expected to capture and enhance the patterns and regularities in the data [17]. Kernel methods proceed in two steps. The first step embeds the data into a feature space of high or infinite dimension, while the second step uses standard algorithms for classification, clustering and principal component analysis to detect the regularities of the data in the feature space. The core of kernel methods relies on the use of kernel functions. A kernel function computes the inner product in a feature space of the embedding of two data points under a certain mapping ϕ. Formally speaking:

Definition 1. *Let X be an n-dimensional input space and F be a N-dimensional inner product feature space F, $N >> n$. A kernel function $K : X \times X \rightarrow \Re$ is a symmetric function such that for all $x, y \in X$ satisfies*

$$K(x, y) =< \phi(x), \phi(y) > . \tag{1}$$

where $\phi : X \to F$ is a mapping between X and F such that for all $x \in X$

$$\phi(x) \to (\phi_1(x), \phi_2(x), \ldots, \phi_N(x)). \tag{2}$$

$\phi_i(x)$, $i = 1, \ldots, N$, are the features of x in the feature space F.

Definition 2. *The normalised kernel $\widetilde{K}$ of a kernel function K is computed as follows:*

$$\widetilde{K}(x, y) = \frac{K(x, y)}{\sqrt{K(x, x)K(y, y)}}. \tag{3}$$

The square distance between the embeddings of two points $x, y \in X$, $\phi(x)$ and $\phi(y)$ respectively, is computed in terms of the kernel function $K(x, y)$:

$$d^2(\phi(x), \phi(y)) = \|\phi(x) - \phi(y)\|^2 = K(x, x) - 2K(x, y) + K(y, y) \ . \tag{4}$$

A kernel function can be formulated by defining the mapping between input space and some feature space where the inner product is computed [17]. A kernel function defined in this way requires the characterization of the feature space F, the specification of the embedding $\phi : X \to F$, and finally the computation of the inner product between the embedding of two points. However, the computation of the features and the evaluation of the inner product have high computational costs that depend on the dimension of the feature space. An efficient approach uses computational methods such as dynamic programming to compute the kernel function for any pair of points without explicitly embedding the points in the feature space and then computing their inner product [16,17], which is the approach applied in this paper. Alternatively, the characterisation of kernel functions as finitely positive semi-definite functions [17] allows determining whether a function is a kernel function without knowing the nature of the feature space and the specification of the mapping ϕ. Finally, kernel functions satisfy several closure properties that allows constructing complex kernel functions by manipulating and combining simpler ones[17].

3.1 Hamming Distance Kernel Function

We formulate a kernel function for categorical data that uses Hamming distance to embed categorical data into a constructed inner product feature space. Our proposed kernel function does not depend on a generative model or a priori information about the nature of the data. The construction of the feature space and the kernel function follows the same methodology proposed in [16].

Definition 3. *Let D_i be a finite domain of categorical values. Let $(a_1, \ldots, a_n)$ be a categorical object such that $a_i \in D_i$. Let $D^n = \prod_{i=1}^{n} D_i$ be the cross product over all the domains of the attributes such that for each $(u_1, \ldots, u_n) \in D^n$, $u_i \in D_i$. Given a categorical object $s = (s_1, \ldots, s_n)$, s_k denotes the value of the k-th attribute of s. The feature space F is a subspace of $\Re^{D^n}$.*

Definition 4. *The mapping of a categorical object s into the feature space F is defined by the u coordinate $\phi_u(s) = \lambda^{H(u,s)}$, for all $u \in D^n$, $\lambda \in (0,1)$. The Hamming distance $H(u,s)$ between s and u is defined as:*

$$H(u,s) = \sum_{i=1}^{n} \delta(u_i, s_i) \ . \tag{5}$$

where $\delta(x,y)$ is 0 when $x = y$ and 1 otherwise. The u coordinate of s according to the mapping ϕ can be rewritten as:

$$\phi_u(s) = \lambda^{H(u,s)} = \prod_{i=1}^{n} \lambda^{\delta(u_i,s_i)} \ . \tag{6}$$

Definition 5. *The kernel function $K_H(s,t)$ between two input categorical objects s and t is defined as:*

$$K_H(s,t) = \sum_{u \in D^n} \phi_u(s)\phi_u(t) = \sum_{u \in D^n} \prod_{i=1}^{n} \lambda^{\delta(u_i,s_i)} \lambda^{\delta(u_i,t_i)} \ . \tag{7}$$

It can be shown that the kernel function $K_H(s,t)$ can be computed recursively in the following manner:

$$\begin{aligned} &K^0(s,t) = 1 \\ &K^j(s,t) = (\lambda^2(|D_j| - 1 - \delta(s_j,t_j)) + (2\lambda - 1)\delta(s_j,t_j) + 1)K^{j-1}(s,t) \quad 1 \leq j \leq n \\ &K_H(s,t) = K^n(s,t) \ . \end{aligned} \tag{8}$$

Due to lack of space, we omit the proof of the correctness of this recursion. Finally, $\widetilde{K}_H(s,t)$ denotes the normalised kernel of $K_H(s,t)$.

3.2 Diffusion Kernels

Kondor and Lafferty [14] proposed a family of kernel functions for categorical data based on an extension of hypercube diffusion kernels. The feature space is a graph induced by the set D^n. Each categorical object $s \in D^n$ is a vertex in the graph. Two vertices v_s and v_t are connected by an edge whenever their underlying categorical objects s and t differ only in the value of one attribute, i.e., $H(s,t) = 1$. Let β be a bandwidth parameter, the family of diffusion kernel functions $K_{DK}(\beta)$ for categorical data with n attributes is defined in the following way:

$$K_{DK}(\beta)(x,y) = \prod_{i=1}^{n} \left(\frac{1 - e^{-|D_i|\beta}}{1 + (|D_i| - 1)e^{-|D_i|\beta}} \right)^{\delta(x_i,y_i)} \ . \tag{9}$$

where $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ are categorical objects.

4 Kernel K-Means

Kernel K-means is an extension of the popular clustering algorithm K-means to discover clusters in a feature space in which the distance is calculated via kernel functions. Let $\phi : X \rightarrow F$ be an embedding of a set X into a feature space F and $K : X \times X \rightarrow \Re$ its associated kernel function. Let $z_1, \ldots, z_k$ be the centroids of clusters $C_1, \ldots, C_k$ respectively. Kernel K-means can be formulated in the following way:

1. Initialization Step: Select k data points and set their embeddings in feature space as the initial centroids $z_1, \ldots, z_k$.
2. Assignment Step: Assign each data point x_i to a cluster C_q such that:

$$q = \arg\min_j d^2(\phi(x_i), z_j)$$

$$z_j = \frac{1}{|C_j|} \sum_{x_p \in C_j} \phi(x_p)$$

$$d^2(\phi(x_i), z_j) = K(x_i, x_i) - \frac{2}{|C_j|} \sum_{x_p \in C_j} K(x_i, x_p) + \frac{1}{|C_j|^2} \sum_{x_p \in C_j} \sum_{x_m \in C_j} K(x_p, x_m) \ .$$

3. Repeat 2 until convergence.

We use the scheme for kernel K-means for large datasets proposed in [21]. Nevertheless, in our implementation of kernel K-means, the initialization step applies a heuristic similar to [13] to select the initial centroids. The heuristic selects k well-scattered points in the feature space by maximizing their minimum pairwise distance. The initialization heuristic proceeds as follows:

1. Selects the two most distant embeddings of the dataset in the feature space as the initial centroids of the first two clusters C_1 and C_2, namely z_1 and z_2.

2. Then, the selection of the initial centroid z_i, $i = 3, \ldots, k$, of the remaining clusters proceeds in an iterative fashion. First, it computes the minimum distance between the embeddings of the remaining points to the existing centroids. Then, the object with the largest minimum distance in the feature space to the existing centroids is selected as the initial centroid z_i of cluster C_i. This step is repeated until k initial centroids have been obtained.

Table 1

Dataset	N. Records	N.Classes	Attributes	Missing Values
Votes	435	2	16	288
Mushrooms	8125	2	22	2480
Soybean	47	4	35	0
Zoo	101	7	15	0

5 Experimental Evaluation

We conducted a series of experiments to evaluate and compare the quality of the clustering produced by kernel K-means using both the kernel functions $\widetilde{K}_H$ (KKM-NH) and diffusion kernels $K_{DK}(\beta)$ (KKM-DK) with ROCK, COOLCAT and K-modes. The experiments were run in a DELL computer equipped with a Pentium 4 running at 2.8 GHz, and 1 Gigabyte of main memory, running SUSE Linux 9.1. We assessed the quality of the clustering produced by aforementioned clustering algorithms on four real categorical datasets obtained from the UCI Machine Learning Repository [4]. Missing values were treated as another attribute value. The characteristics of the datasets are summarized in Table 1.

5.1 Clustering Quality Measures

Validation of the quality of the clustering produced by a clustering algorithm is one of the most important issues in clustering analysis. Several quantitative measures have been proposed to evaluate the quality of the clustering solution found by a clustering algorithm [12,10]. In our experiments, we used three quantitative measures, External Entropy, F-Measure and Category Utility, to assess the quality of the solutions produced by the aforementioned algorithms. When comparing several clustering algorithms, if a clustering algorithm outperforms the others in most of these measures for a given dataset, it is assumed to be the best clustering algorithm for that dataset [19].

External Entropy. The external entropy is a measure of the purity of the clusters found by a clustering algorithm. Let $D = \{x_1, \ldots, x_N\}$ be a dataset of categorical objects. Let $C = \{C_1, \ldots, C_k\}$ be a clustering and let $c = \{c_1, \ldots, c_k\}$ be the classes in the data. The expected entropy of the clustering C is the weighted external entropy of each cluster:

$$E(C) = \sum_{i=1}^{k} \frac{|C_i|}{N} \sum_{j=1}^{k} P(c_j|C_i) log(P(c_j|C_i) \quad . \tag{10}$$

F–Measure. The F–measure is a combination of precision and recall measurements from information retrieval [15]. Let $P(c_i, C_j)$ be the precision of a class c_i in a cluster C_j and $R(c_i, C_j)$ be the recall of a class c_i in a cluster C_j. The F-measure of a class c_i in a cluster C_j is defined as follows:

$$F(c_i, C_j) = \frac{2R(c_i, C_j)P(c_i, C_j)}{R(c_i, C_j) + P(c_i, C_j)} \quad . \tag{11}$$

The overall F–measure of a clustering is given by [19]:

$$F = \sum_{i} \frac{|c_i|}{N} \max_{j}\{F(c_i, C_j)\} \quad . \tag{12}$$

A larger F–measure indicates a better quality of the clustering.

Category Utility (CU). Category utility [8] measures the increase of the expected probability of attribute values of the objects in the same clusters over the expected probability of the attributes. Category Utility is computed as follows:

$$CU = \sum_{j=1}^{k} \frac{|C_j|}{N} \sum_{i=1}^{n} \sum_{v \in D_i} [P(A_i = v|C_j)^2 - P(A_i = v)^2] \ . \tag{13}$$

5.2 Experimental Details

We performed several experiments on synthetic datasets to empirically determine the parameters λ and β for KKM-NH and KKM-DK respectively that produce the best clustering of the data. Our experiments indicated that the parameters for KKM-NH and KKM-DK that achieve the best clustering are dataset dependent. Nevertheless, KKM-NH with λ between 0.6 and 0.8 produced a good clustering of the data with respect to External Entropy, F-Measure and Category Utility. In our experiments on the UCI datasets, we set the parameter λ to 0.6 for KKM-NH and the parameter β for KKM-DK was set between 0.1 and 2.0.

COOLCAT is sensitive to the size of the sample used to seed the initial clusters as well as the ordering of the data points in the datasets [1]. For each dataset, we ran COOLCAT on twenty random orderings. In each run, the whole dataset was set as the sample used to find the initial seeds of the clusters.

The quality of the clustering produced by ROCK is highly influenced by both the choice of the threshold θ as well as the ordering of the data. The threshold θ that results in the best performance is dataset dependent. For each dataset, we generated twenty random orderings and ran ROCK with threshold θ between 0.1 to 0.95.

Both K-modes and kernel K-means are sensitive to the initial centroids of the clusters. In our experiments, we ran K-modes with twenty random restarts. Kernel K-means KKM-NH and KKM-DK were run using twenty different initial centroids selected according to the initialization heuristic explained in section 4.

Finally, the quality measures reported for all the clustering algorithms are averages over all the runs. For ROCK and KKM-DK, we also report the algorithm's parameter that produced the best average results.

5.3 Results

The average quality measures produced by K-modes, COOLCAT, ROCK, KKM-HN and KKM-DK on the UCI datasets are shown in Table 2.

KKM-NH and KKM-DK achieved the best clustering for Congressional Votes and Soybean with respect to the three quality measures. On the ZOO dataset, KKM-NH and KKM-DK outperformed the other algorithms with respect to External Entropy and Category Utility. F-measure obtained by both KKM-DK and KKM-DK on this dataset were comparable to the F-Measure obtained by ROCK. On the Mushrooms dataset, K-modes produced the best results with

Table 2

Dataset	Clustering Algorithm	EE	F	CU
Congressional Vote	K-Modes	0.519	0.864	2.896
	ROCK (θ=0.73)	0.654	0.798	1.891
	COOLCAT	0.511	0.864	2.839
	KKM-NH	0.477	0.880	2.941
	KKM-DK (β=1.6)	0.475	0.880	2.941
Mushrooms	K-Modes	0.751	0.706	1.504
	ROCK (θ=0.8)	0.849	0.653	1.064
	COOLCAT	0.791	0.701	1.465
	KKM-NH	0.910	0.618	1.313
	KKM-DK (β=0.5)	0.811	0.634	1.510
	KKM-NH(*)	0.715	0.751	1.404
	KKM-DK(*) (β=0.3)	0.786	0.713	1.183
Soybean	K-Modes	1.229	0.560	2.950
	ROCK (θ=0.75)	0.021	0.996	5.493
	COOLCAT	0.033	0.986	5.489
	KKM-NH	0.000	1.000	5.558
	KKM-DK (β=0.6)	0.000	1.000	5.558
Zoo	K-Modes	1.229	0.560	2.950
	ROCK (θ=0.69)	0.294	0.898	4.127
	COOLCAT	0.376	0.793	4.320
	KKM-NH	0.272	0.803	4.454
	KKM-DK (β=1.2)	0.262	0.844	4.476

(*) Random centroids

respect to the three quality measures. However, the poor performance of KKM-NH and KKM-DK can be explained by an inadequate selection of the initial centroids using our initialization heuristic. To confirm this hypothesis, we ran 20 trials of KKM-NH and KKM-DK selecting the initial centroids at random (Table 2). Our experiments showed an overall improvement in the quality of the clustering produced by KKM-NH with respect to External Entropy, F-Measure and CU. The results for KKM-DK(β=0.3) showed an improvement of External Entropy and F-Measure. Nevertheless, its CU was significantly lower than the one obtained by KKM-DK(β=0.5) applying the initialization method explained in section 4.

6 Conclusions and Future Work

In this paper, we have proposed the use of kernel clustering methods to cluster categorical data in a constructed feature space via kernel functions. We have introduced a new kernel function for categorical data, $\widetilde{K}_H$, based on the Hamming distance. We have applied kernel K-means to cluster categorical data embedded in a feature space via the kernel functions $\widetilde{K}_H$ and diffusion kernels $K_{DK}(\beta)$. The results of our experiments indicate that the embedding of categorical data by means of the kernel functions $\widetilde{K}_H$ and diffusion kernels $K_{DK}(\beta)$ preserves the clusters in the data. Furthermore, our results demonstrate that the solutions produced by kernel K-means embedding categorical data through the new kernel function $\widetilde{K}_H$ (λ=0.6) are generally better than the other categorical clustering algorithms compared in this paper. With regard to KKM-DK, our experiments show that the choice of the parameter β is crucial for discovering the clusters in the data. As a consequence, the application of KKM-DK for clustering categorical data is deterred by the selection of the appropriate parameter β that fits the data.

In our future work, we will focus on an incremental approach for kernel K-means to overcome the disk-space and I|O requirements of the method when dealing with massive datasets. In addition, we plan to investigate the performance of KKM-NH on datasets containing noise and missing values. Finally, we will evaluate the sensitivity of KKM-HN to the number of classes and number of relevant attributes defining the classes of a dataset.

References

1. Andritsos, P., Tsaparas, P., Miller, R. J., Sevcik., K. C.: LIMBO: Scalable Clustering of Categorical Data. In Proceedings of the 9th International Conference on Extending Database Technology (EDBT 2004), Heraklion, Crete, Greece, March 2004.
2. Barbara, D., Couto, J., Li Y.: Coolcat: An Entropy-based algorithm for Categorical Clustering. In Proceedings of the 11th ACM Conference on Information and Knowledge Management (CIKM 02), McLean, Virginia, USA, November 2002, ACM Press, pp. 582–589.
3. Ben-hur, A., Horn, D., Siegelmann, H.T., Vapnik V.: Support Vector Clustering. Journal of Machine Learning Research 2, pp. 125–137.
4. Blake, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/˜mlearn/MLRepository.html. University of California, Department of Information and Computer Science, Irvine, CA.
5. V. Ganti, J. Gehrke, and R. Ramakrishnan.: CACTUS: Clustering Categorical Data using Summaries. In Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), San Diego, CA, USA, August 1999, ACM Press, pp. 73–83.
6. Girolami, M.: Mercer Kernel Based Clustering in Feature Space. IEEE Transactions on Neural Networks, 13(4), pp. 780–784, 2002.
7. Gibson, D., Kleinberg, J., Raghavan, P.: Clustering Categorical Data: An Approach Based on Dynamical Systems. In Proceedings of the 24th International Conference on Very Large Data Bases, (VLDB), New York, NY, USA, August 1998, Morgan Kaufmann, pp. 311–322.
8. Gluck, A., Corter, J.: Information, Uncertainty, and the Utility of Categories. In Proceedings of the 7th Annual Conference of the Cognitive Science Society, Irvine, California, 1985, Laurence Erlbaum Associates, pp. 283–287.
9. Guha, S., Rastogi, R, Shim, K.: ROCK: A Robust Clustering Algorithm for Categorical Attributes. Journal of Information Systems, 25(5), pp. 345–366, 2000.
10. M. Halkidi ,Y. Batistakis, M. Vazirgiannis.: On Clustering Validation Techniques. Journal of Intelligent Information Systems, 17(2–3), pp. 107–145, 2001.