An Algorithm for Multidimensional Data Clustering

S. J. WAN, S. K. M. WONG, and P. PRUSINKIEWICZ
University of Regina

A new divisive algorithm for multidimensional data clustering is suggested. Based on the minimization of the sum-of-squared-errors, the proposed method produces much smaller quantization errors than the median-cut and mean-split algorithms. It is also observed that the solutions obtained from our algorithm are close to the local optimal ones derived by the k-means iterative procedure.

Categories and Subject Descriptors: I.5.3 [**Pattern Recognition**]: Clustering—*algorithms*

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Clustering, divide-and-conquer strategy, divisive algorithm, k-d tree, partition, quantization

1. INTRODUCTION

A clustering procedure can be viewed as one of finding groupings in a set of events by extremizing some criterion function. In a variety of problems, such as unsupervised learning [3], multivariate data analysis [6], and digital image processing [5, 7], the collection of data can be represented as a set of points in a multidimensional vector space. One of the most widely used criterion functions for clustering analysis is the sum of squared Euclidean distances measured from the cluster centers. The main task in clustering analysis is then to seek the groupings that minimize the sum-of-squared-errors.

Consider N input data points $\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_N$ in a m-dimensional vector space Ω. The objective is to find K ($K \ll N$) cluster centers $\vec{r}_1, \vec{r}_2, \ldots, \vec{r}_K$ such that the average sum-of-squared-errors [3] defined by

$$E = \frac{1}{N} \sum_{i=1}^{N} \| \vec{s}_i - \vec{R}(\vec{s}_i) \|^2 \tag{1}$$

is minimum, where $\vec{R}(\vec{s}_i) \in \{\vec{r}_1, \vec{r}_2, \ldots, \vec{r}_K\} \subset \Omega$ is the cluster center closest to $\vec{s}_i$, namely,

$$\vec{R}(\vec{s}_i) = \arg \operatorname*{Min}_{1 \le j \le K} \| \vec{s}_i - \vec{r}_j \|^2. \tag{2}$$

Authors' address: Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada S4S OA2.

In general, the N input data points are not necessarily distinct. Let $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n$ denote the n distinct points ($n < N$), and $p(\vec{x}_1), p(\vec{x}_2), \ldots, p(\vec{x}_n)$ the corresponding occurrence frequencies. Equations (1) and (2) can be rewritten as

$$E = \sum_{i=1}^{n} p(\vec{x}_i) \| \vec{x}_i - \vec{R}(\vec{x}_i) \|^2, \tag{3}$$

$$\vec{R}(\vec{x}_i) = \arg \operatorname*{Min}_{1 \le j \le K} \| \vec{x}_i - \vec{r}_j \|^2. \tag{4}$$

The boundaries of the K clusters defined by Eq. (4) form a m-dimensional Voronoi diagram in Ω.

It is known that the problem of finding the global minimum solution for Eqs. (3) and (4) is NP-complete [8]. For this reason, a number of approximate clustering techniques [3, 7, 9, 13] have been developed. These techniques can be divided into two categories: iterative optimization and heuristic methods. The algorithm most frequently used for seeking a local minimum solution is the k-means iterative procedure [9]. However, the converging time required by the iterative schemes can easily become unmanageable, particularly for large clustering problems.

Instead of finding a local minimum solution, the heuristic approach is designed to reduce the computational complexity, and at the same time produce an acceptable solution. There are two basic heuristic methods: the agglomerative and the divisive techniques. In this paper we are concerned only with the latter. A divisive approach is a procedure that partitions the vector space sequentially into K disjoint subregions. For simplicity, the partition hyperplanes are assumed to be perpendicular to one of the coordinate axes. This means that each of the subregions is actually a hyperbox. The centroids of the K-resulting hyperboxes are chosen to be the cluster centers.

A well-known method in the divisive approach is the divide-and-conquer strategy [1]. In this method, a problem of N points in a m-dimensional space is reduced to first recursively solve two problems each with $N/2$ points in a m-dimensional space, and then recursively solve a problem of N points in a $(m - 1)$-dimensional space. Based on this technique, several divisive procedures such as the median-cut [7] and mean-split [13] algorithms have been proposed. It should be pointed out here that there is a drawback in using a recursive strategy for our clustering problem (see Sections 2.1 and 2.2).

This paper presents a new divisive algorithm for clustering. The salient features of the proposed method are

— it is not recursive;
— the partition strategy is based on the minimization of the sum-of-squared-errors.

The experimental results show that our clustering algorithm always produces much smaller errors than other divisive methods. More importantly, the sum-of-squared-errors produced by our method is very close to those obtained from the k-means algorithm.

In Section 2, the k-means iterative procedure and two divisive algorithms—the median-cut (k-d tree) and the mean-split algorithms—are reviewed and analyzed. Our algorithm is introduced in Section 3. A comparison of the performance of different algorithms is presented in Section 4.

2. REVIEW OF OTHER METHODS

2.1 Median-Cut Algorithm (k-d Tree)

Heckbert [7] proposed the median-cut algorithm to find the representative colors for image quantization. In this method, the 3-dimensional color space is *recursively* subdivided into rectangular hyperboxes. To subdivide any hyperbox, the orientation of the partition plane is chosen to be perpendicular to the coordinate axis with the largest color spread, and passes through the median point of the projected color distribution along this axis. As a result of this operation, each final hyperbox contains nearly the same number of data points. The centroids of these hyperboxes are chosen to be the representative colors (cluster centers).

Using the spatial-storage scheme (i.e., storing input data point-by-point), the complexity of this algorithm is $O(m \log KN \log N)$ for time and $O(mN)$ for space.

The theoretical basis of the median-cut algorithm can be traced to the k-d tree method proposed by Bentley et al. [2, 4]. The k-d tree algorithm was originally designed to minimize the expected cost of searching for the nearest neighboring records in information retrieval. For this purpose, it is desirable to assign approximately an equal number of records to all terminal nodes in the search tree. However, such a rationale may not be applicable to the clustering problem of image quantization. There is no sound justification to require that each of the resulting hyperboxes should contain a nearly equal number of data points, while ignoring entirely how these points are distributed. Furthermore, adopting the recursive strategy to partition the space has an inherent drawback, because in such a method the decision of whether or not to subdivide a hyperbox is based solely on its position in the k-d tree, without any regard to the contents of the hyperbox. Consequently, a hyperbox with a large quantization error in one subtree may not be chosen for further partition, whereas a hyperbox containing only one point (with high occurrence frequency) in another subtree may be chosen to be subdivided instead. Although this problem can be alleviated by reassigning the partition to the largest hyperbox among the remaining terminal nodes in the current tree, such an ad hoc remedy is inconsistent with the recursive nature of the algorithm.

2.2 The Mean-Split Algorithm

Using the divide-and-conquer strategy for both *spatial* and *quota* divisions, Wu and Witten [13] suggested the mean-split algorithm. Its main features are summarized below:

— Hyperplanes are used for partition as in the median-cut algorithm.
— The partition point is chosen to be the mean rather than the median of the projected distribution with the largest spread.

— Let L be the number of clusters originally assigned to a hyperbox. When the hyperbox is split into two smaller ones, one assigns L_i clusters to the smaller hyperbox i according to the following formula:

$$L_i = L\left[q\frac{n_i}{n_1 + n_2} + (1 - q)\frac{V_i}{V_1 + V_2}\right], \qquad (i = 1, 2), \tag{5}$$

where n_i is the number of data points and V_i the volume containing all the data points in hyperbox i. The heuristic parameter q is restricted to the range $0.5 \leq q \leq 0.7$.

— A minimum hyperbox size is prespecified. Any hyperbox smaller than this size will not be further subdivided.

Indeed, some of the drawbacks in the Heckbert algorithm are partially removed by this technique. However, there remain a number of problems associated with this method. To partition a hyperbox by a plane passing through the mean instead of the median does not necessarily lead to a lower quantization error. The inherent drawback of the recursive strategy still exists. For instance, when attempting to partition a hyperbox smaller than the prespecified size, one may still have to reassign the partition to the largest hyperbox as in the median-cut algorithm. Moreover, there is also a lack of theoretical justification for the choice of a number of parameters involved in this algorithm. In fact, a better criterion than Eq. (5) seems to be

$$L_i = L\frac{\sigma_i^2}{\sigma_1^2 + \sigma_2^2}, \qquad (i = 1, 2) \tag{6}$$

where σ_i^2 is the variance of hyperbox i. The need to introduce a parameter such as q is also avoided.

The main advantage of the mean-split algorithm is that it has a lower computational cost — $O(mN \log K)$ time and $O(mN)$ space for the spatial-storage scheme—mostly because it is simpler to compute the mean than the median.

2.3 The K-Means Algorithm

The k-means iterative procedure has received considerable attention in clustering analysis. This algorithm can be summarized as follows. First select K initial cluster centers. Then, the K clusters are formed by associating each data point with its closest cluster center. The centroids of these K clusters become the new cluster centers. The above procedure is repeated until the new cluster centers are the same as the previous ones. Although the k-means algorithm has been widely used in many applications, it has been shown only recently [10] that it converges to a local minimum solution in a finite number of iterations if a quadratic metric is used.

With the spatial-storage scheme, the time complexity of this algorithm is proportional to $mTNK$. The number of iterations T necessary for the algorithm to converge depends on the distribution of the data points, the number of clusters required, the size of the space, and the choice of initial cluster centers. For a large clustering problem, the computation can be very costly. For example, it

may take more than twenty hours on a VAX 780 computer to produce 256 clusters for a full-color image [13].

3. THE PROPOSED CLUSTERING ALGORITHM

There are two fundamental issues in the design of a divisive clustering algorithm. At each step of the partition, one must first decide which hyperbox should be partitioned and at the same time choose an appropriate hyperplane to subdivide the hyperbox. In our algorithm, both decisions are made based on the minimization of the sum-of-squared-errors.

How to choose the hyperbox for further partition. Let Ω be the hyperbox that contains all the input data points, $p(\vec{x})$ the occurrence frequency of $\vec{x}$, and $\sum_{\vec{x}\in\Omega} p(\vec{x}) = 1$. Given a hyperbox $\Omega_l \subseteq \Omega$, the centroid (mean) $\vec{\mu}_l$ and the variance σ_l^2 are defined by

$$\vec{\mu}_l = \sum_{\vec{x}\in\Omega_l} \frac{p(\vec{x})}{W_l}\,\vec{x}, \tag{7}$$

$$\sigma_l^2 = \sum_{\vec{x}\in\Omega_l} \|\vec{x} - \vec{\mu}_l\|^2 \frac{p(\vec{x})}{W_l}, \tag{8}$$

where $W_l = \sum_{\vec{x}\in\Omega_l} p(\vec{x}) \leq 1$ is the weight of the hyperbox Ω_l.

The process of mapping all the points within a hyperbox onto a representative point is called *quantization*. In this paper the quantization error is measured by the sum-of-squared-errors. It is easy to see that using the centroid of a hyperbox as its representative point will produce the smallest quantization error.

The quantization error attributed to a hyperbox Ω_l is determined by its *weighted* variance $\hat{\sigma}_l^2$:

$$\hat{\sigma}_l^2 = W_l\sigma_l^2 = \sum_{\vec{x}\in\Omega_l} \|\vec{x} - \vec{\mu}_l\|^2 p(\vec{x}). \tag{9}$$

In order to reduce the total quantization error, at each step of the subdivision the hyperbox with the highest weighted variance is partitioned. That is, the hyperbox with the largest quantization error is chosen to be further subdivided.

How to choose the partition hyperplane. Suppose the hyperbox Ω_l is split into two smaller hyperboxes Ω_{l1} and Ω_{l2} by a hyperplane Γ perpendicular to one of the coordinate axes. The quantization error is given by the weighted sum of variances:

$$E(\Gamma) = W_{l1}\sigma_{l1}^2 + W_{l2}\sigma_{l2}^2 = \sum_{\vec{x}\in\Omega_{l1}} \|\vec{x} - \vec{\mu}_{l1}\|^2 p(\vec{x}) + \sum_{\vec{x}\in\Omega_{l2}} \|\vec{x} - \vec{\mu}_{l2}\|^2 p(\vec{x}), \tag{10}$$

where W_{li}, $\vec{\mu}_{li}$, and σ_{li}^2 are the weight, mean, and variance of the ith hyperbox ($i = 1, 2$). The *optimal* partition hyperplane Γ_{opt} is defined as the one that minimizes the quantization error:

$$\Gamma_{\text{opt}} = \arg \min_{\Gamma} E(\Gamma), \tag{11}$$

where $\{\Gamma\}$ is the set of all possible hyperplanes perpendicular to the coordinate axes.

In practice, to find the optimal partition hyperplane defined by Eq. (11) is a very time-consuming task. One way to simplify the computation is to consider the projected distributions.

Let μ be the mean and σ^2 the variance of a 1-dimensional distribution. Let t be a cut-point which splits the distribution into two intervals. The *optimal* cut-point t_{opt} is defined as the one that maximizes the reduction of variances:

$$t_{\text{opt}} = \arg \operatorname*{Max}_{t} [\sigma^2 - (w_1\sigma_1^2(t) + w_2\sigma_2^2(t))], \tag{12}$$

where w_i and $\sigma_i^2(t)$ are the weight and variance of the ith interval ($i = 1, 2$). It can be easily verified that the reduction of expected variance can be calculated by the simple formula:

$$\sigma^2 - [w_1\sigma_1^2(t) + w_2\sigma_2^2(t)] = \frac{w_1}{w_2}[\mu - \mu_1(t)]^2, \tag{13}$$

where $\mu_1(t)$ is the mean of the first interval. Furthermore, it has been proved in [12] that the optimal cut-point is given by

$$t_{\text{opt}} = \arg \operatorname*{Max}_{(\mu+\text{lower})/2 \le t \le (\mu+\text{upper})/2} \left[\frac{w_1}{w_2}(\mu - \mu_1(t))^2\right], \tag{14}$$

where the symbols *lower* and *upper* specify the boundaries of the distribution. Based on Eq. (14), the optimal cut-point and the reduction of expected variance can be computed at the same time. Consequently, only half of the range needs to be searched.

Therefore, in our algorithm the partition hyperplane can be chosen in the following way. Given a hyperbox, compute m 1-dimensional distributions by projecting all points within the hyperbox onto each of the coordinate axes. For each projected distribution, compute the optimal cut-point and the corresponding reduction of expected variance from Eq. (14). We choose the partition hyperplane perpendicular to the axis along which the reduction of expected variance is the largest among all projected distributions. The partition hyperplane passes through the optimal cut-point of this axis.

How to form clusters. The above process of partition is repeated until the required number of clusters is generated. The cluster centers are chosen to be the centroids of the resulting hyperboxes. The clusters are formed by mapping each data point onto its closest cluster center. The local search technique [7] provides an efficient way to perform this mapping operation.

The proposed algorithm is summarized below:

(1) Choose the hyperbox with the largest weighted variance (see Eq. (9)) for further partition.
(2) Project all data points in the hyperbox onto each of the m coordinate axes. For each of the m projected distributions, calculate the optimal cut-point and the reduction of expected variance from Eq. (14).
(3) Partition this hyperbox by the hyperplane perpendicular to the axis along which the reduction of expected variance is the largest. This hyperplane intersects this axis at the optimal cut-point.

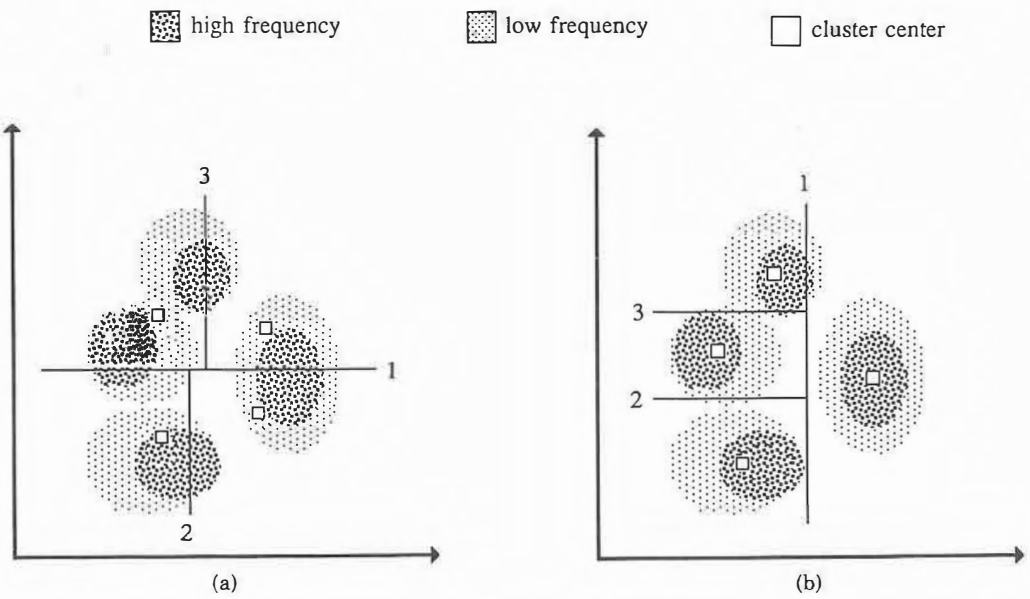

Fig. 1. Two different partition methods ($K = 4$): (a) the median-cut algorithm; (b) the proposed algorithm.

(4) Compute the weighted variance for each of the two smaller hyperboxes.
(5) Repeat steps (1) to (4) until the number of the hyperboxes reaches the required number of clusters.
(6) Calculate the centroids of the resulting hyperboxes, which form the desired cluster centers.
(7) Map each data point to its closest cluster center.

The basic idea behind our algorithm is to adaptively assign more clusters to the regions with high quantization errors such that the total quantization error is reduced. As a result, in contrast to the k-d tree method, different hyperboxes may contain quite different numbers of data points, but the contribution of quantization error from each hyperbox is nearly the same.

The difference in partition strategy between the proposed algorithm and the median-cut algorithm can be illustrated by a 2-dimensional example. In Figure 1, the sequence of partitions is labelled 1, 2, and 3. It is clearly demonstrated by this example that our method is able to appropriately assign cluster centers to the significant peaks of the distribution, whereas the median-cut algorithm fails to do so.

In the implementation of the clustering algorithms, two storage schemes may be used: spatial-storage and frequency-storage. In the spatial-storage scheme, the N input points are stored in a 2-dimensional array containing $m \times N$ elements. The main drawback of such a scheme is the ineffective use of storage space, since the same input points are stored individually. In order to require the statistical information of a nonterminal hyperbox in the tree, the whole array must be searched. On the other hand, the frequency-storage scheme provides the occur-

Table I. Comparison of the Sum-of-Squared-Errors

Problems	K	Median-Cut	Mean-Split	Our method
$m = 2$	8	12.3081	10.4428	7.6542
$N = 256 \times 256$	64	1.1299	0.9995	0.7924
$m = 3$	8	28.0788	27.5144	23.2011
$N = 256 \times 256$	64	7.4650	6.0831	4.7257
$m = 4$	8	38.7182	35.9201	33.5446
$N = 256 \times 256$	64	12.9982	12.8487	8.5401

rence frequencies of the input points in a m-dimensional array. The size of the array depends on the spread of each component axis. This scheme is also consistent with the geometric positions of the hyperboxes. Thus, to partition a hyperbox, the search is limited to a small local region only. Another advantage of the frequency-storage scheme is that the computational time is independent of the number of input data points.

If the dimension is large, we should reduce the number of data points in the vector space before applying the clustering operations. Since the number of data points in the vector space is usually much larger than the number of clusters required, such a compression would not significantly increase the errors, but it would greatly simplify the computation. For example, in color image quantization, using 5 bits (instead of 8 bits) per color component has no noticeable impact on the image quality. However, the size of the RGB color space is reduced from 256 $\times$ 256 $\times$ 256 to 32 $\times$ 32 $\times$ 32. The computational complexity of our clustering algorithm is $O(mN_1)$ for storage space and $O(mKN_1)$ for time, where N_1 is the number of data points in the compressed space.

4. PERFORMANCE OF DIFFERENT CLUSTERING ALGORITHMS

In order to evaluate the proposed method, we compare it with the median-cut, the mean-split, and the k-means algorithms. In our experiments, we chose three collections of data (with different dimensions m = 2, 3, 4) from a color image database. For efficiency, we adopted the compressed frequency-storage scheme.

The objective of the first group of experiments is to compare the quantization errors produced by the different divisive algorithms. Each set of data is quantized into 8 and 64 clusters, respectively. The total sum-of-squared-error (computed from Eq. (3)) for each case is listed in Table I.

Our results indicate that the mean-split algorithm performs better than the median-cut algorithm. However, in all the cases we studied, our method produces much smaller errors than both the median-cut and mean-split algorithms.

The second group of experiments is designed to see how much improvement can be achieved by the k-means iterative procedure. The quantization errors produced by applying the k-means algorithm with the starting cluster centers derived from different algorithms are given in Table II.

The differences in the quantization errors between Table I and Table II are shown in Table III. It can be seen that our solutions are quite close to the local optimal solutions obtained from the k-means algorithm. However, this is not the case for the other two algorithms.

Table II. The Sum-of-Squared-Error Produced by the k-Means Algorithm Based on Different Initial Cluster Centers

Problems	K	Median-Cut	Mean-Split	Our method
$m = 2$	8	7.6692	7.2185	7.2645
$N = 256 \times 256$	64	0.8622	0.8579	0.7840
$m = 3$	8	24.4701	25.3078	22.6893
$N = 256 \times 256$	64	4.5508	4.8853	4.5399
$m = 4$	8	33.0417	33.4320	32.8519
$N = 256 \times 256$	64	8.0713	8.2480	8.1022

Table III. Difference of Quantization Error Between the k-Means Algorithm and Other Methods

Problems	K	Median-Cut	Mean-Split	Our method
$m = 2$	8	4.6389	3.2243	0.3897
$N = 256 \times 256$	64	0.2677	0.1416	0.0084
$m = 3$	8	3.6087	2.2066	0.5118
$N = 256 \times 256$	64	2.9142	1.1978	0.1858
$m = 4$	8	5.6765	2.4881	0.6927
$N = 256 \times 256$	64	4.9269	4.6007	0.4379

5. CONCLUSIONS

We have presented an effective and efficient divisive algorithm for multivariate data clustering. This algorithm is designed to minimize the average sum-of-squared-errors.

We have successfully applied the proposed algorithm to color image quantization [11]. Our method is able to produce quantized images of higher quality than the other divisive methods. We have also shown that our results are very close to the local optimal solutions obtained from the k-means iterative procedure.

Our approach is applicable to a variety of applications such as pattern recognition, information retrieval, and intelligent information systems.

REFERENCES

1. BENTLEY, J. L. Multidimensional divide-and-conquer. *Commun. ACM 23*, 4 (Apr. 1980), 214–229.
2. BENTLEY, J. L., AND FRIEMAN, J. H. Data structure for range searching. *ACM Comput. Surv. 11*, 4 (Dec. 1979), 397–409.
3. DUDA, R. O., AND HART, P. E. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
4. FRIEMAN, J. H., BENTLEY, J. L., AND FINKEL, R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw. 3*, 3 (Sept. 1977), 209–226.
5. HALL, E. L. *Computer Image Processing and Recognition*. Academic Press, New York, 1979.
6. HARTIGAN, J. A. *Clustering Algorithms*. Wiley, New York, 1975.
7. HECKBERT, P. Color image quantization for frame buffer display. *ACM Trans. Comput. Gr. 16*, 3 (July 1982), 297–307.
8. HYAFIL, L., AND RIVEST, R. L. Construction optimal binary decision trees is NP-complete. *Inf. Process. Lett. 5* (May 1976), 15–17.

9. MacQueen, J. B. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkley Symposium on Mathematical Statistics and Probability 1* (1967), 281–297.
10. Selim, S. Z., and Ismail, M. A. K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Trans. Pattern Anal. Mach. Intell. PAMI-6*, 1 (1984), 81–87.
11. Wan, S. J., Wong, S. K. M., and Prusinkiewicz, P. Variance-based color image quantization for frame buffer display. Submitted for publication.
12. Wong, S. K. M., Wan, S. J., and Prusinkiewicz, P. Monochrome image quantization. Submitted for publication.
13. Wu, X., and Witten, I. H. A fast k-means type clustering algorithm. Dept. Computer Science, Univ. of Calgary, Canada, May 1985.

Received December 1987; accepted March 1988