01 Jan 1997

Evolutionary Computation Applied to Adaptive Information Filtering

Daniel R. Tauritz
Missouri University of Science and Technology, tauritzd@mst.edu

Ida G. Sprinkhuizen-Kuyper

Joost N. Kok

Recommended Citation

D. R. Tauritz et al., "Evolutionary Computation Applied to Adaptive Information Filtering," *Proceedings of the 9th Dutch Conference on Artificial Intelligence (NAIC'97) (1997, Antwerp, The Netherlands)*, pp. 17-26, Bolesian BV/NV, Jan 1997.

Evolutionary Computation applied to Adaptive Information Filtering

D.R. Tauritz, I.G. Sprinkhuizen-Kuyper, and J.N. Kok
Department of Computer Science, Leiden University
P.O. Box 9512, 2300 RA Leiden, The Netherlands
{dtauritz,kuyper,joost}@wi.leidenuniv.nl

Abstract

Information Filtering is concerned with filtering data streams in such a way as to leave only pertinent data (information) to be perused. When the data streams are produced in a changing environment the filtering has to adapt too in order to remain effective. Adaptive Information Filtering is concerned with filtering in changing environments. The changes may occur both on the transmission side (the nature of the streams can change), and on the reception side (the interest of a user can change).

Weighted trigram analysis is a quick and flexible technique for describing the contents of a document. A novel application of evolutionary computation is its use in adaptive information filtering for optimizing various parameters, notably the weights associated with trigrams. The research described in this paper combines weighted trigram analysis, clustering, and a special two-pool evolutionary algorithm, to create an Adaptive Information Filtering system with such useful properties as domain independence, spelling error insensitivity, adaptability, and optimal use of user feedback while minimizing the amount of user feedback required to function properly. We designed a special evolutionary algorithm with a two-pool strategy for this changing environment.

1 Introduction

Our goal is building an adaptive information filtering system which automatically filters incoming data streams to topics a specific user is interested in. In this paper we describe a case study whose purpose was to investigate whether evolutionary computation can be useful in AIF systems. In this case study we took as input articles from a fixed number of different Internet newsgroups and gave them to the system. The goal of the system is to cluster these articles in groups: that is, each article should be assigned a label which corresponds to a newsgroup. We propose an AIF system for this task based on the novel combination of weighted trigram analysis, incremental clustering, and evolutionary computation.

An incremental clustering algorithm is applied to weighted trigram representations of the documents creating a classification of the documents. In incremental clustering the number of clusters is not determined in advance, and can change over time [Moo88].

To find the right weights for the trigram analysis and some parameters of the cluster algorithm we designed an evolutionary algorithm. The most complex step in this AIF system is finding the (near) optimal weights for the trigrams: even using only the 26 letters of the Latin alphabet and the space as delimiter symbol, that still leaves 19683 (27^3) weights to optimize. This optimization problem is user dependent, so it cannot be performed a priori, and must also be able to adapt to the changing information needs of the user. Since the clustering has to be adaptive, the fitness of a trial solution can only be statistically approximated over time and will probably change over time. It takes some time before the fitness of a new member of the population is estimated. Therefore we splitted the population in two pools, one for the new "unproven" members, and one for the "adults".

The remainder of this paper consists of the following. In section 2 the representation of the documents as normalized trigram vectors is described. Section 3 describes the clustering algorithm. In section 4 the evolutionary computation algorithm we used is described. Section 5 contains a description of our experiments with Internet news and finally section 6 gives our conclusions.

2 Representation

A basic requirement for AIF is to be able to determine if an object differs "too much" from a user interest. This implies the necessity to be able to measure the distance between the objects and the user interests. To be able to measure the distance we need to transform the objects and the user interests to a common space and define a metric for that space. A space in which both objects and user interests are expressed in terms of their semantics would be ideal. However, as even the semantical analysis of a "simple" object such as written text is still an open problem in computer science [Sch93], this is currently not feasible. An alternative is to choose as common space a syntactical space. A practical choice for common space is a vector space where the vectors describe certain syntactical characteristics of the objects and the user interests. A vector space is a practical choice because many well known distance measures exist for it, such as Euclidean distance.

The difficulty with syntactical transformations lies in the problem of choosing one for which the distance measure in the vector space reflects the semantical distance as accurately as possible. Two possible transformations for written text are keywords [Rij79] and n-grams [Hee82]. Because of problems in using keywords in a changing environment we chose to use n-grams for the representation. In n-gram analysis [Hee82] the n stands for a positive integer. In 2-gram analysis the occurrence of pairs of letters is determined, in 3-gram analysis that of triplets (in this context called trigrams), etc. The larger n, the more accurately the distances between n-gram vectors correspond with the semantical distances of the original documents [Sch93]. The price paid for larger values of n is an exponentially increasing demand for memory, namely proportional to 27^n, see figure 1.

For the memory demands are still moderate enough for practical use, $n = 4$ is for the moment only interesting for research purposes, and $n > 4$ is currently not of any use. In practice this means only $n = 3$ is normally used as the results for $n < 3$ are not accurate enough to be of any use [ST88]. The great advantage of trigram analysis is that it is domain independent and the set is small enough to be used entirely so that

n	1	2	3	4	5
# elements	27	729	19683	531441	14348907

Figure 1: $N-$gram analysis memory requirements for various values of n.

the problem of having to expand the set because of domain changes is circumvented.

Trigram analysis alone does not provide sufficient discriminating power to accurately cluster textual documents, but weighted trigram analysis does. In weighted trigram analysis each trigram is assigned a weight. The discriminating power can be substantially increased by choosing the right weights.

It is to be hoped when comparing two textual documents of the same topic, that they will be considered similar enough to be grouped together, independent of the length of each document. However, the difference in frequency between documents is clearly length dependent. We solved this problem by normalizing the vectors using the cumulative frequency as normalizer.

While both keyword analysis and trigram analysis work to a certain degree, they are far from perfect (as one would expect of any syntactical transformation). Techniques which are able to help increase the discriminatory power of transformations from the space of textual documents to syntactical spaces are thus of great utility. According to [Sch93] such techniques are in fact of such importance that without them trigram analysis is not effective. They have in common that they can be applied before either keyword analysis or trigram analysis. In the case study of this paper the technique of stop lists was utilized. Filtering out common words (called 'stop' or 'fluff' words such as 'the', 'a', 'it', 'from', etc.) is computationally non-intensive and has two great advantages. First of all, it typically reduces the length of documents by between 30 and 50 per cent [Rij79]. And secondly, it increases the discrimination between documents.

3 Clustering

A clustering algorithm is an algorithm which takes as input a set of objects and produces as output a set of clusters and a mapping of each input object to a cluster. There are many clustering algorithms which all differ with respect to complexity, performance, and clustering properties. An important distinction can be made between clustering algorithms which require the whole set of objects to cluster in advance and those which can process one object at a time and present the results of the clustering after each newly processed object. The first class of algorithms will further be referred to as batch clustering algorithms, the second as incremental clustering algorithms. As a typical AIF system has a perpetual incoming data stream, we will use an algorithm of the second class. For an overview of clustering algorithms see for example [Bac95].

As our case study was concerned with the clustering of textual documents, we define an object as a textual document which we will consider to be a list of one or more lines, each line in its turn being composed of words separated by exactly one space (acting as separator symbol) and each word consisting of one or more lowercase letters of the Latin alphabet. We further restrict this simple class of textual

Step 1 – Initialization
Start with no cluster prototype vectors
Step 2 – Present and transform new object
Let $V = (v_1, v_2, \ldots, v_n)$:= normalized trigram analysis (new object)
Step 3 – Find the closest cluster (if any exist)
Find the $C = (c_1, c_2, \ldots, c_n)$ to minimize $d(C, V) = \sqrt{\sum_i (w_i(c_i - v_i))^2}$
Step 4 – Is the closest cluster close enough?
If $d(C, V) > r$, or if there are no cluster prototype vectors yet, then create a new cluster, with prototype vector equal to V; goto step 2
Step 5 – Update a matched cluster
Let $C := (1 - \lambda)C + \lambda V$; goto step 2

Figure 2: The cluster algorithm

documents by requiring the documents to be syntactical and semantically correct English non-fiction. The transformation is the normalized weighted trigram analysis, which is discussed in section 2.

Thus a common vector space has been defined to which all the objects to be clustered are transformed. The clusters are represented by hyperglobes in the common vector space. A hyperglobe is described in terms of its center, represented by a vector in the common vector space, called a prototype vector, and a parameter r indicating its radius, which is the same for all clusters.

The cluster algorithm used is described in Figure 2.

4 Evolutionary computation

Generally speaking any AIF system using a weighted vector representation has to deal with the problem of finding optimal values for the weights. Other parameters, such as the radius and the adaptation parameter of the cluster algorithm, also require optimization. The particular set of weight values greatly influences the clustering process. The user provides the system with feedback indicating satisfaction or dissatisfaction with the clustering of a document. This feedback is essential for the system to be able to "learn" to predict in the future with higher accuracy how the user wants the documents to be clustered. It is important to note that because of the changing needs of users, the system will always need to have the ability to adapt to those changes. In other words, we need a learning system capable of optimization in a perpetually changing environment. The required adaptive behavior of such a system can be facilitated by evolutionary computation.

Evolutionary computation is the process of finding (near) optimal solutions for computational problems using methods inspired by evolution theory. In our experiments the evolutionary algorithm is required to work incrementally. This has two im-

portant consequences. First, the fitness of a trial solution cannot be calculated, only statistically approximated over time. And, secondly, as the environment changes, the fitness of a trial solution may change too. In other words, the fitness being estimated over time will probably change over time, further complicating the task.

The elements of the population are vectors of weights extended with fields for the radius and the learning parameter for the cluster algorithm. The weights are represented by integers. The fitness of an element at any given time is defined by dividing its score through its age. The age is defined by the number of articles that have been evaluated by the element of the population up to that moment. The score is the number of times that it correctly classified an article. This means that the score is never larger than the age, which results in the welcome property that the range of the fitness values is limited to the interval between zero and one. This fitness function also allows the fitness of members of various ages to be meaningfully compared.

The reliability of the fitness value of an element at a particular moment depends on the age of that element. This is, however, not reflected in the fitness function. The solution adopted was to split the population into two pools, one for the new unproven members with ages below a certain value, and one for the adults. For each member of the first pool the fitness is only an estimation. This takes some time, and only if a member has reached a certain age, it is removed from this pool and put in the pool of adults. Hence one can view the first pool as the waiting room for the second pool. A fixed number of children is produced, but never more than there were members in the adult pool. They are not put directly in the adult pool, but have to wait some time until their fitness estimation is more reliable. We worked with a fixed size for the total population, so the worst adults are removed from the population in order to add to the pool of children. So both pools of children and adults can vary in size, but their combination is always reduced to the original population size.

The only genetic operator used is mutation. We also tried 0.5-uniform crossover; no benefits were found for our application. A single application of the mutation operator causes Gaussian noise (mean zero, unit variance) to be added to all the weight representing vector elements of the population member being mutated. The radius and the learning parameter of the cluster algorithm are mutated by adding Gaussian noise with a variance of 0.02 and 0.05, respectively. Selection of a parent from the adult pool is implemented as follows. First the member with the highest fitness gets a chance to be selected. Selection is performed by choosing a random number between zero and one and comparing it with the selective pressure parameter. If it is smaller the current member is selected, otherwise the member with the next highest fitness becomes the current member and the process is repeated by again choosing a random number between zero and one. Applying the mutation operator one time to the selected parent creates a new member which is added to the child pool, and after reaching the age threshold is moved to the adult pool. The values for system parameters such as population size, selective pressure parameter, age threshold and replacement rate (the number of adults which are replaced with children for each evaluation) were chosen after trying a number of random values in test runs. The population size was 100, selective pressure parameter 0.1, age threshold 10 and replacement rate 5 (which means the population stabilizes at 50% adults and 50% children).

5 Experiments

In this section we describe the results of our experiments. Before we give the results we discuss some problems associated with Internet news and how we obtained our data.

There are a number of problems associated with Internet news. First of all one has to find the appropriate newsgroup. If the topic does not have a dedicated newsgroup it may be split over a large number of groups. Another problem is that the more popular groups have such a high traffic rate that finding the information one is interested in can be very time consuming. To deal with these problems the ideal system would be one which can rearrange messages according to the specific topic interests of the user and filter out all non-relevant data. This is precisely what an AIF system attempts to do.

The goal of the case study was to ascertain if constructing the AIF system using the proposed combination of normalized weighted trigram analysis, incremental clustering, and evolutionary computation for parameter optimization, is viable. In this case study we want to show that weighted trigram analysis has sufficient discriminating power to separate a dynamic stream of documents belonging to multiple topics. To this end it was necessary to design and implement a system capable of incrementally clustering a dynamic stream of documents with the capability of dynamic parameter optimization in a non-stationary environment. For such a system to be successful two important properties are further necessary, namely, scalability of the number of topics to be separated and sufficient generalizing power of the weighted trigram analysis not to degrade the accuracy of the clustering too much when new documents are presented with which the system was not trained. And last but not least, the system must also be able to incorporate user feedback.

The case study consisted of two parts. The first without user feedback, the second with. There were two main reasons for this. First of all, to be able to measure the performance of the system many thousands of documents needed to be presented which is not feasible if user feedback is required for all of them. And secondly, the system first needs to be trained to reach an adequate level of performance before it becomes operational. Below first the basic architecture of the system will be discussed followed by the specifics of the part without user feedback, including results. The description of our demonstrator with user feedback can be found in [Tau96].

The Internet news articles were acquired in this experiment via a TCP/IP connection, employing a TCL script to retrieve them from a NNTP server. To facilitate the processing of the retrieved documents, which are in ASCII format, they are first converted to the set of all lower-case letters and the space denominator. This is done by converting all upper-case letters to lower-case, and all non-letters between two letters to one single space denominator. Only the actual text of the news articles was considered, not the article headers. As last preprocessing step all the words included in the stop list employed are filtered out, thus enhancing the discriminatory power of the clustering algorithm. The stop list employed can be found in [Tau96]. After the preprocessing trigram analysis is performed upon the preprocessed files creating document vector files representing the articles.

In the experimental setup without user a substitute was necessary to determine if a specific clustering was correct or not. This was done by creating a file when

```
For each article {
  For each member {
    Apply clustering algorithm with current member
      to the current article to update the fitness
      of the current member, but without adaptation
      of the prototype vectors
  }
  Apply clustering algorithm with the fittest member
    to the current article to adapt the prototype vectors
}
```

Figure 3: Dynamic topic separation algorithm

retrieving a batch of news articles, in which, for each article, the newsgroup from which it had been retrieved was indicated. All the trial solutions were evaluated for each article by using them to cluster that article. The new score was determined by comparing the assigned cluster with the correct one. Two important assumptions were made. First of all, that the articles retrieved from a specific newsgroup showed greater similarity to each other than to articles from the other newsgroups. And secondly, that all the articles belonging to a particular newsgroup were suited to be clustered together. To fulfil these assumptions as accurately as possible newsgroups were chosen for the experiments which were moderated. Moderated newsgroups are newsgroups for which articles cannot be posted freely, but must be approved by a human moderator who tries to filter out non-relevant postings. One experiment was done with an unmoderated newsgroup.

The experiments were initialized by creating the start clusters through computing document vector averages of the first n documents (values tried for n ranged from 10 to 30), and the weight vectors at random but within ranges equal to those used in the parameter file. The basic algorithm of these experiments is shown in figure 3. Figures 4 to 6 show the graphs of some experiments conducted. The graphs are offered in sets showing the average fitness and the highest fitness of the population. The x-axis indicates the number of the article being presented, the y-axis the average or highest population fitness.

In the first set the results of the two newsgroup experiment are shown. The task was to split the newsgroups `misc.news.bosnia` (moderated news about Bosnia) and `misc.news.southasia` (moderated news about south Asia). After presenting the training set (200 articles) about ten times the system seems to converge to an average accuracy rate in excess of 90% the highest fitness even to around 100 % (see figure 4).

The three newsgroup experiment had as task to split the newsgroups `misc.news.bosnia`, `sci.military.moderated` (moderated discussions of military warfare), and `comp.os.os2.announce` (moderated announcements concerning the operating system OS/2). After presenting the training set (200 articles) about ten times the system seems to converge to an average accuracy rate in excess of 90%, the highest fitness to around 95% (see figure 5).

The next set shows the results of the four newsgroup experiment. The task

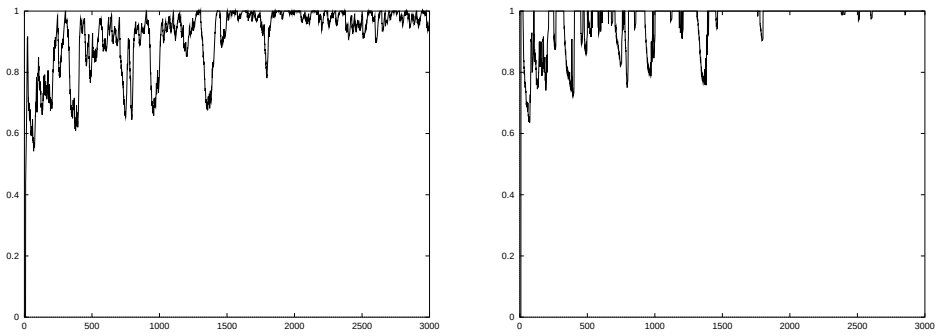

Figure 4: Two newsgroup experiment: The average fitness values (left) and the highest fitness values (right).

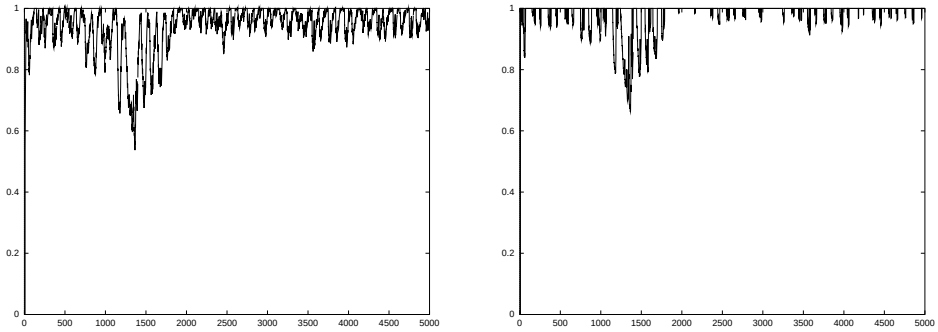

Figure 5: Three newsgroup experiment: The average fitness values (left) and the highest fitness values (right).

was to split the newsgroups `misc.news.bosnia`, `misc.news.southasia`, `comp.os.os2.announce`, and `comp.lang.java.announce` (moderated announcements concerning the programming language Java). After presenting the training set (200 articles) a large number of times, the system seems to converge to accuracy rates in excess of 90% (figure 6), with the exception of dips from time to time. These are probably in part due to the varying quality of the documents being processed, but randomness also seems to play a role, possibly indicating an instability due to the population getting stuck in local optima from time to time. Notice that the time to converge is quite a bit higher than with the previous experiments with fewer newsgroups.

To see what would happen when an unmoderated newsgroup with a very low S/N ratio is presented to the system, the unmoderated newsgroup experiment was conducted (400 articles). The task was to split the newsgroups `misc.news.bosnia`, `misc.news.southasia`, and `comp.os.ms-windows.nt.setup.hardware` (unmoderated news about setup problems with Microsoft's Windows NT). As expected the addition of an unmoderated newsgroup lowered the accuracy rate.

The final experiment was to determine if the system is capable of generalizing from what it has learned during the processing of a training set, and applying that knowledge to separating a test set. To this end the weight vector files produced by the three newsgroup experiment were taken as starting point, and then a test set (200

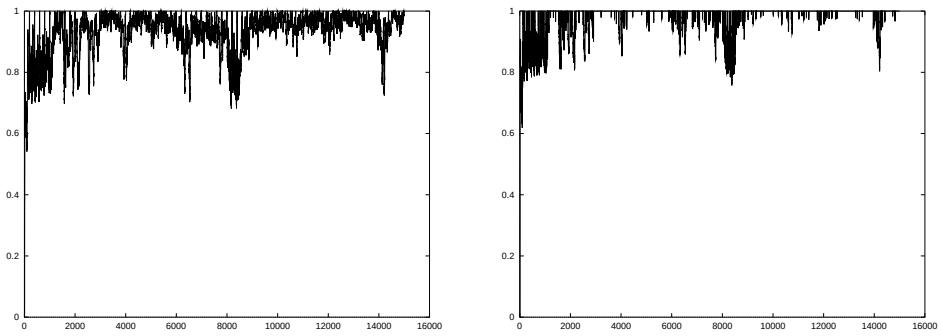

Figure 6: Four newsgroup experiment: The average fitness values (left) and the highest fitness values (right).

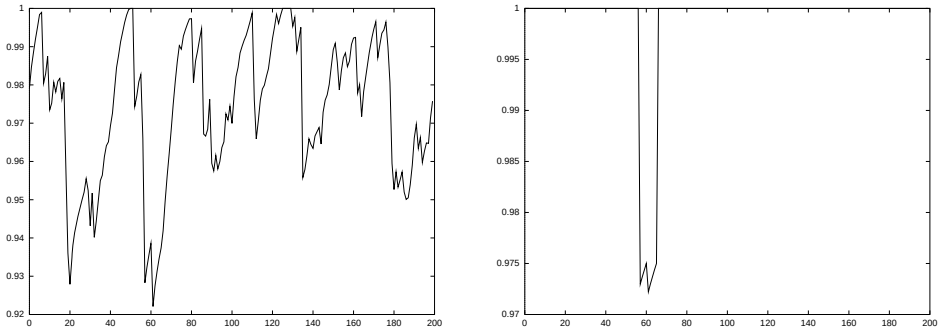

Figure 7: Test set experiment: The average fitness values (left) and the highest fitness values (right, note the higher resolution).

articles) offered, consisting almost entirely of new documents retrieved from the same newsgroups as used in the three newsgroup experiment. The results (figure 7) indicate accuracy comparable to the training set.

6 Conclusions

A combination of weighted trigram analysis, clustering, and evolutionary computation is sufficient to separate a static set of textual documents. The results as presented in section 5 of this paper show that an AIF system based on this approach is also capable of accurately separating a dynamic stream of documents. Furthermore, the experiments with a varying number of clusters indicate that increasing the number of clusters only effects the time needed to converge, not the accuracy rate, which would mean that the system is scalable. Also, the experiment with a test set as shown in figure 7 indicates that after sufficient training the system is capable of processing untrained documents with an accuracy rate comparable to that of processing the trained documents. This means that the system successfully generalizes.

The system in its present state works well. Of course, it is sensitive to the quality of the presented documents, as the experiment with the unmoderated newsgroup

demonstrated. One has to take into consideration, however, that in the dynamic topic separation experiments the system was not allowed to add new clusters. In an experiment where this will be allowed it might very well be that the "junk" documents will be clustered separately and the accuracy rate not or only slightly degrades.

It is interesting to see that trigram analysis is not restricted to text: also images and digital audio can be considered. For example, in image analysis trigrams of values of neighboring pixels can be used [HPL96]. Hence a major advantage of our method is that it probably generalizes to multimedia documents.

To conclude, the initial incarnation of this AIF system shows great promise. Clearly further research is warranted.

References

[Bac95] Eric Backer, *Computer-assisted reasoning in cluster analysis* (Prentice Hall, 1995).

[Hee82] T. D. Heer, The application of the concept of homeosemy to natural language information retrieval, *Information Processing and Management* 18(5) (1982) 229–236.

[HPL96] D. Huijsmans, S. Poles, and M. Lew, 2d pixel trigrams for content-based image retrieval, in A. Smeulders and R. Jain, eds., *Image Databases and Multi-media search: Proceedings first International Workshop* (1996).

[Koh90] T. Kohonen. The self-organizing map, in *Proceedings of the IEEE* Vol.78 (1990) 1464–1480.

[Moo88] B. Moore, Art 1 and pattern clustering, In D. Touretzky, G. Hinton, and T. Sejnowski, eds., *Proceedings of the 1988 Connectionist Models Summer School* (Morgan Kaufmann, San Mateo, CA, 1989) 174–185.

[Rij79] C. J. van Rijsbergen, *Information Retrieval*, 2nd edition (Butterworths, London, 1979).

[ST88] Stephanie Schmidt, and Bernd Teufel, Full text retrieval based on syntactic similarities, *Information Systems* Vol.13, No.1 (1988) 65-70.

[Sch93] Johannes Cornelis Scholtes, *Neural Networks in Natural Language Processing and Information Retrieval*, Ph.D. thesis, University of Amsterdam (1993).

[Tau96] Daniel R. Tauritz, *Concepts of Adaptive Information Filtering*, Internal Report 96-19, Department of Computer Science, Leiden University, The Netherlands. Available via http://www.wi.LeidenUniv.nl/home/dtauritz (1996).