A Comparative Study of Compound Critique Generation in Conversational Recommender Systems

Jiyong Zhang and Pearl Pu

Human Computer Interaction Group,
Ecole Polytechnique Fédérale de Lausanne (EPFL),
CH-1015, Switzerland
{jiyong.zhang, pearl.pu}@epfl.ch

Abstract. Critiquing techniques provide an easy way for users to feedback their preferences over one or several attributes of the products in a conversational recommender system. While unit critiques only allow users to critique one attribute of the products each time, a well-generated set of compound critiques enables users to input their preferences on several attributes at the same time, and can potentially shorten the interaction cycles in finding the target products. As a result, the dynamic generation of compound critiques is a critical issue for designing the critique-based conversational recommender systems. In earlier research the Apriori algorithm has been adopted to generate compound critiques from the given data set. In this paper we propose an alternative approach for generating compound critiques based on the multi-attribute utility theory (MAUT). Our approach automatically updates the weights of the product attributes as the result of the interactive critiquing process. This modification of weights is then used to determine the compound critiques according to those products with the highest utility values. Our experiments show that the compound critiques generated by this approach are more efficient in helping users find their target products than those generated by the Apriori algorithm.

Keywords: conversational recommender system, critiquing, compound critique, multi-attribute utility theory, interaction cycle.

1 Introduction

Critiquing techniques have proven to be a popular and successful approach in conversational recommender systems because it can help users express their preferences and feedbacks easily over one or several aspects of the available product space[1][2][3][4]. The simplest form of critiquing is unit critiquing which allows users to give feedback on a single attribute or feature of the products at a time[1]. For example, *[CPU Speed: faster]* is a unit critique over the *CPU Speed* attribute of the PC products. If a user wants to express preferences on two or more attributes, multiple interaction cycles between the user and the system

V. Wade, H. Ashman, and B. Smyth (Eds.): AH 2006, LNCS 4018, pp. 234–243, 2006.

are required. To make the critiquing process more efficient, a wise treatment is to generate compound critiques dynamically to enable users to critique on several attributes in one interaction cycle[2][3]. Typically, for each interaction cycle there are a large number of compound critiques available. However, the system is able to show only a few of them on the user interface. Thus a critical issue for recommender systems based on compound critiques is to dynamically generate a list of high quality compound critiques in each interaction cycle to save the users' interaction effort.

McCarthy et al.[5] have proposed a method of discovering the compound critiques through the Apriori algorithm used in the *market-basket analysis* method [6]. It treats each critique pattern as the shopping basket for a single customer, and the compound critiques are the popular shopping combinations that the consumers would like to purchase together. Based on this idea, Reilly et al.[2] have developed an approach called dynamic critiquing to generate compound critiques. As an improved version, the incremental critiquing[3] approach has also been proposed to determine the new reference product based on the user's critique history. A typical interaction process of both dynamic critiquing and incremental critiquing approach is as follows. First the system shows a reference product to the user. At the same time the system generates hundreds of compound critiques from the data set via the Apriori algorithm, and then determines several of them according to their support values for the user to critique. After the user's critique is chosen, the system then determines a new reference product and updates the list of critiques for the user to select in the next interaction cycle. This process continues until the target product is found.

The Apriori algorithm is efficient in discovering compound critiques from a given data set. However, selecting compound critiques by their support values may lead to some problems. The critiques determined by the support values can only reveal "what the system would provide," but cannot predict "what the user likes." For example, in a PC data domain if 90 percent of the products have a faster CPU and larger memory than the current reference product, it is unknown whether the current user may like a PC with a faster CPU and larger memory. Even though the system based on the incremental critiquing approach maintains a user preference model to determine which product to be shown in the next interaction cycle, some good compound critiques may still be filtered out before the user could choose because their support values do not satisfy the requirement. If the users find that the compound critiques cannot help them find better products within several interaction cycles, they would be frustrated and give up the interaction process. As a result, a better approach for generating compound critiques should allow the users to gradually approach the products they preferred and to find the target products with less number of interaction cycles.

In this paper we argue that determining the compound critiques based on the user's preference model would be more efficient in helping users find their target products. We propose a new approach to generate compound critiques for conversational recommender systems with a preference model based on

multi-attribute utility theory(MAUT)[7]. In each interaction cycle our approach first determines a list of products via the user's preference model, and then generates compound critiques by comparing them with the current reference product. In our approach, the user's preference model is maintained adaptively based on user's critique actions during the interaction process, and the compound critiques are determined according to the utilities they gain instead of the frequency of their occurrences in the data set. We also carry out a set of simulation experiments to show that the compound critiques generated by our approach can be more efficient than those generated by the Apriori algorithm.

This paper is organized as follows. The related research work is reviewed in section 2. Section 3 describes our approach of generating compound critiques based on MAUT. Section 4 reports a set of simulation experiments to compare the performance of various critique approaches. Discussions and conclusions are given in section 5 and 6 respectively.

2 Related Work

Other than the unit critiquing and compound critiquing approaches that we have mentioned, a number of various critiquing approaches based on examples also have been proposed in recent years. The ATA system [8] uses a constraint solver to obtain a set of optimal solutions and shows five of them to the user (three optimal ones and two extreme solutions). The Apt Decision [9] uses learning techniques to synthesize a user's preference model by critiquing the example apartment features. The SmartClient approach[10] gradually refines the user's preference model by showing a set of 30 possible solutions in different visualizations to assist the user making a travel plan. The main advantage of these example-based critiquing approaches is that users' preferences can be stimulated by some concrete examples and users are allowed to reveal preferences both implicitly (choosing a preferred product from a list) and explicitly (stating preferred values on specific attributes). In fact, these example-based critiquing approaches can also "generate" compound critiques easily by comparing the critique examples with the current recommended product. But they are more viewed as tradeoff navigation because users have to state the attribute values that they are willing to compromise against those that they are hoping to improve[11]. The approach of generating compound critiques that we proposed here can be regarded as an example-based critiquing approach because we determine the compound critiques from a list of critique examples. However, the difference is that our approach concentrates on constructing user's preferences automatically through the choice of the compound critiques, and the user can save some effort in stating the specific preferences values during the interaction process.

Generating *diverse* compound critiques is also an important issue for conservational recommender systems as a number of researchers have pointed out that diversity has the potential to make the interaction more efficient[12][13][14]. For example, in [14] diversity is enhanced by reducing the overlap among those compound critiques generated by the dynamic critiquing approach. In our approach

PM— user's preference model; *ref*— the current reference product;
IS— item set; *CI*— critique items; *CS*— critique strings; *U*— utility value;
β— the weight adaptive factor

```
    //The main procedure
 1. procedure Critique_MAUT ( )
 2.    PM = GetUserInitialPreferences ( )
 3.    ref = GenInitialItem (PM)
 4.    IS ⟵ all available products – ref
 5.    while not UserAccept (ref)
 6.       CI = GenCritiqueItems (pm, IS)
 7.       CS = GenCritiqueStrings (ref, CI)
 8.       ShowCritiqueInterface (CS)
 9.       id = UserSelect (CS)
10.       ref' = CI_id
11.       ref ⟵ ref'
12.       IS ⟵ IS – CI
13.       PM = UpdateModel (PM, ref)
14.    end while
15.    return

    //user select the critique string
16. function UserSelect (CS)
17.    cs = the critique string user selects
18.    id = index of cs in CS
19.    return id

    //select the critique items by utilities
20. function GenCritiqueItems (PM, IS)
21.    CI = {}
22.    for each item O_i in IS do
23.       U(O_i) = CalcUtility(PM, O_i)
24.    end for
25.    IS' = Sort_By_Utility (IS, U)
26.    CI = Top_K (IS')
27.    return CI

    //Update user's preferences model
28. function UpdateModel(PM, ref)
29.    for each attribute x_i in ref do
30.       [pv_i, pw_i] ⟵ PM on x_i
31.       if (V(x_i) ≥ pv_i)
32.          pw'_i = pw_i × β
33.       else
34.          pw'_i = pw_i / β
35.       end if
36.       PM ⟵ [V(x_i), pw'_i]
37.    end for
38.    return PM
```

Fig. 1. The algorithm of critiquing based on MAUT

the weight of each product attribute is revised adaptively during the critiquing process, thus we believe that there is a certain degree of diversity between the compound critiques determined by our approach. The detail investigation of generating diverse compound critiques will be left in our future work.

3 Generating Compound Critiques Based on MAUT

MAUT[7] is a well known and powerful method in decision theory for ranking a list of multi-attribute products according to their utilities. Here we only use its simplified weighted additive form to calculate the utility of a product $O = \langle x_1, x_2, ..., x_n \rangle$ as follows:

$$U(\langle x_1, \cdots, x_n \rangle) = \sum_{i=1}^{n} w_i V_i(x_i) \tag{1}$$

where n is the number of attributes that the products may have, the weight $w_i (1 \leq i \leq n)$ is the importance of the attribute i, and V_i is a value function of the attribute x_i which can be given according to the domain knowledge during the design time.

The general algorithm of the interaction process with this proposed approach (called Critique_MAUT) is illustrated by Figure 1. We use a preference model which contains the weights and the preferred values for the product attributes to represent the user's preferences. At the beginning of the interaction process, the initial weights are equally set to $1/n$ and the initial preferences are stated by the user. In each interaction cycle, the system generates a set of critique strings for the user to select as follows. Instead of mining the critiques directly from the data set based on the Apriori algorithm, the Critique_MAUT approach first determines top K (in practice we set $K = 5$) products with maximal utilities, and then for each of the top K products, the corresponding critique string is determined by comparing it with the current reference product. This "from case to critique pattern" process of producing compound critique strings is straightforward and has been illustrated in [5].

After the user has selected one of the critique strings, the corresponding critique product is assigned as the new reference product, and the user's preference model is updated based on this critique selection. For each attribute, the attribute value of the new reference product is assigned as the preference value, and the weight of each attribute is adaptively adjusted according to the difference between the old preference value and the new preference value. If the new preference value is equal or better than the old preference value, the current weight on the given attribute is multiplied by a factor β, otherwise it is divided by β (See line 30-36 on Figure 1). In practice we set the factor $\beta = 2.0$. Based on the new reference product and the new user preference model, the system is able to recommend another set of critique strings for the user to critique until the user finds the target product or stops the interaction process.

Figure 2 shows a screen shot of a personal computer recommender system that we have developed based on the proposed approach. In this interface, the user can see the detail of a reference product, and he or she can either conduct a unit critique or a compound critique to reveal additional preferences. It is very similar to the user interface proposed in [3] except that we show 5 different compound critiques generated by our approach in each interaction cycle.

4 Experiments and Results

We carried out a set of simulation experiments to compare the performance of the basic unit critiquing approach (Critique_Unit), the incremental critiquing approach which generates compound critiques with the Apriori algorithm (Critique_Apriori)[3], and our approach generating compound critiques based on MAUT (Critique_MAUT). In the The experiment procedure is similar to the simulation process described in [3] except for two differences. One is that we assume at the beginning the user will reveal several preferences to the system. We observed that an average user states about 3 initial preferences. Thus we randomly determine the number of the initial preferences from 1 to 5. Another difference is that in each interaction process we simply appoint a product as the target product directly instead of the *leave-one-out* strategy. Both the Critique_Apriori

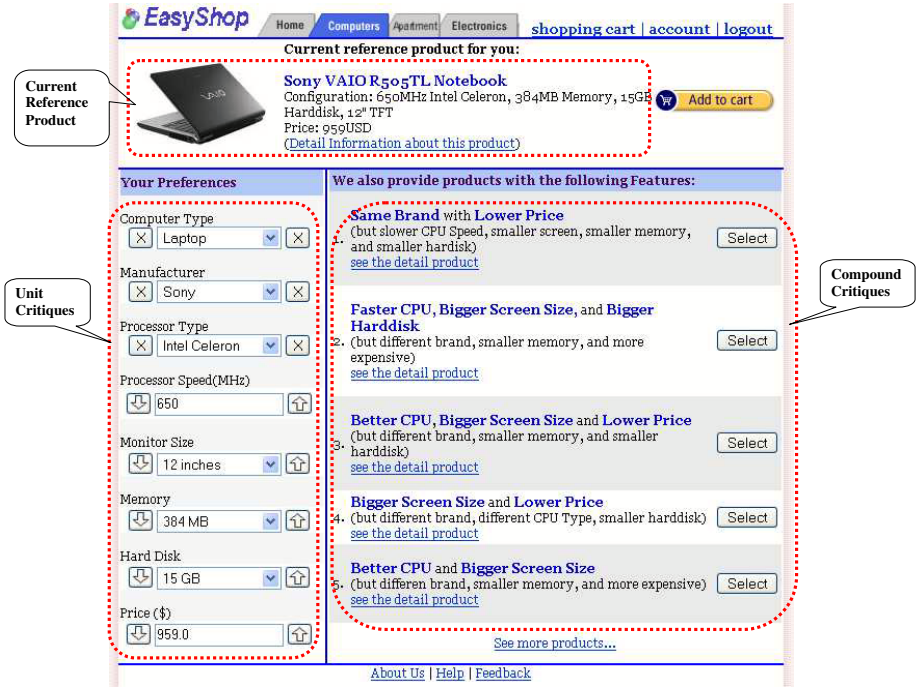

Fig. 2. The user interface with both unit and compound critiques

and the Critique_MAUT approaches generate 5 different compound critiques for user to choose in each interaction cycle. The Critiue_Apriori approach adopts the *lowest support* (LS) strategy with a minimum support threshold of *0.25* to generate compound critiques. In our experiments each product in the data set is appointed as the target choice for 10 times and the number of interaction cycles for finding the target choice are recorded. Two different types of data set are utilized in our experiments. The apartment data set used in [11] contains 50 apartments with 6 attributes: *type*, *price*, *size*, *bathroom*, *kitchen*, and *distance*. The PC Data set [2] contains 120 PCs with 8 different attributes. This data set is available at *http://www.cs.ucd.ie/staff/lmcginty/PCdataset.zip*.

Figure 3 (1) and (2) show the average interaction cycles of different approaches. Compared to the baseline Critique_Unit approach, the Critique_Apriori approach can reduce the average interaction cycles by 15% (for apartment data set) and 28% (for PC data set) respectively. This validates earlier research that the interaction cycles can be reduced substantially by utilizing compound critiques. Moreover, the results show that the proposed Critique_MAUT approach can reduce the interaction cycles over 20% compared to the Critique_Apriori approach (significant difference, $p < 0.01$).

We define the *accuracy* as the percentage of finding the target product successfully within a certain number of interaction cycles. As shown in Figure 3 (3)

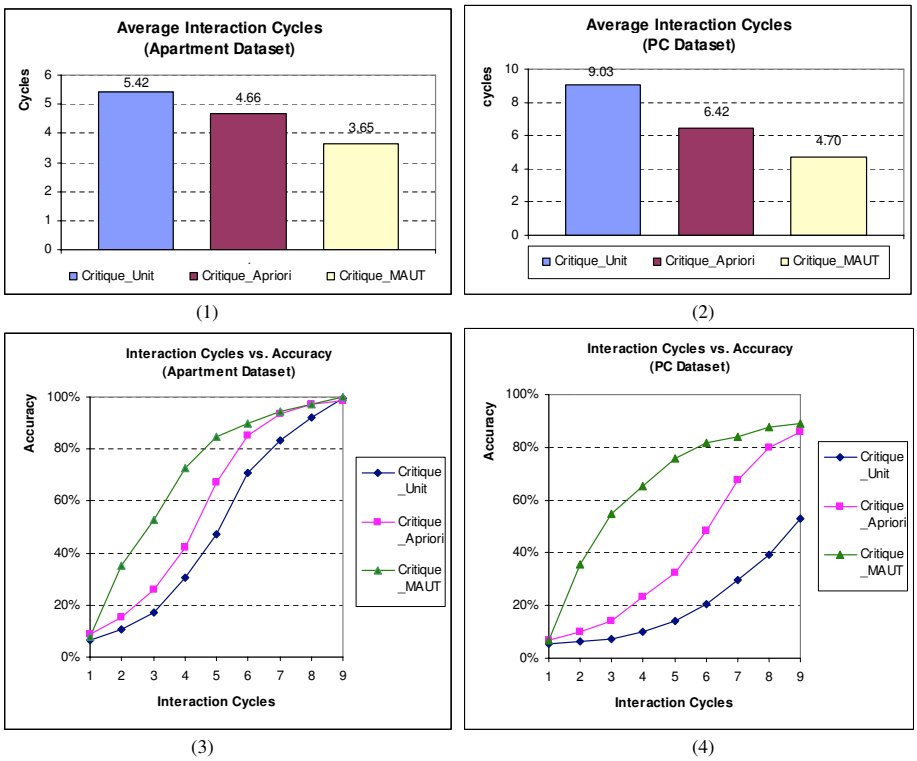

Fig. 3. The results of the simulation experiments with the PC data set and the apartment data set. (1)The average interaction cycles for the apartment data set; (2)The average interaction cycles for the PC data set; (3) the accuracy of finding the target choice within given number of interaction cycles for the apartment data set; (4) the accuracy of finding the target choice within given number of interaction cycles for the PC data set.

and (4), the Critique_MAUT approach has a much higher accuracy than both the Critique_Unit and the Critique_Apriori approach when the number of interaction cycles is small. For example, in the apartment data set, when the user is assumed to make a maximum of 4 interaction cycles, the Critique_MAUT approach enables the user to reach the target product successfully 85% of the time, which is 38% higher than the Critique_Unit approach, and 18% higher than the Critique_Apriori approach.

Compound critiques allow users to specify their preferences on two or more attributes simultaneously thus they are more efficient than unit critiques. When the compound critiques are shown to the user, it is interesting to know how often they are applied during the interaction process. Here we also compared the application frequency of compound critiques generated by MAUT and the

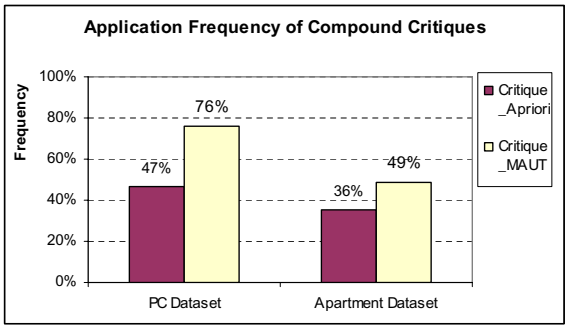

Fig. 4. Application frequency of compound critiques generated by MAUT and the Apriori algorithm

Apriori algorithm in our experiments. As shown in Figure 4, the application frequency of compound critiques generated by the Critique_MAUT method are much higher than those generated by the Critique_Apriori method for both the PC data set (29% higher) and the Apartment Data set (13% higher). We believe this result offers an explanation of why the Critique_MAUT method can achieve fewer interaction cycles than the Critique_Apriori method.

5 Discussions

The key improvement of the proposed Critique_MAUT approach is that the compound critiques are determined through their utility values given by MAUT instead of their support values given by the Apriori algorithm. Since a utility value measures the product's attractiveness according to a user's stated preferences, our approach has the potential to help the user find the target choice in an earlier stage. The simulation experiment results verified this advantage of the Critique_MAUT approach by comparing it with the Critique_Unit and the Critique_Apriori approaches.

Modeling user's preferences based on MAUT is not a new idea. In fact, MAUT approach can enable users to make tradeoff among different attributes of the product space. For example, Stolze has proposed the scoring tree method for building interactive e-commerce systems based on MAUT[15]. However, in our approach we designed an automatic manner to gradually update the user's preference model according to the critique actions. The users are not obliged to state their preference value or to adjust the weight value on each attribute explicitly thus the interaction effort can be substantially reduced.

There are several limitations in our current work. The user's preference model is based on the weighted additive form of the MAUT approach, which might lead to some decision errors when the attributes of the products are not mutually

preferentially independent.[1] If some attributes are preferentially dependent, our approach is still able to generate the compound critiques. However, the user needs to spend some extra effort to determine the utility function which is more complicated than equation (1). Furthermore, currently the experiments are based on artificial users with simulated interaction processes. We assume that the artificial user has a clear and firm target in mind during the interaction process. In reality this assumption is not always true because the user may change his or her mind during the interaction process. Moreover, our approach determines the compound critiques only based on utility values. Some researchers have pointed out that the approach of combining similarity and diversity may provide better performance[12]. So far we haven't compared the Critique_MAUT approach with the approach based on similarity and diversity. Nevertheless, in this paper the newly proposed Critique_MAUT approach can generate compound critiques which are more efficient in helping users find their target product than those compound critiques based on the Apriori algorithm.

6 Conclusions and Future Work

Generating high quality compound critiques is essential in designing critique-based conversational recommender systems. The main contribution of this paper is that we propose a new approach in generating compound critiques based on the multi-attribute utility theory. Unlike the popular method of generating compound critiques directly by the Apriori algorithm, our approach adaptively maintains the user's preference model based on MAUT during the interaction process, and the compound critiques are determined according to the utility values. Our simulation experiments show that our approach can reduce the number of interaction cycles substantially compared to the unit critiques and the compound critiques generated by the Apriori algorithm. Especially when the user is willing to make only a few interactions with the system, our approach enables the user with a much higher chance in finding the final target product. In the future, we plan to organize a set of real user studies to compare the performance of these critiquing approaches in terms of the actual number of interaction cycles as well as the degree of users' satisfaction. We will also further improve the performance of our approach by integrating a certain degree of diversities into the compound critique generation process.

Acknowledgments

Funding for this research was provided by Swiss National Science Foundation under grant 200020-103490. The authors thank the anonymous reviewers for their helpful comments.

[1] The attributes $X_1, \cdots, X_n$ are *mutually preferentially independent* if every subset Y of these attributes is preferentially independent of its complementary set of attributes[7].

References

1. Burke, R.D., Hammond, K.J., Young, B.C.: The FindMe approach to assisted browsing. IEEE Expert **12**(4) (1997) 32–40
2. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Dynamic critiquing. In Funk, P., González-Calero, P.A., eds.: Proceedings of the European Conference on Case-Based Reasoning (ECCBR-04). Volume 3155 of Lecture Notes in Computer Science., Springer (2004) 763–777
3. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Incremental critiquing. Knowledge Based Systems **18**(4-5) (2005) 143–151
4. Faltings, B., Pu, P., Torrens, M., Viappiani, P.: Designing example-critiquing interaction. In: International Conference on Intelligent User Interfaces, Island of Madeira (Portugal), ACM (2004) 22–29
5. McCarthy, K., Reilly, J., McGinty, L., Smyth, B.: On the dynamic generation of compound critiques in conversational recommender systems. In: Proceedings of the Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems(AH 2004). Volume 3137 of LNCS., Springer (2004) 176–184
6. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In Bocca, J.B., Jarke, M., Zaniolo, C., eds.: Proceedings of the 20th International Conference Very Large Data Bases(VLDB), Morgan Kaufmann (1994) 487–499
7. Keeney, R., Raiffa, H.: Decisions with Multiple Objectives: Preferences and Value Tradeoffs. John Wiley and Sons, New York (1976)
8. Linden, G., Hanks, S., Lesh, N.: Interactive assessment of user preference models: The automated travel assistant. In: Proceedings of the 6th International Conference on User Modeling (UM97). (1997)
9. Shearin, S., Lieberman, H.: Intelligent profiling by example. In: Proceedings of the Conference on Intelligent User Interfaces, ACM Press New York, NY, USA (2001) 145–151
10. Pu, P., Faltings, B.: Enriching buyers' experiences: the smartclient approach. In: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press New York, NY, USA (2000) 289–296
11. Pu, P., Kumar, P.: Evaluating example-based search tools. In: Proceedings of the ACM Conference on Electronic Commerce (EC'04), New York, USA (2004) 208–217
12. Smyth, B., McClave, P.: Similarity vs. diversity. In Aha, D.W., Watson, I., eds.: Proceedings of the 4th International Conference on Case-Based Reasoning (ICCBR). Volume 2080 of Lecture Notes in Computer Science., Springer (2001) 347–361
13. McGinty, L., Smyth, B.: On the role of diversity in conversational recommender systems. In Ashley, K.D., Bridge, D.G., eds.: Proceedings of the 5th International Conference on Case-Based Reasoning (ICCBR). Volume 2689 of Lecture Notes in Computer Science., Springer (2003) 276–290
14. McCarthy, K., Reilly, J., Smyth, B., McGinty, L.: Generating diverse compound critiques. Artificial Intelligence Review **24**(3-4) (2005) 339–357
15. Stolze, M.: Soft navigation in electronic product catalogs. International Journal on Digital Libraries **3**(1) (2000) 60–66