UNIVERSITÄT AUGSBURG

Context-Aware Preference Search for Outdoor Activity Platforms

Werner Kießling[1], Martin Soutschek[2], Alfons Huhn[1], Patrick Roocks[1], Markus Endres[1], Stefan Mandl[1], Florian Wenzel[1], and Andreas Zelend[1]

[1] Chair for Databases and Information Systems, University of Augsburg, Germany
[2] Alpstein Tourismus GmbH & Co. KG, Immenstadt, Germany

Report 2011-15 November 2011

INSTITUT FÜR INFORMATIK

D-86135 AUGSBURG

Supported by:

on the basis of a decision
by the German Bundestag

Context-Aware Preference Search for Outdoor Activity Platforms

Abstract

Complex application domains like outdoor activity platforms demand a powerful search interface that can adapt to personal user preferences and to changing contexts like weather conditions. Today most platforms offer a search technology known as Faceted Search, also named Parametric Search, where a user iteratively adapts his/her search parameters by a tedious and time-consuming trial-and-error process until the quality and quantity of the query results somehow corresponds to his/her expectations. This process gets even more cumbersome in mobile environments. Here we present a sophisticated approach called Preference Search, which we have prototypically implemented in a commercial outdoor activity platform. Preference Search replaces lengthy user sessions by one single user request. Technically, this request is automatically compiled into one single Preference SQL query, which efficiently retrieves those items that best match the user's expectations within the current context. A benchmark was applied to Faceted Search as well as Preference Search. The evaluation of the benchmark indicates that Preference Search substantially improves the user's search satisfaction in comparison to Faceted Search.

Keywords: personalization; context aware systems; outdoor activity; preference handling; query performance; customer satisfaction;

1 Introduction

Many e-Business enterprises, in particular within the tourism industry, maintain large databases of items with many variations. Several guidelines are used when searching such huge amount of data.

(1) Hierarchies classify the search objects leading to *Hierarchical Search* also known as *Navigational Search*. Taxonomies are established to implement a 'drill down' approach for narrowing the search process. Thus, users traverse a search tree during Hierarchical Search and may backtrack to higher-level concepts in case of error or dead-end. This process heavily relies on the correct interpretation of semantic concepts, since users must choose a semantic refinement of the current topic. For example, Hutchinson shows the benefits of Hierarchical Search for the user class of children and handicapped in 2003. Since this process is error-prone and time-consuming, flat hierarchies are favoured.

(2) Search objects have attributes, which describe or even distinguish the variants. The *Faceted Search* also known as *Parametric Search* (see Sacco et al. 2009) is a well-established technology where subspaces of the search space are specified by hard constraints on some of the attributes. Instead of traversing concepts, the search is now guided by features like prize, size, or colour. This process remains time-consuming by

adapting the criteria to produce ‘better’ results. In contrast to Hierarchical Search two new phenomena arise:

Information Flooding. If users express non-constraining criteria, the subspace is not reduced significantly. Disappointed by the result, users often overcompensate by trying to constrain the criteria too excessively. However, this behaviour may produce the effect of an empty result.

Empty Result. If users are over-constraining any criteria, the subspace may even be reduced to be empty. As a reaction, users relax any criterion, which may again result in information flooding.

Before judging the quality of the received result according to the search parameters, users take care of the quantity of the result. Having no information about data distributions of attributes, users are restricted to explore the data by trial-and-error.

(3) Schemata are already defined feeding a relational database. Obviously, the *Attribute-Based Search* paves the way to *perfect hits* by formulating SQL-statements. This paradigm is based on mathematical foundations of relations and declaratively describes the characteristics of the result set. Meanwhile, even complex types like multimedia data are supported (Feris et al. 2011). The main problem arises if perfect hits do not exist. Users receive an empty result enforcing them to reformulate the query, a procedure comparable to the Faceted Search. Therefore, a posterior relaxation or a prior vagueness is preferable. Instead of searching for perfect hits, the query should return a result containing *similar* articles, if perfect hits are missing. Clearly, the similarity should satisfy an optimality criterion: the result should only contain *the best matching objects*.

(4) Users are accustomed to *Full Text Search* as offered by Digital Libraries (Fox 1999) to search through the contents of books. Using only a bag of words as search parameters, the Full Text Search delivers those books, which are rated at highest score with regard to the input. Thus scoring expresses some kind of similarity.

The above-mentioned categorization is not exclusive. In practice, the advantages of different approaches are often combined. In this report we discuss the results of a running project where *Faceted Search* has been replaced by a sophisticated *Preference Search* paradigm. Preference handling in database has been a very active and productive research area recently (Stefanidis et al. 2011). Here we pursue the approach of extending standard SQL towards Preference SQL (Kießling 2002 and Kießling et al. 2011). Preference Search can easily deal with context-adaptive personal user preferences and also includes automatic optimization techniques, which unburden users from the hassle of trial-and-error induced by the Faceted Search.

The remainder of this report is organized as follows. In *Section 2* the domain-specific search requirements of outdoor activities are discussed. *Section 3* describes the used preference framework and the Preference SQL query language as required for the purposes presented here. Then in *Section 4* we evaluate our use case and discuss the

advantages of Preference Search over Faceted Search. *Section 5* describes the procedure how to compare Faceted Search to Preference Search by evaluating a benchmark. Finally in *Section 6* we present our conclusions and an outlook.

2 Search Requirements

The combination of search techniques mentioned in Section 1 can be found in the outdoor activity platform (www.outdooractive.com), with the current search mask depicted in Fig. 1.

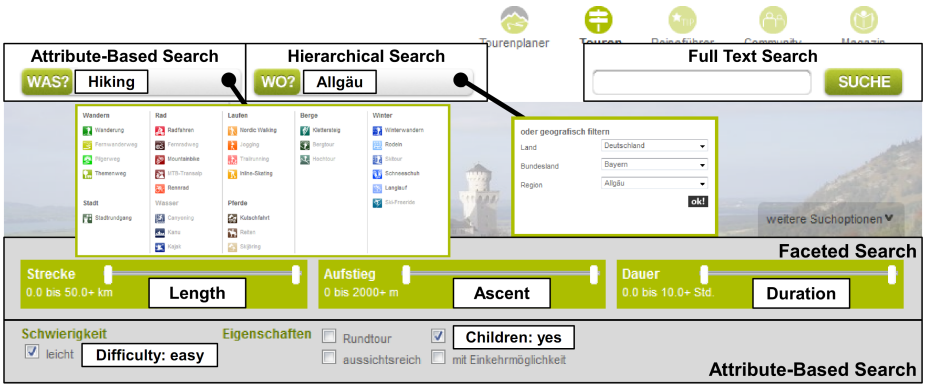

Fig. 1. Annotated Search Mask of Outdooractive

Users define the type of activity via Attribute-Based Search, the location via Hierarchical Search and further tour attributes via Faceted Search. Alternatively, tours can be found via Full Text Search by specifying meaningful keywords.
In an ongoing research project, this status quo is enhanced by replacing these search processes with a preference based search paradigm while keeping the changes to the human machine interface to a minimum. An integrated recommender component augments the user input in a context-sensitive and user-adaptive fashion, leading to a significant improvement in result quality and consequently user satisfaction. This is the first step towards a fully personalized one-click recommendation for outdoor enthusiasts.

2.1 Case Study

In order to grasp the inherent complexity of the outdoor domain, a case study is presented that describes Paul's endeavour of finding the perfect hiking tour.

Running Example. Paul definitively plans a hiking trip to the Allgäu region in the German Alps, following the recommendation of his friends. He considers himself as a tourist without practical experience in mountaineering. Nevertheless, he wants to go on an enjoyable hiking tour near by his lodging. He refers to an outdoor activity platform to find an appropriate route. Due to the lack of experience, Paul feels

insecure at first. An assistant function asks him to select the most suitable type of hiker matching his interests. The choices are *athlete*, *family*, *tourist*, or *bon vivant*. He immediately picks the tourist type. The system provides the following default values for Paul:

Region (*Allgäu*), activity (*Hiking*), duration (*4 h*), ascent (*200 m*), length (*10 km*).

Paul's input values concerning type of activity, destination region, and choice of tour difference are depicted in Fig. 1. As Paul is a community member, his username is also known after logging in. Without knowledge about which tours may be most suitable for him, Paul is worried about starting the time-consuming Faceted Search process of iterative trial-and-error to find *his best* tour.

2.2 Factors of Influence

This case study points out a plurality of factors that have an immediate effect on the suitability of a tour. While some factors are of personal nature, others arise from social contacts and the user context. At first, *intrapersonal factors* are considered.

Vagueness. As an inexperienced user, Paul is unsure which initial values for the search attributes to choose. While the search interface of Faceted Search might assist by allowing the input of search ranges instead of specific values for numeric attributes, only results are retrieved that are within that given range, thus forcing the user to choose input values carefully. There is nevertheless no *search tolerance*, which is common sense in everyday life.

Contradictions. Especially in the outdoor domain, dependencies between attributes exist, e.g. a correlation between distance and duration of a hiking tour. In addition, these dependencies are oftentimes activity-specific. Due to unawareness and a lack of experience, consumers often contradict themselves when entering values for correlated attributes. For a long distance hike, users may specify a far too short duration.

Roles. Roles describe people sharing common goals or behaviour. A classification of tourists into roles (Gibson et al. 2002) has been proven to be helpful in various touristic applications. While Paul is not able to state his personal interests in detail, he is able to determine which role best suits his self-image.

Topics. Topics are formed by aggregation of POIs (Points of Interest) with respect to specific needs. They are inspired by Maslov's *Hierarchy of Needs* (1943), forming categories like 'shelters' (e.g. alpine huts), 'food' (e.g. mountain inns), and 'infrastructure' (e.g. funiculars). Topics are summarized by super topics like 'fallback'. Context also triggers appropriate topics. E. g., shelters are appreciated if rainy weather is forecasted.

Additionally, *contextual* and *interpersonal factors* have to be considered.

Incompleteness. Outdoor activity platforms depend on providers for services and information. The delivered data may have the deficiency of *incompleteness*. Every search approach has to tackle incompleteness in a consistent and correct manner. Even the concept of 'Context' needs the handling of only partial knowledge, when sensors fail or a web service is unreachable in outdoor scenarios.

Context. Mountain weather is known for its unexpected changes. Suddenly, hiking tourists may be surprised by heavy rain or strong winds. For a better support of hikers with regard to current weather conditions, a context-aware component is needed to automatically anticipate weather changes in the near future.

Social Networks. Finally, Paul's first impulse to choose the Allgäu region was influenced by recommendations of his friends. The *social network* of a person gives valuable hints which tours should be preferred. As a person knows the performance of a peer-group of friends, users may just rely on the estimations of their peer-group.

2.3 Derived Objectives

Objectives can be derived from the presented influencing factors to tackle the challenges of search processes in general and the outdoor domain in particular.

Vagueness and contradictions as well as incompleteness are leading to *information flooding* or *empty results*. Thus, a paradigm change from perfect matches to best matching objects has to take place. A system retrieving the best matches for a given search query facilitates the search process by disburdening the user of repeated parameter changes. The search techniques presented so far do not allow for such a *Best-Matches-Only* query model. Topics as an aggregation of POIs are the foundation for roles, which provide default values for the search mask and guide the composition of preferences. They are associated with tours so that suitable tours are retrieved and should thus be included in the search process to facilitate the expression of personal preferences. Besides role membership, other context parameters are significant aspects in the search process, thus context-aware preferences should adapt to the current situation. A *situation model* is needed to handle discrete situations, to abstract the information content from sensors and to trigger all relevant preferences. Theoretical aspects behind our situation model and its context-adaptive approach are handled in the paper 'Preference SQL and its Query Composition for Context-adaptive Recommender Systems' which has been submitted for publication elsewhere. In mobile outdoor scenarios, partial information also needs to be handled. Besides the classical search functionality, the outdoor activity portal also hosts a social network service for outdoor enthusiasts. A personalized search should benefit from retrieved 'friendship' relations and should include recommendations of friends.

Summarizing the objectives, 'perfect hits' are rarely achieved at first trial, leading to a trial-and-error task for the frustrated user. Instead of absolute values for tour attributes, users think in qualitative terms like 'better', 'worse', 'equal', and 'incomparable' to express the suitability of outdoor tours. A 'best-matching objects' approach overcomes the effects of previously presented search techniques, taking intrapersonal and interpersonal factors as well as the user situation into account.

3 Preference Search

Discussing different opportunities, people use sentences like '*I prefer y over x*' considering a set of attributes A. 'y' and 'x' are representing values of the domains of A. Formally, Strict *Partial Orders* as a means to express this behaviour are the foundation of Preference Algebras by Kießling (2002) and Chomicki (2003).

Since preferences are defined over sets, Preference Algebra is an extension of Relational Algebra. Thus, Preference SQL (Kießling et al. 2002) expands standard SQL-92 by Preferences.

Users now gain expressiveness by formulating

- Hard constraints with their 'Perfect Hits' semantics,
- Soft constraints with their 'Best-Matches-Only' semantics.

For a broader perspective on preference research, see Stefanidis et al. in 2011. In the following, the 'Best-Matches-Only' model guarantees optimality with respect to the corresponding preference. The syntax of Preference SQL fragments is not explained in detail here (see Kießling et al. 2011).

3.1 Best-Matching-Only Query Model

The Preference-Based Search relies on the 'Best-Matches-Only' query model, which guarantees that the result only contains the best matching objects with regard to the corresponding preference. This convenient and goal-driven declarative semantics is formally formulated as follows:

$$\sigma[P](R) := \{t \in R \mid \neg\exists v \in R : t[A] <_P v[A]\} \quad (1)$$

This formula defines the *preference selection* for a relation *R* with respect to a preference *P*. As a result, perfect matches of *R* with respect to attributes *A* are returned if such tuples exist and best matching objects otherwise, but nothing worse.

3.2 Base Preferences

Base preferences are defined on numeric or categorical domains:

Categorical Preferences. For categorical attributes only discrete values of an attribute domain are allowed. For instance, in the case of traffic lights only the values 'red', 'yellow', and 'green' are defined for the attribute 'colour'.

Now let us continue our running example: Paul's children are visiting. Thus, Paul also checks the box for 'family' as shown in Fig. 1. The system configuration defines that

children like pizzerias and fast food better than inns or even hotels. Thus, Paul expects a tour that matches his preferences and that has associated POIs fulfilling these additional food requirements.

Using Preference SQL, the above-mentioned example is easily expressed by the 'LAYERED' preference:

P_{Child}(food) := food LAYERED ((‘pizzeria’, ‘fast food’), (‘inn’), (‘hotel’)) [#14]

[# number] is used later on to reference preferences defined here. The statement specifies four layers. The best three layers are populated as mentioned above. The worst layer is automatically added and contains all other values of 'food'.

Having the choice between 'easy', 'normal', and 'hard', Paul prefers easy tours. This preference is modelled by the 'POS' preference, syntactically written as 'IN', which is a specialization of the 'LAYERED' preference.

P_{Child}(difficulty) := difficulty IN (‘easy’) [#13]

Numerical Preferences. Continuing our running example, Paul's further preferences for a suitable tour are concerning numerical attributes as 'length', 'duration', and 'ascent'. By common sense, the input of numerical search parameters should account for a certain degree of tolerance. This deviation is defined by the 'd-parameter'. Without this parameter, attribute values are strictly evaluated according to their deviation from the perfect match. With the parameter, in contrast, values within a deviation of *d* are considered as equally acceptable. Thus search parameters are implicitly widened by intervals expressed by the 'BETWEEN' preference.

Paul likes tours with a length of about 10 kilometres, or tours with a duration of about 4 hours, as well as tours with a total ascent of about 200 metres. Regarding deviations from his preferences, a ten-percent rule of thumb associated to his role of 'tourist' is acceptable. The stated interval preferences are expressed by 'BETWEEN', taking the lower and upper bound of the preferred interval as well as the d-parameter as arguments. Thus, his preferences are modelled as follows:

P_{Paul}(length) := length BETWEEN 9, 11, 1 [#10]
P_{Paul}(duration) := duration BETWEEN 3.6, 4.4, 0.4 [#11]
P_{Paul}(ascent) := ascent BETWEEN 180, 220, 20 [#12]

Since each type of user has its own view of requirements and wishes, it is useful to count interesting POIs of a tour like points with regard to the target group. Higher values indicate higher convenience. The 'MORE THAN' preference is a sub-preference of 'BETWEEN' that replaces the upper limit by positive infinity. Classified as tourist, Paul gets the following preference:

$P_{Tourist}$(tourist) := tourist MORE THAN 50, 10 [#20]

The lower limit is 50 and the minimum may be zero. Thus, all values above 50 are perfect hits. The d-parameter of 10 specifies six layers.

3.3 Complex Preferences

As in everyday life, the quality of results depends on more than one attribute. Remember Paul's attributes: length, duration, and ascent. There exist two qualitative complex preferences to define preferences over several attributes:

- Attributes, which are *more important than* others, are combined by a *Prioritization* preference (syntactically expressed by 'PRIOR TO').
- Attributes, which are *of equal importance*, are combined by a *Pareto* preference (syntactically expressed by 'AND').

Prioritized Preferences. If an attribute A is more important than B, then A is considered first and B is only decisive if tuples are equal according to A. Thus, the optimization method of prioritization implements cascaded soft filters. Maslov's *Hierarchy of Needs* (1943) gives suitable cues for arranging the preferences. For Paul, the length of a tour is more important than its ascent:

P_{Paul}(length) *PRIOR TO* P_{Paul}(ascent) :=

length	BETWEEN 9, 11, 1	*PRIOR TO*	[#10]
ascent	BETWEEN 180, 220, 20		[#12]

Pareto Preferences. If attributes are equally important, the Pareto preference is used. Pareto states that only those tours are better than others, if at least one of its attributes is better and all other attributes are not worse at the same time. For Paul, the length and duration of a tour are of equal importance:

P_{Paul}(length) *AND* P_{Paul}(duration) :=

length	BETWEEN 9, 11 , 1	*AND*	[#10]
duration	BETWEEN 3.6, 4.4, 0.4		[#11]

The result of a Pareto preference is also known as 'Pareto frontier'. In 1975 Kung et al. examined the calculation of the Pareto frontier by finding the maxima of a set of vectors. Thus, the calculation of a Pareto preference belongs to the field of *Multi-Objective Optimization*. If there are no best matches according to both attributes, compromises are offered, also for the case of contradicting preferences.

Finally, Paul expresses that his whole length and duration preference is more important than the ascent preference:

(P_{Paul}(length) *AND* P_{Paul}(duration)) *PRIOR TO* P_{Paul}(ascent) [#10-12]

4 Preference Search Evaluation

Referring to our running example, the advantages of Preference Search are now discussed.

4.1 Preference SQL Query Composition

For evaluation purposes, basically the same search mask as for Faceted Search (see Fig. 1) including the 3 attributes of length, ascent and duration is used as depicted in Fig. 2.

Fig. 2. Annotated Search Mask of Preference Search

After a user has filled the search form and hits the search button, the Preference SQL query composition is started as outlined in Fig. 3.

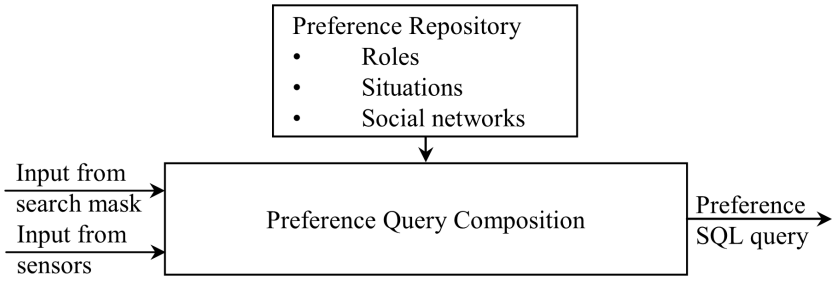

Fig. 3. Preference query composition

The Preference SQL query composition gets its input from the user's search mask and sensor input such as the current GPS position or weather data of the target region as

retrieved via the YAHOO Weather API. The composition further processes the context models of ‘Roles’, ‘Situations’, and ‘Social Networks’, which are stored in the ‘Preference Repository’ (Holland et al. 2004). User profiles of the outdooractive community that are in addition linked to Facebook accounts are acting as data source to determine friendship relations and tour rating of fellow community members.

Thus, directly stated user preferences are combined with context-aware preferences to formulate a *single Preference SQL statement*. The generated statement for Paul’s use case is shown in Fig. 4.

```
#1   SELECT  tour.pid
#2   FROM    oa_tour tour, gs_georegion g,  ua_annotation a,
#3           um_user u, ua_friend f
#4   WHERE   (tour.pid = g.pid AND g.name = ‘Allgäu’) AND
#5           (tour.pid = a.pid) AND
#6           (tour.activity IN (‘hiking trail, ‘pilgrim track’,
#7                              ‘city trail’, ‘via ferrata’)) AND
#8           (u.name = ’Paul’ AND u.pid = f.pid)
#9   PREFERRING
#10          ( ( ( (tour.length BETWEEN 9, 11, 1)  AND
#11                (tour.duration BETWEEN 3.6, 4.4, 0.4))
#12              PRIOR TO (tour.ascent BETWEEN 180, 220, 20))  AND
#13            ( (tour.difficulty IN (‘easy’) )  AND
#14              (a.food LAYERED ((‘pizzeria’, ‘fast food’), (‘inn’), (‘hotel’)))))
#15          PRIOR TO
#16          ( (a.fallback MORE THAN 50, 5)  PRIOR TO
#17            (tour.altitude_max LESS THAN 2000, 200)  PRIOR TO
#18            (tour.activity IN (‘city trail’) NOT IN (‘via ferrata’)))
#19          PRIOR TO
#20          ( (a.tourist MORE THAN 50, 10)  PRIOR TO
#21            ( (tour.altitude_min MORE THAN 507, 200)  AND
#22              (tour.altitude_max LESS THAN 2174, 200)))
#23   ORDER BY f.recommendation DESC;
```

Fig. 4. Preference SQL statement for the running example

All information of Paul’s use case is stored in the database relations of tours, regions, annotations, users, and friends [#2-3]. The WHERE-Clause defines hard constraints for the region [#4] by selecting all tours located in ‘Allgäu’, kind of activity [#6-7], and his social network [#8] by selecting all friendship relations for his user. The preference query composition uses Paul’s input [#10-14], his role [#20], his role-dependent preference for an altitude range leading to a convenient temperature [#21-22], his social ties [#23], and additional weather-dependent preferences [#16-18]. The [# number] markers are referring to preferences defined in previous sections.

Note that ‘AND’ in the WHERE-clause denotes Boolean conjunction whereas ‘*AND*’ in the PREFERRING-clause stands for the complex Pareto preference.

The Preference SQL middleware between database and application is also responsible for complex preference optimization techniques and evaluation algorithms (for details see Kießling et al. 2011).

4.2 Sample Search Session

A typical Preference Search user session proceeds as follows:

(1) Filling out the search mask (see Fig. 2).

(2) Hitting the search button, triggering Preference SQL query composition, Preference SQL optimization, and evaluation, afterwards search result display at the client (which may be a smartphone).

(3) Inspecting the displayed search results.

Regarding Paul’s use case, step (2) needs around four seconds, returning 5 tour recommendations out of 1889 initial tours matching the hard search conditions for ‘hiking’ and ‘Allgäu’. Thus in step (3) Paul can see at one single glance all tours which are optimal with regard to his input (1) and the automatically available context information. The retrieved tours are no perfect hits (which usually happens), but best available alternative compromises.

In comparison, a typical Faceted Search user session proceeds like this:

(1) Filling out the search mask iteratively (see Fig. 1). All involved attributes have to be filled out one after the other at the user's trial-and-error choice. After each step an intermediate search result is retrieved and displayed. If at some point an empty result pops up, the user has to retrace and start over with different input choices.

(2) After all involved attributes are set the user can examine the quality of the displayed search result.

(3) Then two typical cases can arise:

   a. In case of an empty search result, the whole procedure has to be repeated.

   b. Otherwise, if the user finds some satisfactory results, the session ends. Apart from that, the whole procedure has to be repeated again.

Regarding Paul’s use case, the duration of step (1) can vary substantially depending on his choices. Instead of seconds it might last for many minutes. Note, that not only Paul's patience is challenged by this system behavior, but also lots of additional system communication overhead is generated which is especially critical in a mobile environment. Depending on Paul's choices, step (3) can be reached with different outcomes. As a typical example we observed one case for which 209 tours were deliver-

ed. Also typical for nowadays usage of search engines, Paul probably only looks at the first result screen. If so, he would be very unfortunate then. As it turns out here, from the 5 tours known to be optimal from Preference Search only one tour is found in the result of Faceted Search with rank number 6 on the 2nd result screen. The other 4 best matching tours are not included in those 209 returned tours.

In a nutshell, Preference Search offers a one-step search interface to the user, retrieving results very efficiently and with a guarantee of finding all best matches. This clearly contrasts to Faceted Search, being a multi-step procedure with no guarantees about the optimality of search results. The discrepancy gets even more striking when using mobile smartphone clients. Needless to mention, Faceted Search becomes even more impractical and cumbersome for higher attribute dimensionality.

5 Benchmark

A benchmark of typical use cases has been defined in order to compare the outcomes of Faceted Search with those of Preference Search implemented by the 'outdooractive.com' outdoor activity platform. For details of the use cases see appendix B.

5.1 Test Configuration

Due to the cooperation with Alpstein Tourismus GmbH & Co. KG, three configurations are possible running the benchmark (see also Fig. 5).

(TC1) outdooractive.com portal in operation implementing Faceted Search

(TC2) testing.outdooractive.com/de/ portal for testing Faceted Search

(TC3) approval.outdooractive.com/de/tours.recommender.jsp portal for testing Preference Search

The optimal combination would have been to run the benchmark in TC2 and TC3 test configuration. Nevertheless the low availability of TC2 urged that the characteristics of Faceted Search are measured in TC1 test configuration, whereas the characteristics of Preference Search are measured in TC3 test configuration.

For better understanding, the test infrastructure is outlined as follows.

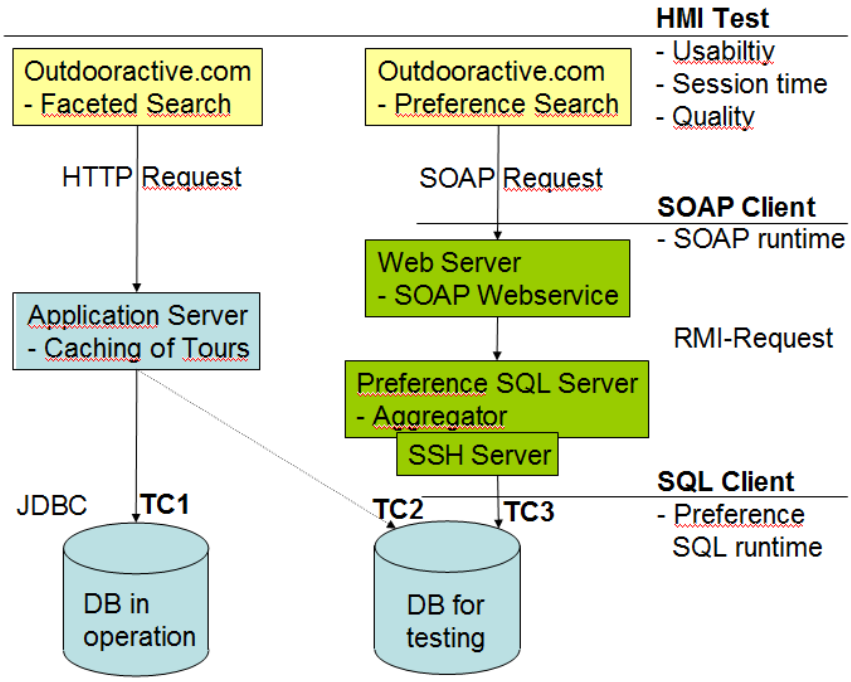

Fig. 5. Test Configuration

Yellow frontend components are hosted by Alpstein Tourismus GmbH & Co. KG and offer identical or similar human machine interfaces (HMI).

Blue middleware and backend components are hosted by Alpstein Tourismus GmbH & Co. KG and implement an application server as well as two different databases for tours. One is used for day-to-day business (DB in operation), whereas the other is used for testing and improving the current state (DB for testing). PostgreSQL 9.0.3 is running on both databases.

Green middleware components are hosted by the University of Augsburg. A SOAP web service as part of a web server is used for decoupling the HMI from the Preference SQL middleware. The web server communicates via RMI to the Preference SQL server hosting the preference query composition and the Preference SQL optimization. The Preference SQL server accesses to the test database of Alpstein Tourismus GmbH & Co. KG. The data is tunnelled by a SSH server for security. Only those tours are delivered to the HMI that fulfil the BMO criterion.

5.2 Criteria

In order to evaluate the benchmark, following criteria are defined and measured:

(1) Technical criteria (see appendix C)

   a. Runtime of Preference SQL queries

   b. Runtime of SOAP requests

(2) User acceptance testing (see appendix D, E, and F)

   a. Session time of test subjects

   b. Behaviour: success, abort, iteration

   c. Comment of test subjects

(3) Quality assessment (see appendix H)

Observations of the supervisors are gathered in appendix G.

5.3 Summary

The evaluation of the benchmark comparing Preference Search to Faceted Search is providing following results.

(1) The Preference SQL run time never exceeded 5 seconds.

(2) The session time of Preference Search was always shorter as the session time of Faceted Search.

(3) The test subjects expect ‘larger’ result sets executing Preference Search, even if they may contain tours which are not belonging to the BMO-set.

(4) Empty results are still present in Preference Search due to hard, above all geographical restrictions.

(5) The flooding effect was never noticed executing Preference Search. The result set mostly consists of 1 to 7 tours.

(6) The average quality of the result set of Preference Search was rated as 1.6 points by domain experts.

6 Conclusion and Outlook

We have demonstrated the power of the Preference Search paradigm for a sample tourism application for outdoor activities. Preference Search implemented by means of Preference SQL distinguishes itself from competitor search methods by several advantages. Personal user preferences can be intuitively expressed using a variety of powerful preference constructors. Preference information can be directly entered into the search mask by the user, or can be derived from pre-defined user roles, as well as extracted from social networks. Context-awareness and adaptation can be achieved dynamically by flexible query composition. Instead of lengthy and tedious query sessions with sub-optimal query results, Preference Search is a one-step search action: The automatically composed single complex Preference SQL query returns exactly those items that match the user's expectation best possible under the given context. Thus it can achieve a much higher user satisfaction compared to other popular approaches like Faceted/Parametric Search. This effect becomes even stronger in mobile environments, with search requests via smartphone, or in high-dimensional multi-attribute search domains. The evaluation results are generated by a systematic benchmark, using real life commercial data and incorporating human domain experts for outdoor activities. Theoretical aspects behind our situation model and context-adaption approach, as well as novel geo-preferences for location-based services are beyond the scope of this report. More details of geo-preferences are available in Wenzel et al. (2012). Details of the situation model and the context-adaption have been submitted for publication elsewhere, pending notification.

References

Chomicki, J. (2003). Preference Formulas in Relational Queries. *ACM Transactions on Database Systems* 28: 427-66.

Feris, R. & Siddiquie, B. & Zhai, Y. & Petterson, J. & Brown, L. & Pankanti, S. (2011). Attribute-based vehicle Search in crowded surveillance videos. *Proceedings of the 1st ACM International Conference on Multimedia Retrieval* Article No.: 18.

Fox, E. A. (1999). The Digital Libraries Initiative - Update and Discussion. *Bulletin of the American Society of Information Science* 26(1): 7-11.

Gibson, H. & Yiannakis, A. (2002). Tourist roles: Needs and the Lifecourse. *Annuals of Tourism Research* 29(2): 358-383.

Holland, S. & Kießling, W. (2004). Situated Preferences and Preference Repositories for Personalized Database Applications. *Proceedings of the 23rd International Conference on Conceptual Modeling*: 511-23.

Hutchinson, H. B. (2003). Children's interface design for hierarchical search and browse. *ACM SIGCAPH Computers and the Physically Handicapped* 75: 11-12.

Kießling, W. (2002). Foundations of Preferences in Database Systems. *Proceedings of the 28th International Conference on Very Large Data Bases*: 311-22.

Kießling, W. & Köstler, G. (2002). Preference SQL – Design, Implementation, Experiences. *Proceedings of the 28th International Conference on Very Large Data Bases*: 990-1001.

Kießling, W. & Endres, M. & Wenzel, F. (2011). The Preference SQL System - An Overview. *Bulletin of the Technical Committee on Data Engineering* 34(2): 12-19.

Kung, H. T. & Luccio, F. & Preparata, F. P. (1975). On Finding the Maxima of a Set of Vectors. *Journal of the ACM* 22(4): 469-76.

Maslov, A. (1943). A Theory of Human Motivation. *Psychological Review* 50(4): 370-96.

Sacco, G. M. & Tzitzikas, Y. (2009). *Dynamic Taxonomies and Faceted Search.* Heidelberg - New York: Springer.

Stefanidis, K. & Koutrika, G. & Pitoura, E. (2011). A Survey on Representation, Composition and Application of Preferences in Database Systems. *ACM Transactions on Database Systems* 36(3).

Wenzel, F. & Soutschek, M. & Kießling, W. (2012). A Preference SQL Approach to Improve Context-Adaptive Location-Based Services for Outdoor Activities. *8th International Symposium on Location-Based Services.* Gartner, G. & Ortag, F. (Eds.): to appear.

Acknowledgements

The project PeToMoTo is funded by the German Federal Ministry of Economics and Technology according to a decision by the German Bundestag, grant no. KF2751301. Real tour data was provided by Alpstein Tourismus GmbH & Co. KG. Special thanks to Anna Schwartz, Philipp Lohmüller, and Simon Lohmüller investing their time in performing the benchmark.

Appendix

(A) Involved Relations

Tours are described by the '*ua_tour*' relation. This relation consists of 58 attributes of which the benchmark takes care of

(1) int8 pid
(2) int4 state
(3) int2 technique
(4) int2 condition
(5) int4 min_time
(6) int4 calc_ascent
(7) float4 calc_length
(8) int2 calc_altitude_max
(9) int2 calc_altitude_min

The '*ua_annotion*' relation is one-to-one related to 'ua_tour' by its foreign key 'pid' referring to 'pid' of ua_tour. This relation contains precompiled features of a tour as e.g.

(1) int8 pid
(2) int4 panorama
(3) int4 food
(4) int4 children
(5) int4 cycle
(6) int4 athlete
(7) int4 tourist
(8) int4 family
(9) int4 bonvivant
(10) int4 fallback
(11) int4 risky

The numbers represent the outcome of a utility function taking care of 'interesting' POIs. The attributes (2) – (5) can be chosen by the HMI. The attributes (6) – (9) represent interests of user types, which may also be selected by the user explicitly. They trigger the internal *user model*. 'Fallback' and 'risky' are relevant to the *situation model*.

The '*bc_relationrole*' relation is a many-to-many relation mapping tours to other concepts. The benchmark uses only geographical relations like 'tours belonging to a country', 'tours belonging to a province', or 'tours belonging to a region'. Other relations are used to precompile the 'ua_annotation' relation, but they do not influence the defined criteria.

(1) int8 source_id
(2) int8 target_id
(3) int8 relationtype_id

Geographical information, activities, and the input of all check boxes are modelled as hard restrictions.

The benchmark criteria depend on the size of the involved relations

Relation	Size in TC1	Size in TC2 / TC3
ua_tour	172.000	132.059
ua_annotation	Not existent!	132.059
bc_relationrole	11.000.000	1.833.805
gs_georegion	45.000	35.725

Tab. 1. Relation Size depending on Test Configuration

(B) Benchmark

A benchmark was defined and used in the test configuration TC1 and TC3. The HMI of Faceted Search is shown by Fig. 1. The HMI of Preference Search is depicted in Fig. 2 differing only in the use of single-sliders instead of double-sliders when specifying 'length', 'ascent', and 'duration'.

The goals G1 – G19 had to be achieved by the users. The values of 'length', 'ascent', and 'duration' can be approximated by intervals in the case of Faceted Search. All other attributes are perfectly achievable.

G1, … , G4 check the interplay with the most restrictive hard condition forced by the selected location. G1 yields an empty result.

G5, … , G8 demonstrate the influence of the selected type because all other parameters are identical. Obviously, the tours are corresponding to the selected type and differ from those of another type or are part of the most generic type.

G9 shows how 'useless' input is handled as ascent (*0 m*) and length (*0 km*).

G10, … , G14 demonstrate the type-specific selection of tours with small parameter variations.

G14, … , G15 demonstrate the influence of the weather context fog v.v. dry.

G15, … , G18 demonstrate the influence of the remaining day light.

G19 includes tours with unknown attributes.

The query result contains the set of IDs of tours fulfilling the BMO-criterion.

G1: *Family* is looking for *theme trails* in *Churfranken* selecting *all* difficulty degrees, duration (*2 h*), ascent (*200 m*), and length (*5 km*). ➔ Result: { } (see protocol_id = 3-442)

G2: *Bon vivant* is looking for *theme trails* in *Bavaria* selecting *all* difficulty degrees, duration (*3 h*), ascent (*200 m*), and length (*4 km*). ➔ Result: {1387083, 1386540, 1398693} (see protocol_id = 3-399)

G3: *Bon vivant* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*3 h*), ascent (*200 m*), and length (*4 km*). ➔ Result: {1362439, 1386972} (see protocol_id = 3-402)

G4: *Bon vivant* is looking for *hiking tour trails* in *France* selecting *medium and hard* difficulty degree, duration (*8 h*), ascent (*1200 m*), and length (*21 km*). ➔ Result: {1398498, 1362673, 1385764} (see protocol_id = 3-428)

G5: *Bon vivant* is looking for *hiking tour trails* in *Allgäu* selecting *all* difficulty degrees, duration (*2 h*), ascent (*600 m*), and length (*18 km*). ➔ Result: {1381451, 1369518, 1377530, 1387115, 1400534, 1398989, 1386908} (see protocol_id = 3-462)

G6: *Tourist* is looking for *hiking tour trails* in *Allgäu* selecting *all* difficulty degrees, duration (*2 h*), ascent (*600 m*), and length (*18 km*). ➔ Result: {1400534, 1386908} (see protocol_id = 3-466)

G7: *Family* is looking for *hiking tour trails* in *Allgäu* selecting *all* difficulty degrees, duration (*2 h*), ascent (*600 m*), and length (*18 km*). ➔ Result: {1381451} (see protocol_id = 3-482)

G8: *Athlete* is looking for *hiking tour trails* in *Allgäu* selecting *all* difficulty degrees, duration (*2 h*), ascent (*600 m*), and length (*18 km*). ➔ Result: {1398989} (see protocol_id = 3-487)

G9: *Bon vivant* is looking for any kind of *hiking tours* in *Taubertal* selecting *all* difficulty degrees, *round trip*, *panorama*, duration (*2 h*), ascent (*0 m*), and length (*0 km*). ➔ Result: {1374651, 1381412} (see protocol_id = 3-507)

G10: *Family* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*1 h*), ascent (*50 m*), and length (*0 km*). ➔ Result: {1362769} (see protocol_id = 3-527)

G11: *Bon vivant* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*2 h*), ascent (*50 m*), and length (*0 km*). ➔ Result: {1362769, 1386972} (see protocol_id = 3-533)

G12: *Tourist* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*3 h*), ascent (*50 m*), and length (*5 km*). ➔ Result: {1362439, 1386972} (see protocol_id = 3-544)

G13: *Athlete* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*8 h*), ascent (*400 m*), and length (*10+ km*). Weather parameters are *dry* and *3*° C. The starting time is 3 p.m. ➔ Result: {1362439} (see protocol_id = 4-37)

G14: *Athlete* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*8 h*), ascent (*500+ m*), and length (*10+ km*). Weather parameters are *dry* and *3*° C. The starting time is 2 a.m. ➔ Result: {1362769, 1362439, 1386972} (see protocol_id = 3-547)

G15: *Athlete* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*8 h*), ascent (*500+ m*), and length (*10+ km*). Weather parameters are *fog* and *7*° C. The starting time is 3 a.m. ➔ Result: {1362769} (see protocol_id = 4-86)

G16: *Athlete* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*8 h*), ascent (*500+ m*), and length (*10+ km*). Weather parameters are *dry* and *6*° C. The starting time is 3 p.m. ➔ Result: {1362769, 1386972} (see protocol_id = 4-97)

G17: *Athlete* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*8 h*), ascent (*500+ m*), and length (*10+ km*). Weather parameters are *dry* and *7*° C. The starting time is 5 p.m. ➔ Result: {1362769} (see protocol_id = 4-113)

G18: *Athlete* is looking for *theme trails* in *Fränkische Schweiz* selecting *all* difficulty degrees, duration (*8 h*), ascent (*500+ m*), and length (*10+ km*). Weather parameters are *dry* and *6*° C. The starting time is 6 p.m. after sunset. ➔ Result: {1362769, 1362439, 1386972} (see protocol_id = 4-123)

G19: *Bon vivant* is looking for *hiking tour trails* in *Allgäu* selecting *all* difficulty degrees, duration (*2 h*), ascent (*300 m*), and length (*12 km*). ➔ Result: {1404036, 1378169, 1393790, 1386908, 1374995, 1395031, 1395165} (see protocol_id = 4-109)

(C) Technical Criteria

The benchmark defined in appendix B was used to measure the technical criteria. The run time depends on the computational complexity of the generated preference query.

Case	Preference SQL run time [s]	SOAP run time [s]	Context	Size of Result
G1	2.5	**2.6**	No	0
G2	**2.2**	3.4	No	3
G3	3.7	3.8	rainy, 2°, early start	2
G4	2.5	**2.6**	No	3
G5	3.6	4.1	rainy, 2°, early start	7
G6	3.6	4.2	rainy, 2°, early start	2
G7	3.6	**4.4**	dry, 2°, early start	1
G8	3.1	3.8	dry, 2°, early start	1
G9	2.9	3.5	dry, 4°, early start	2
G10	3.3	4.0	dry, 4°, early start	1
G11	3.0	3.5	dry, 4°, early start	2
G12	2.9	3.4	dry, 3°, early start	2
G13	3.5	4.3	dry, 3°, late start	1
G14	3.4	3.9	dry, 3°, early start	3
G15	**3.8**	4.3	fog, 7°, early start	1
G16	3.5	4.1	dry, 6°, late start	2
G17	3.7	4.2	dry, 6°, before sunset	1
G18	3.2	3.9	dry, 6°, after sunset	3
G19	3.3	4.2	dry, 6°, late start	7

Tab. 2. Technical Criteria in TC3

The measured Preference SQL runtime spreads from *2.2* to *3.8* seconds. The result delivers only tours, which are optimal according to the BMO-criterion. The measured SOAP runtime spreads from *2.6* to *4.4* seconds. The critical limit of 5 seconds was never violated. Additional load was added by a SSH-Server tunnelling the connection to the Alpstein database in the TC3 test configuration. A closer integration may further reduce the measured runtimes.

(D) User Acceptance of Preference Search

The benchmark was also executed by a small group of test subjects to measure the user acceptance of Preference Search in a field test.

Case	Session time [m:s]	Test subject	Success / Abort	Comment
G1	1:35	1	-	Ready or not?
G1	1:03	2	-	Ups, no results!
G1	1:48	3	-	Ready or not?
G2	1:54	1	S	Location by text or by hierarchy?
G2	0:27	2	S	Slider does not update input value.
G2	0:35	3	S	One duplicate (german – english)!
G3	1:03	1	S	
G3	0:28	2	S	Slider discretization is annoying.
G3	0:25	3	S	Wrong: distance of 13.1 km
G4	1:35	1	S	
G4	1:08	2	S	
G4	0:53	3	S	Correct country?
G5	1:37	1	S	
G5	1:01	2	S	
G5	0:45	3	S	2nd result is best.
G6	0:34	1	S	
G6	0:21	2	S	
G6	0:24	3	S	1st result is best.
G7	0:24	1	S	
G7	0:35	2	S	
G7	0:28	3	S	
G8	0:24	1	S	
G8	0:25	2	S	Result is not correct!
G8	0:24	3	S	Result size is too small!
G9	1:59	1	S	
G9	0:34	2	S	
G9	0:31	3	S	Result is very good.
G10	0:47	1	S	
G10	0:44	2	S	Result is very good.
G10	0:34	3	S	
G11	0:25	1	S	What is bon vivant?
G11	0:15	2	S	No children!
G11	0:19	3	S	1st result is best!
G12	0:33	1	S	
G12	0:21	2	S	
G12	0:32	3	S	2nd result is best, but …

G13	0:46	1	S	
G13	0:42	2	S	Distance is too short!
G13	0:45	3	S	Result size is too small!
G19	0:41	1	S	
G19	0:42	2	S	The result set should be greater (<15).
G19	0:45	3	S	Result size is sufficient (<10). Results of the first page are better.

Tab. 3. Benchmark running on Preference Search, TC3

The test subjects can be classified by these attributes

- *age*: 1 older person v.v. 2 younger persons,
- *gender*: 1 female person v.v. 2 male persons,
- *search experience*: 3 experienced persons.

(E) User Acceptance of Faceted Search

Since some features (user type, context) of the benchmark cannot be modelled in Faceted Search the goals of the benchmark are reduced by Gi'.
Goals G5 to G8 differ only in the user type. The reduced goal is denoted G5'. Goals G10 to G14 differ only in different context. The reduced goal is denoted G10'.

Case	Session time [m:s]	Test subject	Success / Abort / Iteration	Comment
G1'	2:18	1	-	Ready or not?
G1'	1:17	2	-	
G1'	1:15	3	-	Search button was pressed.
G2'	3:30	1	S + I	Double slider is hard to handle
G2'	0:33	2	S + I	Use of intervals, because wished values are improbable.
G2'	0:31	3	S	1st result is best.
G3'	2:35	1	S + I	
G3'	0:55	2	S + I	Nearly abort!
G3'	0:48	3	A	No result!
G4'	2:22	1	S + I	
G4'	1:30	2	S + I	More results are preferable!
G4'	1:47	3	A	No result! Distance of 21 km is outside of my preferred range! Distance is the most important attribute!
G5'	1:12	1	S + I	
G5'	1:31	2	S + I	1st criterion is size of result. 2nd criterion is quality of items.
G5'	1:25	3	S + I	Results are good, but more results would be better.
G9'	1:24	1	S + I	Many tours but satisfied user
G9'	1:38	2	S + I	
G9'	1:38	3	S + I	Bad compromise!
G10'	1:05	1	S + I	
G10'	1:26	2	S + I	
G10'	1:09	3	A + I	I can not express ma goals!
G19'	3:35	1	S + I	Many iterations to reduce tours
G19'	1:52	2	S + I	Fine tuning is impossible by the discretization of sliders.
G19'	1:23	3	A + I	

Tab. 4. Benchmark running on Faceted Search, TC1

The benchmark was also executed by the same group of test subjects to measure the user acceptance of Faceted Search.

(F) Comparison of Faceted Search vice versa Preference Search

The benchmark offers the opportunity to compare the average session time of Preference Search (see appendix D) and Faceted Search (see appendix E).

Test TC3,TC1	**Preference Search (PS) average session time [m:s]**	**Faceted Search (FS) average session time [m:s]**	**Percentage (FS - PS) / FS [%]**
G1,G1'	1:29	1:37	8.28
G2,G2'	0:59	1:31	35.77
G3,G3'	0:39	1:26	55.04
G4,G4'	1:12	1:53	36.28
G5,G5'	1:08	1:23	18.15
G9,G9'	1:01	1:33	34.29
G10,G10'	0:42	1:13	100.00
G19,G19'	0:40	2:17	70.73

Tab. 5. Comparison of Session Time of Preference Search in TC3 vice versa Session Time of Faceted Search in TC1

The session time of Preference Search is shorter in all cases even if the users had to choose the 'type' parameter additionally.

(G) Observations

In both approaches, the users mostly expected a larger result size if only a handful of results are returned to offer them a 'better' choice. Users often assume that the first page of results is better than the results of following pages. This opinion is surely wrong in the case of Preference Search. One user also claimed that the delivery of tours with an unknown attribute is not useful. The quality assessment by users follows simple intrapersonal guidelines (e.g. Distance is most important. Duration doesn't matter, because I am normally walking faster than others.) This user also stated that some elements of the result set of Preference Search are wrong. The termination criterion of Faceted Search is above all determined by the size of the result set, second by the quality of the result. If an empty result is achieved, users release any parameters. The choice of a parameter is intuitive, the choice of the upper or the lower limit to release too. If too many tours are returned, users often avoid refining the parameters, because they claim that they are satisfied by the first offered tours. Users are consuming a lot of time in order to reduce a too large set of tours or to widen an empty set of tours by tuning search parameters. Thus users often break the process claiming that they are satisfied. Users did not reflect which order is the best for changing parameter values. The order normally was left to right. One user was very consequent and aborted the iteration after having released some parameters because intrapersonal limits should not be broken.

(H) Quality Criteria

The benchmark was afterwards analyzed by domain experts of Alpstein Tourismus GmbH and Co. KG.
The assessment used three quality levels:

- 1 as ‘good’
- 2 as ‘may be ok’
- 3 as ‘bad’

Case	Assessment	Comment
G1	1	
G2	1	
G3	3	
G4	2	
G5	2	
G6	1	
G7	2	Breitachklamm is expected!
G8	1	
G9	1	
G10	2	
G11	2	
G12	2	
G13	2	
G14	1	
G15	1	
G16	1	
G17	1	
G18	3	
G19	1	
Avg.	**1.6**	Result size is too small! Fun factor is missing.

Tab. 6. Quality Assessment in TC3

The average assessment of 1.6 points supports the correctness of the BMO-criterion by the semantic inspection of domain experts.

(I) Preference SQL Configuration

Following Preference SQL configuration was used during the benchmark.

RELOAD = false

# parameter for the cost and rule based optimizer
USE_OPTIMIZER=true
# unoptimized / merged
USE_UOM=true
# optimized / not merged
USE_ONM=false
# optimized / merge
USE_OM=false

```
CBO_RULE_PATH = opt-nomerge.xml
CBO_MERGER_PATH = noopt-merge.xml
HEURISTIC_PATH = opt_me_cutoff.xml
FOREIGN_KEY_CONSTRAINTS = alpstein_constraints.xml

# parameter for the Hexagon algorithm
#HEXAGON_MAX_EQUIVALENCE_CLASSES = 30000000
# btg datatype is long. However, in Java only int array indices are allowed, i.e. max
      2147483647
# will be checked in AFParetoPreference
# set max BTG_SIZE to 2147483647-1, because we have to handle NULL values in
      level 2147483647
HEXAGON_MAX_BTG_SIZE = 2147483646
HEXAGON_K = 32000000000

ALG_LIST =
      preference.sql.algorithm.HexagonMemOpt:preference.sql.algorithm.HexagonI
      nMemoryAlgorithm:preference.sql.algorithm.Less:preference.sql.algorithm.B
      NLplusplusAlgorithm:preference.sql.algorithm.Bnl2d:preference.sql.operator.
      BnlToppedOperator:preference.sql.operator.BnlPipedRelOp

USE_HEXAGON = true
USE_SEMIPARETO = true
USE_BNLPLUSPLUS = true
# use a level based algorithm for the evaluation of a Prioritization preference
# otherwise BNL would be used
USE_LEVELBASEDPRIO=false
USE_PREFERENCEBASEDPRIO=false

# only set this property if you want to force an algorithm for Pareto evaluation.
# Otherwise a cost-based algorithm selection will be done
PARETO_ALGORITHM =


# JPO
#be used by JpoPanelRulesSetting.java
QUERY_TEST_RERUNS = 100
#be used by JpoPanelRulesSetting.java
DROP_LH = NO
#be used by JpoPanelRulesSetting
CONSTRAINT_FILE = ../Preference/optimizer/constraint.xml
#will be used by QueryOptimizer.java
OPTIMIZE_XML_FILE = ../PreferenceSQL/optimizer/opt_me_cutoff.xml

# JPO: be used by SettingsReader.java
DATABASE_DRIVER = com.mysql.jdbc.Driver
#be used by SettingsReader.java
DATABASE_SOURCE = jdbc:mysql://localhost/test
#be used by SettingsReader.java
DATABASE_USER = users
#be used by SettingsReader.java
DATABASE_PASSWORD = pass
```

```
# parameter for the Preference SQL JDBC driver
#log rmi
LOG_RMI = false
#used by PSQLRemoteDriverImpl to define a log file for RMI
RMI_LOG_FILE = psql_rmi_logging.log

#writes the debug output from DebugLevel.java to a log file
DEBUG_LOG_FILE = psql_debug_logging.log

#be used by DebugLevel.java to define a debug log level
# QUIET = 0, LOW = 1, NORMAL = 2, HIGH = 3, INSANE = 4
DEBUGLEVEL=1
# print used algorithm
DEBUGLEVEL_ALGORITHM=0
# print cost based optimization information
DEBUGLEVEL_CBO=0

#configure number of CPUs for SkylineAlgorithmParallelBNLLinkedListLazySync
numCPUs = 2


# Block size for Cartesian product, SemiJoin, BlockNestedHashJoin,
      BlockNestedLoopJoin. Standard is 2000 hard coded in CartesianProduct.java
BLOCKSIZE=10000000

# BNL CACHESIZE
BNL_CACHESIZE=10000000

# CacheSize for LevelBasedPrio evaluation of WOPs Prioritization
LEVELBASEDPRIO_CACHESIZE=1000000

# initialCapacity, loadFactor for Hash in BlockNestedHashJoin
# default: 11
BLOCKNESTEDHASHJOIN_INITIALCAPACITY=1000000
# default: 0.75
BLOCKNESTEDHASHJOIN_LOADFACTOR=0.6

# Whitelist for query execution in Preference SQL. See PSQLExecutor.java
PSQL_WHITELIST=PREFERRING:DESCRIBE:HELP:LIMIT:LEVEL:DISTANCE
      :TOP:QUALITY:RANKFUNCTION:DEFINE:DEF:PRINT

# Update of TableStatistics in DatabaseConnector implementation
# update table statistics all 10000 calls
# 1 means every call
DC_ANALYZE_COUNTER=10000

# logging in SQLTreeBuilder
USE_LOGGER=false

# use JDBC Connection in BoundStatementCursor or the Native PostgreSQL
      Interface to retrieve the ResultSet
# still under development, so use JDBC
# database connection = NATIVE | JDBC
DBCONNECTION_INTERFACE=JDBC
```

```
# Handle NULL values
# 0: level(NULL) = 0
# -1: level(NULL) = Integer.MAX_VALUE
NULL_LEVEL=-1

# Preference SQL should parse and execute all statements, even if they do not contain
        a preference clause
# true: Preference SQL only executes statements containing a preference clause given
        in Preference SQL_WHITELIST
# false: Preference SQL executes all statements
EXEC_ONLY_PREFERENCES=true

# use optimization rules based on foreign keys,
FOREIGN_KEY_RULES=true
```