International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 2 | ISSN : 2456-3307

Bug Tracking and Resolving

K. Sivagami[1], Dr. A. Jayachandran[2]

[1]PG Scholar, Department of M.Sc(Software Engineering), PSN College of Engineering & Technology, Tirunelveli, Tamilnadu, India

[2] Research Supervisor, Department of M.Sc(Software Engineering), PSN College of Engineering & Technology, Tirunelveli, Tamilnadu, India

ABSTRACT

Bug Tracking, Help Desk Ticketing, issue raising, search facility, help information, issue resolution. Issues related to software projects can be raised, tracked and resolved by Employees of different departments. Resolved issues can be allowed to access from Knowledge Base as Knowledge elements. The different groups and representatives can interact each other through emails. The issue tracking system does all the jobs that are done in conventional system but ,here , everything is done in more formal and efficient manner. All the users of organization can interact with each other through the Issue Tracking System. This system acts as an interface between the employees thereby enabling them to forward their issues to the centralized Issue tracking system. Hence, making the work easy for both the issue raiser and the resolved. It totally avoids the involvement of middlemen in getting resolution for a particular issue.The Issue Tracking system is an intranet application, which provides information about issues in software projects, in detail. This product develops a system that can be used by all the departments of a software organization. In the conventional method, all the issues are dealt manually .The progress of the issues are also checked in person, which is a tedious task. Here, in Issue Tracking, it fulfills different requirements of administrator and employees of a software development organization efficiently. The specific purpose of the system is to gather and resolve issues that arise in different projects handled by the organization.

Keywords : PrefDB, Network,data, SQL, Query parser, tuples.

I. INTRODUCTION

A computer network or data network is a telecommunications network which allows computers to exchange data. In computer networks, networked computing devices exchange data with each other using a data link. The connections between nodes are established using either cable media or wireless media. The best-known computer network is the Internet.

Network computer devices that originate, route and terminate the data are called network nodes. Nodes can include hosts such as personal computers, phones, servers as well as networking hardware. Two such devices can be said to be networked together when one device is able to exchange information with the other device, whether or not they have a direct connection to each other .Computer networks differ in the transmission medium used to carry their signals, communications protocols to organize network traffic, the network's size, topology and organizational intent.Computer networks support an enormous number of applications and services such as access to the World Wide Web, digital video, digital audio, shared use of application server, printers, and fax machines, and use of email and instant messagingapplications as well as many others. In most cases, application-specific communications protocols are layered (i.e. carried as payload) over other more general communications protocols.

A computer network facilitates interpersonal communications allowing users to communicate efficiently and easily via various means: email, instant messaging, chat rooms, telephone, video telephone calls, and video conferencing. Providing access to informationon shared storage devices is an important feature of many networks. A network allows sharing of files, data, and other types of information giving authorized users the ability to access information stored

on other computers on the network. A network allows sharing of network and computing resources. Users may access and use resources provided by devices on the network.

On the other hand, native methods consider preference evaluation and filtering as one operation. Due to this tight coupling, these methods are also tailored to one type of query. Furthermore, they require modifications of the database core, which may not be feasible or practical in real life. Overall, both native and plug-in approaches do not offer a holistic solution to flexible processing of queries with preferences.

II. METHODS AND MATERIAL

A. The Proposed System

PrefDB is a prototype system that is based on the preference and extended relational data and query models that we presented earlier. Section 2 provides an overview of its functionality and architecture and also describes the implementation of p-relations and the operators. Query processing in PrefDB Figure 2 depicts the system's architecture. Modules depicted in yellow are provided by the native DBMS, whereas the blue-colored ones are those developed for PrefDB. As shown, PrefDB offers two alternative query options: preferences can be provided along with the input query or the system can enrich a non-preferential query with related preferences. In the first query option, preferences are specified in a declarative way, additionally to the standard SQL query part. In the second case, relevant preferences are provided by the profile manager module, which accesses user preferences stored in the database. Stored preferences can be collected from user ratings or by analyzing past queries or clickthrough data [7]. Since preference collection is orthogonal to query processing, which is the primary goal of PrefDB, in our implementation, we simply store preferences specified by users through a visual tool we have developed [7] as well as preferences specified in past Query Parser Query + Preferences Query Optimizer Extended Query Plan SQL Execution Engine Database Engine Scoring, aggregate functions Data Operators σ, π, λ, Optimized Query Plan Profile manager Query + Preferences

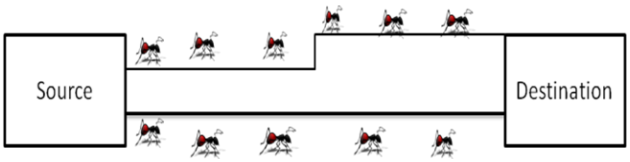


Figure 1. Ants take the shortest path from source to destination

user queries. For both query options, the query and the preferences are given as input to the query parser. Apart from the core PrefDB query processing strategies that blend preference evaluation into query processing, we have also implemented a set of plug-in methods, which are described in the Appendix. Below is an overview of the core PrefDB modules

- The profile manager selects from the database preferences that can be combined with the conditions of the issued query. For this purpose, we use the preference selection algorithm proposed in [20]
- The query parser takes as input the query and preferences and generates an extended query plan that is passed to the PrefDB query optimizer.
- The query optimizer improves the input plan by applying a set of algebraic rules. This improved plan and a cost model for preference evaluation are used for generating alternative plans that interleave preference evaluation and query processing in different ways and for picking the plan with the cheapest estimated cost.
- The execution engine realizes the execution of the query plan selected by the query optimizer using one of our execution methods. We discuss

B. Related Work

The concept of preference-aware query processing appears in many applications, where there is a matter of choice among alternatives, including query personalization [10], [18], [20], recommendations [4] and multi-criteria decision making [9], [13]. We discuss prior work with respect to how preferences are represented in the context of relational data and how they are integrated and processed in queries. In representing preferences, there are two approaches. In the qualitative approach, preferences are specified using binary predicates called preference relations [5], [10], [18]. In quantitative approaches, preferences are expressed as scores assigned to tuples [6], [23] be

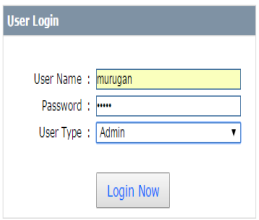
specified based on any combination of scores, confidences and context. Our framework allows us to process in a uniform way all these different query and preference types. In terms of preference integration and processing, one approach is to translate preferences into conventional queries and execute them over the DBMS [14], [19], [20], [21], [24]. Several efficient algorithms have been proposed for processing different types of queries, including top-k queries [13] and skylines [9]. These algorithms as well as query translation methods are typically implemented outside the DBMS. Thus, they can only apply coarse grained query optimizations, such as reducing the number of queries sent to the DBMS. Further, as we will also demonstrate experimentally plug-in methods do not scale well when faced with multi-join queries or queries involving many preferences. Native implementations modify the database engine by adding specific physical operators and algorithms. RankSQL [23] extends the relational algebra with a new operator called rank that enables pipelining and hence optimizing top-k queries. Another example of operator is the winnow operator [10], which selects all tuples corresponding to the Pareto optimal set. Our approach is different from existing works in several ways. First, existing techniques are limited to a particular type of query. In contrast to these approaches, we consider preference evaluation (how preferences are evaluated on data) and selection of the preferred tuples that will comprise the query answer as two operations. We focus on preference evaluation as a single operator that can be combined with other operators and we use its algebraic properties in order to develop generic query optimization and processing techniques. Finally, we follow a hybrid implementation that is closer to the database than plug-in approaches yet not purely native, thus combining the pros of both worlds. A different approach to flexible processing of queries with preferences is enabled in FlexPref [22]. FlexPref allows integrating different preference algorithms into the database with minimal changes in the database engine by simply defining rules that determine the most preferred tuples. Once these rules are specified a new operator can be used inside queries. It is worth noting that both FlexPref and our work are motivated by the limitations of plug-in and native approaches. FlexPref approaches the problem from an extensibility viewpoint. Our focus is on the problem of preference evaluation as an operator that is separate from the selection of preferred answers, and we study how this operator can be integrated into query processing in an effective yet not obtrusive to the database engine way.

C. Proposed Methodology

In this paper, we first construct an extended query plan that contains all operators that comprise a query and we optimize it. Then, for processing the optimized query plan, our general strategy is to blend query execution with preference evaluation and leverage the native query engine to process parts of the query that do not involve a prefer operator. Given a query with preferences, the goal of query optimization is to minimize the cost related with preference evaluation. Based on the algebraic.

III. RESULTS AND DISCUSSION

The implementation results can be shown as figure below



IV.CONCLUSION

Various web based application makes the human work schedule as a simple one, as well as in this web based project tracking and resolving the clients send their queries to the developer of the software via online. The people who are going to access in this software project are the system administrator, staff, and user.

The administrator controls the number of modules and the administrator maintains user queries and everything. The person who develop this project and who answer various queries of the user is the staff. The users are the persons who are use the software and asking query to developer.

V. REFERENCES

[1]. DBLP computer science bibliography. http://dblp.uni-trier.de/. 1K. Xie , J. Wu , W. Yang , C.Y. Sun , K-means clustering based on density for scene image classification, in: Proceedings of the 2015Chinese Intelligent Automation Conference, 2015, pp. 379–438 .

[2]. Y.W. Chen, D.H. Lai, H. Qi, J.L. Wang, J.X. Du, A new method to estimate ages of facial image for large database, Multimed. Tools Appl. (2015) 1–19, doi: 10. 1007/s11042-015-2485-9 .

[3]. W. Zhang, J. Li, Extended fast search clustering algorithm: widely density clusters, no density peaks. arXiv:1505.05610, 2015, doi: 10.5121/csit.2015.50701 .

[4]. G. Chen , X. Zhang , Z.J. Wang , F.L. Li , Robust support vector data description for outlier detection with noise or uncertain data, Knowledge-Based Syst. 90 (2015) 129–137 .

[5]. L. Parsons , E. Haque , H. Liu , Subspace clustering for high dimensional data: a review, ACM SIGKDD Explor. Newslett. 6 (1) (2004) 90–105 .

[6]. H.L. Chen , B. Yang , G. Wang , J. Liu , X. Xu , S.J. Wang , D.Y. Liu , A novel bankruptcy prediction model based on an adaptive fuzzy k-nearest neighbor method, Knowledge-Based Syst. 24 (8) (2011) 1348– 1359 .

[7]. T. Basu , C.A. Murthy , Towards enriching the quality of k-nearest neighbor rule for document classification, Int. J. Mach. Learn. Cybern. 5 (6) (2014) 897–905.

[8]. M. Hao , Z. Qiao , Identification of the pesticide fluorescence spectroscopy based on the PCA and KNN, in: Proceedings of 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), 3, 2010, pp. 184–186.

[9]. P.S. Hiremath , M. Hiremath , 3D face recognition based on radon transform, PCA, LDA using KNN and SVM, Int. J. Image Graph. Signal Process. 6 (7) (2014) 36–43 .

[10]. J. Schneider , M. Vlachos , Fast parameterless density-based clustering via random projections, in: Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, 2013, pp. 861–866 .
[11]. E. Aksehirli , B. Goethals , E. Muller , J. Vreeken , Cartification: A neighborhood preserving transformation for mining high dimensional data, in: Proceedings of the 13th IEEE International Conference on Data Mining (ICDM), 2013, pp. 937– 942 .
[12]. A. Gionis , H. Mannila , P. Tsaparas , Clustering aggregation, ACM Trans. Knowl. Discov. Data 1 (4) (2007) 341–352 .
[13]. L. Fu , E. Medico , FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data, BMC Bioinf. 8 (1) (2007) 399–408 .
[14]. P. Fränti , O. Virmajoki , Iterative shrinking method for clustering problems, Pattern Recognit. 39 (5) (2006) 761–775 .
[15]. S.F. Ding , H.J. Jia , Z.Z. Shi , Spectral clustering algorithm based on adaptive Nyström sampling for big data analysis, J. Softw. 25 (9) (2014) 2037–2049 .