IJDCST @November Issue- V-1, I-7, SW-48
ISSN-2320-7884 (Online)
ISSN-2321-0257 (Print)

Adaptive Monitoring Solution Satisfy User Profiles for Targeted Online Data Delivery

[1]Rathna Kumari Kilari , [2]Y.Surekha , [3]G.Lalitha Kumari , [4]V.Sowjanya

[1] Student, PVPSIT, KANURU, VIJAYAWADA, KRISHNA DIST.

[2] Assistant Prof, PVPSIT,KANURU,VIJAYAWADA,KRISHNA DIST.

[3] Sr.Assistant Prof, PVPSIT,KANURU,VIJAYAWADA,KRISHNA DIST.

[4] Assistant Prof, PVPSIT,KANURU,VIJAYAWADA,KRISHNA DIST.

Abstract: We present an alternative and more flexible approach that maximizes user utility by satisfying all users. It does this while minimizing the usage of system resources. We discuss the benefits of this latter approach and develop an adaptive monitoring solution Satisfy User Profiles (SUPs). Through formal analysis, we identify sufficient optimality conditions for SUP. Using real (RSS feeds) and synthetic traces, we empirically analyze the behavior of SUP under varying conditions. Our experiments show that we can achieve a high degree of satisfaction of user utility when the estimations of SUP closely estimate the real event stream, and has the potential to save a significant amount of system resources. We further show that SUP can exploit feedback to improve user utility with only a moderate increase in resource utilization.

I. INTRODUCTION

The Web is becoming a universal medium for disseminating information of all kinds, including highly dynamic information. Significant amount of valuable dynamic information is being posted to the Web and people want to access it. Direct manual viewing of dynamic Web pages is not an adequate mode of access for one or both of the following two reasons:

- Most information posted on the Web is not made available forever and may disappear or be replaced by new information at any time
- Many applications require automated synthesis of information from multiple dynamic Web sources

There is significant interest in systems that monitor and process updates to frequently updated Web pages automatically. THE diversity of data sources and Web services currently available on the Internet and the computational Grid. We consider a number of scenarios including RSS news feeds and auctions on the commercial Internet and scientific data sets and Grid computational resources.

Push, pull, and hybrid protocols have been used to solve a variety of data delivery problems. Push-based consistency in the context of caching dynamic Web content. Push is typically not scalable and reaching a large number of potentially transient clients is expensive. Pushing information may overwhelm the client with unsolicited information. Several hybrid push-pulls solutions have also been presented. We focus on pull-based resource monitoring and satisfying user profiles. The burden of when to probe an RSS resource lies with the client. Much of the existing research in pull-based data delivery casts the problem of data delivery as follows: Given some set of limited system resources. We refer to this problem as OptMon1.

A solution toOptMon1 is accompanied by the need to meet rigid a priori bounds on system resource constraints. A rigid a priori setting may also have the unintended consequence of forcing excessive resource consumption even when there is no additional utility to the user. While resource consumption is dynamic and changes with needs with this class of problems user needs are set as the constraining factor of the problem. We present an optimal algorithm in the OptMon2 class. SUP is simple yet powerful in its ability to generate optimal scheduling of pull requests. SUP is an online

algorithm; at each time point. SUP depends on an accurate model of when updates occur to perform correctly.

Most work on continuous query processing assumes that data is "pushed" into the query engine in the form of *data streams*. Only heuristics with no formal guarantees on effectiveness have been proposed for converting pull- oriented Web sources into push-oriented data streams. Periodic polling breaks down in the presence of a large number of frequently updated Web sources, when resources become inadequate for polling all Web pages at a fast rate.

Dynamic Web pages undergo updates over time and each updated version of the page potentially contains new information of value to the application. Due to the nature of Web protocols, obtaining updates to Web pages generally requires polling those pages. Typically, it is not feasible or desirable to provision systems with adequate communication bandwidth and processing power to support exhaustive and rapid polling of a large number of Web pages. Polling must be performed selectively and some criteria for deciding when to poll each page must be established. It may not be possible to capture all changes to all pages of interest in a timely fashion due to resource limitations.

II. Monitoring the Web & Dual Framework

Our models for the Web monitoring scheduling problem and the way in which Web pages change extend the framework. Let P be the set of Web pages under consideration for monitoring. Time is divided into discrete time instants and monitoring is performed in epochs of N consecutive time instants. We focus on the problem of scheduling monitoring of the pages in P during a single epoch. The monitoring a page includes the duties of fetching the page from its remote source and determining whether it has undergone one or more changes of interest. This simplification is based on the assumption that the fixed overhead for the operations required is the dominant factor. Let C denote the maximum number of pages that can be monitored in a single time instant. Value of C depends on the availability of resources for monitoring. If C equals or exceeds the number of pages then the scheduling problem is trivial: simply monitor each page at every instant. A legal *monitoring schedule* for an epoch is one that performs at most C monitoring of pages during each time instant $T_1, T_2, \ldots, T_N$.

Let $R = \{r_1; r_{2;} \ldots ; r_n\}$ be a set of resources and T be an epoch and let $\{T_1; T_2; \ldots ; T_N\}$ be a set of equidistant chronons1 in T. schedule $S = \{si,j\}_{i=1\ldots n;j=1\ldots N}$ is a set of binary decision variables.

The OptMon1 formulation assumes that system constraints are hard constraints where their assignment is a priori independent of specific user utility maximization task. OptMon1 involves a system resource constraint of the maximum number of probes per chronon for all resources. This constraint represents the number of monitoring tasks that a Web monitoring system can allocate per chronon for the task of maximizing the utility gained from capturing updates to Web pages. The benefits of OptMon1 are apparent whenever there are hard system constraints on resources. OptMon1 formulation has two main limitations:

- We expect that there will be periods of varying intensity with respect to the intensity of updates at the server(s) as well as the intensity of probes needed to satisfy client profiles
- The rigidity of OptMon1 algorithms with respect to system resource allocation

Solutions to OptMon1 have not dynamically attempted to reduce resource consumption. Once the upper bound on bandwidth has been set, bandwidth can no longer be adjusted and user needs may not be fully met. We propose a dual formulation OptMon2 setting the fulfillment of user needs as the hard constraint. It assumes that the system resources that will be consumed to satisfy user profiles should be determined by the specific profiles and the environment.

III. TARGETED DATA DELIVERY MODEL

The centerpiece of our model is the notion of execution intervals and a simple modeling tool for representing dynamically changing client needs. We then turn our attention to the formal definition of a schedule and the utility of probing. We present a case

study using RSS a popular format for publishing information summaries on the Web. The diverse data types are nowadays available as publications in RSS. The use of RSS feeds is continuously growing. A user of such a reader can customize her profile by specifying the rate of monitoring each RSS feed and is supported by a pull-based protocol. The RSS protocol was extended with special metatags such as server side TTL that hint when new updates are expected. Despite these, feature a client who is only interested in being alerted of updates for a particular item in some news category. A profile should be easy to specify and sufficiently rich to capture client requirements. Specified in the trigger part of the notification rule, the trigger condition is immediately evaluated and if it is true.

The period in which a notification rule is executable was referred to in the literature as life. We emphasize here the difference between the executable period of a notification (life) and the period in which rules. An execution interval starts with an event and its length is determined by the relevant life policy. With pull-based monitoring, content is delivered upon request with limited effectiveness in estimating object freshness. Hybrid approach combines push and pull, based either on resource constraints or on role definition. The mediator can monitor servers by periodically pulling their content and determine when to push data to clients based on their content delivery profiles.

It was argued that the use of an update model based on Poisson processes suits well updates in a Web environment. The Poisson processes are suitable for modeling a world where data updates are independent from one another such as updates to auction Web sites. Such a model reflects well scenarios in which e-mails arrive more rapidly during work hours and more bids arrive toward the end of an auction.

IV. SUP Algorithm

Let $R = \{r_1; r_2; \ldots; r_n\}$ be a set of resources and T be an epoch and let $\{T_1; T_2; \ldots; T_N\}$ be a set of equidistant chronons1 in T. schedule $S = \{s_{i,j}\} \in S$ be a schedule. $P = \{p_1; p_2; \ldots; p_m\}$ be a set of user profiles. Given a notification rule η and the set of its execution intervals EI(η). SUP identifies the set of resources $Q_I \eta$ Domain(Q, η) that must be probed in an execution interval $I \in$ EI(η).

The main intuition behind the SUP algorithm is to identify the best candidate chronon in which the assignment of probes to resources maximizes the number of execution intervals that can benefit from each probe. We identify the best candidate chronons by delaying the probes of execution intervals to the last possible chronon in which the utility is still positive. We now provide a description of the algorithm. Pseudo code of the SUP algorithm and the two routines:

a. *AdaptiveEIsUpdate*
b. *UpdateNotificationEIs*

The algorithm builds a schedule iteratively. It starts with an empty schedule and repeatedly adds probes. Then determines the earliest chronon in which to probe and the notification rule associated with this monitoring task. SUP depends on an accurate set of execution intervals to perform correctly. Determining a set of execution intervals suffers from two main problems:

- The underlying update model that is used to compute the execution intervals is stochastic in nature
- It is possible that the underlying update model is incorrect and the real data stream behaves differently than expected

We propose to exploit feedback from probes to revise the probing schedule in a dynamic manner after each monitoring task. We first introduce the general scheme of SUP that addresses the first problem and does not require changes to any parameters.

SUP uses the AdaptiveEIsUpdate routine to apply the adaptive modifications. Routine first applies adaptive modification to notification rule η by recalculating a new execution interval I^* to be scheduled. The routine determines a set of notification rules that may be associated with execution intervals that need to be modified by identifying those notification rules that reference resource r_i in their trigger part. The UpdateNotificationEIs routine is called to ensure that

resources that belong to overlapping intervals are only probed once. Let l = η be the assignment of SUP where η is the notification rule whose execution interval I is processed at time T_j.

V. ALGORITHM PROPERTIES

SUP assumes the availability of a stream of execution intervals generated using this or that update model. The algorithm to focus on the monitoring of execution intervals, SUP optimal solution depends only on the number of execution intervals it is required to consider during the monitoring task. SUP is executed in an online fashion, where execution intervals are introduced right before they are required to be considered by SUP. We can exploit the feedback gathered during the monitoring scheme to better improve the probing of future scheduled execution intervals by adaptive monitoring. SUP accesses O(K) execution intervals bounded by Nn. We expect K to be much smaller than Nn. since K serves as a measure of the amount of data users expect to receive during the monitoring process.

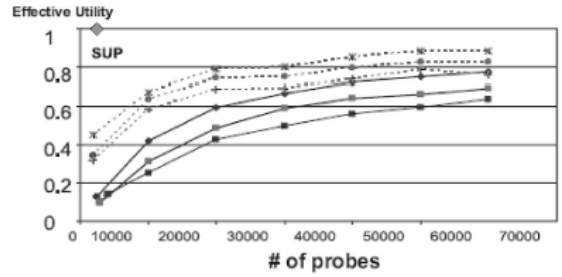
The dual optimization problems OptMon1 and OptMon2 cannot be compared directly. User profiles satisfactions may violate system constraints and satisfying system constraints may fail to satisfy user profiles. Whenever the resources consumed by SUP satisfy the system constraints of OptMon1 then SUP is guaranteed to solve the dual OptMon1 and maximize user utility. Assume that in the schedule of SUP, the maximum number of probes in any chronon satisfies M. SUP utilizes in each chronon only the amount of probes that is needed to satisfy the profile expressions. A schedule S, generated by SUP with no bound on system resource usage and a set of desired system resource constraints. S can be used to avoid over probing in chronons when less updates are expected.

VI. EXPERIMENTAL ANALAYSIS

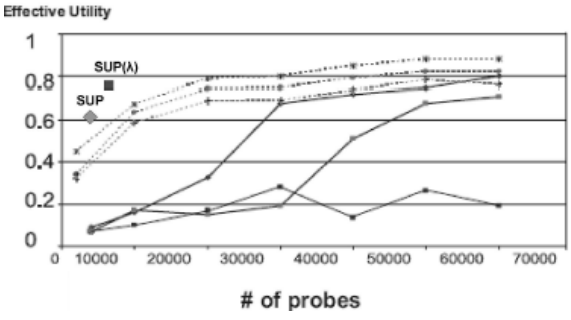
We implemented SUP experimented with it on various trace data sets, life parameters and profiles. Traces of update events include real RSS feed traces and synthetic traces. We also implemented WIC to determine a schedule for OptMon1 and TTL as another OptMon2 solution. Recall that while OptMon1 problems set hard constraints on system resources and OptMon2 aims at minimizing system resource utilization. OptMon2 secures the full satisfaction of user specification, while OptMon1 can only aim at maximizing it. We make the following indirect comparison:

a. We compare the system resource utilization of the different solutions
b. Given some level of system resource utilization when we compare the effective utility of the different solutions

The TTL solution will use the server provided TTL to determine when the next probe to a resource should be to satisfy a profile. WIC is a solution to the OptMon1 provides the system resource utilization and corresponding utility of the three algorithms. We add a parameter denoted by M to represent a system constraint on the total number of probes allowed per chronon.



(a)



(b)

Fig. 1. SUP, WIC, and TTL for Synthetic Data 1 data set for (a) FPN(1) and (b) Poisson.

The optimal number of probes for SUP is 2462 for this data set. We also varied the M level for WIC. Given that, TTL is allowed to probe the same total number of probes as WIC (N.M) and assuming that there are n resources. We now focus on the data set

and the Poisson update model of (b). The effective utility for SUP is about 0.62, the effective utility is represented by a single point.

VII.CONCLUSION

We focused on pull-based data delivery that supports user profile diversity. The minimizing the number of probes to sources is important for pull-based applications to conserve resources and improve scalability. The solutions that can adapt to changes in source behavior are also important due to the difficulty of predicting when updates occur. We have addressed these challenges with a new formalism of a dual optimization problem, reversing the roles of user utility and system resources. We have formally shown that SUP is optimal for OptMon2 and under certain restrictions can be optimal for OptMon1 as well. SUP is adaptive and can dynamically change monitoring schedules. The experiments show that using feedback in SUP improves the performance with a moderate increase in the number of needed probes. OptMon2 is defined in such a way that satisfaction of a user profile is a hard constraint. Profile may state preferences rather than hard constraints. Extending the problem to handle profile preferences poses a new challenge to this problem. The algorithmic solution changes to identify the Pareto curve of feasible solutions. Another way of adding preferences to this work is by redefining utility to include a variety of dimensions. We consider this problem as another challenge and an avenue for future research.

VIII. REFERENCE

[1] Haggai Roitman, Avigdor Gal and Louiqa Raschid, "A Dual Framework and Algorithms for Targeted Online Data Delivery," IEEE Transactions on Knowledge and Data Engineering, VOL. 23, NO. 1,pp. 5-21, JANUARY 2011.

[2] Sandeep Pandey, Kedar Dhamdhere_, Christopher Olston, "WIC: A General-Purpose Algorithm for Monitoring Web Information Sources," NSF ITR grants CCR-0085982 and CCR-0122581, Proceedings of the 30th VLDB Conference, Toronto, Canada, 2004

[3] C. Liu and P. Cao, "Maintaining Strong Cache Consistency on the World Wide Web," Proc. Int'l Conf. Distributed Computing Systems (ICDCS), 1997.

[4] J. Yin, L. Alvisi, M. Dahlin, and A. Iyengar, "Engineering Server- Driven Consistency for Large Scale Dynamic Web Services," Proc. Int'l World Wide Web Conf. (WWW), pp. 45-57, May 2001.

[5] H. Liu, V. Ramasubramanian, and E.G. Sirer, "Client and Feed Characteristics of rss, a Publish-Subscribe System for Web Micronews," Proc. Internet Measurement Conf. (IMC), Oct. 2005.

[6] A. Adi and O. Etzion, "Amit—The Situation Manager," Int'l J. Very Large Data Bases, vol. 13, no. 2, pp. 177-203, May 2004

[7] J. Cho and H. Garcia-Molina. Synchronizing a database to improve freshness. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 2000.

About Author:

Rathna kumari kilari received the MCA PG Degree from JMJ College for Women,Tenali,affiliated to Acharaya Nagarjuna University, India. She has more than 3 years teaching experience. She is currently purshuing her M.tech in PVP Siddhartha Institute of Technology,Kanuru, Vijayawada affiliated to JNTU Kakinada University, India. Her interests are Networks,Data Mining.

Y.Surekha received the M.tech (CSE) Degree from Bapatla engineering college, Acharaya Nagarjuna University,India.She has more than 4 years teaching experience. She is currently working as an Asst Professor in the Dept of Computer Scienc Engineering, PVP Siddhartha Institute of Technology, Vijayawada and affiliated to JNTU Kakinada University, India. Her interests are Neural Networks, Data Mining.