Smart Miner: A New Framework for Mining Large Scale Web Usage Data

Murat Ali Bayır

University at Buffalo
The State University of New York

Ismail Hakkı Toroslu

METU
NORTHERN CYPRUS CAMPUS

Ahmet Coşar

Middle East Technical University

Güven Fidan

AGMLAB
information technology

Sponsored BY

TÜBİTAK

Ekinoks Yazılım

TURKCELL

AGMLAB information technology

Outline

- Motivation and Contributions
- Smart Miner Framework
- Session Construction Problem
- Pattern Discovery
- Distributed Implementation of Smart Miner
- Experimental Results
- Decision Support System Application

Motivation and Contributions

Why we design Smart Miner?

- Mining Web Usage Logs is important process for improving commercial web sites structurally and semantically.
- Due to security reasons and changes in the internal structure of the web sites, site owners wants to keep their own data and process their server logs offline. (installation code based systems are not attractive)
- For the web siwith large visitor population, the problem should be handled by distributed system due to size of incoming click through data everyday.
- Site owners may want to execute intelligent filters over usage data to detect specific type of user behavior or analyzing usage of specific structure in the web site.
- To solve the problems addressed above we have proposed Smart Miner framework.

Motivation and Contributions

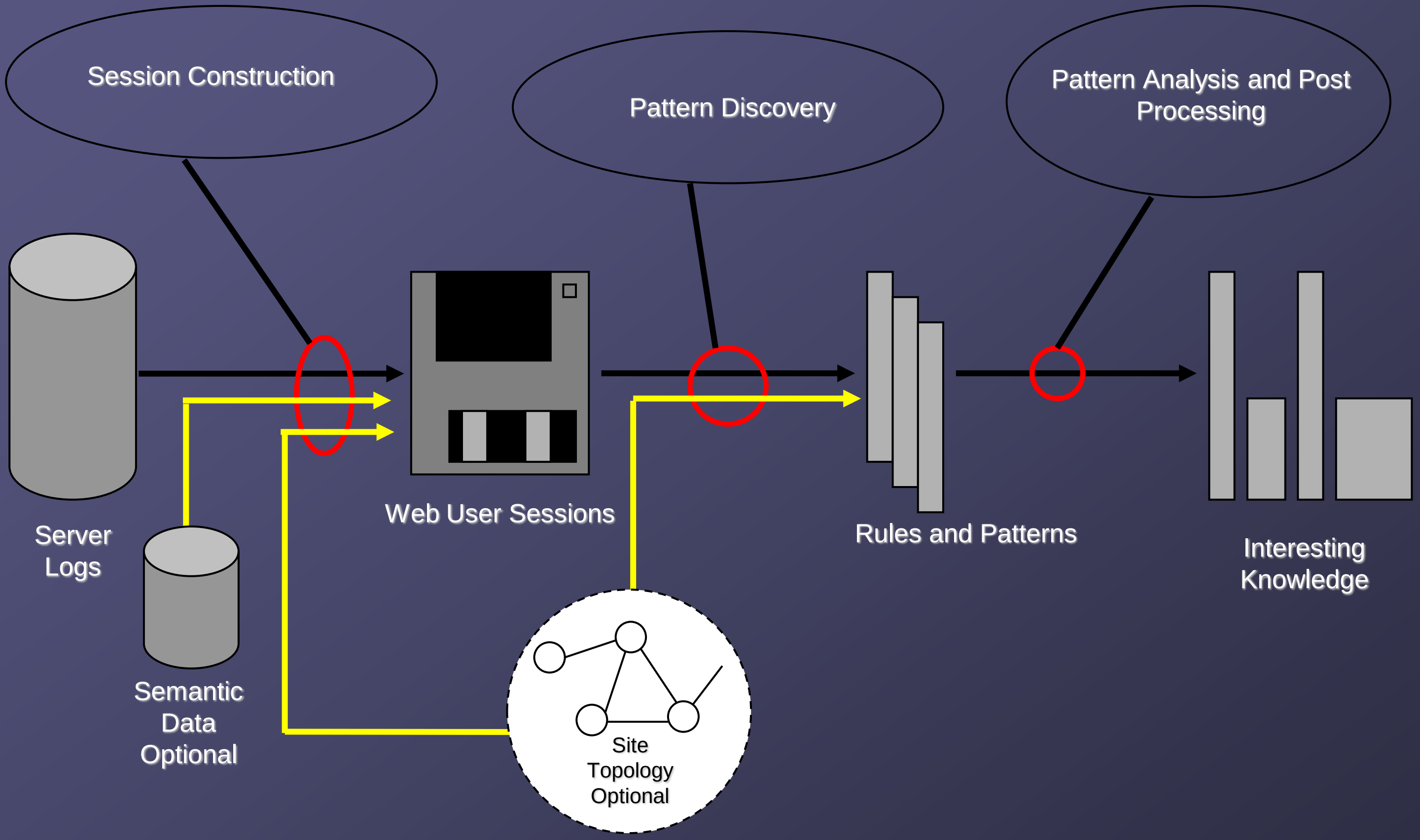
Contributions

- Implemented Scalable Framework for WUM
- Modeled Session Construction problem as Graph Problem and proposed Smart-SRA.
- Proposed Efficient Solution for Pattern Discovery
- Proposed new Web User Model for Simulation purposes
- Map/Reduce version of Session Construction and Pattern Discovery
- Show scalability framework over very large real data
- Decision Support System Application

Outline

- Motivation and Contributions
- Smart Miner Framework
- Session Construction Problem
- Pattern Discovery
- Distributed Implementation of Smart Miner
- Experimental Results
- Decision Support System Application

Smart Miner Framework



Session Construction

- Most important phase of WUM since the later phases depends on the sessions generated by this phase.
- General Session (Definition): a sequence of web page requests by single user on particular web domain for a specific time period.
- Two Traditional Approaches for Session Construction from Server Logs (called Reactive Approaches)
  - Time Oriented Approach
  - Navigation Oriented Approach

Session Construction

- Time Oriented Approaches do not consider link information.
  - ❑ Type-1: Duration of a discovered session is limited with a threshold δ_1
  - ❑ Type-2: Time spend on any page limited with threshold δ_2

- Navigation Oriented Approaches considers link information. Adding page P_{N+1} to a session $[P_1, P_2, \ldots, P_N]$ is performed as follows:
  - ❑ If P_N has a links towards or referrer of P_{N+1} (1)

    $[P_1, P_2, \ldots, P_N, P_{N+1}]$
  - ❑ Else P_k is the nearest page satisfying (1) for P_{N+1} add backward browser movement until P_k

    $[P_1, P_2, \ldots, P_N, P_{N-1}, P_{N-2}, \ldots, P_k, P_{N+1}]$

Session Construction

Drawbacks of Time and Navigation Oriented Heuristics

- Time Oriented Approach has no link information
- Backward Movements are major problem of Navigation Oriented Heuristics. Recent studies HCI area addressed that majority of backward movements occurs due to location of pages rather than content.
- Backward Movements cause performance problem in later application phases like recommender systems and page pre-fetching.
- We propose new session model called Link Based Session Model to solve problems mentioned here.

Session Construction

Link Based Session Model

A Link based Session contains a set of navigation sequences which corresponds to paths on the web graph. If $S = \{S_1, S_{2,} \ldots\ldots S_n\}$ is a session with n navigation sequences and $\forall S_x = <Px_1, \ldots Px_i, Px_{i+1}, \ldots Px_n>$ must satisfy the following rules:

Timestamp Ordering Rule:

- ❑ $\forall i: 1 \leq i < n, Tx_i < Tx_{i+1}$
- ❑ $\forall i: 1 \leq i < n, (Tx_{i+1} - Tx_i) \leq r$ (page stay time)
- ❑ $(Tx_n – Tx_1) \leq \delta$ (session duration time)

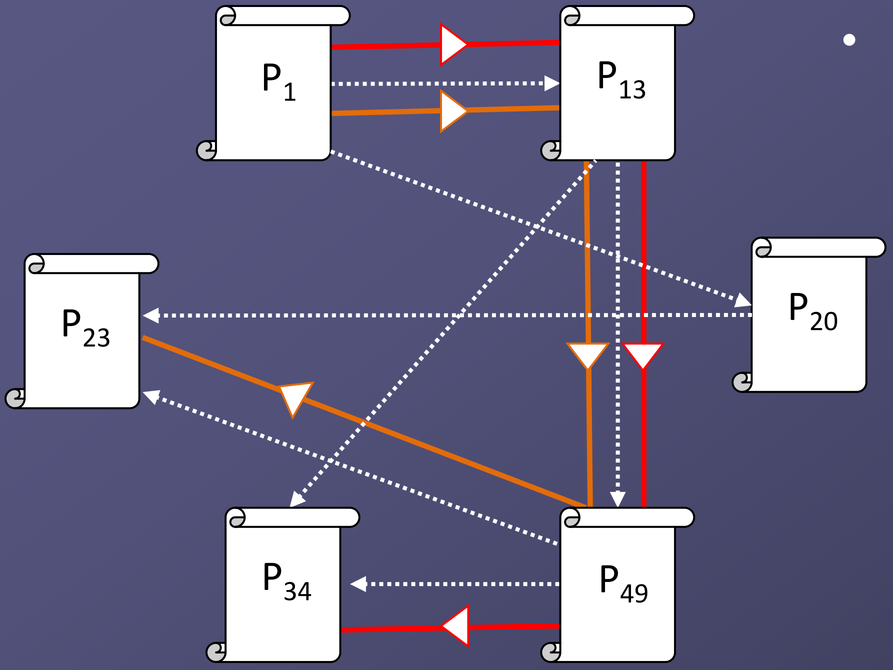
Topology Rule: = true

- ❑ $\forall i: 1 \leq i < n$, Link$[Px_i, Px_{i+1}]$ = true OR Ref$[Px_i, Px_{i+1}]$ = true

Maximality Rule:

- ❑ $\forall S_x \in S$ there exist no S_y such that $S_y \in S$ and $S_x \subset S_y$ = (subsequence relation similar to substring relation)

Smart-SRA



- Smart-SRA (Smart Session Reconstruction Algorithm) is the new algorithm that produce link based sessions from web usage logs
- Smart-SRA has 2 phases.
  - In Phase-1 Candidate Sessions are constructed from web usage logs by applying page stay and session duration time criteria.
  - In Phase-2 Candidate Sessions are partitioned into maximal navigation sequences satisfying properties of link based session model. (Main Phase). All navigation sequences generated by Smart-SRA may overlap but they are maximal.

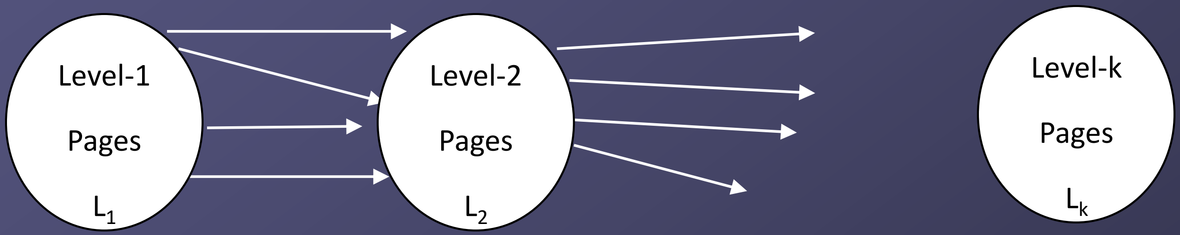
For the candidate session [P_1, P_{13}, P_{49}, P_{34}, P_{23}] two navigation sequences are generated:

- [P_1, P_{13}, P_{49}, P_{34}]
- [P_1, P_{13}, P_{49}, P_{23}]

Smart-SRA

Complexity in Extreme Cases

- The complexity of Smart-SRA is exponential on specific cases which contains crawler like sessions including Breath First Order of web pages.



- For the candidate session [{Level-1}, {Level-2}, …. {Level-k}] the number of navigation sequences are L_1 x L_2 x … $L_{k.}$, and the complexity is also exponential in terms of average length candidate sessions (len) and out degree of pages in candidate sessions (out). ~ $O(len)^{out}$

- For the candidate sessions including depth first order of web pages the complexity is linear. O(len)

April 20th-24th, 2009
18th Int. World Wide Web Conference

Smart-SRA

Complexity in Average Case

- The average complexity of Smart-SRA is polynomial $O(len^2 out^2)$
- (Proof): Phase-2 of Smart-SRA removes all pages in candidate session at each step that does not have referrer with smaller time stamp.
- We define an expectation function which gives the number of web pages subtracted from candidate session at each iteration:

$$E(P) = \sum_{\forall P_x \in P} p(x) f(x)$$

- Here p(x) is the probability of removing page P from the candidate session (depends on probability of no links before any page) and f(x) is contribution of each page to sum (clearly 1).
- By using uniform probability model p(x) is $(1-(out)/len)^{k-1}$ for k-th page in session. Complexity becomes polynomial under this assumption.
- In fact out tends to be very small in real user sessions and it is efficient to run Smart-SRA as we shown in experiments sections.

Outline

- Motivation and Contributions
- Smart Miner Framework
- Session Construction Problem
- Pattern Discovery
- Distributed Implementation of Smart Miner
- Experimental Results
- Decision Support System Application

Pattern Discovery

- We have implemented Sequential Apriori with topological constraint, the important feature of our approach is listed below:
  - Each pattern satisfy topology constraint. $\forall P_k$ of Pattern P, (k<|P|), Link[P_k][P_{k+1}]=TRUE.
  - String matching must be satisfied in support checks. [1, 2, 3] does not support pattern <1, 3> although 3 comes after 1 in both of them. However, it supports <1, 2>.
  - While calculating the supports, pattern specific statistical attributes of web pages is calculated such as average visit time of P_k in pattern P.

Pattern Discovery

Sequential Apriori

- Inputs:
  - Sessions constructed by Smart-SRA
  - Minimum support for frequent patterns
- At each iteration we join length-k patterns with length-1 patterns (by using topological constraint)
  - If the support of the length-k+1 pattern is greater than the required support, it becomes a supported pattern.
  - In addition, the new length-k+1 pattern becomes maximal.
  - The extended length-k pattern and the appended length-1 pattern become non-maximal.

Outline

- Motivation and Contributions
- Smart Miner Framework
- Session Construction Problem
- Pattern Discovery
- Distributed Implementation of Smart Miner
- Experimental Results
- Decision Support System Application

Distributed Implementation of Smart Miner

- We have implemented Map/Reduce version of Session Construction and Pattern Discovery phases over the cluster of 100 computers.

- In Map/Reduce, large computations are represented by sequence of map and reduce operators.
- In order to employ map/reduce, the data should be stored in the format of key/value pairs.

Distributed Implementation of Smart Miner

Session Construction

- **Mapper:** read usage data containing Url and IP fields, create key: <domain(Url),IP>
- **Reducer:** get all urls belonging to same domain and same IP on same machine, execute Smart-SRA

2 Phased Pattern Discovery

- First Phase:
  - Mapper: Navigation Sequences are distributed to different machines by using random hash function with respect to number of idle machines N
  - Reducer: Frequency of each candidate pattern on each node is calculated.

- Second Phase:
  - Mapper: <candidate pattern, frequency> pair is emitted (N times)
  - Reducer: N pairs of same candidate pattern reduced to same machine, frequencies are merged to calculate support.

Outline

- Motivation and Contributions
- Smart Miner Framework
- Session Construction Problem
- Pattern Discovery
- Distributed Implementation of Smart Miner
- Experimental Results
- Decision Support System Application

Experimental Results

- Accuracy Experiments shows the correlation between patterns generated from sessions of heuristics and patterns generated from the real Link Based sessions that is provided by evaluation framework.
  - These experiments also analyzes effect navigation behaviors on sessions.
  - How the backward movements and discarding link information affect the performance of NO and TO approaches in terms of link based session evaluation
    - Simulated Data
    - Real Data
- HPC Experiments shows the scalability of the framework on huge size data.
  - Real Data

Experimental Results

Agent Simulator

- Agent Simulator is implemented to model web user navigation, and analyze basic navigation behavior on the effects of sessions. Our agent simulator generates sessions wrt link based model

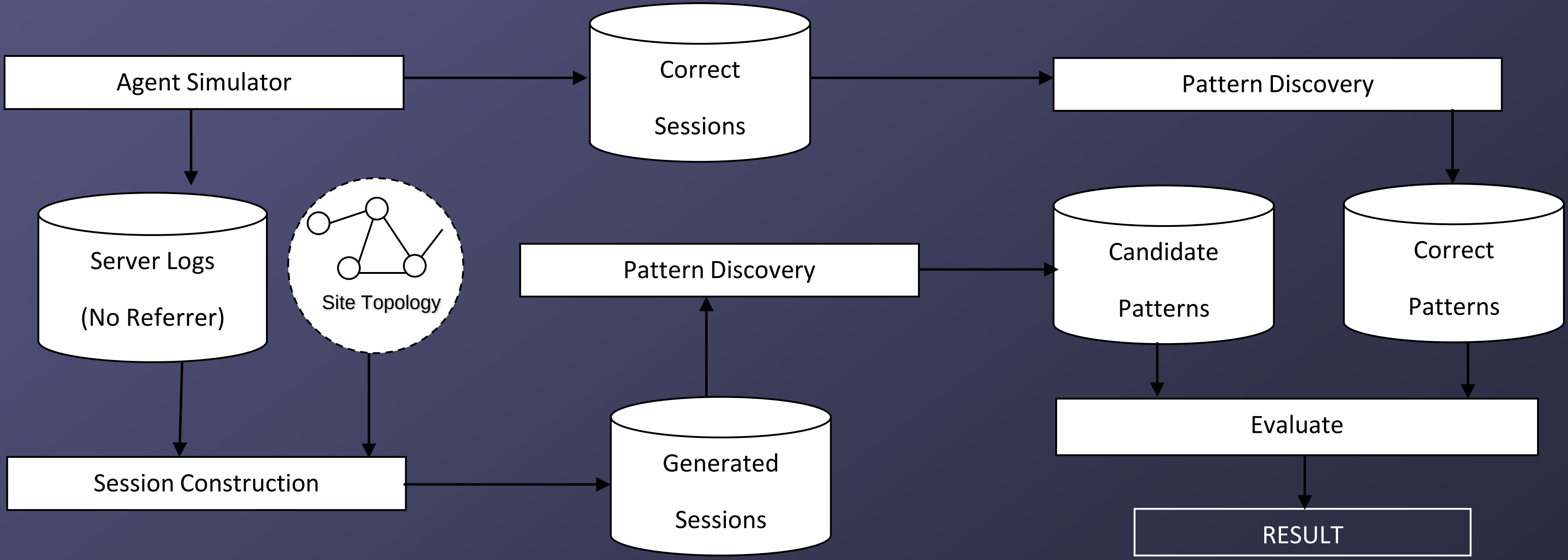
Following user behaviors are implemented:

- Session Termination (terminating whole session)
- Link From Previous Page to next Page
- Link From Current Page
- New Initial Page (start of new navigation sequence, regardless of entrance type, write address bar or enter from search engine).

Experimental Results

Evaluation Framework

- For Evaluation, agent simulator generates correct sessions (including correct navigation sequences with known referrer) and server logs without any referrer (worst case data, only <url, IP, page> and topology given outside). Each method, Smart-SRA, TO and NO heuristics are executed and patterns generated from sessions of each heuristics are compared with correct patterns.



Experimental Results

Accuracy Metric

- We have two sets Maximal Patterns of Agent simulator MP_A
- Maximal Patterns of Each Heuristics MP_H
- Accuracy is defined by geometric mean of recall and precision

$$A_H = \sqrt{(REC_H * PRE_H)}$$

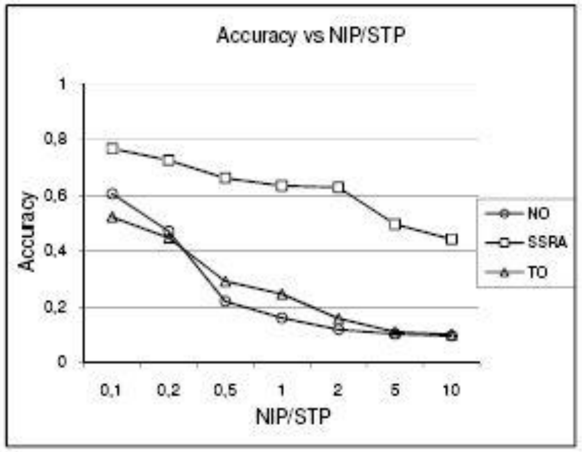
- Recall and Precision is calculated as:

$$REC_H = \frac{|MP_A \cap MP_H|}{|MP_A|} \qquad PRE_H = \frac{|MP_A \cap MP_H|}{|MP_H|}$$

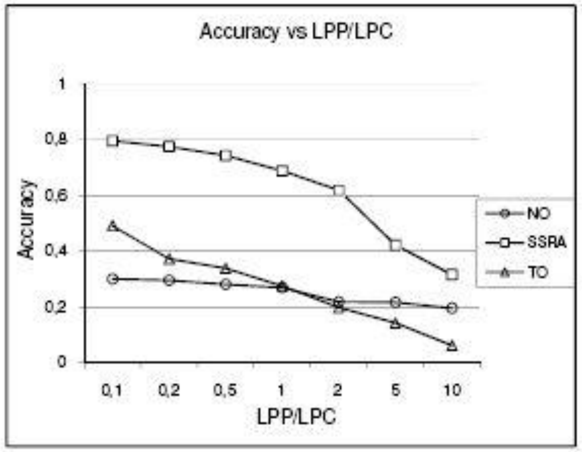
Experimental Results

Accuracy Results on Simulated Data

- We define two parameters for representing navigation behavior
  - (LPP/LPC) Link from Previous Page Prob. over Link from Current Page Prob.
    - Bigger values leads complex sessions with different overlapping paths
  - (NIP/STP) New Initial page Prob. over Sessions Termination Prob.
    - Bigger values leads complex sessions with new paths may overlap or not overlapped



(a) Accuracy vs NIP/STP

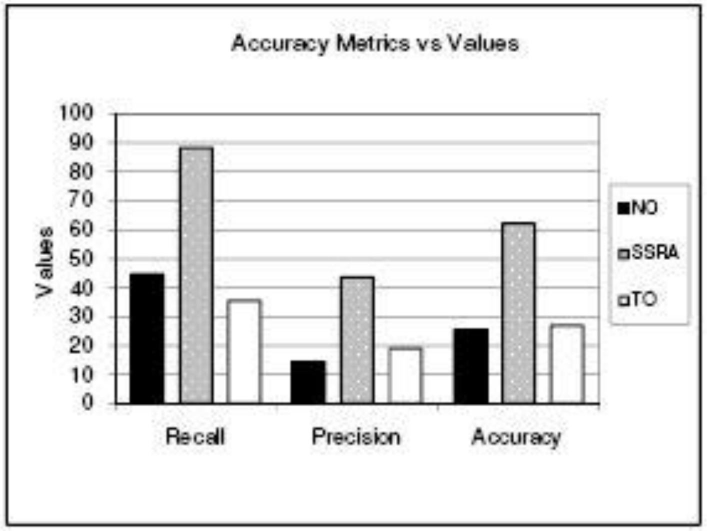


(b) Accuracy vs LPP/LPC

- {0.001, 0.0025, 0.005, 0.0075,and 0.01}
- On the average, the patterns of Smart-SRA is 30% similar to real pattern than patterns of other approaches.

Experimental Results

- For Evaluation on real data, we have used small sized real web site, AGMLAB company site.
- We replace agent simulator with client side browser action tracker script in the evaluation framework.
- From the action tracker logs we obtained real sessions and input logs to heuristics. We have tracked several actions like ‘backward clicks’, ‘open new browser’ etc.



(a) Accuracy on Real Data

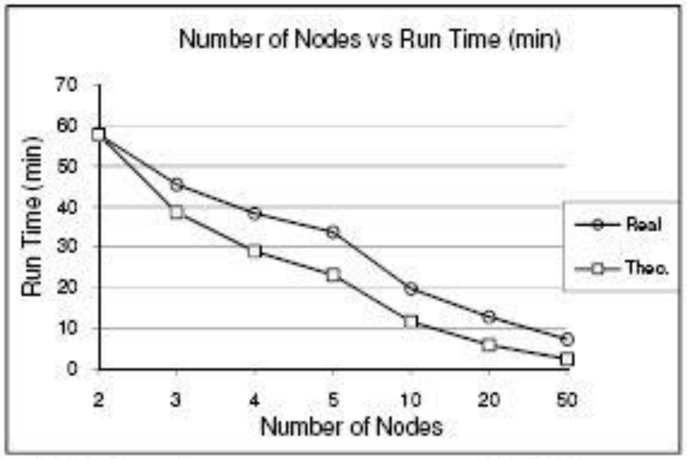
- On the average, the patterns of Smart-SRA is 35% similar to real patterns.

- We have also divided real data into two parts and used the patterns of training data to construct HMM for predicting next pages from server in the test set. In this experiment, Smart-SRA is at least 25% better than other approaches in terms of predicting next page.

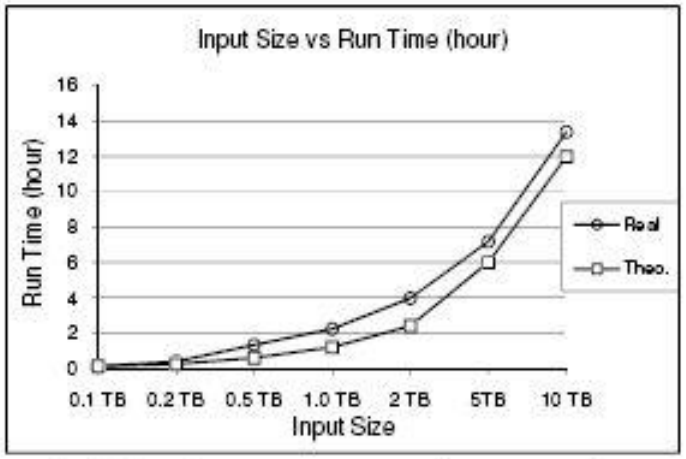
Experimental Results

Performance of Distributed Implementation

❑ Our data comes from customers of AGMLAB including web site of biggest telephone service operator having 35 billion number subscriber, their web site is daily visited by more than 200K web user.



(b) Performance on 100GB Data



(c) Performance with 50 nodes

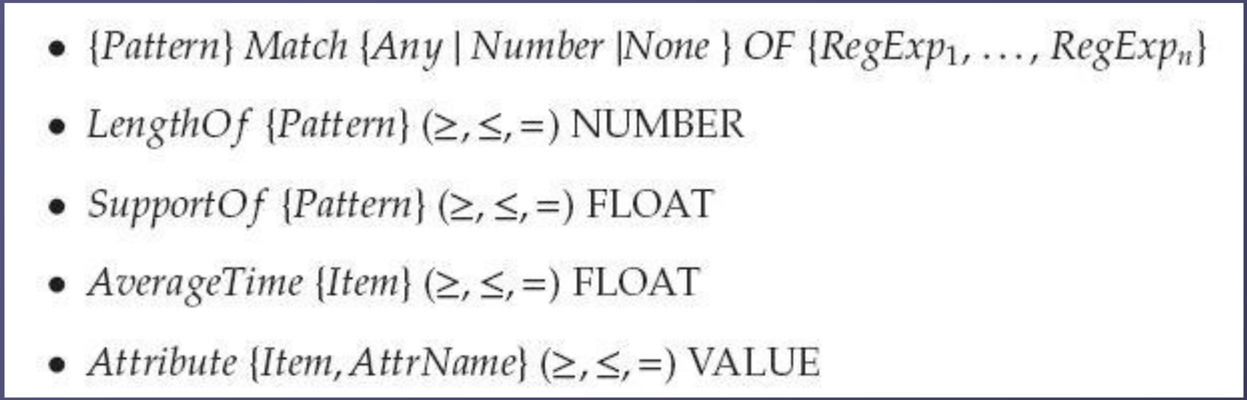
Outline

- Motivation and Contributions
- Smart Miner Framework
- Session Construction Problem
- Pattern Discovery
- Distributed Implementation of Smart Miner
- Experimental Results
- Decision Support System Application

Decision Support System

❑ We illustrate the benefits of our framework on Decision Support System application for analyzing site specific navigation behavior of web users.

❑ For Decision support systems, we have defined Filter Language over frequent patterns that includes the following 5 primitives;



- $\{Pattern\}$ *Match* $\{Any \mid Number \mid None\}$ *OF* $\{RegExp_1, \ldots, RegExp_n\}$
- *LengthOf* $\{Pattern\}$ $(\geq, \leq, =)$ NUMBER
- *SupportOf* $\{Pattern\}$ $(\geq, \leq, =)$ FLOAT
- *AverageTime* $\{Item\}$ $(\geq, \leq, =)$ FLOAT
- *Attribute* $\{Item, AttrName\}$ $(\geq, \leq, =)$ VALUE

❑ A filter is defined as collection of primitives that has nested structure with AND, OR operations between groups or primitives like:

– ((P1 AND P2) OR P3 OR P4)

❑ A set of filters are executed over frequent patterns by using map reduce job and a set of patterns that evaluates each filter TRUE are listed separately.

Decision Support System

- By using our filter language one can design a filter that analyzes the followings:
  - Navigation Behavior of web users provided via external traffic through advertisement.
  - Navigation Behavior of web users after visiting given fixed page.
  - Common exit pages of the web site.
  - The patterns leads to common errors (like ending with 404 page error).
  - The set of web pages that are less important in popular pattern in terms of elapsed time.

Conclusion

Questions, Comments, Improvements?

☺