Motivation:
The World Wide Web continues to grow at an astounding rate and is being recognized as popular resource for information, especially highly dynamic information. This brings in a need for systems that can capture and inform the users about the events that are of interest to them. One way to accomplish this would be to allow the users to interactively create triggers over the Internet. For example, in banking systems users can create triggers so that they would receive a personal notification when the balance goes below a specified amount. Even in systems like inventory management, the users would find it useful to be able to create triggers that would inform them about critical events. This would definitely lead to an explosively large number of triggers to be processed and would require a scalable trigger processing system.

Problem Statement: Formally define the problem addressed in the paper. Briefly explain the significance of the problem in the context of the course.

Given: A Relational database, trigger statements
Find: Predicate Index Structure of the triggers.
Objective: Minimize the trigger processing time.
Constraints: Performance scalable to the number of triggers.

The proposed approach achieves scalability of triggering system by exploiting the feature that most triggers share a common structure except for certain constants or they are said to share the same expression signature. The system proposed in the paper groups predicates from trigger conditions based on expression signatures (structure of the trigger) and stores these in an efficient main memory data structure. It also proposes a method for concurrent token processing and action execution, thus enabling the system to handle high rates of data updates and simultaneous execution of triggers. The proposed method is significant in scenarios where the database systems are expected to handle large number of triggers.

Related Work and Limitations:
Most of the trigger systems operate on a ECA (Event-Condition-Action) model. In such systems condition-checking and actions are performed in the same transaction as the update event. Triggers that are expensive, especially in condition checking, can slow down the processing considerably in these systems. As an attempt to speed up the processing, work was done in indexing range predicates [Hans96b]. This approach did not scale to large number of triggers and used a large amount of memory. AI production systems were used to improve the scalability [Forg82, Mira87]. These methods assumed the number of rules was small enough to fit in the
main memory. Though attempts were made to parallelize these systems [Acha92], they did not address the issue of large number of rules. The paper analyzed here addresses the scalability of trigger systems through predicate indexing and the authors claim that this is the first work to do so.

**List the major contributions of the paper. Which do you think is most significant and why?**

1. The proposed architecture for trigger processing
2. Trigger condition testing algorithm.
3. Expression signatures and scalable predicate indexing using expression signatures.
4. Concurrent processing of triggers at various levels.

The identification of the common underlying structure of triggers is the key concept exploited in this paper. Viewed from this perspective, we feel that the most significant contribution is the concept of expression signatures and predicate indexing (Contribution (3)). Expression signatures represent the logical structure of a trigger condition. The constants that distinguish the individual triggers are separated from the structure. The paper also proposed an index structure on the expression signatures based on the conjuncts in their predicates. The concept of expression signatures enables the trigger processing system to exploit the common underlying structure of the triggers and allows for an efficient organization and storage of the trigger classes. Every strategy proposed in the paper towards improving the performance (and hence the scalability) while processing a large number of triggers is based on the expression signatures and the predicate index structure.

**What are the key concepts behind the approach in this paper? Provide simple explanation of those. Also provide a couple of simple exercises for the audience to check their understanding of the key concepts.**

1. **Architecture**

The trigger processor system consists of the following components
(i) TriggerMan Datablade
(ii) Data Sources which are local or remote tables or streams.
(iii) TriggerMan Client Applications.
(iv) A driver program that periodically perform trigger condition testing and action execution.
(v) A console that enables the user to interact with the system.
The system uses A-TREAT network for testing join conditions in triggers. Figure 2 shows an example trigger and the A-TREAT network that represents the trigger. The network has relations represented as nodes and selection conditions represented by special nodes called $\alpha$-nodes. To minimize storage usage, the nodes corresponding to selection conditions would store the predicate for the selection rather than the qualifying data itself, if a large number of tuples qualify for the selection and such nodes are called virtual $\alpha$-nodes. In the example shown in figure 2, a large number of tuples are assumed to satisfy the selectivity condition $\text{emp.sal} > 30000$, and hence is represented by a virtual node.

```
define rule SalesRule
if emp.sal > 30000
and emp.dno = dept.dno
and dept.name = "Sales"
and emp.jno = job.jno
and job.title="Clerk"
then action
```

2. Expression Signature

An expression signature for a predicate expression in the trigger statement would consist of the data source id that identifies the relation, an operation code and a generalized expression. The generalized expression represents the selection predicate in the conjunctive form, with the constants in the conditions replaced by placeholders. A generalized expression is represented by a syntax tree. All conditions that follow the same syntax tree are classified into the same equivalence class. For example, the conditions: a) on insert to emp when emp.salary > 50000 and b) on insert to emp when emp.salary > 80000 fall in the same equivalence class since they have the same syntax tree, though they differ in a constant. Figure 3 shows the syntax tree for these conditions.

(i) emp.salary > 50000
(ii) emp.salary > 80000

Figure 1: Architecture of the Trigger Processing System

Figure 2: Trigger Statement and its A-TREAT Network Representation

Figure 3: Syntax Tree for Conditions
3. **Predicate Index Structure**

The proposed system defines a main memory data structure called a predicate index. The most selective conjunct in a predicate is indexed and a predicate index structure is maintained which contains the predicate indexes for all tables and the expression signature lists. An expression signature list for each relation is maintained in the structure and this list has an entry for each unique expression signature used by one or more triggers on that relation.

To make the match between selection predicates and tokens more efficient, the predicate index structure is normalized. Each expression signature entry in the index structure is linked to a set of constants which would finally replace the placeholders in the tree. In the normalized version, each constant is linked to a trigger ID set which stores the identifiers of all triggers that use the same constant in the predicate.

![Figure 3: Expression Signature Tree](image)

**Figure 5: Predicate Index Structure**

4. **Concurrency**

The proposed system provides the facility to concurrently process triggers. It can be done at token-level, condition-level, rule-action level and data-level. It has been implemented at the token-level, and the authors plan to implement the other three.

**Key Concepts by Example:**

The following example explains the above mentioned key concepts used.

**Triggers**

<table>
<thead>
<tr>
<th>Trigger 1</th>
<th>Trigger 2</th>
<th>Trigger 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>create trigger T1</code></td>
<td><code>create trigger T2</code></td>
<td><code>create trigger T3</code></td>
</tr>
<tr>
<td><code>from</code> bank_account,</td>
<td><code>from</code> bank_account</td>
<td><code>from</code> bank_account</td>
</tr>
</tbody>
</table>
Expression Signatures

E1: bank_account.account.id = CONSTANT (used in T1, T2, T3)

E2: bank_account.balance < CONSTANT (used in T1, T2)

Predicate Index Structure for Triggers

Concurrency

Token-level concurrency can be implemented by evaluating the tokens (bank_account, checking_account) in parallel.

What is the validation methodology (e.g. case studies, statistical hypotheses testing, proofs, simulation) used in this paper? Describe the strengths and weaknesses of the methodology. Why did the authors choose this methodology?

The paper hints at the fact that the proposed method has been implemented on an INFORMIX server. The trigger processing system called “TriggerMan” is reported to be implemented. The underlying concepts are derived from techniques that have
already been proved in the literature. The effectiveness of these methods in the scenario of trigger processing has not been explicitly dealt with. Under the assumption that most of the triggers would follow similar structure (which the authors claim to be the case), the proposed method which hinges on the hypotheses would work effectively. The authors rely on the correctness of the underlying methods used to validate the system. This method fails to bring about the effectiveness of the system as a whole in various real life scenarios. The authors describe the design of a system based on the triggerman architecture in a paper cited as a reference [Hans97b]. Though this can be treated as validation, the authors indicate that they plan to use an optimized version of A-TREAT network [Hans97b]. The improvised version or its performance is not explicitly described in the paper. Since the paper’s contribution is in the aspect of scalability, it would be more convincing if the authors had evaluated the system in a scenario where triggers are generated in large volumes.

List the assumptions made by the authors. Critique an assumption that you believe is unreasonable. What is the impact of removing this unreasonable assumption on the solution proposed by the authors?

The paper assumes that though a large number of triggers would be created, it would be possible to classify these into a very small number of classes, with each class containing triggers with same predicate structure.

Here the authors identify a structure where the attributes in the predicates are the same and the values of the constants are different. Though there is a good chance that the entire set of triggers would fall into a small number of categories, there is a possibility of triggers being raised on values of different attributes when we consider the number of choices available in tables with a large number of attributes. Authors do not report any case study to prove that their assumption is valid in most real world scenarios. If this assumption proves to be invalid, the paper has no contribution since all the methods used to efficiently process the triggers is based solely on this assumption.

In this paper, the authors assume that the entire predicate index structure and the normalized version would fit into the main memory.

List the future work.

1) Develop ways to handle temporal trigger processing.
2) Extend the scalable trigger processing to include aggregates in trigger conditions.
3) Add extensibility to main-memory and disk-based structures that organize constants so that they can work effectively with new operators and data types.

If you were to rewrite this paper today, what would you preserve and what would be revised? Briefly justify.
Under the assumption that most triggers in a system follow the same structure, the paper proposes a scalable trigger system. Application of TREAT and predicate indexes in improving the scalability of trigger systems is commendable.

The validation of the technique needs to be addressed in the paper more explicitly. Though there are no questions regarding the correctness of the methods used, it would serve the audience better if there is a case study to validate the assumptions. Even in cases where the assumption is valid, experimental study to show the improvement of the proposed method over existing methods (or even a naive approach where no steps are taken towards improving scalability) would contribute a great deal towards the quality of the paper.