A Novel Event Rule Derivation for Processing Uncertain Data Events using Genetic Network Programming

M. Sivasankari M.C.A[1]

[1] *Mphil.Scholar, Department of Computer Science, KG College of Arts and Science,Coimbatore. Tamil nadu, India*

Abstract—Continuously growing size and number of databases in a variety of domains has boosted development of numerous data mining methods during the last decade. There is an increasing requirement to discover associations and relations among large and uncertain databases, which may be tackled by association rule mining. Two main challenges exist when designing a solution for event derivation under uncertainty. First, event derivation should scale under heavy loads of incoming events. Second, the associated probabilities must be correctly captured and represented. Current work proposes a solution to both problems by introducing a novel generic and formal mechanism and framework for managing event derivation under uncertainty. To solve this problem, the proposed system uses Genetic Network Programming (GNP) for event rule derivation. A method for association rule mining from large, heterogeneous and uncertain databases is proposed using an evolutionary method named Genetic Network Programming (GNP). Some other association rule mining methods cannot handle uncertain data directly, they are inapplicable or computational inefficient under such a model. GNP utilizes direct graph structure and is able to extract rules without generating frequent item sets to improve mining efficiency.

Keywords— Complex event processing, rule-based reasoning with uncertain information, Genetic Network Programming

I. INTRODUCTION

In recent years, there has been a growing need for event driven (or active) systems, i.e., systems that react automatically to events. The earliest event-driven systems in the database realm impacted both industry (triggers) and academia (view materialization). New applications in areas such as Business Process Management (BPM), sensor networks, security applications (e.g., bio hazards and computer security), engineering applications (e.g., forecasting networked resources availability); and scientific applications (e.g., utilization of grid resources) all require sophisticated mechanisms to manage and react to events.

Some events are generated externally and deliver data across distributed systems, while other events and their related data need to be derived by the system itself, based on other events and some derivation mechanism. In many cases, such derivation is carried out based on a set of rules. Carrying out such event derivation is hampered by the gap between the actual occurrences of events, to which the system must respond, and the ability of event-driven systems to accurately generate events. This gap results in uncertainty and may be attributed to unreliable event sources (e.g., an inaccurate sensor reading or an unreliable Web service), an unreliable network (e.g., packet loss at routers), or the inability to determine with certainty whether a phenomenon has actually occurred given the available information sources. Therefore, a clear trade-off exists between deriving events with certainty, using full and complete information, and the need to provide a quick notification of newly revealed events. Both responding to a threat without sufficient evidence and waiting too long to respond may have undesirable consequences. One way of managing the gap between actual events and event notifications is to explicitly handle uncertainty. This could be done by modeling events uncertainty as a probability associated with each event, whether such events are generated externally or derived. However, a major challenge in such explicit management of events' uncertainty is that rule-based systems need to process multiple rules with multiple event sources. Correctly calculating event probabilities while taking into account various types of uncertainty is not trivial. Clearly, correct quantification of the probability of derived events serves as an important tool for decision making. Event generation under uncertainty should therefore be accompanied with an appropriate mechanism for probability computation. Another major challenge, related to the need to enable timely response to events, is efficient event derivation, sometimes under a heavy load of incoming events from various sources. Event derivation should also scale for a large number of (possibly interrelated) rules and complex rules that involve several sources of evidence. Clearly, a brute force solution in which the arrival of a new event is evaluated against all possible rules does not scale, as it may involve an exponential number of evaluations (depending on the amount of dependency among rules). To illustrate this point, consider a natural way of interpreting uncertain events by assigning an explicit probability to each possible subset of events. Clearly, such explicit representation is practically infeasible. Therefore, our main goal is to provide an

efficient and accurate (yet generic) mechanism for reasoning with uncertain events.

II. RELATED WORK

Complex event processing is supported by systems from various domains. These include ODE [1], Snoop [2], and others [3] for active databases and the Situation Manager Rule Language [4], a general purpose event language.

The majority of existing models do not support event uncertainty.

As a result, solutions adopted in the active database literature, such as the Rete network algorithm [5] presented by C.L Forgy fail to provide an adequate solution to the problem, since they cannot estimate probabilities.

In [6] Shmueli and Fienberg demonstrated that over-the counter and pharmacy medication sales, calls to nurse hotlines, school absence records, and complaints of individuals entering hospital emergency departments can all serve as indicators of disease outbreak. These measures can be collected by querying local stores, physician databases, or data warehouses. The presentation of the data as a time series illustrates the problem of uncertainty in event information.

In [7] J. Pearl presented a common mechanism for handling uncertainty reasoning is a Bayesian network a method for graphically representing a probability space, using probabilistic independencies to enable a relatively sparse representation of the probability space. Qualitative knowledge of variable interrelationships) is represented graphically, while quantitative knowledge of specific probabilities is represented as Conditional Probability Tables (CPTs). The network is, in most applications, manually constructed for the problem at hand.

In [8] Breese et.al presented Knowledge Based Model Construction (KBMC) paradigm which separates uncertain knowledge representation from inference, which is usually carried out by transforming the knowledge into a Bayesian network that can model knowledge at the propositional logic level knowledge.

In [9] Kersting and Readt follow the KBMC paradigm. In this work, the quantitative knowledge, as well as the quantitative deterministic knowledge, is represented as a set of Horn clauses and the qualitative probabilistic knowledge is captured as a set of CPTs.

The work done in [10] by Cowie.et.al used for processing complex events in specific domains tailor probabilistic models or direct statistical models (e.g., regression) to the application. No general framework was defined there to derive uncertain events, a gap we strive to fill in this work. .

III. PROPOSED WORK

Priority rules are referred as Queue-based. Instead of guaranteeing optimal solution, these techniques aim to find reasonable solutions in a relatively short time used for solving scheduling problem based on backfilling techniques. The purpose of backfilling is to improve system utilization of scheduler that used First Come First Serve (FCFS). Although FCFS is a simple policy and have been widely used, it suffers from the low system utilization. This happens because there is a gap between two jobs that make the resource idle. Backfilling improves resource utilization by allowing small job to fill in those gaps. Job which is lower in the queue is moved to the idle machines without delaying the execution of the job at the top of the queue.

Biggest Hole (BH) strategy is to make a large process faster. Large process is the process that usually involved many calculations and huge data that consumes a lot of time. While traditional backfilling intends to fill the holes with smaller jobs and Earliest Gap Earliest Deadline First (EG-EDF) fill the first holes with possible jobs that can fix in, Biggest Hole search for the biggest holes in the queue and match them with any jobs that can fix in. In the case of EG-EDF, if the hole or gap does not match the jobs or too small for the job, the system has to search again and the possibility of the candidate gap to be lost are higher because as time pass by, the candidate gap can be shrinks smaller or even lost from the system. This is not happening with BH technique. BH-PR carries out the same procedure as EG-PR algorithms. Instead of accepting the earliest gap in the queue, BH-PR search for the biggest gap in schedule at that particular time. This is carried out by sorting the gap decreasingly based on the gap size.

The system architecture of the proposed framework is shown in fig 1:

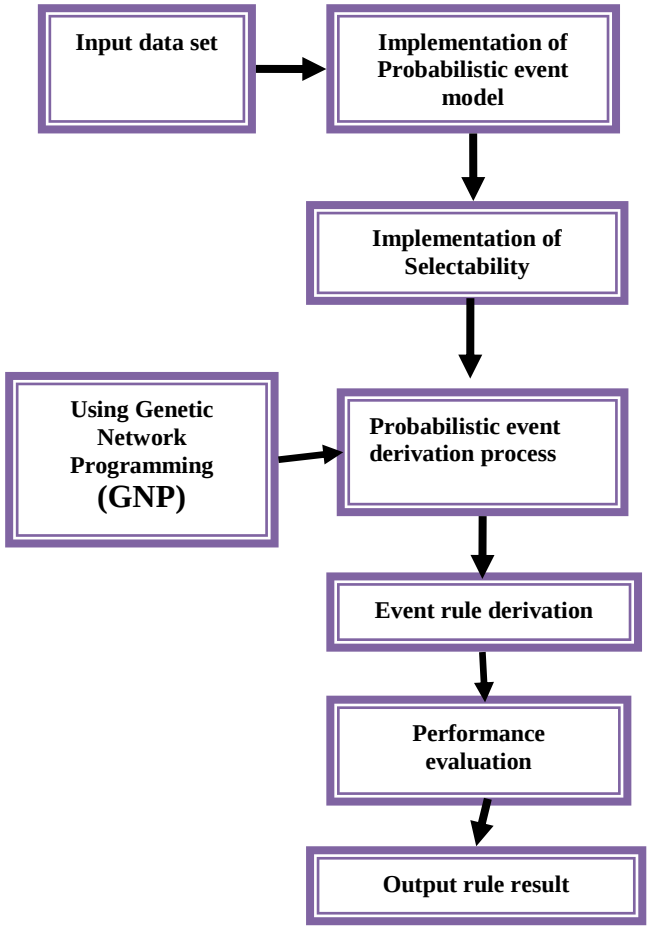


Figure 1: Proposed system achitecture

IV. METHODOLOGY

A. *IMPLEMENTATION OF PROBABILISTIC EVENT MODEL*

An event is an actual occurrence or happening that is significant (falls within a domain of discourse), and atomic (it either occurs or not). This definition, while limited, suits our specific needs. Data can be associated with an event occurrence. Some data types are common to all events (e.g., occurrence time), while others are specific (e.g., sales in an OtCCMS event). The data items associated with an event are termed attributes. Derived events in our model are inferred using rules. A rule defines how many new events (or EIDs) should be derived and helps calculate their attributes and probabilities. Intuitively, assume that the set of possible event histories prior to evaluating rule r is H. Moreover, assume that the actual event history is known to be some h 2 H.

A major novelty of our framework is in the support of the calculation of probabilities associated with derived events, at a given time point t. At time t, the set of possible derived events is determined by the explicit EIDs known at t (together with the defined rules). Therefore, since different sets of explicit EIDs may be available at different time points, a (possibly) different probability space needs to be defined for each time point separately.

B. *IMPLEMENTATION OF SELECTABILITY*

Selectability, as defined by function sr in a rule specification, plays an important role in event derivation, in both the deterministic and the uncertain settings. First, it defines which events are relevant to derivation according to rule r—an important semantic distinction. Just by analyzing the definition of sr it is clear to a human which events are defined as being relevant to derivation according to r, and which events are ignored in this derivation. Selectability significantly influences the performance of the inference algorithm.

As in the uncertain setting derivation is carried out on EIDs, algorithms are required to compute which EIDs, from a given system event history H, are selectable. Deciding whether an EID E is selectable by rule r may, by itself, incur significant computational effort. This is because, according to Definition 1, selectability depends on the possible event histories in which the event corresponding to E participates.

Algorithm 1. calculateSelectableEIDs(H,r)

1. selectableEIDs← Φ
2. esEIDs← Φ
3. for all E ϵ H
4. for all e ϵ E
5. if es_r (e) ={{e}}
6. esEIDs ←esEIDs ∪ E
7. end if
8. end for
9. end for
10. for all h ϵ esEIDs
11. for all e ∈ s_r(h)
12. E ←getCorrespondingEID(e)
13. selectableEIDs ← selectableEIDs ∪ E
14. end for
15. End for
16. Return selectableEIDs

To calculate the complexity of Algorithm 1, we will denote by n the number of EIDs representing the system event history H and by m the size of the state space of the largest EID. Using this notation, the complexity of lines 1-9 in Algorithm 1 is O(mn), polynomial in the number of EIDs and the state space size of the EIDs.

C. *PROBABILISTIC EVENT DERIVATION PROCESS*

In this module, we provide a high level description of an algorithm for uncertain derivation of events. In a nutshell, the proposed algorithm works as follows: Given a set of rules and a set of EIDs at time t, we automatically construct a Bayesian network that

correctly represents the probability space at t. Probabilities of new events are then computed using standard Bayesian network methods.
Two important properties that must be maintained in any algorithm for event derivation (both in the deterministic and uncertain setting), are determinism and termination. Determinism ensures that for the same set of explicit EIDs, the algorithm outputs the same set of derived EIDs. Termination ensures that the derivation algorithm terminates. To ensure determinism and termination, we enforce rule ordering such that 1) rules are triggered according to the defined order, 2) rule ordering is independent of event histories, and 3) an event may trigger a rule at most once.

The last requirement ensures termination. The second requirement ensures determinism. In our algorithm, the Bayesian network is constructed from explicit events and rule definitions that describe the desired probability space. With a new event arrival we traverse the rules according to the predefined ordering, and for each rule r, a Bayesian network segment (i.e., a segment with relevant nodes, edges, and CPTs) is created, satisfying the rule semantics, and utilizing probabilistic independencies. After carrying out these steps for all rules, the network is complete, and the probability space at time point t is calculated based on the probability space described by the entire network. It was shown in that the algorithm is exponential in the number of events and polynomial in the number of possible worlds.

The sampling algorithm described in the previous section generates a Bayesian network from which the exact probability of each event can be computed. Given an existing Bayesian network, it is also efficiently possible to approximate the probability of an event occurrence using a sampling algorithm (several such algorithms are known), as follows: Given a Bayesian network with nodes $E_1,\ldots E_n$.
we calculate an approximation for the probability that E_i={occurred} by first generating m ndependent samples using a Bayesian network sampling algorithm. Then,
$Pr(E_i=\{occurred\})$ is approximated by $\frac{\#E_i=\{occurred\}}{M}$, where $\#E_i$={occurred} the number of samples in which Ei has received the value is occurred.

Algorithm 2. RuleSamp, triggered by a new event arrival

1. h← Φ
2. for all E ϵ H_0
3. e ←probSampling (E)
4. h← h∪ e
5. End for
6. Order ← $obtainTriggeringOrder$
7. While order ≠ Φ
8. r← deleteNextRule(order)
9. h'← Φ
10. selEvents ← S_R(h)
11. if pr(selEvents) = true
12. assocT uples ← a_r(sel Events)
13. for all tuple ∈ assocT uples
14. { s1,….sn}← m_r(tuple)
15. prob← pr_r(tuple, m_r(tuple))
16. probsamp← $sampleBernoulli(prob)$
17. if probsamp=1
18. e←{occurred,s1….sn}
19. Else
20. e←{notOccurred}
21. end if
22. h←h'∪{e}
23. end for
24. end if
25. h←h∪h'
26. end while
27. return h

D. EVENT RULE DERIVATION USING GENETIC NETWORK PROGRAMMING (GNP)

In this proposed system, Genetic Network Programming (GNP) is used for event rule derivation. Genetic Network Programming is one of the evolutionary optimization algorithms, which evolves directed graph structures as solutions instead of strings (Genetic Algorithms) or trees. The main aim of developing GNP was to deal with dynamic environments efficiently by using the higher expression ability of graph structures.

The basic structure of GNP is shown in Fig. The graph structure is composed of three types of nodes that are connected on a network structure: a start node, judgment nodes (diamonds), and processing nodes (circles). Judgment nodes are the set of J1, J2, . . . , Jp, which work as *if- then* conditional decision functions and they return judgment results for assigned inputs and determine the next node to be executed. Processing nodes are the set of P1, P2, . . . , Pq, which work as action/processing functions. The start node determines the first node to be executed. The nodes transition begins from the start node; however there are no terminal nodes. After the start node is executed, the next node is determined according to the node's connections and judgment results.

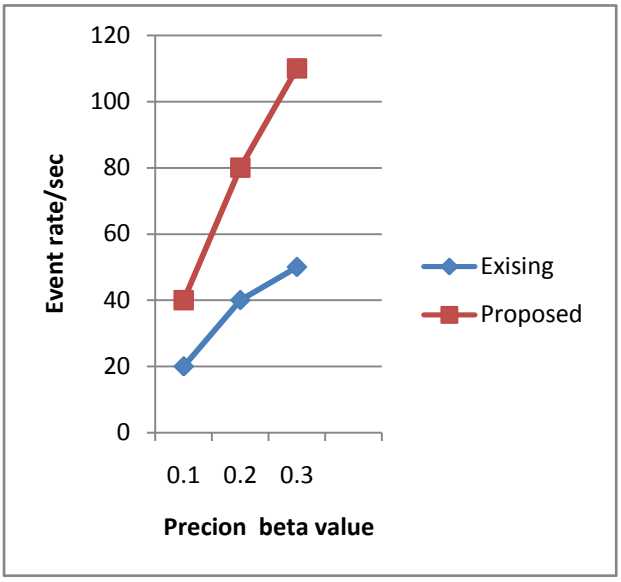
.

V. EXPERIMENTAL RESULTS

The proposed method can be evaluated based on number of events for precision analysis. The effect on performance is observed by drawing the log (base 10) of the actual number of events processed per second in various cases, where the difference between these cases are the percentage of events relevant to event derivation.

The graph shows the performance difference both in the deterministic case and proposed method as follows**.**

Figure 2: Event rate Comparision



The above graph in figure shows that the precision analysis for event rate. In graph Event rate in seconds are represented in Y-axis and precision beta value is represented in X-axis. Beta value represents the confidence width of values, 0.1, 0.2, and 0.3 respectively. Thus the proposed work gives higher event rate per second for increasing beta value when compare with the existing system of work.

.

VI. CONCLUSION

Thus two main challenges exist when designing a solution for event derivation under uncertainty. First, event derivation should scale under heavy loads of incoming events. Second, the associated probabilities must be correctly captured and represented. We present a solution to both problems by introducing a novel generic and formal mechanism and framework for managing event derivation under uncertainty. To solve this problem, in this system, Genetic Network Programming (GNP) is used for event rule derivation. A method for association rule mining from large, heterogeneous and uncertain databases is proposed using an evolutionary method named Genetic Network Programming (GNP). Some other association rule mining methods cannot handle uncertain data directly, they are inapplicable or computational inefficient under such a model. GNP uses direct graph structure and is able to extract rules without generating frequent item sets to improve mining efficiency.

REFERENCES

[1] H.N. Gehani, H.N. Jagadish, and O. Shmueli, “Composite Event Specification in Active Databases: Model and Implementation,” Proc. 18th Int’l Conf. Very Large Data Bases (VLDB), pp. 23-27, 1992.

[2] S. Chakravarthy and D. Mishra, “Snoop: An Expressive Event Specification Language for Active Databases,” Data and Knowledge Eng., vol. 14, no. 1, pp. 1-26, 1994.

[3] N.W. Paton, Active Rules in Database Systems. Springer, 1999.

[4] Adi and O. Etzion, “Amit—The Situation Manager,” Int’l J. Very Large Data Bases, vol. 13, no. 5, pp. 177-203, 2004.

[5] C.L. Forgy, “Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem,” Artificial Intelligence, vol. 19, pp. 17-37, 1982.

[6] G. Shmueli and S. Fienberg, “Current and Potential Statistical Methods for Monitoring Multiple Data Streams for Biosurveillance,” Statistical Methods in Counterterrorism, pp. 109-140, Springer Verlag, 2006.

[7] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, 1988

[8] J.S. Breese, R.P. Goldman, and M.P. Wellman, “Introduction to the Special Section on Knowledge-Based Construction of Probabilistic and Decision Models,” IEEE Trans. Systems, Man and Cybernetics, vol. 24, no. 11, pp. 1577-1579, Nov. 1994

[9] K. Kersting and L. De Readt, “Bayesian Logic Programming,” An Introduction to Statistical Relational Learning, pp. 291-322, MIT Press, 2007

[10] J. Cowie, A.T. Ogielski, B. Premore, and Y. Yuanb, “Internet Worms and Global Routing Instabilities,” Proc. SPIE, vol. 125, p. 4868, 2002