ACTIVE MONITORING USING REAL-TIME METRIC LINEAR TEMPORAL LOGIC SPECIFICATIONS

Gabor Simko and Janos Sztipanovits
Institute for Software Integrated Systems, Vanderbilt University, 1025, 16th Ave S, Nashville, U.S.A.

Keywords: Real-time monitoring, Temporal logic, Active monitoring, Policy monitoring, Control design.

Abstract: Monitoring temporal relationships among events in event streams has wide scale applicability in health information systems. From detecting violations of privacy policies in message sequences to diagnosing conditions in physiological data streams real-time event monitoring of temporal invariants is becoming an important tool for system design. We developed an Active Real-Time Event Monitoring and Integration System (ARTEMIS) capable of integrating event streams and monitoring the existence of temporal invariants among events expressed in a safety fragment of metric first-order temporal logic (MFOTL). The paper discusses the mathematical foundations of the monitor, and demonstrates the application concepts in a physiological alarm generator and clinical information workflow system.

1 INTRODUCTION

The concept of policies is a widely used abstraction in health information system design. Policies may be defined with different purposes. For example, privacy policies (Bartha et al., 2006) express restrictions on information flows among actors of a care delivery environment. Alert policies define the rules for signalling alerts in clinical environments. Treatment policies capture the decision rules for applying and ordering treatment activities. These policies share some common characteristics, namely, all of them can be modelled by formal logic and most of the time they contain temporal relationships. Huge difference is found however in time scales (from seconds to years), and whether they are only passive monitors or active integrators of event streams. Policies defined for passive monitoring express logical and temporal invariants over event streams. The passive monitors observe the event sequences and indicate if the invariants are violated. However, most treatment policies and several alert policies need active participation, for generating new events and integrating policy groups via the generated events (Figure 1). This is necessary for requesting actions (e.g. approvals in privacy policies or tests in treatment policies) that may be time and resource consuming and must be scheduled only by demand. The request events $\{r_1, \dots r_m\}$ and the outcome of the requested activities $\{t_1, \dots t_m\}$ may be used for integrating other policies in the monitoring process. The interplay between the different policies leads to the concept of Active Real-Time Event Monitoring and Integration System, (ARTEMIS) which monitors different events, displays or logs policy violations and starts up activities, which may also affect other policies.

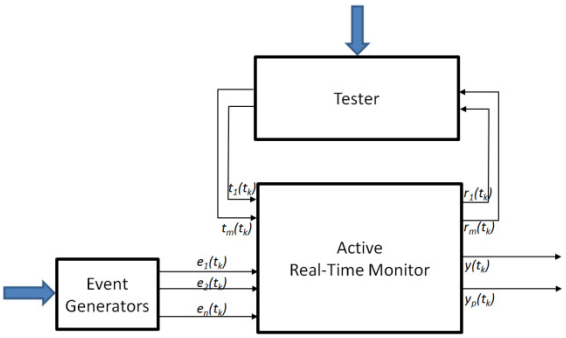

Figure 1: Schema of Active Real-Time Monitoring System: large arrows denote the access points to external resources, small arrows represent data flow. Monitor can activate Tester Unit on demand, and the feedback mechanism is provided to the Monitor.

Policies may be modelled with two different approaches, rule-based and statistical based. Due to space limitations we do not discuss statistical based methods here. Temporal logic is often used to

Simko G. and Sztipanovits J..
ACTIVE MONITORING USING REAL-TIME METRIC LINEAR TEMPORAL LOGIC SPECIFICATIONS.
DOI: 10.5220/0003768703700373
In *Proceedings of the International Conference on Health Informatics* (HEALTHINF-2012), pages 370-373
ISBN: 978-989-8425-88-1

describe rule-based policies (Basin, Klaedtke, and Muller, 2010) and to monitor events in a rule-based fashion. Our choice based on algorithmic considerations fall on a subset of temporal logic called Metric First Order Temporal Logic (MFOTL) (Basin et al., 2008). MFOTL can be used to specify a broad set of complex temporal constraints, while real-time operation and reasonable computation complexity is achievable. We built our monitor based on the concepts defined in their work, but over an extended $MFOTL_I$ language and using a significantly different monitoring algorithm.

The structure of the paper is the following: in Section 2 we introduce $MFOTL_I$ language and $MFOTL_I$ monitoring. Section 3 details the concept of ARTEMIS, and in section 4 we show a clinical workflow example. Finally, in section 5 we give the conclusions.

2 BACKGROUND

2.1 Syntax of $MFOTL_I$ Language

We define the logic $MFOTL_I$ as a superset of MFOTL (Basin et al., 2008), such that point and interval based semantics are both supported. $MFOTL_I$ may also be considered as the union of metric temporal logic (MTL_N) (Alur and Henzinger, 1991); (Koymans, 1990) and metric interval temporal logic ($MITL_{[a,b]}$) (Nickovic and Maler, 2007); (Alur et al., 1991) extended with predicates and quantification. The past temporal modalities are interpreted on possibly infinite intervals, while the future temporal modalities are bounded. Description of the $MFOTL_I$ language is based on the work of Basin et al., (2008) and Nickovic and Maler, (2007).

Formulae of $MFOTL_I$ are inductively defined in Backus-Naur Form by the following grammar:

$$p := t \mid \neg\, p \mid p_1 \wedge p_2 \mid \exists x.p \mid p_1 U_I p_2 \mid p_1 S_I p_2,$$

where I is a possibly singular time interval and t is a basic term, i.e. a function compared to a constant value, or a boolean predicate value. The operators represent the standard negation, conjunction, existence, until and since operators, respectively.

Based on the basic formulae we can express other standard logic operators and constants, such as true ($\top$), false (F), disjunction ($p_1 \vee p_2$) and universality ($\forall x.p$). Also, we can express release and trigger operators, eventually and always operators, and their past versions, once and historically:

$$p_1 R_I p_2 := \neg(\neg p_1 U_I \neg p_2)$$

$$p_1 T_I p_2 := \neg(\neg p_1 S_I \neg p_2)$$

$$\Diamond_I p := \top\, U_I\, p \quad \blacklozenge_I p := \top\, S_I\, p$$

$$\Box_I p := \mathrm{F}\, R_I\, p \quad \blacksquare_I p := \mathrm{F}\, T_I\, p$$

Here we illustrate the $MFOTL_I$ language with a frequently arising example: the policy declares that a patient p had to give their consent in the last 8 days to disclose their lab results prior to the disclosure taking place. The purpose of the monitor is to detect if illegal disclosure happened. The policy is easily expressed using $MFOTL_I$:

$$\mathrm{disclosed_t(x)} \rightarrow \blacklozenge_{[8\mathrm{days},0]} \mathrm{consent_t(x)}$$

If this expression is not satisfied at any time t with respect to patient x, the policy was violated for that patient.

2.2 $MFOTL_I$ Monitoring Algorithm

The main idea behind online monitoring is to incrementally build an inner representation of previous states without storing all the unnecessary details. A possible solution is to introduce auxiliary relations describing these past states. Satisfiability in the current state is answered by evaluating only these additional relations, i.e. answering the satisfiability of a first-order logic expression. Our monitoring algorithm works by transparently building and evaluating these auxiliary relations.

3 ARTEMIS ARCHITECTURE

ARTEMIS systems are built from three different kinds of components (Figure 1), sources, testers, and monitor. Sources are independent event generators (e.g. measurement devices or audit systems), which set up relations and functions used by the $MFOTL_I$ monitor. The sources send all the necessary information to the monitor, which automatically extracts and stores the relevant data. The monitor continuously checks whether the policies are satisfied and on satisfaction the appropriate testers (e.g. treatment procedures, lab tests) are activated. Finally, the results of testers are fed back to the monitor, which may lead to other coupled actions.

The controller actions are declared in the form of Horn clauses ($\mathrm{head(x)} \leftarrow \mathrm{body(x)}$), where the head is the action and body is the conditions leading to the action. We can define any number of such expressions as long as the body of the Horn clause is temporal sub-formula domain independent (Basin et al., 2008).

The expressions are ordered which defines the

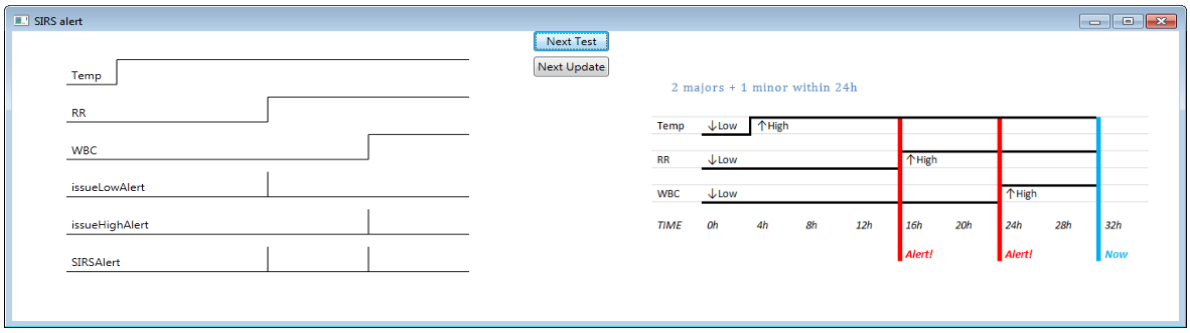

Figure 2: Snapshot from our test application showing the relations of SIRS alert. Left side shows the relations of ARTEMIS, right side shows the ground-truth data (signal high represents true, signal low represents false).

order of evaluation. Expressions may refer to each other: any expression may use the past values of any other expression including themselves or the current results of any expressions defined earlier in the order.

ARTEMIS shows several advantages over systems built with low level languages (Table 1). The fact that policies are well represented in formal logic leads to a compact, readable and easily maintainable policy code in ARTEMIS. The optimized monitoring algorithm results in high performance and optimal resource management. Furthermore, extensibility is easily achieved by using formal logic conjunction and disjunction operators without touching any of the previously written code.

Table 1: Advantages of ARTEMIS.

	ARTEMIS with MFOTL monitoring	Traditional system built with a low level language
Policy code complexity	Very compact	Highly complex
Performance	Automatically high, optimal	High, if optimized
Extensibility	Easily extensible	Extensible, but complicated
Maintenance	Easy maintenance	Cumbersome maintenance

Comparison to rule-based workflow management systems like Drools does not show this great difference. The main advantage of ARTEMIS is its MFOTL engine which can evaluate temporal logic with complex temporal expression. Although Drools is very efficient to describe simple temporal policies, it cannot handle compound temporal operators, which may seriously limit its applicability in some scenarios (e.g. signal processing).

4 EVALUATION

We evaluated ARTEMIS using the initial phases of the sepsis alert and treatment protocol (Figure 2). By monitoring several physiological data of patients we could express the Systemic Inflammatory Response Syndrome (SIRS) alert system (Shapiro et al., 2006). After SIRS alert was issued, ARTEMIS had to request the approval of a doctor to begin the sepsis treatment protocol. In case the approval arrived, a lab test request was issued to analyse additional conditions. Only after the receipt of approval and lab tests could the sepsis treatment start (Dellinger et al., 2008).

The SIRS alert protocol (Shapiro et al., 2006) defines validity ranges for physiological data. In case the measured function is outside the validity range, the measured data is abnormal. Abnormal body temperature and white-blood cell count are major criteria, abnormal respiration rate and heart rate are minor criteria. The protocol defines two kinds of alerts: high priority alerts are issued if two major criteria were met in the last 24 hours; low priority alerts are issued if at least one major and one minor criterion were met in the last 24 hours, and no alert were issued in the last 24 hours.

Our system contained four measured functions interpreted on patients: *temp* (temperature), *wbc* (white-blood cell count), *rr* (respiratory rate) and *hr* (heart rate). Using these functions we could express the policy as seen in Table 2. The derived abnormal functions were satisfied for a patient x, if their measurement was abnormal (i.e. out of normal range). *majorCriteria* was true, if at least one major criterion held, *minorCriteria* was true if at least on minor criterion held. Based on these criteria we could define *highPriority* and *lowPriority*, which were satisfied when the high priority or low priority alert requirements (see above) were met

Table 2: Sepsis treatment policy expressed in ARTEMIS.

$abnormalWbc(\mathrm{x})$	$\leftarrow$	$\mathrm{wbc(x)} < 4000 \mid wbc(x) > 20000$
$abnormalTemp(\mathrm{x})$	$\leftarrow$	$temp(\mathrm{x}) < 96.8 \mid temp(\mathrm{x}) > 100.4$
$abnormalRR(\mathrm{x})$	$\leftarrow$	$rr(\mathrm{x}) > 20$
$abnormalHR(\mathrm{x})$	$\leftarrow$	$hr(\mathrm{x}) > 90$
$majorCriteria(\mathrm{x})$	$\leftarrow$	$abnormalWbc(\mathrm{x}) \vee abnormalTemp(\mathrm{x})$
$minorCriteria(\mathrm{x})$	$\leftarrow$	$abnormalRR(\mathrm{x}) \vee abnormalHR(\mathrm{x})$
$highPriority(\mathrm{x})$	$\leftarrow$	$\blacklozenge_{[24\mathrm{hrs},0]} abnormalWbc(\mathrm{x}) \wedge \blacklozenge_{[24\mathrm{hrs},0]} abnormalTemp(\mathrm{x})$
$lowPriority(\mathrm{x})$	$\leftarrow$	$\blacklozenge_{[24\mathrm{hrs},0]} majorCriteria(\mathrm{x}) \wedge \blacklozenge_{[24\mathrm{hrs},0]} minorCriteria(\mathrm{x})$
$issueHighAlert(\mathrm{x})$	$\leftarrow$	$highPriority(\mathrm{x})$
$issueLowAlert(\mathrm{x})$	$\leftarrow$	$lowPriority(\mathrm{x}) \wedge \neg \blacklozenge_{[24\mathrm{hrs},0)} \big(issueLowAlert(\mathrm{x}) \vee issueHighAlert(\mathrm{x})\big)$
$SIRSAlert(\mathrm{x})$	$\leftarrow$	$\big(issueHighAlert(\mathrm{x}) \vee issueLowAlert(\mathrm{x})\big) \wedge \neg \blacklozenge_{[24\mathrm{hrs},0)} SIRSAlert(x)$
$Sepsis\_lab(\mathrm{x})$	$\leftarrow$	$approval(\mathrm{x})$
$Sepsis\_treatment(\mathrm{x})$	$\leftarrow$	$\big(\blacklozenge_{[24\mathrm{hrs},0]} approval(\mathrm{x})\big) \wedge labtests\_done(\mathrm{x})$

issueHighAlert and *issueLowAlert* signalled, when a high priority or low priority alert had to be sent out for patient x. For any patient x, when we reached *SIRSalert*, we initialized the Verification tester (once in every 24 hours, because the test takes significant amount of time; also note that this is not part of the original protocol, we used it for demonstration purposes only) for that patient. Relation *approval* signalled the results of the verification. If approval was received, the patient was sent to lab tests by issuing *Sepsis_lab*. On the arrival of lab tests, relation *labtests_done* was updated. If lab tests were done within 24 hours after the approval of sepsis treatment, we could start the sepsis treatment action signalled by *Sepsis_treatment*.

Even though the performance of the algorithm heavily depends on several factors, we simulated SIRS alert with 100000 distinct events affecting 100 patients leading to 25074 issued SIRS alert to demonstrate the order of magnitude. The average performance was 0.21ms / event.

5 CONCLUSIONS

We showed the concept of Active Real-Time Event Monitoring and Integration System (ARTEMIS) which is an extension of traditional real-time monitoring systems with active participation from the part of the monitor. As a workflow management system ARTEMIS competes with systems like Drools, but supports a broader and more expressive set of temporal expressions which might show immediate advantages in signal processing scenarios.

REFERENCES

Barth, A., Datta, A., Mitchell, J. C., Nissenbaum, H., 2006. Privacy and contextual integrity: Framework and applications. *SP'06*, pp.184-198.

Basin, D., Klaedtke, F., Muller, S., 2010. Monitoring security policies with metric first-order temporal logic. *SACMAT*, pp.23-34.

Basin, D., Klaedtke, F., Muller, S., Pfitzmann, B., 2008. Runtime monitoring of metric first-order temporal properties. *FSTTCS*, 2, pp.49-60.

Chomicki, J., 1995. Efficient checking of temporal integrity constraints using bounded history encoding. *TODS*, 20(2), pp.149-186.

Alur, R., Henzinger, T., 1991. Logics and models of real time: A survey. *REX Workshop on Real Time: Theory in Practice*, pp.74-106.

Koymans, R., 1990. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2, pp.255-299.

Nickovic, D., Maler, O., 2007. AMT: A property-based monitoring tool for analog systems. *FORMATS*, 4763, pp. 304-319.

Alur, R., Feder, T., Henzinger, T. A., 1991. The benefits of relaxing punctuality. *Tenth Annual Symposium on Principles of Distributed Computing*, pp.139-152.

Shapiro, N. I., Howell, M. D., Talmor, D. et al., 2006. Implementation and outcomes of the Multiple Urgent Sepsis Therapies (MUST) protocol. *Critical Care Medicine*, 34, pp.1025-1032.

Dellinger, R., Levy, M., Carlet, J. et al., 2008. Surviving Sepsis Campaign: International guidelines for management of severe sepsis and septic shock. *Intensive Care Medicine*, 34, pp.17-60.