Real Time Control
by means of a Knowledge Based System specified in Multi-valued Temporal Logic

Gonçal Escalada-Imaz
Artificial Intelligence Research Institute (IIIA)
Scientific Research Spanish Council (CSIC)
Campus UAB, s/n
08193 Bellaterra, Barcelona (Spain)
e-mail:gonzalo@iiia.csic.es

Abstract. The control of systems in different areas of the real world such us Chemistry, Medicine, Robotics, etc. has been tackled for decades with approaches developed in the classical Control Systems field. In this paper, we will propose a Real Time controller relying on a Knowledge Rule Based System with a Temporal Multi-valued Language. The proposed controller deals with the typical features of a Real Time framework such as imprecision produced by the sensors, smart and continue changes of the physical variables and it sends the required control signals in bounded time. This is a major restriction that must be fulfilled by a real time controller. Also, we will describe a methodology in order to validate the Rule Base System and we will analyse the complexity of the involved algorithms.

Keywords: Automated Reasoning, Computational Complexity, Search, Theorem Proving.

1 Introduction

The aim of a controller is to convey a physical process to a certain state following a criteria imposed from outer, e.g. by an expert, and whose principle fulfills some constraints required in specific applications. Generally, these constraints include the minimisation of the needed resources and the warranty that the demanded state must be reached in due short time. The govern of the process state must be done during a certain interval.

The more the physic process to be controlled is complex the more difficult is to design its model using developed grounds inside the classical material about Control Systems.

The interesting features shown by the first Expert Systems together with the mature theoretical basis existing currently to model and process uncertain and temporal information, were the original technical fact for envisaging new applications by a part of the research community in Artificial Intelligence inside the field of Control Systems. Currently, the performances accomplished by Fuzzy Control techniques are widely recognised today.

The mentioned new applications have in common their strong time restrictions in supplying actions in real time. In other words, to the existing difficulties of uncertain and temporal knowledge representation, a step ahead begun to be considered to developed Artificial Intelligence techniques to cope with time restrictions too. Thus, in the eighties, around fifteen years after the development of the first Expert Systems, new tools for real time control relying on Knowledge Based Systems (KBRS) emerged.

These tools posses the well-known advantages inherent to the Knowledge Based Systems: only explicit knowledge must be supplied, control processing steps are opaque to the user, knowledge granularity, facility to modify the knowledge base in subsequent information owing to available information, use of validation KBRS tools, etc.

Thus, new theoretical material and schemes were developed progressively till such Artificial Intelligence techniques turned out to be an important alternative in order to perform Real Time control of complex processes.

One of the challenge of these new techniques is to represent and to process inexact information, more specifically, imprecise, fuzzy, incomplete and noisy information. Thus, several languages have been proposed and deeply analysed to model such types of inexact information. Here we propose a Multi-valued Logic that, as we will argue later, bears imprecise and incomplete information both being present in Automatic Control Systems. Indeed, imprecision is inevitably introduced by sensors while incomplete information can be due to the bad functioning of some hardware or software elements in an input chain of a source information. Currently, many theoretical results on Multi-valued Logic have been found and real applications have been developed [3].

Another aspect in the representation and process of real time signals is their continuous changes. In order to deal with this temporal dimension, we extend the multi-valued language to a Temporal Multi-valued logic. Thus, the KBRS controler is modeled by a set of Multi-valued Temporal Propositional Rules.

The controler inputs are the outputs of the physic system and the control signals are established by a human expert or it is generated by a program computer taking into account the difference between the desired and the current state of the process. The control signals must be sent to the physical system within a bound time after the reception of some variation in an input signal meaning that the physic process has changed of state. This property is called Reactivity.

Reactivity is a capital feature in some real world applications. For example, if the trajectory of a airplane suffers a deviation due to some unpredictable disturbances produced by irregularities in the air, and this deviation is not corrected in limit short time then an accident can happen.

Thus, we shall analyse the involved algorithms an we shall show that the final complexity is in O(1).

The plan of the article is the following. Next we give one real world problem. Section 3 introduces the syntax and semantics of our Rule Based System. Section 4 discusses knowledge representation issues. Afterwards, we describe the Logical Calculi. Section 6 describes the main characteristics of our KBRS aspects and the associated interpreter. Next, the algorithms involved in the automatic validation step are presented. Finally, section 8 analyses the complexity issues.

2 Related Work

In [9, 10] is studied the connection between Real-Time processes and Artificial Intelligence techniques.

Reactivity is analysed for instance in [12, 4]. In [5, 11] necessary conditions that a KRBS interpreter must fulfill to possesses the reactivity has been stated.

In [8] some issues relating incomplete information, precision in the answer and time response are addressed.

The idea of anytime algorithm have turned out of remarkable relevance in designing algorithms supplying answers in any time of the processing [1].

Search space problems with bounded computational time and where the quality of the heuristic is linked to the relative approximation of the answer to a global optimal are studied in [7, 6].

3 A Real World Example

The described problem is part of a project aiming to implement a Real Time KBRS at the Clinical Hospital of Barcelona (Spain) to execute automatically the required reactive actions in a Pediatric Intensive Care Unit (PICU). The main difference between reactivity in PICU with respect to any other ICU is that temporal constraints are stronger due to the *fragile nature* of new borns. Physicians in a ICU must perform both classical diagnosis and reactive diagnosis. Indeed, one main feature of patients in ICU's is that they are in critical states which could evolve dangerously to unstable states with dramatical outcomes.

Before computer technologies were available, the evolution of critical patients was sensored by means of analogical material and all patient bio-signals were under continuous supervision of medical personal. However, this continuous supervision provoke a natural tireness of the medical personal implying an unavoidable lost of the pursuit of the dynamic of the monitored signals during some moments. Unfortunately, within these unsupervised intervals dangerous pathological tendencies can appear and thus, the emergence state is detected with some delay. This delay could difficult seriously the back of the patient critical state to a controled state out of danger.

Thus, to leave the monitoring, supervision, diagnostic and therapeutic steps to an automatic controler will avoid such severe drawbacks existing in manual operations.

The way of physician reason in a PICU share some aspects regarding the construction of the well known Expert Systems applied in medicine.

4 Syntax and Semantics

Let us start by the classical boolean logic to better grasp a straightforward extension called boolean multi-valued logic. This logic is then generalised to take into account the temporal dimension. In Real Time Knowledge Based Systems we deal with Horn Theories. A set of facts models a state and a set of implication rules allow to deduce implicit information associated to the states.

4.1 Syntax

Definition 4.1 Boolean Bi-valued Logic BBL. *It has two possible semantic values* $\{0,1\}$ *for each propositional variable. Classically, these literals are noted* p *and* $\neg p$ *respectively.*

Definition 4.2 Boolean Multi-valued Logic. *It is a straightforward extension of the BBL, the only change relies on the cardinality of the set N of truth values that an interpretation can assign to a proposition p. Thus, a literal in BML is noted S:p and its complemented literal N/S:p.*

Remark A large class of applications relies on this logic [3], but its suitability for Real Time Systems has not been considered yet.

Definition 4.3 Temporal Multi-valued Logic. *A literal is a triple* (T, S, p), *where S and p are as defined before and T is an interval [lb up]. If lb=up, the literal represents instantaneous information.*

Definition 4.4 Knowledge Based System. *A KRBS is composed by a set of fact and rules. Facts are TML literals. The antecedent of a rule is a conjunction of BML literals and its consequent is a TML literal.*

4.2 Semantics

Definition 4.5 . Interpretation *An interpretation is a map of the set of propositions to the couples of time intervals* S_T *and value intervals* S_S*:*

$$I : P \rightarrow S_T \text{ x } S_S$$

Definition 4.6 Literal Satisfiability. *An interpretation I satisfies a literal (T,S,p) if* $I(p) = (T_p, S_p)$ *and* $T_p \subseteq T$ *and* $S_p \subseteq S$.

Definition 4.7 Rule Satisfiability. *An interpretation satisfies a rule if it does not satisfies at least one literal antecedent or satisfies its literal consequent.*

Definition 4.8 Logical Consequence *A literal is a Logical consequence of a KBRS if all the interpretation that satisfies the KBRS satisfies also the literal.*

5 Representation Issues

Reasoning without considering imprecision and/or missing information could lead to erroneous decisions and hence provoke considerable pains. Indeed, the controller could lost the control of the process. We discuss briefly below the sources of these inexact information as well as some few concepts related with it.

The source of imprecision could be due to the analytical or digital sensors and the way experts reason in its expertise domain, this is due to the so called linguistic labels appearing in the statements they do to explain the relationship among physical variables.

5.1 Sensor Imprecision

This is a feature inherent to sensors. It indicates the minimal variation of the signal perceptible by a particular sensor. Namely, a variation smaller than the resolution sensor has no effects of the output of the sensor.

Unaccurate information. The existence of a Sensor Resolution entails that the output value of the sensor, which is possibly the input to the information process, is not the real input value. More precisely, if SR is the sensor resolution value, then its output value is related to the real physic value by:

$$val(physic) \in [val(sensor) - SR, val(sensor) + SR]$$

Digital Sensor. As we address here the control by means of a computer, the output sensor must be digital. If the computer processes data represented by at most n bits, then $SR = 1/2^n$.

5.2 Imprecision due to Linguistic Labels

Some inputs to the system can be provided by an expert. For example, in medicine the physician can furnish some informations relative to the clinical story of a patient, the dynamic of a certain illness, the relationship among particular parameters (cardiac rhythm, blood pressure, corporal temperature, etc.) and so on. Thus, an expert can provide his knowledge stating for instance "if a variable is between high and very high then ...". This is, firstly the expert *qualifies* the quantitative values of a variable mapping its real values to a limit set of the so-called Linguistic Labels

$$< low, high, very - high, \ldots >$$

and second, he refers to it by mentioning intervals, for instance [high very-high]. In such situation, one has:

$$LB(val(var)) \in [LB_1 \; LB_2]$$

where LB(val(var)) stands for the linguistic label associated to val(var) and LB_1 and LB_2 are two linguistic labels, the first one associated to smaller values than those associated to the second one.

Unaccurate information modeling. The effect of the Sensor resolution can be captured easely by a BML literal (T,S,p) doing S=[val(sensor)-SR, val(sensor)+SR].

5.3 Incomplete Information

The controler sequentially reads the input signals, makes a certain reasoning and afterwards sends the output signals. A sensor breakdown or a connection problem during a certain time can cause that a particular physic variable was not measured when the automatic controler needed it. Similarly, the human expert for some reason can omit or unknow certain information at one determined moment and then do not supply it to the automatic controler. These situations can be covered by a literal $(T, Dom(p), p)$ where Dom(p) is the whole domain of possible values of the unknown variable.

6 Logical Calculi

The Inferences rules are two: One corresponding to a direct extension of the Modus Ponens to our Logic and another refering to the intervals of values and time.

Intervals Rule (IR)

$$\frac{(T,S,p), T \supseteq T', S \supseteq S'}{(T',S',p)} \text{(IR)}$$

Temporal Multivalued Modus Ponens (TMMP)

We note $L = (T,S,p)$ and $I(L) = (T',S',p)$, with $T \supseteq T', T \supseteq T'$.

$$\frac{L_1, \ldots, L_k, I(L_1) \wedge \ldots \wedge I(L_k) \rightarrow (U,V,q)}{(U,V,q)} \text{(TMMP)}$$

Theorem 6.1 Soundness *Let us note* $\vdash_{\{IR,TMMP\}}$ *the logical calculi formed by the IR and TMMP rules.* $\vdash_{\{IR,TMMP\}}$ *is a sound calculi.*

Theorem 6.2 Completeness *If* $KBS \models L$ *then* $KBS \vdash_{\{IR,TMMP\}} L$

Both proofs are omitted in this article.

7 KRBS Design Issues

Notation We will distinguish between input Variables I, output variables O and state variables S.

In control Systems the dynamic of systems is mathematically well established by:

$$S(t) = f(I(t), S(t-1), S(t-2), \ldots, S(t-k)$$

$$O(t) = g(I(t), S(t))$$

The control in real time imposes that the computation of $g(I(t), S(t))$ may take at most O(1) time to fulfill the Reactivity property.

Each output control variable o is associated with a set of rules whose consequent is a literal of kind (.,.,o).

Each input signal i coming from a sensor is associated with literals appearing in rules antecedents of kind (.,.,i).

Other literals are defined by the expert who needs to declare intermediate deduction. The set of intermediate deduced variables are the consequent literals of a set of rules. The different values of these variables taken in different instants conform the state of the controler. Intuitively, this state is related to the diagnostic of the external process. Thus, according to the combination of these values, the control signals O are defined to maintain the system continuously under control.

In order to design of the Knowledge Rule Based System the expert must have clear in mind:

The domain of each input variable, the set of different diagnostic of the physical state or similarly, a proper classification of physic states and the action to be pursued in each diagnosed state. Experience with Expert Systems, Control Fuzzy and other KBRS technologies have proved that for many cognitive fields there exists a Rule Base System whose interpretation by an algorithm in an on-line process matches the suitable properties of a efficient control.

In this article, we are proposing a language and the required algorithms that enable the existence of such KRBS in order to have an adequate controler. As mentioned, KRBS have arisen since several decades as a serious alternative to control when the system to be controled is very difficult to be modeled.

We consider three types of rules:

- *Input rules*: all their antecedent literals are associated to input sensors.
- *Output rules*: its consequent literal is associated to a control output.
- *Deduction rules*: they are nor input nor output rules, i.e. the consequent represents deduced information used by the expert.

The KBRS is associated to a graph AND/OR as usual. Each node corresponds to a propositional variable. Each rule to an AND connector. The set of connectors whose literal consequent contains the same propositional variable defines an OR connector. The literals that do not have descendent connectors are input literals. The literals that do not have ascendent connector are associated to output control signals. Nodes having ascendent and descendent are intermediate deduced variables that corresponds to state variables S. As mentioned, they help the expert to make a diagnosis of a part of the physic system and also they provide a preliminary information required by the expert before stating with confidence the control signals.

Human experts do not associate to each infinite combination of the input variable a different diagnostic and hence a different control. What an expert does it is to qualify the values of variables with infinite or big number of possible values and classify the values in a finite set (between three and ten) of values. Each set is refered to by a linguistic label.

Intuitively, this is because the input values of a defined set, or more generally, to a part of a hyperspace of values, correspond to a same qualitative situation. Namely, although the state of the system is not exactly the same for two different values of the same region, the variation is not significant enough to change the qualitative diagnostic.

The values of the state variable are also qualified and then these values take a limit number of variables that usually goes from three

$$< Yes, unknow, Not >$$

to for example ten

$$< very - high, high, ..., very - low >$$

To each value of a qualified variable corresponds a virtual range of values in a numerical reference.

To take into account the variations of the signal with respect to time, a propositional variable v(t) is stored as an array:

$$t => v(t) : v(t-1) : \ldots : v(t-k)$$

To keep a record of these values in the next instant is enough to execute a simple shift of the set of consecutive memory cells and to add the new value v(t+1):

$$t+1 => v(t+1) : v(t) : \ldots : v(t-k+1)$$

The principle of the algorithm consists in propagating the value of the antecedents of a rule to the consequent. This propagation begins by the input signals and ends in the signal control accomplishing therefore a bottom-up strategy.

8 Validation

The control process performed by means of a Rule Based System, in addition to many other advantages, allows to check the final disegned Rule Base in an automatic way. In other words, some errors or omission committed by the expert can be detected automatically.

Here, we describe the steps to check two mean features of the final Rule Base in the two following sections.

For the sake of simplicity and better understanding, we make abstraction in the whole section of the temporal dimension. Also, Intervals $[\infty\ ub]$ are written $\leq ub$. We use the same notation for intervals defined by the lowest bound. Similarly, for semi-open intervals, for instance we will write $< ub$ for the open interval $[\infty\ ub[$.

8.1 Consistency

When the expert is designing the Rule Base, by several reasons, he can specify two different control strategies for the same situation.

Example 8.1 *Assume now that inside the Rule Base there exists the following two rules:*

$$([-\infty, 2], x_1) \wedge ([-\infty, 2], x_2) \rightarrow ([High, High], o)$$

$$([-\infty, 2], x_1) \wedge ([2, 2], x_1) \rightarrow ([Low, Low], o)$$

One can verify that the control signal output is bud defined for the case where $x_1 = 2$ and $x_2 = 2$ because the first rule sends the value o=High meanwhile the second one sends the value o=Low.

A formal definition is given below.

Definition 8.1 *A RBS is inconsistent iff there exist at least one variable x and at least a set of facts F such that $F \wedge RBS \models (v_1, x)$ and $F \wedge RBS \models (v_2, x)$ and $v_1 \neq v_2$.*

Thus, to check whether the Rule Base is consistent it is enough to test that for each variable x (state or output) the antecedents of the rules with a same consequent value (val, x) are never fired in a same facts configuration.

Theorem 8.1 *If for each couple of rules r and r'*

$$r = (L_1 \wedge L_2 \wedge \ldots \wedge L_k \rightarrow (T, S, q))$$

$$r' = (L'_1 \wedge L'_2 \wedge \ldots \wedge L'_n \rightarrow (T', S', q))$$

with $S \cap S' \neq \emptyset$ and $T \cap T' \neq \emptyset$ the condition $\exists [L_i = (T_i, S_i, p), L'_j = (T'_j, S'_j, p)]$ such that $T_i \cap T'_j = \emptyset$ or $T_i \cap T'_j = \emptyset$ is fulfilled then, the Rule Base is consistent.

An algorithm to determine the consistency of a Rule Base checking the condition of the theorem is given below.

Search.rule.couples(RB) returns the set of couples of rules having a same proposition in their literal consequent and whose interval values have a common set of values, namely the intersection between the two intervals is not empty.

Compare(r,r') checks if the antecedents of the rules r and r' have literals with the same proposition and whose interval values are disjoint. In the affirmative case it returns True otherwise it returns False.

Procedure 8.1
Consistency(RB)

```
    $Consistency \leftarrow True$
    $Couples \leftarrow Search.rule.couples(RB)$
    For each $(r, r') \in Couples$ and while Consistency=Tru
        $Consistency \leftarrow Compare(r, r')$
If Consistency=True then return("True") else return("Fal
End
```

Theorem 8.2 Consistency Complexity. *The Consistency algorithm is in $O(R^2.M^2)$, where R is the maximal length of the rule antecedents and M is the total set of rules, namely the cardinality of the Rule Base.*

Remark. Notice that the consistency check is executed in OFF-line.

8.2 Completeness

Another anomaly in defining a Rule Base consists in the existence of an incompleteness situation. As for the inconsistency case, we introduce this concept be means of an example.

Example 8.2 *Assume now that inside the Rule Base there exists the following two rules:*

$$([-\infty, 2], x_1) \wedge ([-\infty, 2], x_2) \rightarrow ([High, High], o)$$

$$([-\infty, 3], x_1) \wedge ([2, 2], x_1) \rightarrow ([Low, Low], o)$$

One can verify that the output control signal is not defined for the case where $x_1 = 3$ and $x_2 = 1$ (we assume that there is no other rules involving x_1 and x_2) .

The fact that for a certain combination of the values of the variables there is no defined an output control signal has two possible explanations:
- The expert has forgoten this situation, or
- The combination is unreal (for instance, the corporal temperature cannot be bigger than 45 degrees).

The role of a automatic validation tool is to communicate to the expert of the incompleteness situation, and next to update the Rule Base in the case where he considers it is necessary.

Now, we formalise the concept of completeness.

Theorem 8.3 *A RBS is incomplete if there exists a tuple $(v_1, v_2, \ldots, v_n) \in Dom(x_1)\text{x}Dom(x_2)\text{x}\ldots\text{x}Dom(x_n)$ where the set of x_i's are the set of input and deduction variables, such $(v_1, x_1), (v_2, x_2), \ldots, (v_n, x_n), RBS$ does not entails logically and output value.*

The algorithm to check the completeness of a Rule Base is easy to design: it has to check that any value of the Cartesian Product $Dom(x_1)\text{x}Dom(x_2)\text{x}\ldots\text{x}Dom(x_n)$ must be associated with an output value.

The worst-case complexity of a simple exhaustive search algorithm is exponential with respect to the partition of the domains of the variables x_i's of the Rule Base. However, notice that this checking is done OFF-line and that some techniques could be elaborated to design an algorithm with a polynomial complexity for most of the applications.

9 Complexity Issues

The forward algorithm is a straightforward extension of the algorithm [2], designed for the propositional case, to our temporal multi-valued language.

In our case, we assume that the connectors must be ranged in levels. Connectors of level 1 are those whose antecedents are all input variable. Connector of level k+1 are those whose antecedents are at most of level k and at least there is a literal with level k.

Linearity The linear complexity O(n) of the algorithm in [2] enable to design an interpreter with linear complexity:

- Connectors are processed by levels and each connector is computed at most once.
- To determine if a rule can be fired each antecedent must have a truth evolution in the current instant. To check this, one needs to match each literal (T,S,p) with the input or deduced correspondent literal (T_d, S_d, p). Thus (T,S,p) is true whenever $T_d \subseteq T$ and $S_d \subseteq S$.

Reactivity. Now, as the complexity of the algorithm is O(n), by definition of big "O", we know that the maximal time elapsed by the propagation process is $k_1.n + k_2$, where k_1 and k_2 are experimental constants. Knowing that, once the KBRS is definitive ant not modified anymore, n is fixed, the maximal time to compute the output signals is fixed and it can be determined experimentally. In other words, with n fixed, the complexity of the algorithm is $O(1)$ verifying in this way the reactivity property.

Coarse Parallelism. This time can be reduced using parallelism. Thus, with a coarse granularity in a multiprocessor architecture of p processors, the reactivity time can be reduced to up $K' < K$ and $K' > K/p$.

Fine Parallelism. If we use a fine granularity architecture with one processor per literal and one per each connector, we can achieve a parallel complexity of $O(D.logR)$, where D is the number of levels in the KBRS and R is the maximal cardinality of the antecedents of the rules. Hence, $K'' < K'$.

10 Conclusion

In this article we have proposed a language, an interpreter an a automated validation system to perform control of physic systems in real time. This method relies on the Knowledge Rule Based System paradigm. The language underlying our approach is a temporal multivalued language which cope with two major issues in representing knowledge: on the one hand, imprecise and incomplete information, and on the other hand, temporal information.

Thus our method arises as an alternative to develop control in real time when the physic process is difficult to model, escaping in this way to the mathematical line attached to the Control Systems field.

Many advantages of our KBRS devise have been pointed out and the strict restrictions in computing the output signal demanded by a real time control loop has been showed to be fulfilled.

Acknowledgements: This work has been supported by the project REST, TIC 96-0721-C02-02. Agradezco sinceramente los comentarios constructivos del revisor que serán tenidos en cuenta en su totalidad en trabajos futuros. Debido a la proximidad de los plazos previstos, sólo he podido integrar algunos de ellos en esta versión.

References

[1] M Boddy and T. Dean. Deliberation scheduling for problem solving in time-constrained environments. *Artificial Intelligence*, 67:245–265, 1994.

[2] W.F. Dowling and J.H. Gallier. Linear-time algorithms for testing the satisfiability of horn propositional formulae. *Journal of Logic Programming*, 3:267–284, 1984.

[3] R. Hahnle and G. Escalada-Imaz. Deductions in many-valued logics: A survey. *Mathware and Soft Computing*, 4(2):69–97, 1997.

[4] M. H. Klein, T. Ray, B. Pollack, R. Obenza, and M. Gonzalez-Harbour. *A practioner's handbook for real-time analysis. Guide to Rate Monotonic Analysis for real-time systems. Klower Academic Publisher, 1993.* Kluwer Academic Publisher, 1993.

[5] E. Kligerman and A.D. Stoyenko. Real-Time Euclid: A language for reliable real-time sytems. *IEEE Transaction on Software Enginereering*, SE-12(9):941–949, 1986.

[6] R. E. Korf. Depth-limited search for real-time problem solving. *Real-Time Systems*, 2(1-2):7–24, 1990.

[7] R. E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42(2-3):189–211, 1990.

[8] R. Michalski and P. Winston. Variable pricision logic. *Artificial Intelligence*, 29(2):121–146, 1986.

[9] D.J. Mulsliner, E.H. Durfee, and K.G. Shin. CIRCA: A cooperative intelligent real-time control architecture. *Trans. on Sytems, Man and Cybernetics*, 23(6):1561–1574, 1993.

[10] D.J. Mulsliner, J.A. Handler, E.H. Durfee, J.K. Strosnider, and C. Paul. The challanges of real-time artififcial intelligence. *Computer IEEE*, 28(1):58–66, 1995.

[11] P. P. Puschner and C. Koza. Calculating the maximum execution time of rela-time programs. *Real-Time Systems*, 1(2):160–176, 1989.

[12] J. A. Stankovic and K. Ramamritham. What is predictability for real-time sytems ? *Real Time Sytems*, 2(4):247–254, 1990.