Fault Tolerant Control: An Imprecise Computation Setting

ADDISON RIOS-BOLIVAR§  LUIS PARRAGUEZ‡  FRANCISCO HIDROBO§
MARGARITA HERAOUI‡  JULIMA ANATO‡  FRANCKLIN RIVAS§

§Universidad de Los Andes
Facultad de Ingeniería
La Hechicera, Mérida 5101
VENEZUELA
ilich,hidrobo,rivas@ula.ve

‡Universidad de Oriente
Departamento de Ingeniería Eléctrica
Núcleo Anzoátegui, Puerto La Cruz
VENEZUELA
parraguezl,anato,heraouim@cantv.net

Abstract: In this contribution the existing relations between the imprecise computation and the fault tolerant control (FTC) are analyzed. From those relations one settles down like constructing FTC systems according to the model of imprecise computation. The found relations establish that the obligatory tasks in the model of imprecise computation correspond with the mission of maintaining the stability of the systems under FTC according to the redundancy degree that assure the structural properties of the systems. On the other hand, the optional tasks in imprecise computation correspond to as the performance criteria of the systems under FTC are satisfied, which can be degraded under adverse conditions of operation of the system. These correspondences are probed in an example.


1 Introduction

The advances in the system theory and in the design of control mechanisms, from the technologies of information and communication (TIC), have allowed that the industrial complexes satisfy high levels of productivity, which are quantified by performance indices. In spite of these advances, the complex systems do not get to totally fulfill the objectives for which they were designed, reason why situations of bad operation can be generated whose effects can be very severe: losses of production, environmental damages, losses of human lives, etc. Therefore, jointly with the falsified and complex systems that allow to reach excellent production indices, specialized systems are required in order to guarantee rigorous objectives of security, reliability and availability.

The conjunction of the mechanisms to obtain performance indices for security and reliability conforming the fault tolerant systems, which leave from the identification of the root cause of the anomalous operations.

Definition 1.1 For a controlled system, the Fault Tolerance (FT) is defined as the ability of the system to maintain the control objectives, in spite of the occurrence of a failure, thus, under anomalous operation can be accepted a degradation of performance index.

A system important to identify the problems of operation in real time and with the required speed, corresponds to the filters for fault detection and isolation (FDI). These filters allow to find the root cause by means of the identification of the components or functional blocks of the system that do not operate of nominal way. The information that is generated from the FDI filters allows, besides recognizing the functional devices or blocks that operate outside their ratings, to establish an operation level satisfying performance objectives, in spite of the presence of failures.

The Fault Tolerance (FT) has as goal to design a control system with a determined structure in order to guarantee the stability and a satisfactory performance, not only when components are under nominal operation, but in case that the components (sensors, actuators, or process elements), fail [4]. Thus, the FT is defined in reference to the system objectives (stability, performance) and the given faults. The designed control systems under this conception is known as fault tolerant control system.

A FTC includes fault diagnosis in real time and the taking of decisions, in order to avoid or to lessen the adverse consequences of those failures, from the reconfiguration or intelligent change of the control systems. Thus, the FTC problem consists in to design, under a given degree of redundancy, intelligent control systems such that integrity of the system is guaranteed and certain performance indices are satisfied with performance [3].

For the analysis of the FT, it is necessary to define when, for a given system and under a failure situation,
is still possible to reach the main targets of operation of the controlled system. The FT synthesis tries to provide to the system a hardware architecture and the software mechanisms that would allow, as far as possible, to reach certain performance objectives not only under normal operation, but also in adverse situations.

Thus, it is easy to infer that in the design of an FTC system is necessary to guarantee that in normal operation the control guarantees the stability and the performance objectives of the system, and that under the fault conditions, for which it was designed, guarantees of obligatory way the stability and of optional way the best performance than can be obtained.

In a Real Time System (RTS) is not only important the logic validity of the answers which provides, but this must be generated before a determined temporary period ends [11]. Then, we clearly highlight that a RTS is a computational system in which the tasks, or at least a part of them, have a maximum temporary delivery period known as delivery period (deadline).

A task can be a computation of a control action, the transmission of a message, the execution of a command, the recovery of a file, etc. The breach of a delivery period, in a critical system, is considered a failure and, therefore, ends in an unacceptable condition [8].

In many situations is preferable a low quality answer (approximated), but in time, than a high quality result (precise), but late. In this sense, some relations between the FTC and the objectives that are considered in the model of Imprecise Computation with the purpose of to orient a mechanism based on the fulfillment of obligatory tasks (stability condition of controlled systems), and optional tasks (certain performance indices), can be established in order to design fault tolerant control systems.

2 Imprecise Computation

The Model of Imprecise Computation sets out to solve situations of transient computer overloads and to reinforce the Fault Tolerance in Systems of Time Real Critics (Hard-RTS), essentially with restrictions of time [9]. To appreciate the basic idea of the technique of Imprecise Computation is highlighted the fact that, often, the global behavior of a system may be tolerable, even in presence of temporary failures, if the more important tasks are completed on time. Therefore, instead of making the operating system handle all the tasks in the same way, the programmer may identify some as obligatory, meaning with this that they must be completed in their respective ranges of time, and other tasks as optionals, indicating that this last can be skipped without causing any failure considered as intolerable. In that order, the operating system, in overload conditions, can skip the less important tasks, trying to execute on time the important ones.

From the point of view of the Real Time community, the Imprecise Computation turns out to be, in principle, a generalization of Anytime Computing [14], as it is known in the field of Artificial Intelligence. The Anytime Algorithms are those where the quality of the results is improve gradually as long as the time of execution increases.

In that order of ideas, there exist situations where a low answer in quality is preferable (approximated), but in time, that a result of great quality (precise), but inopportunistically, as it is the particular case of the controlled processes [2]. In industrial processes controlled by computers during the initialization of the system or in an operation of emergency, occur transients increases at the computational load, that degrade the quality, accuracy and opportunity for the controller of the system to respond.

Thus, the technique of Imprecise Computation is based on the observations before mentioned, as well as of the fact that a good approximate quality can, often, to obtain by far less computer resources, especially time of processor. A system of imprecise computation allows choosing the computational accuracy of each task, in order to reach to the purposes of the handler of the system with a transient overload of the computation, annotating the time of the services to keep the operation in the conditions imposed for the planning that has been established.

2.1 Basic model of Imprecise Computation

The workload used to characterize the Imprecise Computation is based on the classical deterministic models of the real time application that define an application as a T set of not expulsive tasks, \( T = \{T_1, T_2, ..., T_n\} \). The tasks can present, as it was specified before, data/control dependencies that impose precedence restrictions on the order of its execution that is denoted by the operator “<”.

**Definition 2.1** A task \( T_i \) is predecessor of another task \( T_j \), and \( T_j \) a successor of \( T_i \), defined by \( T_i < T_j \), if \( T_j \) can not start the execution until \( T_i \) has ended.

**Definition 2.2** A task \( T_i \) is immediately predecessor of another task \( T_j \), and \( T_j \) an immediate successor of \( T_i \), if \( T_i < T_j \) and there is not another task \( T_k \) such that \( T_i < T_k < T_j \).

**Definition 2.3** Two tasks \( T_i \) and \( T_j \) are independents when there is no precedence relationship \( T_i < T_j \) or \( T_j < T_i \) and they can be executed in any order.
In the basic model of imprecise computation, each task $T_i$ in the system is defined by the following parameters: Arrival Time (ready time) $r_i$; Delivery Period (deadline) $d_i$; Processing time $τ_i$; Assigned Time of Processor $σ_i$; Relative weight $ω_i$. Each task $T_i$ is decomposed in two tasks, also called sub-tasks or parts: Mandatory Part $M_i$; Optional Part $O_i$.

The arrival time and the delivery period of the tasks $M_i$ and $O_i$ are the same for $T_i$. $M_i$ is an immediate predecessor of $O_i$. If it is denoted by $τ^m_i$ the time of the mandatory task and by $τ^o_i$ the time of the optional part of the process, where $τ^m_i$ and $τ^o_i$ are rational numbers, then $τ^m_i + τ^o_i = τ_i$.

A planning is an assignment of the processor for the task in $T_i$, in not associated intervals of time. A task is said planned, in an interval of time, if the processor is assigned to the task of that interval. The quantity of time of the processor assigned to the task is the sum of the not associated intervals of time. In any valid planning, for systems with just a processor, the assigned time of the processor ($σ_i$), is assigned only to a task at the time, and every task is assigned after its arrival time, and all the predecessor tasks have been finished, and the quantity of time for the processor assigned to the mandatory part is equal to $τ^m_i$ and the completion of this must occur for the most at the delivery time $d_i$. The optional part of a monotonous task can be completed at any time instant $t ≤ d_i$. From here, the quantity of time assigned $σ_i$ for the processor to a monotonous task must be in the interval $τ^m_i ≤ σ_i ≤ τ_i$.

The quantity of time for the processor assigned to a task with 0/1 restrictions can be $τ_i$ or $τ^m_i$, according to if there will executed or not its optional part. When $σ_i = τ_i$, we say the task is precisely planned; on the contrary, if the quantity of time for the processor assigned is less than the time for the processor required for the total computation task, $σ_i < τ_i$, then we are talking about imprecise planning.

A task $T_i$ is completed in a planning, if its mandatory part is finished in a traditional sense. A valid planning is a reachable planning if all the tasks are finished before its delivery time. The global performance of the system under imprecise computation can be measured from the evaluation of errors, [1, 6, 5, 7, 13].

### 3 FTC and Imprecise Computation

Normal operation of any control system settles down by the fulfillment of certain relations between the different variables from the controlled process. These relations allow to define, with a high degree of precision, the behavior to future of the system, which can be verified from the measured data of the process. From the operational point of view, those same relations also allow the characterization of the performance of the system in a temporary space, being able to establish what it has been denominated like operation in real time.

On the basis of the previous thing, from an performance model, the anomalous and normal operation regions can be characterized and taking into account the answers from the process in certain times previously established. Consequently, a region of normal operation is that one where all the relations between the variables of the process are satisfied within the established times. This means that the region of normal operation characterizes in that all the tasks (relations between variables) obligatory are executed according to its times and margins of time for optional tasks are allowed, in which those factors are included that can disturb the operation of the process (exogenous signals and/or uncertainties), without reducing their global performance based on the operational objectives, (see Figure 1).

![Figure 1: Process Operation Regions.](image-url)
evaluate the operation of the processes when the imprecise tasks are admitted, and there exists the possibility of the recovery towards the region of precise operation. This demand to analyze the propagation of the failure and the level of redundancy that would allow to maintain, on the one hand, the certain possibility of the execution of the obligatory tasks and a certain temporary degree for the execution of optional tasks. Thus, the FTC based on imprecise computation would allow the construction of mechanisms, based on obligatory and optional tasks, that they still assure the availability and the performance the systems in adverse conditions.

At the same time, the process model, jointly with its valuation and from a structural analysis, allows to the definition and construction of the mechanism of detection and diagnosis. Thus, the model and the fault propagation also will govern the design of the actions to take in situations of anomalous operation, such that as the supervision supports of the reliability system centered in the fault diagnosis.

On the base of the imprecise computation, the fault model is defined by the structuring and temporary planning of the tasks. That is to say, the model is associated to the execution or not of the obligatory and optional tasks according to the assigned times. Everything what is outside those time intervals constitutes as failure, whose effects will have to be mitigated with the actions be executed.

3.1 Passive FTC

For FTC based on imprecise computation, some conditions of passive and active FTC possible can be established, such as in other techniques. Thus, the passive FTC is defined as the robust computation, that is, a computational system is designed (control system), which under certain conditions of failures, executes the obligatory tasks that are coordinated for to guarantee the stability of the controlled system. In the measurement of the possibilities, the totality or some optional tasks will be executed, which will guarantee a degree of acceptable performance, although degraded in a certain quantitative and/or qualitative level.

It is evident that the greater commitment for the design of the robust system is based on the capacity of the planning of the tasks that, as far as possible, must be constructed for a maximum utility, according to the metric criteria associated to the performance indices, (to see the Section 2.1). This means that of the planned tasks, besides the obligatory ones, those optional ones must be executed, in order to guarantee the performance index, which is constructed on the base of a criterion of optimization of maximum utility.

3.2 Active FTC

In the context of the Active FTC under imprecise computation, a computational system must be to designed, under a given degree of redundancy, such that the system integrity is guaranteed, which means that the obligatory tasks must be executed, and the performance is satisfactory, still degrading quantitatively and/or qualitatively the optional tasks, through the planning involving the reconfiguration or accommodation.

Again, the design consists in the construction of a detector system and a supervisor system, which operating of simultaneous way to diminish the failure effects, taking the corresponding actions. The detector and the supervisor correspond to the supervision-diagnosis block and the effector is related to the reconfiguration mechanisms. In this last one, the actions are taken, from the impact of the failure, of its propagation and the level of redundancy available. Thus, this mechanism allows to guarantee the integrity of the operation according to the obligatory part, and as optional tasks must be suppressed or be delayed, assuring a certain performance level.

Needing the tasks and actions within the operational framework in real time, the Figure 2 allows to distinguish as it is the handling of signals between the different functional blocks.

![Figure 2: Active FTC in Real Time.](image-url)
4 Example

Let’s be system

\[
\begin{align*}
\dot{x}(t) &= \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix} x(t) + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u(t) + \begin{pmatrix} -1 \\ 0 \end{pmatrix} \nu(t) \\
y(t) &= \begin{pmatrix} 1 & 1 \end{pmatrix} x(t);
\end{align*}
\]

where the fault mode \( \nu(t) \) has the same form that the control signal \( u(t) \), that is, \( \nu(t) = u(t) \) unless the fault appears in an unknown time. This characterization describes actuator faults, where, at the time in that the anomaly is presented, the structure of the system changes, conserving structural properties such as the controllability of this system. This condition establishes an analytical redundancy in the actuator.

In order to control the system without fault, a PID controller has been designed, with \( K_p = 2, K_I = 10 \) and \( K_D = 0 \). This type of control requires the calculation of a proportional action and an integral action, which allow to assure the stability of closed loop (obligatory task), and some conditions of performance (optional tasks). In the absence of fault, the system responses are showed in the Figure 3.

![Figure 3: Controlled system signals without faults.](image)

For the fault diagnosis a filter based on state observers has been designed, [10]. The filter dynamic is given by

\[
\begin{align*}
\dot{\hat{x}}(t) &= \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix} \hat{x}(t) + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u(t) + \begin{pmatrix} 1 \\ 4 \end{pmatrix} (y(t) - \hat{y}(t)) \\
\hat{y}(t) &= \begin{pmatrix} 1 & 1 \end{pmatrix} \hat{x}(t);
\end{align*}
\]

which allows the fault diagnosis in virtue that the dynamics of the estimation error is asymptotically stable.

For the simulation aims, the fault appears in \( t = 10s \), from which the actuator changes since the control matrix will be, as a result of the anomaly \( B = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \).

With that structure, the PID controller, who before the fault has regulated satisfactorily the system, it cannot stabilize the system in closed loop, such as is showed in the Figure 4.

![Figure 4: System signals without tolerant control.](image)

Consequently, the fault diagnosis filter generates residues that will be used to diminish the failure effects allowing to take action in that sense.

On the contrary, a FTC has been designed, which allows to redefine the structure of the controller, in order to maintain the obligatory operation requirements. In the moment of the fault detection, the controller is switched to an action appropriate, in this case, that action is obtained by a controller by static output feedback \( u(t) = K_y y(t) \), where \( K_y = -10 \) is the feedback gain, which allows to assure the obligatory task of stability in closed loop, executing itself as well, with a smaller requirement of calculations. The switching is realized through the evaluation of the residues, which are generated by the fault detection filter.

The results of the FTC are shown in the Figure 5. In this way, the effectiveness of the method is demonstrated because the performance requirements are satisfied.

5 Conclusions

In this work an analysis on the relation of the Fault Tolerant Control (FTC) and the imprecise computation has been presented. From the study of the fault tolerant control systems and of the imprecise computation some relations have been established, that allow to orient schemes for the construction of FTC systems according to the techniques of imprecise computation. These relations characterize that the obligatory
tasks in imprecise computation correspond to the tasks that guarantee the stability of the systems under FTC and according to the degree of redundancy, which is assured by the structural properties of the systems. Whereas the optional tasks correspond to satisfy performance criteria of the controlled systems, which can be degraded according to the operation of the systems under adverse conditions. The results have been validated from a numerical example.

Acknowledgements: The research was supported by the FONACIT under the grant 2005000170, and by the CDCHT-ULA, through grant I-1103-08-02-A.

References:


