UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FISICAS Y MATEMATICAS
DEPARTAMENTO DE INGENIERIA ELECTRICA

“DESIGN OF HYBRID PREDICTIVE CONTROL STRATEGIES FOR OPTIMIZATION OF OPERATIONAL PROCESSES IN DYNAMIC TRANSPORT SYSTEMS”

TESIS PARA OPTAR AL GRADO DE DOCTOR EN INGENIERIA ELECTRICA

ALFREDO ANTONIO NÚÑEZ VICENCIO

PROFESORA GUÍA:
DORIS ANDREA SÁEZ HUEICHAPAN

PROFESOR CO-GUÍA:
CRISTIAN EDUARDO CORTÉS CARRILLO

MIEMBROS DE LA COMISION:
Aldo Cipriano Zamorano
Michel Gendreau
Guillermo González Rees
Jorge Silva Sánchez

SANTIAGO DE CHILE
Diciembre 2009

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE
DOCTOR EN INGENIERIA ELECTRICA
POR: ALFREDO ANTONIO NÚÑEZ VICENCIO
FECHA: 11/12/2009
PROFESORA GUÍA: SRA. DORIS SÁEZ H.
PROFESOR CO- GUÍA: SR. CRISTIÁN CORTÉS C.

“Diseño de Estrategias de Control Predictivo Híbrido para la Optimización de Procesos Operacionales de Sistemas Dinámicos de Transporte”.

El objetivo de esta tesis es el desarrollo de nuevas metodologías de diseño de estrategias de control predictivo híbrido para sistemas dinámicos no-lineales que incluyen variables discretas y continuas. La metodología se diseña para aplicaciones reales, en particular para el estudio de sistemas dinámicos de transporte, incluyendo políticas operacionales y de servicio, así como reducción de costos. La formulación del controlador se basa en una definición adecuada de las variables esenciales del proceso y su evolución en el futuro, en una función objetivo flexible capaz de capturar las predicciones de las variables esenciales, y un algoritmo de optimización eficiente, principalmente proveniente de la Inteligencia Computacional, para optimizar en tiempo real los índices de desempeño de las aplicaciones.

El marco teórico de la nueva metodología de control predictivo híbrido es genérica, y extensible a otros procesos industriales que involucran dinámicas no lineales y variables tanto continuas como discretas. Se consideran técnicas de Inteligencia Computacional como modelación difusa y algoritmos evolutivos, debido a que la formulación predictiva resultante involucra tanto modelación no lineal como optimización no lineal entera mixta (problemas del tipo NP-Hard).

Una característica importante de la nueva metodología desarrollada es el uso de dos enfoques de optimización. Dadas las propiedades de las aplicaciones, primero se ocupa un enfoque clásico mono-objetivo; y luego, de forma novedosa se propone el uso de un enfoque basado en optimización multi-objetivo, en el cual se tienen objetivos contrapuestos y la decisión de control se selecciona observando el compromiso entre soluciones Pareto óptimas (por ejemplo entre costos de usuarios y costos operacionales en el caso de la aplicación en sistemas de transporte).

En resumen, los principales aportes de esta tesis son los siguientes. Primero, se presenta una nueva clase de modelos híbrido-difuso y una metodología de identificación para el caso de modelos tipo Witsenhausen modificados usando clustering difuso y análisis de componentes principales. Se diseña un nuevo tipo de controlador predictivo híbrido multi-objetivo, el cual genera frentes de Pareto dinámicos de los cuales se escogen las acciones de control adecuadas (según un criterio). Se presenta una nueva formulación del problema de control predictivo mono-objetivo y multi-objetivo para el sistema dial-a-ride considerando demanda y condiciones de tráfico incierta. Se propone un nuevo esquema de detección de situaciones anormales para el sistema dial-a-ride, el cual detecta condiciones de tráfico inesperadas. Finalmente, se formula y diseña un problema de control integrado para un sistema dial-a-ride que interactúa con un corredor de transporte público.

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE
DOCTOR EN INGENIERIA ELECTRICA
POR: ALFREDO ANTONIO NÚÑEZ VICENCIO
FECHA: 11/12/2009
PROFESORA GUÍA: SRA. DORIS SÁEZ H.
PROFESOR CO- GUÍA: SR. CRISTIÁN CORTÉS C.

“Design of Hybrid Predictive Control Strategies for Optimization of Operational Processes in Dynamic Transport Systems”

The core of this thesis is to develop a new methodology for the design of predictive control strategies for non-linear dynamic hybrid systems, including discrete and continuous variables. The methodology is designed for real applications, particularly the study of dynamic transport systems, considering operational and service policies, as well as costs reduction. The control structure is based on a proper definition of the key variables and their evolution in the future, a flexible objective function able to capture the predictive behaviour of the key variables, and efficient algorithms, mainly coming from the computational intelligence framework, to optimize performance indices for real-time applications.

The framework of the proposed predictive control methodology is generic, and extendible to other industrial processes involving non-linear dynamics with both continuous and discrete variables. As the resulting predictive formulations involve both non-linear modelling and non-linear mixed integer optimization, which is known to be NP-Hard, computational intelligence methodologies are considered, among them fuzzy modelling and evolutionary algorithms.

One major feature of the proposed developments is the methodology utilized in the optimization procedure under the predictive control approach. Given the properties of the applications, it was decided to explore first, a classical mono-objective approach, and later to propose a new approach based on a multi-objective optimization procedure, in which many objectives are opposed and the trade-off between Pareto optimal solution is obtained (for instance users versus operational costs in case of transport applications).

In summary, the main contributions of this thesis are as follows. First, a new class of hybrid fuzzy models and an identification methodology for Modified Witsenhausen models using fuzzy clustering and principal component analysis are derived. A new multi-objective hybrid predictive control design is derived generating control actions from a dynamic Pareto front. A new formulation of mono-objective and multi-objective predictive control of a dial-a-ride system considering uncertain demand and traffic conditions is postulated, formalized and tested through simulation experiments. A new fault detection scheme for abnormal situations of a dial-a-ride system is designed for detecting unpredictable traffic conditions. Finally, the design of an integrated control problem is formulated for a dial-a-ride problem of a fleet of vehicles together with a fixed-route public transport system.

ACKNOWLEDGMENTS

I would like to thank to all the people who helped me in the developing of this thesis. First of all, to my parents Leticia and Guillermo, my sister Natania and my brother Guillermo, for their unconditional support and for have been always available to help me whenever I need them. I would like to thanks professor Doris Sáez, for teaching me to enjoy working as researcher. She is a dedicated control expert, devoted educator, and my most important influence as researcher and the model reference I would like to track. I would also like to thank professor Cristian Cortes, for his kindly and his strong support in all the stages of my doctorate studies. Their thoughtful comments and meticulous reading of my thesis have enhanced the quality of my work. Also I feel very lucky to have a very supportive and encouraging thesis committee, Prof. Gendreau, Prof. Cipriano, Prof. Gonzalez and Prof. Silva.

In this paragraph, I explicitly thank the support of CONICYT, as this thesis was financiered by them. Also, my presentations in the World Congress Conference on Computational Intelligence (Vancouver, Canada), IFAC World Congress (Seoul, Korea) and my pre-doctoral research stay in the University of Ljubljana (Slovenia), were all possible because of their financial support, and of course I am very gratefully for that.

I want to thank to Gorazd Karer and professor Igor Skrjanc, for making my stay in Ljubljana a very nice one, and for been a huge influence in my decision to follow an academic career. Without their influence, I wouldn't make the decision to apply for a postdoc in The Netherlands, which is my next project (in TUDelft).

I would like to recognize the contribution of each people evolved in the research work. The ones that belong to the working-team of professors Doris Sáez and Cristián Cortés are Diego Muñoz, Marcela Riquelme, Alejandra Pillajo, Patricio Torres, Fredy Milla, Alejandro Tirachini, Pablo Medina, Roberto Zuñiga and Javier Causa. The ones that belong to the professor Igor Skrjanc team are Gorazd Karer and Simon Oblak.

I would like to mention in a very special part Professor Guillermo Gonzales for all his support and for the best classes I had during my career. He is definitely the best teacher I had.

Finally, I’m going to mention all those people who during my doctorate studies made my life more funny, without them I would never finish my thesis. My nephews Guito and Diego, and my sister-in-law Patricia. My friends Alejandra Pillajo, Paola Veliz, Loreto Carrasco, Paulina Gaona, Loreto Boitano, Pablo Medina, Fabiana Maldonado, Larry Uribe, Rigo Gaona, Seba Garelli, Esteban Cortés, Rodrigo Alvarado and Sergio Monsalve. My Slovenian friends Vito, Nusa, Simon, Miha, Vik, Mateja, Meta, Maja, Zori, Ursula, Nina, Stane, Biba, Jadry. The Croatian ones Skedy, Josepa and Pavla.

At last but not least I want to thank God and all my relatives in the heaven.

1. Introduction.

Hybrid Systems represent a large class of systems that contains continuous and discrete/integer variables. Systems described by physical laws, logic rules, operating constraints described by both differential and algebraic equations are hybrid systems too. Hybrid Systems have received much attention from computer science and from the community of control in the recent years. Given the high complexity of hybrid systems, development of ad-hoc hardware and mathematical tools available to model and treat them are required.

In this thesis, a methodology is developed for the design of predictive control strategies for non-linear dynamic hybrid systems, including discrete and continuous variables. The methodology is designed for real applications, particularly the study of dynamic transport systems, considering operational and service policies, as well as costs reduction. The control structure is based on the modelling of key variables that describe the system, a flexible objective function able to capture predictions of future behaviour associated with key variables, and efficient algorithms to solve and optimize performance indices for real-time applications.

Although the methodologies were originally thought for dynamic transport applications, the framework turned out to be more generic, and extendable to other industrial processes involving non-linear dynamics with both continuous and discrete variables (for instance power plants, chemical plants, etc.). As the resulting predictive formulations involve both non-linear modelling and non-linear mixed integer optimization, which is known to be NP-Hard, computational intelligence methodologies are considered, among them fuzzy modelling and evolutionary algorithms.

One major feature of the proposed developments is the methodology utilized in the optimization procedure under the predictive control approach. Given the properties of the applications, it was decided to explore first, a classical mono-objective approach, and later, a multi-objective optimization procedure, in which many objectives are opposed (for instance users versus operational costs).

The present thesis is structured considering the following chapters.

Chapter 2 presents identification methods of hybrid systems, which are systems in discrete-time and that have mixed continuous and discrete input/states. The methods are based on Piece-wise Affine (PWA) models and Modified Tanaka Model (MTM), which is a hybrid fuzzy model.

First, a new class of hybrid models is presented. The class, denoted fuzzy hybrid model, is introduced and will be used to model hybrid systems with different non-linearities defined in different operating regions. Then, an identification methodology for the PWA model is presented. The method determines first a partition for the data set by using fuzzy clustering, and then in each region a local linear model is determined. An illustrative experiment on a Batch Reactor system is conducted to compare PWA with fuzzy identification method.

Later, an identification methodology for the hybrid fuzzy model called Modified Tanaka Model (MTM) by using fuzzy clustering and principal component analysis is described. The method is inspired in the "inverse" form of the merge method for clusters, which makes it possible to identify the consecutive clusters that are more different and, therefore, to use this idea to identify the unknown switching points of a process based on just input-output data and then to obtain the number of sub-models to be identified. An illustrative experiment on a hybrid tank system is conducted to show the benefits of the proposed approach, compared with the classical Takagi-Sugeno identification.

Chapter 3 presents Hybrid Predictive Control (HPC) methods of hybrid systems. The fuzzy hybrid models using the identification techniques proposed in chapter 2, are used for the HPC design, where the optimization problem is solved efficiently by Genetic Algorithms (GA). Illustrative experiments on a hybrid tank system and in a Batch Reactor were conducted to demonstrate the benefits of the proposed approaches.

Additionally in Chapter 3, a multi-objective hybrid predictive control (MO-HPC) based on fuzzy hybrid modelling is presented. At every instant, a proper optimization algorithm is used to find the dynamic Pareto optimal front. Provided that only one input can be applied to the system, the controller must use a criterion to choose a proper solution from the Pareto set (among those solutions typical hybrid predictive controller solution). Then, the controller can change the importance of the objectives without tuning or solving a new optimization problem, by just exploring in different ways the Pareto optimal front, using optimal solutions at every instant. Illustrative experiments on a hybrid tank system were conducted to show the advantages of the proposed MO-HPC. As an additional application, the behaviour of a MO-HPC was emulated using a Hybrid Predictive Controller. Using the emulator, the operator/dispatcher does not have to supervise the MO-HPC as his (her) decisions were modelled in a HPC with time-varying weighting factors.

Chapter 4 presents a hybrid predictive controller, as described in Chapter 3, applied for dial-a-ride problem to incorporate future information regarding unknown demand and expected traffic conditions, in the context of a dial-a-ride problem with fixed fleet size. As the routing problem is dynamic, several stochastic effects have to be considered within the analytical expression of the dispatcher assignment decision objective function. This approach is focused on two issues: one is the extra cost associated with potential rerouting arising from unknown requests in the future, and the other is the potential uncertainty in travel time coming from non-recurrent traffic congestion from unexpected incidents. These effects are incorporated explicitly in the objective function of the hybrid predictive controller. In fact, the proposed predictive control strategy is based on a multivariable model that includes both discrete/integer and continuous variables. The vehicle load and the sequence of stops correspond to the discrete/integer variable, adding the vehicle position as an indicator of the traffic congestion conditions.

In addition, Chapter 4 includes an analytical formulation of the proposed prediction models that allow us to search over a reduced feasible space (no-swapping). Demand prediction is based on a systematic fuzzy clustering methodology, resulting in appropriate call probabilities for uncertain future. As the dynamic multi-vehicle routing problem considered is NP-hard, the use of Genetic Algorithms (GA) is proposed that provide near-optimal solutions for the three, two and one-step ahead problems. Promising results in terms of computation time and accuracy are presented through a simulated numerical example that includes the analysis of the proposed fuzzy clustering, and the comparison of myopic and new predictive approaches solved with GA. The HPC based on GA is later analyzed under two new scenarios. The first one considers a predictable congestion obtained using historical data (off-line method) requiring a predictive model of velocities distributed over zones. The second scenario that accepts unpredictable congestion events generates a more complex problem that is managed by using both fault detection and isolation and fuzzy fault tolerant control approaches for abnormal situations. Results validating these approaches are presented through a simulated numerical example.

In Chapter 5, a framework of multi-objective Hybrid Predictive Control approach (MO-HPC), described in Chapter 3, is applied for solving the dial-a-ride problem based on a dynamic objective function that considers two dimensions: user and operator costs. As these two components aim at opposite goals, the problem is formulated and solved through multi-objective optimization. At every instant, the algorithm finds the optimal Pareto front associated with the solutions of the problem by means the dynamic routes of those vehicles in service. Since only a single solution has to be applied to the system every time a new request appears, several criteria are proposed in order to properly use the information provided by the dynamic optimal Pareto front. Thus, by using MO, the trade-off between the two conflicting objectives will become clear for the dispatcher when making dynamic routing decisions. Illustrative experiments through simulation of the process are presented to show the potential benefits of the new approach.

Chapter 6 presents the formulation of a hybrid predictive control (HPC) approach for the integrated dial-a-ride system and public transport system. Based on the prediction of state space variables, traffic conditions and demands, the dispatcher routes the fleet of the dial-a-ride system considering both user and operational costs, assuming a regular operation of the public transport system. As the optimization variables are mixed-integer, two hybrid predictive controllers (one for controlling the dial-a-ride system and one for the public transport system) are formulated. As the resulting optimization problem is NP Hard, some recommendations are included in the analysis for a real-time implementation of this strategy.

Finally in Chapter 7, main contributions of this thesis and further research are presented.

2. Fuzzy Model Identification of Non-linear Hybrid Systems.

2.1. Literature Review.

Hybrid systems represent a large class of systems that contains continuous and discrete/integer variables. Those systems given by physical laws, logic rules, operating constraints that are described by both differential and algebraic equations are hybrid systems too. Hybrid systems have received much attention from both the computer science and control communities. The reasons are, among others, the high complexity of hybrid systems and the inadequate hardware and available mathematical tools to model and treat them. Therefore new tools have to be developed for hybrid-system identification and control design in the context of industrial processes.

Yang and Blanke (2007) summary the most important contributions related to the controllability of hybrid control systems. They propose a unified approach comprising global reachability analysis at the discrete event system level, local reachability analysis at the continuous time dynamical system level and a discrete path-searching algorithm. The method was derived from Discrete Event Systems theory.

Margaliot (2006) recognizes that the difficulty in the stability analysis of hybrid systems arises from two principal factors. First, unlike ordinary differential equations, a hybrid system admits an infinite set of trajectories for any initial conditions and second, their trajectories can be much more complex. In this work, a specific approach for stability analysis based on variational principles for switched system is proposed and a link between the variational approach and the stability analysis of switched systems using Lie-algebraic considerations is presented. Mao *et al*. (2007) determine whether or not a stochastic feedback control can stabilize or un-stabilize a given non-linear hybrid system. However, the results are limited to models where the functions grow linearly, so it is not a general result.

In the present thesis, specifically, the integrated dynamic pickup and delivery problem of a fleet of vehicles (dial-a-ride system) together with a public transport system will be formulated, analyzed and solved using the predictive control approach. As the integrated system contains both continuous and discrete variables in the state and inputs, a hybrid predictive control will be used in order to include the hybrid characteristics of the system in the control actions. Some of the continuous variables are the positions of vehicles and buses, arrival times to stops, headways, etc. Regarding the discrete variables it can be mentioned the number of passengers of buses and vehicles, the sequences of task to be followed by vehicles for the study of dial-a-ride systems, station skipping strategy applied to a fixed-route bus system, etc. In this chapter, new identification methods of hybrid systems will be presented. These methods could be used, for example, in the demand pattern modelling, which in real-systems depends on some specific conditions of the system, for example conditions during rush hours, weekends, holidays, and other characteristics. With hybrid modelling it is possible to determine first the set of conditions that best represent a partition of the input space and then, to set a good model for each partition.

Next, a review of hybrid identification methods is presented, where hybrid systems identification and fuzzy modelling are highlighted.

Hybrid systems can be represented by different types of models; for example, Bemporad and Morari (1999) proposed the Mixed Logical Dynamic (MLD) models, where continuous/discrete inputs, states or outputs are considered. Heemels *et al.* (2001) established equivalencies among five classes of hybrid dynamical models: MLD, linear complementarity systems, extended linear complementarity systems, Piece-Wise Affine (PWA) systems, and max-min plus scaling systems. Each sub-class has its own advantages over the others. For example, the control techniques for MLD hybrid models, the stability criteria for PWA systems and the conditions of existence and uniqueness of the solution trajectories for linear complementarity systems.

Ferrari-Trecate *et al.* (2003) proposed a methodology for the identification of discrete-time hybrid systems in the PWA form, formulated as a discontinuous PWA map. The algorithm, based on clustering, linear-identification, and pattern-recognition techniques, identifies both the affine sub-models and the polyhedral partition of the domain on which each sub-model is valid, avoiding gridding procedures. The clustering step, used for classifying the data points, allows the identification of different sub-models that share the same coefficients but are defined on different regions. The measures of confidence on the samples are introduced and exploited in order to improve the performance of both, the clustering and the final linear regression procedure. However, if non-linear functions are considered in the regression vector, the method would tend to approximate the non-linearities with multiple linear sub-models, overestimating the real number of sub-models.

Nakada *et al*. (2005) addressed the problem of identifying a Piece-Wise AutoRegressive eXogenous (PWARX) systems by using statistical clustering. The method consists first, in the clustering of the measured data, then an estimation of the boundary hyper-planes and finally parameters estimation. Clustering, based on statistical approach, is applied by assuming the probability density of data as a mixture of Gaussian multivariate distributions. The parameters of the Gaussian densities are tuned in order to obtain the maximum for a suitable log-likelihood function so that the mixture model fits the data as accurate as possible. In this approach the number of sub-models must be given beforehand, the implementation has the numerical problem of the covariance matrix inversion, and in the algorithm the regression vector is also composed of only past inputs and outputs, so the non-linear functions of past data are not considered.

An algebraic identification procedure to cope with the identification problem of Switched AutoRegressive eXogenous (SARX) systems was proposed by Ma and Vidal (2005). Multiple ARX models are encoded in a single polynomial expression, which decouples the calculus of the parameters from the switching mechanism. The procedure allows estimating all the unknown variables that define the structure of the model, the number of discrete states and the model orders.

The Bayesian procedure proposed by Juloski *et al*. (2005) exploits some prior knowledge about the discrete states and the parameters of sub-models. The parameters of the models are treated as random variables, and described through their probability density functions. The algorithm associates each data point to a discrete mode which maximizes the probability of generating the point. In addition, the algorithm provides the misclassification weights to be used in standard multi-category robust linear programming. The Bayesian procedure requires knowing the number of discrete states and model orders, and provides a sub-optimal solution to the identification problem.

The bounded-error procedure was proposed by Bemporad *et al*. (2005) in order to identify PWARX systems. The first step simultaneously classifies the data, computes the sub-model parameters and estimates the number of discrete modes by solving the partition into a minimum number of feasible subsystems. The main feature of the method is to ensure that the module of the identification error is bounded by a fixed number, for all the data points. The bound is used as a tuning knob between complexity and accuracy.

On the other hand, many advances in fuzzy identification systems have arisen in recent years. In Celikyilmaz and Turksen (2008) a new fuzzy identification technique, which uses a combination of the function estimation method and an improved fuzzy clustering, is proposed. The new clustering algorithm considers the classical fuzzy c-means distance and the fuzzy regression residual, where membership values are used as additional inputs. This approach can better approximate the system compared with other classical fuzzy rule models.

Nefti *et al.* (2008) presented a new method for merging fuzzy sets based on clustering in the parameter space. The degree of inclusion associated with each data point is evaluated with respect to a prototype in the parameter space. The fuzzy sets are replaced by the most compatible prototypical fuzzy set, which is determined from the inclusion-based clustering algorithm.

Hadjili and Wertz (2002) proposed an identification method for Takagi-Sugeno (T&S) models (Takagi and Sugeno, 1985), incorporating the selection of optimal rules and input variables. The subtractive clustering algorithm, based on compactness and the separation of clusters, is performed in order to determine the number of rules. Then, an input variable is discarded if the fuzzy partition does not change significantly when this variable is eliminated. On the other hand, Roubos and Setnes (2001) proposed a complexity-reduction algorithm based on genetic-algorithm optimization procedures to find redundancy among the rules with a criterion based on the maximum accuracy and the maximum set similarity.

In addition, Kim *et al.* (1997) presented a combined identification method, based on the Takagi-Sugeno (T&S) and the Sugeno-Yasukawa models, in order to preserve the advantages of both algorithms. The approach implements fuzzy regression clustering as an initial tuning of the parameters and the gradient descent method to adjust them accurately.

In Abonyi *et al.* (2002), a modified Gath-Geva fuzzy clustering algorithm for the identification of T&S models is proposed to directly obtain the parameters of membership functions by using the parameters of the clusters. A linear transformation of the input variables permits to accurately recover the fuzzy partition of the antecedents. However, linear combinations of the input variables cannot be easily interpreted by the user. Then, a new cluster prototype is introduced in order to avoid the use of transformed input domains.

Zeng *et al*. (2008) proposed a new representation theorem for hierarchical systems when the discrete input space is considered. The theorem shows that one-to-one mapping for low-level functions is required to obtain a flexible hierarchical representation. Moreover, they demonstrated that flexible hierarchical fuzzy systems satisfy the universal approximation property, which allows us estimating any hierarchical function to any degree of accuracy. A hierarchical fuzzy identification method that combines expert human knowledge and limited numerical data is presented.

Although most of the developments have been made in conventional fuzzy systems, a few hybrid fuzzy identification methods are found in the literature. Palm and Driankov (1998) presented a hierarchical identification for fuzzy switched systems. The proposed method considers a black-box fuzzy identification by using fuzzy clustering and measurable discrete states in order to obtain a model for continuous state and discrete transitions. Although good performance is observed with the estimation, prior knowledge about the discrete modes is required.

Next, Girimonte and Babuska (2004) described two structure-selecting methods for non-linear models with mixed discrete and continuous inputs. The first method, based on fuzzy clustering, uses fuzzy sets to obtain the relevant inputs. The second approach is an induction algorithm included in a searching method. The results show that fuzzy clustering is faster in terms of computation time. However, the drawback of the methods is the high computation time associated with the increment of the search horizon. In the present thesis, a new identification method is proposed for non-linear hybrid systems that identify first the discrete transitions (switching points) and then all other kind of non-linearities only by means of input-output data of the process, where prior knowledge of the discrete modes is not required.

The next sections of this chapter are structured as follows. Section 2.2 presents the most important classes of hybrid systems models; among them the hybrid fuzzy models that will be used in the identification methods are highlighted. In section 2.3 a fast identification method based on fuzzy clustering for PWA models is presented. Results of the proposed method for a batch reactor process is presented and compared with alternative hybrid fuzzy modelling. Then, in section 2.4, a hybrid fuzzy identification method for MTM based on fuzzy clustering and the principal components is presented. Results of the proposed hybrid fuzzy modelling are reported for a hybrid tank system. Finally the conclusions and further research are discussed.

2.2. Classes of Hybrid Systems Models.

A general discrete-time model of the following form is considered (Bemporad *et al.*, 2002).

$$\mathbf{x}(t+1)=f\left(\mathbf{x}(t),\mathbf{u}(t),\mathbf{w}(t)\right) \tag{2.1.a}$$

$$\mathbf{y}(t)=h\left(\mathbf{x}(t),\mathbf{u}(t),\mathbf{w}(t)\right) \tag{2.1.b}$$

$$\mathbf{0}\leq g\left(\mathbf{x}(t),\mathbf{u}(t),\mathbf{w}(t)\right) \tag{2.1.c}$$

$\mathbf{x}(t)\in R^n$ is the state vector, $\mathbf{u}(t)\in R^m$ is the input vector, $\mathbf{y}(t)\in R^r$ is the output vector and $\mathbf{w}(t)\in R^l$ is a vector of auxiliary variables. Functions $f:R^n\times R^m\times R^l\rightarrow R^n$, $h:R^n\times R^m\times R^l\rightarrow R^r$ and $g:R^n\times R^m\times R^l\rightarrow R^q$ are defined.

The evolution of this model is determined in the following way. First, given $\mathbf{x}(t)$ and $\mathbf{u}(t)$, the inequalities (2.1.c) are solved for $\mathbf{w}(t)$. Then $\mathbf{w}(t)$ is substituted in (2.1.a) and (2.1.b), from where the state $\mathbf{x}(t+1)$ is updated and the current output $\mathbf{y}(t)$ is obtained.

In this Chapter, the hybrid system given in (2.1) will be assumed to be well-posed in the space of the input-state pairs. This property means that for all the pairs $\left(\mathbf{x}(t),\mathbf{u}(t)\right)$ in the input-state space, equations (2.1) have a solution $\left(\mathbf{x}(t+1),\mathbf{y}(t),\mathbf{w}(t)\right)$, and moreover, $\left(\mathbf{x}(t+1),\mathbf{y}(t)\right)$ are uniquely determined. Then, even though inequalities (2.1.c) do not uniquely determine$\mathbf{w}(t)$, the state and the output are unique functions of $\left(\mathbf{x}(t),\mathbf{u}(t)\right)$, as it happen in real systems.

The hybrid system (2.1) allows attaining discrete values for some input, state or output, by setting inequalities (2.1.c) in a proper way. Some examples of how to deal with discrete values, logic operators (namely if, then, else, and, or, and so on), etc., can be obtained in the hybrid systems literature (see for example Bemporad *et al.*, 2002 or Bemporad and Morari, 1999). Different classes of hybrid systems are determined by choosing specific forms for the functions $f(\bullet)$, $h(\bullet)$ and $g(\bullet)$. Next, the classes of hybrid systems Wittsenhausen, PWA and MLD systems are presented. Also a new class of hybrid systems called "hybrid fuzzy" is presented, and the aim of this chapter regards the identification procedure for such a class.

2.2.1. Witsenhausen Systems.

Witsenhausen systems are switched hybrid systems where the continuous states remain continuous even when the discrete/quantized states changed. The transition of a system state occurs when one or more continuous states satisfy the conditions defined for each transition. This type of hybrid system can be generically described as (Witsenhausen, 1966):

$$\begin{aligned} \mathbf{x}(t+1) &= f_{q(t)}\left(\mathbf{x}(t),\mathbf{u}(t)\right) \\ q(t) &= g\left(\mathbf{x}(t),q(t-1)\right) \end{aligned} \tag{2.2}$$

where $\mathbf{x}(t)\in R^n$ is the state vector, $\mathbf{u}(t)\in R^m$ is the input vector, and $q(t)\in\{1,2,...,s\}$ is the discrete/quantized state variable. The hybrid-system state is described at any instant by $\left(\mathbf{x}(t),q(t)\right)\in R^{n+1}$. The local behavior of the system is described by the vectorial function $f_{q(t)}(\bullet)$ and the discrete/quantized state variable is determined by the function $g(\bullet)$.

In this chapter a modified version of the Witsenhausen hybrid system is considered, where the discrete/quantized state variable depends only on the state vector $\mathbf{x}(t)$ and does not depend on the previous discrete/quantized state $q(t-1)$. So, (2.2) can be written as:

$$\begin{aligned} \mathbf{x}(t+1) &= \sum_{i=1}^{s} f_i\left(\mathbf{x}(t),\mathbf{u}(t)\right)\delta_i\left(\mathbf{x}(t)\right) \\ \delta_i\left(\mathbf{x}(t)\right) &= \begin{cases} 1, & g\left(\mathbf{x}(t)\right)=i \Leftrightarrow \mathbf{x}(t)\in\chi_i \\ 0, & otherwise \end{cases} \\ \mathbf{y}(t) &= h\left(\mathbf{x}(t),\mathbf{u}(t)\right) \end{aligned} \tag{2.3}$$

where $\mathbf{x}(t)$, $\mathbf{u}(t)$, $f_i(\bullet)$ and $g(\bullet)$ are defined in (2.2), $\mathbf{y}(t)\in R^r$ is the output vector determined by the function $h(\bullet)$ and $\delta_i\left(\mathbf{x}(t)\right)$ is a binary variable that equals *1* if $g\left(\mathbf{x}(t)\right)$ equals i and *0* otherwise. The equation $g\left(\mathbf{x}(t)\right)=i$ indicates that the state vector $\mathbf{x}(t)$ belongs to the region of $\chi_i\in R^n$. This kind of systems is a sub-class of the hybrid system given by (2.1).

The aim in this thesis chapter is to present a systematic method for determining the regions χ_i and the functions $f_i(\bullet)$ given only the input-output data of the process. The state-space partition

χ_i will be assumed to be hyper-cubic, and $f_i(\bullet)$ could be a non-linear function that will be identified by the T&S models.

2.2.2. PWA Systems.

PWA systems have been studied by several authors (for example, Sontag, 1981, Bemporad *et al.*, 2000 and their references). As it is stated in Bemporad *et al.* (2000), PWA systems represent the simplest extension of linear systems that still can model non-linear processes and capable of handling with the hybrid behavior.

PWA systems are represented by the following piece-wise linear affine models, whose dynamics are affine and can be different in different regions of the state-input space. They are defined by

$$\begin{cases} \mathbf{x}(t+1) = A_i\mathbf{x}(t) + B_i\mathbf{u}(t) + f_i \\ \mathbf{y}(t) = C_i\mathbf{x}(t) + D_i\mathbf{u}(t) + g_i \\ \text{if} \quad \left[\mathbf{x}(t) \quad \mathbf{u}(t)\right]^T \in \chi_i \quad \Leftrightarrow \quad G_i^x\mathbf{x}(t) + G_i^u\mathbf{u}(t) \leq G_i^C \end{cases} \tag{2.4}$$

where t, $\mathbf{x}(t)$, $\mathbf{u}(t)$ and $\mathbf{y}(t)$ are defined as in (2.1), the sub-index i takes values $1,...,N_{PWA}$, where N_{PWA} is the number of PWA dynamics defined over a polyhedral partition *S*. Every partition χ defines the state-input space over which the different dynamics are active. The dynamics are defined by the matrixes A_i, B_i, C_i, D_i and vectors g_i and f_i. The partitions are defined by hyper-planes given by matrixes G_i^x, G_i^u and G_i^C. The model (2.4) is supposed to be well-posed, and then the partition should satisfy:

$$\begin{aligned} &\chi_i \cap \chi_j = \emptyset, \quad \forall i \neq j, \\ &\bigcup_{i=1}^{N_{PWA}} \chi_i = \chi \end{aligned} \tag{2.5}$$

PWA systems (2.4) belong to the general class (2.1) by choosing functions $f(\bullet)$ and $h(\bullet)$ to be PWA functions (the auxiliary variable $\mathbf{w}(t)$ is not used, as the inequalities defined by the function $g(\bullet)$).

Equations (2.5) imply that the PWA system is well-posed. Then the set of inequalities $G_i^x\mathbf{x}(t)+G_i^u\mathbf{u}(t)\leq G_i^C$ should be split in strict inequalities (<) and non-strict inequalities ($\leq$). For simplicity in the notation this issue will be neglected. Also, because it is not important from the numerical point of view, as continuous systems are considered.

2.2.3. MLD Systems.

Hybrid systems can be modeled using the Mixed Logical and Dynamics (MLD) framework, as a linear system of differential equations and a set of linear inequalities. From the general case in (2.1), when $f(\bullet)$ and $h(\bullet)$ are linear functions and $g(\bullet)$ is an affine linear function, then the linear MLD is obtained as shown in Bemporad and Morari (1999):

$$\begin{cases}\mathbf{x}(t+1)=A\mathbf{x}(t)+B_1\mathbf{u}(t)+B_2\boldsymbol{\delta}(t)+B_3\mathbf{z}(t)\\ \mathbf{y}(t)=C\mathbf{x}(t)+D_1\mathbf{u}(t)+D_2\boldsymbol{\delta}(t)+D_3\mathbf{z}(t)\\ -E_5\leq E_1\mathbf{u}(t)-E_2\boldsymbol{\delta}(t)-E_3\mathbf{z}(t)+E_4\mathbf{x}(t)\end{cases}\tag{2.6}$$

where $t\in\mathbb{Z}$, $\mathbf{x}(t)=\left[x_c^T(t),x_l^T(t)\right]\in\mathrm{R}^{n_c}\times\{0,1\}^{n_l}$ is the state of the system, whose component are distinguished between continuous and binary states, $\mathbf{u}(t)=\left[u_c^T(t),u_l^T(t)\right]\in\mathrm{R}^{m_c}\times\{0,1\}^{m_l}$ are the continuous and binary inputs, $\mathbf{y}(t)=\left[y_c^T(t),y_l^T(t)\right]\in\mathrm{R}^{p_c}\times\{0,1\}^{p_l}$ are the continuous and binary outputs, and $\boldsymbol{\delta}(t)\in\{0,1\}^{r_l},\mathbf{z}(t)\in\mathrm{R}^{r_c}$ represent auxiliary logical and continuous variables. *A*, *B₁*, *B₂*, *B₃*, *C*, *D₁*, *D₂*, *D₃*, *E₁*, *E₂*, *E₃*, *E₄* and *E₅* are matrices that define the model equation and constraints. $\mathbf{w}(t):=\left[\boldsymbol{\delta}(t),\mathbf{z}(t)\right]$ is the auxiliary variables vector.

Clearly (2.6) forms a subclass of (2.1). MLD systems have been used for the modeling of much kind of systems through linear equations and discrete variables and propositional logic statements modeled as mixed-integer linear inequalities (see Bemporad and Morari, 1999). MLD also can model those systems than can be modeled through the hybrid system description language HYSDEL (see Torrisi and Bemporad, 2002).

As it was stated in the literature review, PWA and MLD systems are equivalent (Heemels *et al.*, 2001). There are also more equivalent classes of hybrid dynamical models like Linear Complementarity (LC), Extended Linear Complementary (ELC) and Max-Min Plus Scaling (MMPS) systems. Next, fuzzy modeling is incorporated for the hybrid modeling above described in order to represent also the continuous non-linearities of the most hybrid systems.

2.2.4. Hybrid Fuzzy Systems.

Hybrid fuzzy systems are a sub-class of models belonging to (2.1) where the functions $f(\bullet)$, $h(\bullet)$ and $g(\bullet)$ correspond to fuzzy models. This new class of models permits to include explicitly in the same model the hybrid characteristic of a system (hard transitions) and the fuzzy models features to represent other non-linearities. In this thesis, based on the hybrid model (2.1) a fuzzy model was incorporated locally. As a further research is the analysis of let say fuzzy-hybrid models, were from a fuzzy model, a local hybrid model is incorporated in each fuzzy rule. A hybrid fuzzy system could be written as:

$$\mathbf{x}(t+1)=f_F\left(\mathbf{x}(t),\mathbf{u}(t),\mathbf{w}(t)\right) \tag{2.7.a}$$

$$\mathbf{y}(t)=h_F\left(\mathbf{x}(t),\mathbf{u}(t),\mathbf{w}(t)\right) \tag{2.7.b}$$

$$\mathbf{0}\le g_F\left(\mathbf{x}(t),\mathbf{u}(t),\mathbf{w}(t)\right) \tag{2.7.c}$$

$\mathbf{x}(t)\in R^n$ is the state vector, $\mathbf{u}(t)\in R^m$ is the input vector, $\mathbf{y}(t)\in R^r$ is the output vector and $\mathbf{w}(t)\in R^l$ is a vector of auxiliary variables, $f_F : R^n\times R^m\times R^l \to R^n$, $h_F : R^n\times R^m\times R^l \to R^r$ and $g_F : R^n\times R^m\times R^l \to R^q$ are considered as fuzzy models.

Next, two equivalent sub-classes of hybrid fuzzy systems are presented. The first one, called Piece-Wise Fuzzy (PWF) model, is based on the PWA model with the difference of considering a Takagi & Sugeno (T&S) model instead of a linear function in each sub-region. The second one is called fuzzy MLD (FMLD), and is based on MLD systems, considering T&S models instead of linear dynamic models. Both sub-classes are useful to model systems where different non-linear behaviours occur within different sub-regions, by explicitly including the hybrid feature of the system.

The following sub-classes (PWF and FMLD) assume that the sub-regions generate a well-posed partition of the subspace. Then, for all pairs $(\mathbf{x}(t),\mathbf{u}(t))$ in the input-state space, the equations (2.7) have a solution $(\mathbf{x}(t+1),\mathbf{y}(t),\mathbf{w}(t))$, and moreover, $(\mathbf{x}(t+1),\mathbf{y}(t))$ are uniquely determined.

In addition, it is assumed that the partition is linear-polyhedral (given by the piece-wise functions in PWF or the inequalities in FMLD). The linear assumption could be relaxed for dealing with systems in which non-linear behaviour is determined by non-linear partitions; however the identification procedure for these kinds of systems is out of the scope of this chapter.

2.2.5. Piece-Wise Fuzzy (PWF) Systems.

PWF systems are a new class of hybrid systems defined by the following piece-wise fuzzy function, whose non-linear dynamics can be different in different regions of the state-input space.

They are defined by

$$\begin{cases}\mathbf{x}(t+1)=f_{TS}^{i}(\mathbf{x}(t),\mathbf{u}(t))\\ \mathbf{y}(t)=h_{TS}^{i}(\mathbf{x}(t),\mathbf{u}(t))\\ \text{if}\quad \begin{bmatrix}\mathbf{x}(t) & \mathbf{u}(t)\end{bmatrix}^{T}\in\chi_i \quad\Leftrightarrow\quad G_i^x\mathbf{x}(t)+G_i^u\mathbf{u}(t)\le G_i^C\end{cases} \tag{2.8}$$

where t, $\mathbf{x}(t)$, $\mathbf{u}(t)$ and $\mathbf{y}(t)$ are defined in (2.7), the sub-index i takes values $1,\ldots,N_{PWA}$, where N_{PWA} is the number of PWF dynamics defined over a polyhedral partition χ.

Every partition χ_i defines the state-input space over which the different dynamics are active. The dynamics are defined by the T&S fuzzy models in the state. The partitions are defined by hyper-planes given by matrices G_i^x, G_i^u and G_i^C.

The model (2.8) is supposed to be well-posed, and then the partition should satisfy the same conditions that apply for PWA systems, explained before in (2.5).

PWF systems (2.8) belong to the general class of hybrid fuzzy systems (2.7). In these systems, functions $f_F(\bullet)$ and $h_F(\bullet)$ are chosen as T&S models (the auxiliary variable $\mathbf{w}(t)$ is not used, neither the inequalities defined by the function $g_F(\bullet)$).

2.2.6. Fuzzy MLD (FMLD) Systems.

Hybrid fuzzy systems can be modeled using the FMLD framework. From the general case of hybrid fuzzy systems in (2.7), the FMLD considers a T&S for modeling the dynamic transitions and for the output, and linear affine functions for the set of inequalities. The FMLD is defined as follows:

$$\begin{cases} \mathbf{x}(t+1) = f_{TS}\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\delta}(t), \mathbf{z}(t)\right) \\ \mathbf{y}(t) = h_{TS}\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\delta}(t), \mathbf{z}(t)\right) \\ -E_5 \le E_1\mathbf{u}(t) - E_2\boldsymbol{\delta}(t) - E_3\mathbf{z}(t) + E_4\mathbf{x}(t) \end{cases} \tag{2.9}$$

where $t \in \mathbb{Z}$, $\mathbf{x}(t) = \left[x_c^T(t), x_l^T(t)\right] \in \mathrm{R}^{n_c} \times \{0,1\}^{n_l}$ is the state of the system, whose component are distinguished between continuous and binary states, $\mathbf{u}(t) = \left[u_c^T(t), u_l^T(t)\right] \in \mathrm{R}^{m_c} \times \{0,1\}^{m_l}$ are the continuous and binary inputs, $\mathbf{y}(t) = \left[y_c^T(t), y_l^T(t)\right] \in \mathrm{R}^{p_c} \times \{0,1\}^{p_l}$ are the continuous and binary outputs, and $\boldsymbol{\delta}(t) \in \{0,1\}^{r_l}, \mathbf{z}(t) \in \mathrm{R}^{r_c}$ represent auxiliary logical and continuous variables.

$f_{TS}(\bullet)$ is the T&S fuzzy model that determines the state equation, $h_{TS}(\bullet)$ is the T&S fuzzy model for the output, *E₁*, *E₂*, *E₃*, *E₄* and *E₅* define the output equation and inequalities. $\mathbf{w}(t) := \left[\boldsymbol{\delta}(t), \mathbf{z}(t)\right]$ is the auxiliary variables vector.

2.2.7. Equivalence between PWF and FMLD Systems.

Assuming that FMLD is well-posed, as defined before, then for a given $\mathbf{x}(t)$ and $\mathbf{u}(t)$, $\boldsymbol{\delta}(t)$, $\mathbf{z}(t)$, $\mathbf{y}(t)$ and $\mathbf{x}(t+1)$ are uniquely defined. Next, we prove a PWF system is equivalent to a FMLD system.

First the transformation from a PWF system to a FMLD system is analyzed. Given a PWF system in the form of (2.8), one transformation to a FMLD in the form of (2.9) could be performed by including a crisp membership function in the fuzzy rules in the following way:

$$\begin{aligned}
&\mathbf{x}(t+1)=\sum_{i=1}^{N_{PWF}} f_{TS}^{i}\left(\mathbf{x}(t),\mathbf{u}(t)\right)\delta_i\left(\mathbf{x}(t),\mathbf{u}(t)\right)=f_{TS}\left(\mathbf{x}(t),\mathbf{u}(t)\right)\\
&\delta_i\left(\mathbf{x}(t),\mathbf{u}(t)\right)=\begin{cases}1 & if \quad G_i^x\mathbf{x}(t)+G_i^u\mathbf{u}(t)\le G_i^C\\ 0 & otherwise\end{cases}\\
&\mathbf{y}(t)=\sum_{i=1}^{N_{PWF}} h_{TS}^{i}\left(\mathbf{x}(t),\mathbf{u}(t)\right)\delta_i\left(\mathbf{x}(t),\mathbf{u}(t)\right)=h_{TS}\left(\mathbf{x}(t),\mathbf{u}(t)\right)
\end{aligned} \tag{2.10}$$

where $\delta_i\left(\mathbf{x}(t),\mathbf{u}(t)\right)$ represents an extra membership function of the fuzzy model, which activates the rules associated with the T&S model of the region, namely $G_i^x\mathbf{x}(t)+G_i^u\mathbf{u}(t)\le G_i^C$. When a PWF model is written in the form of (2.10), PWF is called Modified Tanaka Model (MTM), corresponding to a version of the Tanaka Model described in Tanaka *et al.* (2001).

The transformation from FMLD to PWF is more complicated. It requires first that the system is well-posed; thus, given $\mathbf{x}(t)$ and $\mathbf{u}(t)$, then $\boldsymbol{\delta}(t)$ and $\mathbf{z}(t)$ are uniquely defined. From inequality (2.9) $-E_5\le E_1\mathbf{u}(t)-E_2\boldsymbol{\delta}(t)-E_3\mathbf{z}(t)+E_4\mathbf{x}(t)$ it is possible to obtain a unique value for $\boldsymbol{\delta}(t)$ and $\mathbf{z}(t)$ (Bemporad and Morari, 1999). So, as the inequality (from where $\boldsymbol{\delta}(t)$ and $\mathbf{z}(t)$ are obtained) is linear, it is possible to state:

$$\begin{aligned}
\boldsymbol{\delta}(t)&=A_1\mathbf{x}(t)+B_1\mathbf{u}(t)+c_1\\
\mathbf{z}(t)&=A_2\mathbf{x}(t)+B_2\mathbf{u}(t)+c_2
\end{aligned} \tag{2.11}$$

Then, considering $\boldsymbol{\delta}(t)$ and $\mathbf{z}(t)$ as premises of the $f_{TS}(\bullet)$, it corresponds to a PWF system with one region χ (N_{PWA}=1).

$$\begin{cases}
\mathbf{x}(t+1)=f_{TS}\left(\mathbf{x}(t),\mathbf{u}(t),A_1\mathbf{x}(t)+B_1\mathbf{u}(t)+c_1,A_2\mathbf{x}(t)+B_2\mathbf{u}(t)+c_2\right)=f_{TS}^{1}\left(\mathbf{x}(t),\mathbf{u}(t)\right)\\
\mathbf{y}(t)=h_{TS}\left(\mathbf{x}(t),\mathbf{u}(t),A_1\mathbf{x}(t)+B_1\mathbf{u}(t)+c_1,A_2\mathbf{x}(t)+B_2\mathbf{u}(t)+c_2\right)=h_{TS}^{1}\left(\mathbf{x}(t),\mathbf{u}(t)\right)\\
\text{if}\quad \left[\mathbf{x}(t)\quad \mathbf{u}(t)\right]^T\in\chi
\end{cases} \tag{2.12}$$

In the following sections, the focus will be on the problem of identification of hybrid fuzzy systems using PWF models. In the approach, based on fuzzy clustering and principal component analysis, it is assumed that the sub-regions are cubic, so they are defined in the following way: $\left[\mathbf{x}(t) \quad \mathbf{u}(t)\right]^T \in \chi_i \quad \Leftrightarrow \quad H_i\left[\mathbf{x}(t) \quad \mathbf{u}(t)\right]^T \leq h_i$, where H_i is a diagonal matrix. This assumption could be relaxed for the analysis of more complex hybrid models as mentioned in the discussion section 2.5.

2.3. Piece-Wise Affine Model Identification.

Many works in the literature have proposed sophisticated PWA model identification method (see for example Ferrari-Trecate *et al.*, 2003; Nakada *et al.,* 2005, among others). However, when the proper identification of a system requires a big amount of data (like in many real-processes), those methods are inefficient. In this thesis, for the identification of PWA models (2.4), a fast algorithm based on fuzzy clustering is proposed.

The fuzzy C-means (FCM) method proposed by Bezdek (1973) is a data clustering technique where each data point belongs to a cluster with a unique degree of membership. In other words, the FCM shows how to split the space into a specific number of representative clusters. The FCM considers fuzzy partitioning, such that a data point on the space can belong to more than one cluster, but with different degree of membership (which varies from 0 to 1). FCM is an iterative algorithm that allows the modeler finding cluster centres (centroids) that minimize the following objective function

$$S(c) = \sum_{k=1}^{n} \sum_{i=1}^{c} (\mu_{ik})^m \left\| x_k - v_i \right\|^2 \tag{2.13}$$

where n is the number of data-samples, c is the number of clusters, u_{ik} is the fuzzy partition between 0 and 1, v_i represents the center of cluster i and m є $[1,\infty]$ is a weighting factor. The details of the fuzzy C-means algorithm are found in Babuska (1999).

For the identification of PWA models (2.4), the following fast algorithm based on FCM is proposed:

Step 1. Choose the number of partitions N_{PWA} of the state-output space *S*. This number equals the number of linear models that the PWA model will have. The optimal number of linear models could be obtained by a sensitivity analysis.

Step 2. Estimate all the state measurement required, using the input-output data available. If the state is unknown, to choose proper regressors for the output and input signals, and to propose a state space model.

Step 3. In the state-output space, perform a Fuzzy C-Means (FCM), with the number of clusters equals to N_{PWA}. In this step, it is important to normalize the data before FCM.

Step 4. Build the partition based on the membership function value of each cluster. A datum will belong to the cluster with a higher membership function value. Data in the border of the clusters are used to obtain the hyper-planes that better separate the clusters. The data on the borders usually have membership function values around 0.4 to 0.6; however, this will depend in the geometry of the clusters.

Step 5. For every cluster, using the data with a membership function equal or higher than 0.7 (tuning parameter), identify the linear model parameters by LMS. It is important not to consider the data in the borders in the LMS. Computational experiments showed that data on the borders could lead to locally unstable models, even for stable plants.

Next, a batch reactor is presented and used to show the proposed method. A scheme of the batch reactor is shown in Figure 2.1. The reactor's core (temperature T) is heated or cooled through the reactor's water jacket (temperature T_w). The heating medium in the water jacket is a mixture of fresh input water, which enters the reactor through on/off valves, and reflux water. The water is pumped into the water jacket with a constant flow Φ. The dynamics of the system depend on the physical properties of the batch reactor, i.e., the mass *m* and the specific heat capacity *c* of the ingredients in the reactor's core and in the reactor's water jacket (here, the index *w* denotes the water jacket). λ is the thermal conductivity, *S* is the contact area and T_0 is the temperature of the surroundings.

The temperature of the fresh input water T_{in} depends on two inputs: the positions of the on/off valves k_H and k_C. However, there are two possible operating modes of the on/off valves. When $k_C = 1$ and $k_H = 0$, the input water is cool ($T_{in} = T_C = 12^\circ C$), whereas if $k_C = 0$ and $k_H = 1$, the input water is hot ($T_{in} = T_C = 75^\circ C$).

The ratio of fresh input water to reflux water is controlled by the third input, i.e., by the position of the mixing valve k_M. There are six possible ratios that can be set by the mixing valve. The share of fresh input water can be either 0, 0.01, 0.02, 0.05, 0.1 or 1.

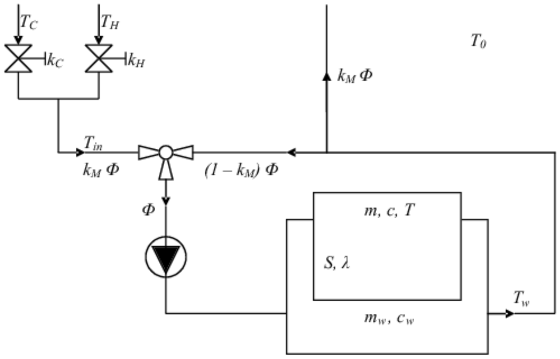


Figure 2.1 Scheme of the batch reactor.

Therefore the batch reactor is a multivariable system with three discrete inputs (k_M, k and k) and two measurable outputs (T and T). Due to the nature of the system, the time constant of the temperature in the water jacket is obviously much shorter than the time constant of the temperature in the reactor's core. Therefore, the batch reactor is considered as a stiff system.

Based on input-output data of the batch reactor, a Piece-Wise Affine model is identified and compared with a fuzzy model in terms of *N*-step-ahead prediction error. The obtained PWA model will be used for the Hybrid Predictive Control of the batch reactor in chapter 3.

A good model for the Temperature in the core (T) is given by:

$$T(t+1) = 0.9967\ T(t) + 0.0033\ T_w(t) \tag{2.13}$$

Then, the aim is to obtain a good model for the Temperature in the water jacket $T_w\left(t+1\right)$. The identification data including the temperature in the core, the temperature in the water jacket, the cold/hot water valve and the mixing valve, is shown in Figure 2.2.

The data is clustered considering first the 2 possible inputs for cold/hot water valve if $u_{kC}\left(t\right)=1$ or $u_{kC}\left(t\right)=0$, and then for both data-set, a fuzzy clustering method (FCM) is used to obtain 6 sub-cluster, where the regressors are $T_w\left(t\right)$, $T\left(t\right)$ and $u_{kM}\left(t\right)$. Then 12 linear models are obtained.

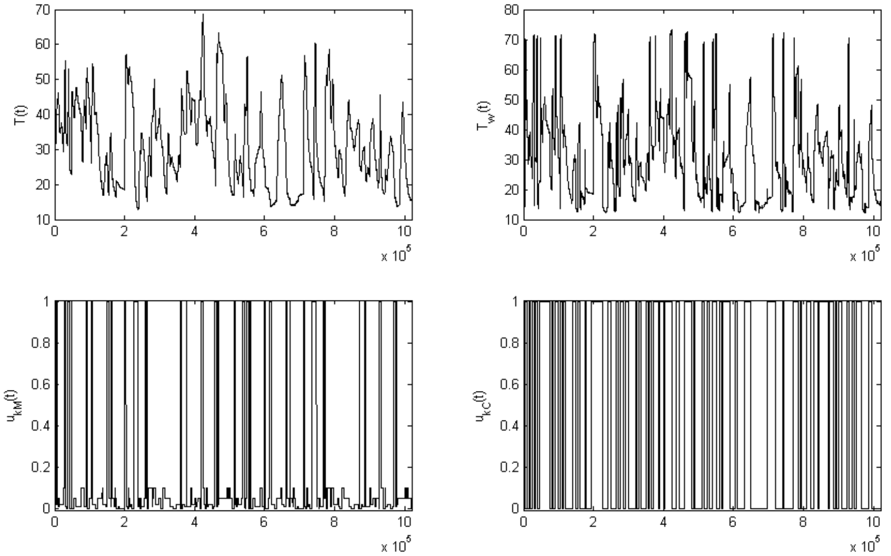


Figure 2.2 Identification Data.

Figures 2.3 and 2.4 show the clustered data. Borders determine the partition. For the partition generation, based on the Figures 2.3 and 2.4, the state-input space is divided with planes in six regions (Polyhedral partition). The planes are chosen in a way that the most representative data of each cluster (in different colors) belongs to one of the six polyhedral regions

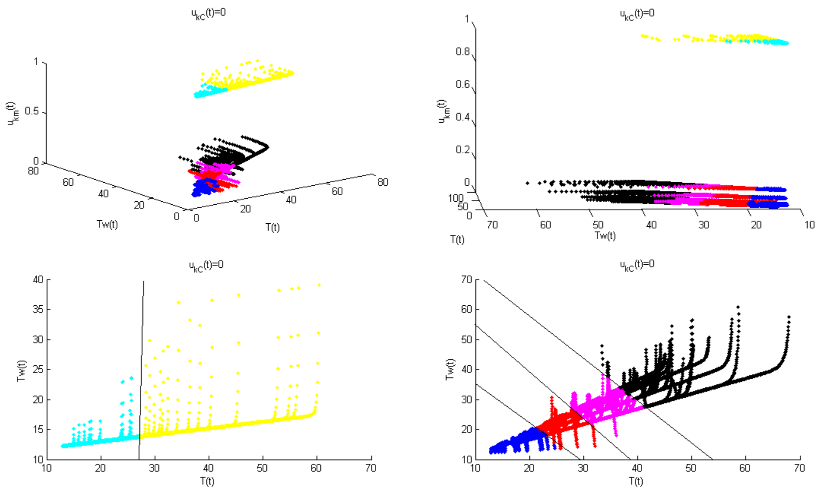


Figure 2.3 Clusters (FCM) when $u_{kC}(t)=0$.

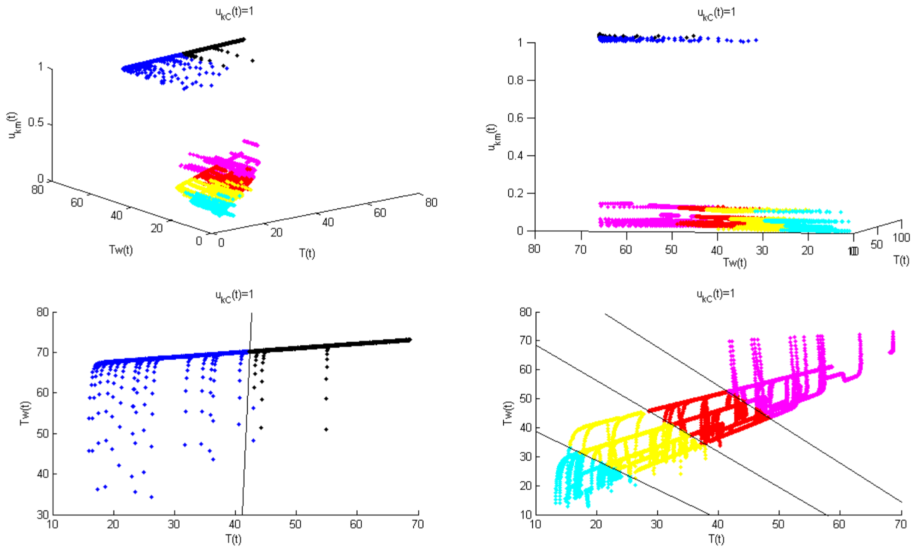


Figure 2.4 Clusters (FCM) when $u_{kC}(t)=1$.

The regions are defined in a way that every data belongs just to one of the twelve regions. The polyhedral partition, generated according Figures 2.3 and 2.4) is the following:

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{01}\Leftrightarrow\begin{cases} u_{Kc}(t)=0\\ u_{Km}(t)=1\\ T_w(t)\leq 1.8750T(t)+7.3447\end{cases}\tag{2.14a}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{02}\Leftrightarrow\begin{cases}u_{Kc}(t)=0\\u_{Km}(t)=1\\T_w(t)>1.8750T(t)+7.3447\end{cases}\tag{2.14b}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{03}\Leftrightarrow\begin{cases}u_{Kc}(t)=0\\u_{Km}(t)<1\\T_w(t)\leq\text{-}1.3617T(t)+48.5957\end{cases}\tag{2.14c}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{04}\Leftrightarrow\begin{cases}u_{Kc}(t)=0\\u_{Km}(t)<1\\T_w(t)>\text{-}1.3617T(t)+48.5957\\T_w(t)\leq\text{-}1.3514T(t)+64.7027\end{cases}\tag{2.14d}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{05}\Leftrightarrow\begin{cases}u_{Kc}(t)=0\\u_{Km}(t)<1\\T_w(t)>\text{-}1.3514T(t)+64.7027\\T_w(t)\leq\text{-}1.5217T(t)+90.5\end{cases}\tag{2.14e}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{06}\Leftrightarrow\begin{cases}u_{Kc}(t)=0\\u_{Km}(t)<1\\T_w(t)>\text{-}1.5217T(t)+90.5\end{cases}\tag{2.14f}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{11}\Leftrightarrow\begin{cases}u_{Kc}(t)=1\\u_{Km}(t)=1\\T_w(t)\leq\text{-}4.6800T(t)+265.6240\end{cases}\tag{2.14g}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{12}\Leftrightarrow\begin{cases}u_{Kc}(t)=1\\u_{Km}(t)=1\\T_w(t)>\text{-}4.6800T(t)+265.6240\end{cases}\tag{2.14h}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{13}\Leftrightarrow\begin{cases}u_{Kc}(t)=1\\u_{Km}(t)<1\\T_w(t)\leq\text{-}0.9146T(t)+47.3232\end{cases}\tag{2.14i}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{14}\Leftrightarrow\begin{cases}u_{Kc}(t)=1\\u_{Km}(t)<1\\T_w(t)>\text{-}0.9146T(t)+47.3232\\T_w(t)\leq\text{-}1.049T(t)+73.8382\end{cases}\tag{2.14j}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{15}\Leftrightarrow\begin{cases}u_{Kc}(t)=1\\u_{Km}(t)<1\\T_w(t)>\text{-}1.049T(t)+73.8382\\T_w(t)\leq\text{-}1.049T(t)+103.5972\end{cases}\tag{2.14k}$$

$$\left(T_w(t),T(t),u_{Kc}(t),u_{Km}(t)\right)\in S_{16}\Leftrightarrow\begin{cases}u_{Kc}(t)=0\\u_{Km}(t)<1\\T_w(t)>\text{-}1.049T(t)+103.5972\end{cases}\tag{2.14l}$$

Then, in every partition, 12 linear model is obtained for the temperature in the water jacket. As the data in the border of the region is not representative, only the data with a membership function greater than 0.8 is considered for obtaining the linear models. Let $x(t)=\left[T(t),T_w(t)\right]^T$ be the state vector of the batch reactor, $y(t)=\left[T(t),T_w(t)\right]^T$ the output and $u(t)=\left[u_{Kc}(t),u_{Km}(t)\right]^T$ the input vector at instant *k*. Then, the PWA model obtained has the following form:

$$\begin{cases}x(t+1)=A_{ij}x(t)+B_{ij}u(t)+f_{ij}\\y(t)=C_{ij}x(t)+D_{ij}u(t)+g_{ij}\\\text{if}\quad\left[x(t)\quad u(t)\right]^T\in S_{ij}\end{cases},\quad i\in\{0,1\},j=1,...,6.\tag{2.15}$$

where S_{ij} , $i\in\{0,1\},j=1,...,6$, are the polyhedral partition defined in (2.14), $C_{ij}=\begin{bmatrix}1&0\\0&1\end{bmatrix}$, $D_{ij}=\begin{bmatrix}0&0\\0&0\end{bmatrix}$ and $g_{ij}=\begin{bmatrix}0\\0\end{bmatrix}$ $\forall i\in\{0,1\},j=1,...,6$, and $A_{01}=\begin{bmatrix}0.9967&0.0033\\0.0333&0.6278\end{bmatrix}$,

$A_{02}=\begin{bmatrix}0.9967&0.0033\\0.0373&0.6492\end{bmatrix}$, $A_{03}=\begin{bmatrix}0.9967&0.0033\\0.0413&0.9349\end{bmatrix}$, $A_{04}=\begin{bmatrix}0.9967&0.0033\\0.0395&0.9386\end{bmatrix}$,

$A_{05}=\begin{bmatrix}0.9967&0.0033\\0.0439&0.9253\end{bmatrix}$, $A_{06}=\begin{bmatrix}0.9967&0.0033\\0.0279&0.9364\end{bmatrix}$,

$A_{11}=\begin{bmatrix}0.9967&0.0033\\0.0306&0.6236\end{bmatrix}$, $A_{12}=\begin{bmatrix}0.9967&0.0033\\0.0352&0.6601\end{bmatrix}$, $A_{13}=\begin{bmatrix}0.9967&0.0033\\0.0625&0.9104\end{bmatrix}$,

$A_{14}=\begin{bmatrix}0.9967&0.0033\\0.0276&0.9512\end{bmatrix}$, $A_{15}=\begin{bmatrix}0.9967&0.0033\\0.0420&0.9323\end{bmatrix}$, $A_{16}=\begin{bmatrix}0.9967&0.0033\\0.0416&0.9304\end{bmatrix}$,

$$B_{01}=\begin{bmatrix}0 & 0\\ 0 & 2.1600\end{bmatrix},\quad B_{02}=\begin{bmatrix}0 & 0\\ 0 & 1.9091\end{bmatrix},\quad B_{03}=\begin{bmatrix}0 & 0\\ 0 & -1.0636\end{bmatrix},\quad B_{04}=\begin{bmatrix}0 & 0\\ 0 & -3.4927\end{bmatrix},$$

$$B_{05}=\begin{bmatrix}0 & 0\\ 0 & -6.1274\end{bmatrix},\quad B_{06}=\begin{bmatrix}0 & 0\\ 0 & -6.2327\end{bmatrix},\quad B_{11}=\begin{bmatrix}0 & 0\\ 0 & 12.4974\end{bmatrix},\quad B_{12}=\begin{bmatrix}0 & 0\\ 0 & 11.1938\end{bmatrix},$$

$$B_{13}=\begin{bmatrix}0 & 0\\ 0 & 15.8199\end{bmatrix},\quad B_{14}=\begin{bmatrix}0 & 0\\ 0 & 9.5677\end{bmatrix},\quad B_{15}=\begin{bmatrix}0 & 0\\ 0 & 11.0815\end{bmatrix},\quad B_{16}=\begin{bmatrix}0 & 0\\ 0 & 6.6972\end{bmatrix},$$

$$f_{01}=\begin{bmatrix}0\\ 2.1600\end{bmatrix},\quad f_{02}=\begin{bmatrix}0\\ 1.9091\end{bmatrix},\quad f_{03}=\begin{bmatrix}0\\ 0.3846\end{bmatrix},\quad f_{04}=\begin{bmatrix}0\\ 0.4712\end{bmatrix},\quad f_{05}=\begin{bmatrix}0\\ 0.8079\end{bmatrix},$$

$$f_{06}=\begin{bmatrix}0\\ 1.2346\end{bmatrix},\quad f_{11}=\begin{bmatrix}0\\ 12.4974\end{bmatrix},\quad f_{12}=\begin{bmatrix}0\\ 11.1938\end{bmatrix},\quad f_{13}=\begin{bmatrix}0\\ 0.4924\end{bmatrix},\quad f_{14}=\begin{bmatrix}0\\ 0.5796\end{bmatrix},$$

$$f_{15}=\begin{bmatrix}0\\ 0.8629\end{bmatrix}, f_{16}=\begin{bmatrix}0\\ 1.2052\end{bmatrix}.$$

Now, the PWA model is compared with the Fuzzy Model reported in Karer *et al.* (2007). Models are compared using the following data for validation shown in Figure 2.5.

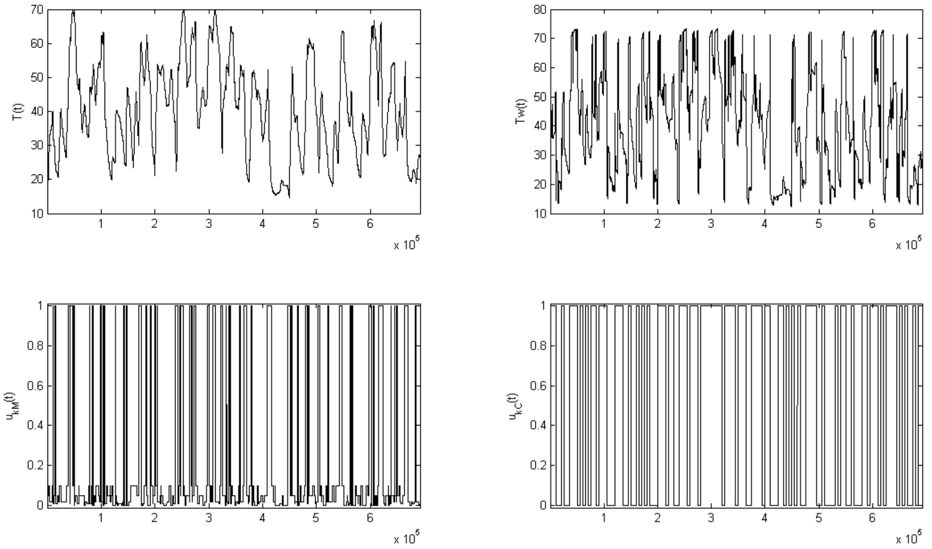


Figure 2.5 Validation Data.

Figure 2.6 shows the *N*-step-ahead (for the controller, i.e, 15 times *N* predictions) versus the prediction error of each model. The *N*-Step-ahead prediction error is higher for the PWA model than for the fuzzy model. In table 2.1 are the values for some prediction error.

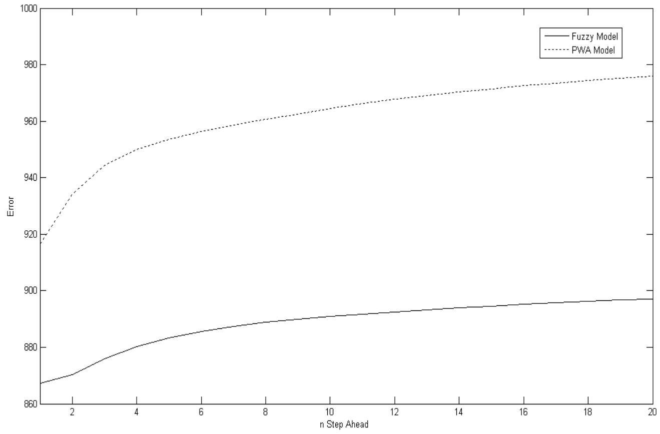


Figure 2.6 *N*-Step-ahead prediction error.

Table 2.1 *N*-step-ahead prediction error

Prediction horizon	PWA model	Fuzzy Model
N=1	916.6983	867.2423
N=5	953.6297	883.2466
N=10	964.3984	890.8699
N=15	970.2901	893.8734
N=20	975.9365	897.0687

As further research, in the identification procedure of the PWA model, it could be possible to generalize the partition method using the membership degree of membership given by FCM. In terms of computational time; this method is faster than the Hybrid Identification Toolbox (HIT) when it processes similar amount of data. Moreover, the HIT Toolbox cannot handle data like the provided by the batch reactor as it is not well distributed and it generate problems with the covariance matrices.

2.4. Hybrid Fuzzy Model Identification

The Witsenhausen system given by (2.2) can be represented by a two-level fuzzy model, which was described by Tanaka *et al.* (2001). Then, the expression (2.3) can be written as a Modified Tanaka Model (MTM), where the corresponding two levels are the local fuzzy level and the discrete/quantized level. The local fuzzy level is a set of T&S fuzzy models with local validity in one region of an estimated hyper-cubic partition $\bar{\chi}_i$, $i=1,...,\bar{s}$. The discrete/quantized level is given by a set of crisp weighting functions $\delta_i\left(\mathbf{x}(t-1)\right)$, which activates the *i*-th local T&S model if the state $\mathbf{x}(t-1)$ is within $\bar{\chi}_i$.

Let us assume that the input-output data is available, and that from the output it is possible to estimate the state vector $\mathbf{x}(t)$. The structure of the two-level fuzzy model (MTM) to be identified for the variable $y(t)$ is described in the following way:

$$
\begin{aligned}
y(t) &= \sum_{i=1}^{\bar{s}}\sum_{j=1}^{R_i}\beta_{ij}\left(\mathbf{z}(t-1)\right)\delta_i\left(\mathbf{x}(t-1)\right)\left(\mathbf{a}_{ij}^T\mathbf{x}(t-1)+\mathbf{b}_{ij}^T\mathbf{u}(t-1)+r_{ij}\right)\\
\delta_i\left(\mathbf{x}(t-1)\right) &= \begin{cases}1 & \mathbf{x}(t-1)\in\bar{\chi}_i\\ 0 & otherwise\end{cases}\\
\beta_{ij}\left(\mathbf{z}(t-1)\right) &= \frac{\prod_{r=1}^{p}A_{ij,r}\left(z_r(t-1)\right)}{\sum_{j=1}^{R_i}\prod_{r=1}^{p}A_{ij,r}\left(z_r(t-1)\right)}
\end{aligned}
\tag{2.16}
$$

where $\mathbf{x}(t-1)\in R^n$ is the state vector, $\mathbf{u}(t-1)\in R^m$ is the input vector, $\mathbf{z}(t-1)^T=\left[z_1(t-1),...,z_p(t-1)\right]^T$ is the vector of the premises, *p* is the number of inputs at the premises.

The index *i* represents the *i*-th region, $\mathbf{a}_{ij}^T$, $\mathbf{b}_{ij}^T$, r_{ij} are the fuzzy model parameters for the region *i* on the rule *j*, $\bar{s}$ is the estimated number of regions, R_i is the number of rules of the fuzzy model at the *i*-th region, $\delta_i\left(\mathbf{x}(t-1)\right)$ is a binary variable that selects the current fuzzy model at the *i*-th region, $A_{ij,r}\left(z_r(t-1)\right)$ is the degree of membership for the input $z_r(t-1)$ at the *i*-th region and rule *j*, and $\beta_{ij}\left(\mathbf{z}(t-1)\right)$ is the degree of activation the *j*-th rule that belongs to the fuzzy model of the *i*-th region.

Note that MTM has the same structure as equation (2.10); thus MTM is equivalent to a PWF model.

For example, in a SISO system, in order to obtain the model for the output $y(t)$, $\mathbf{x}(t-1)=\left[y(t-1),...,y(t-n_a)\right]$ should be chosen as the state-space vector and $\mathbf{u}(t-1)=\left[u(t-1),...,u(t-n_b)\right]$ as the input vector as well.

Note also that the MTM given by (2.16) is a fuzzy model with the following rules:

$$\begin{aligned}&R_{ij} : \textbf{if } \mathbf{x}(t-1)\in \bar{\chi}_i \textbf{ and } z_1(t-1)\in A_{ij,1} \textbf{ and} \ldots \textbf{and } z_p(t-1)\in A_{ij,p} \quad \textbf{then}\\ &y_{ij}(t)=\mathbf{a}_{ij}^T\mathbf{x}(t-1)+\mathbf{b}_{ij}^T\mathbf{u}(t-1)+r_{ij}, \quad i=1,...\bar{s},\, j=1,...,R_i.\end{aligned} \tag{2.17}$$

2.4.1. Identification Procedure.

When the transition of a discrete/quantized state is triggered, a sudden change in the data distribution occurs. Thus, an analysis of the cluster slopes using the main components is proposed to identify the switching region where the spatial orientation of the clusters varies abruptly (Torres, 2009).

This method is presented as an inverse form of the merge method of clusters presented in Babuska (1998) and Kaymak and Babuska (1995), where instead of merging similar clusters, the clusters that are very different will be used to define a hard partition of the state space. With the cluster slope and the center of different consecutive clusters the switching points will be detected and the hyper-cubic partition $\bar{\chi}_i$, $i=1,...,\bar{s}$, over the regressor space will be defined.

Only based on the information of the input-output data of the process, the identification problem consists of estimating the parameters of the MTM. Therefore, the number of regions s should be estimated, the partition $\bar{\chi}_i$, $i=1,...,\bar{s}$, each T&S model, the number of rules R_i, the membership functions $A_{ij,r}(\bullet)$ and the parameters $\mathbf{a}_{ij}^T$, $\mathbf{b}_{ij}^T$, r_{ij} should be estimated.

It is assumed that N input/output data associated with the vector $\left[\mathbf{x}(t),\mathbf{u}(t)\right]^T$, have been collected:

$$\Phi = \begin{bmatrix} y(1) & \mathbf{x}(0) & \mathbf{u}(0) \\ y(2) & \mathbf{x}(1) & \mathbf{u}(1) \\ \vdots & \vdots & \vdots \\ y(N) & \mathbf{x}(N-1) & \mathbf{u}(N-1) \end{bmatrix}_{N\times(n+m+1)}$$

N denotes the number of data samples, *y(t)* is the output variable to estimate with the MTM, $\mathbf{x}(t)\in R^n$ is the state vector and $\mathbf{u}(t)\in R^m$ is the input vector.

The identification procedure is as follows:

Step 1: Determine the fuzzy clusters over the data Φ, using the Gustafon-Kessel (GK) algorithm (Gustafson and Kessel, 1979). It is well known that the GK algorithm does not give an indication of the required correct number of clusters.

A large number of clusters will result in a complicated rule-based model, while a small number of clusters results in a poor model. Then, to obtain the optimum number of clusters, the use of the compatible cluster merging method is proposed, as suggested for the identification of the T&S models in Babuska (1998). It is important to preserve the small clusters in the interesting regions, which may have been found when clustering with an initially large number of clusters.

The GK algorithm provides the centers of the clusters $\mathbf{v}_l = \left[v_l^1, v_l^2, ..., v_l^{n+m+1}\right]^T$, the *c* covariance matrices for each fuzzy cluster *l*, with *n+m+1* eigenvectors $\{\varphi_{1,l}, \varphi_{2,l}, ..., \varphi_{n+m+1,l}\}$and with the corresponding *n+m+1* eigenvalues $\{\lambda_{1,l}, \lambda_{2,l}, ..., \lambda_{n+m+1,l}\}$.

Step 2: Select the eigenvector φ_l^* associated with the maximum eigenvalue λ_l^* for each cluster $l = 1,...,c$. $\lambda_l^* = \max\{\lambda_{1,l}, \lambda_{2,l}, ..., \lambda_{n+m+1,l}\}$

For the detection of the switching points it is proposed to analyze the most important eigenvectors (the main vectors or the principal components), towards which directions the maximum amount of information is obtained.

Step 3: For every cluster $l=1,...,c$ and every component of the state-space vector $x_k(t)$, $k=1,...,n$, calculate the vector $\hat{\pi}_{lk}$, which represents the projection of the eigenvector φ_l^* on the subspace given by the inputs and the state-space variable $x_k(t)$. $\hat{\pi}_{lk}$ is given by:

$$\hat{\pi}_{lk} = \frac{\Phi_k \varphi_l^*}{\left\|\Phi_k \varphi_l^*\right\|}, \quad l=1,...,c, \quad k=1,...,n. \tag{2.18}$$

where φ_l^* is the eigenvector chosen in step 2 and Φ_k is the matrix of dimension $(n+m+1)\times(n+m+1)$, whose elements are defined as:

$$(\Phi_k)_{\ell,\wp} = \begin{cases} 1 & if \quad \ell=\wp=k+1 \\ 1 & if \quad \ell=\wp \ and \ \ell>n+1 \\ 0 & if \quad otherwise \end{cases}. \tag{2.19}$$

Note that the vector is normalized, so $\left\|\hat{\pi}_{lk}\right\|=1$.

Step 4: For every vector $\hat{\pi}_{lk}$, determine $\hat{\pi}_{lk}^u$ which represent the projection of $\hat{\pi}_{lk}$ in the subspace generated by the inputs. $\hat{\pi}_{lk}^u$ is obtained in the following way:

$$\hat{\pi}_{lk}^u = \frac{\Phi_u \hat{\pi}_{lk}}{\left\|\Phi_u \hat{\pi}_{lk}\right\|}, \quad l=1,...,c, \quad k=1,...,n, \tag{2.20}$$

where $\hat{\pi}_{lk}$ is the vector obtained in step 3, and Φ_u is the matrix of dimension $(n+m+1)\times(n+m+1)$, whose elements are defined as:

$$(\Phi_u)_{\ell,\wp} = \begin{cases} 1 & if \quad \ell=\wp \ and \ \ell>n+1 \\ 0 & if \quad otherwise \end{cases}. \tag{2.21}$$

Note that the vector is normalized, so $\left\|\hat{\pi}_{lk}^u\right\|=1$.

Let $\hat{\gamma}_{lk}$ be the estimation of the angle between $\hat{\pi}_{lk}$ and $\hat{\pi}_{lk}^{u}$. It is possible to obtain this angle by calculating the $\arccos\left(\hat{\pi}_{lk}^{T}\cdot\hat{\pi}_{lk}^{u}\right)$. Finally, for each cluster l and every state-space variable $x_k\left(t\right)$, compute the cluster slope $\Gamma_{lk}=\tan\left(\hat{\gamma}_{lk}\right)$ given by:

$$\Gamma_{lk}=\sqrt{\frac{1}{\left(\hat{\pi}_{lk}^{T}\cdot\hat{\pi}_{lk}^{u}\right)^{2}}-1},\quad l=1,...,c,\quad k=1,...,n, \tag{2.22}$$

As an example, in Figure 2.2 $\mathbf{x}\left(t-1\right)=y\left(t-1\right)$, $\mathbf{u}\left(t-1\right)=u\left(t-1\right)$ and $\mathbf{x}_1\left(t-1\right)=\left[y\left(t-1\right),u\left(t-1\right)\right]$.

Figure 2.7a) shows the data with the corresponding clusters, and the lines inside the cluster represent the vectors φ_l^{*} associated with the maximum variance for each cluster. Figure 2.7b) shows the projections of the vectors φ_l^{*} over $\mathbf{x}_1$ and the angles $\hat{\gamma}_{lk}$.

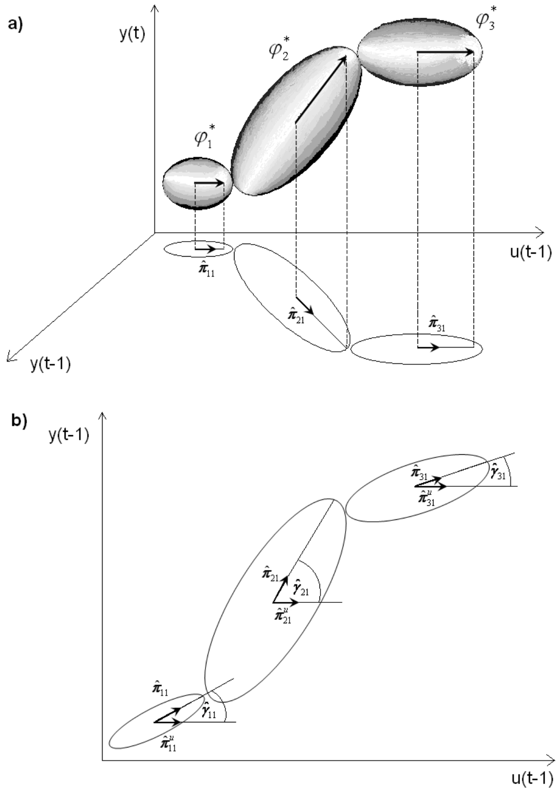


Figure 2.7 a) Principal components, b) Projections of principal components.

Step 5: In this step the idea is to determine the switching point for every state-space variable $x_k(t)$ by obtaining the slope rates among the consecutive clusters given by:

$$\Delta\Gamma_{l_1k} = \left|\Gamma_{l_1k} - \Gamma_{l_2k}\right| \tag{2.23}$$

For obtaining the s_k switching point, if l_1 and l_2 are consecutive clusters (in descending order regarding variable $x_k(t)$), to evaluate the slope rate $\Delta\Gamma_{l_1k}$.

Then, from the cluster centre obtained in step 1, choose the coordinates *k+1* ($v_{l_1}^{k+1}$ and $v_{l_2}^{k+1}$) of the consecutive clusters $\mathbf{v}_{l_1}$ and $\mathbf{v}_{l_2}$ where the slope rate $\Delta\Gamma_{l_1k}$ has a variation greater than the threshold. As this threshold considers the mean value of $\Delta\Gamma_{lk}$ ($\Delta\overline{\Gamma}_k$) plus twice its standard deviation ($\Sigma_{\Delta\Gamma_k}$). Then, if all slope rates are similar, it means that there is not a switching point in the variable $x\ (t)$. Otherwise, just the clusters with a larger variation will be considered.

If the possible number of switching points is known (s_k), then just choose the s_k consecutive clusters with the largest slope rate. The switching points are in between the coordinates $v_{l_1}^{k+1} < v_{l_2}^{k+1}$ of the consecutive clusters. The location of the switching point $V_k^{l_1}$ is estimated in the following way:

$$V_k^{l_1} = \frac{\dfrac{v_{l_1}^{k+1} + \sqrt{\lambda_{l_1}^*}\varphi_{l_1}^{k+1}}{\lambda_{l_1}^*} + \dfrac{v_{l_2}^{k+1} + \sqrt{\lambda_{l_2}^*}\varphi_{l_2}^{k+1}}{\lambda_{l_2}^*}}{\dfrac{1}{\lambda_{l_1}^*} + \dfrac{1}{\lambda_{l_2}^*}} \tag{2.24}$$

where $\lambda_{l_1}^*$ and $\lambda_{l_2}^*$ are the eigenvalues obtained in step 2 corresponding to the clusters l_1 and l_2 respectively and $\varphi_{l_1}^{k+1}$ and $\varphi_{l_2}^{k+1}$ are the coordinates k+1 of the corresponding eigenvectors. The set V_k contains the coordinates of the s_k switching points V_k^l.

$$
\begin{aligned}
V_k &= \left\{ V_k^l \,/\, \Delta\Gamma_{lk} \geq \Delta\bar{\Gamma}_k + 2\Sigma_{\Delta\Gamma_k} \right\} \\
\Delta\bar{\Gamma}_k &= \left(\frac{1}{c-1} \right) \sum_{l=1}^{c-1} \Delta\Gamma_{lk} \qquad , \\
\Sigma_{\Delta\Gamma_k} &= \sqrt{\left(\frac{1}{c-1} \right) \sum_{l=1}^{c-1} \left(\Delta\Gamma_{lk} - \Delta\bar{\Gamma}_k \right)^2 }.
\end{aligned}
\tag{2.25}
$$

Let $\vec{v}_k$ be a vector with the following components: first, the minimum value ($\underline{x_k}$) for the variable $x_k(t)$, then (if there are switching points) the elements of V_k, and, finally, the maximum ($\overline{x_k}$) associated with the variable $x_k(t)$, $k = 1,...,n$. The elements of $\vec{v}_k$ are, in ascending order, thus $v_k^{\alpha_k} < v_k^{\alpha_k+1}$, $\forall \alpha_k = 1,...,s_k + 1$, where s_k is the number of elements of V_k.

Step 6: Generate the partition $\{\bar{\chi}_i\}_{i=1}^{s}$, of the space $\left[\mathbf{x}(t), \mathbf{u}(t)\right]^T$. Each sub-region $\bar{\chi}_i$ is defined as follows:

$$
\bar{\chi}_i = \left\{ \left[\mathbf{x}(t-1), \mathbf{u}(t-1)\right]^T / H_i \left[\mathbf{x}(t-1), \mathbf{u}(t-1)\right]^T \prec h_i \right\}. \tag{2.26}
$$

The symbol $\prec$ is used to generate a complete partition of the regressor space. H_i and h_i are:

$$
\begin{aligned}
H_i &= \begin{bmatrix} I_{(n+m)\times(n+m)} \\ -I_{(n+m)\times(n+m)} \end{bmatrix} \\
h_i &= \begin{bmatrix} \bar{v}_i & \bar{u}_{\max} & \hat{v}_i & -\bar{u}_{\min} \end{bmatrix}^T \\
\bar{v}_i &= \left[v_1^{\alpha_1+1}, ..., v_k^{\alpha_k+1}, ..., v_n^{\alpha_n+1} \right]^T \\
\bar{u}_{\max} &= \left[\bar{u}_1, \bar{u}_2, ..., \bar{u}_m \right]^T \\
\hat{v}_i &= \left[-v_1^{\alpha_1}, ..., -v_k^{\alpha_k}, ..., -v_n^{\alpha_n} \right]^T \\
\bar{u}_{\min} &= \left[\underline{u}_1, \underline{u}_2, ...\underline{u}_m \right]^T
\end{aligned}
\tag{2.27}
$$

where I is the identity matrix; $\bar{u}_{\max}$ and $\bar{u}_{\min}$ are vectors with the maximum and minimum values of the inputs.

The index *i* is a function of the indexes $\alpha_1,...,\alpha_k,...,\alpha_n$ obtained in the Step 5, ($\alpha_k = 1,...,s_k + 1$); it is used to enumerate the combinations of the elements of the vectors $\vec{v}_k$ to generate the partition.

Step 7: For each sub-region $\overline{\chi}_i$, a local T&S model is identified. Each T&S model is optimized for the number of fuzzy clusters and their regressor structure is obtained by a sensitivity analysis, as in Hadjili and Wertz (2002), Nefti *et al.* (2008) and Sáez and Cipriano (2001).

For the identification of the T&S models, it is recommended to use the approach described in Karer *et al.* (2007), which turns out to be suitable because in the previous steps a partition of the state-space variable was obtained, necessary for applying this identification procedure. Then, for every partition $\overline{\chi}_i$, just considering the data that belongs to the sub-region, the number of rules R_i and the membership functions $A_{ij,r}(\bullet)$ are obtained with a clustering method (GK). The idea of the approach is to identify directly the consequent parameters of each rule of the T&S model by weighting the data for the corresponding activation degree of each rule. It is reported in Karer *et al.* (2007) that due to better conditioning the matrices obtained when separating the data belonging to different regions, compared to the conditioning of the whole data matrix, this approach leads to a better estimate of the hybrid fuzzy parameters. In other words, the variances of the estimated parameters are smaller compared to the classic approach. Let us write all the consequent parameters for the fuzzy rule *j* in the region *i* as follows:

$$\Theta_{ij} = \begin{bmatrix} \mathbf{a}_{ij} \\ \mathbf{b}_{ij} \\ r_{ij} \end{bmatrix}_{(n+m+1)\times 1} \qquad (2.28)$$

The model parameters for the rule *j* of region *i* can be obtained using the least-squares identification method as follows:

$$\Theta_{ij} = \left(\Psi_{ij}^T \Psi_{ij}\right)^{-1} \Psi_{ij}^T \mathbf{Y}_{ij} \qquad (2.29)$$

where the matrices Ψ_{ij} and $\mathbf{Y}_{ij}$ are the following:

$$\Psi_{ij} = \begin{bmatrix} \beta_{ij}\left(\mathbf{z}(0)\right)\left[\mathbf{x}^T(0) \quad \mathbf{u}^T(0) \quad 1\right] \\ \beta_{ij}\left(\mathbf{z}(1)\right)\left[\mathbf{x}^T(1) \quad \mathbf{u}^T(1) \quad 1\right] \\ \vdots \\ \beta_{ij}\left(\mathbf{z}\left(N_{ij}-1\right)\right)\left[\mathbf{x}^T\left(N_{ij}-1\right) \quad \mathbf{u}^T\left(N_{ij}-1\right) \quad 1\right] \end{bmatrix}_{N_{ij}\times(n+m+1)}$$

$$\mathbf{Y}_{ij} = \begin{bmatrix} \beta_{ij}\left(\mathbf{z}(0)\right)y(1) \\ \beta_{ij}\left(\mathbf{z}(1)\right)y(2) \\ \vdots \\ \beta_{ij}\left(\mathbf{z}\left(N_{ij}-1\right)\right)y\left(N_{ij}\right) \end{bmatrix}_{N_{ij}\times 1} \tag{2.30}$$

$\mathbf{x}(t-1)$, $\mathbf{u}(t-1)$, $\mathbf{z}(t-1)$, $\beta_{ij}(\bullet)$ and $y(t)$ are defined in (2.16), and N_{ij} is the number of input-output data pairs corresponding to the rule *j* of the region *I* considering only the data that belongs to the region *I* and that $\beta_{ij}\left(\mathbf{z}(t-1)\right)\leq\delta$, with δ a small positive number essential for obtaining suitable conditioned matrices, (Hathaway and Bezdek, 1993).

Finally, the identified MTM could be use for the prediction and analysis of dynamic systems such as demand arrival rate to a stop, or any process with different non-linear behaviour in different regions.

2.4.2. Identification results of a tank system.

Let us consider the hybrid tank system shown in Figure 2.8, similar to that utilized in Gegundez *et al.* (2008). In the figure, *A* is the cross-section of the tank, *S* is the cross-section of the outlet hole, *g* is the acceleration due to gravity, *Q* is the input flow and *h* is the level of the tank. The hybrid tank system is divided into two regions because the cross-section of the tank is larger when the level is higher than 0.3[m].

In this example, for a fixed input flow, more time will be needed for increasing the level when it is higher than 0.3[m] than when it is lower, because the cross-section is larger. This means that the level value 0.3[m] is the switching point in the sense that this level is the border of the two different operating regions, both showing different dynamics. This effect could be detected by looking at the signals $h(t)$ and $Q(t)$, as shown in Figure 2.9; however, in more complex systems this could be very difficult to do.

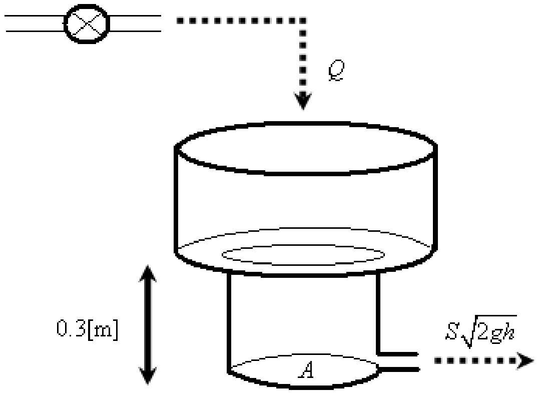


Figure 2.8 Tank System.

Next, the detection of the switching point is proposed by analyzing the principal component of the clusters' variance matrices, provided by the GK algorithm. As shown in Figure 2.10, the effect of the switching point in the example is that the directions of the main components are different when comparing consecutive clusters belonging to the two different regions ($h(t) > 0.3$ and $h(t) \leq 0.3$).

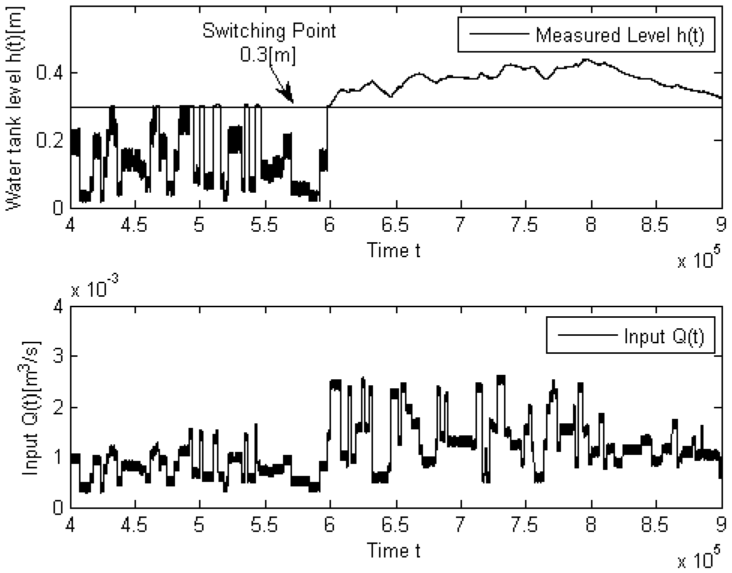


Figure 2.9 Input-Output signals and switching point.

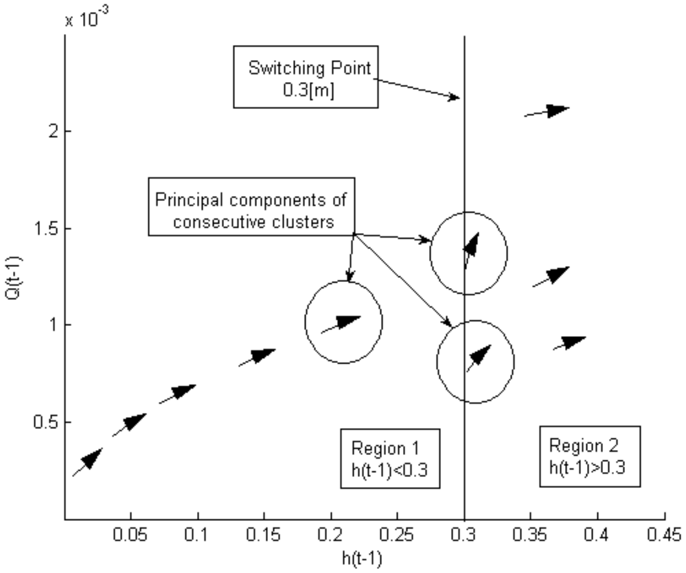


Figure 2.10 Definition of switching regions.

The following non-linear equations describe the dynamics of the tank system:

$$\frac{dh}{dt}=\begin{cases}\frac{1}{A}\left(Q(t)-S\sqrt{2gh(t)}\right) & if \;\; h(t)<0.3\\ \frac{1}{3A}\left(Q(t)-S\sqrt{2gh(t)}\right) & if \;\; h(t)\geq 0.3\end{cases}, \tag{2.31}$$

where $h(t)$ is the level of the tank, $u(t)=Q$ is the input flow, *A*=0.0154 is the cross-section of the tank, *S*=0.0005 is the cross-section of the outlet hole and *g*=9.81 is the acceleration due to gravity.

The hybrid tank system is divided into two regions because the cross-section of the tank is three times longer when the level is higher than 0.3. Assume that just the input-output data shown in Figure 2.11 are available for the training, test and validation.

The identification problem is to find the relation between $h(t)$ and $Q(t)$ considering the input/output data. The main goal is to find the number of switching regions and the switching point (in this case $h(t)=0.3$), which defines the partition. The input/output data considered are $x(t-1)=h(t-1)$ as the output and $u(t-1)=Q(t-1)$ as the input.

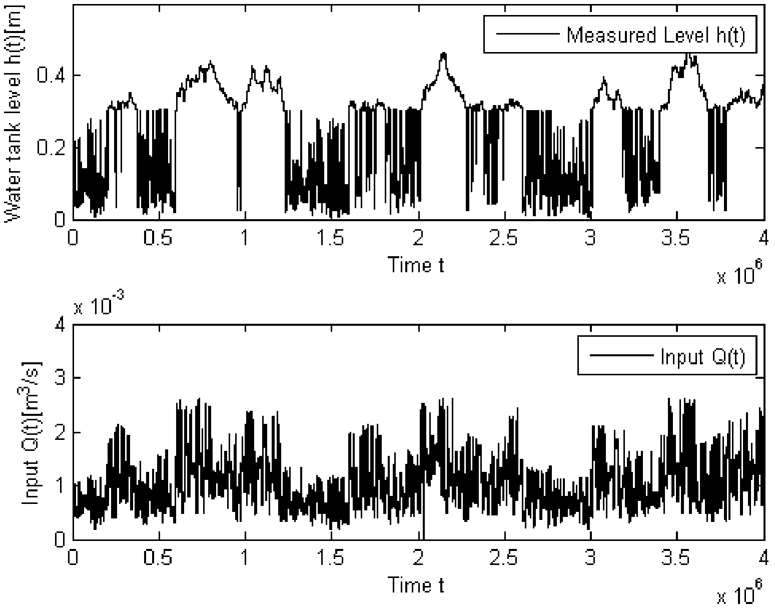


Figure 2.11 Input/output data.

In order to evaluate the performance of both the MTM and the T&S models, the Root Mean Squared (RMS) error is used. The signals were sampled with T_s =10[s]. A total of 100,000 samples were used as the training set, 100,000 as the test set and 50,000 as the validation set.

Next, T&S and MTM modelling results are described and compared.

2.4.3. T&S Model Results.

The GK algorithm was used to obtain the clusters. The T&S model is obtained for a different, increasing number of clusters (sensitivity analysis). The number of clusters obtained from the sensitivity analysis was ten. The T&S model is given by:

$$R_j : \textbf{if } x(t-1) \in A_{j,1} \textbf{ and } u(t-1) \in A_{j,2} \quad \textbf{then}$$
$$x(t) = a_{j1}\mathbf{x}(t-1) + b_{j1}u(t-1) + r_j, \quad j = 1,...,10.$$

where $A_{j,l}\left(z_l(t-1)\right) = e^{-0.5\left(c_{1,j,l}\left(z_l(t-1)-c_{2,j,l}\right)\right)^2}$.

The premises were obtained from the GK algorithm with the normalized data. The consequent parameters were obtained using the method proposed in Karer *et al.* (2007), explained in Step 7, just to be fair in the comparison with the MTM identification.

The parameters of the premises and the consequences of the T&S model are summarized in Table 2.2. Note that rules 1 and 9 are unstable as $\left|a_{j1}\right| > 1$.

Table 2.2. Parameters of T&S model

Rules j	$c_{1,j,1}$	$c_{2,j,1}$	$c_{1,j,2}$	$c_{2,j,2}$	a_{j1}	b_{j1}	r_j
1	6.7676	0.3864	127.4043	0.0010	1.0024	0.3347	-0.0014
2	0.8687	0.1039	992.5746	0.0003	0.9912	3.5180	-0.0012
3	5.3484	0.3730	161.2102	0.0010	1.0000	0.5347	-0.0008
4	1.5217	0.3169	566.6053	0.0008	0.9936	1.7366	-0.0002
5	1.9047	0.3368	452.6675	0.0009	0.9942	1.6730	-0.0003
6	2.7924	0.3219	308.7721	0.0008	0.9952	1.4675	-0.0004
7	1.6739	0.3220	515.1062	0.0008	0.9939	1.4369	0.0001
8	2.8313	0.0459	304.527	0.0001	0.9884	3.5940	-0.0009
9	5.7361	0.4207	150.3129	0.0011	1.0017	0.3440	-0.0012
10	0.9107	0.1965	946.7726	0.0005	0.9917	2.7907	-0.0006

Figure 2.12 presents the T&S output for the one-step-ahead prediction and the measured output using the validation set. Figure 2.13 shows the T&S output for the infinite-step-ahead prediction and the measured output, using the validation-data set.

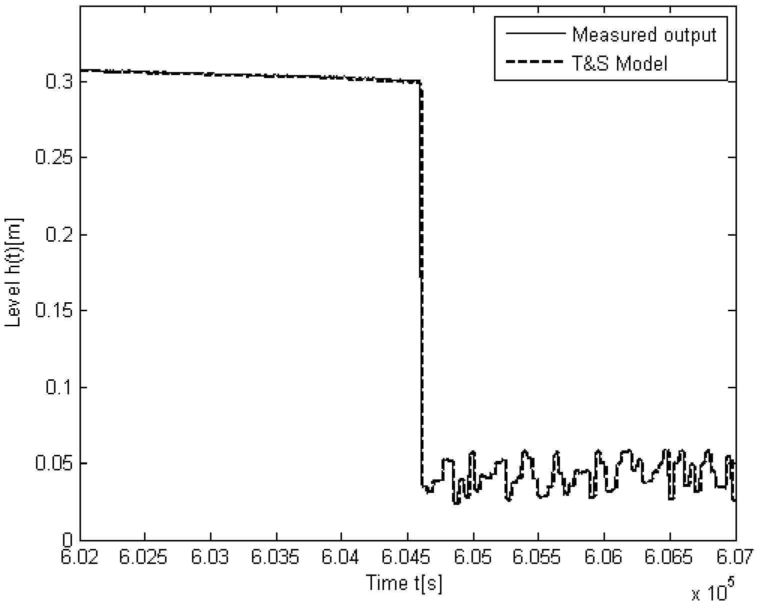


Figure 2.12 Measured output and T&S output, one-step-ahead.

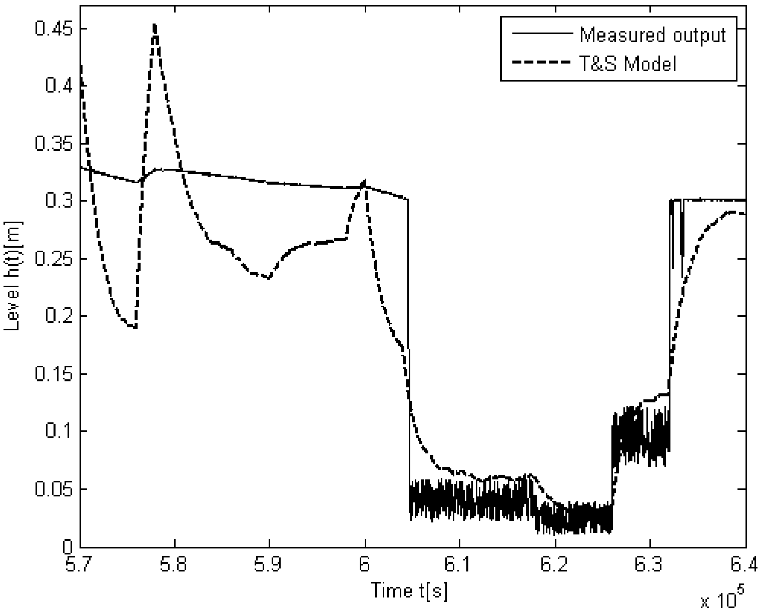


Figure 2.13 Measured output and T&S output, infinite-step-ahead.

2.4.4. Modified Tanaka Model (MTM) Results.

Step 1: The same procedure, based on the GK algorithm used for the T&S model in order to get the optimum number of clusters, is performed. Ten clusters were obtained.

Step 2: From step 1. using the covariance matrixes, given by the GK algorithm, the eigenvalues and the eigenvectors associated with each cluster were determined. Each cluster has 3 eigenvalues, and 3 eigenvectors.

Then, the eigenvector (φ_l^*) associated with the largest eigenvalue for each cluster is considered. So, ten eigenvectors associated with each one of the ten clusters were chosen. Figure 2.14, shows the data and the resulting principal eigenvectors for each cluster.

Step 3: The projection of the eigenvectors obtained from step 2 are determined. The eigenvectors are projected in the space: $\mathbf{x}_1 = \left[x(t-1), u(t-1)\right]$, and Figure 2.10 shows the vectors.

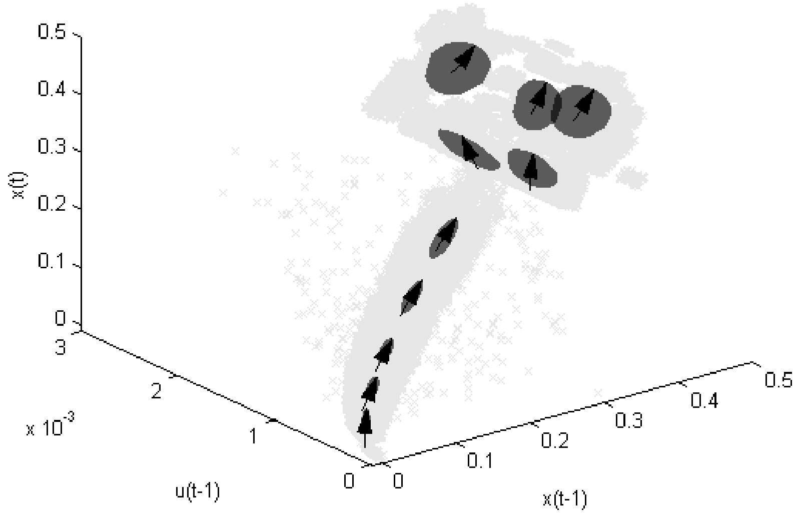


Figure 2.14 Data and principal eigenvectors for each cluster.

Step 4: The slopes of the projected eigenvectors with respect to the coordinate $u(t-1)$ were computed using (2.22). Figure 2.15 shows the slopes associated with the centre of each cluster in the coordinate $x(t-1)$.

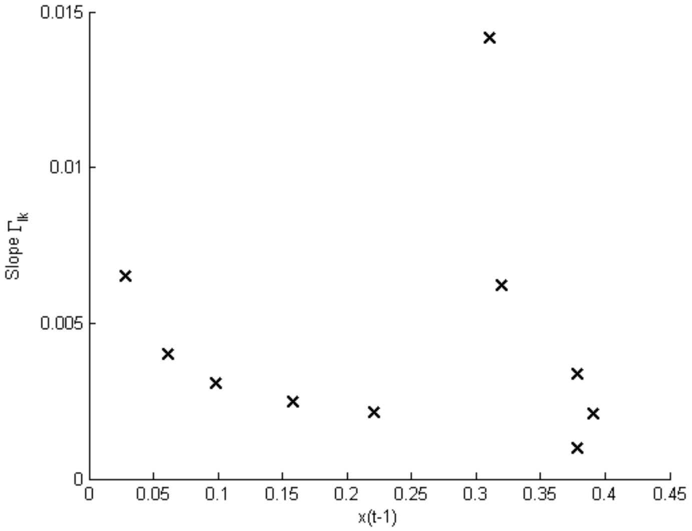


Figure 2.15 Slopes of projected eigenvectors.

Step 5: Figure 2.16 shows the slope rates. Each slope rate is associated with the centre of each cluster in the coordinate $x(t-1)$. The threshold level was set to 0.0133, which equals the average of the slope rate $\Delta\Gamma_{lk}$ ($\Delta\overline{\Gamma}_k = 0.0034$) plus two times the standard deviation ($\Sigma_{\Delta\Gamma_k} = 0.0040$).

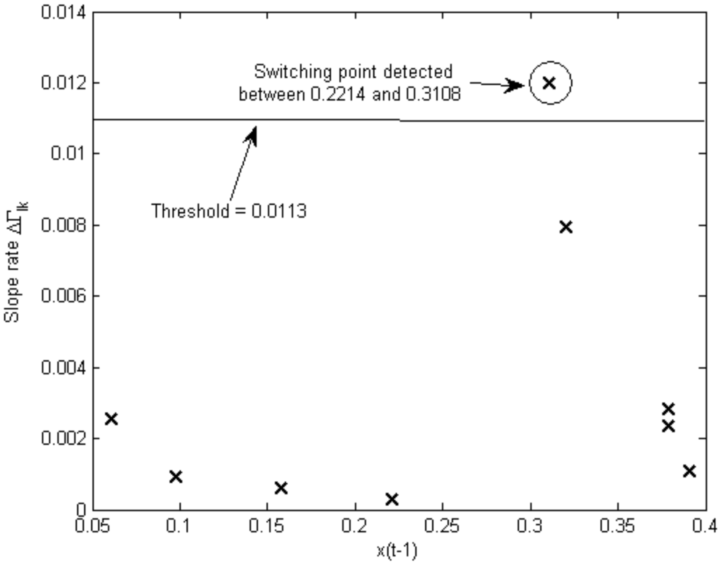


Figure 2.16 Slopes rates.

Based on Figure 2.16, a switching point is detected between the centre of clusters $x(t-1)=0.2214$ and $x(t-1)=0.3108$. Then, using (2.24), the switching point is estimated to be in $x(t-1)=0.2959$ (the real value is 0.3).

Step 6: The partition in the space, considering the estimated switching point $x(t-1)=0.2959$ is generated. There are two subregions ($\bar{s}=2$): the first one when $x(t-1)>0.2959$ and the second when $x(t-1)<0.2959$. Let us set $\underline{x}=0$, $\bar{x}=1$, $\underline{u}=0$ and $\bar{u}=1$. Then the sub-regions of the partition $\bar{\chi}_1$ and $\bar{\chi}_2$ are defined as:

$$\bar{\chi}_1=\left\{\begin{bmatrix}x(t-1)\\u(t-1)\end{bmatrix}/\begin{bmatrix}1&0\\0&1\\-1&0\\0&-1\end{bmatrix}\begin{bmatrix}x(t-1)\\u(t-1)\end{bmatrix}\prec\begin{bmatrix}1\\1\\-0.2959\\0\end{bmatrix}\right\}. \tag{2.32}$$

$$\bar{\chi}_2=\left\{\begin{bmatrix}x(t-1)\\u(t-1)\end{bmatrix}/\begin{bmatrix}1&0\\0&1\\-1&0\\0&-1\end{bmatrix}\begin{bmatrix}x(t-1)\\u(t-1)\end{bmatrix}\prec\begin{bmatrix}0.2959\\1\\0\\0\end{bmatrix}\right\}. \tag{2.33}$$

Step 7: Using the proposed identification method, two local T&S models for the corresponding two switching regions are computed, optimizing the number of clusters per region.

Three rules for region 1 and seven rules for region 2 are used, so the results will be comparable with the 10 rules T&S model obtained. Finally, the structure of the MTM is given by:

$$R_{1j}:\textbf{if }\left[x(t-1),u(t-1)\right]^T\in\bar{\chi}_1\textbf{ and }x(t-1)\in A_{1j,1}\textbf{ and }u(t-1)\in A_{1j,2}\quad\textbf{then}$$
$$y(t)=a_{1j1}x(t-1)+b_{1j1}u(t-1)+r_{1j},\quad j=1,\dots,3.$$

$$R_{2j} : \textbf{if } \left[x(t-1),u(t-1)\right]^T \in \bar{\chi}_2 \textbf{ and } x(t-1)\in A_{2j,1} \textbf{ and } u(t-1)\in A_{2j,2} \quad \textbf{then}$$
$$y(t)=a_{2j1}x(t-1)+b_{2j1}u(t-1)+r_{2j}, \quad j=1,...,7.$$

where $A_{ij,r}\left(z_r\left(t-1\right)\right)=e^{-0.5\left(c_{1,ij,r}\left(z_r\left(t-1\right)-c_{2,ij,r}\right)\right)^2}$.

The parameters for the MTM are given in Table 2.3. The partitions $\bar{\chi}_1$ and $\bar{\chi}_2$ are given by (2.32) and (2.33). Note that the models for $\bar{\chi}_1$ are very similar, meaning that the data is almost lineal in that zone.

Figure 2.17 shows the MTM outputs obtained, for the one-step-ahead prediction and compared with the measured output. Figure 2.18 shows the MTM outputs for the infinite-step-ahead prediction.

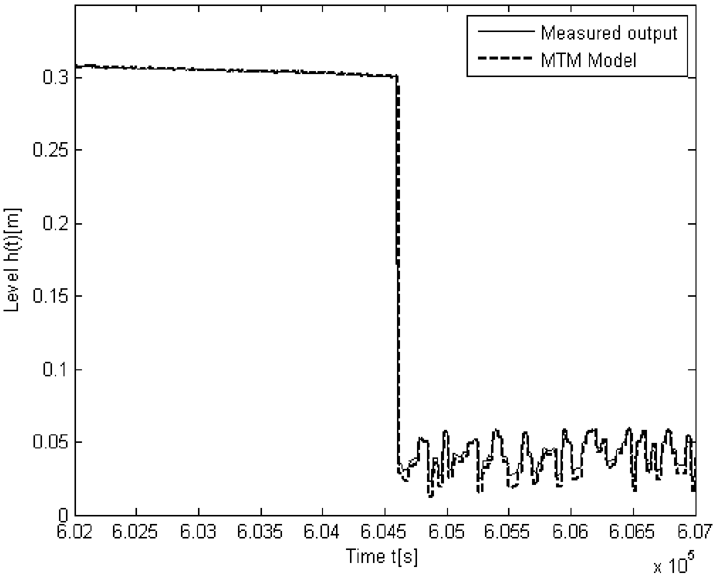


Figure 2.17 Measured output and MTM output, one-step-ahead prediction.

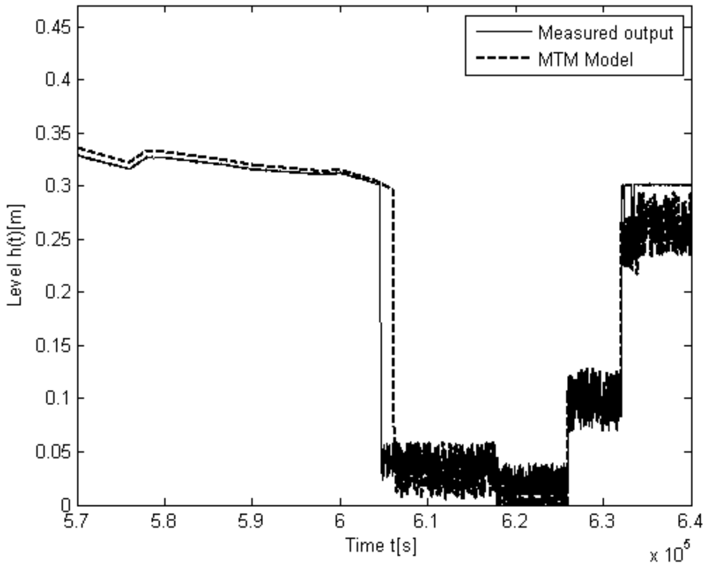


Figure 2.18 Measured output and MTM output, infinite-step-ahead prediction.

Table 2.3. Parameters of MTM.

$\overline{\chi}_1$	$c_{1,1j,1}$	$c_{2,1j,1}$	$c_{1,1j,2}$	$c_{2,1j,2}$	a_{1j1}	b_{1j1}	r_{1j}
j=1	4.5949	0.4047	713.0957	0.0013	0.9999	0.0528	0.0298e-03
j=2	4.3666	0.3212	750.3884	0.0011	0.9999	0.0527	0.0305e-03
j=3	6.5704	0.3576	498.7000	0.0012	0.9999	0.0527	0.0311e-03
$\overline{\chi}_2$	$c_{1,2j,1}$	$c_{2,2j,1}$	$c_{1,2j,2}$	$c_{2,2j,2}$	a_{2j1}	b_{2j1}	r_{2j}
j=1	10.6975	0.2259	159.7337	0.0005	0.3731	171.0725	-0.0534
j=2	5.6316	0.0351	303.4206	0.0001	0.4726	145.0107	-0.0445
j=3	10.7204	0.0689	159.3931	0.0001	0.3397	205.2806	-0.074
j=4	9.9211	0.1090	172.2351	0.0002	0.3875	173.0473	-0.0558
j=5	9.5639	0.1681	178.6678	0.0003	0.2957	244.0445	-0.0997
j=6	8.5983	0.1090	198.7310	0.0002	0.3560	206.6365	-0.0766
j=7	8.3488	0.1681	204.6696	0.0003	0.4240	154.9728	-0.0470

2.4.5. Analysis of Results.

From Figure 2.16, the switching point was detected in $h(t-1)=0.2959$. These results demonstrate that the proposed method can detect this kind of non-linearity. In the case of $h(t-1)$, the real switching point was set to 0.3[m], which is a fairly good estimation.

From Figure 2.13, the T&S model becomes useless for an infinite-step-ahead prediction as the local models of T&S do not consider the switching point. On the other hand, for an infinite-step-ahead prediction, the performance of the MTM is much better than T&S (from Figure 2.18).

Table 2.4 contains the RMS errors divided by the number of data, for the MTM and T&S models, considering the validation data set for one, 100, 200, 300, 400, 500 and 600 step-ahead predictions. Figure 2.19 shows the RMS errors divided by the number of data, for the MTM and T&S models, considering the test data set, in the function of *N*-step-ahead.

Table 2.4. RMS error, T&S and MTM Validation data.

Steps	T&S	MTM
N=1	0.00001864	0.00002442
N=100	0.00373161	0.00008595
N=200	0.00961309	0.00016285
N=300	0.01359353	0.00024115
N=400	0.01556864	0.00032666
N=500	0.01636838	0.00041540
N=600	0.01663034	0.00049908

As shown in Table 2.4 and Figure 2.19, the MTM provides better estimations than T&S when comparing the *N*-step-ahead predictions.

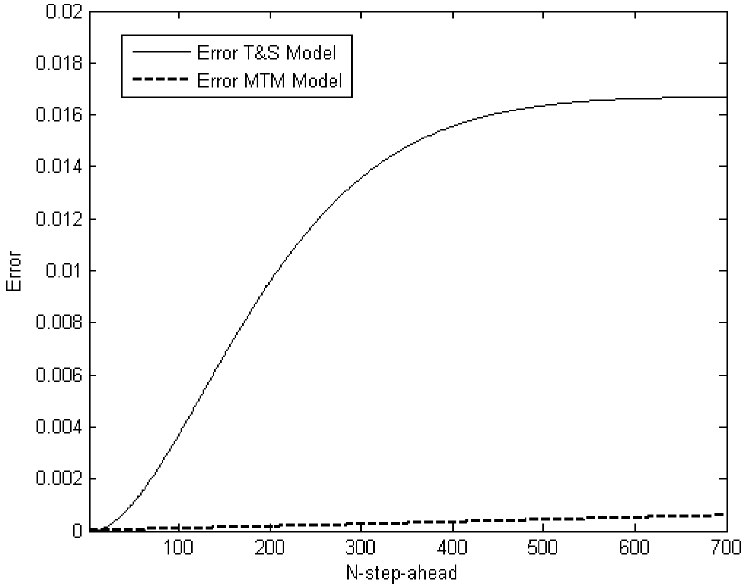


Figure 2.19 RMS divided by the number of data for *N*-step-ahead prediction.

2.5. Conclusions.

This chapter presents new approaches for the identification of non-linear systems with mixed integer and continuous states and inputs. The key element of the hybrid system identification methods is the detection and estimation of the switching regions. In the case of the PWA-model, the identification is conducted based on an ad-hoc fuzzy clustering method and in the case of the hybrid fuzzy-model identification; it is performed by a combination of fuzzy clustering and principal eigenvector analysis.

In identification of MTM, a two-level fuzzy model is identified, which consist on a local fuzzy level and the discrete/quantized level. Thus, MTM incorporates explicitly the hybrid behaviour. Moreover, the method was implemented and applied to a tank-system benchmark problem. The detection of the switching points (discrete transitions) was successfully demonstrated for this system. The use of the main component was not only demonstrated to be very useful in the detection of switching points but also efficient in terms of the computation time as no expensive optimization process was included. The comparisons demonstrated the better performance of the fuzzy hybrid model MTMs with respect to the conventional T&S model when comparing the *N*-step-ahead prediction performance.

In summary, the main contribution of this chapter is the new class of hybrid systems, called fuzzy hybrid system, a fast identification method for PWA systems and the identification method for a class of fuzzy hybrid systems using principal component analysis and fuzzy clustering in fuzzy modelling.

Future work will be focused on generalizing the methodology of fuzzy identification for hybrid non-linear systems. In further research, new approaches of fuzzy hybrid modelling will be analyzed such as a fuzzy clustering that generates both the fuzzy and hard partitions. The stability issues of the proposed fuzzy hybrid modelling will be also studied. Also many transport systems applications could be solved with this method, from demand predictions, traffic, user behaviour, etc.

Also, a new class of model could be analized by including hybrid models (PWA, MLD, etc) into the rules of a fuzzy model.

2.6. References.

Abonyi, J., Babuska, R., Szeifert, F., (2002). “Modified Gath-Geva Fuzzy Clustering for Identification of Takagi-Sugeno Fuzzy Models”. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 32, pp. 612-621.

Babuska, R. (1998). “Fuzzy Modelling for Control”. KAP.

Bemporad, A., Morari, M. (1999). “Control of Systems Integrating Logic, Dynamics and Constraints”. Automatica, vol. 35, pp. 407-427.

Bemporad, A., Ferrari-Trecate, G., Morari, M. (2000). “Observability and Controllability of Piecewise Affine and Hybrid Systems”. IEEE Trans. Automatic Control, vol. 45, pp. 1864-1876.

Bemporad, A., Heelms, W.P.M.H., De Schutter, B. (2002). “On Hybrid Systems and Closed-Loop MPC Systems”. IEEE Transactions on Automatic Control, Vol. 47, No. 5.

Bemporad, A., Garulli, A., Paoletti, S., Vicino, A. (2005). “A Bounded-error Approach to Piecewise Affine System Identification”. IEEE Trans on Automatic Control, vol. 50, pp. 1567-1580.

Celikyilmaz A., Burhan, I. (2008). “Enhanced Fuzzy System Models With Improved Fuzzy Clustering Algorithm”. IEEE Transactions on Fuzzy Systems, vol. 16, pp. 779-794.

Ferrari-Trecate, G., Muselli, M., Liberat, D., Morari, M. (2003). “A Clustering Technique for the Identification of Piecewise Affine Systems”. Automatica, vol. 39, issue 2, pp. 205-217.

Gegundez, M.E., Aroba, J., Bravo, J.M. (2008). “Identification of Piecewise Affine Systems by Means of Fuzzy Clustering and Competitive Learning”. Engineering Applications of Artificial Intelligence, vol. 21, pp. 1321-1329.

Girimonte, D., Babuska, R. (2004). “Structure for Non-linear Models with Mixed Discrete and Continuous Inputs: A Comparative Study”. In: Proc of IEEE International Conf on Systems, Man and Cybernetics, pp. 2392-2397.

Gustafson, D.E., Kessel, W.C. (1979). “Fuzzy Clustering with a Fuzzy Covariance Matrix”. In: Proc IEEE CDC, pages 761-766, San Diego, USA.

Hadjili, M., Wertz, V. (2002). “Takagi-Sugeno Fuzzy Modeling Incorporating Input Variables Selection”. IEEE Transactions on Fuzzy Systems, vol. 10, pp. 728-742.

Hathaway, R.J., Bezdek, J.C. (1993). “Switching Regression Models and Fuzzy Clustering”. IEEE Transactions on Fuzzy Systems, Vol. 1, pp. 195-204.

Heemels, W.P.M.H., De-Schutter, B., Bemporad, A. (2001). "Equivalence of Hybrid Dynamical Models". Automatica, Vol. 37, pp. 1085-1091.

Juloski, A., Weiland, S., Heemels, W.P.M.H. (2005). "A Bayesian Approach to Identification of Hybrid Systems". IEEE Trans on Automatic Control, vol. 50, pp. 1520-1533.

Karer, G., Music, G., Skrjanc, I., Zupancic, B. (2007). "Hybrid Fuzzy Model-based Predictive Control of Temperature in a Batch Reactor". Computers & Chemical Engineering, vol. 31, pp. 1552-1564.

Kaymak, U., Babuska, R. (1995). "Compatible Cluster Merging for Fuzzy Modelling". Proceedings of FUZZ-IEEE/IFES'95, Yokohama, Japan, pp. 897-904.

Kim, E., Park, M., Ji, S., Park, M. (1997). "A New Approach to Fuzzy Modelling". IEEE Transactions on Fuzzy Systems, vol. 5, pp. 328-337.

Ma, Y., Vidal, R. (2005). "A Closed Form Solution to the Identification of Hybrid ARX Models via the Identification of Algebraic Varieties". Hybrid Systems Computation and Control, pages 449 - 465.

Mao, X., Yin, G., Yuan, C., (2007). "Stabilization and Destabilization of Hybrid Systems of Stochastic Differential Equations". Automatica 43, pp. 264-273.

Margaliot, M., (2006). "Stability Analysis of Switched Systems Using Variational Principles: An Introduction". Automatica 42, pp. 2059-2077.

Nakada, H., Takaba, K., Katayama, T. (2005). "Identification of Piecewise Affine Systems Based on Statistical Clustering Technique". Automatica, vol. 41, pp. 905-913.

Nefti, S., Oussalah, M., Kaymak, U. (2008). "A New Fuzzy Set Merging Technique Using Inclusion-based Fuzzy Clustering". IEEE Transactions on Fuzzy Systems, vol. 16, pp. 145-161.

Palm, R., Driankov, D. (1998). "Fuzzy Switched Hybrid Systems -Modeling and Identification". In: Proc of the 1998 IEEE ISCI/CIRA/SAS Joint Conf, Gaithersburg MD, pp. 130-135.

Roubos, H., Setnes, M. (2001). "Compact and Transparent Fuzzy Models and Classifiers through Iterative Complexity Reduction". IEEE Transactions on Fuzzy Systems, vol. 9, pp. 516-524.

Sáez, D., Cipriano, A. (2001). "A New Method for Structure Identification of Fuzzy Models and its Application to a combined Cycle Power Plant". Engineering Intelligent Systems for Electrical Engineering and Communications, vol. 9, pp. 101-107.

Sontag, E.D. (1981). “Non-linear Regulation: the Piecewise Linear Approach”. IEEE Trans Automatic Control, vol. AC-26, pp. 346-357.

Takagi, T., Sugeno, M. (1985). “Fuzzy Identification of Systems and its Applications to Modeling and Control”. IEEE Trans Systems, Man, Cybernetics, vol. 15, pp. 116-132.

Tanaka, K., Iwasaki, M., Wang, H. (2001). “Switching Control of an R/C Hovercraft: Stabilization and Smooth Switching”. IEEE Trans Systems, Man, Cybernetics, vol. 31, pp. 853-863.

Torres, P. (2008). “Diseño e Implementación de una Metodología de Identificación Difusa para Sistemas Híbridos No-lineales (in Spanish)”. Tesis para optar al grado de Magister en Cs. de la Ing. Mención Eléctrica, Universidad de Chile.

Torrisi, F.D., Bemporad, A. (2002). “HYSDEL – A Tool for Generating Computacional Hybrid Models”. ETH, Tech. Report AUTO2-03.

Witsenhausen, H. (1966). “A Class of Hybrid-state Continuous Time Dynamic Systems”. IEEE Trans. on Automatic Control, vol. 11, pp. 161-167.

Yang, Z., and Blanke, M., (2007). “A Unified Approach to Controllability Analysis for Hybrid Control Systems”. Nonlinear Analysis: Hybrid Systems 1, 212-222.

Zeng, X., Goulermas, J., Liatsis, P., Wang, D., Keane, J. (2008). “Hierarchical Fuzzy Systems for Function Approximation on Discrete Input Spaces with Application”. IEEE Trans. on Fuzzy Systems, vol. 16, pp. 1197-1215.

3. Hybrid Predictive Control: Mono-objective and Multi-objective design.

3.1. Literature review.

Different methods for the analysis and design of hybrid systems controllers have emerged over the last few years. Among them, the design of optimal controllers and associated algorithms are the most studied. Next reviews of Hybrid Predictive Control (HPC), considering a mono-objective optimization along with a multi-objective HPC extension are presented.

3.1.1. Hybrid Predictive Control (HPC).

Borrelli *et al.* (2005) provides basic theoretical results on the structure of the optimal solution and on the value function in the optimal control problem of discrete-time linear hybrid systems. The authors describe how the optimal control law can be constructed by combining multi-parametric and dynamic programming. They solve the Hamilton Jabobi Bellman equation by using a simple multi-parametric solver, using their algorithm applied to a wide range of problems. However, the algorithm is limited to linear models and requires a hard computational off-line procedure to synthesize optimal control laws based on the minimization of quadratic and linear performance indexes. Baric *et al.* (2007) present an algorithm for the computation of explicit optimal control laws for Piece-Wise Affine (PWA) systems with polyhedral performance indices, which is an extension of the Borrelli algorithm. Based on dynamic programming, the algorithm improves the efficiency of the off-line procedure by exploiting the geometric structure of the optimization problem.

Many authors have focused on hybrid predictive control and a wide range of applications. For instance, Slupphaug and Foss (1997) and Slupphaug *et al.* (1997) describe a predictive controller with continuous and integer input variables that is solved using non-linear mixed integer programming. It was shown that it performs better than a predictive control strategy with separation of continuous and integer variables. In this case, the proposed algorithms were applied to simulate the control of the level and temperature in a tank system. Bemporad and Morari (2000) and Bemporad *et al.* (2002a) present a predictive control scheme for hybrid systems including operational constraints and is solved using mixed-integer quadratic programming

(MIQP). The proposed algorithm is applied by simulation of a gas system, which incorporates integer-manipulated variables.

The main problem of the MIQP is the computational complexity that increases the time to find the solution. To overcome this problem, Thomas *et al*. (2004) propose a partition of the state space domain. In every partition some variables change while the others remain constant. This approach reduces the computation time. Potočnik *et al*. (2004) propose a hybrid predictive control algorithm with discrete input based on reachability analysis. The computation time is reduced by building and pruning an evolution tree. The algorithms were applied for the optimal control of a multi-product batch plant. All the previous works related to HPC are based on linear models. However, the majority of industrial processes are non-linear in nature. Karer *et al*. (2007) present a suitable optimization algorithm for systems with discrete inputs under a hybrid fuzzy modelling approach. The benefits of the MPC algorithm employing the proposed hybrid fuzzy model were verified on a batch-reactor simulation example and they established that the approach clearly outperforms the approach when a linear model is used.

The application of evolutionary computation techniques for optimization problems with high evaluation cost, like hybrid predictive control problems, is an increasingly important area of research. Although it has been established that evolutionary computation techniques are powerful optimization tools, researchers are facing the challenge of reducing computational cost in problems where the size, complexity and fidelity of the model together with the large number of function evaluations involved in the optimization process produce a very high computational cost. Furthermore, the causes of high computational cost that can be afforded differ widely from one problem to another.

Van der Lee *et al*. (2008) presented a generalized automated tuning algorithm for Model Predictive Controllers (MPCs) combining Genetic Algorithm (GA) with multi-objective fuzzy decision-making. Na and Upadhyaya (2006) applied a combination of MPC, GA optimization and fuzzy identification to the design of the thermoelectric power control. Sarimveis and Bafas (2003) used the GA in fuzzy predictive control without discrete state variables to provide reasonable solutions in a reduced computation time. One of the strong points of the approach is that the feasibility of the optimization solution in each time sample is guaranteed, in contrast to the conventional optimization techniques, which can potentially fail due to the complexity of the optimization problem.

In this chapter the problems of non-linearity and the hybrid nature of a system are tackled by the inherent use of a PWA and a hybrid fuzzy model in HPC. As the optimization of the objective function in the case of the hybrid fuzzy predictive control (HPC) is a highly non-linear problem, the genetic optimization algorithm was employed, similar to the application by Man *et al.* (1998). The problems solved in this chapter are even more complex that the mentioned before because of the discrete states, so that the use of a GA is fully justified as it reduces the computational load substantially.

Regarding the hybrid predictive control strategies, not just a good model is important, also a proper objective function together with an ad-hoc optimization algorithm. In this chapter, a quadratic objective function is used for minimizing the tracking reference error and control effort; however, the objective function could be changed into another more suitable, for example to user costs and operational costs in the context of a dial-a-ride system. Regarding the optimization algorithm, in systems like the dial-a-ride and the integrated transport system, the decisions should be made in a short time, due to any delay in the response could affect dramatically the system costs. Then, the Branch and Bound (BB) and Genetic Algorithms (GA) properties are discussed and the algorithms are compared; then, depending on the computational capacity for controlling a dynamic transport system, the most appropriate algorithm is proposed.

3.1.2. Multi-objective Optimization for Control.

Regarding the application of multi-objective techniques in the context of control, most processes contains multiple and opposite objectives. In the solution of predictive control schemes, classical approaches reduce the multiple objectives into a single objective that minimizes a weighted sum of objectives. However, the determination of these weights is difficult, mainly when the importance of each objective varies with time. Besides, the control law of conventional predictive control is not transparent for the operator in the sense that the trade-off between optimal solutions is not given by the conventional predictive controller. Then, multi-objective seems to be a suitable approach for dealing with predictive control problems.

In the literature, predictive control based on multi-objective optimization was reported under different approaches. Alvarez and Cruz (1998) propose a multi-objective dynamic optimization method for discrete time systems. First, a multi-objective sub-problem is solved with general constraints at each time step. Then, policies that satisfy the necessary optimality conditions for this problem are derived. The priorized policies are used as criteria for choosing the optimal control action. The modelling of discrete time systems is based on state space variables. Numerical results for a continuous binary distillation column are presented. Kerrigan *et al.* (2000) present several methods for handling a large class of multi-objective formulations and priorizations for model predictive control of hybrid systems, using a MLD framework. The methods are flexible and systematic, and use propositional logic and the MLD modelling formalism for prioritizing soft constraints in MPC and guaranteeing the satisfaction of the maximum number of hard constraints.

Next, Kerrigan and Maciejowski (2002) solve the multi-objective predictive control problem based on priorized constraints and objectives. In this case, the most important optimization problem is solved first and the solution to this problem is then used to impose additional constraints on the second optimization. The control action of the predictive controller proposed is obtained using convex programming techniques by considering certain convexity assumptions. Thus, the priorized multi-objective predictive controller can be solved on-line without re-designing off-line the controller; however, this increase in flexibility also demands an increase in the amount of on-line computational power. Núñez-Reyes *et al.* (2002) present a comparison of different multi-objective predictive controllers applied to an olive oil mill. A typical MPC approach based on mono-objective function, a priorized multi-objective predictive controller and structure MPC controller are compared. The last structured MPC, uses a decision list to select the current objective function which must be supplied to the MPC control action. Based on simulation tests, the priorized multi-objective predictive controller gives the best results without the need of tuning weights as the mono-objective MPC. Complex software is required and therefore, a big computational cost is needed. An intermediate solution is the structured MPC. However, abrupt behaviour in the switching between different objectives is observed.

Zambrano and Camacho (2002) describe a multi-objective model predictive control algorithm based on a goal attainment method, which considers the different objective functions as constraints for the minimization of the relaxation variable. This multi-objective predictive controller allows the specification of different goals, such as the economic factor, at different operation points and was applied to a solar refrigeration plant. The results show benefits of including the multi-objective approach. Labidi and Bouani (2004) present a multi-objective control strategy for non linear uncertain dynamic systems modelled by means of a neural network. Non-dominated sorting genetic algorithm (NSGA) is used for solving the multi-objective optimization problem. Each objective function corresponds to the conventional MPC objective function (minimizing the tracking error and the control effort) obtaining predictions with different neural networks models of the system. The criterion for choosing the optimal control action considers taking only the solution that gives the minimum sum of the objective functions.

Flores *et al.* (2005), present the application of fuzzy predictive control to a solar power plant. The proposed predictive controller uses fuzzy characterization of goals and constraints, based on the fuzzy optimization framework for multi-objective satisfaction problems. Subbu *et al.* (2006) present a multi-predictive multi-objective optimization approach for thermal power plants and Hu *et al.* (2007) discuss the development of a dynamic simulation-model-based, considering multi-objective predictive control system for generating cost-effective control strategies for a bioremediation site. Yano and Sakawa (2009) proposed a hierarchical multi-objective programming problem where multiple decision makers in a hierarchical organization have their own multiple objective functions. They proposed an interactive algorithm based on a dual decomposition method to obtain the satisfactory solution, which reflects not only the hierarchical relationships among multiple decision makers but also their own preferences for their objective functions. The proposed algorithm was successfully applied to the industrial pollution control problem in Osaka City in Japan.

Thus, although multi-objective predictive controllers reported are interesting; the systematic tuning methodology design is not complete. Then, in the next section a new approach for dynamic multi-objective hybrid predictive controller that provides generic solutions is proposed.

In section 3.2, Hybrid Predictive Control is presented and the piece-wise-affine models as well as the fuzzy models are highlighted. Optimization algorithms for Hybrid Predictive Control (HPC) are reported and the cases of HPC design based on Branch and Bound (BB) and Genetic Algorithm (GA) are discussed. Simulation results of the control for the two hybrid systems are presented. First a comparison of hybrid fuzzy versus PWA modelling is presented for a Batch Reactor with discrete input. Then a comparison among three optimization algorithms (B&B, EE and GA) is presented and applied for a hybrid tank system. In Section 3.4, Hybrid Predictive

Control based on multi-objective (MO-HPC) optimization is stated. Simulation results of a hybrid tank system are shown. Finally, section 3.5 the MO-HPC is emulated with a HPC.

3.2. Hybrid Predictive Control design.

The PWA and hybrid fuzzy models, both described in chapter 2, are considered for the Hybrid Predictive Control design, where a proper objective function is required. This objective function should represent all the control aims; for example in a regulation problem the tracking error and the control effort should be included, while in the context of a dynamic pick-up and delivery problem for passengers, user and operational costs are opposite goals that must be incorporated as chapters 4 and 5 show. Thus, the controller will obtain future control actions that minimize the objective function.

Next, the cases HPC based on a PWA model and HPC based on a fuzzy hybrid model are highlighted.

3.2.1. Hybrid Predictive Control based on a PWA model (HPC-PWA).

A Hybrid Predictive Controller (HPC) can be designed for minimizing any objective function based on the requirements of a process. For example, the aim of the HPC in the simulation results of this chapter are tracking a reference and minimizing the control effort. For those purposes, a quadratic objective function is usually used as shown in (3.1). Analytically,

$$\begin{aligned} J_t^{t+N} = \sum_{j=0}^{N_u-1} \left\| u(t+j) - u_e(t+j) \right\|_{Q_1}^2 + \sum_{j=1}^{N} \left\| \delta(t+j) - \delta_e(t+j) \right\|_{Q_2}^2 + \left\| z(t+j) - z_e(t+j) \right\|_{Q_3}^2 \\ + \left\| \hat{x}(t+j+1) - x_e(t+j+1) \right\|_{Q_4}^2 + \left\| \hat{y}(t+j) - y_e(t+j) \right\|_{Q_5}^2 \end{aligned} \tag{3.1}$$

Equation (3.1) depends on the vector variables of the inputs $u(t+j)$, the auxiliary variables $\delta(t+j)$ and $z(t+j)$, the estimated state $\hat{x}(t+j+1)$ and the estimated output $\hat{y}(t+j)$ considering a hybrid model, N is the prediction horizon, N_u is the control horizon and $u(t+j)$

is assumed constant for $j \geq N_u$. Based on the rolling horizon procedure, the control action $u(t)$ is applied to the system and in the following sampling time the whole optimization procedure is repeated. u_e, δ_e, z_e, x_e and y_e are vectors whose values are an equilibrium point or the references. The operator $\|\cdot\|_{Q_n}^2$ satisfies for any vector $\vec{h}$ the following: $\|\vec{h}\|_{Q_n}^2 = (\vec{h})^T \cdot Q_n \cdot \vec{h}$. Then, Q_1, Q_2, Q_3, Q_4 and Q_5 are weighing matrices.

Once the optimization problem is solved, the optimal control sequence (3.2) is obtained.

$$\vec{u}^* = \left[u(t)^*, u(t+1)^*, \ldots, u(t+N_u-1)^* \right]^T \tag{3.2}$$

According to the rolling horizon procedure, from (3.2) just the first component $u(t)^*$ is used and applied to the system. Once the control action is applied, the system is conducted to a new state $x(t+1)$ and then, the whole optimization procedure is repeated. As a result, the control action moves the systems variables close to the equilibrium point while considering all the constraints.

The Hybrid Predictive Control based on PWA affine model (HPC-PWA) strategy use the PWA linear affine model to predict the behaviour of the hybrid system by including both discrete/integer and continuous variables. In general, the HPC minimizes the following objective function:

$$\begin{gathered}\min_{\{u(t),u(t+1),\ldots,u(t+N_u-1)\}} J = J_1 + \lambda J_2 \\ J_1 = \sum_{j=N_1}^{N_y} \left(\hat{y}(t+j) - r(t+j)\right)^2, \quad J_2 = \sum_{j=N_1}^{N_u} \Delta u(t+j-1)^2\end{gathered} \tag{3.3}$$

where J is the objective function, $\hat{y}(t+j)$ corresponds to the j-step-ahead prediction for the controlled variable with a PWA model, $r(t+j)$ is the reference, $\Delta u(t+j-1)$ is the increment of the control action, and λ is the weighting factor. N_1, N_y and N_u are the prediction horizons and the control horizon, respectively. The model predictions are given by the PWA linear affine model of the process, i.e.,

$$\hat{y}(t+j)=f_{PWA}\left(y(t+j-1),....,u(t+j-1),....\right) \tag{3.4}$$

where $f_{PWA}(\bullet)$ is the non-linear function defined by a PWA model (2.4), defined in chapter 2. The optimization results in a control sequence $\{u(k),...,u(k+N_u-1)\}$ that minimizes the objective function (3.3). As the HPC problems solved in this chapter includes discrete variables, the optimization could be solved by any Mixed Integer Non-Linear Optimization algorithm (Floudas, 1995).

3.2.2. Hybrid Predictive Control based on Hybrid Fuzzy Models.

In this section, the control of hybrid systems based on hybrid fuzzy models is presented. The Hybrid Predictive Control (HPC) based on a hybrid fuzzy model strategy is a generalization of model-predictive control (MPC), where the prediction model includes both discrete/integer and continuous variables. In general, the HPC minimizes the following objective function:

$$\begin{gathered}\min_{\{u(t),u(t+1),....,u(t+N_u-1)\}} J=J_1+\lambda J_2 \\ J_1=\sum_{j=N_1}^{N_y}\left(\hat{y}(t+j)-r(t+j)\right)^2, \quad J_2=\sum_{j=N_1}^{N_u}\Delta u(t+j-1)^2\end{gathered} \tag{3.5}$$

where J is the objective function, $\hat{y}(t+j)$ corresponds to the j-step-ahead prediction for the controlled variable, $r(t+j)$ is the reference, $\Delta u(t+j-1)$ is the increment of the control action, and λ is the weighting factor. N_1, N_y and N_u are the prediction horizons and the control horizon, respectively. The model predictions are given by the hybrid fuzzy model of the process, i.e.,

$$\hat{y}(t+j)=f_{fuzzy}\left(y(t+j-1),....,u(t+j-1),....\right) \tag{3.6}$$

where $f_{fuzzy}(\bullet)$ is the non-linear function defined by the fuzzy model in (2.16). The optimization results in a control sequence $\{u(t),...,u(t+N_u-1)\}$.

As it is assumed that the HPC problem includes discrete variables, the optimization could be solved by explicitly evaluating all the possible feasible solutions (EE), Branch & Bound (BB) and other algorithms shown in Floudas (1995). Next, in section 3.2.3, an efficient optimizer based on GA is presented in detail. Experimental results of the hybrid fuzzy identification and control of a Hybrid Tank System are shown in section 3.2.4.

3.2.3. Optimization algorithms for Hybrid Predictive Control.

In general, as a Hybrid Predictive Control problem incorporates discrete/integer variables in the model, at every instant, a constrained mixed integer programming problem has to be solved. As stated in Bemporad and Morari (1999), mixed integer programming problems are usually NP-complete, which means that in the worst case, the solution time grows exponentially with the problem size. As a consequence, the application of HPC for solving large scale systems is an interesting research topic. Several algorithms have been proposed an applied for large size application; however they usually do not reach the global optimum. For a detailed description of this fact and also mixed integer programming algorithms see Raman and Grossmann (1991) or Floudas (1995).

Floudas (1995) classified the mixed integer optimization algorithms into four major classes. The first one is cutting plane methods, where the feasible domain in reduced adding new constraints (or “cuts”) to the optimization problem, until an optimal solution is found. The decomposition methods exploit the mathematical structure of the optimization problems by analysing partitioning, duality, and applying relaxation methods. The logic-based methods utilize symbolic inference techniques, which can be expressed in terms of binary variables. In the branch and bound (BB) methods, the possible solutions are explored through a tree of decisions by partitioning the feasible region and generating upper and lower bounds used to avoid (branch) the enumeration of all the possible solutions.

As HPC have to solve a NP-Hard optimization problem at every instant, within the sampling time; it could happen in medium and large scale problem that the application of traditional optimization techniques cannot guarantee even the calculation of a feasible solution. This could happen due to the complexity of the optimization problem, as reported in Sarimveis and Bafas (2003). Then, heuristic methods have emerged for solving NP-Hard problems, which could incorporate previous knowledge of the problems and fast methods for finding good solutions close to optimality within the sampling time.

Among the heuristic methods, which are typically developed for solving particular problems, the evolutionary algorithms based optimization methods (Man *et al.* 1998) are quite utilized. Specifically, Genetic Algorithms (GA) for solving HPC problems are analyzed, as GA is general enough for including HPC features in the algorithm and due to their capability of solving complex non-linear constrained optimization problems.

There are many publications that use GA and consider constraints in optimization problems. Back *et al*. (2000), Coello (2002); Michalewicz (1995) report excellent reviews and methods, but a general methodology has not been proposed so far. One of the most important methods is GENOCOP proposed by Michalewicz (1995b), who developed this genetic algorithm-based program for constrained and unconstrained optimization. Recent work has shown promise results for a Feasible-Infeasible Two-Population (FI-2Pop) genetic algorithm for constrained optimization (Kimbrough *et al*., 2008). The FI-2Pop GA has proved to be better than standard methods for handling constraints in GAs; inclusive it has regularly produced better decisions for comparable computational effort than GENOCOP. Moreover FI-2Pop GA is a high-quality GA solver engine for constrained optimization problems generating excellent decisions for problems that cannot be handled by GENOCOP.

Next, the branch and bound method and genetic algorithms are presented and adapted for solving HPC problems.

3.2.3.1. Branch and Bound (BB).

According to the HPC literature, branch and bound is the most frequently utilized solver for mixed integer programs. Fletcher and Leyer (1995) report that branch and bound is superior by an order of magnitude compared with other algorithms like outer approximation and generalized bender decomposition.

The BB algorithm consists of solving and generating new relaxed problems in accordance with a tree search, where the nodes of the tree correspond to relaxed optimization sub-problems.

Branching is obtained by generating child-nodes from parent-nodes according to branching rules, which can be based, for instance, on a priori specified priorities on integer variables, or on the amount by which the integer constraints are violated. The algorithm stops when all nodes have been fathomed. The success of the branch and bound algorithm relies on the fact that whole sub-trees can be excluded from further exploration by fathoming the corresponding root nodes. This happens if the corresponding sub-problem is either infeasible or an integer solution is obtained. The corresponding value of the cost function serves as an upper bound on the optimal solution of the optimization problem, and is used to further fathoming other nodes having larger optimal value or lower bound (Bemporad and Morari, 1999; Floudas, 1995).

The control algorithm used in this chapter is thoroughly described in Karer *et al.* (2007) and Potocnik *et al.* (2004). Even it is limited to systems with discrete inputs only, its extension to continuous and discrete inputs is straightforward, by solving at each node the corresponding relaxed non-linear optimization problem for the continuous variables. The possible evolution of the system up to a maximum prediction horizon N_u can be illustrated by a tree of evolution, as shown in Figure 3.1 for $N_u = 4$ and 3 possible input vectors. The nodes of the tree represent reachable states, and branches connect two nodes if a transition exists between the corresponding states.

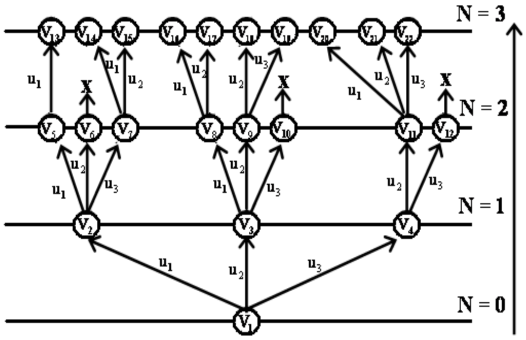

N = 3
N = 2
N = 1
N = 0
V13 V14 V15 V16 V17 V18 V19 V20 V21 V22
V5 V6 V7 V8 V9 V10 V11 V12
V2 V3 V4
V1
u1 u2 u3
X

Figure 3.1 Tree of evolution, Branch and Bound.

For a given root-node V_1, representing the initial states ($x(t), q(t)$), the reachable states are computed and inserted in the tree as nodes V_i , where i indexes the nodes as they were successively computed. A cost value J_i is associated with each new node, and based on the cost value the most promising node is selected. After labelling the node as explored, new reachable states emerging from the selected node are computed. The construction of the tree of evolution continues upwards first, until one of the following conditions occurs:

- The value of the cost function at the current node is larger than the current optimal one ($J_i > J_{opt}$).
- The maximum step horizon is reached.

If the first condition occurs, the node is labelled as non-promising (a "X" shown in Figure 3.1) and thus eliminated from further exploration. On the other hand, if the node satisfies the second condition only, it becomes the new current optimal node ($J_i = J_{opt}$), whereas the sequence of input vectors leading to it becomes the current optimal one.

The exploration continues from the topmost step horizon, where unexplored nodes can be found, and so on, until all the nodes are explored and the optimal input vector can be derived and then applied to the system and the whole procedure is repeated at the next time step.

For an insight into the computational complexity issues and the approaches and properties used for dealing with them, see Karer *et al.* (2007).

3.2.3.2. Optimization based on genetic algorithm.

The genetic algorithm is used to solve the optimization of an objective function because it can efficiently cope with mixed-integer non-linear problems. Another advantage is that the objective-function gradient does not need to be calculated, which substantially reduces the computational effort.

A potential solution of the genetic algorithm is called an individual. The individual can be represented by a set of parameters related to the genes of a chromosome and can be described in binary or integer form. The individual U^i represents a possible control-action sequence $U^i = \{u^i(t), u^i(t+1), ..., u^i(t+N_u-1)\}$ where an element $u^i(t+j)$, $j = 1, ..., N_u - 1$, is a gene, *i* denotes the *i*-th individual from the population of possible individuals, and the individual length corresponds to the control horizon.

Using genetic evolution, the fittest chromosome is selected to ensure the best offspring. The best parent genes are selected, mixed and recombined for the production of an offspring in the next generation. For the recombination of the genetic population, two fundamental operators are used: crossover and mutation. For the crossover mechanism, the portions of two chromosomes are exchanged with a certain probability in order to produce the offspring. The mutation operator alters each portion randomly with a certain probability (for more details see Man *et al.*, 1998).

In this chapter the control-law derivation will be based on the simple genetic algorithm (SGA) as in Man *et al.* (1998). Assume that the range of the manipulated variable is $[u_{\min}, u_{\max}]$, quantized by steps of size *q*, so that there are *q* possible inputs at each time instant. Therefore, the set of feasible control actions is $U = \left\{ u \setminus u = n \cdot \frac{u_{\max} - u_{\min}}{q} + u_{\min}, n = 1, 2, ..., q \right\}$. Furthermore, assume that the probability of two selected parent individuals U^i and U^l undergo a crossover is p_c, and for mutation the probability is p_m. The control strategy can be represented by the following steps:

Step 1. Set the iteration counter to 1, and initialize a random population of *P* individuals, i.e., create *P* random integer feasible solutions of the manipulated variables for the HPC problem. As the control horizon is N_u, there are q^{N_u} possible individuals.

Step 2. Evaluate the objective function (3.1) for all the initial individuals of the population.

Step 3. Select random parents from the population *P* (different vectors of the future control actions).

Step 4. Generate a random number between 0 and 1. If the number is lower than the probability p_c, choose an integer $0 < c_p < N_u - 1$ (c_p denotes the crossover point) and apply the crossover to the selected individuals in order to generate an offspring. Figure 3.2 describes the crossover operation for two individuals, U^i and U , resulting in U^i_{cross} and U^l_{cross}.

$$U^i = \left\{ \boxed{u^i(t), u^i(t+1), \ldots, u^i(t+c_p-1)}, \boxed{u^i(t+c_p), \ldots, u^i(t+N_u-1)} \right\}$$

$$U^l = \left\{ \boxed{u^l(t), u^l(t+1), \ldots, u^l(t+c_p-1)}, \boxed{u^l(t+c_p), \ldots, u^l(t+N_u-1)} \right\}$$

$$\Downarrow$$

$$U^i_{cross} = \left\{ \boxed{u^l(k), u^l(k+1), \ldots, u^l(k+c_p-1)}, \boxed{u^i(k+c_p), \ldots, u^i(k+N_u-1)} \right\}$$

$$U^l_{cross} = \left\{ \boxed{u^i(k), u^i(k+1), \ldots, u^i(k+c_p-1)}, \boxed{u^l(k+c_p), \ldots, u^l(k+N_u-1)} \right\}$$

Figure 3.2 Crossover of two individuals.

Step 5. Generate a random number between 0 and 1. If the number is lower than the probability p_m, choose an integer $0 < c_m < N_u - 1$ (c_m denotes the mutation point) and apply the mutation to the selected parent in order to generate an offspring. Select a value $u^i_{mut} \in U$ and replace the value in the c_m-th position in the chromosome. Figure 3.3 describes the mutation operation for an individual U^i resulting in U^i_{mut}.

$$U^i = \left\{ u^i(k), u^i(k+1), \ldots, u^i(k+c_m-1), \boxed{u^i(k+c_m)}, u^i(k+c_m+1), \ldots, u^i(k+N_u-1) \right\}$$

$$\Downarrow$$

$$U^i_{mut} = \left\{ u^i(k), u^i(k+1), \ldots, u^i(k+c_m-1), \boxed{u^i_{mut}}, u^i(k+c_m+1), \ldots, u^i(k+N_u-1) \right\}$$

Figure 3.3 Mutation of an individual.

Step 6. Evaluate the fitness given by the objective function (3.1) of all the individuals of the offspring population.

Step 7. Select the best individuals according to the objective function. Replace the weakest individuals from the previous generation with the strongest individuals of the new generation.

Step 8. If the objective-function value reaches the defined tolerance or the maximum generation number is reached (stopping criteria), then stop. In other cases, go to step 3.

The tuning parameters of the GA method are the number of individuals, the number of generations, the crossover probability p_c, the mutation probability p and the stopping criteria.

The genetic-algorithm approach in HPC provides a sub-optimal discrete control law close to the optimal one. When the best solution is maintained in the population, it was shown in Rudolph (1994) and Sarimveis (2003) that the GA converges to the optimal solution. However, due to the limited time between the sampling instances reaching the global optimum is not guaranteed. Nevertheless, the probabilistic nature of the algorithm ensures that it finds an approximately optimal solution. In contrast to that, following the Remark 5.3 in Sarimveis (2003), the application of traditional optimization techniques to solve the same problem cannot guarantee even the calculation of a feasible solution because of the complexity of the optimization problem. Since in this case the problem is a complex mixed integer and non-linear programming, using the GA optimization is justified.

Using the GA optimization makes it easy to include the input and output constraints in the computation of the control variable. The procedure is described in Sarimveis (2003); in general, it means a narrowing of the space for feasible solutions in each optimization step. However, this case is beyond the scope of this work.

Solving constrained optimization problems using GA is a very complex issue due to the genetic operators (mutation and crossover) do not guarantee solution feasibility. Although much attention has been given to solve these issues, no general and systematic solution has been proposed. For a review of these algorithm, Back *et al*. (2000), Coello (2002); Michalewicz (1995) proposed excellent reports.

3.2.4. Hybrid Predictive Control for a Batch Reactor.

The HPC using Branch and Bound approach to the optimization problem arising from the optimal control problem was tested on a simulation example of a real batch reactor that is located in a pharmaceutical company and is used in the production of medicines. The batch reactor was described before in section 2.3. The scheme could be seen in Figure 2.1. The goal is to control the temperature of the ingredients stirred in the reactor core so that they synthesize into the final product. In order to achieve this, the temperature has to follow the trajectory reference, given in the recipe, as accurately as possible.

A comparison between HPC based on Fuzzy Model and PWA model is presented. The PWA model obtained is described in chapter 2 by (2.13), (2.14), (2.15) and the hybrid fuzzy model is the one reported in Karer *et al.* (2007). For each HPC method, the Branch and Bound (BB) optimization algorithm is used. The objective function is the following.

$$\begin{aligned} J = w_1 \sum_{h=1}^{N_y} \left(T(t+h) - T_{ref}(t+h) \right)^2 + w_2 \sum_{h=1}^{N_u} k_C(t+h-1) k_H(t+h-1) + \\ w_3 \sum_{h=1}^{N_u} \left| \Delta k_M(t+h-1) \right| k_H(t+h-1) \end{aligned} \tag{3.7}$$

with $w_1 = 1/15, w_2 = 15, w_3 = 0.03$.

Table 3.1 shows the Objective Functions values (tracking error and control effort) and the computation time for the different strategies. Figures 3.4 and 3.5 show the results of HPC based on hybrid fuzzy model with BB (HPC-BB). Figures 3.6 and 3.7 show the results of HPC based on PWA model with BB (HPC-PWA-BB). As the figures and tables show HPC based on Hybrid fuzzy model is better in terms of objective function than HPC based on PWA model, but in terms of computational time, HPC-PWA becomes faster. This difference in time could be explained by the longer evaluation time required for fuzzy models, as its structure is more complex than PWA model.

Table 3.1 N-Step ahead prediction error

Strategy	Jy	Ju	Time [s]
HPC-BB	11371.256	15.1926	197.5640
HPC-PWA-BB	11386.274	15.1932	118.8750

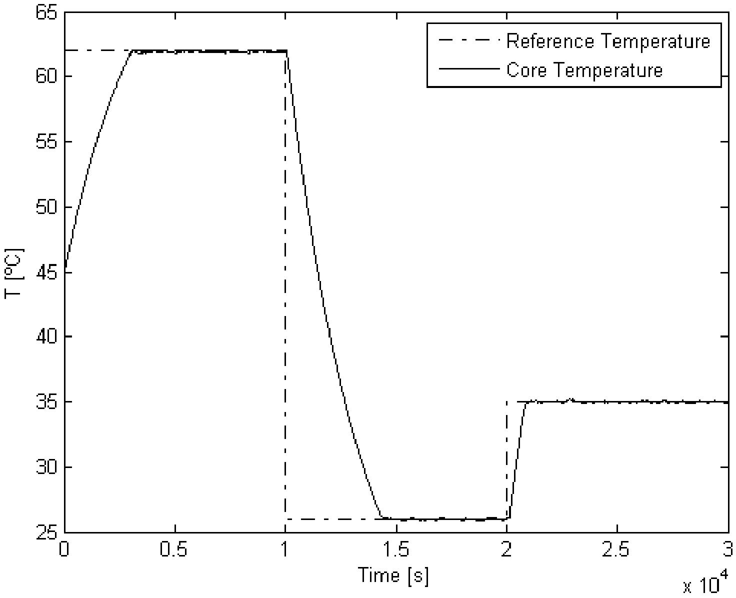


Figure 3.4 Temperature in the core and reference HPC-BB.

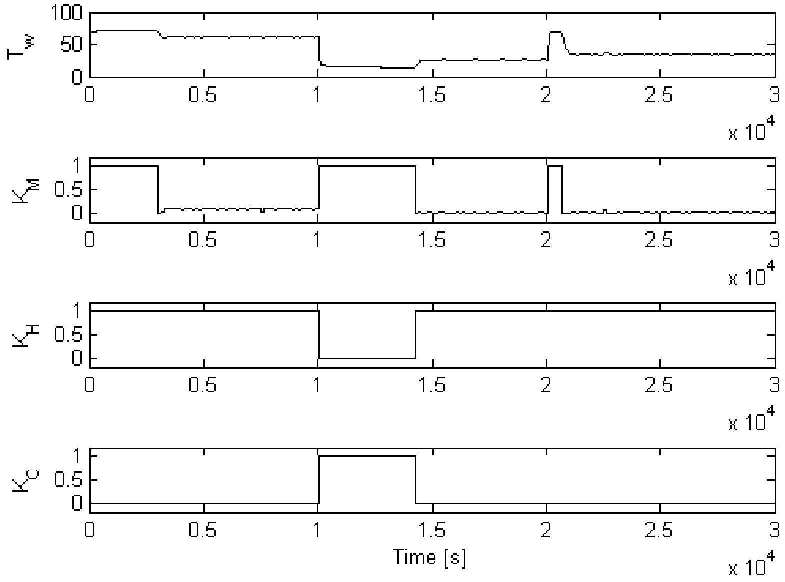


Figure 3.5 Outputs HPC-BB.

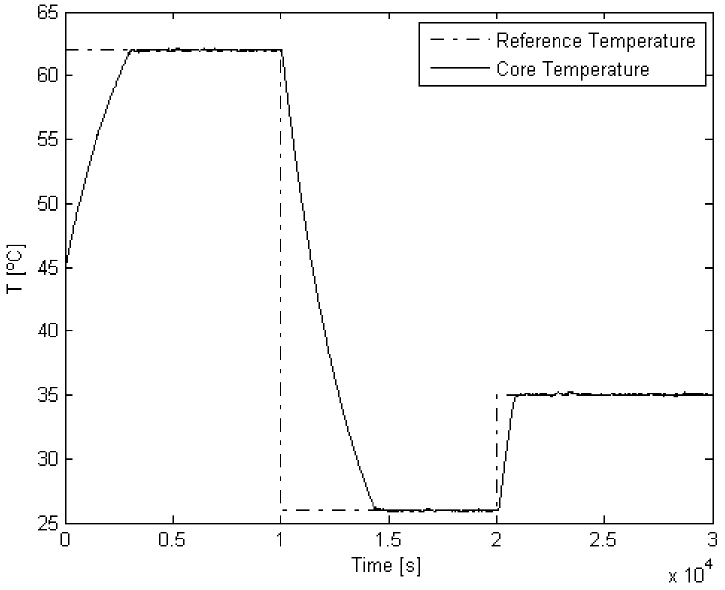


Figure 3.6 Temperature in the core and reference HPC-PWA-BB.

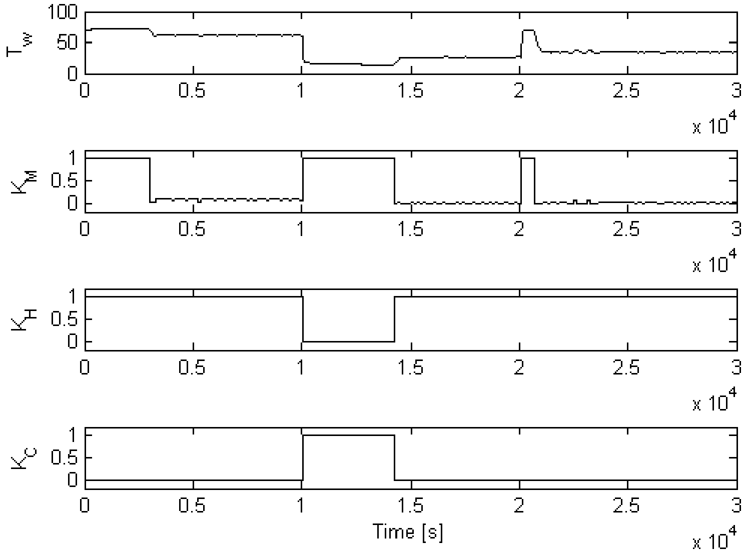


Figure 3.7 Outputs HPC-PWA-BB

3.2.5. Hybrid Fuzzy Predictive Control for a Tank System.

The behaviour of the tank system, shown in Figure 3.8, is defined by the following non-linear differential and algebraic equations, which define the switching regions:

$$\begin{gathered}
\frac{dh_1}{dt}\cdot\pi\cdot\frac{R_1^{\,2}}{H_1^{\,2}}h_1^{\,2} = K_{CP}\cdot u+\phi_{ONOFF2}-\overbrace{V_1 h_1}^{\phi_{V1}}-\phi_{ONOFF1} \\
\frac{dh_2}{dt}\cdot\pi\cdot R_2^{\,2} = \overbrace{V_1 h_1}^{\phi_{V1}}+\phi_{ONOFF1}-\overbrace{V_2 h_2}^{\phi_{V2}}-\phi_{ONOFF2} \\
\text{If } \left(h_2 \ge H_{2\min}\right) \text{ and } \left(h_1 < H_{1\max}\right) \quad \text{then} \quad \phi_{ONOFF2} = K_{ONOFF2} \\
\text{If } \left(h_1 \ge H_{1\max}\right) \text{ and } \left(h_2 < H_{2\max}\right) \quad \text{then} \quad \phi_{ONOFF1} = K_{ONOFF1}
\end{gathered} \tag{3.8}$$

where h_1 and h_2 stand for the level of the liquid in the first and the second tank, and $H_{1\min}$, $H_{2\min}$, $H_{1\max}$ and $H_{2\max}$ stand for the switching levels.

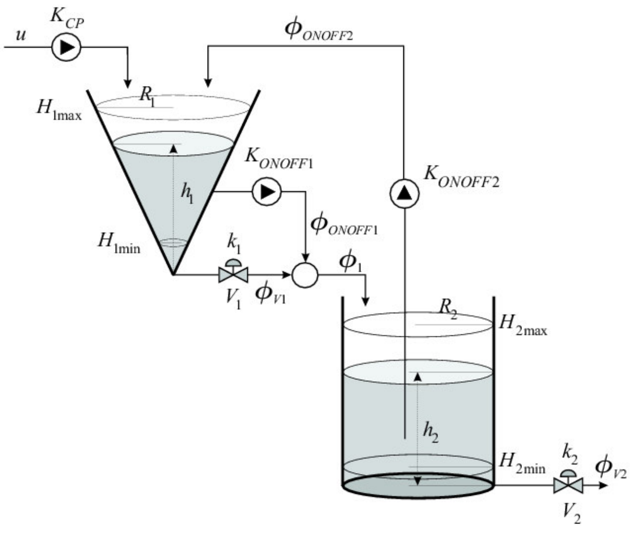


Figure 3.8 Hybrid Tank System

The controlled variable in this case is the level in the first tank h_1, and the manipulated variable is the voltage of the pump at the inlet u, which has discrete levels. It is also assumed that both levels, h_1 and h_2, are measured, and the measurements are corrupted with white noise that has a variance equal to 1. The excitation and the output signals of the plant are shown in Figures 3.9 and 3.10. The signals were sampled with $T_s = 10[s]$.

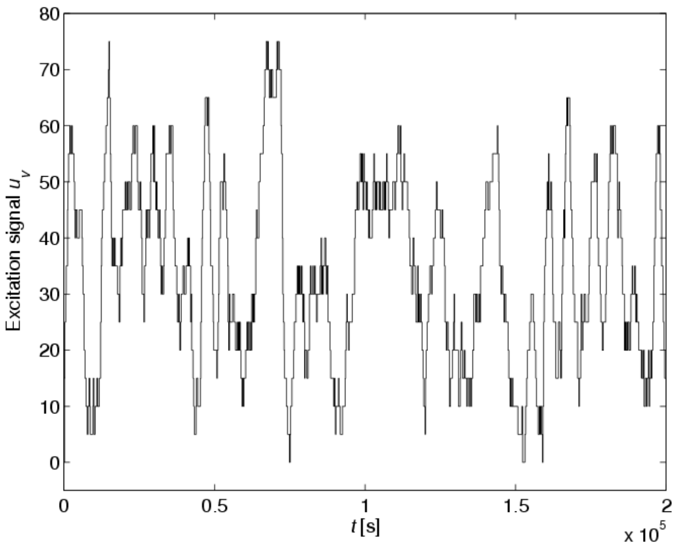


Figure 3.9 Identification data, input signal.

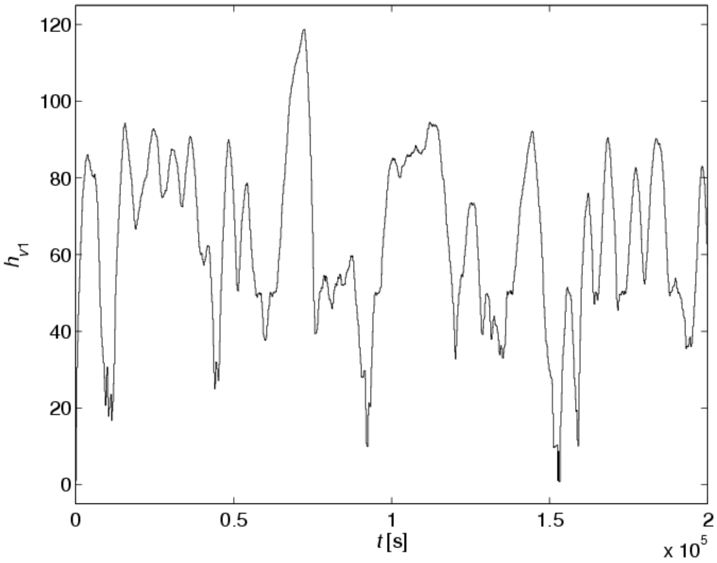


Figure 3.10 Identification data, output signal.

Note that the rules in (3.8) represent the switching or hybrid behaviour of the system. The parameters used in the model are $R_1 = 25[cm]$, $V_1 = 0.5\left[cm^2/s\right]$, $R_2 = 15[cm]$, $V_2 = 0.65\left[cm^2/s\right]$, $H_1 = 100[cm]$, $H_{1\min} = 5[cm]$, $k_{CP} = 1\left[cm^3/s\right]$, $k_{onoff1} = 4\left[cm^3/s\right]$, $H_{1\max} = 50[cm]$, $H_{2\max} = 90[cm]$, $k_{onoff2} = 4\left[cm^3/s\right]$.

The behaviour of the hybrid system will be modelled by the fuzzy-model structure from (2.16). The design of the membership-function distribution is the key element of the modelling procedure. In our case, it is obtained by analyzing the principal eigenvectors of the covariance matrices of the clusters. The clusters are obtained from the data matrix, which is composed of measurements (the variables $h_1(t)$ and $u(t)$).

The analysis of the main eigenvectors for all the clusters is presented in Figure 3.11, where the eigenvector--element ratio corresponds to its own cluster. It is clear that around the level of $h_2(t)=50cm$ there is an abrupt change of the eigenvector ratio. This change implies a change in the system's behaviour and potentially indicates the switching region of the system. The idea is to put two membership functions around each local extreme (the minimum and maximum of the eigenvector ratios). This is done because the switching region cannot be exactly defined (mainly in the case of noisy data). This idea involves a tolerance band around the switching regions. In Figure 3.11 the corresponding membership functions are shown as well.

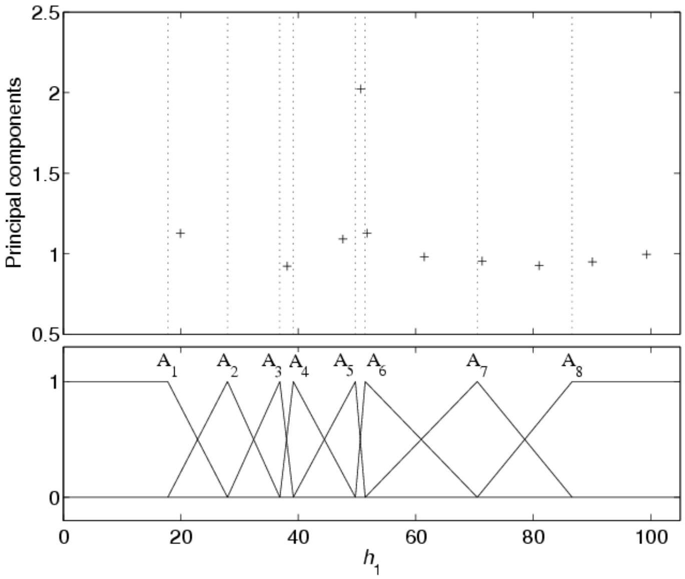


Figure 3.11 Principal component and membership functions.

The structure of the fuzzy model follows the definition in (2.16), where the variable in the premise is $h_1(t)$ and the consequent vector is equal to $\left[h_1(t),u(t),1\right]^T$. The parameters of the fuzzy model $\theta_i = [a_i, b_i, r_i]^T$, obtained by a linear least-squares estimation, are reported in Table 3.2.

Table 3.2 Parameters fuzzy Model

i	a_i	b_i	r_i
1	0.8376	0.3403	0.0386
2	0.9764	0.0522	0.0511
3	0.9873	0.0290	0.0305
4	0.9747	0.0196	0.7656
5	0.9933	0.0125	-0.0136
6	0.9946	0.0091	0.0265
7	0.9987	0.0066	-0.2163
8	1.0015	0.0045	-0.4334

The validation of the designed fuzzy model is shown in Figure 3.12. The proposed model results in a very good estimation of the process output, and inherently incorporates the hybrid (switching) nature of the system.

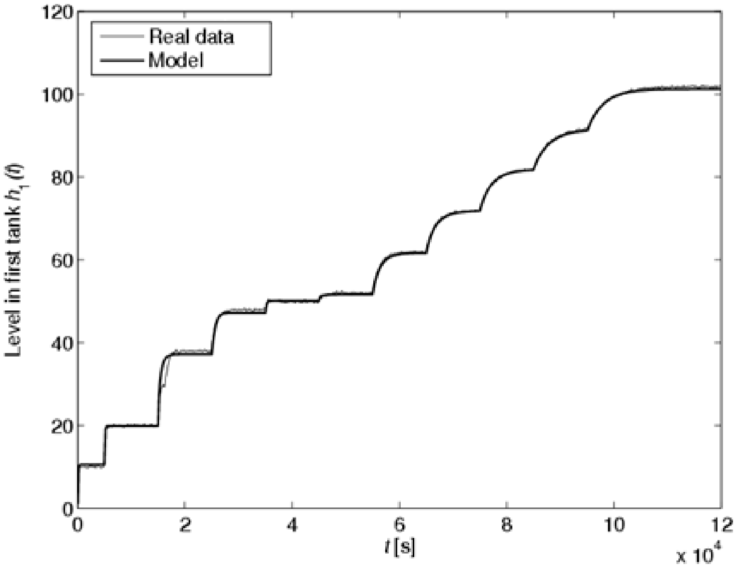


Figure 3.12 Validation of hybrid fuzzy model, output signal.

The tuning parameters of the objective function in (3.5) are $N_1 = 1$, $N = N_y = N_u = 3$, and $\lambda = 0.001$. The total computation time required for the HPC will be evaluated using a Intel Core(TM) 2 CPU, 2.40 GHz processor and 3.25 GB of RAM.

The sampling time is 10[*s*] and the total simulation time is 6000 [*s*]. The results of the proposed method based on GA (HPC-GA) are compared with the results obtained by using Branch-and-bound method (HPC-BB) and explicit enumeration (HPC-EE). The latter evaluates all the feasible control actions at every instant, while the HPC-GA and HPC-BB consider only a reduced space search. The parameters for HPC-GA are as follows: mutation probability $p_m = 0.001$, crossover probability $p_c = 0.7$ and for the stopping criterion the maximum number of generations is used, obtained by further analyses. 50 replications were conducted for each statistic obtained with GA.

Figure 3.13 shows the objective function as a function of the generation number for different numbers of individuals. Based on this figure, 30 generations with 14 individuals are selected in this example. Figure 3.14 shows how this selection brings a trade-off between the computation time and the value of the objective function.

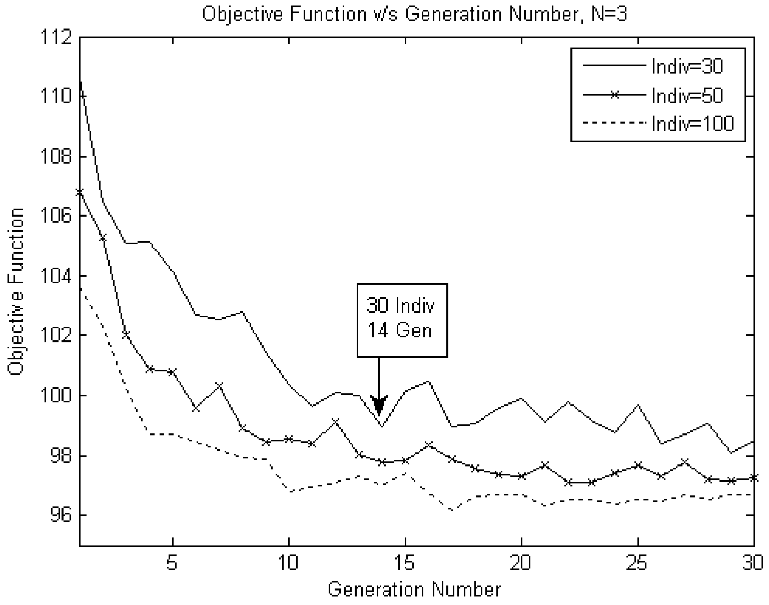


Figure 3.13 Objective functions v/s generation number.

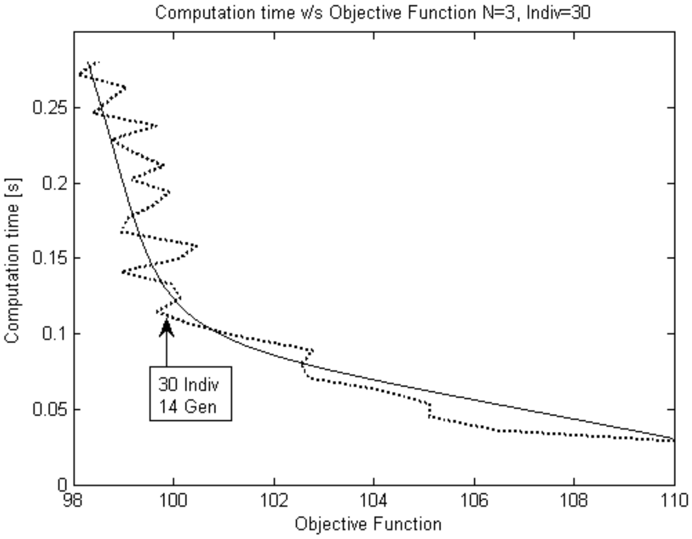


Figure 3.14 Pareto front, Objective functions and Computation time.

Figure 3.15 presents the computation time as a function of the number of generations for different numbers of individuals. The computation time linearly depends on the generation number, and its slope slightly increases with the number of individuals. It is clear that the time required to calculate the solution in each sampling time is shorter than the sampling time for all cases. This means that all the proposed control strategies are suitable for real-time control in the sense of time consumption. For 30 generations with 14 individuals, the computation time was approximately 84.3 [*s*] (1.41% of the total simulation time), and the computation time during each iteration was smaller than the sampling time.

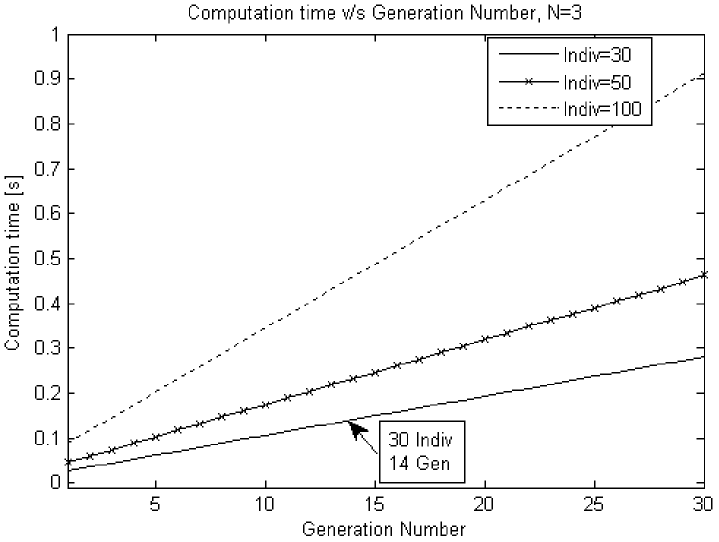


Figure 3.15 Generation number v/s Computation time.

With optimal values of 30 generation with 14 individuals, the results of the HPC-GA were obtained. Figures 3.16 and Figure 3.18 show the controlled variable (conic tank level $h_1(t)$) and the manipulated variable (discrete voltage of pump u), respectively, for the HPC-GA, HPC-EE and HPC-BB. Figure 3.17 and Figure 3.19 show the response detail for 3500 to 5000 [*s*].

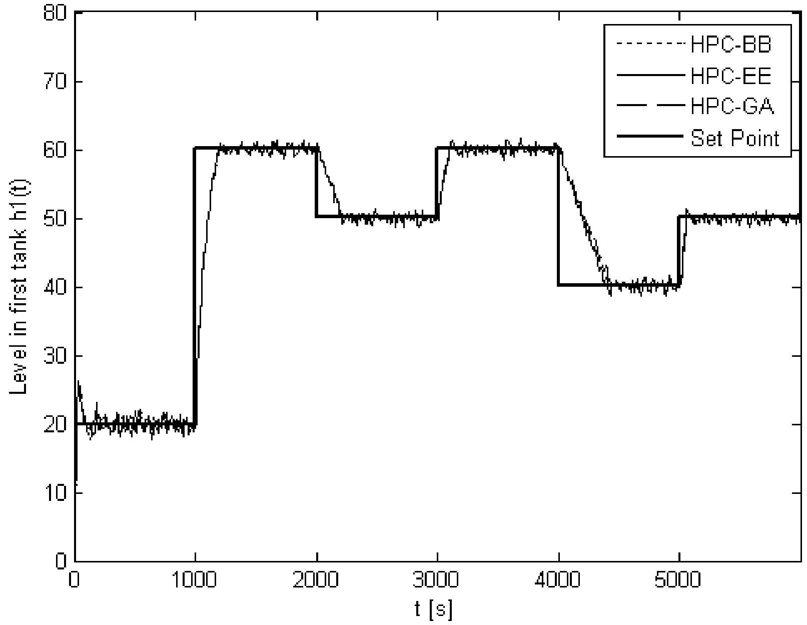


Figure 3.16 Controlled variable.

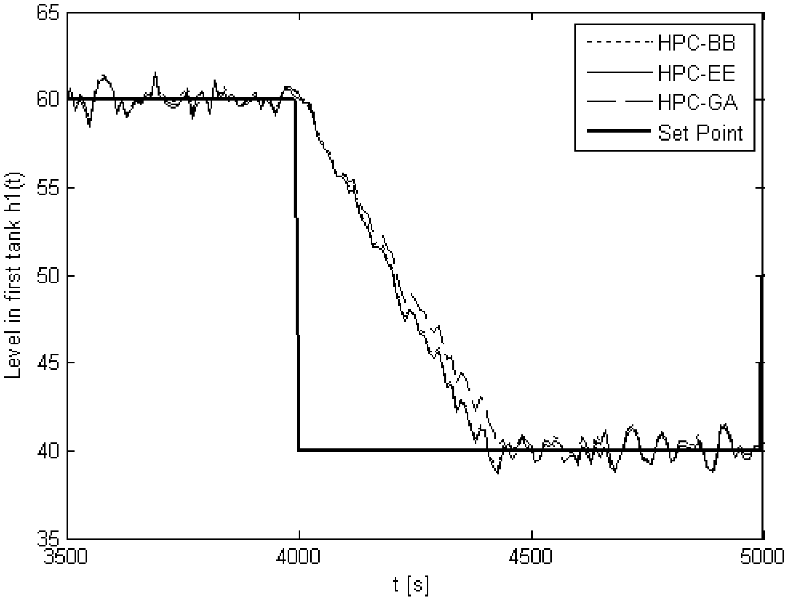


Figure 3.17 Controlled variable (details).

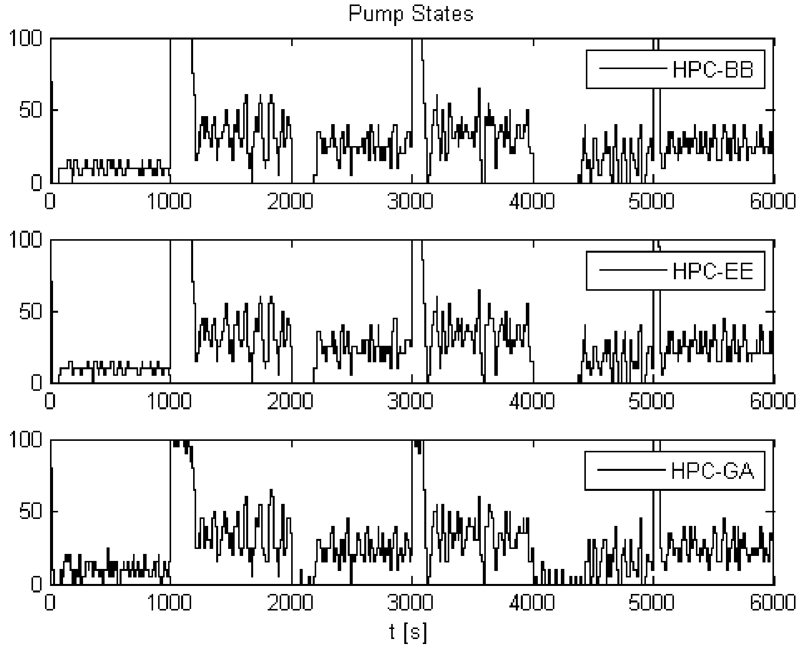


Figure 3.18 Pump States.

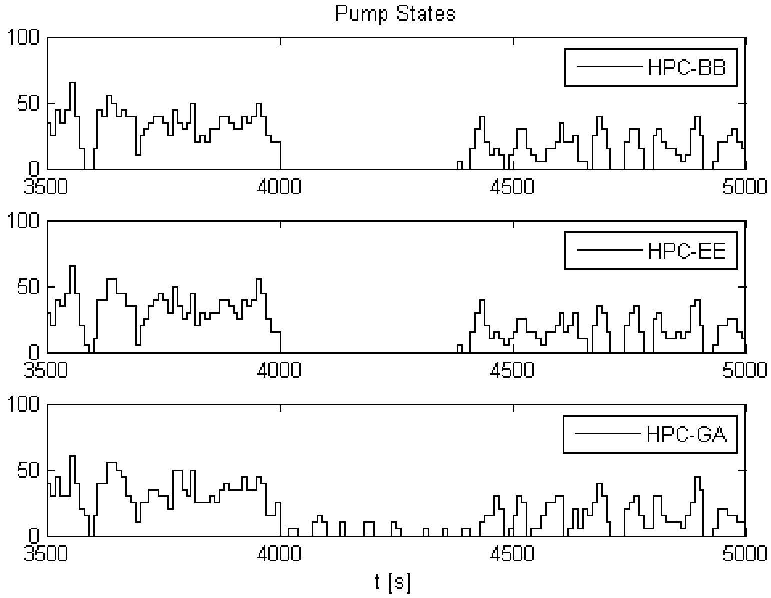


Figure 3.19 Pump States, zoom.

In Table 3.3 the mean values of the objective function, the total computation times and the mean computation times for the same simulation test are presented. Table 3.4 presents the statistical values of the controlled and manipulated variables.

Table 3.3 Performance indexes.

N_2=N_u=3, λ=0.001	J_1	J_2	J	Total computing time	Mean computing time by sampling time
HPC-EE	96.69	432.4	97.12	1741.7 [s]	2.898 [s]
HPC-GA (30,14)	98.93	488.6	99.48	84.3 [s]	0.140 [s]
HPC-BB	97.03	427.9	97.46	208.9 [s]	0.348 [s]

Table 3.4 Performance indexes.

N_2=N_u=3, λ=0.001	Mean(|y-r|)	Mean(|Δu|)	Std(|y-r|)	Std(|Δu|)
HPC-EE	2.0910	7.1500	4.8468	9.7999
HPC-GA (30,14)	2.2161	8.5502	4.8619	9.7494
HPC-BB	2.1113	7.1833	4.8539	9.6983

As the HPC-GA is a heuristic search algorithm, some differences compared with the HPC-EE and HPC-BB for the controlled and manipulated variables can be seen in Figures 3.16, 3.17, 3.18 and 3.19. However, the HPC-GA response is near to the optimal solution given by the HPC-EE, (benchmark) as shown in figures 3.16 and 3.18, as well as in Table 3.3.

As shown in Table 3.3 and Table 3.4, the manipulated-variable indices (Mean(|Δu|) and Std(|Δu|)) are slightly in favour of the HPC-GA case. However, this brings only a 0.4% better tracking response for the optimal HPC-EE method (Mean(|y-r|) and Std(|y-r|)). This proves that the HPC-GA method is nearly optimal, and it brings a considerable reduction in the computational load.

Figure 3.20 shows a comparison of mean computation times for all three cases, In comparison with the HPC-EE, a 95.2% reduction in the computation time on account of a 2.37% increase in the cost function is obtained with the HPC-GA. Comparing the results with the HPC-BB, a 59.6% reduction in the computation time brings only a 2.03% increase in the cost function. By limiting the number of computations via the selection of the numbers of individuals and generations, it is still possible to achieve near optimal tracking results on account of a considerable reduction in the computational load.

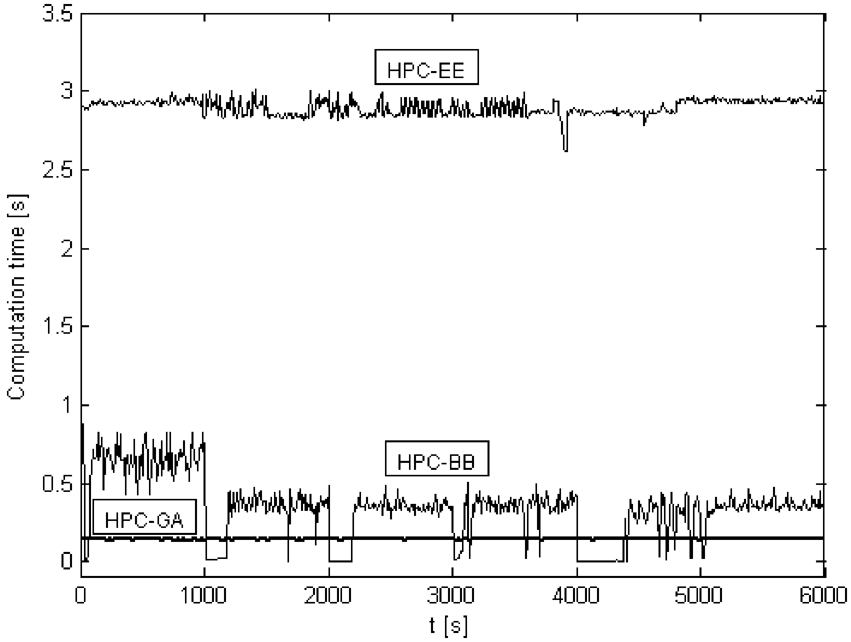


Figure 3.20 Computation time.

3.3. Hybrid Predictive Control based on Multi-objective Optimization

3.3.1. Hybrid Predictive Control (HPC)

As mentioned in section 3.2, the HPC strategy is a generalization of model predictive control (MPC), where the prediction model includes both discrete/integer and continuous variables. Consider the following HPC optimization problem with a variable weighting function:

$$\begin{aligned} &\min_{\{u(k),u(k+1),\ldots,u(k+N_u-1)\}} J = J_1 + J_2 \\ &J_1 = \sum_{j=N_1}^{N_y} \lambda(k+j)\left(\hat{y}(k+j)-r(k+j)\right)^2 \\ &J_2 = \sum_{j=N_1}^{N_u} \left(1-\lambda(k+j)\right)\Delta u(k+j-1)^2 \end{aligned} \tag{3.9}$$

where J is the objective function, $\hat{y}(k+j)$ corresponds to the *j*-step-ahead prediction of the controlled variable based on a hybrid model, $r(k+j)$ is the reference, $\Delta u(k+j-1)$ is the increment of the control action, and $\lambda(k+j)$ is the weighting factor sequence. N_1, N_y and N_u are the prediction horizons and the control horizon, respectively. The optimization results in a control sequence namely $\{u(k),...,u(k+N_u-1)\}$.

J_1 and J_2 are the objective functions which are weighted by $\lambda(k+j)\in[0,1]$ giving more importance to the tracking the reference (J_1) or in minimizing the control effort (J_2). In this case both objectives are opposites because when J_1 is minimized, J_2 increase its value. It is important to say that the stability of the controller depends also in the weighting factor. However, finding a proper weighting function sequences is not an easy task. Therefore, a fixed weighting factor is commonly used (Nunez-Reyes *et al.*, 2002).

In a more general expression, consider the following HPC which two opposite objectives.

$$\begin{aligned} &\min_{\{\mathbf{u}(k),\mathbf{u}(k+1),...,\mathbf{u}(k+N_u-1)\}} J = \lambda J_1 + (1-\lambda)J_2 \\ &J_1 = f_1\left(\hat{\mathbf{y}}(k+1),...,\hat{\mathbf{y}}\left(k+N_y\right),\mathbf{u}(k),...,\mathbf{u}(k+N_u-1)\right) \\ &J_2 = f_2\left(\hat{\mathbf{y}}(k+1),...,\hat{\mathbf{y}}\left(k+N_y\right),\mathbf{u}(k),...,\mathbf{u}(k+N_u-1)\right) \end{aligned} \tag{3.10}$$

where $\hat{\mathbf{y}}(k+j)$ is the *j*-step-ahead of the vector of controlled variables based on a hybrid fuzzy model, $\mathbf{u}(k+j-1)$ is the input vector in instant $k+j-1$, J_1 and J_2 are the opposite objectives and $\lambda\in[0,1]$ is a fixed weighting factor.

When (3.10) is solved, usually one optimal solution will be obtained, and based on the rolling horizon procedure, the optimal input is applied. If the importance between the objectives functions changed, a new HPC should be solved with a different weighting factor. However, the trade-off between optimal solutions will not be clearly obtained, so it is difficult to visualize the consequences of changing the importance in the objective function. For this reason and others, next, the multi-objective hybrid predictive control (MO-HPC) approach is explained.

3.3.2. Multi-objective Hybrid Predictive Control (MO-HPC).

The MO-HPC strategy is a generalization of HPC, where control objectives are similar to HPC but the optimal control action must be chosen based on a criterion that selects a solution from the Pareto Optimal region of the following problem:

$$\begin{aligned}&\min_{\{\mathbf{u}(k),\mathbf{u}(k+1),\ldots,\mathbf{u}(k+N_u-1)\}}\{J_1,J_2\}\\&J_1=f_1\left(\hat{\mathbf{y}}(k+1),\ldots,\hat{\mathbf{y}}(k+N_y),\mathbf{u}(k),\ldots,\mathbf{u}(k+N_u-1)\right)\\&J_2=f_2\left(\hat{\mathbf{y}}(k+1),\ldots,\hat{\mathbf{y}}(k+N_y),\mathbf{u}(k),\ldots,\mathbf{u}(k+N_u-1)\right)\end{aligned}\tag{3.11}$$

where J_1 , J_2 are the objective functions to minimize depending on the process. The optimization solution is a control sequence region called the Pareto Optimal set. To formalize the notion, the following concepts are important to define.

Let us consider $U^i=\left\{\mathbf{u}^i(k),\ldots,\mathbf{u}^i(k+N_u-1)\right\}$ a control action sequence, where $\mathbf{u}^i(k)$ belongs to the set of feasible control action. A solution U^i Pareto-dominates to a solution U^j if and only if, $\left(J_1(U^i)\leq J_1(U^j)\right)\wedge\left(J_2(U^i)<J_2(U^j)\right)$ or $\left(J_2(U^i)\leq J_2(U^j)\right)\wedge\left(J_1(U^i)<J_1(U^j)\right)$.

A solution U^i is said to be Pareto optimal if and only if there is not U that Pareto-dominates U^i. Pareto optimal set P_S contains all Pareto optimal solutions. The set of all objective function values corresponding to the solutions in P_S is $P_F=\left\{\left(J_1(U^i),J_2(U^i)\right):U^i\in P_S\right\}$. P_F is known as Pareto optimal front. If the discrete manipulated variable case is considered, where the feasible input set is finite, the size of P is also finite.

Then, the multi-objective hybrid predictive control solves a multi-objective problem, obtaining a set of Pareto optimal solutions (control actions). The difference between HPC and MO-HPC is shown in Figure 3.21. As Figure 3.21 shows, MO-HPC provides a set of solutions, but just one input $u(k)$ has to be applied to the system. In this case, at every instant, the controller (operator) has to make a decision on how using the information provided by the Pareto set. Then, together with the MO-HPC, a criterion is used in order to find the control sequence that better suits the objectives. The chosen control sequence belongs to the Pareto front

$U^i = \{u^i(k),...,u^i(k+N_u-1)\}$ and then is optimal. Note that the solution provided by HPC belongs to the solutions set of MO-HPC.

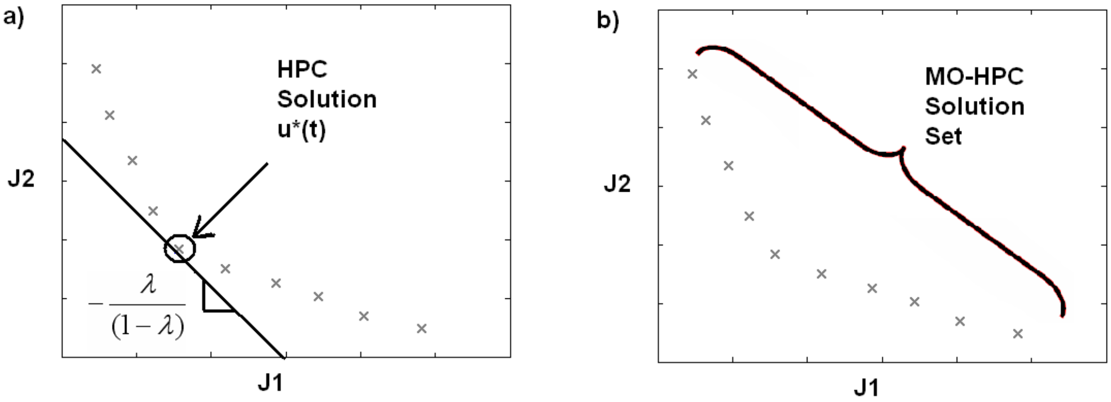


Figure 3.21. a) HPC solution, b) MO-HPC solution set, among its elements is the HPC solution.

Regarding the criterion for MO-HPC, it could be related to tracking error J_1 as well as control effort J_2. Some examples of possible trade-offs in different applications are defined later. In problems where there is a flexibility to decide which criterion is better, MO-HPC suits very well, as it is a tool that support the controller (operator) as its helps choose a solution, considering graphically the trade-off between Pareto optimal solutions.

The multi-objective optimization could be solved by evaluating all solutions (Explicit Enumeration), through Branch & Bound or other algorithms. However MO-HPC strategies generate NP-hard problems that have to be solved by efficient solvers. Next based on the genetic algorithm proposed in section 3.2.3.2 , an efficient optimizer based on Genetic Algorithms (GA) is described for this problem, reaching pseudo-Pareto front but keeping the same computational effort than the HPC strategy.

3.3.3. MO-HPC solved using Genetic Algorithms

Evolutionary multi-objective optimization (EMO) has been applied for a large number of static problems. Some works have been developed for dynamic multi-objective problems, although there are not general methodologies to be applied so far (Farina *et al.*, 2004). The dynamic multi-objective problems are associated with real-time applications where the parameters of the objective functions and/or the constraints changes on-line and many objectives are involved. Farina *et al.* (2004) propose a base algorithm to solve this kind of problems and strongly suggest the necessity of using state of the art EMO methods such as NSGA-II (non-dominated sorting GA II), SPEA2 (strength Pareto evolutionary algorithm) or PESA (Pareto envelope-based selection algorithm), etc.

Within the last years, different efficient EMO algorithms have been developed based on genetic algorithms. NSGA-II is the most used and it was introduced by Deb *et al.* (2000). NSGA-II consists of a non-dominated sorting approach with a lower computational complexity than previous algorithms. A selection operator is considered which creates a mating pool by combining the parent and child populations and selecting the best solutions (elitist approach). It also considers less sharing parameters, reducing the difficult of tuning such parameters. Simulation results show that NSGA-II is able to find a much better spread of solutions. Tan *et al.* (2003) propose a distributed cooperative evolutionary algorithm which involves multiple solutions in the form of cooperative subpopulations and exploits the inherent parallelism by sharing the computational workload among computers over the network. The method provides solutions to not only be pushed to the true Pareto front but also well distributed and with a very competitive performance and computation time.

Hu & Eberhart (2002) and Zhang *et al.* (2003) present particle swarm optimization (PSO) algorithms for multi-objective problems. The main advantage of the PSO is given by the accuracy and speed solution provided. Hu & Eberhart (2002) modify PSO by using a dynamic neighbourhood strategy, new particle memory updating and one-dimension optimization to deal with multiple objectives. Zhang *et al.* (2003) improve the selection manner for global solution and individual solution for the PSO applied to MO problems.

Coello & Becerra (2003) propose a “cultural algorithm” based on evolutionary programming, considering Pareto ranking and elitism. The comparison of the proposed algorithm with NSGA-II validates the method for MO problems. Besides, Coello *et al.* (2004) present an approach in which Pareto dominance is incorporated into PSO in order to allow the heuristic to handle MO problems. The new algorithm improves the exploratory capabilities of PSO by introducing a mutation operator whose range of action varies over time. The results show that the algorithm is a viable alternative since it has an average performance highly competitive with respect to some of the best EMO algorithms known at present. In fact, they report that their algorithm was the only algorithm from those adopted in the study that was able to cover the full Pareto front of all the functions used.

Knowles (2006) presents a ParEGO algorithm for solving multi-objective optimization in scenarios where each solution evaluation is financially and/or temporally expensive. ParEGO is an extension of the single objective efficient global optimization (EGO) algorithm, and uses a design of experiments inspired in an initialization procedure and learns a Gaussian processes model of the search landscape, which is updated after every function evaluation. ParEGO exhibits good performance on the tested function offering a more effective search on problems like the instrument setup optimization problem where only one function evaluation can be performed at each time.

Goh *et al.* (2010) presents a competitive and cooperative co-evolutionary approach adapted for multi-objective particle swarm optimization algorithm design, which have considerable potential for solving complex optimization problems by explicitly modeling the co-evolution of competing and cooperating species. The modeling helps to produce the reasonable problem decompositions by exploiting any correlation and interdependency among components of the problem.

Genetic algorithm is used to solve the multi-objective HPC because it can efficiently cope with mixed-integer non-linear problems. The idea is to find the Pareto optimal set and then from the Pareto optimal front that will be used to obtain the control action. A potential solution of the GA is called individual. The individual can be represented by a set of parameters related to the genes of a chromosome and can be described in a binary or integer form. The individual represents a possible control-action sequence $\left\{\mathbf{u}(k),...,\mathbf{u}(k+N_u-1)\right\}$, where each element is a gene, and the individual length corresponds to the control horizon N_u.

Using genetic evolution, the fittest chromosomes are selected to assure the best offspring. The best parent genes are selected, mixed and recombined for the production of an offspring in the next generation. For the recombination of genetic population, two fundamental operators are used: crossover and mutation. For the crossover mechanism, the portions of two chromosomes are exchanged with a certain probability in order to produce the offspring. The mutation operator alters each portion randomly with a certain probability.

In order to find the Pareto Optimal set of MO-HPC, the best individuals are the ones that belong to the best Pareto Optimal set found until current iteration (due to the fact that there are solutions that belong to the Pareto Optimal set but they are not found yet). Solutions that belong to the best Pareto Optimal set will have a fitness function equal to a certain threshold (0.9 in this case) and the other solution fitness will be equal to a lower threshold (for example 0.1) in order to hold the solution diversity.

The complete procedure for the GA applied to this MO-HPC control problem is as follows:

Step 1. Set the iteration counter to i=1, and initialize a random population of n individuals, i.e., create n random integer feasible solutions of the manipulated variable sequence. As the control horizon is N_u, there are Q^{N_u} possible individuals. Not all individuals are feasible because of the constraints explained above. The size of the population is n individuals per generation.

$$\text{Population } i \;\Leftrightarrow\; \begin{pmatrix} \text{Individual 1} \\ \text{Individual 2} \\ \vdots \\ \text{Individual } n \end{pmatrix}$$

In general, individual j means that the vector of the future control action is:

$$\textit{Individual}\, j = \left[u^j(k)\ ,\ u^j(k+1)\ ,\ldots,\ u^j(k+N_u-1) \right]^T_{N_u \times 1}$$

Step 2. For every individual, evaluate J_1 and J_2 corresponding to the defined objective functions in (3.11). Then, obtain the fitness function of all individual in the population. In fact, when considering individuals belonging to the best pseudo-optimal Pareto set, fitness function equal to $n_{\max}$ will be set; otherwise $n_{\min}$ will be used, in order to maintain the solution diversity. If the individual is not feasible, penalize it (pro-life strategy).

Step 3. Select random parents from the population i (different vectors of the future control actions).

Step 4. Generate a random number between 0 and 1. If the number is less than the probability p_c, choose an integer $0 < c_p < N_u - 1$ (c_p denotes the crossover point) and apply the crossover to the selected individuals in order to generate an offspring. The next scheme describes the crossover operation for two individuals, U^j and U^l, resulting in U^j_{cross} and U^l_{cross}.

$$
\begin{aligned}
U^j &= \left\{ \boxed{u^j(k), u^j(k+1), ..., u^j(k+c_p-1)}, \boxed{u^j(k+c_p), ..., u^j(k+N_u-1)} \right\} \\
U^l &= \left\{ \boxed{u^l(k), u^l(k+1), ..., u^l(k+c_p-1)}, \boxed{u^l(k+c_p), ..., u^l(k+N_u-1)} \right\} \\
&\Downarrow \\
U^j_{cross} &= \left\{ \boxed{u^l(k), u^l(k+1), ..., u^l(k+c_p-1)}, \boxed{u^j(k+c_p), ..., u^j(k+N_u-1)} \right\} \\
U^l_{cross} &= \left\{ \boxed{u^j(k), u^j(k+1), ..., u^j(k+c_p-1)}, \boxed{u^l(k+c_p), ..., u^l(k+N_u-1)} \right\}
\end{aligned}
$$

Step 5. Generate a random number between 0 and 1. If the number is less than the probability p_m, choose an integer $0 < c_m < N_u - 1$ (c_m denotes the mutation point) and apply the mutation to the selected parent in order to generate an offspring. Select a value $u^j_{mut} \in U$ and replace the value in the c_m-th position in the chromosome. The next scheme describes the mutation operation for an individual U resulting in U^j_{mut}.

$$
\begin{aligned}
U^j &= \left\{ u^j(k), u^j(k+1), ..., u^j(k+c_m-1), \boxed{u^j(k+c_m)}, u^j(k+c_m+1), ..., u^j(k+N_u-1) \right\} \\
&\Downarrow \\
U^j_{mut} &= \left\{ u^j(k), u^j(k+1), ..., u^j(k+c_m-1), \boxed{u^j_{mut}}, u^j(k+c_m+1), ..., u^j(k+N_u-1) \right\}
\end{aligned}
$$

Step 6. Evaluate the objective functions J_1 and J_2 of all the individuals of the offspring population. Then obtain the fitness of each individual by following the fitness definition described in step 2. If the individual is unfeasible, penalize its corresponding fitness.

Step 7. Select the best individuals according to their fitness. Replace the weakest individuals from the previous generation with the strongest individuals of the new generation.

Step 8. If the tolerance given by the maximum generation number is reached (stopping criteria, *i* equals the number of generation), then stop. Otherwise, go to step 3. Note that since the focus is on a real-time control strategy, the best stopping algorithm criterion corresponds to the number of generations (so, the computational time could be bounded).

The tuning parameters of the MO-HPC method based on GA are the number of individuals, the number of generations, the crossover probability p_c , the mutation probability p_m and the stopping criteria. At each stage of the algorithm, to find the pseudo-optimal Pareto set, the best individuals will be those who belong to the best Pareto set found until the current iteration. From the pseudo-optimal Pareto front, it is necessary to select only one control sequence $u^* = \left\{u^*(k),...,u^*(k+N_u-1)\right\}$ and from that, apply the current control action $u^*(k)$ to the system according to the receding horizon concept. For the selection of this sequence, a criterion related to the importance given to both objectives J_1 and J_2 in the final decision is needed, as the experiments conducted show, detailed in section 3.3.4 next.

The genetic algorithm approach in MO-HPC provides a sub-optimal Pareto front very close to the optimal one. The tuning parameters of the GA method are the number of individuals, number of generations, crossover probability, mutation probability and the stopping criteria. Once the best Pareto front is found, different criteria could be applied in order to select the best control action at every instant. The following criteria are proposed:

1) Choose the control action solution from the Pareto front that has a minimal tracking error value.

2) Fix a bounded tracking error and choose the control action solution from the Pareto front that satisfies that tolerance and has a minimal control effort.

Next, the design of HPC and MO-HPC of a tank system are described and compared in order to show the advantages of the proposed MO-HPC.

3.3.4. MO-HPC for a Tank System.

The tank system consists of a conic tank, a cylindrical tank, valves and pumps as shown in Figure 3.8. The controlled variable is the level in the first tank h_1, and the manipulated variable is the voltage of the pump in the inlet (u), which has discrete levels. It is also assumed that both levels h_1 and h_2, are measured. The behavior of the system is defined by the non-linear differential equations and algebraic equations (3.8), which define the switching regions. Note that the rules in (3.8) represent the switching or hybrid behavior. The following multi-objective problem will be solved:

$$\begin{gathered}\min_{\{u(k),u(k+1),\ldots,u(k+N_u-1)\}}\{J_1,J_2\}\\ J_1=\lambda\sum_{j=N_1}^{N_y}\left(\hat{y}(k+j)-r(k+j)\right)^2\\ J_2=(1-\lambda)\sum_{j=N_1}^{N_u}\Delta u(k+j-1)^2\end{gathered}\tag{3.12}$$

Based on input/output data the same hybrid fuzzy model as in section 3.2.5 is used. The tuning parameters of the multi-objective function in (3.12) are given by $N_1 = 1$, $N = N_y = N_u = 3$.

For the optimization based on GA, the mutation probability equals 0.001, the crossover probability equals 0.7, the generations number equals 50, the individuals number equals 30 and the maximum number of generations is used as stopping criterion. The controllers will be compared with a conventional HPC with $\lambda = 0.001$.

HPC-EMO is tested using the criteria defined in section 3.3.3:

- HPC-EMO1. To choose the solution from the Pareto front that has a minimal tracking error value.
- HPC-EMO2. To fix a bounded tracking error equal to 0.5[cm] and to choose the control action from the Pareto front that satisfies that tolerance and has a minimal control effort.
- HPC-EMO3. To fix a bounded tracking error equal to 1[cm] and to choose the control action from the Pareto front that satisfies that tolerance and has a minimal control effort.

Figure 3.22 and Figure 3.23 show the controlled variable (conic tank level h_l) and the manipulated variable (discrete voltage of pump u), respectively for the three criteria HPC-EMO1, HPC-EMO2, HPC-EMO3 and HPC with λ = 0.001. Figure 3.24 and Figure 3.25 show the controlled and the manipulated variables detailed in the range of 1100 to 2000 [s].

From figures 3.22 to 3.25 and as expected from the criteria definitions, HPC-EMO satisfies each criterion applied to the controlled variable and the control effort is reduced as the tracking error increases. The conventional HPC has a larger control effort than HPC-EMO2 and HPC-EMO3, but its response follows the reference in a better way. HPC-EMO1 reaches the lowest tracking error, but its control effort is the largest. In Table 3.5 the mean values and standard deviation of tracking error and control effort are shown for data of figures 3.22 and 3.23 (performance with a fixed reference). From Table 3.5, HPC-EMO3 reaches the lowest control effort, but the largest tracking error as observed also from figures 3.24 and 3.25. Therefore, Table 3.5 shows that the solutions of the different criteria belong to a Pareto front, which is shown in Figure 3.26.

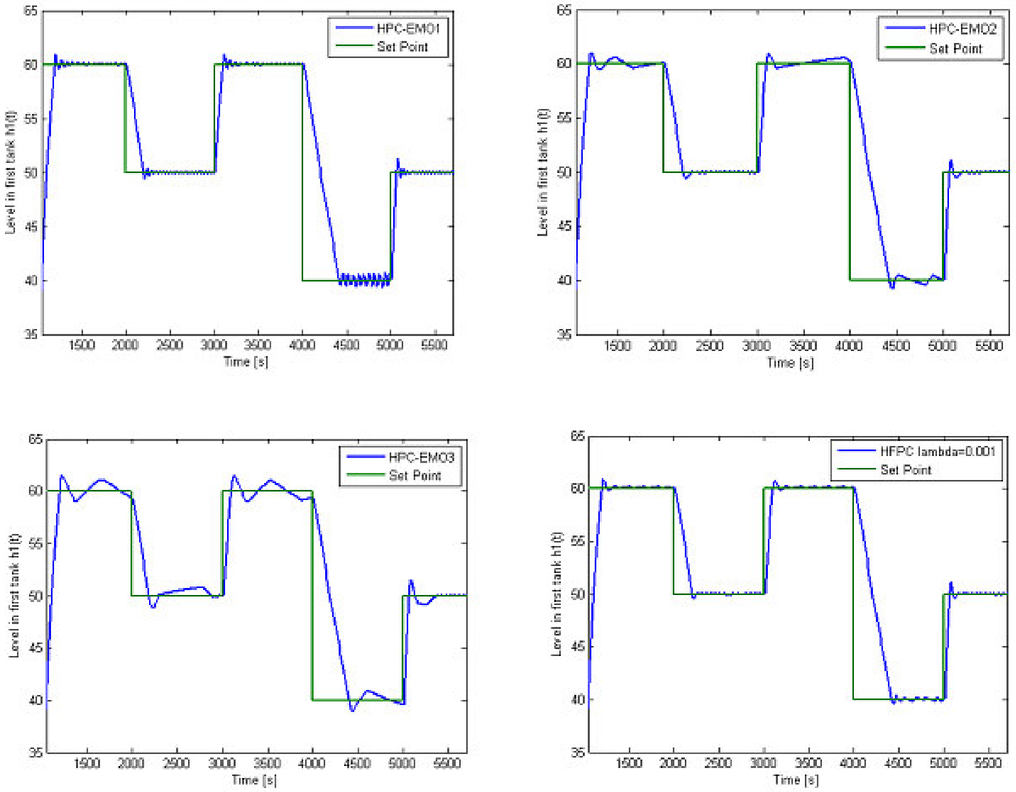


Figure 3.22. Controlled variable, Criterion 1, 2, 3 and HPC.

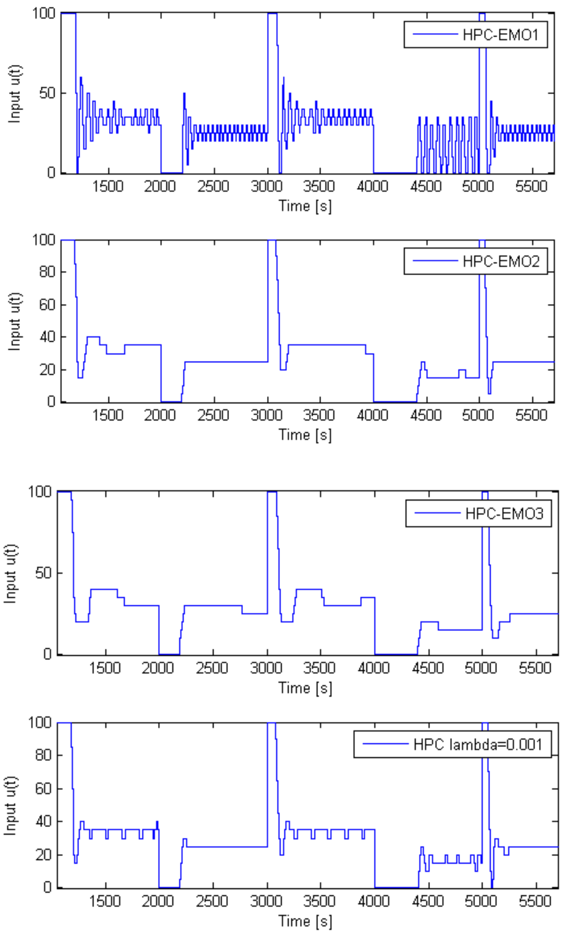


Figure 3.23. Simulation test. Manipulated variable.

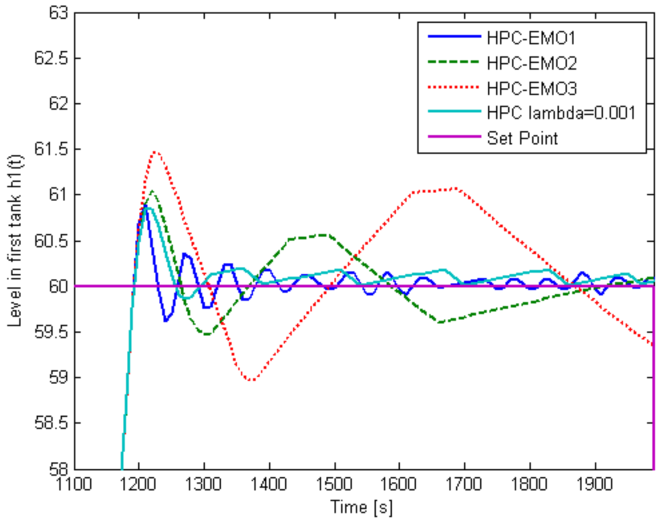


Figure 3.24. Controlled variable.

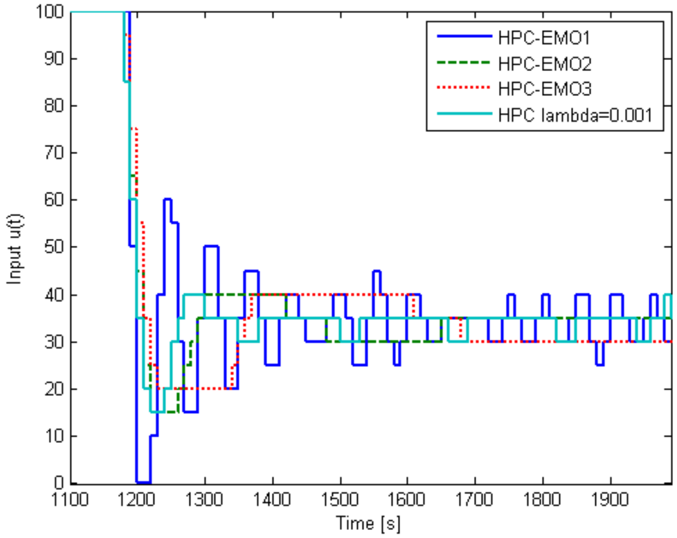


Figure 3.25. Manipulated variable.

Table 3.5 Performance indexes.

	Mean(y-r)^2	Std (y-r)^2	Mean Δu^2	Std Δu^2
HPC-EMO1	4.2864	17.5866	118.7500	389.1165
HPC-EMO2	4.3693	17.5682	19.6023	76.7000
HPC-EMO3	4.6954	17.4941	17.0455	73.4559
HPC λ=0.001	4.2884	17.5685	25.0000	98.6984

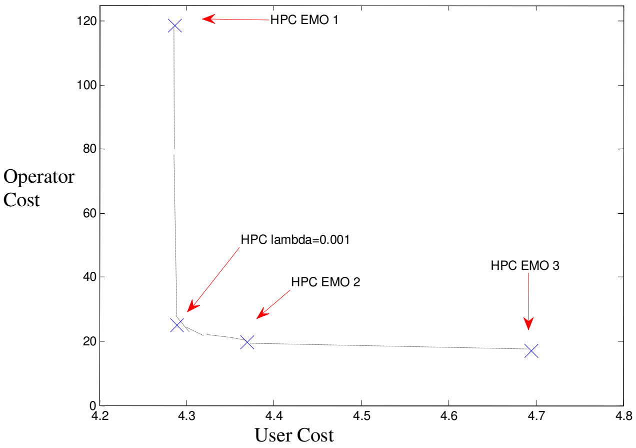


Figure 3.26. Pareto front.

3.4. Emulation of a MO-HPC by tuning a HPC.

Once the MO-HPC is working, a decision process is performed for obtaining one optimal control action. By using this input/output data from a MO-HPC, a conventional HPC is tuned to emulate the behavior of the controller (operator/dispatcher) and so. With an off-line model, a MO-HPC is used to obtain the responses of the system. Based on the dynamic Pareto optimal front, the weight value λ at instant k could be estimated, which connects the MO-HPC solution with the HPC. Then in the real-time application the estimated weighting function $\lambda(k)$ from MO-HPC could be used instead of a fixed value. This also could be interpreted as a new tuning method for the weighting factor of typical MPC.

Once the Pareto Optimal front is obtained as a function of instant k (dynamic front), the equivalent HPC problem is obtained by identifying the weighting factor $\lambda(k)$. Provided that an analytical solution of the Pareto front is not available, two methods are proposed in order to estimate the $\lambda(k)$ factor:

A) LS Method.- By non-linear regression or least mean squares, to estimate an analytical function of the Pareto front using non-linear regression. After that, at the optimal solution chosen from Pareto front, the slope of this analytic function is obtained and it is related with the $\lambda(k)$ factor.

B) IM Method.- In this case, first a range of possible $\lambda(k)$ is determinated considering if (J_1*, J_2*) is the selected Pareto front point and the following inequalities have to be satisfied:

$$\lambda \geq 0\,,\ \forall (J_1, J_2) \in P_F, J_1 + \lambda J_2 \geq J_1 * + \lambda J_2 * \tag{3.13}$$

After that, $\lambda(k)$ is equal to the minimum λ that satisfies equation (3.13). Once $\lambda(k)$ is obtained by using one of these two alternatives and registered for a time period. After that, a model for $\lambda(k)$ could be identified and will provide a tuning method at every instant for a HPC. Thus, a conventional HPC is proposed with a weighting factor $\lambda(k)$ tuned from the multi-objective problem (MO-HPC). The method is explained here to emulate a MO-HPC although it could be applied for the emulation of any controller.

Next the dynamic Pareto front is shown for HPC-EMO2 in the instants range between 1000 and 2000 [*s*] (Figure 3.27). For this problem, the Pareto front has different shapes at every instant *k* as shown in Figure 3.28.

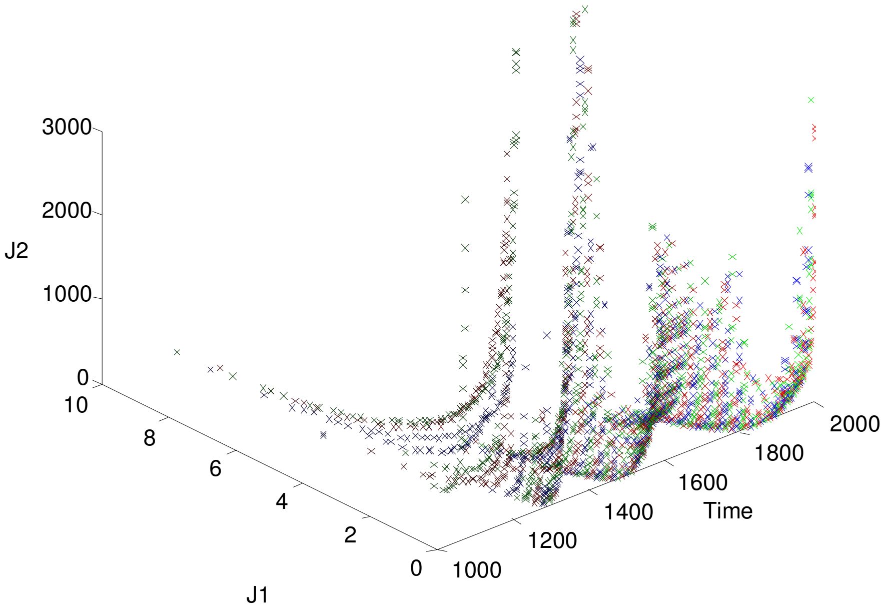


Figure 3.27. Dynamic Pareto front, HPC-EMO2.

From figure 3.28 and using the analytical LS method described in A), assume that at every instant *k*, the Pareto front belongs to the family of curves $J_2 = a_k \cdot J_1^{-b_k}$, with a_k and b_k being positive constants parameters at instant *k*. The slope of those curves, evaluated at the optimal objective function values, provides and estimation of *λ(k)* given by:

$$J_2\text{'} = -a_k b_k \cdot J_1^{-b_k - 1} = -\frac{1}{\lambda(k)} \tag{3.14}$$

Parameters a_k and b_k are obtained by least mean squares at every instant *k*. Also, few Pareto dominant solutions at some instants are observed (see figure 3.28, instant 4, 5 and 200). That happens when the optimization problem either has activated constraints or the control algorithm has converged. In those cases (P_F have 1 or 2 elements), the IM method B) is considered in order to obtain the *λ(k)* values.

Figure 3.29 shows the function *λ(k)* for HPC-EMO2, determined based on a LS method A) and based on the IM method B). Note that both estimations are similar. Figure 3.30 shows the evolution of the tracking error $\left|e(k)\right|$ and the control effort $\left|\Delta u(k-1)\right|$. From figures 3.29 and 3.30, it is possible to realize that there is a relationship between $\lambda(k)$ and $\left|e(k)\right|$, $\left|\Delta u(k-1)\right|$ at every instant. Thus, two options are proposed to tune the *λ(k)*:

1) By least mean squares based on historical data, to identify the parameters of the following proposed linear model:

$$\lambda(k) = \theta_1 \lambda(k-1) + \theta_2 \left|e(k)\right| + \theta_3 \Delta u(k-1).$$

2) *λ(k)* is chosen fixed and equals to the mean value of the observed signal.

Table 3.6 shows the mean value of *λ(k)* and the parameters θ_1, θ_2 and θ_3 of the linear model (option *1*), obtained for each criterion based on analytical *λ(k)* by using LS method. Table 3.7 also shows the parameters when *λ(k)* is obtained using IM method and option *2)*. Figure 3.31 shows *λ(k)* and $\hat{\lambda}(k)$ obtained based on LS (A) and IM method (B) for HPC-EMO2.

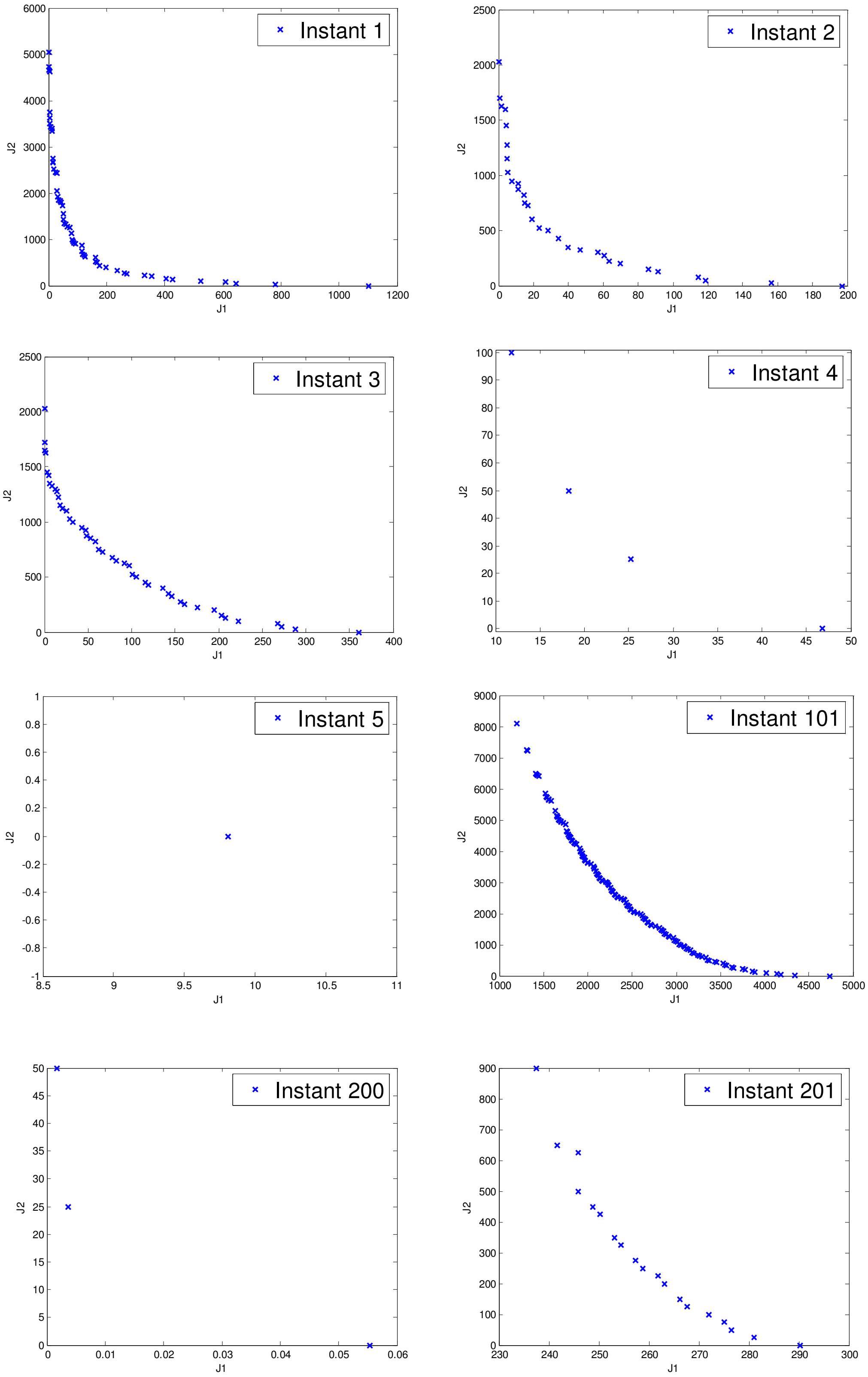


Figure 3.28. Dynamic Pareto front, HPC-EMO2. Each figure represents the Pareto front at one instant.

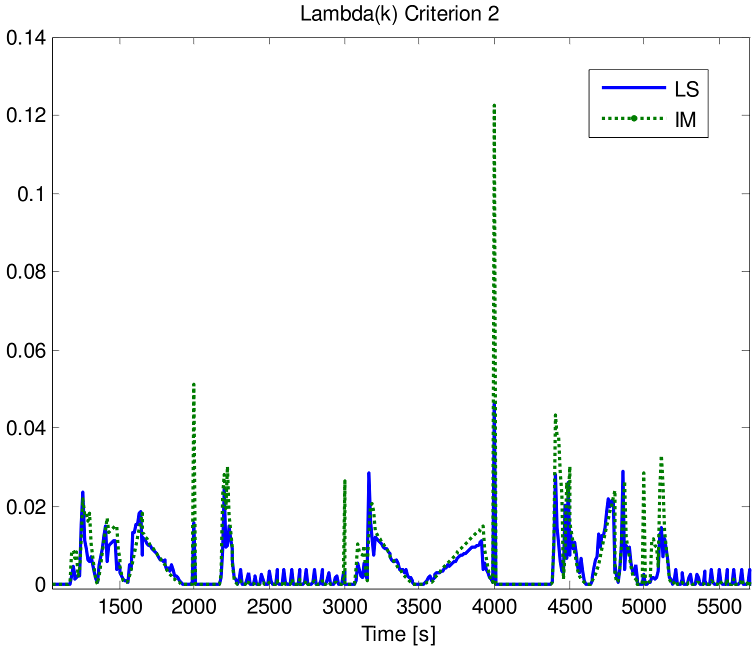


Figure 3.29. Lambda, HPC-EMO2. LS and IM.

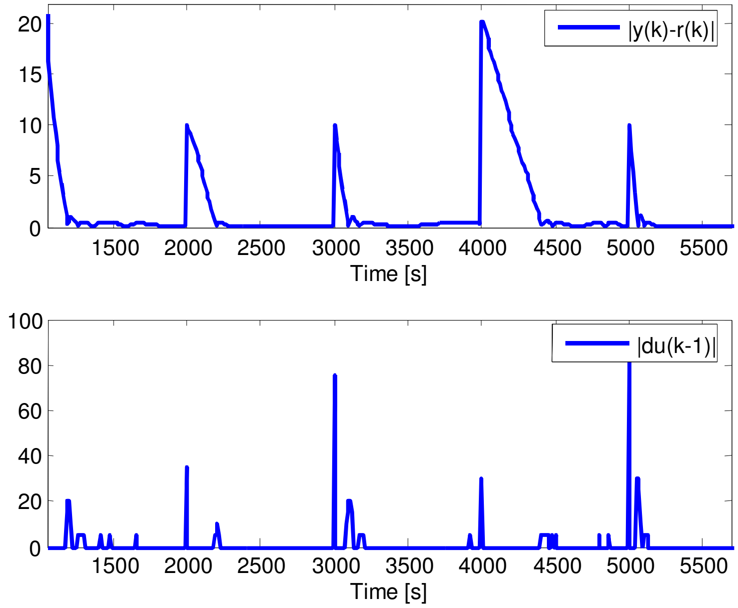


Figure 3.30. HPC-EMO2. Tracking error $|y(k)-r(k)|$, control effort $|\Delta u(k-1)|$ indexes.

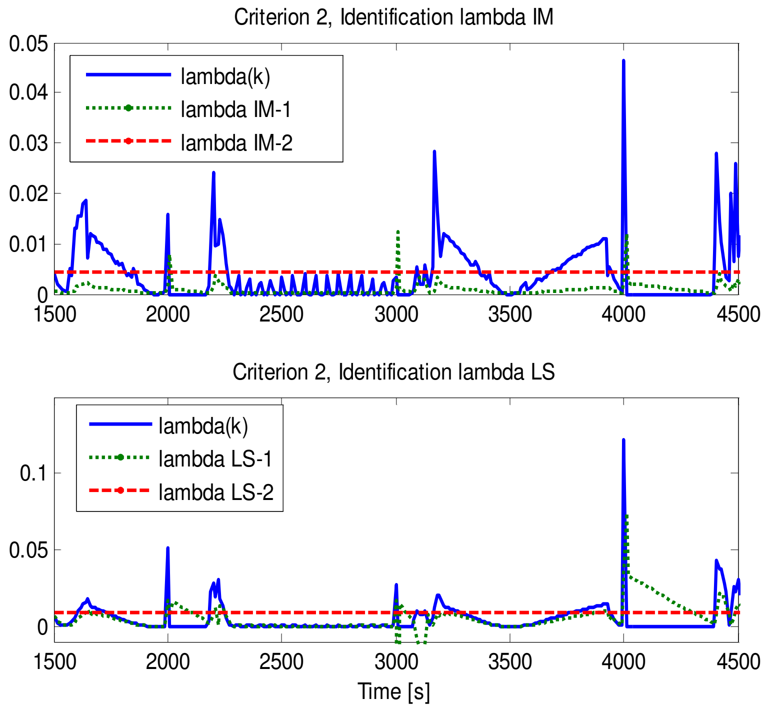


Figure 3.31. Evolution of *λ(k)* for HPC-EMO2. LS-1: LS method with option 1), LS-2: LS method with option 2), and IM-1: IM method with option 1), IM-2: IM method with option 2).

Table 3.6. LS method. Mean values of *λ(k)* and parameters for the linear model

	Mean(λ(k))	θ_1	θ_2	θ_3
HPC-EMO 1	4.2864	17.5866	118.7500	389.1165
HPC-EMO 2	4.3693	17.5682	19.6023	76.7000
HPC-EMO 3	4.6954	17.4941	17.0455	73.4559

Table 3.7. IM method. Mean values of *λ(k)* and parameters for the linear model

	Mean(λ(k))	θ_1	θ_2	θ_3
HPC-EMO 1	0.0074	0.27276	0.0018884	-0.000107
HPC-EMO 2	0.0086	0.62658	0.0016658	-0.001209
HPC-EMO 3	0.0182	0.62506	0.62506	-0.001268

Figure 3.32, Figure 3.33, Figure 3.34 and Figure 3.35 show the system responses using the conventional HPC algorithm with the tuned lambda obtained from HPC-EMO2.

Table 3.8 shows mean values of tracking error and control effort of HPC using *λ(k)* obtained from HPC-EMO2 with a fixed reference. From Table 3.8, the LS method (A) gives better results than the IM method (B) due the solutions are very close to the HPC-EMO2.

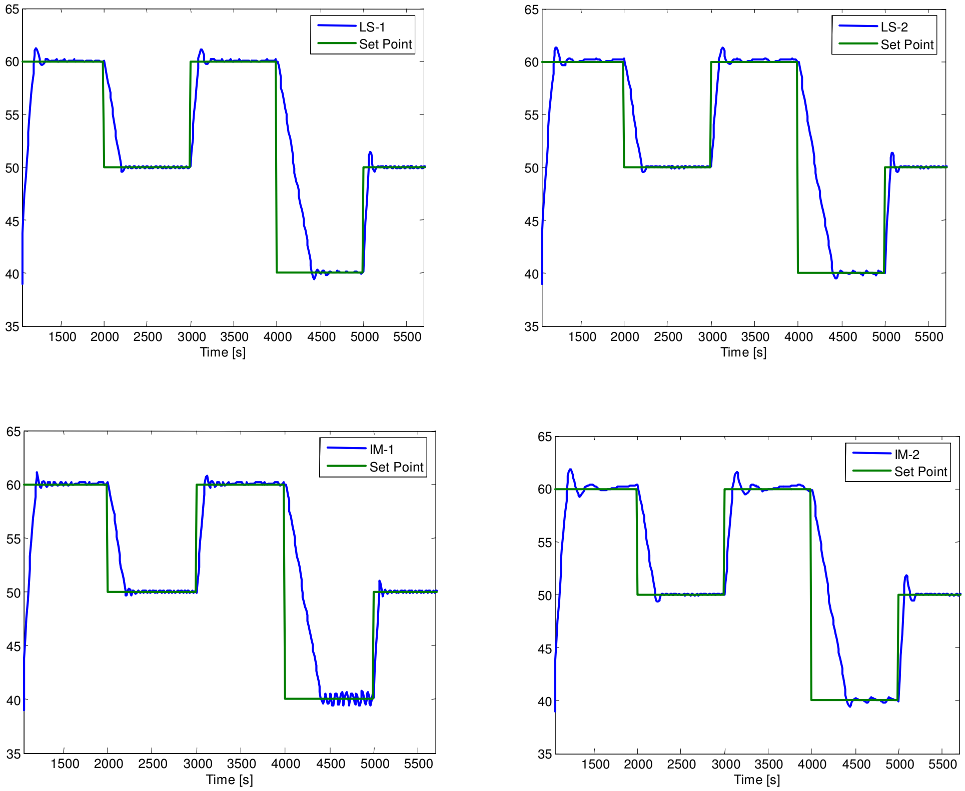


Figure 3.32. Controlled variable, LS-1, LS-2, IM-1 and IM-2.

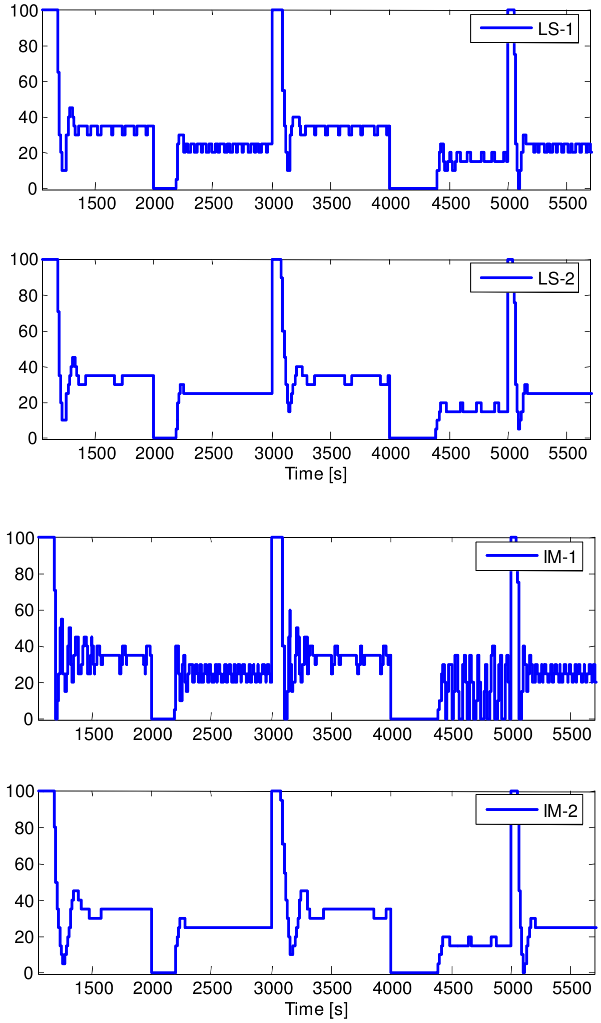


Figure 3.33. Manipulated variable, LS-1, LS-2, IM-1 and IM-2.

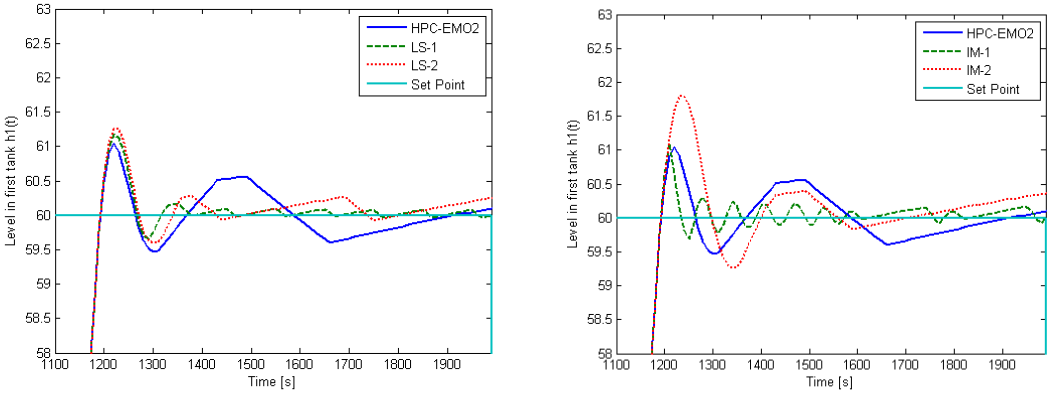


Figure 3.34. Controlled variable, LS-1, LS-2, IM-1 and IM-2.

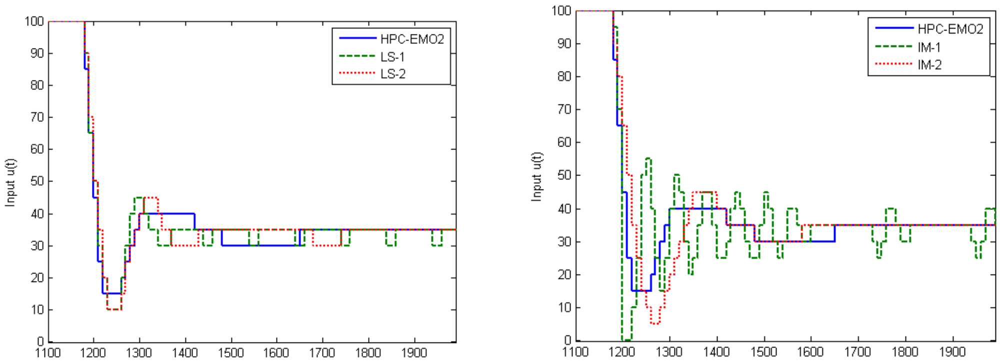


Figure 3.35. Manipulated variable, LS-1, LS-2, IM-1 and IM-2.

Table 3.8 Mean values of tracking error and control effort, HPC-EMO2.

	Mean(y-r)^2	Std (y-r)^2	Mean Δu^2	Std Δu^2
HPC-EMO2	4.3693	17.5682	19.6023	76.7000
LS-1	4.3213	17.5792	25.8523	83.9445
LS-1 λ=0.0042	4.3504	17.5727	20.7386	69.5037
IM-1	4.2925	17.5856	108.5227	527.3264
IM-2 λ=0.0086	4.5085	17.5448	16.1932	45.1752

3.5. Discussion.

The optimization of the predictive objective function is an NP-Hard problem in the case of hybrid non-linear systems, which can be efficiently solved by branch and bound and genetic algorithms. The proposed HPC-GA control algorithm was successfully tested on the hybrid tank system in terms of accuracy and computation time. In a comparison with an optimal explicit-enumeration method and the Branch-and-bound method it is shown that the proposed method gives comparable reference-tracking results in a considerable reduction of the computational load. This characteristic of GA will be very useful in the applications of HPC for transport systems, such as the dynamic pick-up and delivery problem (designed to handle a dial-a-ride system with real-time requirements) and its combination with other fixed-route transit systems.

In such operation schemes, quick on-line responses are required for an efficient operation, and the trade-off between computation time and quality of the solutions is very important as current technology does not permit to solve large instances ensuring reaching the global optimum in an adequate computation time. Other evolutionary algorithms for efficient optimization such as PSO could also be investigated, and the convergence or trade/off with computation time of those algorithms.

This chapter presents a new approach of the Hybrid Predictive Control problem using the Evolutionary Multi-objective Optimization. Two different criteria are proposed in order to obtain an optimal control action from the Pareto front. Both criteria are directly related to the tracking error and control effort measurements. This fact could be an efficient tool for the controller designers in real time plants instead of the typical Model Predictive Control.

Thus, a tuning method for finding the weighting factor of typical MPC based on the EMO solution was proposed. In this case, two alternatives are considered to obtain the weighting values and it is concluded that the model of the Pareto front identified through last mean squares gives the best results.

Further work will be focused on the generalization of the multi-objective predictive control design. In chapter 5 the same MO concepts are applied to the aforementioned transport problem (dial-a-ride system) where the identified trade-off has physical meanings, in terms of the operator who pursues the minimization of its operational expenses on one hand, and the users who want to maximize their level of service by means of low waiting and travel times on the other.

3.6. References

Alvarez, J., Cruz, C. (1998). "Multiobjective Dynamic Optimization of Discrete-time Systems". Proceedings of the 37th IEEE Conference on Decision & Control, Tampa, Florida USA.

Bäck, T., (2000). "An Overview of Parameter Control Methods by Self-adaptation in Evolutionary Algorithms". Annales Societatis Mathematicae Polonae. Series 4: Fundamenta Informaticae 1998, Vol. 35, nr 1-4, s.51-66.

Baric, M., Grieder, P., Baotic, M. and Morari, M., (2007). "An Efficient Algorithm for Optimal Control of PWA Systems with Polyhedral Performance Indices". Automatica, doi: 10.1016/j.Automatica.2007.05.005

Bemporad, A., Morari, M., (1999). "Control of Systems Integrating Logic, Dynamics and Constraints". Automatica, Vol. 35, pp. 407-427.

Bemporad, A., Morari, M., (2000). "Predictive Control of Constrained Hybrid Systems," Automatica, pp. 71–78.

Bemporad, A., Borrelli, F., Morari, M., (2002). "On the Optimal Control Law for Linear Discrete Time Hybrid Systems". Proc. 5th International Workshop, Hybrid Systems: Computation and Control, 25 - 27 March 2002, Stanford, California, USA, Tomlin C. J. and Greenstreet M. R. (eds.), Lecture Notes in Computer Science, no. 2289, pp. 105 - 119, Springer Verlag.

Borrelli, F., Baotic, M., Bemporad, A., Morari, M., (2005). "Dynamic Programming for Constrained Optimal Control of Discrete-time Linear Hybrid Systems". Automatica (41), pp 1709-1721.

Coello, C.A.C. (2002). "Theoretical and Numerical Constraint Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art". Computer Methods in Applied Mechanics and Engineering 191, 1245-1287.

Coello, C., Becerra, R., (2003). "Evolutionary Multiobjective Optimization using a Cultural Algorithm". Swarm Intelligence Symposium, April, pp. 6-13.

Coello, C., Pulido, G., Kendall, G., (2004). "Handling Multiple Objectives with Particle Swarm Optimization". IEEE Transactions on Evolutionary Computation. Volumen 8, issue 3, pages 256-279.

Deb, K., Agrawal, S., Pratab, A., Meyarivan, T., (2000). “A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II”. Indian Inst. Technol., Kanpur, India, KanGAL Rep. 200001.

Farina, M., Deb, K., Amato, P., (2004). “Dynamic Multiobjective Optimization Problems: Test Cases, Approximations, and Applications,” in IEEE Transactions on Evolutionary Computation, Vol 8, No 5, pp. 425-430.

Fletcher, R., Leyffer, S., (1995). “Solving Mixed Integer Nonlinear Programs by Outer Approximation”. Math. Progr., 66(3): 327.

Flores, A., Sáez, D., Araya, J., Berenguel, M., Cipriano, A., (2005). “Fuzzy Predictive Control of a Solar Power Plant”. IEEE Transactions on Fuzzy Systems, Vol. 13, Nº 1, pp. 58-68.

Floudas, C., (1995). “Non-linear and Mixed Integer Optimization”. Oxford University Press, 1995.

Goh, C.K., Tan, K.C., Liu, D.S., Chiam, S.C., (2010). “A Competitive and Cooperative Co-evolutionary Approach to Multi-objective Particle Swarm Optimization Algorithm Design”. European Journal of Operational Research, Volume 202, Issue 1, Pages 42-54.

Hu, X., Eberhart, R., (2002). “Multiobjective Optimization Using Dynamic Neighorhood Particle Swarm Optimization”. Evolutionary Computation 2, pp.1677-1681.

Hu, Z., Chan, C., Huang G., (2007). “Multi-objective Optimization for Process Control of the In-situ Bioremediation System Under Uncertainty”. Engineering Applications of Artificial Intelligence Vol. 20, pp. 225-237.

Karer, G., Mušič, G., Škrjanc, I., Zupančič, B., (2007). “Hybrid Fuzzy Model-based Predictive Control of Temperature in a Batch Reactor”. Computers & Chemical Engineering, Volume 31, Issue 12, December 2007, Pages 1552-1564.

Kerrigan, E.C., Bemporad, A., Mignone, D., Morari,, M., Maciejowski, J.M., (2000). “Multi-objective Priorisation and Reconfiguration for the Control of Constrained Hybrid Systems”. Proceedings of the American Control Conference, Chicago Illinois. Page 1694-1698.

Kerrigan, E.C., Maciejowski, J.M., (2002). “Designing Model Predictive Controllers with Prioritised Constraints and Objectives,” in Computer Aided Control System Design, 2002. Proceedings. 2002 IEEE International Symposium on 18-20 Sept. 2002 Page(s): 33 – 38.

Kimbrough , S., Koehler, G., Lu., M., Wood, D., (2008). “On a Feasible–Infeasible Two-Population (FI-2Pop) Genetic Algorithm for Constrained Optimization: Distance Tracing and No Free Lunch”. IEEE Transactions on Evolutionary Algorithms, Pages 310-327.

Knowles, J., (2006). “ParEGO: A Hybrid Algorithm with On-line Landscape Approximation for Expensive Multiobjective Optimization Problems”. IEEE transactions on Evolutionary Computation. Volumen 10, issue 1, pages 50-66

Labidi K. and Bouani F. (2004) “Genetic Algorithms for Multiobjective Predictive Control”, Proceedings of the 2004 IEEE International Symposium on Control, Communications and Signal Processing, pp 149-152.

Man, K., Tang, K., Kwong, S., (1998). “Genetic Algorithms, Concepts and Designs”. Springer.

Michalewicz, Z., Nazhiyath, G., (1995). “Genocop III: A Co-evolutionary Algorithm for Numerical Optimization with Nonlinear Constraints”. In Fogel, D.B., ed.: Proceedings of the Second IEEE International Conference on Evolutionary Computation, Piscataway, New Jersey, IEEE Press (1995) 647-651.

Michalewicz, Z., Schoenauer, M., (1995b). “Evolutionary Algorithms for Constrained Parameter Optimization Problems”. Evolutionary Computation 4(1) , pp. 1-32.

Na, M.G., Upadhyaya, B.R. (2006). “Application of Model Predictive Control Strategy based on Fuzzy Identification to an SP-100 Space Reactor”. Annals of Nuclear Energy, vol. 33, no. 17-18, pp. 1467-1478.

Nunez-Reyes, A., Scheffer-Dutra S.C., Bordons, C., (2002). “Comparison of Different Predictive Controllers with Multiobjective Optimization. Application to an Olive Oil Mill,” in Proceedings of the 2002 International Conference on Control Applications. Volume 2, 18-20 Sept. 2002 Page(s):1242 - 1247 vol.2

Potocnik, B., Music, G., Zupancic, B., (2004). “Model Predictive Control Systems with Discrete Inputs,” in Proc.12th IEEE Mediterranean Electrotechnical Conference, Dubrovnik, Croatia, 2004, pp:383–386.

Raman, R., Grossmann, I.E., (1991). “Relation Between MILP Modelling and Logical Inference for Chemical Process Synthesis”. Computers and Chemical Engineering 15, p. 73.

Rudolph, G. (1994). “Convergence Analysis of Canonical Genetic Algorithms”. IEEE Trans. Neural Networks, vol. 5, no. 1, pp. 96-101.

Sarimveis, H., Bafas, G. (2003). "Fuzzy Model Predictive Control of Non-linear Processes Using Genetic Algorithms". Fuzzy Sets and Systems, vol. 139, pp. 59-80.

Slupphaug, O., Vada, J., Foss, B., (1997). "MPC in Systems with Continuous and Discrete Control Inputs," in Proc. of American Control Conference, Alburquerque, NM, USA.

Slupphaug, O., Foss, B., (1997). "Model Predictive Control for a Class of Hybrid Systems," in Proc. of European Control Conference, Brussels, Belgium.

Subbu, R., Bonissone, P., Eklund, N., Weizhong, Y., Iyer, N., Feng, X., Rasik, S., (2006). "Management of Complex Dynamic Systems based on Model-Predictive Multi-objective Optimization". Computational Intelligence for Measurement Systems and Applications, Proceedings of 2006 IEEE International Conference, Page(s):64 – 69.

Tan, K., Yang, Y., Lee, T., (2003). "A Distributed Cooperative Coevolutionary Algorithm for Multiobjective Optimization". Proceedings of Congress on Evolutionary Computation CEC 2003, Vol. 4, 8-12 Dec. 2003, pp. 2513-2520.

Thomas, J., Dumur, D., Buisson, J., (2004). "Predictive Control of Hybrid Systems Under a Multi-mld Formalism with State Space Polyhedral Partition". In Proc. of American Control Conference, Boston, Massachusetts, USA.

Van der Lee, J.H., Svrcek, W.Y., Young, B.R. (2008). "A Tuning Algorithm for Model Predictive Controllers based on Genetic Algorithms and Fuzzy Decision Making". ISA Transactions, vol. 47, pp. 53-59.

Yano, H., Sakawa, M., (2009). "A Fuzzy Approach to Hierarchical Multiobjective Programming Problems and its Application to an Industrial Pollution Control Problem". Fuzzy Sets and Systems, Volume 160, Issue 22, 16, Pages 3309-3322.

Zambrano, D., Camacho, E., (2002). "Application of MPC with Multiple Objective for a Solar Refrigeration Plant," in Proceedings of the 2002 International Conference on Control Applications, Volume 2, 18-20 Sept. 2002 Page(s):1230 - 1235 vol.2.

Zhang, L., Zhou, C., Liu, X., Ma, Z., Liang, Y., (2003). "Solving Multiobjective Optimization Problems Using Particle Swarm Optimization". Evolutionary Computation, Vol. 4, pp. 2400-2405.

4. Hybrid Predictive Control for a dial-a-ride system.

4.1. Literature review.

The dynamic pick-up and delivery problem (DPDP) can be formulated as a set of service requests (characterized by pick-up and delivery loads, time windows and spatial coordinates) served by a fleet of vehicles located initially at several depots (Desrosiers *et al.*, 1986, and Savelsbergh and Sol, 1995). The dynamic dimension appears when a subset of the requests is unknown in advance and most dispatch decisions have to be made in real-time. The DPDP is of great interest for practitioners, mainly due to the fast growth in communication and information technologies, as well as the current interest in real-time dispatching and routing.

In the literature, dynamic vehicle routing problems (dynamic VRP) are formulated assuming that inputs may change or have to be updated during the execution of the solution algorithm. Within this family of problems, the DPDP has been designed to solve the dynamic dial-a-ride problem (DARP), which has been intensely studied in the last 20 years (Psaraftis, 1980, 1988, Gendreau *et al.*, 1999 and Kleywegt and Papastavrou, 1998). The final output of such a problem is a set of routes for all vehicles, which dynamically change over time. With regard to real applications Madsen *et al.* (1995) adapt the insertion heuristics by Jaw *et al.* (1986) and solve a real-life problem for moving elderly and handicapped people in Copenhagen, while Dial (1995) proposes a modern approach to many-to-few dial-a-ride transit operation ADART (Autonomous Dial-a-Ride Transit), currently implemented in Corpus Christi, TX, USA.

With regard to other interesting dynamic VRPs, dynamic TSP (DTSP) introduced by Psaraftis (1988) is firstly mentioned. This work motivates the dynamic travelling repairman problem (DTRP), defined by Bertsimas and Van Ryzin (1991) and next extended in Bertsimas and Howell (1993). Lately Swihart and Papastavrou (1999), and Thomas and White (2004) formulate and solve two variants of the DTRP. Kleywegt and Papastravrou (1998) and (2001), Papastravrou *et al.* (1996) study a problem called the dynamic and stochastic knapsack Problem (DSKP), in which demands for a given resource occur according to some stochastic process. Larsen (2000) develops a nice review of the different dynamic problems. Eksioglu *et al.* (2009) and Berbeglia *et al.* (2009) present a recent review of dynamic pick-up and delivery problems, where general issues as well as solution strategies are described. They conclude that is necessary to develop more studies on policy analysis associated with dynamic many-to-many pick-up and delivery problems

There are several key aspects for improving the efficiency of a real implementation behind a DPDP instance (see Crainic *et al.*, 2009). Fundamentally, it is crucial to utilize a correct definition of a decision objective function for dispatching, including total travel and waiting times for users as well as a performance measure for vehicles (proxy of operational costs). When the problem is dynamic, a proper objective function must consider prediction of both future demand and expected waiting and travel times experienced by customers in the system due to potential rerouting decisions decided in the future. This last issue has been mostly underestimated in the dynamic vehicle routing literature, restricting the development of algorithms to myopic models (current decisions not affected by unknown future demand events). In dynamic as well as stochastic problems, the way in which the current decision considers future information of the system differentiates the approaches as being myopic and non-myopic. The myopic research line considers only the current information, i.e, it does not consider explicitly the expected future information of the system to improve the current solution, while the non-myopic option consider a mechanism to update information regarding the future to take better decisions at present. Such future data may be imprecise or unknown, and therefore developing consistent information update tools are essential for getting good predictions and take better real-time dispatch decisions.

Nevertheless, there exists some relevant literature in the field of vehicle routing and dispatching (of both freight and passengers) trying to exploit information about future events to improve decision-making (Ichoua *et al*., 2006, 2007 and Spivey and Powell, 2004). Solution approaches found in this research line are diverse, with formulations based upon dynamic network models (see Powell, 1988), dynamic and stochastic programming schemes (Godfrey and Powell, 2002, Topaloglu and Powell, 2005), etc.

Powell and his team have worked for many years in a non-myopic line of research that incorporates explicit stochastic and dynamic algorithms with the current information and probabilities of future events to produce more efficient solutions than those obtained through myopic deterministic strategies. They solve the problem of dynamically assigning drivers to loads that arise randomly over time motivated from long-haul truckload trucking applications. Powell (1988) first considers the potential advantages of relocating vehicles in anticipation of future demands. He writes a two-stage stochastic program including a recourse function representing the future cost. Spivey and Powell (2004) propose a very general class of dynamic assignment models, and propose an adaptive, non-myopic algorithm that iteratively solves sequences of assignment problems. Topaloglu and Powell (2005) propose a distributed solution approach to a certain class of dynamic resource allocation problems.

Larsen (2000) in his thesis investigated the use of future information, by relocating empty vehicles in anticipation to future demands. Ichoua *et al.* (2005), develop a strategy based on probabilistic knowledge about future request arrivals to better manage the fleet of vehicles for real-time vehicle dispatching, and is solved using a parallel tabu search technique.

Besides, Cortés and Jayakrishnan (2004) and Cortés (2003) realize that the problem could be modelled under a model based predictive control scheme (MPC), considering that potential rerouting of vehicles could affect the current dispatch decisions, through the extra cost of inserting real-time service requests into predefined vehicle routes while vehicles are moving. In this thesis a formulation of the dial-a-ride as a HPC is presented, by stating the state space variables and models. Based on such an approach, a family of solution algorithms is developed based upon artificial intelligence for solving real size instances.

The aforementioned non-myopic vision to deal with the dial-a-ride incorporates an important source of stochasticity in real-time routing decisions, which are the extra travel and waiting time for users as well as an extra operational cost for the dispatch company, due to the insertion of potential customers in the future, unknown at the time of a real-time service decision. However, there is another relevant source of stochasticity that could affect dynamic routing decisions, mainly in the context of urban transport systems. That is, the uncertainty behind the traffic network conditions, interfering the operation of the vehicles under the dispatch rules. This new source of uncertainty has not been treated extensively in the literature associated to dynamic routing problems, mainly because of the computational complexity arising from the resulting formulations. Nevertheless, lately some interesting research effort for adding traffic congestion into dynamic as well as probabilistic/stochastic vehicle routing problems is worth to mention.

Berman and Simchi-Levi (1989) considers a variant of the probabilistic travelling salesman problem (PTSP), including a random subset of customers requiring service and random travel times as well. With regard to stochastic vehicle routing problems, Kao (1978), Sniedovich (1981), and Carraway (1989) solve the stochastic TSP, considering arcs having independent and normally distributed travel times. Laporte *et al*. (1992) study the stochastic vehicle routing problem with stochastic travel as well as service times. They solve instances on networks with 10 to 20 nodes and 2 to 5 scenarios. Lambert *et al*. (1993) solve an optimization of collection routes through bank branches in a network with stochastic travel times. Keyton and Morton (2003) also solve stochastic vehicle routing problems on a network with random travel and service times, by using a branch-and-cut scheme within a Monte Carlo sampling-based procedure. Most of the work described above is based on static models that do not reoptimize routes after realizing the random parameters.

With regard to VRPs including traffic conditions, Hill and Benton (1992) define the nodes of the road network with time-dependent piecewise constant speeds and compute the travel time on a link from the average speed of the incident nodes. Malandraki and Daskin (1992) formulate a mixed integer optimization problem for the VRP with time windows (VRPTW) and piecewise constant travel times, which is solved via heuristic methods.

There are just a couple of examples of dynamic VRPs, in which routes can be modified in real-time from updated information of travel time on links and some prediction of the system based upon updated data. Fleishmann *et al*. (2004) consider a dynamic routing system that dispatches a fleet of vehicles according to customer requests asking for service randomly over a planning period. The authors propose a solution of such a problem, relying on online travel time information from a traffic management center, formulating three routing procedures for event-based dispatching. On the other hand, Kim *et al*. (2005) examines the value of real-time traffic information to optimal vehicle routing in a non-stationary stochastic network. The authors develop optimal routing policies under time-varying traffic flows based on a Markov decision process formulation.

In this thesis, a hybrid predictive control formulation for a DPDP that combines both sources of uncertainty when taking real-time vehicle routing decisions is going to be designed. On the one hand, the formulation will consider uncertainty from possible future demand influencing routes of current customers. Apart from that, it is considered to also add the uncertainty regarding the traffic congestion conditions that could also propose to modify the preplanned schedule of vehicle routes based on traffic information around their routes.

In this approach, traffic congestion is modelled through the distribution of commercial speed of the vehicles on both relevant dimensions: time and space. Traffic conditions of an urban area normally change along the day, and they are different depending on where each vehicle is travelling. It is assumed the availability of real-time as well as historical data regarding several system inputs: demand for service, network speed data obtained from fixed measurement stations as well as mobile stations (vehicles). From this database, it is possible to compute expected demand profiles, and speed distribution profiles over the city, calibrated from historical data.

This approach allows modelling not only predictable congestion conditions, but also unpredictable situations, such as incidents occurring unexpectedly at any location on the traffic network. In the second case, the online (real-time) data regarding speed conditions from the fleet of vehicles moving around serving the demand is also utilized. The present formulation can be extended to the use of fixed stations monitoring traffic conditions at strategically chosen locations over the urban area.

In this thesis, first, the HPC that allows systematizing the formulation of the dial-a-ride system as a control problem, which open more possibilities for using sophisticated techniques, not only to properly characterize the dynamic problem, but also to solve complex DPDP configurations unable to be treated without such a framework. Second, in the specialized literature there is no experience in modelling the DPDP with a HPC formulation allowing prediction of both future demand and future traffic conditions. Third, it is quite attractive (in terms of both computation time and quality solutions) to use solution methods coming from the computational intelligence literature such as GA, Fuzzy logic and others, in the context of this problem.

Moreover, the addition of the speed distribution in the model ensures a better estimation of both waiting and travel times, not only due to demand prediction but also because of traffic congestion predictions, generating better real-time routing decisions, and consequently better performance of the dispatch service. The more information we have regarding the system, the better the performance obtained from the HPC framework.

The HPC approach for the DPDP problem generates a highly non-linear optimization problem, which is NP-Hard. Due to this feature of the problem, it is not feasible in terms of computational time to solve it by using traditional algorithms for mixed-integer problems. Then, Genetic Algorithms (GA) in the way it was explained before in chapter 3 are applied to find good quality solutions for the DPDP problem.

Next, and for the sake of completeness, the recent literature is described in the use of heuristic and metaheuristic methods for solving different kinds of vehicle routing problems (VRP), either dynamic or static.

With regard to solution methods to handle different DVRPs, Gendreau *et al.* (1999) modify the tabu search heuristics to solve the DVRP with soft time windows motivated from courier service applications, which is implemented in a parallel platform. Tabu search methods are derived in more sophisticated versions, such as granular tabu search (Toth and Vigo, 2003) and adaptive memory-based on Tabu search (Tarantilis, 2005). Tighe *et al.* (2004) propose a priority based solver that considers sub problems of a real-time vehicle routing in order to obtain an optimal solution in less time by using fuzzy decisions.

As VRP is NP-Hard, GA based on evolutionary techniques have been analyzed in the specialized literature. Specifically, GA have been applied to different versions of the VRP, considering various chromosome representations and genetic operators according to the particular problem. Skrlec *et al.* (1997) propose a GA optimization approach with handy heuristic techniques for the single VRP that allows further reducing the computation time by using a certain selection of the initial population. In addition, in Filipec *et al.* (1998) the same approach was applied to a multi-vehicle routing problem.

Moreover, Zhu (2003) describes specialized genetic algorithms based on adaptive parameters to solve the static VRP with time windows that prevents the solution search from a premature convergence and improves the results when compared with the typical GA method. Tong *et al.* (2004) considers a GA method for the static VRP with time windows under uncertain fleet size. To solve this problem, a special gene codification associated with the number of vehicles and routes is considered. Haghani and Jung (2005) applied a GA optimization method for the multi-vehicle dynamic VRP with time-dependent travel time and soft time windows. This method provides promising results in terms of computation times.

Jih and Yung-Jen (1999) and Osman *et al.* (2005), present a successful comparison of GA against dynamic programming in terms of computation time. The former method is used to solve the DVRP with time windows and capacity constraints while the latter one is addressed to solve a multi-objetive VRP. Moreover, a hybrid method including both algorithms is described, from which accurate results are obtained in reasonable computation time.

With regard to other heuristics used in the context of the Dynamic VRP, new metaheuristics inspired by the behavior of real ant colonies (Ant Colony Optimization) have been applied to solve such problems (Montemanni *et al.*, 2005; Dréo *et al*, 2006). These methods are especially appropriate to efficiently solve combinatorial optimization problems, and are characterized by the combination of a constructive and a memory-based approach on learning mechanisms (Dorigo and Stutzle, 2004). Montemanni *et al.* (2005) also apply ant colony optimization to a realistic case study that obtains promising results. Dréo *et al.* (2006) present good results for a static VRP by optimizing the fleet size as well as the vehicle route plans.

The two general metaheuristics described above (GA and Ant Colony Optimization) have been applied only on myopic dynamic VRP formulations without considering future demand scenarios for improving current dispatch decisions. In this chapter, an application of GA on a non-myopic formulation for the dynamic VRP (dial-a-ride) is presented, based upon an HPC scheme, i.e., the proposed framework in chapter 3.

In summary, GA is used as an efficient optimization solver for the DPDP problem, where the optimization variables identify the stops that must be satisfied by the vehicle fleet. The individuals are the feasible sequences, fulfilling the load, precedence and no swapping constraints. The gene of an individual considers the following components: the vehicle *j* used for the new insertion and the sequence position of the new call (for both pick-up and delivery) within the previous sequence, assuming the *no-swapping* policy. Due to the precedence and *no-swapping* constraints, the previous sequence is held.

For more than one-step-ahead, GA is conducted for each scenario associated with a specific demand pattern. Previously, the demand patterns are categorized by a fuzzy clustering technique, as detailed before. As GA considers random generation of individuals, the genetic operators (mutation or crossover) could provide infeasible solutions that have to be removed or repaired (typically through the capacity constraint). The number of individuals for each population has to be smaller than the total number of feasible combinations in order to avoid solving the explicit enumeration method.

The complexity of the GA is proportional to the number of individuals for each iteration (generation) multiplied by the number of generations. Both, the number of individuals and the number of generations are parameters to be tuned by the GA designer. The individuals at each iteration are randomly chosen by using genetic operators (selection, mutation and crossover) and the number of generations is stated as the GA stopping criterion. This procedure allows a considerable reduction in computation time providing near optimal solutions.

The proposed closed-loop controlled routing system is shown in Figure 4.1. The hybrid predictive control represented by the dispatcher, takes the routing decisions $S_j(k)$ in real-time based on the information it has from the routing system (process) and the values for the attributes of the vehicle fleet and the transport system (state space variables of the model, like load between consecutives stops, departure time to a stop and position, $L_j(k)$, $T_j(k)$ and $X_j(k)$ respectively). The demand η_k and the traffic conditions ($\varphi(t,p)$) are disturbances (stochasticity). The objective function is influenced by the prediction of the uncertain demand and traffic conditions ($h, p_h(k+\ell)$ and $\hat{v}(t,p)$ respectively).

Then, (i) the control actions are the sequences $S_j(k)$; (ii) the traffic conditions $\varphi(t,p)$ is a disturbance measured by the vehicle, but unmeasured in the whole network; (iii) the demand η_k is a measured disturbance; (iv) the continuous space variables are the departure time $T_j(k)$ and position $X_j(k)$, while the discrete state space variable is the number of passenger (load) $L_j(k)$; (v) the available sensor are located in the vehicles (GPS for the position and the own velocity of the car for the traffic conditions) and the dispatcher receive the calls of users η_k, assign the sequences, they calculate the load, and predict the departures time.

In this chapter, the formulation of the dial-a-ride system under a HPC scheme as proposed by Núñez (2007) is extended to capture the network traffic conditions and provide a more realistic representation of the transport system uncertainty. For doing that, it is necessary to define a set of state space variables, which is used in order to characterize the key elements of the system at certain instant and are needed to provide a formal predictive control formulation to the DPDP problem.

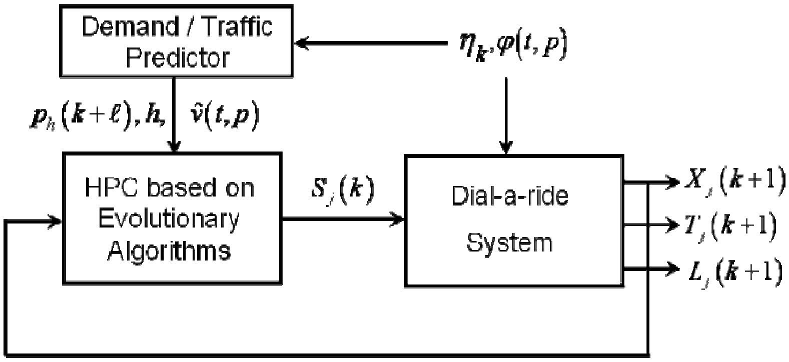

Demand / Traffic Predictor
$\eta_k, \varphi(t,p)$
$p_h(k+\ell), h,$ $\hat{v}(t,p)$
HPC based on Evolutionary Algorithms
$S_j(k)$
Dial-a-ride System
$X_j(k+1)$
$T_j(k+1)$
$L_j(k+1)$

Figure 4.1. Closed-loop diagram of a hybrid predictive approach for DPDP

In this case, three state space variables are considered: departure time, vehicle load at stops and position of the vehicles. The last variable (position of vehicles) is added in order to incorporate the traffic conditions as a function of the network speed distribution. Regarding the objective function, it includes both user and operational costs. The operational cost is approximated by the total vehicle time traveled and the user cost considers both waiting and travel time. The fleet size is assumed known, and the cost function does not include time windows on either pick-up or delivery points.

Next, in Section 4.2 the dynamic model for representing the DPDP is formulated. Then, in Section 4.3, the corresponding objective function formulation is established, completing the presentation with a description of the optimization method in Section 4.4. Results are presented in Section 4.5.

4.2. Modelling a dial-a-ride System.

Let us assume an influence urban area *A* and a fleet of homogenous vehicles of size *F*. The fleet is currently in operation traveling within the area according to predefined routing rules. When a new call for service appears, a selected vehicle is then routed in order to insert the new request into its predefined route. The procedure to find the optimal vehicle-request assignment requires a proper objective function that depends on predictions of state space variables as described hereafter.

The modeling approach is discrete in time, and the time-steps are triggered whenever a new relevant event happens, such as the occurrence of a real-time request for service demand (namely η_k). The index *k* represents the *kth* instant in the discrete sequence of events. Notice that η_k is unknown, comes up in real-time and can be characterized by two positions, indicating the pick-up and the delivery, the time of the call, a label for the request and by the number of passengers.

In addition, the demand is characterized by four attributes, namely $\eta_k = \left(P_k, r_k, \Omega_k, \tau_k\right)$, which corresponds to the last call and have all the information about the request (position, label, load and time).

At any instant *k*, each vehicle *j* has been assigned to follow a sequence of tasks that include pick-ups and deliveries. Such a sequence can be represented by a function $S_j\left(k\right)$ in which the *ith* row represents a specific *ith* stop along vehicle *j*'s route, and $w_j(k)$ is the number of scheduled stops. The manipulated variable corresponds to the set of sequences $u\left(k\right)=S\left(k\right)=\left\{S_1\left(k\right),...,S_j\left(k\right),..,S_F\left(k\right)\right\}$ associated with all the vehicles in the fleet. The proposed HPC dispatcher selects the optimal sequences based on the minimization of an ad-hoc objective function (as shown in Section 4.3 next). Thus, a sequence of stops assigned to vehicle *j* at time *k*, $S_j\left(k\right)$ is given by:

$$S_j\left(k\right)=\begin{bmatrix} z_j^0\left(k\right) & P_j^0\left(k\right) & r_j^0\left(k\right) & \Omega_j^0\left(k\right) \\ z_j^1\left(k\right) & P_j^1\left(k\right) & r_j^1\left(k\right) & \Omega_j^1\left(k\right) \\ z_j^2\left(k\right) & P_j^2\left(k\right) & r_j^2\left(k\right) & \Omega_j^2\left(k\right) \\ \vdots & \vdots & \vdots & \vdots \\ z_j^{w_j(k)}\left(k\right) & P_j^{w_j(k)}\left(k\right) & r_j^{w_j(k)}\left(k\right) & \Omega_j^{w_j(k)}\left(k\right) \end{bmatrix} \tag{4.1}$$

In expression (4.1), $z_j^i\left(k\right)$ is a binary variable defined at instant *k*, which is equal to 1 if the stop *i* is a pick-up, 0 if the stop *i* is a delivery. $P_j^i\left(k\right)\in R^2$ is a two-dimensional vector that shows the geographical position of stop *i* assigned to vehicle *j* in terms of spatial coordinates *x* and *y*, $r_j^i\left(k\right)$ is a tag to identify the passenger who is calling and $\Omega_j^i\left(k\right)$ is the number of passengers to be transported between the origin and destination associated with request $r_j^i\left(k\right)$. The first row of the sequence of stops in (4.1) represents the initial conditions, which correspond to the last stop already visited by the corresponding vehicle *j*.

Figure 4.2 shows a sequence $S_j\left(k\right)$ assigned to a vehicle *j* at time *k*, which is a picture of the assigned vehicle tasks. $\hat{T}_j^i\left(k\right)$ represents the expected departure time of the vehicle *j* at stop *i*, $\hat{L}_j^i\left(k\right)$ is the expected vehicle load when vehicle *j* leaves stop *i*.

$X_j\left(k,\varphi(t_k)\right)$ is the current position (coordinates) computed at instant time *k* that depends on the traffic conditions $\varphi(t)$. t_k is a variable connecting the continuous time (clock time) with the discrete model in time (index *k*). Notice that $X_j\left(k,\varphi(t_k)\right)$ must be in between $P_j^0\left(k\right)$ and $P_j^1\left(k\right)$. To simplify the notation, hereafter it will use simply denote $X_j\left(k\right)$ to represent $X_j\left(k,\varphi(t\ )\right)$. Notice that the traffic conditions ($\varphi(t)$) affect the current position of each vehicle $X_j(k,\varphi(t_k))$, which is a measurable output of the system. The vehicle position is a random variable, and $X_j(k,\varphi(t_k))$ is a realization of such a variable.

These three types of variables ($\hat{T}_j^i\left(k\right)$, $\hat{L}_j^i\left(k\right)$, $X_j\left(k\right)$) conform the state space vector as described next. Moreover, $L_j^0\left(k\right)$ and $T_j^0\left(k\right)$ are the vehicle conditions when the last call request was satisfied located at $P_j^0\left(k\right)$.

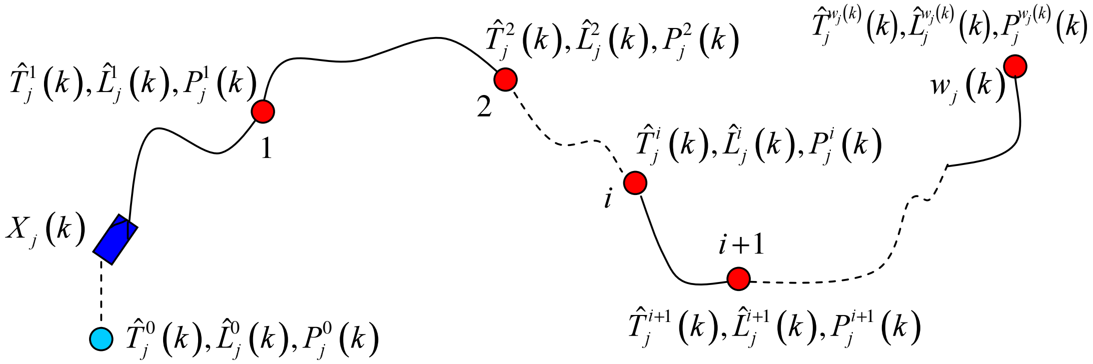

$\hat{T}_j^1\left(k\right),\hat{L}_j^1\left(k\right),P_j^1\left(k\right)$
$\hat{T}_j^2\left(k\right),\hat{L}_j^2\left(k\right),P_j^2\left(k\right)$
$\hat{T}_j^{w_j(k)}\left(k\right),\hat{L}_j^{w_j(k)}\left(k\right),P_j^{w_j(k)}\left(k\right)$
$w_j\left(k\right)$
2
1
$\hat{T}_j^i\left(k\right),\hat{L}_j^i\left(k\right),P_j^i\left(k\right)$
i
$X_j\left(k\right)$
$i+1$
$\hat{T}_j^0\left(k\right),\hat{L}_j^0\left(k\right),P_j^0\left(k\right)$
$\hat{T}_j^{i+1}\left(k\right),\hat{L}_j^{i+1}\left(k\right),P_j^{i+1}\left(k\right)$

Figure 4.2. Vehicle sequence representation

For simplicity, in this application a conceptual network with Euclidean norm as a distance estimator is considered. Although the distance is computed through a fixed measure depending on the coordinates of the initial and final conditions, the modelled travel times on segments experienced by vehicles are not fixed, since the speed is variable.

Analytically for any vehicle j, the state space model is given by:

$$\hat{\chi}_j(k+1) = \begin{bmatrix} \hat{X}_j(k+1) \\ \hat{T}_j(k+1) \\ \hat{L}_j(k+1) \end{bmatrix} = \begin{bmatrix} f_X\left(S_j(k),\hat{v}(t,p),\eta_k\right) \\ f_T\left(X_j(k,\varphi(t_k)),\hat{T}_j(k),S_j(k),\hat{v}(t,p),\eta_k\right) \\ f_L\left(\hat{L}_j(k),S_j(k),\eta_k\right) \end{bmatrix} \tag{4.2}$$

where $\hat{\chi}_j(k)$ is the vector of state space variables defined for vehicle j at next instant $k+1$, as function of the control action $S_j(k)$, the estimators of the disturbances η_k, the speed model $\hat{v}(t,p)$ and the state space variables at instant k, ($\hat{T}_j^i(k), \hat{L}_j^i(k), X_j(k)$).

The estimated departure time vector $\hat{T}_j(k)=\left[T_j^0(k) \;\; \hat{T}_j^1(k) \;\; \cdots \;\; \hat{T}_j^{w_j(k)}(k)\right]^T$ and the estimated load vector is $\hat{L}_j(k)=\left[L_j^0(k) \;\; \hat{L}_j^1(k) \;\; \cdots \;\; \hat{L}_j^{w_j(k)}(k)\right]^T$ are vectors of the same dimension as that of the sequence.

Notice that only the first component of both the expected departure time and expected load vectors at instant k are known, since the remaining components of both vectors are really expectations of what is supposed to happen at the scheduled stops of each vehicle defined in each sequence, which will depend on the expected disturbances along the vehicle routes. Thus, to compute the estimated departure time at stops the predictive model is utilized starting from the current vehicle position $X_j\left(k,\varphi(t_k)\right)$ (continuously being affected by the disturbance $\varphi(t)$). Besides, the expected load as well as the expected departure time at future stops, will also depend on the demand over space and time, from where potential reroutings could affect the future load and departure times at stops.

In the proposed approach, traffic congestion is modeled through the distribution of commercial speed of the vehicles on both relevant dimensions: time and space, since traffic conditions of an urban area normally change along the day, and are different depending on where each vehicle is traveling. The real speed distribution is unknown $v(t,p,\varphi)$ and it depends on a stochastic source that comes from the network traffic conditions $\varphi(t)$ (if the specification is additive, then $\varphi(t)$ will be measured in speed units). Also a known velocity distribution of the urban area during a typical period of recurrent congestion is assumed available based on historical data, which is represented by a model of the speed $\hat{v}(t,p)$. All of them specified in terms of the continuous time *t* and the spatial coordinate *p*. The functions f_X, f_L and f_T in equations (4.2) define the state space model, and are specified in equations (4.3) to (4.6).

First, the dynamic model for the position associated with vehicle *j* is given by

$$\hat{X}_j\left(k+1\right)=P_j^0(k)+\int_{t_k}^{t_k+\tau}\hat{v}\left(t,p(t)\right)\frac{\left(P_j^1(k)-P_j^0(k)\right)}{\left\|P_j^1(k)-P_j^0(k)\right\|_2}dt \qquad (4.3)$$

where $t_k \leq t \leq t_k+\tau$. So, the model requires a variable stepsize (τ), defined by the interval between the occurrence of a probable future call asking for service ($t_k+\tau$) and the occurrence of the previous call t_k. τ is calculated as a tuning parameter for the HPC by using a sensitivity analysis. Note that $P_j^1(k)-P_j^0(k)$ provides the information with regard to the direction of the vehicle *j* speed. If a request is fulfilled, an adaptive mechanism uploads $P_j^0(k)$ since this variable represents always the last stop position already visited, at every instant *t*.

Besides, the departure time vector depends on the vehicle speed, and can be computed as follows:

$$\hat{T}_j\left(k+1\right)=\begin{bmatrix}T_j^0\left(k\right) & t_k+\kappa_j^1(k) & t_k+\sum_{s=1}^{2}\kappa_j^s(k) & \cdots & t_k+\sum_{s=1}^{w_j(k)}\kappa_j^s(k)\end{bmatrix}^T \qquad (4.4)$$

where

$$\kappa_j^1(k)=\int_{X_j(k,\varphi(t))}^{P_j^1(k)}\frac{1}{\hat{v}(t_j(\omega),\omega)}d\omega,\quad \kappa_j^i(k)=\int_{P_j^{i-1}(k)}^{P_j^i(k)}\frac{1}{\hat{v}(t_j(\omega),\omega)}d\omega\quad i=2..w_j\left(k\right) \qquad (4.5)$$

$\kappa_j^i(k)$ is an estimate of the time interval between stop *i-1* and stop *i* in the sequence of vehicle *j*, at time *k*. When *i=1*, the reference for computing the arrival time is the current position of the vehicle instead of the previous stop. $t_j(\omega)$ is the continuous time at which vehicle j reaches position ω. In (2.5), the integration is performed along the line between two consecutives stops.

The dynamics embedded in the vehicle load vector depends exclusively on the current sequence and the previous load variable at instant *k*. Analytically,

$$\hat{L}_j(k+1)=\left[L_j^0(k) \quad L_j^0(k)+\sum_{s=1}^{1}\left(2z_j^s(k)-1\right)\Omega_j^s \quad \cdots \quad \cdots \quad L_j^0(k)+\sum_{s=1}^{w_j(k)}\left(2z_j^s(k)-1\right)\Omega_j^s\right]^T \tag{4.6}$$

with z_j^s and Ω_j^s defined in expression (4.1).

Vehicle sequences as well as state space variables have to satisfy a set of constraints that depend on the real conditions of the modeled DPDP. Specifically, precedence, capacity and consistency constraints are added into the dynamic model to generate only feasible sequences. Those constraints can be written as logical conditions, as follows:

Constraint 1- Constraint of precedence. The delivery of a passenger cannot happen before its pick-up. Then:

If a sequence contains twice the same label, then the first task is the pick-up, and the second is the delivery. So, If $r_j^{i_1}(k)=r_j^{i_2}(k)$, then $z_j^{i_1}(k)=1$ and $z_j^{i_2}(k)=0$.

If a sequence contains just once a given label, then, the task is to deliver the passenger. So, If $\forall i_2 \le w_j(k), i_2 \ne i_1, r_j^{i_1}(k) \ne r_j^{i_2}(k)$, then $z_j^{i_1}(k)=0$.

Therefore, the final node of every sequence has to be a delivery. In short, $z_j^{w_j(k)}(k)=0, \quad \forall j:1...F$.

Constraint 2.- A destination $P_j^i(k)$ must be visited only once, and is assigned to only one label (customer). In fact, every row in a sequence consists in the information of just one user pick-up or delivery point.

Constraint 3.- Consistency. Once a group of passengers get on a specific vehicle, they have to be delivered to the destination by the same vehicle.

Constraint 4.- Capacity load constraint. A vehicle will not be able to carry more passengers than its maximum load, that is $L_j^i(k) \leq L_{\max}$.

All those constraints will be considered once a possible sequence is generated. The controller should provide feasible sequences.

Once the state space variables are analytically defined, the objective function and the optimization procedure are needed, in order to complete the description of the controller. Moreover, the state space models defined in Section 4.2 along with the objective function permit the prediction at one, two and more step-ahead, which are necessary for implementing the HPC control strategy. Next, the objective function is presented and discussed.

4.3. Objective Function.

The request-vehicle assignment is decided by the dispatcher (controller) based on a proper objective function that depends on predictions of the state space variables and consequently, on the future control actions applied to the system. The objective function is specified in terms of both the total expected waiting and travel time for passengers. The idle travel time (vehicles moving around without passengers) is also included in the formulation in order to consider a *proxy* for the operational cost in the decision.

The major issue in the definition of the objective function is to define a reasonable prediction horizon *N*, which depends on the studied problem. A prediction at one-step-ahead is equivalent to performing a myopic assignment, since only the new request (arising at *k*) is considered when taking the routing decision. When a predictive horizon greater than one is assumed, the decision maker (controller) adds the predictive feature into the formulation, since decisions taken at *k* will depend not only upon the new request at *k*, but also on possible events (new service requests unknown at the decision instant *k*) occurring at future instants (*k+1, k+2*, …etc). These new requests are estimated by using fuzzy clustering based on historical demand data.

A set of consecutive expected calls $\left\{\eta_{k+1}^{h},\eta_{k+2}^{h},...,\eta_{k+N-1}^{h}\right\}$ define a trip pattern *h* (note the superscript *h* in the call representation above to join a pattern with the calls associated to it). Thus, the central dispatcher (controller) computes the following set of sequences $S(k)\cup\bigcup_{h=1}^{H}\left\{S(k+1)\big|_{\eta_{k+1}^{h}},...,S(k+N-1)\big|_{\eta_{k+N-1}^{h}}\right\}$, which corresponds to the decisions for the entire control horizon *N* and for each pattern *h*. Then, the dispatcher applies just the next step sequence $S(k)$, based on receding horizon control. It is important to note that $S(k)$ includes the new request to be assigned (η_k), which is known (deterministic) at the decision time. The quality of the dispatcher routing decisions will depend on how well the system predicts the impact of rerouting passengers due to unknown insertions as well as traffic congestion. Notice that deterministic decisions are continuously made by the dispatcher based on the information of each call that enters the system along with a forecast of a future decision corresponding to each possible pattern (scenario).

The objective function for a generic prediction horizon *N*, can be written as follows

$$\underset{S(k)\cup\bigcup_{h=1}^{H}\left\{S(k+1)\big|_{\eta_{k+1}^{h}},...,S(k+N-1)\big|_{\eta_{k+N-1}^{h}}\right\}}{Min}\sum_{j=1}^{F}\sum_{h=1}^{H}p_h\cdot C_j(k+N)\big|_h \tag{4.7}$$

$$C_j(k+N)\big|_h=\sum_{i=1}^{w_j(k+N)}\left(\underbrace{\left(\hat{L}_j^{i-1}(k+N)+1\right)\left(\hat{T}_j^{i}(k+N)-\hat{T}_j^{i-1}(k+N)\right)}_{\text{J travel time}}+\underbrace{z_j^{i}(k+N-1)\alpha\left(\hat{T}_j^{i}(k+N)-T_j^{0}(k+N)\right)}_{\text{J waiting time}}\right)\Bigg|_h \tag{4.8}$$

where $C_j(k+N)\big|_h$ in (4.8) is the cost function of vehicle *j* at instant $k+N$, provided that the trip pattern *h*, characterized by $\left\{\eta_{k+1}^{h},\eta_{k+2}^{h},...,\eta_{k+N-1}^{h}\right\}$, occurs. Such a cost also depends directly of the set of sequences to be applied, namely $\left\{S(k),S(k+1)\big|_{\eta_{k+1}^{h}},...,S(k+N-1,)\big|_{\eta_{k+N-1}^{h}}\right\}$, which are the optimization variables. *H* is the number of trip patterns considered, p_h is the probability of occurrence of the h^{th} trip pattern (future demand). $w_j(k+N)$ is the number of stops estimated for vehicle *j* at instant $k+N$.

The future instants *k+1*, *k+2*, etc. are generated by using a variable time step. Then, the expected call associated with pattern *h*, to happen *N*-steps-ahead is $\eta_{k+n}^{h}=\left(P_{k+n}^{h},r_{k+n}^{h},\Omega_{k+n}^{h},\tau_{k+n}^{h}\right)$, where τ_{k+n}^{h} is the expected occurrence time of such a call in the future. Due to the large number of parameters, the computations are simplified by assuming $\tau_{k+n}^{h}=\tau_{k+n} \quad \forall h$. Besides, $\tau_{k+n}=\tau_{k+n-1}+\Delta\tau$ with $\Delta\tau$ tuned through a sensitivity analysis. Finally, α is a weight for the waiting time to differentiate its contribution compared with that of travel time in the objective function. The number of future demand patterns H and their probabilities of occurrence p_h are parameters in the objective function, and they have to be computed based on either real-time data, historical data, or a combination of both. In this case, fuzzy clustering is used to model the demand $(\hat{\eta}_{k+1})$ by considering only historical data.

Note that in the first component of the objective function expression in (4.8), the expected travel time is weighted by $\hat{L}_{j}^{i-1}\left(k+N\right)+1$. In such a computation, the expected load captures the user cost associated to travel time, while the added *one* roughly incorporates a *proxy* for the operational cost through the total time travelled by vehicles, even though some of them do not carry any passenger on certain segments of their routes.

With regard to the step-size to be used in the prediction, George and Powell (2005), develop and discuss many interesting methods to incorporate a good estimation of optimal step-size (like Kalman Filter and others). None of these methods properly replicated the dial-a-ride conditions, considering that in addition to represent a good estimation of the time between calls, what it is aimed is to calibrate a parameter for optimizing the system performance function over time, in order to get the optimal routing strategy including future information. In order to do that, a sensitivity analysis was conducted from simulated data to find the step-size value that minimizes the objective function for more than one-step-ahead. It is very important to highlight the fact that these variables are continuous and non-optimal behaviour could occur if they are not properly adjusted by sensitivity analysis. For the two-step-ahead application this parameter is denoted by τ , and as discussed above, physically it represents the expected time for a predicted request to happen. However, what τ really represents is the best instant for inserting the future expected call in order to optimize the routing scheme. In general, these parameters are tuneable for each step-ahead of prediction.

In this chapter, it is compared a myopic strategy (one-step-ahead) with the two-steps and three-steps-ahead predictive approaches that includes future information from the system, to show the improvements in routing when considering a predictive component in the routing decisions in a dial-a-ride system.

It will prove for the three cases it deals with in this chapter, that the optimization problem given by (4.7) is equivalent to the following one.

$$\underset{\bar{S}=S(k)\cup\bigcup_{h=1}^{H}\left\{S(k+1)\big|_{\eta_{k+1}^{h}},\ldots,S(k+N-1)\big|_{\eta_{k+N-1}^{h}}\right\}}{Min} \sum_{t=1}^{N}\sum_{j=1}^{F}\sum_{h=1}^{H(k+t)} p_h(k+t)\cdot\left(C_j(k+t)-C_j(k+t-1)\right)\Big|_h \tag{4.9}$$

The one-step-ahead strategy means that the prediction horizon is N = 1, and $H(k+1)=1$ since the new requirement is one and known, and therefore its probability is equal to 1, obtaining the following expression for the objective function using (4.9):

$$\begin{aligned}\underset{\bar{S}}{Min}\ J &= \sum_{t=1}^{1}\sum_{j=1}^{F}\sum_{h=1}^{H(k+t)=1} p_h(k+t)\cdot\left(C_j(k+t)-C_j(k+t-1)\right)\Big|_{S_j(k+t-2),h} \\ &= \sum_{j=1}^{F}\overbrace{p_1\ (k+1)}^{=1}\cdot\left(C_j(k+1)-C_j(k)\right)\Big|_{S_j(k-1),1} = \sum_{j=1}^{F}\left(C_j(k+1)-\overbrace{C_j(k)}^{\text{known constant}}\right)\Bigg|_{S_j(k-1),1}\end{aligned}$$

where

$$C_j(k+1)\Big|_{S_j(k-1),1} = \sum_{i=1}^{w_j(k)}\left\{\underbrace{\left[\hat{L}_j^{i-1}(k+1)+1\right]\left(\hat{T}_j^{i}(k+1)-\hat{T}_j^{i-1}(k+1)\right)}_{\text{J travel time}}+\underbrace{r_j^{i}(k)\alpha\left(\hat{T}_j^{i}(k+1)-T_j^{0}(k+1)\right)}_{\text{J waiting time}}\right\}\Bigg|_{S_j(k-1),1}$$

Note that the difference $\left(C_j(k+1)-C_j(k)\right)\Big|_{S_j(k-1),1}$ is evaluated considering the control action in the previous instant, represented by $S_j(k-1)$. Conceptually, J represents the insertion cost when the system accepts a new call, computed in real-time and considering the entire vehicle fleet. Note the equivalence between the optimization problems (4.7) and (4.9). The only different between them is just a constant, which do not change the optimization problem.

The two-step-ahead prediction's objective function is different from the previous one, since it includes a prediction of where the following call is going to fall, and with which probability. The controller selects the vehicle's sequence that minimizes the general two-step-ahead objective function, which is as follows,

$$\underset{S(k)}{Min}\,J=\sum_{t=1}^{2}\sum_{j=1}^{F}\sum_{h=1}^{H(k+t)} p_h\,(k+t)\cdot\left(C_j\,(k+t)-C_j\,(k+t-1)\right)\Big|_{S_j(k+t-2),h}=$$

$$\sum_{j=1}^{F}\left[\cancel{C_j\,(k+1)\Big|_{S_j(k-1),1}}-C_j\,(k)+\sum_{h=1}^{H(k+2)} p_h\,(k+2)\cdot C_j\,(k+2)\Big|_{S_j(k),h}-\cancel{\overbrace{\sum_{h=1}^{H(k+2)} p_h\,(k+2)}^{=1}\cdot\overbrace{C_j\,(k+1)\Big|_{S_j(k-1),1}}^{\text{Independent of h}}}\right]=$$

$$\sum_{j=1}^{F}\left[\sum_{h=1}^{H(k+2)} \underbrace{p_h\,(k+2)}_{p_h}\cdot C_j\,(k+2)\Bigg|_{S_j(k),h}-\overbrace{C_j\,(k)}^{\text{known constant}}\right]$$

where

$$C_j\,(k+2)\Big|_{S_j(k),h}=$$

$$\sum_{i=1}^{w_j(k+1)}\left\{\underbrace{\left[\hat{L}_j^{i-1}\,(k+2)+1\right]\left(\hat{T}_j^{i}\,(k+2)-\hat{T}_j^{i-1}\,(k+2)\right)}_{\text{J travel time}}+\underbrace{r_j^{i}\,(k+1)\alpha\left(\hat{T}_j^{i}\,(k+2)-T_j^{0}\,(k+2)\right)}_{\text{J waiting time}}\right\}\Bigg|_{S_j(k),h}$$

In the case of the one-step-ahead strategy (myopic), the new requirement is one and known, and therefore its probability is equal to 1. In case of the two-step-ahead prediction, the objective function requires the estimation of probabilities that the new call entering the system two-steps-ahead falls into each demand pattern. A distribution for the time interval between successive calls is also assumed in order to compute time interval probabilities.

Another interesting case, is the three-step-ahead objective function, again computed from the generic expression, as follows:

$$J=\sum_{j=1}^{F}\left[\left(\sum_{h_3=1}^{H(k+3)}\left(\underbrace{\sum_{h_2=1}^{H(k+2)} p_{h_2}\,(k+2)\cdot p_{h_3}\,(k+3)}_{p_h}\right)\cdot C_j\,(k+3)\Bigg|_{S_j(k),h_2,h_3}\right)-\overbrace{C(k)}^{\text{known constant}}\right]$$

For illustrative the proposed methodology as shown figure 4.3, let us concentrate on the three-step-ahead prediction case for an example of two origin-destination pairs at two step, and four at three step, in which the strategy would be to evaluate the following chain of scenarios.

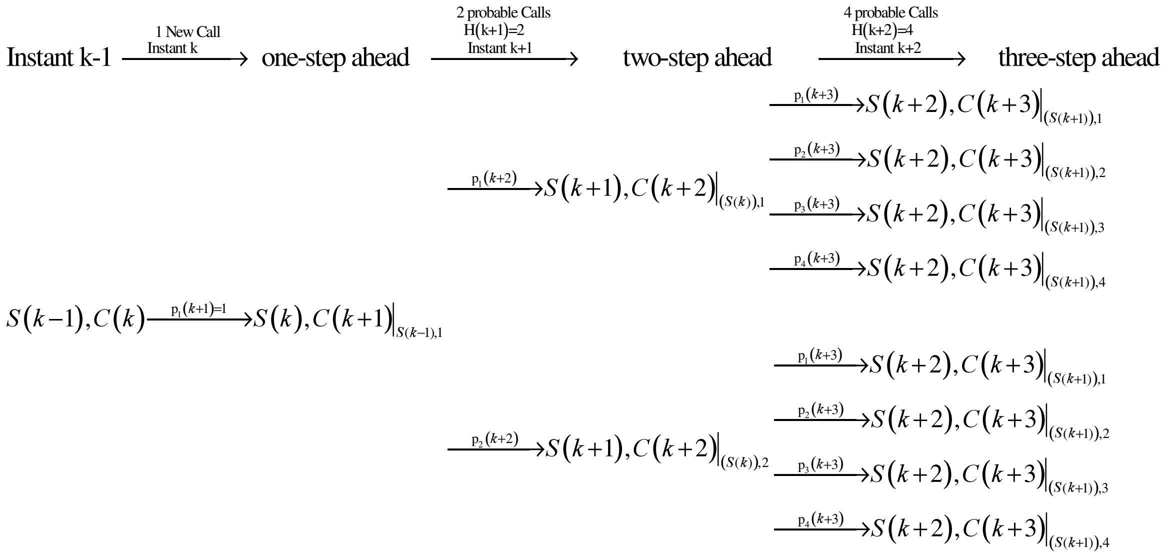


Figure 4.3. Potential combinations of sequences at future.

At instant *k*-1, vehicles follow certain sequence $S(k-1)$ associated with a total cost $C(k)$. Whenever a new service request enters the system, there are several feasible sets of sequences $S(k)$ to be evaluated by the controller (each alternative inserting the new pick-up and delivery in feasible segments of the sequence of a specific vehicle). At one-step-ahead, one call is considered (instant *k* with probability equals to 1). At two-step-ahead, it fixes two potential calls appearing in the next time step *k*+1, with probabilities $\mathrm{p}_1(k+2)$ and $\mathrm{p}_2(k+2)$ respectively. At three-step-ahead, it fixes four potential calls appearing in the next time step *k*+2, with probabilities $\mathrm{p}_1(k+3)$, $\mathrm{p}_2(k+3)$, $\mathrm{p}_3(k+3)$ and $\mathrm{p}_4(k+3)$ respectively in order to incorporate the dynamic nature of the problem, and consequently to have good estimations of both travel and waiting times for the cost function decision. Finally, eight potential cases are evaluated for all possible scenarios, containing three new sequential insertions each (the known new call that comes up at one-step-ahead and the potential calls that appear at two and three-steps-ahead)

In order to perform a good estimation of future scenarios in the objective function expressions, the historical data is analyzed through a systematic methodology for determining the future trip patterns and their corresponding occurrence probabilities. Next, a fuzzy clustering approach is proposed to deal with this issue.

A systematic zoning methodology is developed to split the space into conceptual regions for a better representation of historical demand patterns, which can be obtained from demand data associated with a representative operation day. This proposal turns out to be an alternative a typical classic zoning approach where the total area is divided into homogeneous and not overlapping-areas. The classic zoning approach could perform badly in cases where typical origin-destination patterns do not match any of the predefined pair of zones according to the classic method. In fact, a wrong zoning methodology could impact the computation of probabilities in the objective function for more than two-step-ahead predictions. The systematic zoning proposed here is based on a fuzzy clustering method that allows us to classify the typical origin-destinations calls in representative and flexible clusters. For simplicity and considering the problem features, the fuzzy C-means is adapted to model such a spatial classification.

In this application, the FCM method is used to determine the representative centers associated with historical origin-destination patterns, which will allow us computing the corresponding predictive probabilities.

The probability of each cluster associated with a given origin-destination pair is computed by following the procedure stated next:

Step 1. The fuzzy clusters are obtained from historical demand data by using the FCM method.

Step 2. Membership degrees associated with each call from the historical database are computed for every fuzzy cluster obtained in Step 1.

Step 3. Each call is associated with only one fuzzy cluster, corresponding to that with the biggest membership degree.

Step 4. Calls with a membership degree smaller than a chosen threshold are not considered in the process.

Step 5. A probability of occurrence of a new request on a specific origin-destination pair is computed as the number of calls that belong to a fuzzy cluster divided by the total number of calls (after removing the negligible data as explained in Step 4).

Step 6. Perform a FCM recalculation of cluster center position from historical demand data without considering the negligible data removed in Step 4.

Notice that the optimal number of clusters determines the number of trip patterns for each time period. The number of potential calls (each one occurring with certain probability) for the *N*-step-ahead will depend on the time period to which the *n* instant belongs, according to the aforementioned clustering method.

In summary, the FCM method permits the modeller to obtain more realistic origin-destination patterns from historical data, and consequently, allows him (her) to systemize and improve the probability calculations. This procedure could improve the prediction power of future uncertainty resulting from the unknown future calls asking for service once they appear, in models with control horizons longer than one-step.

For example, the FCM model performs quite well for jumbled up trip patterns, in which representative zones could be spatially overlapped. Next, a one-dimension example is shown to illustrate the application of the method in the context of the DPDP.

A simple example for a single-vehicle dynamic routing problem is presented in Figure 4.4 in order to clarify the application of the FCM for forecast the probable calls as previously described. Let us assume door-to-door requests occurring on a one-dimensional path of nine kilometres, for pick-up and delivery positions. In the example, suppose that ten call requests occur over certain time-period (Figure 4.4), and suppose that all stops are considered to determine the optimal zoning and the corresponding probabilities associated with such a partition.

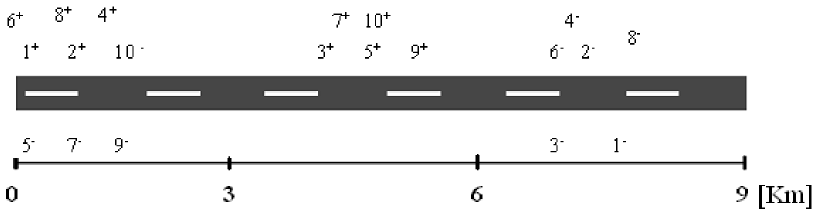


Figure 4.4. Single vehicle requests in a certain period of time.

Figure 4.5 shows a two-dimension representation of pick-up and delivery coordinates, for those requests shown in Figure 4.4. By looking at Figure 4.5, trip patterns could be identified just by looking at the points and identify those that are close by, since the problem is defined on a one-dimensional path. However, when the problem is defined on a two-dimensional path, the analysis needs an automatic methodology as fuzzy clustering proposed. From the historical data shown in Figure 4.5, the fuzzy C-means is used in order to obtain the optimal zoning associated with such a database. To do this, a fixed number of fuzzy clusters are selected and thus, Figure 4.6 shows the results of FCM for 2 and 3 fuzzy clusters, respectively. As explained before, the cluster centres are obtained and denoted by "x" marks in the figure.

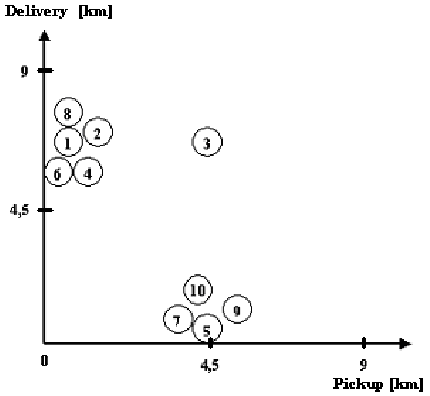


Figure 4.5. Pick-up-Delivery coordinates of historical demand over a certain time period.

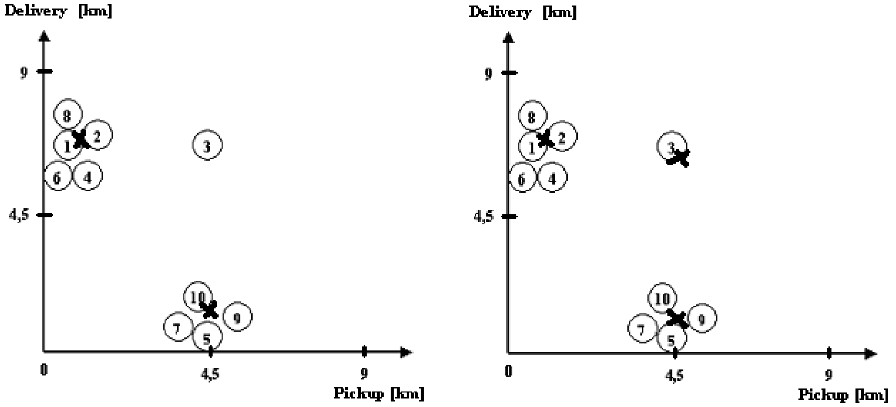


Figure 4.6. Cluster centers for 2 and 3 clusters selected.

Then, the mass centers are obtained after applying the FCM method corresponding to the resulting trip patterns, for this particular example. From an analysis of Figure 4.6, it seems reasonable to use 2 clusters instead of 3, since most requests are grouped around two mass centers. In general, stating the number of clusters is not as easy as in this example, and in such cases, the modeler should use methodologies that are more systematic as for example, the fuzzy cluster merging method (Babuska, 1999).

Figure 4.7 shows the membership degree as function of the ten call requests for 2 fuzzy clusters. As shown in Figure 4.7, the threshold selection determines that call 3 does not belong to any of the two fuzzy clusters, and therefore that datum has to be removed from the historical data.

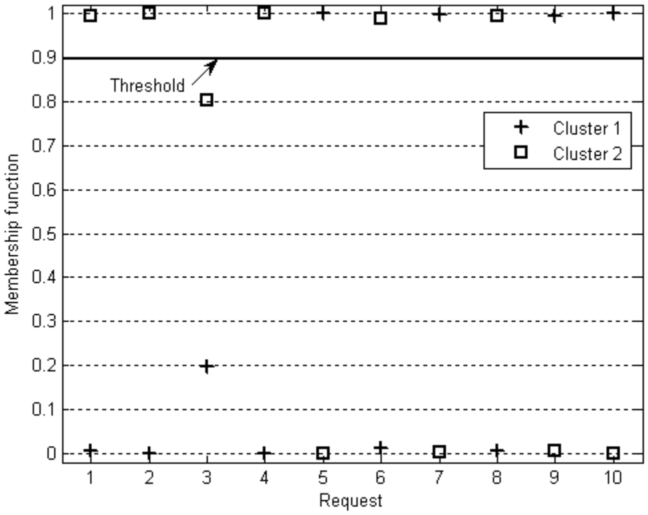


Figure 4.7. Membership degree of historical demand over a certain time period for 2 clusters.

Finally, and using the FCM procedure, the probabilities associated with trip patterns are shown in Table 4.1 for 2 fuzzy clusters.

Table 4.1 Probabilities for the trip patterns using 2 fuzzy clusters.

Trip pattern	Pick-up position	Delivery position	Probability
Fuzzy cluster 1	0.7194	6.9800	4/9
Fuzzy cluster 2	4.4748	0.2750	5/9

The proposed FCM methodology is applied to a more complex simulated example of a DPDP in Section 4.5, and is compared with a classical zoning approach. Once the optimization problem is stated (objective function and model), an efficient optimization algorithm is required to solve it. In the next Section 4.4, Genetic Algorithms for HPC are proposed to solve efficiently the optimization problem, in terms of both quality of solutions and computation time. Next, HPC design based on the proposed modeling described in section 4.2 and objective function with fuzzy prediction of demand proposed in section 4.3 is developed.

4.4. Genetic Algorithm for solving HPC in the context of the dial-a-ride system.

As explained before in chapter 3, the most used strategies of HPC involve two optimization algorithms: Explicit enumeration (EE) and Branch and Bound (BB). Both allow to solving mixed integer optimization problems (Floudas, 1995), but the elevated computational effort, especially in the case of EE, results in inefficient solutions for real-time problems.

On the contrary, GA has proved to be an efficient tool to solve MIOP (Man *et al.*, 1998). Thus, as VRP problems are NP-hard, HPC based on GA optimization is considered to face the DPDP problem. The framework used is based in the explained before in chapter 3.

Next, the manipulated variable is shown in detail, so the optimization problem and the simplifications assumed will be better understood. The original manipulated variable $S(k)$ is replaced by a matrix of binary activation values $G=(g_{ir})_{\substack{i=1..n\\ r=1..n}}$ that is associated with $P_j^i(k)$, which is a component of $S(k)$. Thus, $n = w_j(k)$ and the matrix element $g_{ir} \in \{0,1\}$ represents the r^{th} activation of stop *i*.

Then, a stop $P_j(k)$ associated with passenger $r_j^i(k)$ assigned to vehicle *j*, can be written as a linear combination of all the known stops ($f_1, f_2, \ldots, f_n$) assigned to the vehicle *j* using the binary factors of activation g_{ir}. Analytically,

$$P_j^i(k) = g_{i1}f_1 + g_{i2}f_2 + \ldots + g_{ir}f_r + \ldots. + g_{in}f_n \tag{4.10}$$

where

$$g_{ir} = \begin{cases} 0 & f_r \text{ is not stop i} \\ 1 & f_r \text{ is stop i} \end{cases} \tag{4.11}$$

Therefore, the stop position vector $P_j\left(k\right)$, excluding the initial condition $P_j^0\left(k\right)$, can be written as follows

$$P_j\left(k\right)=\begin{bmatrix} P_j^1(k) \\ P_j^2(k) \\ \vdots \\ \vdots \\ P_j^{n-1}(k) \\ P_j^n(k) \end{bmatrix}=\begin{bmatrix} g_{11} & g_{12} & \cdots\cdots & g_{1(n-1)} & g_{1n} \\ g_{21} & g_{22} & \cdots\cdots & g_{2(n-1)} & g_{2n} \\ \vdots & \vdots & \vdots\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots\ \vdots & \vdots & \vdots \\ g_{(n-1)1} & g_{(n-1)2} & \cdots\cdots & g_{(n-1)(n-1)} & g_{(n-1)n} \\ g_{n1} & g_{n2} & \cdots\cdots & g_{n(n-1)} & g_{nn} \end{bmatrix}\cdot\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_{n-1} \\ f_n \end{bmatrix}=G\cdot f \tag{4.12}$$

From this modeling framework, the constraint 2 above (a stop must be visited just once) can be written in terms of logical constraints. Thus, the following new constraints in terms of the g_{ir} values are generated:

$$g_{i1}+g_{i2}+...+g_{in}=1,\ \forall i=1,...,n \tag{4.13}$$

$$g_{1r}+g_{2r}+...+g_{nr}=1,\ \forall r=1,...,n \tag{4.14}$$

Among the set of stops, using the sequence, they will be pick-up or delivery. By respecting the precedence stops as well as all other logical constraints defined above in this section, state analytical relations are stated between elements of the G matrix in order to satisfy such constraints (a pick up has to happen before the associated delivery, etc.). When matrix G is used as the optimization variable instead of the sequence, the expected load can be expressed as the sum of the initial load plus all the activations of the previous pick-ups less the activations of all previous deliveries, as shown in (4.15) next:

$$\hat{L}_j(k+1)=\left[L_j^0\left(k\right)\ \cdots\ L_j^0\left(k\right)+\sum_{m=1}^{i}\left(\sum_{r\in P}\Omega\left(f_r\right)g_{mr}-\sum_{r\in D}\Omega\left(f_r\right)g_{mr}\right)\ \cdots\ \cdots\ 0\right]^T \tag{4.15}$$

where $\Omega\left(f_r\right)$ equals the number of passenger at stop f_r (this value depends on the request) and $P=\left\{r: f_r \text{ is a pick-up}\right\}$, $D=\left\{r: f_r \text{ is a delivery}\right\}$. By using (4.15), the capacity load constraint (constraint 4) can be written based on the activation factors of the matrix G. Analytically:

$$L_j^0(k)+\sum_{m=1}^{i}\left(\sum_{r\in P}\Omega(f_r)g_{mr}-\sum_{r\in D}\Omega(f_r)g_{mr}\right)\leq L_{\max} \qquad i=2,\ldots,n\text{-}1 \tag{4.16}$$

In addition, and to complete the state space model, the departure time vector can be expressed as function of the matrix *G*. In short,

$$\hat{T}(k+1)=\left[T^0(k) \quad T^0(k)+G^1Q(k)G^{2^T} \quad \cdots \quad T^0(k)+\sum_{r=1}^{i-1}G^rQ(k)G^{r+1^T} \quad \cdots \quad T^0(k)+\sum_{r=1}^{n-1}G^rQ(k)G^{r+1^T}\right]^T \tag{4.17}$$

with G^r denotes the r^{th} row of *G*, *Q(k)* is a matrix containing the network and transfer times computed between stops (from estimations based on Euclidean distance and traffic conditions).
In this model, an expansion and reduction matrix size technique is developed to capture the dynamic effect caused by the real operation. The idea is to either increase or reduce the stop position vector shown, resulting in changes on the load and time vectors as well. For example, when certain vehicle accepts a new service request, the dimension of the position vector increases in two rows, accounting for the customer pick-up and delivery stops. Additionally, when a vehicle reaches any stop, that point has to be removed from the original position vector, reducing its dimension in two rows.

4.4.1. Reduction of feasible search space: No swapping case.

In this application, the optimization is performed over a reduced space of solutions that satisfy the *no-swapping* constraint. This constraint ensures that sequences are constructed by locating the pick-up and delivery of the last call within the previous sequence (the order of previous stops does not change).

There are practical reasons for considering the no-swapping case in the model instead of exploring over a larger feasible search space. First, any other re-optimization strategy is very time-consuming for our algorithm, and not needed in most cases as discussed next. In fact, in all dynamic systems, it is necessary to use the previous information in order to make real-time decisions. Therefore, the configuration of the previous sequences (those scheduled before the insertion) must be considered as a relevant input to the optimization process. Additionally, in most pick-up and delivery problem configurations, the optimal solution of inserting a new request does not alter the order of previous sequences, as shown from simulation experiments by Cortés (2003). He found that the no-swapping strategy was optimal in more than 70% of the cases, and in the remainder not-optimal cases, the gap to optimality was negligible.

The global optimum of the dynamic routing problem in terms of the new optimization matrix *G* can be obtained by optimally choosing the activation factors g_{ir}, for each vehicle in the fleet. Indeed, *G* determines an optimal sequence of stops $P_j(k)$ for each vehicle *j* that minimizes the objective function defined in the next section, whenever a new real-time request has to be inserted into some previous sequence. Explicitly, the optimal $P_j(k)$ vector is given by:

$$P_j(k)=\begin{bmatrix} P_j^1(k) \\ P_j^2(k) \\ \vdots \\ \vdots \\ P_j^{n-1}(k) \\ P_j^n(k) \end{bmatrix}=\begin{bmatrix} g_{11} & g_{12} & \cdots & \cdots & g_{1(n-1)} & g_{1n} \\ g_{21} & g_{22} & \cdots & \cdots & g_{2(n-1)} & g_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_{(n-1)1} & g_{(n-1)2} & \cdots & \cdots & g_{(n-1)(n-1)} & g_{(n-1)n} \\ g_{n1} & g_{n2} & \cdots & \cdots & g_{n(n-1)} & g_{nn} \end{bmatrix}\cdot\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_{n-1} \\ f_n \end{bmatrix}=G\cdot f \tag{4.18}$$

where *f* is a vector containing the list of scheduled stops in the whole system at time *k*. In the *no-swapping* case, new calls are inserted directly in previous assigned sequences; by keeping the order of previously scheduled stops (only insertions on previous segments are allowed). As previous sequences hold, $(f_1, f_2, ..., f_{n\text{-}2})$, the new insertion added to the *f* vector at the bottom (pick-up, delivery), and denoted by $(f_{n\text{-}1}, f_n)$, imposes the following conditions on relation (4.18) above. Analytically,

$$P_i(k)=\begin{cases} g_{11}f_1+g_{1,n-1}f_{n-1}=(x_1,y_1) & \text{if} \quad i=1 \\ g_{21}f_1+g_{22}f_2+g_{2,n-1}f_{n-1}+g_{2,n}f_n=(x_2,y_2) & \text{if} \quad i=2 \\ g_{i,i-2}f_{i-2}+g_{i,i-1}f_{i-1}+g_{i,i}f_i+g_{i,n-1}f_{n-1}+g_{i,m}f_n=(x_i,y_i) & \text{if} \quad i=3,...,(n-2) \\ g_{n-1,n-3}f_{n-3}+g_{n-1,n-2}f_{n-2}+g_{n-1,n-1}f_{n-1}+g_{n-1,n}f_n=(x_{n-1},y_{n-1}) & \text{if} \quad i=n-1 \\ g_{n,n-2}f_{n-2}+g_{n,n}f_n=(x_n,y_n) & \text{if} \quad i=n \end{cases} \tag{4.19}$$

where (x_i, y_i) are the spatial coordinates of the *i*-stop. For example, the first term of (4.19) $(i = 1)$ represents the first component of the stop sequence that must be either the new pick up or the first stop of the previous sequence. The second term $(i = 2)$ represents the second component of the stop sequence that has more options, either the first stop of the previous sequence, the second stop of the previous sequence, the new pick-up stop request or the new delivery stop, and so on.

Equation (4.19) can also be written in the form of general expression (4.18), obtaining the following sparse *G* matrix (optimization decision matrix):

$$G=\begin{bmatrix}
g_{11} & 0 & 0 & 0 & 0 & 0 & \dots & \dots & \dots & \dots & 0 & 0 & 0 & g_{1(n-1)} & 0 \\
g_{21} & g_{22} & 0 & 0 & 0 & 0 & \dots & \dots & \dots & \dots & 0 & 0 & 0 & g_{2(n-1)} & g_{2n} \\
g_{31} & g_{32} & g_{33} & 0 & 0 & 0 & \dots & \dots & \dots & \dots & 0 & 0 & 0 & g_{3(n-1)} & g_{3n} \\
0 & g_{42} & g_{43} & g_{44} & 0 & 0 & \dots & \dots & \dots & \dots & 0 & 0 & 0 & g_{4(n-1)} & g_{4n} \\
0 & 0 & g_{53} & g_{54} & g_{55} & 0 & \dots & \dots & \dots & \dots & 0 & 0 & 0 & g_{5(n-1)} & g_{5n} \\
0 & 0 & 0 & g_{64} & g_{65} & g_{66} & \dots & \dots & \dots & \dots & 0 & 0 & 0 & g_{6(n-1)} & g_{6n} \\
: & : & : & 0 & . & . & . & . & . & . & : & : & : & : & : \\
: & : & : & : & . & . & . & . & . & . & : & : & : & : & : \\
: & : & : & : & . & . & . & . & g_{(n-4)(n-6)} & g_{(n-4)(n-5)} & g_{(n-4)(n-4)} & 0 & 0 & g_{(n-4)(n-1)} & g_{(n-4)n} \\
: & : & : & : & . & . & . & . & 0 & g_{(n-3)(n-5)} & g_{(n-3)(n-4)} & g_{(n-3)(n-3)} & 0 & g_{(n-3)(n-1)} & g_{(n-3)n} \\
0 & 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & g_{(n-2)(n-4)} & g_{(n-2)(n-3)} & g_{(n-2)(n-2)} & g_{(n-2)(n-1)} & g_{(n-2)n} \\
0 & 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & g_{(n-1)(n-3)} & g_{(n-1)(n-2)} & g_{(n-1)(n-1)} & g_{(n-1)n} \\
0 & 0 & 0 & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & 0 & g_{n(n-2)} & 0 & g_{nn}
\end{bmatrix}$$

This analytical problem formulation allows us to generalize the *N*-step-ahead optimization criteria defined in the next section and to evaluate different nonlinear mixed integer optimization methods, as the GA method it will describe next. If the *no-swapping* operational constraint is relaxed, the search space for optimization increases, resulting in a less sparse matrix *G*, allowing the optimization procedure to obtain a solution closer to a less restrictive global optimum. An intermediate case (*partial swapping*) is currently being studied as discussed in the further research section.

4.4.2. HPC based on GA for a dial-a-ride system.

The GA method is suitable for the dial-a-ride system since optimization variables are discrete, and therefore the binary codification is not necessary. In other words, genes of the individuals (feasible solutions) are given directly by the integer optimization variables. In addition, gradient computations are not necessary as in conventional non-linear optimization solvers, which allow us to significantly save computation time.

HPC based on GA, described in chapter 3, is used as an efficient optimization solver for the DPDP problem, where the optimization variables identify the stops that must be satisfied by the vehicle fleet. The individuals are the feasible sequences, fulfilling the load, precedence and no swapping constraints defined before. The gene of an individual considers the following three components: the vehicle j used for the new insertion and the sequence position of the new call (for both pick-up and delivery) within the previous sequence, assuming the *no-swapping* policy.

To explain the gene codification, a simple example for one individual is presented. Let us assume the following vector $P_j(k-1)$, associated with the sequence at the previous instant *k-1* ($S_j(k-1)$).

$$P_j(k-1)=\begin{bmatrix} P_j^1 \\ P_j^2 \\ P_j^3 \\ P_j^4 \end{bmatrix}=\underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{G}\cdot\underbrace{\begin{bmatrix} b(1^+) \\ b(2^+) \\ b(1^-) \\ b(2^-) \end{bmatrix}}_{f} \tag{4.20}$$

where $b(x)$ denotes the position of stop *x*. For this example, a new customer labeled as 3 enters the system, and has to be inserted. The new optimization variable can be represented in terms of $P_j(k)$ as shown in the following matrix equation system, by adding the request in the last two rows of vector f, increasing the dimension of matrix *G*.

$$P_j\left(k\right)=\begin{bmatrix}P_j^1\\P_j^2\\P_j^3\\P_j^4\\P_j^5\\P_j^6\end{bmatrix}=\underbrace{\begin{bmatrix}g_{11}&0&0&0&g_{15}&0\\g_{21}&g_{22}&0&0&g_{25}&g_{26}\\g_{31}&g_{23}&g_{33}&0&g_{35}&g_{36}\\0&g_{24}&g_{34}&g_{36}&g_{45}&g_{46}\\0&0&g_{35}&g_{37}&g_{55}&g_{56}\\0&0&0&g_{38}&0&g_{66}\end{bmatrix}}_{G}\cdot\underbrace{\begin{bmatrix}b(1^+)\\b(2^+)\\b(1^-)\\b(2^-)\\b(3^+)\\b(3^-)\end{bmatrix}}_{f} \tag{4.21}$$

Due to the precedence and *no swapping* constraints, the previous sequence is held, and the decision variables are given by the last two columns of matrix *G*. By using the proposed gene codification, a feasible population of 7 individuals for vehicle *j* is presented by considering the previous sequence and the new call request:

$$\text{Population}\Leftrightarrow\begin{pmatrix}\text{Individual 1}\\\text{Individual 2}\\\text{Individual 3}\\\text{Individual 4}\\\text{Individual 5}\\\text{Individual 6}\\\text{Individual 7}\end{pmatrix}\Leftrightarrow\begin{pmatrix}(j,1,4)\\(j,1,6)\\(j,5,6)\\(j,3,5)\\(j,4,6)\\(j,1,6)\\(j,2,4)\end{pmatrix}\Leftrightarrow\begin{pmatrix}j,\boxed{3^+}\to1^+\to2^+\to\boxed{3^-}\to1^-\to2^-\\j,\boxed{3^+}\to1^+\to2^+\to1^-\to2^-\to\boxed{3^-}\\j,1^+\to2^+\to1^-\to2^-\to\boxed{3^+}\to\boxed{3^-}\\j,1^+\to2^+\to\boxed{3^+}\to1^-\to\boxed{3^-}\to2^-\\j,1^+\to2^+\to1^-\to\boxed{3^+}\to2^-\to\boxed{3^-}\\j,\boxed{3^+}\to1^+\to2^+\to1^-\to2^-\to\boxed{3^-}\\j,1^+\to\boxed{3^+}\to2^+\to\boxed{3^-}\to1^-\to2^-\end{pmatrix} \tag{4.22}$$

For example, the individual $(j,1,4)$ in terms of $P_j(k)$ can be written as:

$$\text{Individual 1}\Leftrightarrow P_j\left(k\right)=\begin{bmatrix}P_j^1\\P_j^2\\P_j^3\\P_j^4\\P_j^5\\P_j^6\end{bmatrix}=\underbrace{\begin{bmatrix}0&0&0&0&1&0\\1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&0&0&0&1\\0&0&1&0&0&0\\0&0&0&1&0&0\end{bmatrix}}_{G}\cdot\underbrace{\begin{bmatrix}b(1^+)\\b(2^+)\\b(1^-)\\b(2^-)\\b(3^+)\\b(3^-)\end{bmatrix}}_{f} \tag{4.23}$$

In short, the last two columns of matrix *G* are the new optimization variables associated with the sequence at instant *k*. As the individuals of a generation are randomly selected, the same individuals can be repeated in the next population. For example in (4.22), individuals 2 and 6 are the same in the population, $(j,1,6)$.

Note that as GA considers random generation of individuals, the genetic operators (mutation or crossover) could provide infeasible solutions that have to be removed (typically through the capacity constraint). In order to have at least one feasible solution of the population, an always feasible individual, such as $\left(j, w_j - 1, w_j\right)$ must be used *(* w_j is the number of stops including the last call). The number of individuals for each population has to be smaller than the total number of feasible combinations in order to avoid solving the explicit enumeration method. The crossover operator is not applied here since the *no swapping* constraint has to be satisfied.

For a two-step-ahead problem, a possible population is:

$$\begin{Bmatrix}\text{individual 1}\\ \text{individual 2}\\ \text{individual 3}\\ \text{individual 4}\end{Bmatrix} \Leftrightarrow \begin{Bmatrix} \left(\left[1,1,4\right],\begin{bmatrix}\left[1,2,4\right]\\ \left[1,3,4\right]\end{bmatrix}\right)\\ \left(\left[1,2,3\right],\begin{bmatrix}\left[2,1,2\right]\\ \left[1,1,3\right]\end{bmatrix}\right)\\ \left(\left[2,2,4\right],\begin{bmatrix}\left[1,3,4\right]\\ \left[2,3,6\right]\end{bmatrix}\right)\\ \left(\left[2,3,5\right],\begin{bmatrix}\left[2,2,3\right]\\ \left[2,1,8\right]\end{bmatrix}\right)\end{Bmatrix} \Leftrightarrow \begin{Bmatrix} \left(\left[1,\boxed{4^+}\to 2^+\to 2^-\to\boxed{4^-}\right],\begin{bmatrix}\left[1,4^+\to\boxed{h_1^+}\to 2^+\to\boxed{h_1^-}\to 2^-\to 4^-\right]\\ \left[1,4^+\to 2^+\to\boxed{h_2^+}\to\boxed{h_2^-}\to 2^-\to 4^-\right]\end{bmatrix}\right)\\ \left(\left[1,2^+\to\boxed{4^+}\to\boxed{4^-}\to 2^-\right],\begin{bmatrix}\left[2,\boxed{h_1^+}\to\boxed{h_1^-}\to 3^+\to 3^-\to 1^+\to 1^-\right]\\ \left[1,\boxed{h_2^+}\to 2^+\to\boxed{h_2^-}\to 4^+\to 4^-\to 2^-\right]\end{bmatrix}\right)\\ \left(\left[2,3^+\to\boxed{4^+}\to 3^-\to\boxed{4^-}\to 1^+\to 1^-\right],\begin{bmatrix}\left[1,2^+\to 2^-\to\boxed{h_1^+}\to\boxed{h_1^-}\right]\\ \left[2,3^+\to 4^+\to\boxed{h_2^+}\to 3^-\to 4^-\to\boxed{h_2^-}\to 1^+\to 1^-\right]\end{bmatrix}\right)\\ \left(\left[2,3^+\to 3^-\to\boxed{4^+}\to 1^+\to\boxed{4^-}\to 1^-\right],\begin{bmatrix}\left[2,3^+\to\boxed{h_1^+}\to\boxed{h_1^-}\to 3^-\to 4^+\to 1^+\to 4^-\to 1^-\right]\\ \left[1,\boxed{h_2^+}\to 3^+\to 3^-\to 4^+\to 1^+\to 4^-\to 1^-\to\boxed{h_2^-}\right]\end{bmatrix}\right)\end{Bmatrix}$$

In this example of codification, it has two patterns request which are h_1 and h_2. A solution of the optimization problem in this case consider a two-step-ahead policy, and the solution set includes three sequences (the first one for the current call, the other two in the case that once the previous request happen, and was located in the sequence of a given vehicle, then the two possible request happen).

Figure 4.8 presents the proposed hybrid predictive control system scheme. The real system of fleet-clients assigns the sequences using the HPC controller based on the state space variables, on a call prediction model and on the new call request information.

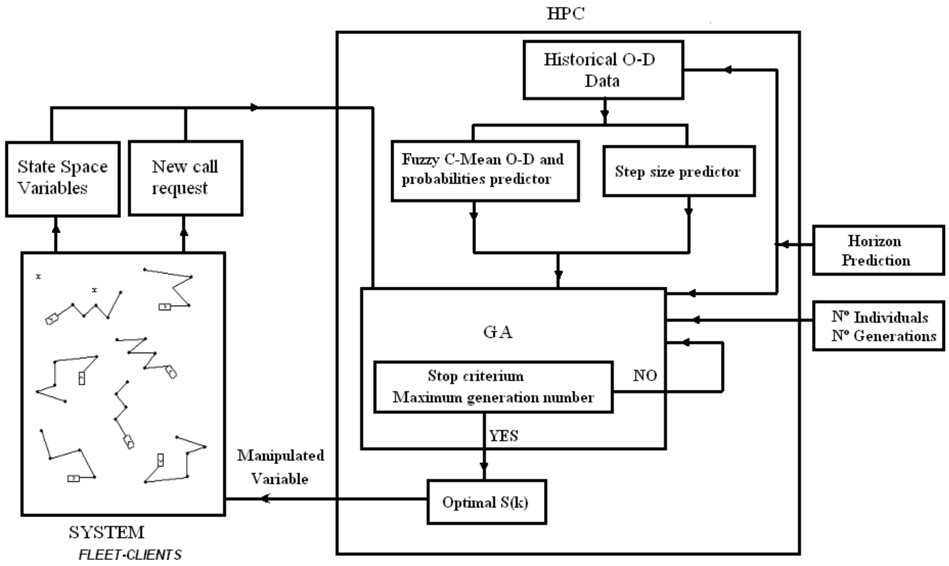


Figure 4.8. Overall block diagram of an HPC for dial-a-ride system.

Next, an application of HPC in the context of dial-a-ride system is summarized, to visualize the advantages of that method when compared with explicit enumeration, mainly in computation time saving.

Illustrative tests using explicit enumeration (EE) and GA methods are conducted to evaluate the performance through the proposed objective function and the corresponding computation times.

The dial-a-ride system with 4 vehicles and an objective function of two-step-ahead with 6 potential calls are considered. Vehicles cover an urban service area of around 81 km^2, traveling at an average speed of 20 kilometers per hour.

The simulations tests considered are:

i) Dynamic vehicle routing under high demand conditions,
ii) Dynamic vehicle routing under normal demand conditions and
iii) Dynamic vehicle routing considering a mixed solution (combining GA and EE methods).

As mentioned before, the GA method considers the number of individuals and generations, and mutation probability as tuning parameters. Results for three different cases of tuning parameters are presented. The first genetic solution G1 considers 5 individuals and 5 generations, G2 uses

10 individuals and 10 generations, and finally G3 considers 20 individuals and 20 generations. The simulation tests were conducted in Matlab version 6.5.1 release 13, on a Pentium IV processor.

4.4.3.1 Test 1: Dynamic vehicle routing under high demand conditions.

In this case, many call requests enter the system over a short time period, generating long sequences and consequently, longer computation times due to a larger search space. Figure 4.9 shows the computation times and the objective function for a certain period over which a lot of calls enter the system (note that the step size in the model is variable, and depends on when the new call is received by the dispatcher).

From Figure 4.9, the request congestion is observed, and therefore GA presents a cumulative cost (see objective function) at each new call because the decision taken at the previous instant (previous sequence) does not always correspond to the global optimum. In addition, the computation time increases exponentially by using EE while the number of stops increases, unlike GA showing stable computation times regardless of the call intensity. In Table 4.2, the mean value of the objective function and computation time are reported by using the data presented in Figure 4.9. According to Figure 4.9 and Table 4.2, when the number of individuals and the number of generations increase, a better tracking of the global optimum objective function is observed (G3, in special) with a significantly short computation time.

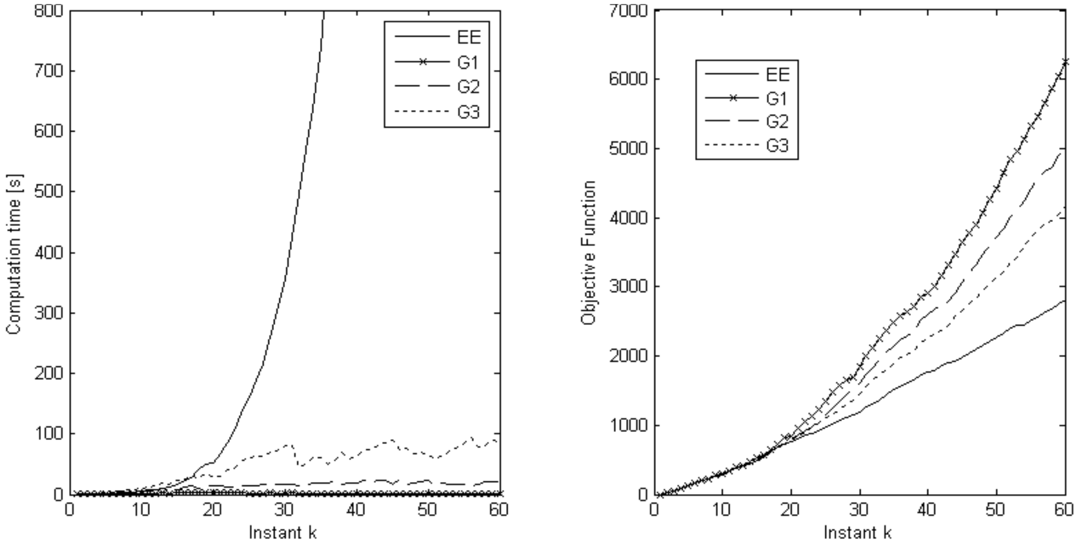


Figure 4.9. Evolution of performance indexes.

Table 4.2 Performance indexes.

Control Strategy Test 1	Objective function mean	Computation time mean
Explicit Enumeration EE	1297.4	1536.7
Genetic Algorithms G1	2288.2	1.4
Genetic Algorithms G2	1945.8	13.9
Genetic Algorithms G3	1694.6	49.7

4.4.3.2 Test 2: Dynamic vehicle routing under normal demand conditions.

In this case, few call requests enter the system over the studied time period. The selection of sub-optimal solutions is not very relevant due to the existence of short sequences since most stops are reached while the system is working.

Figure 4.10 and Table 4.3 show computation times and objective function values. By looking at the objective function evolution in Figure 4.10, the GA behavior looks similar to the optimal one (EE), while a non-significant computation time effort is observed using GA. Table 4.3 shows that as the number of individuals and generations increase, the solution converges to the optimal global solution (EE). Notice that the G3 solution is the same as that provided by EE, because G3 computes almost all possible solutions, consuming a longer computation time though.

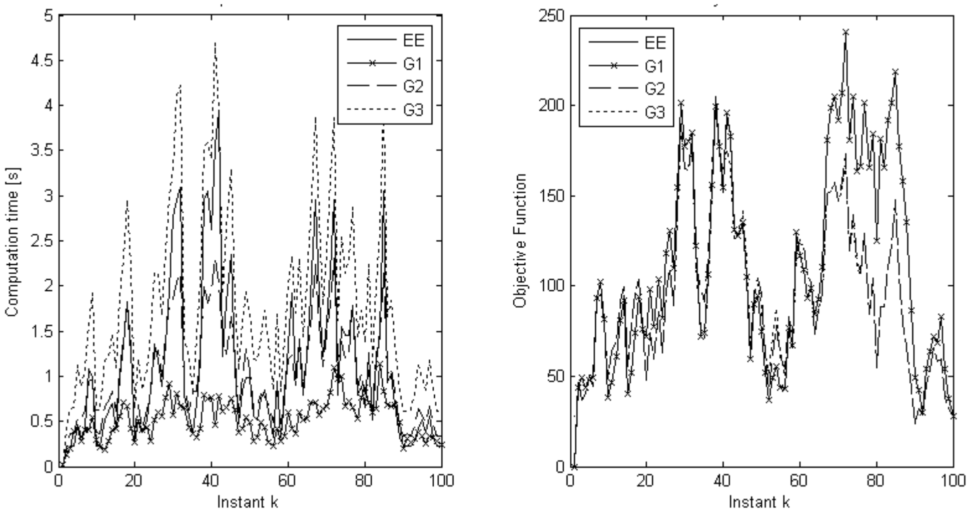


Figure 4.10. Evolution of performance indexes.

Table 4.3 Performance indexes.

Control Strategy Test 2	Objective function mean	Computation time mean
Explicit Enumeration EE	94.5	1.1
Genetic Algorithms G1	110.9	0.5
Genetic Algorithms G2	95.4	1.1
Genetic Algorithms G3	94.5	1.8

4.4.3.3 Test 3: Dynamic vehicle routing considering a mixed solution (combining GA and EE methods).

This case is similar to Test 1, but here the previous sequences for the GA method are calculated by EE, that is to say, at any instant optimization, a good initial solution is used. Figure 4.11 and Table 4.4 show the objective function evolution and its corresponding error with respect to the optimal solution obtained by the EE method. Although the sequence is longer, the GA objective function error is not significantly increased.

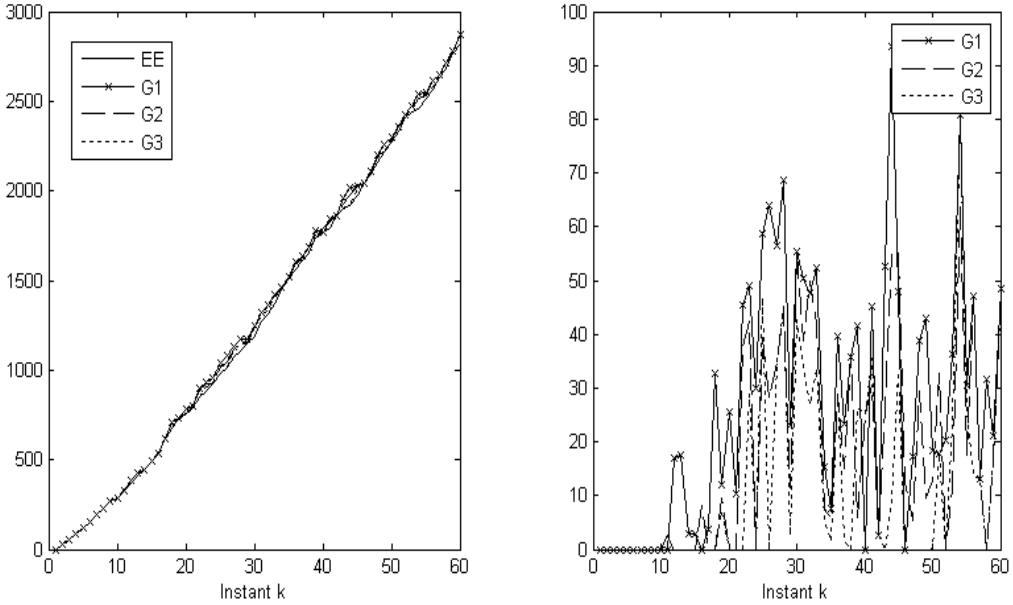


Figure 4.11. Evolution of performance indexes.

Table 4.4 Performance indexes.

Control Strategy Test 3	Objective function mean	Computation time mean
Explicit Enumeration EE	1297.4	------
Genetic Algorithms G1	1324.0	26.6
Genetic Algorithms G2	1315.1	17.7
Genetic Algorithms G3	1309.3	11.9

According to Figure 4.11 and Table 4.4, dispatch decisions obtained by GA are very similar to EE, regardless of the number of planned stops.

In the next section, two more detailed applications are presented. The first one including FCM and GA for one, two and three-steps-ahead problems. The second one compares the effect of traffic conditions when the model considers variations under predictable traffic conditions.

4.5. Simulation results for HPC applied to a dial-a-ride system.

4.5.1. HPC with demand prediction.

A discrete-event system simulation for a two-hour period is conducted in order to evaluate the performance of both fuzzy zoning and genetic algorithm method by using a *no-swapping* operational policy. A fleet of nine vehicles, with capacity for four passengers each, is considered. The simulation tests are implemented in Matlab version 6.5.1 release 13 running on a Pentium IV processor.

The future origin-destination trip patterns are assumed unknown. However, historical demand obtained from the average demand measured over a week before or so, is available. This scenario is not real although, the demand patterns follow a heterogeneous distribution inspired on real data.

An urban service area of approximately 81 km^2 is considered. Vehicles are assumed to travel straight between stops at an average speed of 20 km/hr within the region. All simulations are performed over two representative hours $(14:00-14:59, 15:00-15:59)$ of a working day.

The historical data generated via simulation follows the trips patterns shown in Figure 4.12 with arrows.

For the simulation test, 120 calls were generated over the whole simulation period of two hours according to a spatial and temporal distribution following the same behaviour as that of the historical data.

Regarding the temporal dimension, a negative exponential distribution is assumed for time intervals between calls with rate of 1 [call/minute] for both the first and second hour of simulation. In terms of spatial distribution, pick-up and delivery points were generated randomly within each corresponding zone. A reasonable warm up period was considered to avoid boundary distortions (10 calls at the beginning and 10 at the end).

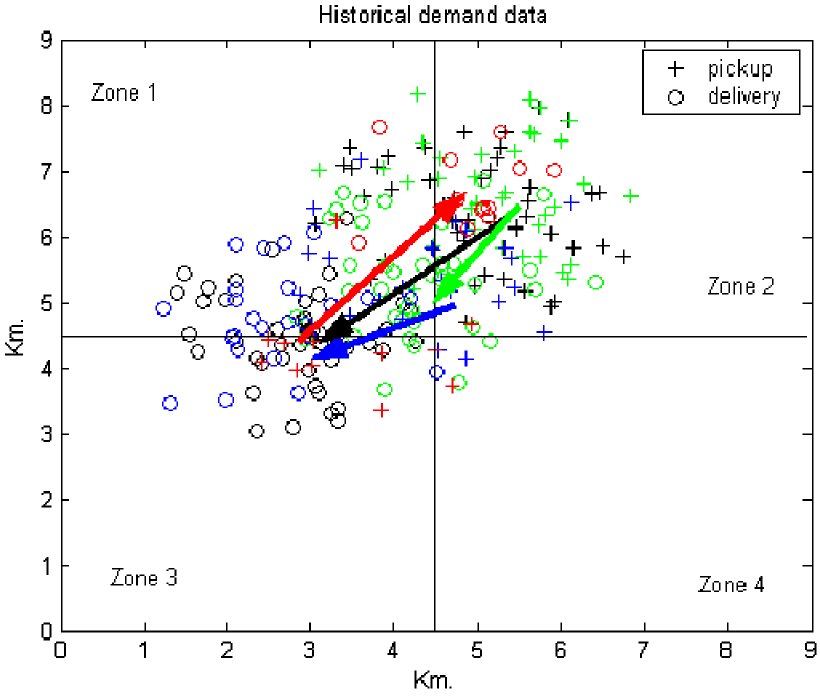


Figure 4.12. Origin-destination trip patterns.

Fifty replications of each experiment were conducted to obtain global statistics. With regard to the cost function, a weight $\alpha = 1$ was used, which means that travel time is as important as waiting time in the cost function expression. In order to compare the performance of the fuzzy zoning proposed with respect to a classic zoning (the four squared areas shown in Figure 4.12), two steps algorithms were tested and explicit enumeration results were considered for benchmarking.

Figure 4.13 shows an application of the procedure described in Section 4.3. In fact, 4 fuzzy clusters are obtained (Step 1), next their membership degrees are depicted (Step 2). Each call is associated with the largest membership degree (Step 3). In addition, the threshold is fixed and equal to 0.6 in order to consider just the data associated with the relevant trip patterns (Step 4). Next the corresponding probabilities are computed (Step 5) and the fuzzy cluster centres are obtained again using FCM (Step 6).

Table 4.5 shows the coordinates of fuzzy cluster centres for pick-up and delivery points of relevant trip patterns and the corresponding probabilities. On the other hand, Table 4.6 shows the classic zoning based upon 4 origin-destination pairs.

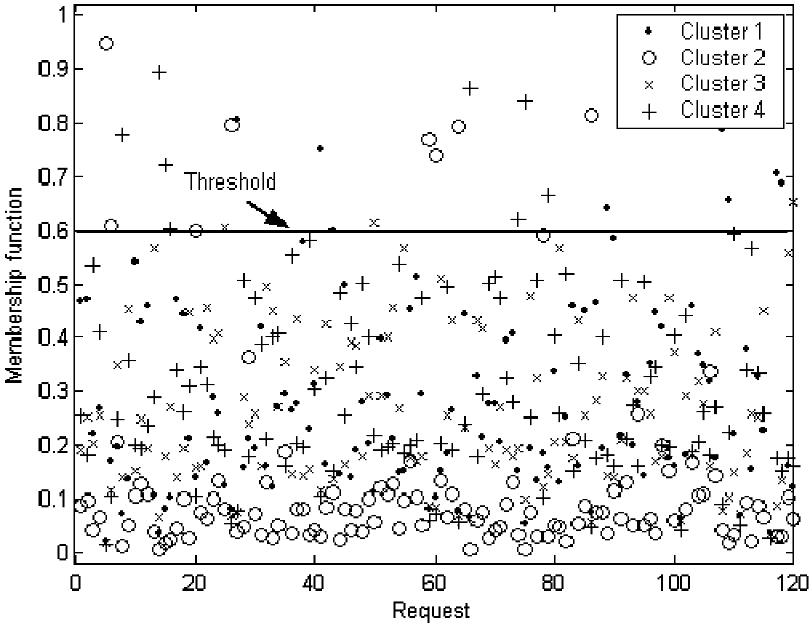


Figure 4.13. Membership degree for call requests.

Table 4.5 Pick-up and delivery coordinates and probabilities: Fuzzy zoning.

X pick-up	Y pick-up	X delivery	Y delivery	Probability
4.5540	5.7155	2.9218	4.7514	0.1282
3.7514	4.4812	5.2293	6.2232	0.2051
4.7989	6.6121	3.0751	4.4972	0.2564
5.2595	6.5057	4.3494	5.5161	0.4103

Table 4.6 Pick-up and delivery coordinates and probabilities: Classic zoning.

X pick-up	Y pick-up	X delivery	Y delivery	Probability
6.75	6.75	6.75	6.75	0.0968
2.25	6.75	2.25	6.75	0.2151
6.75	6.75	2.25	2.25	0.3118
6.75	6.75	2.25	6.75	0.3763

One fine-tuning parameter is the predicted in time between successive calls, τ , which is relevant when evaluating the performance function of more than one-step-ahead algorithms. The optimal value of such a parameter is found by conducting a sensitivity analysis around the observed inter-arrival times from the historical data report.

Figures 4.14 and 4.15 show the effective objective function (considering user as well as operation cost) using different τ values for both classic and fuzzy zonings. Ten replications for each considered τ value were used in order to obtain optimal values. For both zoning methods, the resulting optimal $\tau = 5$.

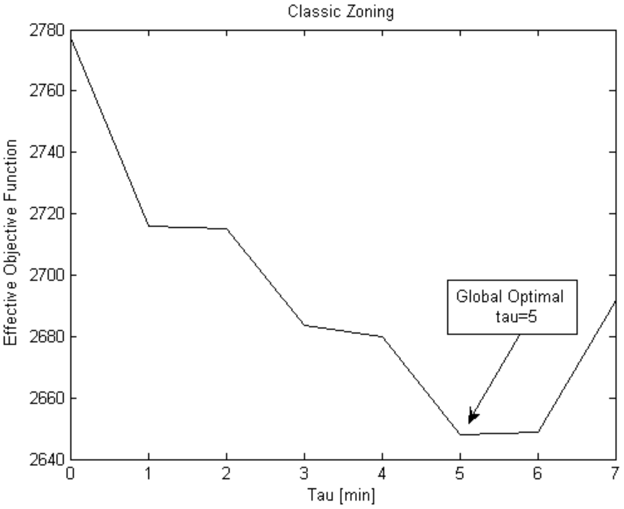


Figure 4.14. Sensitivity analysis for τ (classic zoning).

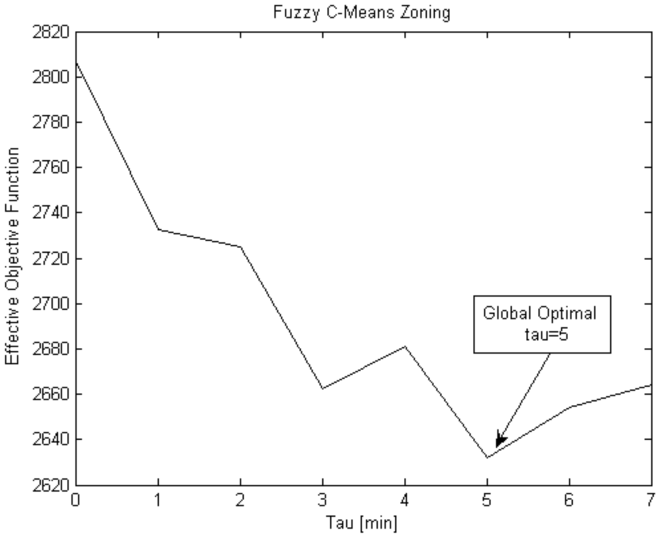


Figure 4.15. Sensitivity analysis for τ (fuzzy zonings).

Using the obtained optimal values of τ , 50 replications of the two-steps-ahead algorithm based on explicit enumeration were conducted in order to compare the performance of both zoning methods. Table 4.7 presents the mean and standard deviations of the waiting, travel and total time for users. The comparison of fuzzy zoning with respect to classic zoning is shown in the same table. It observed that waiting time is significantly reduced (3.36%) while travel time remains almost constant and consequently, total time also decreases (1.71%).

Table 4.7 User costs.

Two-step-ahead algorithm	Waiting time (min)		Travel time (min)		Total time (min)	
	Mean	Std	Mean	Std	Mean	Std
Classic zoning	6.1437	0.87	10.2358	0.71	16.3795	1.44
Fuzzy zoning	5.9370	0.72	10.1629	0.76	16.0999	1.36
Savings	0.2067		0.0729		0.2796	
Improv. (%)	3.36%		0.71%		1.71%	

Operational costs for the entire vehicle fleet are presented in Table 4.8. In addition, the total cost including user and operational cost (as in the objective function) is also shown in Table 4.8. A moderate improvement is observed for both components. However, the proposed fuzzy zoning methodology is a systematic alternative that allows determining trip patterns and their corresponding probabilities over a more realistic dynamic dial-a-ride system with jumbled up trip patterns.

Table 4.8 Vehicle and total costs.

Two-step-ahead algorithm	Operational costs (min)		Total effective cost (min)	
	Mean	Std	Mean	Std
Classic zoning	117.9	8.81	2699.4	122.84
Fuzzy zoning	115.7	8.12	2651.1	112.86
Savings	2.2618		48.3163	
Improv. (%)	1.92%		1.79%	

In order to analyze and evaluate the performance of both the proposed fuzzy zoning and the HPC based on GA, simulation tests were conducted for one, two and three-step-ahead problems under the same conditions. The results of 50 replications with GA solver are presented by using 20 individuals and 20 generations. It also assumes the same trip patterns and probabilities obtained for the two and three-step-ahead scenarios.

Table 4.9 shows the effective waiting, travel and total times of passengers, by using the fuzzy HPC based on GA for different prediction horizons.

It is observed that waiting time is significantly reduced by using the two-step-ahead method (15.04%) and even more from the three-step-ahead (22.30%), when compared with the myopic one-step-ahead method. In addition, a moderate improvement in travel time is observed.

An interesting case is the comparison between the two-step-ahead with the three-step-ahead predictive method in terms of travel time. In fact, savings in travel time are greater for the two-step-ahead method, mainly due to the greater uncertainty as the prediction horizon increases, affecting the reliability of the estimated probabilities. Due to this compensatory fact, the total time saving obtained with the three-step-ahead method is almost the same as that of the two-step-ahead (9.78% and 9.45% respectively).

Table 4.9 Performance comparison for one, two and three-step-ahead problems.

	Waiting time (min)		Travel time (min)		Total time (min)	
	Mean	Std	Mean	Std	Mean	Std
One-step-ahead	6.969	0.82	10.877	0.89	17.847	1.46
Two-step-ahead	5.921	0.67	10.238	0.79	16.159	1.42
Three-step-ahead	5.415	0.53	10.687	0.65	16.102	1.35
Savings 2 step	1.048		0.639		1.688	
Improv. (%)	15.04%		5.87%		9.45%	
Savings 3 step	1.554		0.190		1.745	
Improv. (%)	22.30%		1.75%		9.78%	

Table 4.10 describes the operational costs for the entire vehicle fleet. In addition, total effective cost is also reported in the table. It observes that vehicle operational costs increase with the two and three-step-ahead methods, however, total effective costs are still reduced by running both the two-step-ahead (5.9%) and the three-step-ahead (4.47%) methods. From the results, it can be said that the two-step-ahead method seems better than the three-step-ahead algorithm, because the longer the prediction horizon, the less reliable the estimated probabilities are.

Table 4.10 Vehicle and total costs comparison for one, two and three-step-ahead problems.

	Operational costs (min)		Total effective cost (min)	
	Mean	Std	Mean	Std
One-step-ahead	105.04	9.76	2730.0	127.832
Two-step-ahead	105.87	11.68	2568.7	114.516
Three-step-ahead	110.86	11.18	2608.0	112.444
Savings 2 step	-0.84		161.27	
Improv. (%)	-0.79%		5.90%	
Savings 3 step	-5.82		122.05	
Improv. (%)	-5.54%		4.47%	

4.5.2 HPC with demand and congestion prediction.

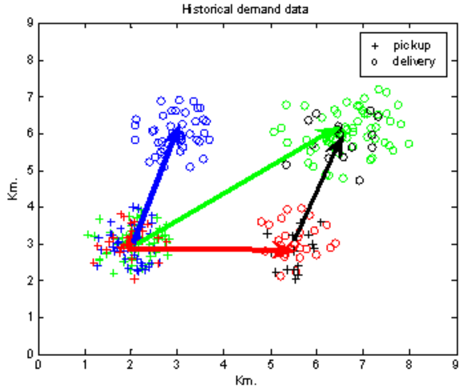
In this section, some simulation tests are carried out in order to quantify the potential benefits of HPC with demand and congestion prediction in the context of a dial-a-ride system. In the experiments a transportation fleet of nine vehicles, with capacity for four passengers each is used. The simulation tests are implemented in Matlab version 7.0.1 release 14 running on a Pentium® D CPU 3.20GHz processor.

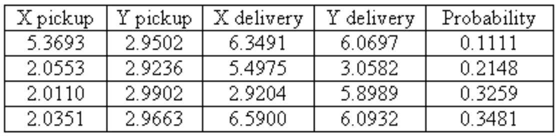
The future origin-destination trip patterns are unknown. However, historical demand obtained from the average demand measured over a week before or so, is available. This scenario is not real. However, the demand patterns follow a heterogeneous distribution inspired on real data from the Origin-Destination Survey in Santiago, Chile, 2001. An urban service area of approximately 81 km^2 is considered and all simulations are performed over two representative hours $(14:00-14:59, 15:00-15:59)$ of a working day. The vehicles are travelling straight between stops and the embedded network following the speed distribution stated in (4.24).

$$v(t,p,\varphi)=20+\left(5-\frac{\mathrm{t}}{12}\right)\cdot e^{-\frac{(p_x-4)^2+(p_y-4)^2}{2}}+\left(\frac{\mathrm{t}}{12}-5\right)\cdot e^{-\frac{(p_x-7)^2+(p_y-6)^2}{2}}+\varphi(t) \qquad (4.24)$$

where t[min] is the clock time, t=0[min] corresponds to 14:00, and t=120[min] to 16:00. $p=(p_x,p_y)$ [km] denotes a position in terms of the plane coordinates inside the urban area. $\varphi(t)$ is the white noise that captures the stochasticity coming from traffic congestion.

The speed distribution shows how the congestion moves from one side of the urban area to the other along the two hours simulation. The historical data generated via simulation follows the trips patterns shown in Figure 4.16 with arrows. From historical data and a fuzzy zoning method, Figure 4.16 also shows the pick up and delivery coordinates and the probabilities for the most relevant trip patterns.





X pickup	Y pickup	X delivery	Y delivery	Probability
5.3693	2.9502	6.3491	6.0697	0.1111
2.0553	2.9236	5.4975	3.0582	0.2148
2.0110	2.9902	2.9204	5.8989	0.3259
2.0351	2.9663	6.5900	6.0932	0.3481

Figure 4.16. Origin-destination trip patterns. Pick-up and delivery coordinates and probabilities: Fuzzy zoning

For the simulation test, 120 calls were generated following the same behavior as that of the historical data. Regarding the temporal dimension, a negative exponential distribution is assumed for time intervals between calls with rate of 0.9 [call/minute]. In terms of spatial distribution, pick-up and delivery points were generated randomly within each corresponding zone. A reasonable warm up period was considered to avoid boundary distortions (10 calls at the beginning and 10 at the end). 50 replications of each experiment were conducted to obtain global statistics. With regard to the objective function, a weight $\alpha=1$ was used, which means that travel time is as important as waiting time into the cost function expression.

In order to analyze and evaluate the performance of HPC strategies, simulation tests were conducted for one and two-step-ahead algorithms under the same conditions. Two-step-ahead algorithm was performed considering the 4 trip patters shown in Figure 4.16. The results of 50 replications with GA solver are presented by using 20 individuals and 20 generations.

Table 4.11 shows the effective waiting and travel times of passengers, by using the HPC based on GA for one and two-step-ahead prediction, and for the two velocity estimations. A constant estimation of velocity means that the expected departure time is computed based on the constant speed. The second estimation (variable velocity) is more realistic since it is adapted to the network velocity conditions through the recurrent model $\hat{v}(t,p)$. The waiting time is significantly reduced by using the two-step ahead method (12%) when compared against the myopic one-step-ahead method. In addition, an improvement in travel time is also observed.

Table 4.12 describes the operational costs for the entire vehicle fleet. In addition, total effective costs are also reported in the table. The vehicle operational costs and the total effective costs are still reduced by running both the constant velocity (8.81%) and the variable velocity (8.00%) methods.

From this example, an improvement of 3.26% in waiting time is found, and still one improvement of 1.68% in total time, only due to the fact of including a more sophisticated prediction of the velocity over the space and time, based on historical data (recurrent congestion).

Table 4.11. Performance comparison for one and two-step-ahead algorithms.

Strategy	Variable velocity estimation				Constant velocity estimation			
	waiting time (min)		travel time (min)		waiting time (min)		travel time (min)	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
One-step-ahead	15.443	1.64	17.879	0.61	15.844	1.25	18.346	0.78
Two-step-ahead	13.618	1.90	16.939	0.65	14.077	1.78	17.002	0.74
Savings 2 step	1.8243		0.9402		1.7671		1.3434	
Improv. (%)	11.81%		5.26%		11.15%		7.32%	

Table 4.12. Operational and total costs.

Strategy	Variable velocity estimation				Constant velocity estimation			
	Operational costs (min)		Effective total costs (min)		Operational costs (min)		Effective total costs (min)	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
One-step-ahead	143.68	7.3172	3809.1	183.23	145.13	7.84	3906.0	189.51
Two-step-ahead	142.95	8.7826	3504.3	256.51	143.21	7.83	3562.0	258.02
Savings 2 step	0.7316		304.82		1.9125		344.07	
Improv. (%)	0.51%		8.00%		1.32%		8.81%	

From the results described above, including a good estimation of the distribution of the speed into the prediction always improves the routing decisions, just from recognizing the variability of the speed (from historical data) as part of the prediction.

Even though the improvement of this modelling scheme above the improvement resulting from the demand prediction seems not very impressive, the integrated approach should produce much better results as the variability of the speed (not only in time but also in space) became larger.

Next, a methodology to deal with unpredictable congestion is developed, under the same HPC formulation developed for recurrent congestion. By following the same line of reasoning as in the previous paragraph, in this case it will be tried to measure the impact of applying this approach to a scenario in which suddenly a big incident occurs, generating for a while a big congestion around the affected area.

The system should react in real-time to the occurrence of such an incident and take proper routing decisions taking into account such a change. Intuitively considerable cost savings in this case are expected, as shown next.

4.6. Fault tolerant control for abnormal situations in the Dial-a-ride system.

The approach described so far seems useful when a speed distribution is available and calibrated in both relevant dimensions, time and space. For that, a statistical work has to be conducted from historical data of the studied area, which allows us to have a good prediction of recurrent (predictable) traffic conditions. However, in real transportation networks, the unpredictable congestion events can also affect the expected vehicle travel times, resulting in bad quality routing with the occurrence of a big incident close to the dispatch areas. In order to incorporate such an effect, a fault detection and isolation (FDI) method is proposed for detecting the unpredictable traffic jam and a fuzzy fault tolerant control (FFTC) to force the vehicles avoiding the affected zones. Both systems will permit to reduce the effect of the incident over the users waiting and travel times. The unpredictable events will be detected and modelled by using real-time information from our vehicle fleet, noting that the method is easily extended to the use of any other sources of online speed data. In the literature, there are some preliminary results for fault detection problems and diagnosis in the transport infrastructure, like traffic monitoring sensors and vehicle mechanical systems Capriglione *et al.* (2004). Related with anomalies, Aronson *et al.* (2002) considers the re-route problem as incident repair method for a multimodal transport system; the considered incidents are changes in freight orders, traffic jams and vehicles faults. Weinstein (2005) present a model oriented to objects to describe the planning of multi-agent systems, which enables to diagnose the anomalies executions.

4.6.1. Procedure.

In this work, the measurements of $v(t,p,\varphi)$ are available for each position p at every instant time t. Besides, a recurrent model of the speed $\hat{v}(t,p)$ is assumed. The speed measurements are compared with the results of the speed distribution model and used for the FDI method. Analytically, the speed residual is given by $e(t)=\hat{v}(t,p)-v(t,p,\varphi)$. Thus, the residual $e(t)$ for a reasonable period of time *TT* is analyzed in order to activate the FDI system. If the system detects a fault during the entire period *TT*, the FDI system will be activated. During *TT*, the information of the real velocity is recorded to modify the recurrent model of velocity $\hat{v}(t,p)$ used by the HPC control strategy in order to avoid the negative effects of the incident. This procedure corresponds to the FFTC method.

After the FDI system is activated, a set of rules have to be defined in order to model the incident impact. These rules generate the new recurrent model that includes the original recurrent model $\hat{v}(t,p)$ and the fuzzy rules for the incident representation. The fuzzy approach is used in order to capture the non-linear behaviour of the incident impact. Moreover, these fuzzy rules permit to distinguish different magnitude and features of the incident.

First of all, the definition of the fuzzy rules require establishing the velocity associated with each type of incident, which is modelled by a Gaussian function (μ, σ, m). In the Gaussian model, μ is the location of the centre of the incident, σ is the affected zone radius and m represents the minimum velocity located at the centre where the incident is supposed to happen. These three parameters are adjusted based on the type of the incident. The duration of Gaussian model is assumed constant. The parameterσ is assumed to be inversely proportional to the Euclidean distance associated with the vehicle movement during *TT*, and μ is associated with the linear trajectory travelled by the vehicle. Analytically,

$$\sigma = \frac{1}{\left\| P_D - P_F \right\|}, \qquad \mu = P_D + \lambda \cdot \left(P_F - P_D \right),\ 0 \le \lambda \le 1 \tag{4.25}$$

where P_D is the position of the vehicle where the fault is detected and P_F the position of the same vehicle after *TT*.

Next, once the type of incident is established, the corresponding fuzzy rules are defined based on the expected behaviour of the system under incident conditions. These rules are fed by two inputs: the speed residual *e(t)* and the increment of the residual along the trajectory $de(t) = e(t) - e(t-1)$. The rule outputs are the movement size λ and the minimum velocity m for each type of incident, the latter proportional to $m^* = \max\left\{de(t), de(t-1)\right\}$. The fuzzy rules and their corresponding membership functions are defined in Figure 3.

R_1: If $e(t)$ is M and $de(t)$ is N then λ is 0.0, m is m^*

R_2: If $e(t)$ is M and $de(t)$ is Z then λ is 0.5, m is $1.1 \cdot m^*$

R_3: If $e(t)$ is M and $de(t)$ is P then λ is 1.0, m is $1.3 \cdot m^*$

R_4: If $e(t)$ is H and $de(t)$ is N then λ is 0.5, m is m^*

R_5: If $e(t)$ is H and $de(t)$ is Z then λ is 1.0, m is $1.2 \cdot m^*$

R_6: If $e(t)$ is H and $de(t)$ is P then λ is 1.0, m is $1.5 \cdot m^*$

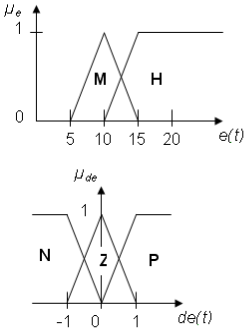


Figure 4.17. Fuzzy rules, and membership functions for the incident velocity model.

The proposed procedure FDI-FFTC method (as shown in Figure 4.18) consists of the following steps:

Step 1. When some vehicle detects the incident traffic jam for a certain period of time FDI is activated.

Step 2. A new recurrent model is generated by considering both the $\hat{v}(t,p)$ and the proposed fuzzy rules. The incident model based on fuzzy rules intends to represent the effects of the unpredictable event.

Step 3. The requests located somewhere inside the affected zone are re-assigned as new calls for the dispatcher system based on HPC, now considering the new recurrent model according to the new traffic conditions detected. As re-routing decisions of the re-assignment calls need to be fast, a one-step-ahead HPC is proposed ($S_F(k)$).

Step 4. After the re-routing, the new call requests are assigned by the HPC strategy $S(k)$ considering the same new recurrent model, and for the two-step-ahead case.

Step 5. If FDI system does not detect an incident, the HPC strategy described in Section 2 is used directly ($S(k)$) for the two-step-ahead case as well.

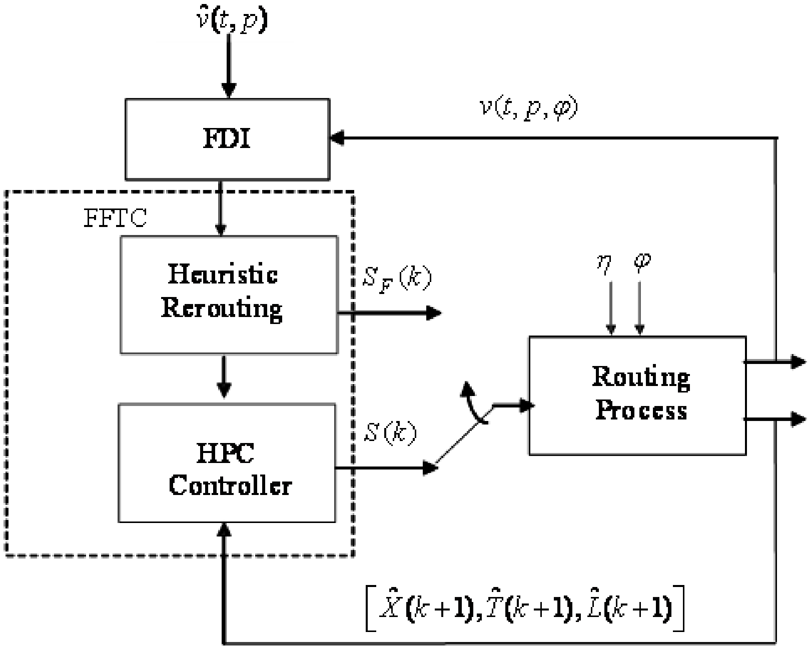


Figure 4.18. FDI-FFTC system for the dial-a-ride system.

4.6.2. Simulation results.

A reduced fleet of 4 vehicles in order to test the fault detection proposal. For the simulation test, 75 calls were generated over the whole simulation period of two hours. In Figure 4.19, the speed distribution defined in equation (4.24) is shown for four instant times. Figure 4.20 shows the recurrent model $\hat{v}(t,p)$ considered for the HPC before the incident. At 15:00, an incident happens (as shown in Figure 4.21) and thus, the fault detection module becomes active by checking the detection rules described in Section 4.6.1.

Table 4.13 reports the waiting time, travel time, total time, Operational cost and Effective total cost for two cases. The former (Case 1) considers the HPC controller by using the speed distribution from the initial recurrent model, without incorporating the incident that start getting reflected in the real speed data taken online by the fleet of vehicles. The latter (Case 2) considers the HPC scheme together with the proposed FDI detection system. Thus, the HPC approach considers a more realistic recurrent model that provides the effect of the incident. In addition, a third case is included as a benchmark, in which the HPC is applied by assuming completely known the distribution of the speed as a result of the incident occurrence (Case 3), and therefore, the routing decisions are preformed based on a velocity model including the fault effect (Figure 4.21).

Table 4.13. Performance comparison for fault detection method.

	Waiting time (min)	Travel time (min)	Total time (min)	Operational Cost (min)	Effective Total Cost (min)
	Mean	Mean	Mean	Mean	Mean
Case 1	9.5110	12.6994	22.2104	132.3360	687.3965
Case 2	7.9461	12.9906	20.9367	132.0360	659.7205
Improv. (%)	16.45%	-2.3%	5.73%	0.2%	4.01%
Case 3	8.1758	11.8525	20.0283	131.9050	632.6113
ΔImprov. (%)	-2.42%	8.96%	4.09%	0.1%	3.94%

The last row in Table 4.13 shows the additional improvement of Case 3 above Case 2 with respect to Case 1, to have an idea of how far the solution is from the ideal situation (Case 3) in which the incident (fault) is completely known at any time. The improvement in this particular case is of the order of 4% (Effective total cost) above the improvement of Case 1 over the model without including speed distribution in the prediction. A relevant improvement is observed in terms of waiting time in case of using the FDI-FFTC method (16.45%), in this case even better than having the information of the fault beforehand.

More tests have to be run in order to completely explain this last result. The intuition suggests that this apparent contradiction can be explained from a trade off between travel and waiting time, favouring the former in Case 3 due to the extra available information with regard to the fault location and impact. Case 2 anyway, performs quite well when compared against the benchmark (Case 3) in all cases, except in travel time, in which the fault detection does not help.

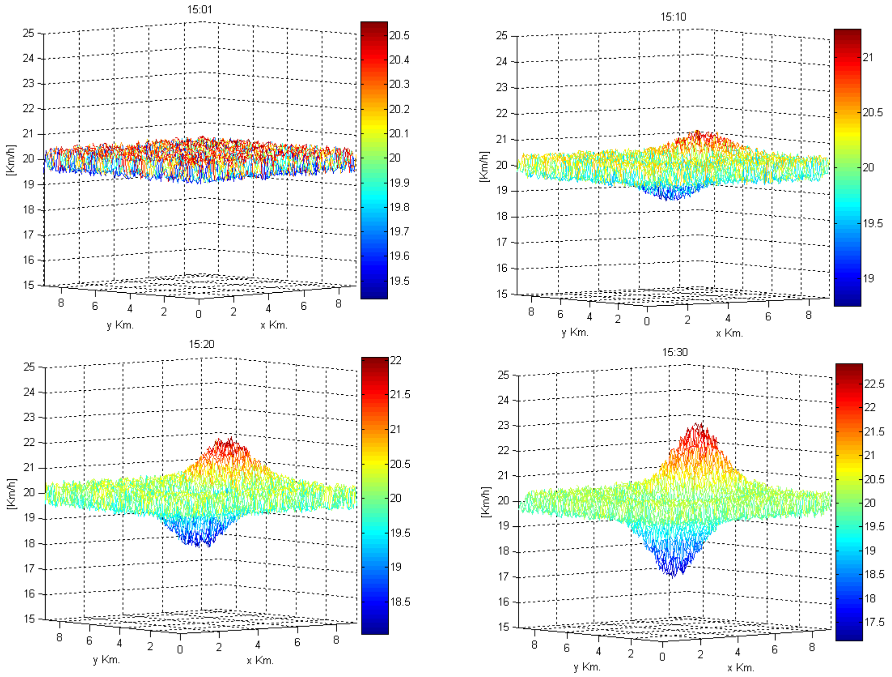


Figure 4.19. Real speed distribution without incident.

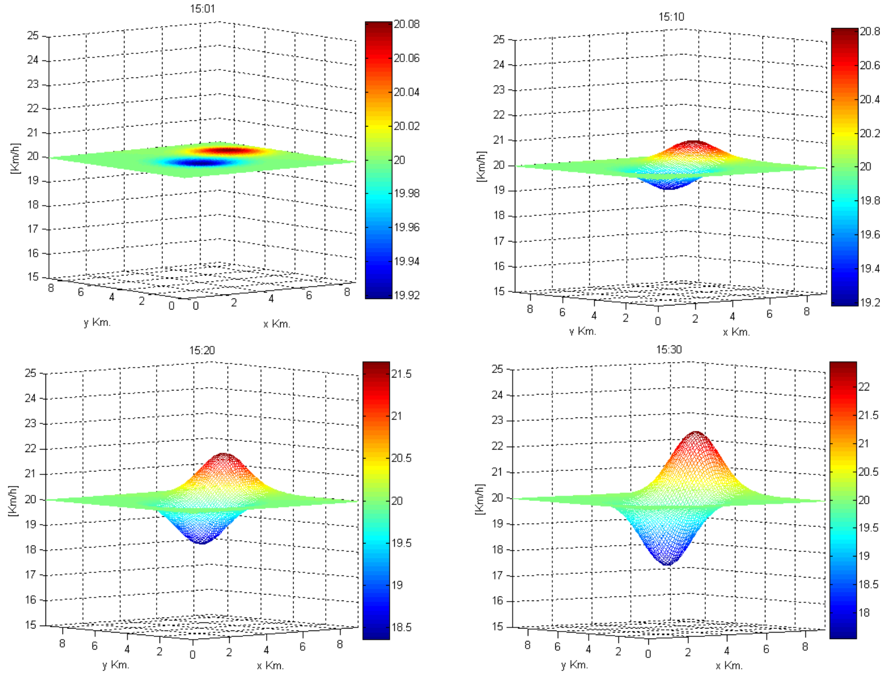


Figure 4.20. Speed distribution for the initial recurrent model.

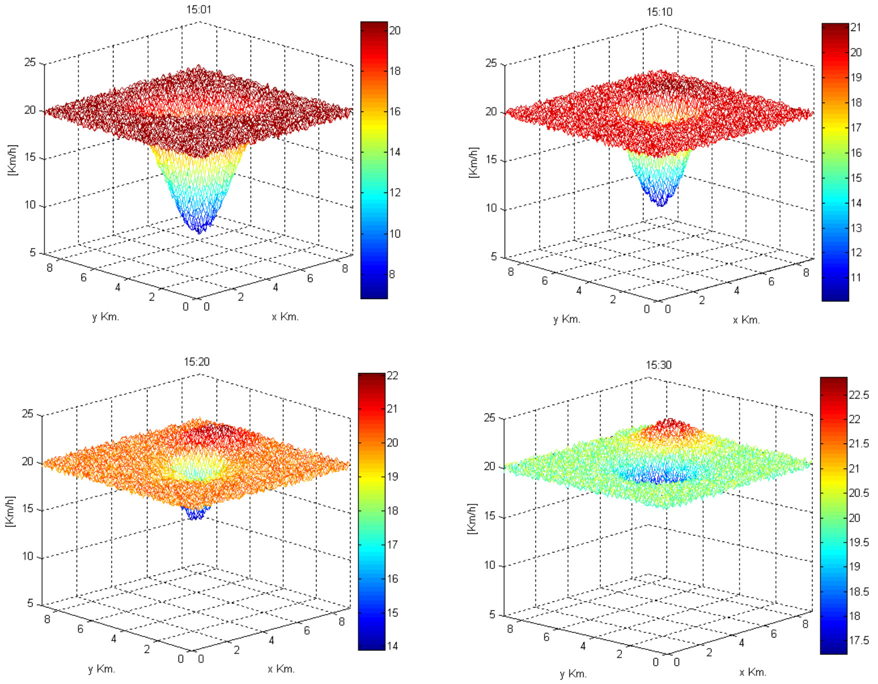


Figure 4.21. Real speed distribution with incident.

Finally, in Figure 4.22 the real situation is compared with the new speed model, which adaptively updates the fault detector whenever the vehicles of the fleet enter the fault impact zone and report its experienced speed. Thus, Figure 4.22a) has to be compared with Figure 4.22b), while Figure 4.22c) has to be compared with Figure 4.22d), for the real and modelled speed respectively at two instants. Results could improve considerably if more speed measurement stations were added to the detection system (both fixed and mobile stations).

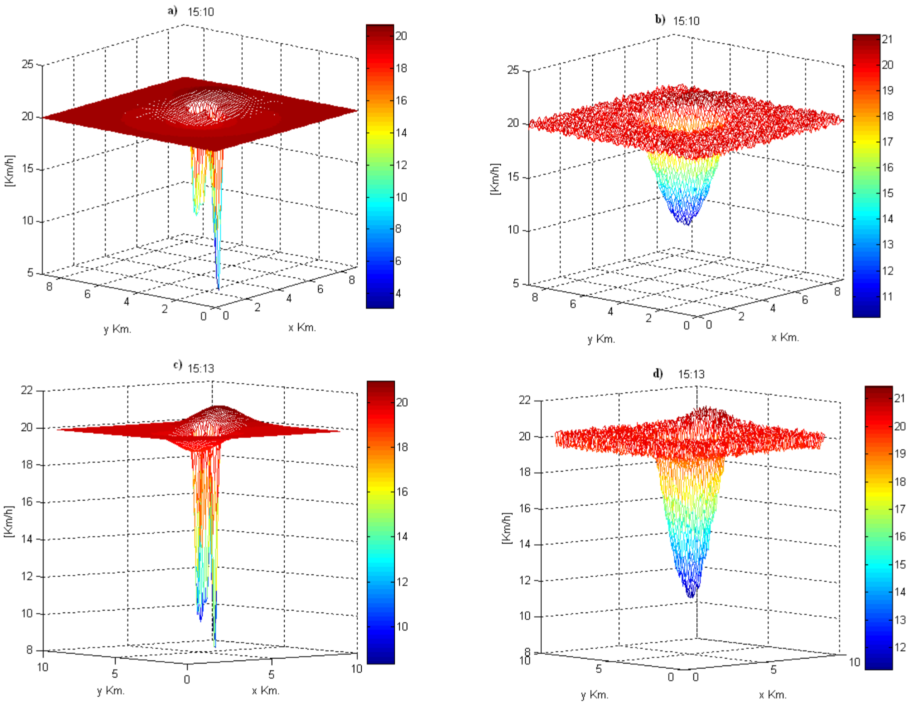


Figure 4.22. Comparison between model and real speed distribution with incident.

4.7. Discussion.

In this chapter an analytical formulation for the dial-a-ride system based on a HPC approach is developed considering historical demand information for a systematic future prediction to improve current dispatch decisions. There are three major contributions of this chapter. First, formal analytical formulations of the state space models are developed. Second, fuzzy zoning is utilized to compute probabilities and trip patters from historical data under more realistic scenarios. Third, based on such an analytical approach, GA are proposed and tested based upon a simulated example.

One major contribution of this formulation is the use of artificial intelligence methods to find better dynamic dispatching decisions under non-myopic scenarios (more than one-step-ahead prediction). Particularly, GA is presented as an efficient solver in computation times for this dial-a-ride system based upon a detailed analytical formulation. Under certain conditions, a scenario of more than two-step-ahead can be solved by using GA in reasonable computation time. The analytical formulation developed in this research can be potentially utilized to fit other numerical methods to solve the dial-a-ride system optimization process.

EE works quite well for small problems (for instance, few planned stops and few vehicles). However, as the problem size increases (for example, under more realistic systems), GA becomes an attractive alternative to solve such problems in manageable computation time. GA applied to this specific problem is a good option to face more complex problems (such as the use of longer sequences, more sophisticated objective functions, relaxed constraint problems, etc.). Note that choosing the number of individuals and generations is a critical point to get reasonable computation time as well as accurate results.

Moreover, a zoning method based on fuzzy clustering is proposed to systematically estimate origin-destination patterns from historical data and consequently obtain more reliable computations of the corresponding prediction probabilities. The proposed fuzzy zoning methodology improves the performance of predictive algorithms, mainly under more realistic historical data characterized by jumbled up trip patterns.

The integrated methodology (Fuzzy HPC based on GA) allows solving for more than two-step-ahead prediction to deal with uncertain and heterogeneous demand pattern scenarios. In a further application, to combine historical data (off-line) with online information is proposed in a more elaborate model able to capture imminent events in demand distribution that could affect the system performance. A fault detection scheme is suggested as it worked nice when detecting unpredictable traffic conditions.

A more complete rigorous expression for the objective function could be used. In the next chapter, in the context of a multi-objective approach to deal with a similar problem, a more realistic objective function is utilized, which can also applied to HPC mono-objective formulation, considering the impact of the rerouting on passengers together with non-linear behavior of the objective function weights according to the time each user has spent on the system. In addition more complex configurations could explore the inclusion of time windows (hard and soft), transfer points (in bus stops for example or another ad-hoc locations), and a better consideration of operational costs. A sensitivity analysis with regard to both parameters α and τ is planned to be also investigated, for two and three-step-ahead problems. It is possible to improve the estimation of tuning variables, such as number of probable calls, future step time prediction (τ) which is unknown, prediction horizon (*N*), service policy, search over different feasible solutions structures, etc. One nice problem could be to solve the version of the problem where the demand is well known a priori (as benchmark). Heuristic like evolutionary algorithms could be applied for finding a good solution in a reasonable computation time. The trade-off between accuracy and computation time should be considered.

In addition, the *no-swapping* operational policy will be also relaxed in further developments to test less restrictive dispatching rules, for which the analytical formulation approach would be useful. Partial-swapping, or local heuristics that improves the nodes where the last call was assigned could improve the performance, however, special attention should be to trying to keep the effect of the *N*-step-ahead predictions. For example, to repair a route without considering the future request could results in myopic assignations.

When considering the predictive velocity distribution, the presented HPC formulation for a dial-a-ride system combines two sources of uncertainty when making real-time vehicle routing decisions. On the one hand, the formulation considers uncertainty from possible future demand influencing routes of current customers, and on the other hand, the scheme also considers the uncertainty behind the traffic congestion conditions. The predictive model is proposed in order to modify the pre-planned schedule of vehicle routes based on traffic information around their routes as well as future insertions coming from unknown real-time service requests. In our approach, traffic congestion is modelled through the distribution of commercial speed of the vehicles on both relevant dimensions: time and space.

The approach allows modelling not only predictable congestion conditions, but also unpredictable situations, such as incidents occurring unexpectedly at any location on the traffic network. In the second case, online (real-time) data is used regarding speed conditions from the fleet of vehicles moving around serving the demand.

Results show the potential benefits of such an approach. Two important contributions of this matter can be mentioned. First, the integrated HPC allows systematizing the formulation of the dial-a-ride system as a control problem, which open more possibilities for using sophisticated techniques, not only to characterize the dynamic problem properly, but also to solve complex DPDP configurations unable to be treated without such a framework. Second, in the specialized literature there is no other dial-a-ride system formulation allowing prediction of both, future demand as well as future traffic conditions. Additional tests have to be conducted to adjust the embedded parameters and sophisticate the methods in order to get better solutions under realistic scenarios. Third, the occurrence of an incident can be treated under a FDI-FFTC scheme, allowing the reaction of the controller and the adjustment of the speed distribution parameters to significantly improve the dispatch rules under such a distorted scenario. The addition of the speed distribution into the model ensures a better estimation of both waiting and travel times, not only due to demand prediction but also because of traffic congestion predictions, generating better real-time routing decisions, and consequently better performance of the dispatch service. The more information we have the system, the better performance can be obtained from the HPC framework.

This chapter represents a first step in the elaboration of a sophisticated HPC approach to model dial-a-ride system and use prediction in the current decisions. The next step is to consider a real network configuration (with specific links and nodes) and replace the generic speed model in space by a velocity distribution model at a link level. This extension requires the coding of a time-dependent shortest path algorithm to compute optimal routes from point to point through the network, with link travel times depending on the time at which vehicles reach the upstream node of such a link. The coding can result harder, however the general framework remains the same. The use of traffic micro-simulation is proposed in order to have a better quantification of the performance of the system in real-time (simulation time). Better velocity models should result in better performance of the HPC scheme. In the case of unexpected incidents, a FDI-FFTC method is proposed. However, the rules can be further improved, sophisticating the way in which the system reacts to the occurrence of the detected fault. One straight extension is to somehow reroute those vehicles whose sequence path fall into the fault area, even though the associated stops are not inside the affected zone. Besides, the present formulation can be extended to the use of fixed stations monitoring traffic conditions at strategically chosen locations over the urban area, in order to have more data available to better trigger the FDI detection.

4.8. References.

Aronson, L.D., Van der Krogt, R.P.J, Zutt, J., (2002). "Automated Transport Planning using Agents". Proceedings of the International Congress on Freight Transport Automation and Multimodality: Organisational and Technological Innoventions (FTAM'02), May 2002. Available on http://pds.twi.tudelft.nl/~jonne/pubs.html

Babuska, R., (1999). "Fuzzy Modeling for Control". Kluwer Academic Publishers.

Berbeglia G, Cordeau J.F., Laporte G., (2009). "Dynamic Pick-up and Delivery Problems." In Press European Journal of Operational Research. doi:10.1016/j.ejor.2009.04.024.

Berman, O., Simchi-Levi, D., (1989). "The Traveler Salesman Location Problem on Stochastic Networks". Transportation Science 23, 54-57 (1989).

Bertsimas, D., Van Ryzin, G., (1991). "A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane", Operations Research 39, 601-615.

Bertsimas, D., Howell, L.H., (1993). "Further Results on the Probabilistic Traveling Salesman Problem". European Journal of Operational Research 65, 68-95.

Bezdek, J., (1973). "Fuzzy Mathematics in Pattern Classification". PhD Thesis, Applied Math. Center, Cornell University, Ithaca.

Capriglione, D., Liguori, C., Pietrosanto, A., (2004). "Analytical Redundancy for Sensor Fault Isolation and Accommodation in Public Transportation Vehicles". IEEE Transactions on Instrumentation and Measurement, 53(4), 993-999.

Carraway, R., Morin, T., Moskowitz, H., (1989). "Generalized Dynamic Programming for Stochastic Combinatorial Optimization", Operations Research 37, 819-829.

Cortés, C.E., (2003). "High-Coverage Point-to-Point Transit (HCPPT): A New Design Concept and Simulation-evaluation of Operational Schemes for Future Technological Deployment". Ph.D. Dissertation, University of California at Irvine, U.S.A.

Cortés, C.E., Jayakrishnan, R., (2004). "Analytical Modeling of Stochastic Rerouting Delays for Dynamic Multi-vehicle Pick-up and Delivery Problems". The fifth Triennial symposium on transportation analysis, TRISTAN V. 13-18 June. Le Gosier, Guadalupe.

Cortés, C.E., Sáez, D., Núñez, A., Muñoz, D., (2009). "Hybrid Adaptive Predictive Control for a Dynamic Pick-up and Delivery Problem". Transportation Science, Volume 43, February 2009, Pages: 27-42.

Crainic, T., Gendreau, M., Potvin J., (2009). “Intelligent freight-transportation systems: Assessment and the contribution of operations research”. Transportation Research Part C: Emerging Technologies 17(6), pp. 541-557.

Desrosiers, J., Soumis, F., Dumas, Y., (1986). “A Dynamic Programming Solution of a Large-Scale Single-Vehicle Dial-a-Ride with Time Windows”, American Journal of Mathematical and Management Sciences 6, 301-325.

Dial, R., (1995). “Autonomous Dial a Ride Transit – Introductory Overview”, Transportation Research - Part C 3, 261-275.

Dorigo, M., Stützle, T., (2004). “Ant Colony Optimization”. The MIT Press.

Dréo, J., Pétrowski, A., Siarry, P., Taillard, E., (2006). “Metaheuristics for Hard Optimization Methods and Case Studies”. Springer-Verlag.

Eksioglu B, Volkan A, Reisman A., (2009). “The Vehicle Routing Problem: A Taxonomic Review”. In Press Computers & Industrial Engineering. doi:10.1016/j.cie.2009.05.009

Filipec, M., Skrlec, D., Slavko, K., (1998). “An Efficient Implementation of Genetic Algorithms for Constrained Vehicle Routing Problem”. Proceedings of IEEE International Conference on System, Man and Cybernetics, SMC'98.

Fleishmann B., Gietz M., Gnutzmann S., (2004). “Time-Varying Travel Times in Vehicle Routing”. Transportation Science 38 (2), 160-173.

Floudas, C., (1995). “Non-linear and Mixed Integer Optimization”. Oxford University Press.

Gendreau M, Guertin F, Potvin J, Taillard E., (1999). “Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching”. Transportation Science 33. pp. 381-390.

Gendreau, M., Guertin, F., Potvin, J., Taillard, E., (1999). “Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching”, Transportation Science 33, 381-390.

George, A., Powell, W. (2005). "Adaptive Stepsizes for Recursive Estimation with Applications in Approximate Dynamic Programming", http://www.castlelab.princeton.edu/.

Godfrey, G., Powell, W.B., (2002). “An Adaptive Dynamic Programming Algorithm for Stochastic Resource Allocation Problems I: Single Period Travel Times”, Transportation Science 36, 21-39.

Haghani, A., Jung, S., (2005). “A dynamic vehicle routing problem with time-dependent travel times”. Computers & Operations Research 32. 2005. p. 2959-2986.

Hill A., Benton W (1992). “Modelling Intra-city Time-dependent Travel Speeds for Vehicle Scheduling Problems”. Journal on Operation Research Soc. 43, 343-351.

Ichoua, S., Gendreau, M., Potvin, J.Y., (2006). “Exploiting Knowledge about Future Demands for Real-time Vehicle Dispatching”, Transportation Science 40 (2), 211-225.

Ichoua, S., Gendreau, M., Potvin, J.-Y., (2007). “Planned route optimization for real-time vehicle routing”. In: Zeimpekis, V., Tarantilis, C.D., Giaglis, G.M., Minis, I. (Eds.), Dynamic Fleet Management: Concepts, Systems & Case Studies. Springer, New York, NY, pp. 1–18.

Jaw, J., Odoni, A., Psaraftis, H., Wilson, N., (1986). “A heuristic algorithm for the multivehicle many-to-many advance-request dial-a-ride problem”, Working paper MITUMTA-82-3, M.I.T., Cambridge, M.

Jih, W., Yun-jen, J., (1999). “Dynamic Vehicle Routing using Hybrid Genetic Algorithms”. Proceeding of the IEEE International Conference on Robotics & Automation, Detroit, Michigan. pp. 453-458.

Kao, E., (1978). “A Preference Order Dynamic Problem for a Stochastic Traveling Salesman Problem”, Operations Research 26, 1033-1045.

Keyton A., Morton D., (2003). “Stochastic Vehicle Routing with Random Travel Times”. Transportation Science, 37 (1), 69-82.

Kim, S., Lewis, M., White, C., (2005). “Optimal Vehicle Routing with Real Time Traffic Information”. IEEE Transactions on Intelligent Transportation Systems 6, 178-188.

Kleywegt, A.J., Papastavrou, J.D., (1998). “The Dynamic and Stochastic Knapsack Problem”, Operations Research, 46, 17-35.

Kleywegt, A.J., Papastavrou, J.D., (2001). “The Dynamic and Stochastic Knapsack Problem with Random Sized Items”, Operations Research 49(1), 26-41.

Larsen, A., (2000). “The Dynamic Vehicle Routing Problem”, Ph.D. Thesis, Technical University of Denmark.

Laporte, G., Louveaux, F., Mercure, H., (1992). “The Vehicle Routing Problem with Stochastic Travel Times”, Transportation Science 26, 161-170.

Lambert, V., Laporte, G., Louveaux, F., (1993). “Designing Collection Routes through Bank Branches”, Computers and Operations Research 20, 783-791.

Madsen, O., Raven, H., Rygaard, J., (1995). “A Heuristics Algorithm for a Dial-a-ride Problem with Time Windows, Multiple Capacities, and Multiple Objectives”, Annals of Operations Research 60, 193-208.

Malandraki, C., Daskin, M., (1992). “Time Dependent Vehicle Routing Problems: Formulations, Properties and Heuristic Algorithms”. Transportation Science 26, 185-200.

Man, K., Tang, K., Kwong, S., (1998). “Genetic Algorithms, Concepts and Designs”. Springer.

Montemanni, R., Gambardella, L., Rizzoli, A., Donati, A., (2005). “Ant Colony System for a Dynamic Vehicle Routing Problem”. Journal of Combinatorial Optimization, Issue: Volume 10, Number 4. 2005. p. 327 – 343.

Núñez, A., (2007). “Estrategias de Control Predictivo Hibrido y su aplicación al Ruteo Dinámico de Vehículos (in Spanish)”. Master Thesis University of Chile, Electrical Engineering Department.

Osman, M., Abo-Sinna, M., Mousa, A., (2005). “An Effective Genetic Algorithm approach to Multi-objetive Routing Problems (MORPs)”. Applied Mathematics and Computation, 163. pp. 769-781.

Papastavrou, J.D., Rajagopalan, G., Kleywegt, A.J., (1996). “The Dynamic and Stochastic Knapsack Problem with Deadlines”, Management Science 42, 1706-1718.

Psaraftis, H., (1980). “A Dynamic Programming Solution to the Single Many-to-many Immediate Request Dial-a-ride Problem”, Transportation Science 14(2), 130-154.

Psaraftis, H., (1988). “Dynamic Vehicle Routing Problems”, B.L. Golden and A.A. Assad editors, Vehicle routing methods and studies, 223-248.

Powell, W.B., (1988). “A Comparative Review of Alternative Algorithms for the Dynamic Vehicle Allocation Problem”, B.L. Golden and A.A. Assad editors, Vehicle routing methods and studies.

Sáez, D., Cortés, C.E., Núñez, A., (2008). “Hybrid Adaptive Predictive Control for the Multi-vehicle Dynamic Pick-up and Delivery Problem based on Genetic Algorithms and Fuzzy Clustering”. Computers & Operations Research, Volume 35, Issue 11, Date: November 2008, Pages: 3412-3438.

Savelsbergh, M., Sol, M., (1995). “The General Pick-up and Delivery Problem”, Transportation Science, 29 (1), 17-29.

Skrlec, D., Filipec, M., Krajcar, S., (1997). “A Heuristic Modification of Genetic Algorithm used for Solving the Single Depot Capacited Vehicle Routing Problem”. Proceedings of Intelligent Information Systems, IIS '97. 1997. p. 184-188.

Sniedovich, M., (1981). “Analysis of the Preference Order Traveling Salesman Problem”, Operations Research 29, 1234-1237.

Spivey, M., Powell, W.B., (2004). “The Dynamic Assignment Problem,” Transportation Science 38(4), 399-419.

Swihart, M.R., Papastavrou, J.D., (1999). “A Stochastic and Dynamic Model for the Single-Vehicle Pick-Up and Delivery Problem”, European Journal of Operational Research 114, 447-464.

Tarantilis, C., (2005). “Solving the Vehicle Routing Problem with Adaptive Memory Programming Methodology”. Computers & Operations Research 32, pp. 2309-2327.

Thomas, B.W., White III, C.C., (2004). “Anticipatory Route Selection”, Transportation Science 38(4), 473-487.

Tighe, A., Smith, F., Lyons, G., (2004). “Priority based Solver for a Real-time Dynamic Vehicle Routing”. IEEE International Conference on Systems, Man and Cybernetics. 2004. p. 6237-6242.

Tong, Z., Ning, L., Debao, S., (2004). “Genetic Algorithm for Vehicle Routing Problem with Time Window with Uncertain Vehicle Number”. Proceeding of the 5th World Congress on Intelligent Control and Automation, June 15-19, Hangzhou, P.R. China. 2004. p. 2846-2849.

Topaloglu, H., Powell, W.B., (2005). “A Distributed Decision-Making Structure for Dynamic Resource Allocation Using Non Linear Functional Approximations”, Operations Research 53(2), 281-297.

Toth, P., Vigo, D., (2003). “The Granular Tabu Search and its Application to the Vehicle-routing Problem”. Informs Journal on Computing 15 (4). pp. 333-346.

Weinstein, R., (2005). “RFID: A Technical Overview and its Application to the Enterprise”. IT Professional, 7 (3), 27–33.

Zhu, K., (2003). “A Diversity-controlling Adaptive Genetic Algorithm for the Vehicle Routing Problem with Time Windows”. 15th IEEE International Conference on Tools with Artificial Intelligence, November 3-5. pp. 176-183.

5. Hybrid Predictive Control based on MO for the Dial-a-ride System.

5.1. Literature review.

In this chapter the multi-objective hybrid predictive control (MO-HPC) framework (presented in chapter 3), is applied to the control of the dial-a-ride system shown in chapter 4.

As discussed in chapter 4 for the control of a dial-a-ride system, a well-defined dynamic problem should be based on an objective function that includes prediction of future demand in current routing decisions, issue not always well treated in the specialized literature. A recent and complete review of dynamic pickup and delivery problems can be found in Berbeglia *et al*. (2009), where general issues as well as solution strategies are described. They conclude that it is necessary to develop more studies on policy analysis associated with dynamic many-to-many pickup and delivery problems.

In chapter 4 of this thesis, an analytical formulation was proposed for the dial-a-ride problem as a hybrid predictive control problem using state space models and algorithms that come from the computational intelligence literature (GA and Fuzzy Clustering).

It seems reasonable that a proper definition of a predictive objective function includes both operator and user costs, computed from the estimated travel time for vehicles and users as well as waiting time for passengers before they are picked up. Thus, the formulation should properly quantify both the impact on the users' level of service affected by real-time routing decisions, as well as the effect on the associated extra operational costs.

It must be noticed that these two dimensions are opposite objectives. On the one hand, the interest of the operator is in minimizing operational costs, and, on the other, the users want to obtain good service, implying more direct trips, resulting in lower vehicle occupancy rates and consequently, higher operational costs to satisfy the same demand, for a fixed fleet. More efficient routing policies from the operator's standpoint will reflect higher occupation rates, longer routes, and consequently, longer waiting and travel time for users.

Thus, the question is how to properly balance both components in the objective function to make proper planning and fleet dispatching decisions. The answer has not been clarified yet in the literature. It depends on who makes the decisions and in what context. To guide the decisions makers in this line, in this chapter a multi-objective HPC (described in chapter 3) is proposed for solving the dial-a-ride problem treated before in chapter 4 under a mono-objective HPC scheme. In the current chapter, the Evolutionary Multi-objective Optimization (EMO) algorithm proposed in chapter 3 is used to solve the dynamic formulation of the dial-a-ride system, considering both opposite dimensions (operator and users) in the objective function.

As mentioned before in chapter 3, multi-objective optimization (MO) has been applied to a large number of static problems. Farina *et al*. (2004) showed several dynamic multi-objective problems found in the literature, pointing out the lack of methods that allow testing them adequately. The use of MO is not new in static vehicle routing problems. Yang *et al*. (2000) for a static vehicle routing problem (VRP) also realized the different goal pursued by users and operators in their costs. Tan *et al*. (2007) considered a multi-objective stochastic vehicle routing problem with limited capacity; for solving it, the authors proposed an evolutionary algorithm that incorporates two heuristics for local searching of optimal solution and simulation to obtain the fitness function. The authors show that the algorithm is capable of finding useful trade-offs and robust solutions.

A complete review of multi-objective vehicle routing problems can be found in Jozefowiez *et al.* (2008), where the different problems are classified according to their application (extension, generalization, or real-case study) and the components of the problem to which the objectives are related (tour, node/arc, or resources). Regarding the multi-objective algorithm for solving them, based on the survey, two main strategies are the most widely used. The first relies on scalar methods and the second relies on multi-objective evolutionary algorithms. The authors concluded the need to define general multi-objective vehicle routing problems as well as more efficient algorithms and operators.

As all the MO applications in VRP are static, the aim of this thesis chapter is to analyze the advantages of using MO for making dynamic decisions under a multi-objective optimization approach, relying on the Hybrid Predictive Control scheme proposed in chapter 3. One major goal of this development is to compare this new scheme with the HPC for a dial-a-ride described in chapter 4 of this thesis, based on a mono-objective and more simplistic objective function for dynamic dispatch decisions.

The multi-objective HPC (MO-HPC) optimization of the dial-a-ride system is non-linear and furthermore NP-Hard. Therefore, an ad-hoc evolutionary algorithm is reformulated for finding the multi-objective solution, which is the Pareto optimal set. The use of MO allows the decision-maker obtaining solutions that are not explored with the typical mono-objective HPC scheme. This extra information is a crucial support for the decision-maker who is finally looking for reasonable options of service policies not only for users but also for operators.

The outline of the chapter is as follows. In Section 5.2 MO-HPC, the dial-a-ride problem and model are summarized. In Section 5.3 MO-HPC is applied to the dial-a-ride problem under two different dynamic objective functions. Simulation results are shown and analyzed, and finally the discussion is highlighted.

5.2. Multi-objective Hybrid Predictive Control (MO-HPC) for the Dial-a-ride.

5.2.1. Multi-objective Hybrid Predictive Control (MO-HPC).

The notation hereafter is similar to that used in the previous chapter for defining a multi-objective problem in two dimensions (J_1 and J_2 to denote the objective functions commanding the process.

In the context of solving a dial-a-ride problem the MO-HPC is dynamic, meaning that real-time decisions related to a service policy are made as the system progresses; for example, the dispatcher could minimize the operational costs J_2 keeping a minimum acceptable level of service for users (through J_1) when deciding an assignment vehicle-user. Nevertheless, this tool could be implemented as a reference to support the dispatcher decision, which has the flexibility of deciding which criterion is more adequate. In this kind of problems, MO-HPC suits very well, as its helps the dispatcher to choose a solution to be applied, considering the trade-off between Pareto optimal solutions. Figure 5.1 shows an example of the dynamic evolution of Pareto front.

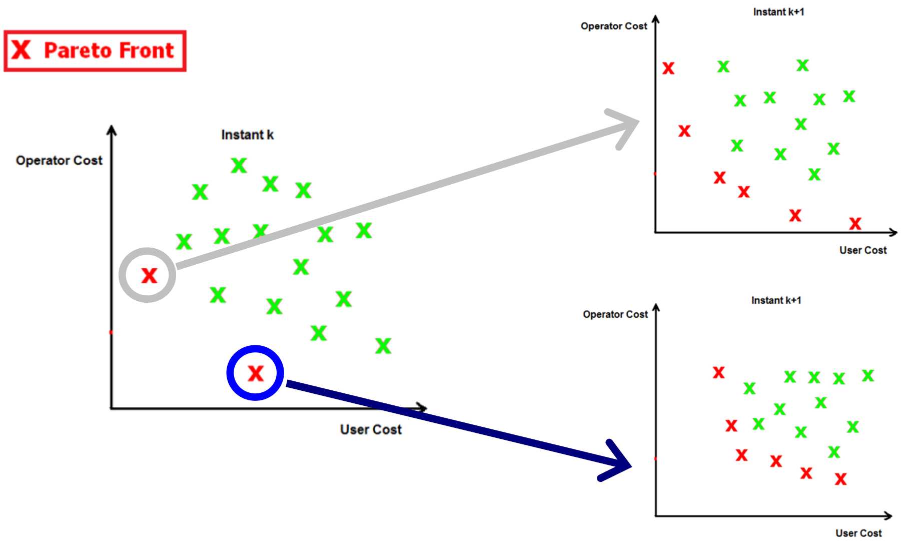


Figure 5.1. Diagram of the MO-HPC for dial-a-ride

As Figure 5.1 shows, the dispatch decision in current instant *k* will affect the Pareto front curve in the following instants. In the figure we appreciate that the decision at instant *k* will strongly affect the evolution of the Pareto front that is formed in the next steps (*k*+1, *k*+2, and so on).

In the next section, the details of MO-HPC with regard to the implementation of these techniques to a dial-a-ride system are described.

5.2.2. Extended Model of the Dial-a-ride Systems.

The extended predictive model for the dial-a-ride system, is formulated in terms of three variables: estimated time of arrival to a stop, vehicle load among stops, and vehicle position. In order to compute these variables, the same two sources of stochasticity considered in chapter 4 are included. The first regarding the unknown future demand entering the system in real-time, and the second coming from the network traffic conditions, in its spatial and temporal dimensions. For this application, let us assume a fixed and known fleet size F over an urban area A. The specific location of a request (which includes its pickup as well as its delivery) is known only after the associated call is received by the dispatcher.

A selected vehicle is then rerouted at real-time to insert the new request into its predefined route (sequence) while the vehicles are in motion. The assignment of the vehicle and the insertion position of the new request into the previous sequence of tasks associated with such a vehicle, are control actions decided by the dispatcher (controller) based on the objective function, which depends on the variables related to the state of the vehicles in real time (following the same procedure used in chapter 4). The fleet is in operation travelling within the area according to predefined routes. In this extended formulation, the service demand η_k that appears in real-time is described slightly differently from what was proposed in the previous chapter, namely $\eta_k = \left[p_k \quad d_k \quad t_{0k} \quad tr_k \quad r_k \quad \Omega_k \right]$. The demand is characterized by two positions, pickup and delivery p_k, d_k, and by the instant of the call occurrence t_{0k}. The expected minimum arrival time tr_k corresponds to the best possible service for that passenger, considering no re-routing of his(her) trip (shortest path) and a waiting time from the call instant associated with the closest available vehicle (in terms of capacity) to pick that passenger up. r_k is the label that identifies the passenger who is making the call, and finally Ω_k denotes the number of passengers waiting there (size of the request).

As stated before, *k* represents the k^{th} instant in the discrete events sequence. At any instant *k*, each vehicle j has assigned a sequence of tasks, which includes several points of pickup and delivery. Recalling some definitions from the previous chapter, the sequences are represented by the function $S_j(k) = \left[s_j^0(k) \quad s_j^1(k) \quad \cdots \quad s_j^i(k) \quad \cdots \quad s_j^{w_j(k)}(k) \right]^T$. Note the set of sequences $S(k) = \left\{ S_1(k), ..., S_j(k), ..., S_F(k) \right\}$ associated with the fleet of vehicles correspond to the control (manipulated) variable $u(k)$, and its specification is detailed in equation (4.1).

In Figure 5.2 an example of a sequence is shown, in order to introduce the extended formulation proposed here for MO. Users labeled as "r_1=1", "r_2=2" and "r_3=3" are assigned to vehicle j. The sequence assigned considers to pick up user "1" (coordinate 1^+), then to pick up user "3" (coordinate 3^+), then to delivery user "1" (coordinate 1^-) and so on. In the figure, users "1" and "3" will experience longer travel times due to rerouting. A different situation happens with user "2" whose pickup occurs just before the delivery. However the sequence could be improved for user "2" if the first stop of the vehicle sequence were the pickup of user

“2” and then its delivery. Then the controller must decide which sequence is better in order to keep a desired user police, and a minimum operation cost.

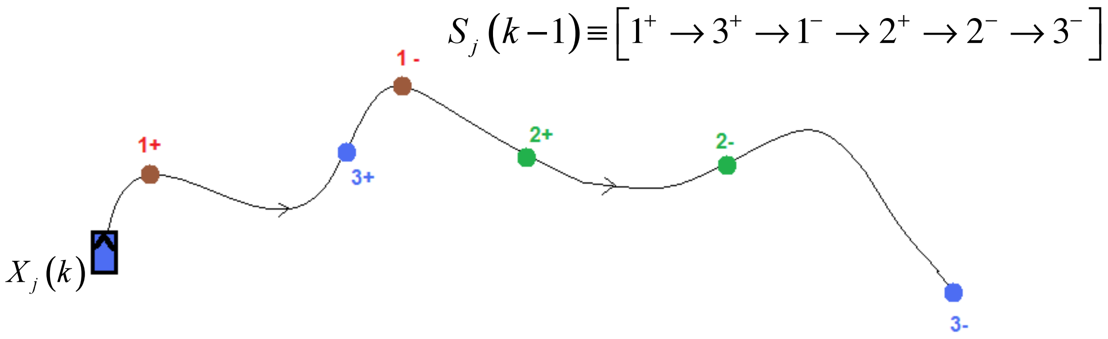

$S_j(k-1) \equiv \left[1^+ \rightarrow 3^+ \rightarrow 1^- \rightarrow 2^+ \rightarrow 2^- \rightarrow 3^-\right]$
1 -
1+
3+
2+
2-
3-
$X_j(k)$

Figure 5.2. Representation of sequence of vehicle *j* and its stops.

In Figure 4.2 another sequence assigned to vehicle j at instant k is shown. $\hat{T}_j^i(k)$ denote the expected departure time of vehicle j from stop i, $\hat{L}_j^i(k)$ the expected load of vehicle j when leaving stop i, and $X_j(k)$ representing the current position of the vehicle at instant k. In the present work, the traffic conditions are modeled by means of a commercial distribution of speeds associated with the vehicles. This distribution considers two dimensions: spatial and temporal. The real distribution of speeds is assumed to be unknown (denoted by $v(t,p,\varphi(t))$) which depends on a stochastic source $\varphi(t)$, representing the traffic conditions of the network as they change in time, and of a position p. A conceptual network will be assumed in this work, where the trajectories are defined as the straight line that joins two consecutive stops. Besides, a speed distribution for the urban zone during a typical period of recurrent congestion, represented by a speed model $\hat{v}(t,p)$, is supposed to be known, which could be obtained from historical speed data.

The closed loop of the dynamic vehicle routing system under MO-HPC is shown in Figure 5.3. The HPC represented by the dispatcher makes the routing decisions in real-time based on the information of the system (process) and the values of the fleet attributes, which allow evaluating the model under different scenarios. Service demand η_k and traffic conditions $\varphi(t,p)$ are disturbances in this system.

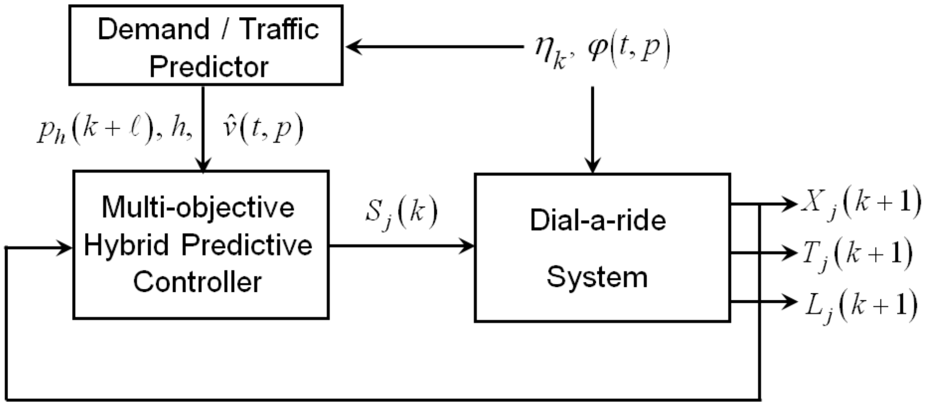

Demand / Traffic Predictor
η_k, $\varphi(t,p)$
$p_h(k+\ell)$, h, $\hat{v}(t,p)$
Multi-objective Hybrid Predictive Controller
$S_j(k)$
Dial-a-ride System
$X_j(k+1)$
$T_j(k+1)$
$L_j(k+1)$

Figure 5.3 Closed loop diagram of the HPC/MO-HPC for the dynamic dial-a-ride problem.

To apply the HPC and the MO-HPC approach, a new dynamic model is proposed to represent the routing process (an extension of model in chapter 4).

For vehicle *j*, the state space variables are the position $X_j(k)$, the estimated departure time vector $\hat{T}_j(k) \in R^{w_j(k)+1}$ and the estimated vehicle load vector $\hat{L}_j(k) \in R^{w_j(k)+1}$. The dynamic model for the vehicle *j* variables is the following.

$$\hat{X}_j(k+1) = \begin{cases} P_j^{i^*}(k) + \int_{t_k}^{t_k+\tau} \hat{v}(t,p(t)) \dfrac{\left(P_j^{i^*+1}(k) - P_j^{i^*}(k)\right)}{\left\| P_j^{i^*+1}(k) - P_j^{i^*}(k) \right\|_2} dt & \text{if } \ i^* < w_j(k) \\ P_j^{i^*}(k) & \text{if } \ i^* = w_j(k) \end{cases} \tag{5.1}$$

$$\hat{T}_j^i(k+1) = \begin{cases} T_j^0(k) & i = 0 \\ t_k + \sum_{s=1}^{i} \kappa_j^s(k) & i \neq 0 \end{cases}, \qquad i = 0,1,\dots,w_j(k) \tag{5.2}$$

$$\hat{L}_j^i(k+1) = \begin{cases} L_j^0(k) & i = 0 \\ L_j^0(k) + \sum_{s=1}^{i} \left(2z_j^s(k) - 1\right)\Omega_j^s(k) & i \neq 0 \end{cases}, \qquad i = 0,1,\dots,w_j(k) \tag{5.3}$$

Equations (5.1), (5.2) and (5.3) are explained in section 4.2.

The proposed vehicle sequences and state variables satisfy a set of constraints given by the real conditions of the dial-a-ride problem. Specifically, must be considered the constraints of precedence, capacity and consistency in the solution of the MO-HPC problem to generate only feasible sequences, as explained in chapter 4 in detail.

In the next section, two experiments with different MO-HPC formulations are conducted. In the first one, the same objective function used in chapter 4 is proposed for a small fleet of vehicles. As some users become particularly annoyed because their services were postponed, a new objective function is proposed and used to control with MO-HPC a larger fleet of vehicles.

5.3. Objective Function Design and Simulation Results.

5.3.1. MO-HPC for the Dial-a-ride System.

The motivation of this MO formulation is to provide to the dispatcher an efficient tool that captures the trade-off between users and operator costs. The objective of the HPC is to minimize an objective function from which the best routes for the vehicles will be selected. The proposed objective function quantifies the costs over the system of accepting the insertion of a new request. Such a function incorporates two decision dimensions, which normally move in opposite directions. The first component is the users' cost which includes both waiting and travel time experienced by each passenger. The second component is the cost associated with the operation of vehicles. In this approach, the latter cost incorporates two types of expenses: the cost per travelled distance unit and the cost spent by operating the vehicles in time units. A fixed fleet size is considered.

5.3.1.1. HPC for a Dial-a-Ride System.

A reasonable prediction horizon *N* is defined, which depends on the problem in study and on the intensity of unknown events, which can occur in the system in real time. If the prediction horizon is greater than one, the controller adds the predictive characteristic into the decision. The controller will compute the decisions for the complete control horizon *N*, namely

$S_k^{k+N}=\{S(k),...,S(k+N-1)\}$, considering the predictions based on historical data, and will apply only the sequence decided for the current instant $S(k)$ to the system according to the rolling horizon method. The performance of the vehicle routing scheme will depend on how well the objective function can predict the impact of possible rerouting, due to insertions caused by unknown service requests. Analytically, a mono-objective version of the proposed objective function for a prediction horizon *N*, can be written as follows:

$$\underset{S_k^{k+N}}{Min}\ \lambda J_1+(1-\lambda)J_2$$

$$J_1=\sum_{\ell=1}^{N}\sum_{j=1}^{F}\sum_{h=1}^{h_{\max}(k+t)} p_h(k+\ell)\left(J_j^U(k+\ell)-J_j^U(k+\ell-1)\right) \tag{5.4}$$

$$J_2=\sum_{\ell=1}^{N}\sum_{j=1}^{F}\sum_{h=1}^{h_{\max}(k+t)} p_h(k+\ell)\left(J_j^O(k+\ell)-J_j^O(k+\ell-1)\right)$$

where

$$J_j^O(k+\ell)=\sum_{i=1}^{w_j(k+\ell)}\left(c_T\left(\hat{T}_j^i(k+\ell)-\hat{T}_j^{i-1}(k+\ell)\right)+c_L D_j^i(k+\ell)\right) \tag{5.5}$$

$$J_j^U(k+\ell)=\sum_{i=1}^{w_j(k+l)}\left(\underbrace{\theta_v\hat{L}_j^{i-1}(k+\ell)\left(\hat{T}_j^i(k+\ell)-\hat{T}_j^{i-1}(k+\ell)\right)}_{J\ \text{TRAVEL TIME}}\right. \\ \left.+\underbrace{\theta_e z_j^i(k+\ell)\left(\hat{T}_j^i(k+\ell)-T_j^0(k+\ell)\right)}_{J\ \text{WAITING TIME}}\right) \tag{5.6}$$

In (5.4), J_j^U and J_j^O denote the user and operator costs respectively, associated with the sequence of stops that vehicle *j* must follow at certain instant. In equations (5.4)-(5.6), $k+\ell$ is the instant at which the ℓ^{th} request enters the system, measured from instant *k*. $h_{\max}(k+\ell)$ is the number of possible call patterns at instant $k+\ell$, $p_h(k+\ell)$ is the probability of occurrence of the h^{th} request, associated with a trip pattern related to a specific pair of zones. The occurrence probabilities $p_h(k+\ell)$ associated with each scenario are parameters in the objective function and must be calculated based on real time or historical data, or a combination of both. In chapter 4 a zoning based method for trip patterns estimation based on Fuzzy Clustering was designed. Expressions (5.5) and (5.6) represent the operator and users cost functions related to vehicle *j* at instant $k+\ell$, which depend on the previous sequence $S_j(k+\ell-2)$ and a new potential request *h* which occurs with probability $p\ (k+\ell)$; $w_j\,(k+\ell)$ is the number of stops estimated for vehicle *j* at instant $k+\ell$. The travel time is weighted by a factor θ_v, and the term related to waiting time is weighted by θ_e. Similarly, we will assume a generic expression for the vehicle operation cost (5.5), with a component which depends on the total traveled distance, weighted by a factor c_L, and another which depends on the total operational time, in this case at unitary cost c_T. Thus, $D_j^i\,(k+\ell)$ represents the distance between stops *i-1* and *i* in the sequence of vehicle *j*. Given the mono-objective nature of this formulation, expression (5.4) is generalized assuming an arbitrary factor λ to be defined by the decision maker.

5.3.1.2. MO-HPC for a Dial-a-Ride System.

The MO-HPC strategy is a generalization of HPC where the optimal control action is selected based on a criterion that takes solutions from the optimal Pareto region considering the following multi-objective problem:

$$\begin{gathered}\underset{S_k^{k+N}}{Min}\ \{J_1, J_2\}\\ J_1=\sum_{\ell=1}^{N}\sum_{j=1}^{F}\sum_{h=1}^{h_{\max}(k+t)} p_h\,(k+\ell)\big(J_j^U\,(k+\ell)-J_j^U\,(k+\ell-1)\big)\\ J_2=\sum_{\ell=1}^{N}\sum_{j=1}^{F}\sum_{h=1}^{h_{\max}(k+t)} p_h\,(k+\ell)\big(J_j^O\,(k+\ell)-J_j^O\,(k+\ell-1)\big)\end{gathered}\tag{5.7}$$

with J_1 and J_2 corresponding to the objective functions defined in (5.4). Note that this scheme does not need to define an arbitrary parameter λ as stated in (5.4).

The solution to this problem corresponds to a set of control sequences, which form the optimal Pareto set. It is considered that $S^i=\{S^i\,(k),...,S^i\,(k+N-1)\}$ is a feasible control action sequence. In this case, as the control sequences are defined within a feasible finite set, the resulting optimal Pareto front corresponds to a set with a finite number of elements.

From the Optimal Pareto front solutions for the dynamic MO-HPC problem, it is necessary to select only one control sequence $S^i=\{S^i\,(k),...,S^i\,(k+N-1)\}$ and from that, apply the control action $S^i\,(k)$ to the system according to the rolling horizon concept. For the selection of this sequence, a criterion related to the importance given to both the user (J_1) and operator (J_2) costs in the final decision is needed. We must point out that the solutions obtained from the MO problem form a set, which includes as a particular case, the optimal point obtained by solving the mono-objective problem. Furthermore, an analytical relation between both solutions can be established; such a relation in the mono-objective case can be represented by the proper selection of the weight factor λ.

A relevant step of this approach in the controller's dispatch decision is the definition of criteria to select the best control action at each instant under the MO-HPC approach. For example, once the Pareto front is found, different criteria regarding a minimum allowable level of service can be dynamically used to take policy dependent routing decisions. Three criteria for level of service will be evaluated:

Criterion 1: user cost under $ P1 per passenger.
Criterion 2: user cost under $P2 per passenger.
Criterion 3: user cost under $P3 per passenger.

P1<P2<P3. In cases where the policy is accomplished for several solutions, the one that minimizes the operator cost will be selected. If the policy cannot be respected (no feasible solution for such a policy exists), the best solution found (the closest to the policy boundaries) is applied. Results and analysis of these operation policies from simulations are reported in Section 5.3.1.3.

5.3.1.3. Simulation Results.

In this section we summarize the simulation tests conducted to show the MO-HPC approach application. A period of two representative hours is simulated, over a service urban area of approximately 81 km^2. A fleet of four vehicles is considered, with a capacity of four passengers each. Assume that the vehicles travel through a straight line between stops and on a transport network that behaves according to an unknown speed distribution. Also assume that the future calls are unknown for the controller. However, historical data is available from where the speed distribution model and typical trip patterns can be extracted. The speed distribution is shown in Figure 5.4 and the historical data generated by simulation follow the trip patterns (arrows) in

Figure 5.5. From the historical data and the fuzzy zoning method proposed in chapter 4, the pickup and delivery coordinates and probabilities are shown in Table 5.1.

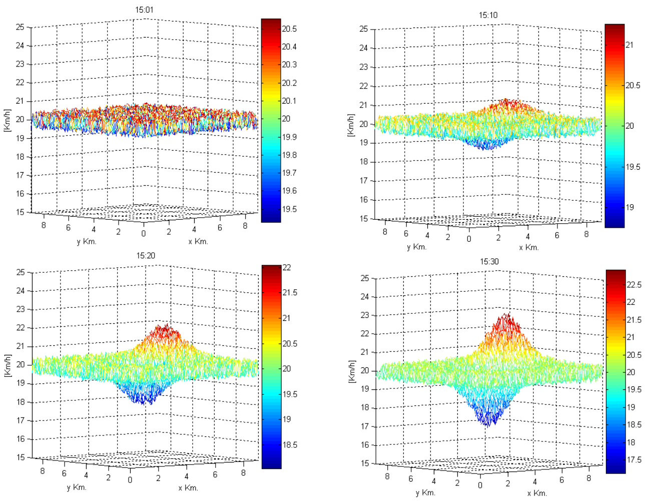


Figure 5.4 Speed distribution.

Table 5.1. Pickup and delivery coordinates and probabilities: Fuzzy zoning.

X pickup	Y pickup	X delivery	Y delivery	Probability
1.9197	2.9006	5.8348	3.1773	0.1027
2.0155	2.9567	3.2173	6.1373	0.1027
1.9364	2.9974	6.5663	6.0621	0.3836
5.5138	2.9972	6.5569	6.0215	0.411

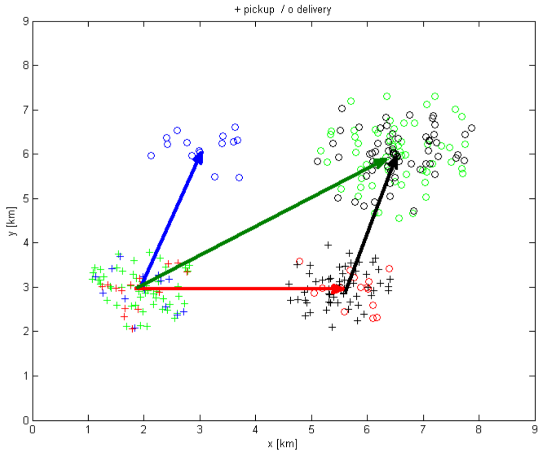


Figure 5.5 Origin-destination trip patterns

Sixty calls were generated over the simulation period of two hours following the spatial and temporal distribution observed from the historical data. Regarding the temporal dimension, a negative exponential distribution for time intervals between calls with rate 2 [call/minute] for both hours of simulation was assumed. Regarding the spatial distribution, the pickup and delivery coordinates were randomly generated within each zone.

The 10 first calls at the beginning and the 10 last calls at the end of the experiments were deleted from the statistics to avoid boundary distortion (warm up period). 10 replications of each experiment were carried out to obtain the global statistics. Each replication took 20 minutes in average, in a Pentium® D, 2.40Ghz processor.

The objective function is formulated at two-steps-ahead, considering parameters: θ_v=16,7[$/min], θ_e =50[$/min], c_T=25[$/min], c_L=350[$/Km]. P1=1000, P2=1125, P3=1250.

The first set of results were of the HPC approach with mono-objective functions, computed for weights $\lambda=1,\ 0.75,\ 0.5,\ 0.25$ and 0, in order to verify that the objectives pursued by users and operator are effectively opposite. Table 5.2 shows average values per user or vehicle according to the case. In order to analyze and evaluate the performance of the MO-HPC strategies, simulations for two-steps-ahead prediction were performed, under the same conditions.

The results are reported in Table 5.3, showing the effective user waiting and travel time, and the average travel time and distance associated with vehicles, for the MO-HPC, with $N=2$ and the three criteria of level of service proposed in Section 5.4.1.2.

Figure 5.6 shows the global results obtained from both approaches: HPC and HPC-EMO, detailing the cost components to global users and operators using the different criteria. The MO-HPC approach generates a range of options for the decision maker to decide the operation policy in real time with richer information, not possible to be provided with a traditional HPC approach. Furthermore, it is possible to add solutions under certain criteria (motivated by user level of service as well as operation savings). In this work three service level criteria were explored. Under criterion 1 we obtained a user cost equal to $1014.4 similar to the $1000 constrained by the service policy. Under criterion 2 a user cost equal to $1088.86 lower than the $1125 specified in the service policy is obtained. Finally, under criterion 3 we obtained a user cost equal to $1177.7 which is lower than $1250 so the service policy is also fulfilled here.

Table 5.2: HPC with different weighting factors.

Weight factor λ	Travel time [min/pax]	Waiting time [min/pax]	Travelled time [min/veh]	Distance travelled [km/veh]
$\lambda = 0$	14.0512	25.3705	82.4936	21.8086
$\lambda = 0.25$	16.2678	12.7871	106.2952	26.8951
$\lambda = 0.5$	16.4896	10.4631	111.3786	27.4946
$\lambda = 0.75$	15.8964	9.4583	113.7029	28.6032
$\lambda = 1$	16.2400	8.4579	121.7460	30.8408

Table 5.3: MO-HPC different criteria.

MO Criteria	Travel time [min/pax]	Waiting time [min/pax]	Travelled time [min/veh]	Distance travelled [km/veh]
Criterion 1	15.8817	14.9941	94.4766	27.3942
Criterion 2	15.3825	16.6497	91.7576	26.8549
Criterion 3	14.8654	18.5962	88.5647	24.1264

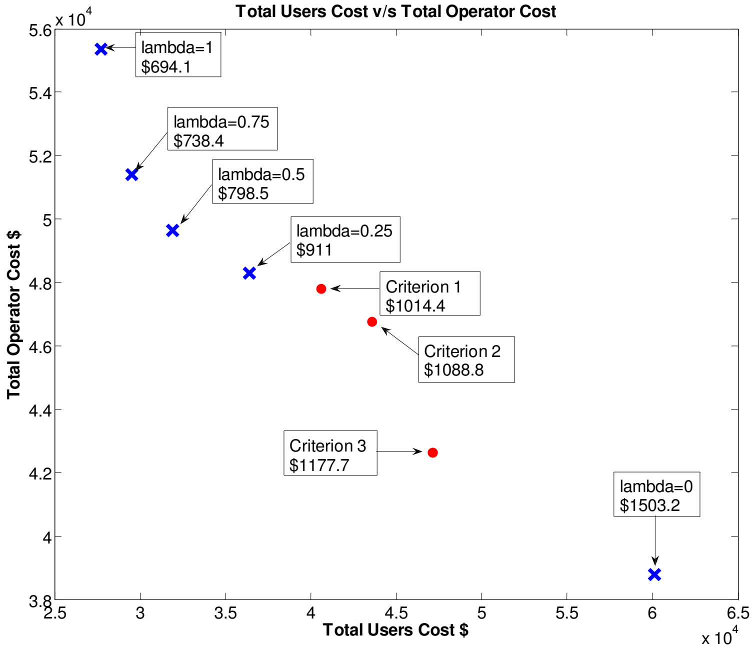


Figure 5.6 Global statistics. HPC with different lambda values and solutions with EMO criteria.

5.3.2. MO-HPC for the Dial-a-Ride System with a User Service Policy.

After analyzing the previous results, several issues regarding users' level of service were raised. To handle some undesired situations in that sense, a new objective function was designed, able to account for the fact that some users can become particularly annoyed if their service is postponed (either pick-up or delivery). See for example in Figure 5.7 the type of situations that could arise if such issues are not considered. In the Figure, circles represent annoyed users in a typical dynamic setting. In a proper formulation, a higher weight in the objective function should penalize differently very-long waiting or travel times. Next in this section, these ideas are formalized in an analytical proposal.

In this section, a mono-objective function of the hybrid predictive controller is defined, which chooses the best routes and vehicles to serve the dynamic demand. The proposed objective function quantifies the costs over the system of accepting the insertion of a new request. Such a function incorporates two dimensions, which as mentioned before, normally move towards opposite directions. The first component that takes into account the users' cost, includes both waiting and travel time experienced by each passenger. The second component is associated with the operational cost of running the vehicles of the fleet. After highlighting the details of the mono-objective specification, the function is extended to a multi-objective structure in order to compare the results.

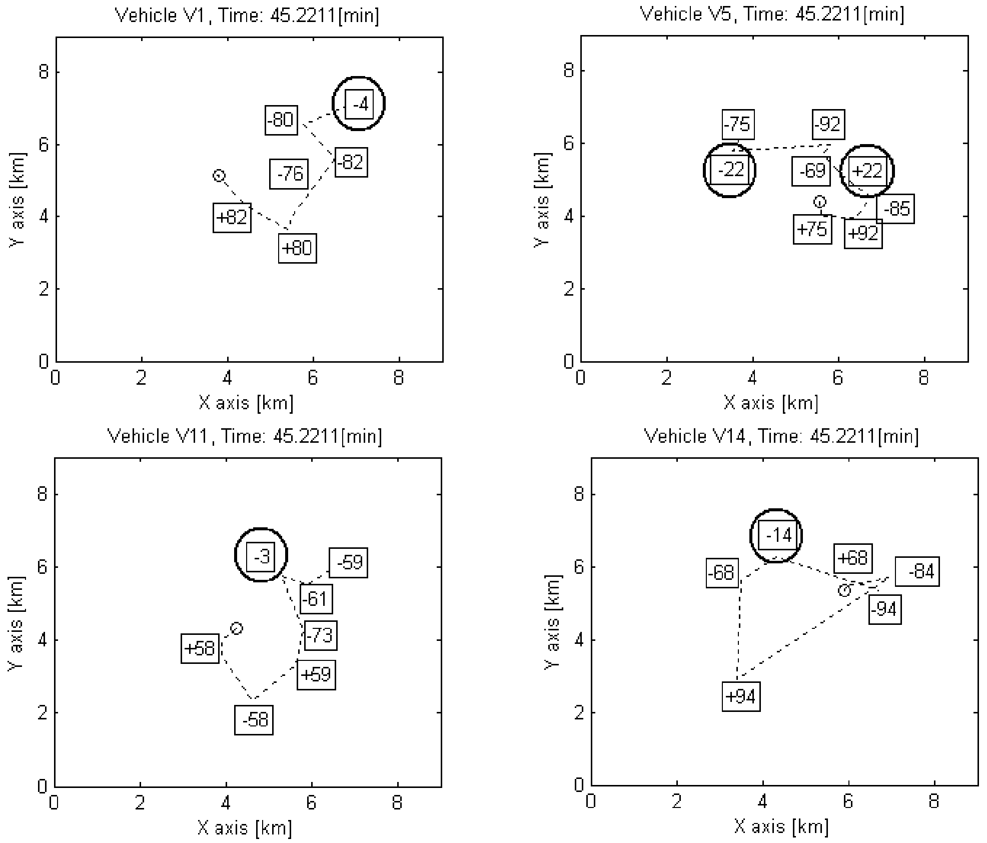


Figure 5.7. Routes showing postponed users.

5.3.2.1. HPC for a Dial-a-Ride System.

The performance of the vehicle routing scheme will depend on how well the objective function can predict the impact of possible rerouting due to insertions caused by unknown service requests. Analytically, a mono-objective version of the proposed objective function for a prediction horizon *N*, can be written as follows:

$$\begin{gathered}\underset{S_k^{k+N}}{Min}\ \lambda J_1+(1-\lambda)J_2\\ J_1=\sum_{\ell=1}^{N}\sum_{j=1}^{F}\sum_{h=1}^{h_{\max}(k+t)}p_h(k+\ell)\left(J_j^U(k+\ell)-J_j^U(k+\ell-1)\right)\\ J_2=\sum_{\ell=1}^{N}\sum_{j=1}^{F}\sum_{h=1}^{h_{\max}(k+t)}p_h(k+\ell)\left(J_j^O(k+\ell)-J_j^O(k+\ell-1)\right)\end{gathered}\tag{5.8}$$

where

$$J_j^O(k+\ell)=c_T\left(\hat{T}_j^{w_j(k+\ell)}(k+\ell)-\hat{T}_j^0(k+\ell)\right)\Big|_h+c_L\sum_{i=1}^{w_j(k+\ell)}\left(D_j^i(k+\ell)\right)\Big|_h\tag{5.9}$$

$$\begin{gathered}J_j^U(k+\ell)=\theta_v\sum_{i=1}^{w_j(k+\ell)}\left(f_v(k+\ell)\left(1-z_j^i(k+\ell)\right)\left(\underbrace{\hat{T}_j^i(k+\ell)-tr_{r_j^i(k+\ell)}}_{\text{re-routing time}}\right)\right)\Bigg|_h+\\ \theta_e\sum_{i=1}^{w_j(k+\ell)}\left(f_e(k+\ell)z_j^i(k+\ell)\left(\underbrace{\hat{T}_j^i(k+\ell)-t_{0r_j^i(k+\ell)}}_{\text{waiting time}}\right)\right)\Bigg|_h\end{gathered}\tag{5.10}$$

In (5.8), J_j^U and J_j^O denote the user and operator costs respectively, associated with the sequence of stops that vehicle *j* must follow at certain instant. In chapter 4 a zoning based method for trip patterns estimation based on Fuzzy Clustering was designed. Expressions (5.9) and (5.10) represent the operator and users cost functions related to vehicle *j* at instant $k+\ell$, which depend on the previous sequence $S_j(k+\ell-2)$ and a new potential request *h* which occurs with probability $p_h(k+\ell)$; $w_j(k+\ell)$ is the number of stops estimated for vehicle *j* at instant $k+\ell$. To explain the flexibility of the formulation and its economic consistency, the term related with the extra time experienced by passengers in this service (delivery time minus the minimum time the user could arrive to its destination) is weighted by a factor θ_v, and the term related to total waiting time of each passenger is weighted by θ_e. Note that the terms in the objective functions for user are weighted by the functions $f_v(k+\ell)$ and $f_e(k+\ell)$, which include a service policy for users, so the cost of a user that entered the system a long time ago is considered more importantly than another user who has just made the request. In this work, we propose the following weighing functions:

$$f_v\left(k+\ell\right)= \begin{cases} 1 & \text{if} \quad \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)} < \alpha\left(tr_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) \\ 1+\hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}-\alpha\left(tr_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) & \text{if} \quad \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)} < \alpha\left(tr_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) \end{cases} \tag{5.11}$$

Expression (5.11) implies that if the delivery time $\hat{T}_j^i\left(k+\ell\right)$ associated with user $r_j^i\left(k+\ell\right)$ becomes greater than α times its minimum total time $\left(tr_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right)$, the weighting function $f_v\left(k+\ell\right)$ grows linearly, resulting in a critical service for such a client. Regarding the waiting time factor,

$$f_e\left(k+\ell\right)= \begin{cases} 1 & \text{if} \quad \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)} \leq TT \\ 1+\hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}-TT & \text{if} \quad \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)} > TT \end{cases} \tag{5.12}$$

The intuition behind (5.11) is analogous to (5.12). In addition, we will suppose an expression for the vehicle operational cost (equation 5.9), with a component depending on the total traveled distance, weighted by a factor c_L, and another on the total operational time, in this case at unitary cost c_T. Thus, $D_j^i\left(k+\ell\right)$ represents the distance between stops $i-1$ and i in the sequence of vehicle j.

5.3.2.2. MO-HPC for a Dial-a-Ride System.

The MO-HPC strategy is a generalization of HPC where the optimal control action is selected based on a criterion that takes solutions from the optimal Pareto region considering the following multi-objective problem:

$$
\begin{gathered}
\underset{S_k^{k+N}}{Min}\ \{J_1, J_2\} \\
J_1 = \sum_{\ell=1}^{N} \sum_{j=1}^{F} \sum_{h=1}^{h_{\max}(k+t)} p_h(k+\ell)\left(J_j^U(k+\ell) - J_j^U(k+\ell-1)\right) \\
J_2 = \sum_{\ell=1}^{N} \sum_{j=1}^{F} \sum_{h=1}^{h_{\max}(k+t)} p_h(k+\ell)\left(J_j^O(k+\ell) - J_j^O(k+\ell-1)\right)
\end{gathered}
\tag{5.13}
$$

A relevant step of this approach in the controller's dispatch decision is the definition of criteria to select the best control action at each instant under the MO-HPC approach. In this case, and given that the level of service is better considered, different criteria regarding a minimum allowable level of service can be dynamically used to take policy dependent routing decisions, based on the Pareto front obtained from the MO scheme. In this work, three criteria for level of service are evaluated:

Criterion 1: Minimum users' cost component.
Criterion 2: Minimum operational cost component.
Criterion 3: The nearest value to a given user cost (travelled time plus waiting time costs).

In those cases where the policy for users is accomplished for several solutions, the one that is the closest to the pursued policy will be selected. So, soft-constraints are included directly, without incorporating them into the optimization problem, although they are considered in the choice process that emulates the dispatcher. Results and analysis of these operation policies from simulations are reported in Section 5.4.3.

5.3.2.3. Simulation Results.

In this section the simulation tests conducted to show the MO-HPC strategy application are presented. A period of two hours representative of a working day (14:00-14:59, 15:00-15:59) is simulated, over an urban area of approximately 81 km^2. A fixed fleet of fifteen vehicles is considered, with a capacity of four passengers each. Assume that the vehicles travel in a straight line between stops and that the transport network behaves according to a speed distribution with mean equal to 20[km/h].

The future calls are assumed unknown for the controller. However, he(she) has historical data from where the typical trip patterns can be extracted. A speed distribution model and the trip patterns are known, from the historical data and the fuzzy zoning method proposed in chapter 4. This fuzzy zoning permits to generate the trip patterns and their probabilities as shown in Figure 5.8 and Table 5.4.

Two hundred and fifty calls were generated over the simulation period of two hours following the spatial and temporal distribution observed from the historical data. Regarding the temporal dimension, a negative exponential distribution for time intervals between calls with rate 0.5 [call/minute] for both hours of simulation was assumed. Regarding the spatial distribution, the pick-up and delivery coordinates were generated randomly within each zone. The 15 first calls at the beginning and the 15 last calls at the end of the experiments were deleted from the statistics to avoid limit distortion (warm up period). 10 replications of each experiment were carried out to obtain the global statistics. Each replication (emulating two hours and 250 on-line decisions) took 20 minutes in average, in a Intel® Core™2 CPU, 2.40Ghz processor.

Table 5.4. Pickup and delivery coordinates and probabilities: Fuzzy zoning.

X pickup	Y pickup	X delivery	Y delivery	Probability
4.007	4.1847	5.6716	4.5576	0.119
3.9312	4.0303	6.4762	6.1463	0.1726
5.4013	4.0548	6.5659	5.9723	0.3512
6.4578	5.9338	3.9844	5.9785	0.3571

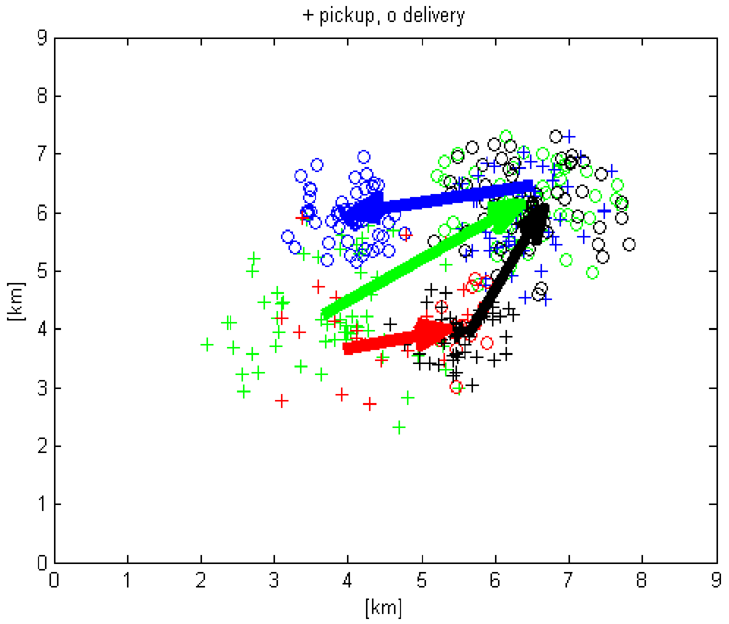


Figure 5.8. Origin-destination trip patterns.

The objective function is formulated at two steps ahead, considering parameters θ_v =16,7[\$/min], θ_e =50[\$/min], c_T=25[\$/min], c_L=350[\$/Km], α = 1.5, TT = 5 [min]. First, the results of the HPC approach with a mono-objective function at two steps ahead are reported, computed for weights λ= *1, 0.75, 0.5, 0.25* and *0*, in order to verify that the objectives pursued by users and operator are effectively opposite. Table 5.5 shows effective travel and waiting average times per user, as well as user cost. Table 5.6 shows effective total travel time, distance travelled per vehicle and total operator cost, all on average.

Table 5.5: HPC with different weighting factors. User indexes.

Weight factor λ	Travel time [min/pax]		Waiting time [min/pax]		Mean user cost [\$]
	Mean	Std	Mean	Std	
λ = *1*	9.36	3.66	4.52	2.74	382.27
λ = *0.75*	9.79	4.25	4.47	2.49	386.89
λ = *0.50*	10.19	4.49	4.60	2.99	399.88
λ = *0.25*	10.48	4.75	5.38	3.06	444.12
λ = *0*	10.01	7.38	15.44	10.80	939.15

Table 5.6: HPC with different weighting factors. Operator indexes.

Weight factor λ	Time Travelled [min/veh]		Distance Travelled [km/veh]		Mean operator cost [$]
	Mean	Std	Mean	Std	
$\lambda = 1$	88.16	7.55	24.84	1.86	10898.28
$\lambda = 0.75$	75.17	11.06	20.61	2.94	9094.22
$\lambda = 0.50$	67.57	12.78	18.62	3.51	8207.24
$\lambda = 0.25$	61.67	12.57	16.95	3.17	7476.06
$\lambda = 0$	43.90	17.94	12.58	5.09	5500.82

Tables 5.7 and 5.8 clearly show a clear trade off between opposite components. Besides, small values for standard deviation imply that the travel and waiting times are more balanced among passengers as a variable weight for them was included in the objective function specification. Notice the extreme case benefiting the operator results in a very poor service for users, not only around the mean but also in terms of bounding the standard deviation.

Additional simulations for two steps ahead were conducted to analyze and evaluate the performance of the MO-HPC strategies. The results are reported in Table 5.7, showing the effective user waiting and travel time. In table 5.8 we also show average travel time and distance associated with vehicles. The results are reported associated with the previously described criteria (Section 5.3.4) for selecting the current policy at each event during the simulation. In case of criterion 3 (the nearest value to a given user cost), three references are considered: 400, 500 and 600 [$] for sub-cases a), b) and c) respectively.

Note from Table 5.7 and 5.8, the first two criteria are equivalent to cases of λ equals to *0* and *1* from Tables 5.5 and 5.6. With regard to criterion 3, the resulting mean user cost over the whole simulation fitted quite well the thresholds defined at each sub-case. Note also the small standard deviation of users cost, as a result of the service policy included by means of the special weighting functions in the new objective function formulation.

Table 5.7: MO-HPC with different weighting factors. User indexes.

MO	Travel time [min/pax]		Waiting time [min/pax]		Mean user cost [$]
	Mean	Std	Mean	Std	
Criterion 1	9.36	3.66	4.52	2.74	382.27
Criterion 2	10.01	7.38	15.44	10.80	939.15
Criterion 3a	10.32	4.75	4.62	2.67	403.17
Criterion 3b	10.76	5.36	5.63	3.58	461.30
Criterion 3c	10.63	6.09	7.25	4.59	540.20

Table 5.8: HPC with different weighting factors. Operator indexes.

MO	Time Travelled [min/veh]		Distance Travelled [km/veh]		Mean operator cost [$]
	Mean	Std	Mean	Std	
Criterion 1	88.16	7.55	24.84	1.86	10898.28
Criterion 2	43.90	17.94	12.58	5.09	5500.82
Criterion 3a	74.99	8.76	20.91	2.19	9193.45
Criterion 3b	69.56	11.52	19.92	3.05	8713.60
Criterion 3c	71.40	10.53	20.39	2.80	8924.53

5.4. Discussion.

This chapter presents a new approach to solve the dial-a-ride problem under a hybrid predictive control scheme using dynamic multi-objective optimization. Three different criteria are proposed to obtain control actions over real-time routing using the dynamic Pareto front. The criteria allow giving priority to a service policy for users, ensuring a minimization of operational costs under each proposed policy.

The service policies are verified approximately on the average of the replications. Under the implemented *on-line* system it is easier and transparent for the operator to follow service policies under multi-objective approach instead of tuning weighting parameters dynamically.

The multi-objective approach allows obtaining solutions that are directly interpreted as part of the Pareto front instead of results obtained with mono-objective functions, which lack of direct physical interpretation (the weight factors are tuned but they do not allow applying operational or service policies such as those proposed here). Thus, more generic solutions are searched.

5.5. References

Berbeglia, G., Cordeau, J., Laporte, G., (2009). “Dynamic pickup and delivery problems”. European Journal of Operational Research. doi:10.1016/j.ejor.2009.04.024.

Farina N., Deb, K., Amato, P., (2004). “Dynamic Multi-objective Optimization Problems: Test Cases, Approximations, and Applications”. IEEE Transactions on Evolutionary Computation 8, No 5, 425-439.

Jozefowiez, N., Semet, F., Talbi, E., (2008). “Multi-objective vehicle routing problems”. European Journal of Operational Research 189, 293-309.

Tan, K.C, Cheong, C.Y., Goh, C.K., (2007). “Solving multi-objective vehicle routing problem with demand via evolutionary computation”. European Journal of operational research 177, 813-839.

Yang, W.H., Mathur, K., Ballou, R.H., (2000). “Stochastic vehicle routing problem with restocking”. Transportation Science 34, 99-112.

6. Hybrid Predictive Control for an Integrated Public Transport System.

6.1. Literature review.

The provision of efficient public transport systems in urban areas is crucial for the better development of a city and for improving the quality of life of public transport users that use the system to develop their regular activities every day. In addition, it is reasonable to offer a more personalized scheme for a subset of users willing to pay more in order to have a point to point transport service, which in essence should belong to the whole public transport system of the studied system. There are other reasons for promoting such integrated systems: in low demand density settings, it is very expensive to run fixed routes. With the development of technology, one should be able to add the dynamic dimension in the design of an efficient operational scheme, obtaining a more flexible integrated public transport system, able to serve not only in-advance but also real-time requests. An interesting way to handle the request of these passengers is by coordinating the operation of traditional fixed-route public transport schemes with a more personalized sub-system working in a rerouting setting, operated by small vehicles of dial-a-ride type. Under real situations, the optimization of such systems can become is extremely complex and more sophisticated procedures are needed. Then, this chapter describes the design of a predictive control strategy for an integrated public transport system (IPTS) of this type, considering operational and service policies, as well as costs reduction.

This chapter represents a first step in the design of coordination between dial-a-ride and other transportation system. The incorporation of transfer points in bus stops is a huge problem and more efficient optimization solvers are required. The inclusion of other transportation systems like taxis, subway, train, etc., is part of the further research.

In the literature, there are many references regarding dynamic routing problems and public transport systems, but limited works trying to integrate door-to-door systems with fixed route lines. These integrated systems are usually called mixed service systems. The idea of combining trunk corridors (fixed route system) with feeder services (reroutable scheme) is in adding flexibility to the system together with reducing the demand pressure for the door-to-door service.

Malucelli *et al*. (1999) and Crainic *et al*. (2001) describe a new flexible collective transportation system. Their system considers conventional fixed route lines combined with lines based on flexible itinerary and timetable. Hickman and Blume (2000) formulate the problem of scheduling both passenger trips and vehicle trips for a proposed integrated service. The service works in three stages: the demand responsive service connects passengers from their origin to the fixed route service and (or) from the fixed route service to their final destination. A schedule for both passenger and vehicle trips is created in order to minimize a measure of the cost of service. They use a modified insertion version of the heuristics by Jaw *et al*. (1986) in order to schedule integrated transit trips that accommodates both passenger and vehicle scheduling objectives. One strong assumption made by Hickman and Blume (2000) when specifying the generalized time or disutility function used in their algorithm, is to add a fixed transfer penalty, independent of the number of transfers realized. The number of transfer is also part of the cost function, but it only has a linear influence on the general expression.

Liaw *et al*. (1996) define the integrated mode as a bimodal dial-a-ride problem (BDARP), including paratransit vehicles as well as fixed route buses. They design a decision support system (DSS), which automatically constructs efficient paratransit vehicle routes and schedules for the BDARP. The insertion heuristics was tested on a data set from Ann Arbor, Michigan. Hickman and Blume (2000) illustrate their method using a case study of transit service in Houston, Texas, showing the advantages in cost as well as the impact on passenger level of service from implementing integrated transit service.

Horn (2002a, 2002b, 2003, 2004) propose and describe the main analytical and procedural components of a modelling system which provides a framework for investigating the performance of urban passenger transport systems with particular attention to demand-responsive transport modes and traveller information technologies (demand coming up in real-time). The modes covered include conventional timetabled services (buses, train, etc.), taxis (both single and multiple hire) and other demand responsive services. Individual requests are resolved as single or multiple leg journeys through the use of request broking and journey-planning modules that seek to minimise travellers' generalised costs. Both modules are designed as embedded control systems and are intended for use in real-time as well as modelling applications. The decisions are made based on an interesting heuristic insertion procedure (time-windowed incremental insertion); however, there is no prediction power in this decision.

Cortés (2003), Cortés and Jayakrishnan (2002) proposes the development and evaluation of a new concept for high-coverage point to point transit systems (HCPPT). The proposed scheme design consists on a set of vehicles that perform both door to door service and fixed route service, so the waiting times in transfer nodes are minimized for all passenger picked-up for a vehicle that travel the fixed route. Sophisticated dynamic real-time routing rules were implemented in a multi-purpose simulation platform that showed that with enough deployed vehicles, the system can be substantially better, even competitive with personal auto travel, and compared to the existing fixed route public transit. HCPPT can be incrementally implemented by contracting out services to existing private operators. The strict optimization formulation and solution considers accounts for future dispatch decisions and can thus be interpreted as form of quasi-optimal predictive adaptive control problem.

So far, the HPC framework to represent the real-time fixed-route public transport control for a system of buses (applying strategies such as holding, station skipping, signal priority, etc.) and the HPC formulation for door-to-door services, described in chapter 4, have been analyzed separately. In Cortés et al. (2009), Sáez *et al.* (2009) a HPC framework was successfully applied for the control of a one corridor fixed-route bus system.

The integrated system proposed here is described using the same HPC formulation for the bus system, which allows the prediction of the headways obtained from the fixed-route controller as inputs of the controller for the dial-a-ride system. Then, the proposed HPC for a dial-a-ride system, explained before in chapters 4 and 5, is adapted for including the demand that use both systems.

Regarding the optimization procedures, in real urban situations and with similar hardware, more sophisticated procedures such as hybrid predictive control (HPC) proposed in chapter 3 would entail much longer execution times than simpler heuristic algorithm. To cope with this problem, as the HPC solves an NP-Hard non-linear mixed integer optimization problem whenever a new decision needs to be made, computational intelligence methodologies of the type described in chapter 3 are proposed to simplify the computation load and, at the same time, to maintain the good quality of solutions. The details in the implementation of such methods is part of further research.

Thus, in this chapter the integrated dial-a-ride problem of a fleet of vehicles together with a public transport system is formulated using a hierarchical hybrid predictive control (H-HPC) approach.

6.2. Integrated Public Transport System.

The major idea in this chapter is to combine a regular fixed-route public transport trunk service (using large buses in the operation) with a typical dynamic dial-a-ride service (served with small vehicles, such as vans) described in chapter 4. The major objective of this development is to write an integrated hierarchical HPC formulation for both systems, where the relations between systems occur at transfer points corresponding to the stops of the bus corridor. The better the coordination and synchronization of transfer operations, the better the performance of the whole system. Considering that a regular passenger will have several options to travel from origin to destination, depending on the location of such points (close or far from a trunk bus route), and on the passenger willingness to pay higher fares for a more personalized service as well.

Then, a proper integrated design should be able to fulfil point-to-point travel requests for five types of travel options represented in Figure 6.1 and Figure 6.2. Option 1 is to travel door-to-door directly on a re-routable small vehicle (say a van). Option 2 shows a combination of a re-routable collector service finishing at a trunk bus stop. Option 3 shows a typical distribution system, in which passengers are picked up at a bus stop and taken to their final destination. Option 4 is a door-to-door option by using both systems, first the small re-routable vehicle, then the bus and finally another small vehicle. Notice that option 4 is not as attractive as the other options as it requires transferring twice. The option 5 is simply the use of the trunk corridor for the passenger trip, if that passenger has both pick-up and delivery locations close to the bus corridor stops.

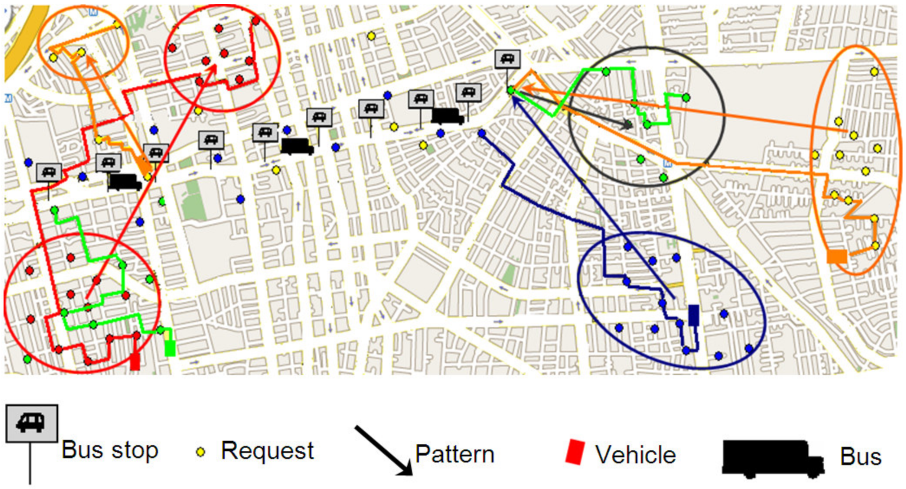


Figure 6.1. Integrated Public Transport System.

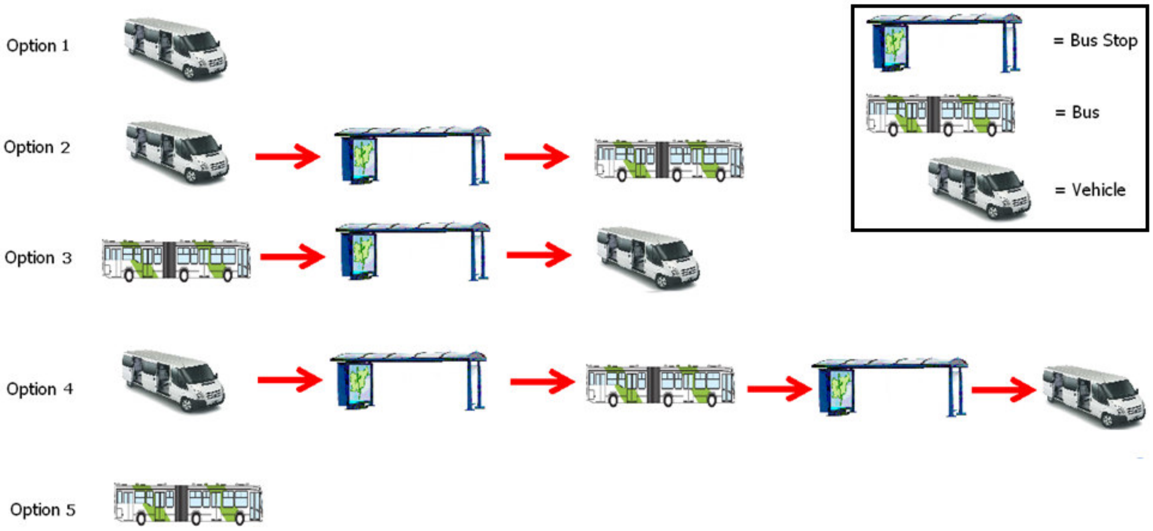


Figure 6.2. Five types of journey modes.

Next, the characteristics of the demand are described for the five types of journey modes shown in Figure 6.2.

The first type of demand uses just the dial-a-ride system (option O_k =1), similarly to the description in chapter 4. The demand is characterized by two positions, pickup and delivery p_k, d_k, by the instant of the call occurrence t_{0k}. The expected minimum arrival time tr_k corresponds to the best possible service for that passenger, considering no re-routing of his(her) trip (shortest path) and a waiting time from the call instant associated with the closest available vehicle (in terms of capacity) to pick that passenger up. r_k is the label that identifies the passenger who is making the call, and finally Ω_k denotes the number of passengers waiting there (size of the request). The service demand η_k comprises the information of the request, namely $\eta_k = \left[O_k \quad p_k \quad d_k \quad t_{0k} \quad tr_k \quad r_k \quad \Omega_k \right]$. The first term equals 1 and indicates the passenger-journey is of the type 1.

Figure 6.3 shows an example of demand of type 1. For the calculation of tr_k, even though vehicle 2 is closer to the pickup point, it is not available, so the best vehicle to serve the new demand is vehicle 1. In the figure, the arrows represent the best service path satisfying the request. St_i are the bus stops, not used as transfer points in this case.

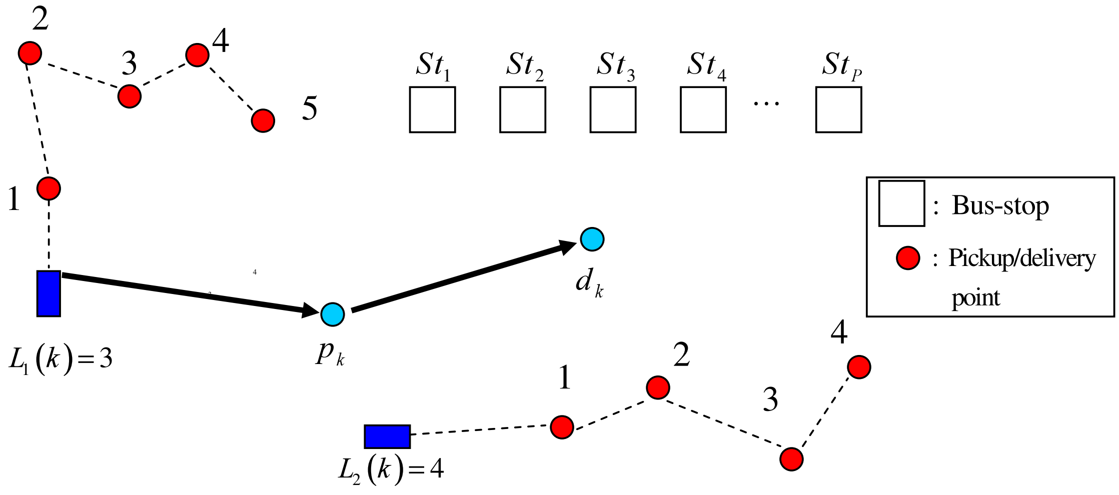


Figure 6.3. Type of journey 1.

The second type of demand is a combination of a re-routable collector service finishing at a trunk bus stop (option O_k =2). The demand is characterized by one coordinate of pick-up p_k, one destination stop st_k^d and by the instant of the call occurrence t_{0k}. The expected minimum arrival time tr_k corresponds to the best possible service for that passenger, considering a waiting time from the call instant associated with the closest available vehicle (in terms of capacity) to pick that passenger up, no re-routing of his(her) trip (shortest path) to the best bus-stop so the waiting time in that stop and total travel time is the minimum. r_k is the label that identifies the passenger who is making the call, and finally Ω_k denotes the number of passengers waiting there (size of the request). The service demand η_k comprises the information of the request, namely $\eta_k = \left[O_k \quad p_k \quad st_k^d \quad t_{0k} \quad tr_k \quad r_k \quad \Omega_k \right]$. The first term O_k =2 means the passenger journey is of type 2. This kind of user could use any transfer point belonging to the transit system (at any bus-stop) and it can be changed dynamically if the re-routing positively affects the system performance. This variable transfer point could turn the optimization problem hard to be solved in real-time, so it will be assumed that among the possible transfer points, only the closest stop to the pickup coordinate plus the consecutives two bus-stops are considered as real transfer options. Figure 6.4 shows an example of demand type 2. For the calculation of tr_k, although vehicle 2 is closer to the pickup point, it is not available, so the best vehicle to serve the new demand is vehicle 1. Then, vehicle 1 have three possibilities for transferring: bus-stop St_1, bus-stop St_2 and bus-stop St_3. In the figure, the arrows show the best path for the vehicle to serve in the best way the request with bus-stop St_2 as the selected transfer point. Even though vehicle 1 and bus-stop St_2 are chosen to compute tr_k, in the final solution, a different vehicle and a different transfer point could be used. In fact, once the demand is assigned to a vehicle, the transfer point can change if the impact of that decision affect positively the system performance. Therefore, the transfer point is variable. In the objective function formulation defined later, tr_k is decoupled in two times, first the time tr_k^1 to reach the transfer point (waiting time plus travel time), and the time $\hat{T}_{st}$ to reach the destination from the transfer point (waiting time at bus station plus travel time). Note that if the transfer point is set to be St_3, a shortest travel time in the bus system is expected; however, the waiting time at the stop depends on the headways of the buses.

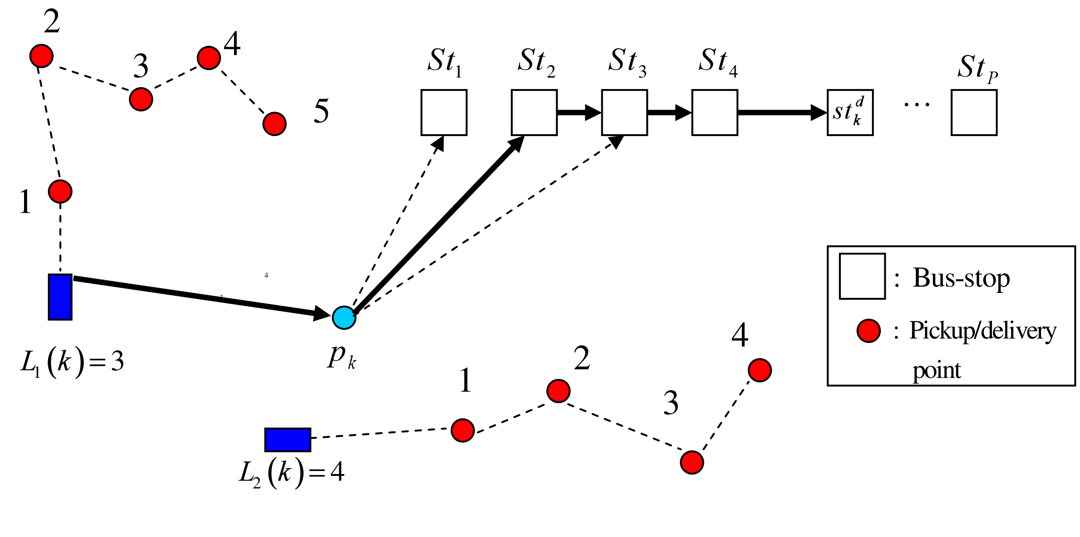


Figure 6.4. Type of journey 2.

The third type of demand is a combination of trunk bus-stop finishing at a re-routable collector service (option O_k=3). The demand is characterized by one bus-stop of origin st_k^p, a delivery coordinate d_k , and by the instant of the call occurrence t_{0k} . The expected minimum arrival time tr_k corresponds to the best possible service for that passenger, considering a waiting time from the call instant in the bus-stop st_k^p, the best travel time in-bus (the best bus-stop used as transfer point) so the minimum waiting and travel time is obtained with the closest available vehicle in terms of capacity, once the user arrives to the transfer point, with no re-routing of his(her) trip (shortest path) to the destination. r_k is the label that identifies the passenger who is making the call, and finally Ω_k denotes the number of passengers waiting there (size of the request). The service demand η_k comprises the information of the request, namely $\eta_k = \left[O_k \quad st_k^p \quad d_k \quad t_{0k} \quad tr_k \quad r_k \quad \Omega_k \right]$. The first term equals 3 and means that the passenger journey is of the type 3. This kind of user could use any transfer point belonging to the transit system (at any bus-stop where the user will wait for being picked up for a vehicle). However, this transfer point cannot be changed dynamically, even in cases where the re-routing would positively affects the system performance. This is for practical reasons, since once the user calls the dispatcher must indicate the transfer bus-stop, which remains as a definite decision. Figure 6.5 shows an example of demand type 3.

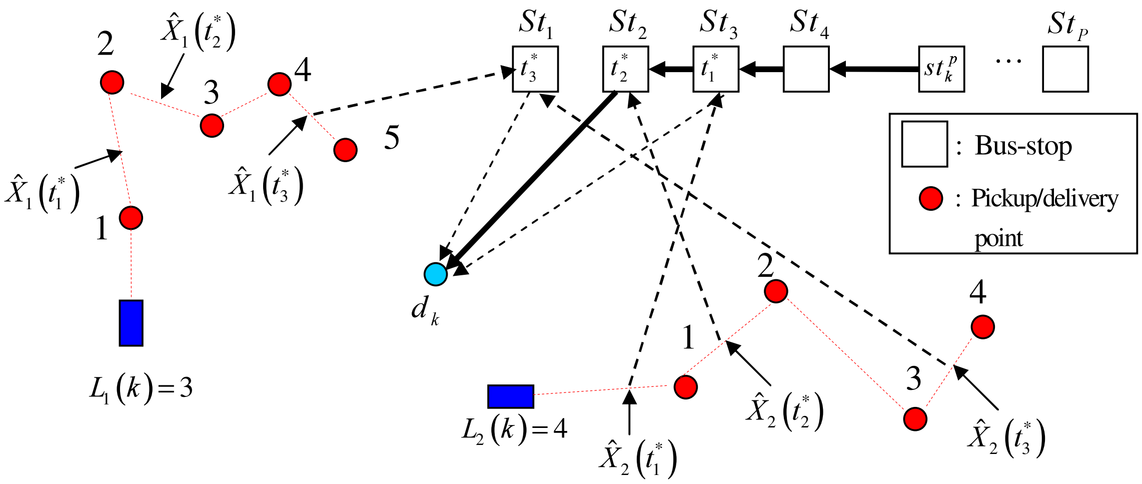


Figure 6.5. Type of journey 3.

In Figure 6.5 for the calculation of time tr_k, first the arrival time to the closest bus-stops to the destination d_k are obtained. In the figure those times are t_1^* to the bus-stop *St₃*, t_2^* to the bus-stop *St₂* and t_3^* to the bus-stop *St₁*. Then, the position of the vehicles are estimated in each arrive time, and the vehicle which is closest and available, that produces the minimum waiting time in the transfer point and minimal travel time is chosen for obtaining tr_k. In Figure 6.5, the transfer point is *St₂* and vehicle 2 is the best for the pickup and delivery of the user. A difference with users of type 2 is that, once the user is inserted, the transfer point will not change (for communication and practical reasons). In the figure, the arrows denote the best path for the user to be served. Even though vehicle 2 and bus-stop *St₂* are used for calculating tr_k, the vehicle used for serving the demand and the transfer point may be different. In the objective function formulation, tr_k is decoupled in two times, first the time to reach the transfer point (including waiting and travel time on bus), and the time to reach the destination from the transfer point tr_k^1 (waiting time in bus station plus travel time). Note that if the transfer point is set to be *St₁*, which seems to be the closest bus-stop to the destination, a longer travel time in the bus system is expected.

The fourth type of demand is a combination of the dial-a-ride service, then trunk bus-stop, finishing at a re-routable collector service (option O_k =4). The demand is characterized by two positions, pickup and delivery p_k, d_k, by the instant of the call occurrence t_{0k}. The expected minimum arrival time tr_k corresponds to the best possible service for that passenger, considering a waiting time from the call instant associated with the closest available vehicle (in terms of capacity) to pick that passenger up, no re-routing of his(her) trip (shortest path) to the best bus-stop so the waiting time in that stop and total travel time is minimum, considering also the best travel time in-bus (the best bus-stop used as transfer point) so the minimum waiting and travel time is obtained with the closest available vehicle in terms of capacity, once the user arrives to the transfer point, with no re-routing of his(her) trip (shortest path) to the destination. r_k is the label that identifies the passenger who is making the call, and finally Ω_k denotes the number of passengers waiting there (size of the request). The service demand η_k comprises the information of the request, namely $\eta_k = \left[O_k \quad p_k \quad d_k \quad t_{0k} \quad tr_k \quad r_k \quad \Omega_k\right]$. The first term equals 4 and means the passenger journey is of the type 4. This kind of user requires two transfer bus-stops. The first transfer could be changed dynamically if the re-routing positively affects to the system. The second transfer bus-stop cannot be changed dynamically, even if the re-routing could positively affect the system.

The last type of demand uses just the transit system (option O_k =5). The demand is characterized by two bus-stops, the origin bus-stop st_k^p and the destination bus-stop st_k^d, by the instant of arrival to the pickup bus-stop occurrence t_{0k}. The expected minimum arrival time tr_k corresponds to the best possible service for that passenger, where waiting and travel time is minimum. r_k is the label that identifies the passenger, and finally Ω_k denotes the number of passengers. The service demand η_k comprises the information of the request, namely $\eta_k = \left[O_k \quad st_k^p \quad st_k^d \quad t_{0k} \quad tr_k \quad r_k \quad \Omega_k \right]$. The first term equals 5 and indicates the passenger-journey is of the type 5.

The modelling approach is in discrete time, where the steps are activated every time a relevant event occurs, that it is when a call asking for dial-a-ride service is received or whenever a bus arrive to a bus-stop (in more sophisticated schemes it could also be the time when the dispatcher decide to change a route due to congestion, accident, etc). k represents the k^{th} instant in the discrete events sequence. The preferred journey mode will be asked/suggest to the user, so this information will be provided to the HPC. In this chapter in-advance requests and time windows are not considered. The demand is inelastic and the value of users´ time will be assumed fixed (no preferences for the users), although annoyed user for long re-routing will be considered through the weighting factors in the objective function. In addition, the dial-a-ride vehicle will not be able to wait for a user at any transfer point if the vehicle has at least one other user on board.

The proposed operational scheme is designed in order to minimize the total operational costs and to optimize the level of service of users, the latter by means of the minimization of travel and waiting times as well as number of transfers. The entire optimization scheme relies on the availability of computer and communication technology in order to allow real-time optimization and coordination/synchronization between subsystems. Fixed route services in transit without near-the-door pickup and delivery are not very attractive to certain users with poor accessibility to the bus route from their origin or destination; however, fixed route services are recommended in cases of some very high-density demand corridors. That is the major reason to propose more flexible alternatives to the user, taking advantages of fixed route (with high capacity vehicles) services on high-demand corridors, in combination with local dial-a-ride systems for low demand density portions of the trip. This type of scheme could become attractive to people who presently prefer the automobile to traditional transit systems for their regular trips but are not willing to pay the fare of a taxi for accomplish it. The closed-loop diagram of the integrated public transport system is shown in Figure 6.6. IPTS stands for Integrated Public Transport System, PTS for Public Transit System and DARS for Dial-a-ride system.

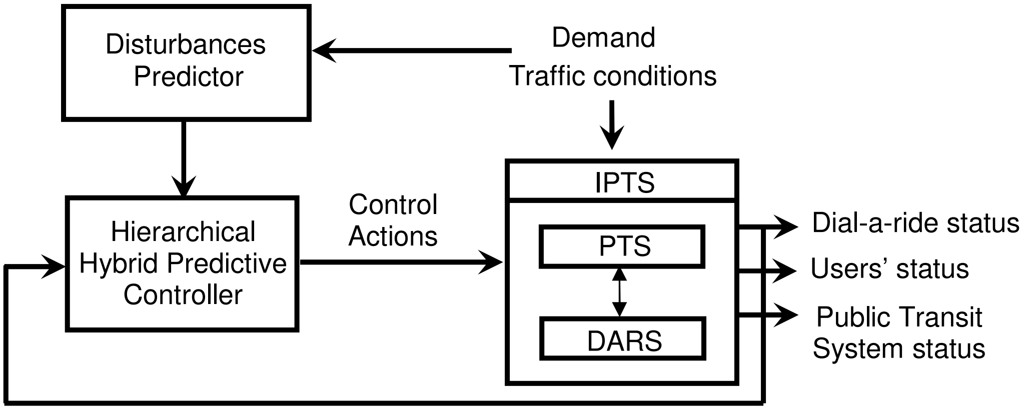


Figure 6.6. Closed-loop, IPTS.

A challenge in this chapter is to design a way to integrate both systems, requiring special attention on the proper way to define the interface between them. A first scheme is to control both systems (buses and re-routable vehicles) with different hybrid predictive controllers (see Figure 6.7). The control actions decided for one system will be considered as disturbances for the other system.

The scheme proposed in this chapter is to control both systems within a hierarchical framework, in which the hybrid predictive controller for the dial-a-ride system will include information of both systems (buses and re-routable vehicles). The advantage in this scheme is that a better service could be reached for users transfering between both systems; however, local controllers are safer in cases where the global controller failed.

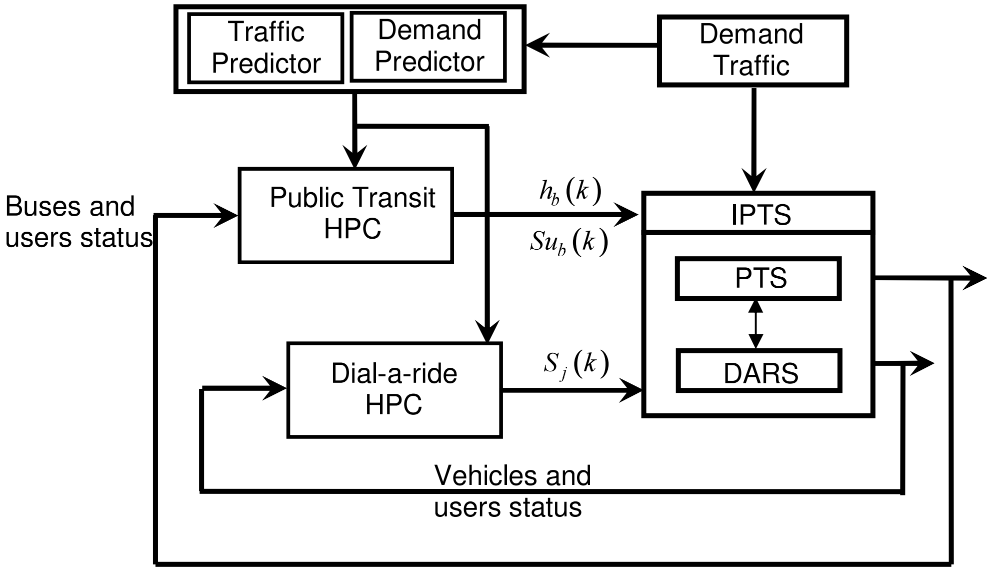


Figure 6.7.- Independent controllers.

6.3. Hierarchical Hybrid Predictive Control (H-HPC) for the Integrated Transit System.

The Hierarchical HPC is described in Figure 6.8. It receives information regarding the status of the dial-a-ride system (DARS) and the public transit system (PTS), as also the status of each user (effective travel times, waiting times, etc). For doing the predictions, Hierarchical HPC also uses an estimator of the disturbance input (demand and traffic conditions), so the controller can look forward and make decisions (control actions) using the predictions of the whole system.

The H-HPC has two levels. The former is the control algorithm for the public transit system (PTS) exclusively. This level will keep the headways of the buses as regular as possible and the effect of the dial-a-ride system will not be considered since we assume a high demand of passengers who only use the transit system (users with journey Option 5). From this level, control actions like station-skipping, holding, etc, will be made based on the predictions of some variables like headways, buses positions, transference times, etc. The information required in this level is the status of users and buses, and demand and traffic conditions predictions.

The control algorithm of the dial-a-ride system (DARS) is established at a second level. This level selects the best vehicle to serve each request, using information that comes from the first level (headways) whenever a user requires the public transit system. Also, based on some heuristic, this level chooses the candidate vehicles to serve the new request avoiding the evaluation of the insertion cost of the whole fleet (obtaining the corresponding computation time savings). At the second level, some specific vehicles evaluate the insertion cost of different scenarios, considering the current request and the demand pattern. Each vehicle sends the costs to the second level, which chooses the best vehicle that provides the minimum cost for user and operator. The information required at this level comes from the first level (predicted headways of buses), the demand and congestion predictor (trip patterns, velocity), and from the incremental cost of each vehicle. The output of this level is the assignment of a request into a vehicle, in order to obtain a minimum incremental cost of the system.

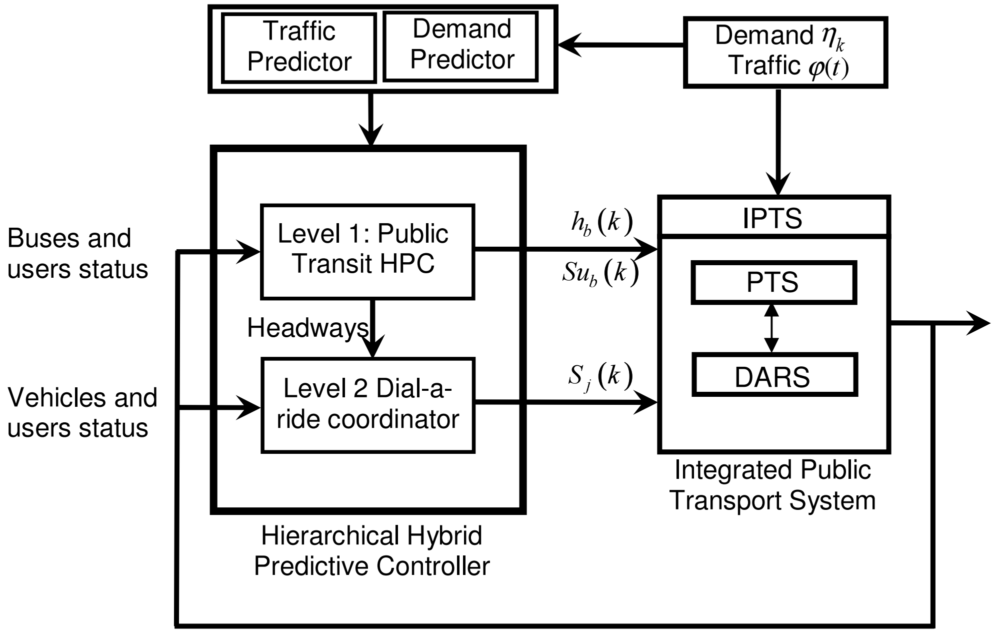

Traffic Predictor
Demand Predictor
Demand η_k
Traffic $\varphi(t)$
Buses and users status
Vehicles and users status
Level 1: Public Transit HPC
Headways
Level 2 Dial-a-ride coordinator
$h_b(k)$
$Su_b(k)$
$S_j(k)$
IPTS
PTS
DARS
Integrated Public Transport System
Hierarchical Hybrid Predictive Controller

Figure 6.8. H-HPC details.

In the Integrated Public Transport System (IPTS), the number of buses and vehicles, their capacity and the number of bus-stops are fixed. In the closed-loop diagram, the main variables of the Public Transit System (PTS) are the buses and users status. The state space variables for each bus b are its load $L_b(k)$, departure time to a stop $Td_b(k)$ and position $x_b(t)$. In the fixed stops, the state space variables are number of passengers waiting for a bus $\Gamma_p(k)$ and the headway $H_p(k)$ at the stop *p*. The manipulated variables are the holding $h_b(k)$ and the station skipping $Su_b(k)$ actions associated with bus *b* at instant *k*. In the Dial-a-ride system (DARS), the main variables, described in detail in chapter 4, are the vehicles and users status. The state space variables of each vehicle *j*, as in chapter 4, are position $X_j(k)$, departure time vector $T_j(k)\in R^{w_j(k)+1}$ and vehicle load vector $L_j(k)\in R^{w_j(k)+1}$. The manipulated variables are the sequences $S_j(k)$ for each vehicle. The user status is given by a measurement of the effective waiting and travel times of each user, considering separately the effective waiting time at the pickup as well as at the transfer points. The demand and the traffic conditions are disturbances (stochasticity). Moreover, the objective function is influenced by the prediction of the uncertain demand and traffic conditions. Next the different objectives functions are defined for each level of the proposed Hierarchical Hybrid Predictive Controller (H-HPC).

6.3.1. Level 1, Hybrid Predictive Control for the Public Transit system.

Whenever a bus arrives to a bus stop, the following objective function is optimized at level 1 by the HPC in order to make the real-time decisions and optimize the dynamic system, as in Sáez *et al*. (2009) and Cortés *et al.* (2009):

$$\begin{aligned}\min_{\{u(k),u(k+1),\ldots,u(k+Np-1)\}} \sum_{\ell=1}^{Np} \Big[& \theta_1 \cdot \hat{H}_b(k+\ell)\hat{\Gamma}_p(k+\ell) + \theta_2 \cdot (\hat{H}_b(k+\ell)-\bar{H})^2 + \\ & \theta_3 \cdot \hat{L}_b(k+\ell)\, h_b(k+\ell-1) + \theta_4 \cdot \hat{L}_b(k+\ell)\,\hat{T}r_b(k+\ell-1) + \\ & \theta_5 \cdot \hat{\Gamma}_p(k+\ell)\hat{H}_{b+1}(k+\ell+z_{b+1})\big(1-Su_b(k+\ell-1)\big)\Big]\Big|_{\substack{b=b(k+\ell-1)\\ p=p(k+\ell-1)}}\end{aligned} \tag{6.1}$$

Objective function (6.1) comprises five components, all of them definitely oriented to user cost through total in-vehicle ride and waiting time. $\{u(k),\ldots,u(k+Np-1)\}$ is the control-action sequence with $u(k+\ell-1)=\begin{bmatrix} h_b(k+\ell-1) \\ Su_b(k+\ell-1)\end{bmatrix}$ when bus *b* triggers event $k+\ell-1$. *Np* is the prediction horizon and *B* is the number of buses in the fleet. Note that $b=b(k+\ell-1)\in\{1,\ldots,B\}$, $p=p(k+\ell-1)\in\{1,\ldots,P\}$, if we consider that the future event $k+\ell-1$ is triggered by one bus

$b(k+\ell-1)$ arriving to a specific station downstream $p(k+\ell-1)$. In expressions (6.1), $\theta_j, j=1,...,5$, are weighting parameters, $\overline{H}$ corresponds to the desired headway (set-point). The first term in (6.1) quantifies the total passenger waiting time at stops and depends on the predicted headway along with the bus stop load. The second term captures the regularization of bus headways, to maintain the headway as close as possible to the design headway. The third component measures the delay associated with passengers on-board a vehicle when they are held at a control station due to the application of the holding strategy. The fourth component corresponds to the extra travel time incurred by the passengers on board due to the transfer of other passengers. The longer the transfer is, the higher this component becomes. Finally, the fifth component is the extra waiting time of passengers whose station is skipped by an "expressed" vehicle, associated with the station-skipping strategy.

Next, the model for transit transport system, described in Sáez *et al.* (2009), is presented. The first state space variable modeled is the bus position at any instant *t*, $\hat{x}_b(t)$, is described as a function of the bus's instantaneous speed $v_b(t)$ that depends on the continuous time and the applied control actions. Let us start computing the position of the bus *b* in continuous time *t* as follows.

$$\hat{x}_b(t) = x_b(t_k) + \int_{t_k}^{t} \hat{v}_b(\vartheta)d\vartheta\,, \tag{6.2}$$

where t_k is the continuous instant at which the event *k* is triggered and $x_b(t_k)$ the position of bus *b* at instant t_k.

The instantaneous speed $\hat{v}$ (*t*) is modeled by assuming a constant speed (v_0) whenever the vehicle is moving, and the speed is equal to zero otherwise, which implies that the processes of acceleration and deceleration of the buses are ignored. Figure 6.9 shows the speed function of bus *b* while it is traveling from the station it reaches at instant *k* until the bus arrives at the next stop along its route (which is associated with future instant *k*+*d*). Notice that *d* corresponds to the time lapses (intervals) triggered by other buses of the fleet arriving at different bus stops, taking place while bus *b* is traveling between its current stop and the next (including the time it is at its current stop).

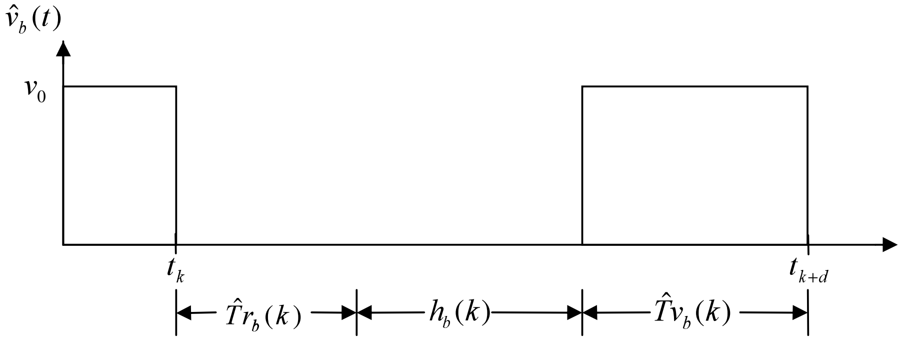


Figure 6.9. Example of bus speed between consecutive stops.

In the figure, $\hat{Tr}_b(k)$ is the estimated time associated with passenger transfer (maximum between the boarding and alighting times) and $\hat{Tv}_b(k)$ is the estimated travel time between two consecutive stations, namely station *p* and the next station. As defined above, the dispatcher decides the holding time at station *p*, denoted $h_b(k)$. Clearly, when a bus is at a bus stop, its velocity equals zero while the bus is transfering passengers and also during the holding period (if the bus is held there), which means that the instant speed actually depends on those variables. In this context, an estimation of the instantaneous speed can be computed as:

$$\hat{v}_b(t) = \begin{cases} 0 & t_k \le t \le t_k + \hat{Tr}_b(k) + h_b(k) \\ v_0 & t_k + \hat{Tr}_b(k) + h_b(k) \le t \le t_{k+d} \end{cases} \tag{6.3}$$

In order to trigger the next event of the dynamic model, the expected remaining time (measured from instant *t*) for the bus *b* to reach the next stop is required; it can be computed as follows:

$$\hat{T}_b(t) = t_k + Su_b(k) \cdot \left(h_b(k) + \hat{Tr}_b(k)\right) + \hat{Tv}_b(k) - t, \quad t_k \le t \le t_{k+d}. \tag{6.4}$$

Estimations of the continuous state space variables of our proposed scheme are given by (6.2) and (6.4). Next, the discrete output variables of the dynamic model, required for the HPC strategy ($\hat{L}_b(k+1)$ and $\hat{Td}_b(k+1)$), are defined and analytically computed.

First, let us define the predicted passenger load $\hat{L}_b(k+1)$, as the estimated number of passengers on bus *i* once it departs from the station. Analytically,

$$\hat{L}_b(k+1) = \begin{cases} \min\left\{\overline{L}, L_b(k) + Su_b(k)\left(\hat{B}_b(k) - \hat{A}_b(k)\right)\right\} & \text{if bus } b \text{ triggered event } k \\ L_b(k) & \text{otherwise} \end{cases}, \tag{6.5}$$

where $\overline{L}$ is the bus capacity, $L_b(k)$ is the load of bus *b* at instant *k*, $\hat{B}_b(k)$ corresponds to the expected number of passenger that will board bus *b*, constrained by the available capacity of the bus, and $\hat{A}_b(k)$ represents the estimated number of passenger alighting from bus *b* at event *k*. Note that $\hat{A}_b(k)$ and $\hat{B}_b(k)$ are obtained through a statistical analysis of data collected from sensors that should be located at stops and buses. In this approach (Sáez *et al.*, 2009), these estimations are obtained from data of both a set of previous similar days (off-line historical data) and dynamic information occurring the same day (on-line data).

Based on off-line data, estimated $\hat{A}_b(k)$ is using the most frequent destination patterns from previous days over the same period; then, those estimations are corrected with online destination data obtained from observed preferences from passengers already in the system. $\hat{B}_b(k)$ is computed based on both the estimated bus stop load $\Gamma_p(k)$ at instant *k* and the bus capacity; it is estimated considering autoregressive moving average models for the arrival time of passengers at stops. Moreover, the estimated transfer time defined before can be analytically described by $\hat{Tr}_b(k) = Max\left\{t_a \cdot \hat{A}_b(k), t_b \cdot \hat{B}_b(k)\right\}$ where t_a and t_b are the marginal rate of boarding and alighting respectively in seconds per passenger.

In addition, the estimated departure time $\hat{Td}_b(k+1)$ once the bus *b* departs from its current stop can be computed as

$$\hat{Td}_b(k+1) = \begin{cases} t_k + Su_b(k) \cdot \left(h_b(k) + \hat{Tr}_b(k)\right) & \text{if bus } i \text{ triggered event } k \\ Td_b(k) & \text{otherwise} \end{cases} \tag{6.6}$$

The prediction of the bus stop load $\hat{\Gamma}_p(k+1)$ (when bus *b* departures from stop *p*), defined as the number of passengers waiting at bus stop (station) *p* associated with the bus *b* that triggered event *k*; it can be computed as follows:

$$\hat{\Gamma}_p(k+1) = \begin{cases} \Gamma_p(k) + \hat{\delta}_p(k) - \hat{B}_b(k) & \text{if bus } b \text{ triggered event } k \\ \Gamma_p(k) \ + \hat{\delta}_p(k) & \text{otherwise} \end{cases}, \tag{6.7}$$

where $\Gamma_p(k)$ is the bus stop load at the same stop *p* at instant *k*. $\hat{\delta}_p(k)$ provides the number of passengers that arrive at the bus stop between instants *k* and the instant when the bus departure from this stop. $\hat{\delta}_p(k)$ is generated based on the statistical analysis of the data in both the previous similar days and the same day (both off and online historical data) and is estimated considering autoregressive moving average models for the arrival time of passengers to stops.

By using the prediction of the departure time as in equation (6.6), it is possible to predict the headway $\hat{H}_p(k+1)$ of the bus stop *p* where the event *k* was triggered, with respect to its precedent bus *b-1* when it reaches the same stop, which corresponds to event $k+1-z_{b-1}$. Analytically:

$$\hat{H}_p(k+1) = \hat{T}d_b(k+1) - \hat{T}d_{b-1}(k+1-z_{i-1}) \tag{6.8}$$

where $\hat{T}d_b(k+1)$ is associated with the bus *b* that triggers the event k, and $\hat{T}d_{b-1}(k+1-z_{b-1})$ represents the predicted departure time of precedent bus *b-1* that triggers the event $k-z_{b-1}$, at the same stop. The variable z_{b-1} represents the number of events between the arrival of the precedent bus *b*-1 and the bus *b*, both reaching the same stop.

Finally, the system based on the dynamic model must satisfy some physical and operational constraints. The first constraint corresponds to the capacity constraint (already stated above). This is a physical constraint in the sense that the bus cannot transport more passengers than its maximum capacity. We can also apply a service policy by setting such a capacity differently in order to avoid overcrowding. Both the precedence constraint and the demand consistency are relevant, because every passenger has a specific origin and destination. Precedence constraints avoid passengers getting off before they get on any bus. With regard to the demand, it is assumed that there are no transfer nodes, and therefore, once a passenger is on board a bus, he (she) will alight from the same bus at his (her) destination stop. Also, once a passenger arrives at their destination, he (she) will always get off the bus there (passengers want to minimize their travel time, so we assume that passengers do not stay on buses in loops).

Regarding bus operation, the model is constrained to stop at a station if there is any passenger requesting to get off, even though the model recommends performing a station-skipping action, similar to what is suggested by Sun and Hickman (2005). Thus, if the next stop is the destination of even one passenger then the skipping action cannot be applied and the bus must stop and the passengers waiting can board. This strategy seems to work better than including that aspect as a penalty in the objective function, in which case some of the passengers could end up getting off at a station different from their planned destination. On the other hand, if the model determines a holding action at a certain stop, which is not physically appropriate for such an operation, then the bus just stops during a lapse required for a normal passenger transfer operation.

As a physical constraint, and also for practical purposes, the holding control action can be applied just at specific stops, properly equipped to perform such an action. On the other hand, station skipping could be applied at every bus stop.

Each bus is identified by a unique internal label. However, the model allows the indices to be updated when a bus arrives at its next stop, sorted in such a way that bus *b-1* always precedes bus *b*. Under certain operational conditions, the model allows buses passing other buses along their route; in such cases the indices are properly updated and sorted.

Figure 6.10 shows the controller H-HPC structure including the inputs and output of this first level of H-HPC. All the variables in the figure were defined above.

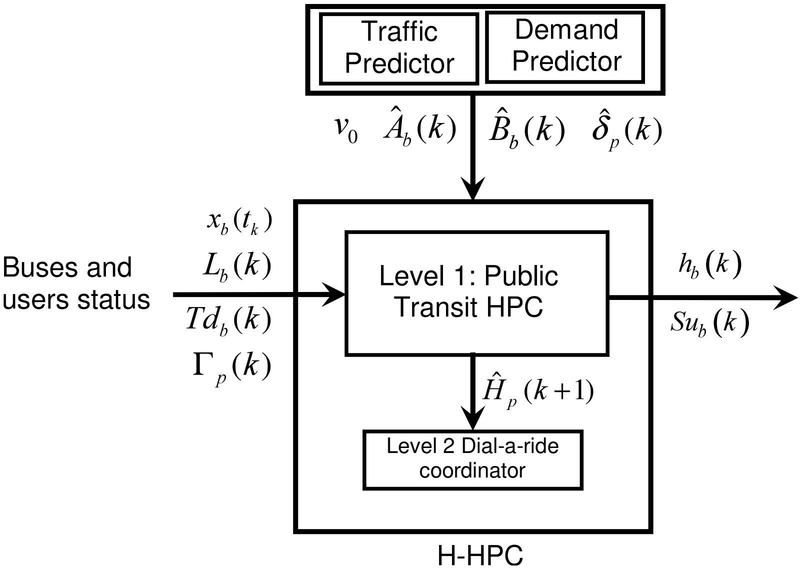


Figure 6.10. Level 1 H-HPC structure.

6.3.2. Level 2, Hybrid Predictive Control for the Dial-a-ride system.

An extended predictive model for the dial-a-ride system, such as that in chapter 4, is formulated in terms of three variables: estimated time of arrival to a stop, vehicle load among stops, and vehicle position. For this application, let us assume a fixed and known fleet size *F* over an urban area *A*. The specific location of a request (which includes its pickup as well as its delivery) is known only after the associated call is received by the dispatcher. A selected vehicle is then rerouted at real-time to insert the new request into its predefined route (sequence) while the vehicles are in motion. The assignment of the vehicle and the insertion position of the new request into the previous sequence of tasks associated with such a vehicle, are control actions decided by the dispatcher (controller) based on the objective function, which depends on the variables related to the state of the vehicles in real time. The fleet is in operation travelling within the area according to predefined routes. The modeling approach was defined in chapter 4, and next is summarized.

The proposed HPC dispatcher selects the optimal sequences based on the minimization of an ad-hoc objective function. The optimization variable in the HPC are the sequence of stops assigned to vehicle *j* at instant *k*, $S_j(k)$, given by:

$$S_j(k) = \begin{bmatrix} s_j^0(k) \\ s_j^1(k) \\ \vdots \\ s_j^{w_j(k)}(k) \end{bmatrix} = \begin{bmatrix} r_j^0(k) & P_j^0(k) & z_j^0(k) & \Omega_j^0(k) \\ r_j^1(k) & P_j^1(k) & z_j^1(k) & \Omega_j^1(k) \\ \vdots & \vdots & \vdots & \vdots \\ r_j^{w_j(k)}(k) & P_j^{w_j(k)}(k) & z_j^{w_j(k)}(k) & \Omega_j^{w_j(k)}(k) \end{bmatrix} \tag{6.9}$$

For vehicle *j*, the state space variables are the position $X_j(k)$, the estimated departure time vector $\hat{T}_j(k) \in R^{w_j(k)+1}$ and the estimated vehicle load vector $\hat{L}_j(k) \in R^{w_j(k)+1}$. The dynamic model, explained in details in chapter 4 and 5, for the vehicle *j* is as follows.

$$\hat{X}_j(k+1) = \begin{cases} P_j^{i^*}(k) + \int_{t_k}^{t_k+\tau} \hat{v}(t, p(t)) \dfrac{\left(P_j^{i^*+1}(k) - P_j^{i^*}(k)\right)}{\left\| P_j^{i^*+1}(k) - P_j^{i^*}(k) \right\|_2} dt & \text{if } i^* < w_j(k) \\ P_j^{i^*}(k) & \text{if } i^* = w_j(k) \end{cases} \tag{6.10}$$

$$\hat{T}_j^i(k+1) = \begin{cases} T_j^0(k) & i = 0 \\ t_k + \sum_{s=1}^{i} \kappa_j^s(k) & i \neq 0 \end{cases}, \quad i = 0,1,\ldots,w_j(k) \tag{6.11}$$

$$\hat{L}_j^i(k+1) = \begin{cases} \min\left\{\overline{L}_j, L_j^0(k)\right\} & i = 0 \\ \min\left\{\overline{L}_j, L_j^0(k) + \sum_{s=1}^{i} \left(2z_j^s(k) - 1\right)\Omega_j^s(k)\right\} & i \neq 0 \end{cases}, \quad i = 0,1,\ldots,w_j(k), \tag{6.12}$$

The details of (6.10), (6.11) and (6.12) can be found in chapter 4 and 5. The performance of the vehicle routing scheme will depend on how well the objective function can predict the impact of possible rerouting due to insertions caused by unknown service requests. Analytically, a mono-objective version of the proposed objective function for a prediction horizon *N*, can be written as follows:

$$\begin{gathered} \underset{S_k^{k+N}}{Min} \; \lambda J_1 + (1-\lambda) J_2 \\ J_1 = \sum_{\ell=1}^{N} \sum_{j=1}^{F} \sum_{h=1}^{h_{\max}(k+t)} p_h(k+\ell)\left(J_j^U(k+\ell) - J_j^U(k+\ell-1)\right) \\ J_2 = \sum_{\ell=1}^{N} \sum_{j=1}^{F} \sum_{h=1}^{h_{\max}(k+t)} p_h(k+\ell)\left(J_j^O(k+\ell) - J_j^O(k+\ell-1)\right) \end{gathered} \tag{6.13}$$

$$J_j^O\left(k+\ell\right)=c_T\left(\hat{T}_j^{w_j(k+\ell)}\left(k+\ell\right)-\hat{T}_j^0\left(k+\ell\right)\right)\Big|_h+c_L\sum_{i=1}^{w_j(k+\ell)}\left(D_j^i\left(k+\ell\right)\right)\Big|_h \tag{6.14}$$

$$J_j^U\left(k+\ell\right)=\sum_{m=1}^{4}\theta_m\sum_{i=\left\{i\in\left[1,\ldots,w_j(k+\ell)\right]/r_j^i(k+\ell)=m\right\}}\left(\Omega_j^i\left(k+\ell\right)\cdot J_m^U\left(r_j^i\left(k+\ell\right),z_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)\right)\Big|_h \tag{6.15}$$

The notation is defined in chapter 4. The expression for the vehicle operational cost (6.14), consists in a component depending on the total traveled distance, weighted by a factor c_L, and another on the total operational time, in this case at unitary cost c_T. Thus, $D_j^i\left(k+\ell\right)$ represents the distance between stops $i-1$ and i in the sequence of vehicle j.

In (6.15), θ_m, $m=1,\ldots,3$ is a weighting factor for each kind of user defined by the option in Figure 6.2. Note that demand of type 5 is not considered as in this level just dial-a-ride users are routed. Demand of type 4 is not considered either because it is not as attractive as the other options as it requires transferring twice. The demand 4 can be added in cases of having very long trips, maybe suburban or even interurban journeys. The analysis of them in the context of a system of this type is part of further research.

$J_m^U\left(\cdot\right)$ is the objective function oriented to measure the user cost of a user $r_j^i\left(k+\ell\right)$ whose journey is type m.

$$\begin{aligned}J_1^U\left(r_j^i\left(k+\ell\right),z_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)=\theta_e\cdot f_e^1\left(k+\ell\right)\cdot z_j^i\left(k+\ell\right)\cdot\left(\underbrace{\hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}}_{\text{waiting time}}\right)+\\ \theta_v\cdot f_v^1\left(k+\ell\right)\cdot\left(1-z_j^i\left(k+\ell\right)\right)\cdot\left(\underbrace{\hat{T}_j^i\left(k+\ell\right)-tr_{r_j^i(k+\ell)}^1}_{\text{re-routing time}}\right)\end{aligned} \tag{6.16}$$

$$\begin{aligned}J_2^U\left(r_j^i\left(k+\ell\right),z_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)=z_j^i\left(k+\ell\right)\cdot\theta_e\cdot f_e^1\left(k+\ell\right)\cdot\left(\underbrace{\hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}}_{\text{waiting time}}\right)+\\ \left(1-z_j^i\left(k+\ell\right)\right)\cdot\theta_v\cdot f_v^1\left(k+\ell\right)\cdot\left(\underbrace{\hat{T}_j^i\left(k+\ell\right)-tr_{r_j^i(k+\ell)}^1}_{\text{re-routing time}}\right)+\\ \left(1-z_j^i\left(k+\ell\right)\right)\cdot\theta_{st^p}\cdot f_{st^p}\left(k+\ell\right)\left(\underbrace{\hat{T}_{st}\left(r_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)-\hat{T}_j^i\left(k+\ell\right)}_{\text{waiting and travel time in transit system}}\right)\end{aligned} \tag{6.17}$$

$$
\begin{aligned}
J_3^U\left(r_j^i(k+\ell), z_j^i(k+\ell), \hat{T}_j^i(k+\ell)\right) &= z_j^i(k+\ell)\cdot\theta_{st^d}\cdot\left(\underbrace{\hat{T}_{st}\left(r_j^i(k+\ell),\hat{T}_j^i(k+\ell)\right)-t_{0r_j^i(k+\ell)}}_{\text{waiting and travel time in transit system}}\right) \\
& z_j^i(k+\ell)\cdot\theta_e\cdot f_e^2(k+\ell)\cdot\left(\underbrace{\hat{T}_j^i(k+\ell)-\hat{T}_{st}\left(r_j^i(k+\ell),\hat{T}_j^i(k+\ell)\right)}_{\text{waiting time in bus-stop}}\right)+ \\
& \left(1-z_j^i(k+\ell)\right)\cdot\theta_v\cdot f_v^2(k+\ell)\cdot\left(\underbrace{\hat{T}_j^i(k+\ell)-tr^1_{r_j^i(k+\ell)}}_{\text{re-routing time}}\right)
\end{aligned}
\tag{6.18}
$$

$\hat{T}_{st}\left(r_j^i(k+\ell),\hat{T}_j^i(k+\ell)\right)$ is the expected arrival time to the transfer bus-stop. The term related to the extra time experienced by passengers in this service (delivery time minus the minimum time the user could arrive to its destination) is weighted by a factor θ_v, and the term related to total waiting time of each passenger is weighted by θ_e. Note that the terms in the objective functions for user are weighted by the functions $f_v(k+\ell)$ and $f_e(k+\ell)$, which include a service policy for users, so the cost of a user that entered the system a long time ago is considered more importantly than another user who has just made the request, just like stated in chapter 5, under the MO-HPC approach for an isolated dial-a-ride system.

In (6.16) and (6.17) the variable weighting function is defined as:

$$
f_v^1(k+\ell)=
\begin{cases}
1 & \text{if } \hat{T}_j^i(k+\ell)-t_{0r_j^i(k+\ell)}<\alpha\left(tr^1_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) \\
1+\hat{T}_j^i(k+\ell)-t_{0r_j^i(k+\ell)}-\alpha\left(tr^1_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) & \text{if } \hat{T}_j^i(k+\ell)-t_{0r_j^i(k+\ell)}\geq\alpha\left(tr^1_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right)
\end{cases}
\tag{6.19}
$$

Expression (6.17) implies that if the delivery time $\hat{T}_j^i(k+\ell)$ associated with user $r_j^i(k+\ell)$ becomes greater than α times its minimum total time $\left(tr_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right)$, the weighting function

$f_v\left(k+\ell\right)$ grows linearly, resulting in a critical service for such a client. Regarding the waiting time factor,

$$f_e^1\left(k+\ell\right)=\begin{cases} 1 & \text{if } \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}\leq TT \\ 1+\hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}-TT & \text{if } \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}>TT \end{cases} \tag{6.20}$$

the intuition behind (6.18) is analogous to (6.17).

In the case of users type 3, in (6.18), the functions $f_v\left(k+\ell\right)$ and $f_e\left(k+\ell\right)$ are the following:

$$f_v^2\left(k+\ell\right)= \\ \begin{cases} 1 & \text{if } \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}<\alpha_2\left(tr^1_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) \\ 1+\hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}-\alpha_2\left(tr^1_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) & \text{if } \hat{T}_j^i\left(k+\ell\right)-t_{0r_j^i(k+\ell)}\geq\alpha_2\left(tr^1_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) \end{cases} \tag{6.21}$$

$$f_e^2\left(k+\ell\right)= \\ \begin{cases} 1 & \text{if } \hat{T}_j^i\left(k+\ell\right)-\hat{T}_{st}\left(r_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)\leq TT_2 \\ 1+\hat{T}_j^i\left(k+\ell\right)-\hat{T}_{st}\left(r_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)-TT_2 & \text{if } \hat{T}_j^i\left(k+\ell\right)-\hat{T}_{st}\left(r_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)>TT_2 \end{cases} \tag{6.22}$$

The idea behind (6.21) and (6.22) is the same that (6.19) and (6.20), but the threshold parameters TT_2 and α_2 could be tuned is a different way to consider that to wait in a transfer point is less comfortable for the user, as well as the travelling in a dial-a-ride vehicle before proceeding to the public transit system. Finally in (6.17), the term related with the waiting and travel on the bus system, after the portion of trip performed in the dial-a-ride system, considers a comparison between the best total trip time $tr_{r_j^i(k+\ell)}$ and the expected arrival time to the bus-stop of destination $\hat{T}_{st}\left(r_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)$. Notice that it is not possible to modify the buses control actions, if the resulting $\hat{T}_{st}\left(r_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)$ is not appropriate, however the system optimization will force the solution to modify the dial-a-ride routes in order to reach a reasonable travel time.

$$\begin{aligned} & f_{st^p}\left(k+\ell\right)=\begin{cases} 1 & \text{if} \quad \Theta\left(k+\ell\right)<0 \\ 1+\Theta\left(k+\ell\right) & \text{if} \quad \Theta\left(k+\ell\right)\geq 0 \end{cases} \\ & \Theta\left(k+\ell\right)=\hat{T}_{st}\left(r_j^i\left(k+\ell\right),\hat{T}_j^i\left(k+\ell\right)\right)-t_{0r_j^i(k+\ell)}-\alpha_2\left(tr_{r_j^i(k+\ell)}-t_{0r_j^i(k+\ell)}\right) \end{aligned} \tag{6.23}$$

In summary, integrated schemes involving fixed route systems and re-routable services for real-time routed transit are becoming increasingly attractive, mainly when information technology is used to determine vehicle and passenger call positions finding a real-time service setting. These schemes that add flexibility to the operation, could improve passenger demand, and improve the overall productivity from the efficient use of fixed route services. The typical constraints in many of the existing formulations, mainly in cases of traditional dial-a-ride type services without transfers, could result in suboptimal solutions. This is particularly significant when larger fleets of vehicles are used for such services resulting in several vehicle tours coming close in time-space. In this context, the aim of this chapter is to design, model and formulate this special type of mixed system, combining trunk corridors (fixed route system) with feeder services (reroutable scheme), allowing a passenger to access directly to either his(her) final origin or destination, or both. An application as well as the development of efficient algorithms to deal with a practical problem are left for the next steps of this research.

6.4. Discussion.

The major idea in this work is to combine a regular fixed-route public transport trunk service (using large buses in the operation) with a typical dynamic dial-a-ride service (served with small vehicles, such as vans). The major objective was to write an integrated HPC formulation for both systems, where the relations between systems are the transfer points. The better the coordination and synchronization of transfer operations, the better the performance of the whole system. Considering that, a regular passenger will have several options to travel from origin to destination, depending on the location of such points (close or far from a trunk bus route), and on the passenger willingness to pay higher fares for a more personalized service as well.

The proposed operational scheme was designed in order to minimize the total operational costs and to optimize the level of service of users, the latter by means of the minimization of travel and waiting times as well as number of transfers. The entire optimization scheme relies on the availability of computer and communication technology in order to allow real-time optimization and coordination/synchronization between subsystems. Fixed route services in transit without near-the-door pickup and delivery are not very attractive to certain users with poor accessibility to the bus route from their origin or destination, or both; however, fixed route services are recommended in case of some very high-density demand corridors. That is the major reason to propose more flexible alternatives to the user, taking advantages of fixed route (with high capacity vehicles) services on high-demand corridors, in combination with local dial-a-ride systems for low demand portions of the trip. The idea is to combine a traditional public transport service on trunk corridors (big buses operating with established stops along the route) with a more flexible system (reroutable vans or big cars), transferring passengers between systems at specific transfer stations. This type of scheme could become attractive to people who presently prefer the automobile to traditional transit systems for their regular trips.

In addition, as the hybrid predictive control optimization problem for the integrated dynamic public transport system is huge at every instant time, as further research it is proposed to study local optimization versus global optimization schemes, under an evolutionary multi-objective optimization predictive control framework. Specific evolutionary algorithms will be developed in order to propose real time optimization of the whole system, properly defining the system cost functions considering the necessity of coordination at transfer points but also considering the operator cost. Some insights regarding the implementation of this type of flexible systems will be provided, which can be incrementally phased or contracted out for private fleet operators. Potential zoning method and heuristics for reducing computational time will be also analyzed.

6.5. References.

Cortés, C.E., (2003). "High-Coverage Point-to-Point Transit (HCPPT): A new Design Concept and Simulation-Evaluation of Operational Schemes for Future Technological Deployment". Ph.D. Dissertation, University of California at Irvine, U.S.A..

Cortés, C.E., Jayakrishnan, R., (2004). "Analytical modeling of stochastic rerouting delays for dynamic multi-vehicle pick-up and delivery problems". Proceeding of the Triennial Symposium on Transportation Analysis (TRISTAN) V, Guadeloupe, French West Indies, June.

Cortés, C.E., Sáez, D., Milla, F., Núñez, A., Riquelme, M., (2009). "Hybrid Predictive Control for real-time optimization of Public transport system' operations based on evolutionary multiobjective optimization". Accepted, Transportation Research Part C.

Crainic T. G., F. Malucelli and M. Nonato (2001). "Flexible many-to-few few-to-many an almost personalized transit system". TRISTAN IV, São Miguel Azores Islands, pp. 435-440.

Malucelli, F., Nonato, M., Pallottino, S., (1999). "Demand Adaptive Systems: some proposals on flexible transit". Operational Research in Industry, T.A. Ciriani, et al., Editors, McMillan Press: London, pp. 157-182.

Hickman, M., Blume, K, (2000). "A method for scheduling integrated transit service". 8th International Conference on Computer-Aided Scheduling of public Transport (CASPT), Berlin, Germany.

Horn, M.E.T., (2002a). "Multi-modal and demand-responsive passenger transport systems: a modeling framework with embedded control systems". Transportation Research Part A, Vol. 36, pp. 167-188.

Horn, M.E.T., (2002b). "Fleet scheduling and dispatching for demand responsive passenger services". Transportation Research Part C, Vol. 10, pp. 35-63.

Horn, M.E.T., (2003). "An extended model and procedural framework for planning multi-modal passenger journeys". Transportation Research Part B, Vol. 37, pp. 641-660.

Horn, M.E.T., (2004). "Procedures for planning multi-leg journeys with fixed-route and demand-responsive passenger transport services". Transportation Research Part C, Vol. 12, pp. 33-55.

Jaw, J., Odoni, A., Psaraftis, H., Wilson, N., (1986). "A heuristic algorithm for the multivehicle many-to-many advance-request dial-a-ride problem". Working paper MITUMTA-82-3, M.I.T., Cambridge, M.

Liaw, C., White, C., Bander, J., (1996). "A decision support system for the bimodal dial-a-ride problem". IEEE Transactions on System, Man and Cybernetics 21, pp. 498-516.

Sáez, D., Cortés, C.E., Riquelme, M., Núñez, A., Tirachini, A., Sáez, E. "Hybrid Predictive Control Strategy for a Public transport system with uncertain demand". Under review, Transportmetrica.

Sun, A., Hickman, M., (2004). "The Holding Problem at Multiple Holding Stations". 9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT) San Diego, California, EE.UU. Available in http://fugazi.engr.arizona.edu/caspt/sun.pdf.

7. Conclusions

In this thesis a methodology for the design of predictive control strategies for non-linear dynamic hybrid systems was developed, including discrete and continuous variables. The methodology is designed for real-time applications, particularly the study of dynamic transport systems, considering operational and service policies, as well as costs reduction. The control structure is based on a proper definition of the key variables and their evolution in the future, a flexible objective function able to capture the predictive behaviour of the key variables of the system and efficient algorithms, mainly coming from the computational intelligence framework, to optimize performance indices for real-time applications. The framework of the proposed predictive control methodology is generic, and extensible to other industrial processes, and it is able to dynamically solve non-linear mixed integer optimization problems, which are known to be NP-Hard. In this chapter, the major contributions of this work are highlighted; the chapter finishes with a section that points out the most relevant future research lines arising from the thesis work.

Thesis Contributions

1.- A new fuzzy hybrid identification method

A new methodology for the identification of non-linear systems with mixed integer and continuous states and inputs was developed. Particulary, based on a hybrid model, local fuzzy models were used to better approximate the local non-linear behaviour of a system. The key element of the hybrid system identification methods is the detection and estimation of the switching regions based only on input-output data. The identification is performed by a combination of fuzzy clustering and principal eigenvector analysis. The use of the principal component was not only demonstrated to be very useful in the detection of switching points but also efficient in terms of the computation time as no expensive optimization process was included. The comparisons demonstrated the better performance of the fuzzy hybrid model identification with respect to the Takagi & Sugeno identification when comparing the *N*-step-ahead prediction performance.

2.- Hybrid Predictive Control Design based on EMO.

A new Hybrid Predictive Control problem was derived using the Evolutionary Multi-objective Optimization, punctually refereed to the use of Genetic Algorithms. Two different criteria were proposed to obtain an optimal control action from the Pareto front. Both criteria are directly related to the tracking error and control effort measurements and permit to define weighting factors of the typical Model Predictive Control. With regard to this last issue, two alternatives are considered to obtain the weighting values. Moreover, it was found that the model of the Pareto front identified through least mean squares provides the best results.

3.- HPC Design for a Dial-a-ride System

A dynamic formulation based on state space models for a dial-a-ride system designed as a HPC based on GA was derived considering historical demand information for a systematic future prediction of the key system variables to improve current dispatch decisions. HPC based on GA is an efficient solver in computation time for the proposed dial-a-ride system. A scenario of more than two-step-ahead tested via simulation provides efficient computation time.

A zoning method based on fuzzy clustering was proposed to systematically estimate origin-destination patterns from historical data and consequently obtain more reliable computations of the corresponding prediction probabilities. The proposed fuzzy zoning methodology improves the performance of predictive algorithms, mainly under more realistic historical data characterized by jumbled up trip patterns.

The integrated methodology (Fuzzy clustering and HPC based on GA) allows solving for more than two-step-ahead prediction to handle uncertain and heterogeneous demand pattern scenarios.

A fault detection scheme for a dial-a-ride system was defined for detecting unpredictable traffic conditions. The formulation considers uncertainty from possible future demand influencing routes of current customers, and the scheme also considers the uncertainty behind the traffic congestion conditions. A predictive model was proposed to modify the pre-planned schedule of vehicle routes based on traffic information around their routes as well as future insertions coming from unknown real-time service requests. Traffic congestion is modeled through the distribution of commercial speed of the vehicles on both relevant dimensions: time and space. The approach allows modeling not only predictable congestion conditions, but also unpredictable situations, such as incidents occurring unexpectedly at any location on the traffic network. In the second case, online (real-time) data is used regarding speed conditions from the fleet of vehicles on service.

The occurrence of an incident is treated under a FDI-FFTC scheme, allowing the reaction of the controller and the adjustment of the speed distribution parameters to significantly improve the dispatch rules under such a distorted scenario. The addition of the speed distribution into the model ensures a better estimation of both waiting and travel times, not only due to demand prediction, but also because of traffic congestion predictions, generating better real-time routing decisions, and consequently better performance of the dispatch service. The more information from the system is available, the better performance can be obtained from the HPC framework.

4.- MO-HPC design for a dial-a-ride system

A hybrid predictive control scheme for a dial-a-ride system using dynamic multi-objective optimization was developed. Different criteria are proposed to obtain control actions over real-time routing using the dynamic Pareto front. The criteria allow giving priority to a service policy for users, ensuring a minimization of operational costs under each proposed policy.

The service policies are verified approximately on the average of the replications. Under the implemented *on-line* system it is easier and transparent for the operator to follow service policies under multi-objective approach instead of tuning weighting parameters dynamically.

The multi-objective approach for such a dial-a-ride service permits to obtain solutions that are directly interpreted as part of the Pareto front instead of results obtained with mono-objective functions, which lack of direct physical interpretation (the weight factors are tuned but they do not allow applying operational or service policies such as those proposed here). Thus, more generic solutions are searched.

5.- Hybrid Predictive Control for an Integrated Public Transport System

An operational scheme of the integrated dial-a-ride problem of a fleet of vehicles together with a public transport system was designed in order to minimize the total operational costs and to optimize the level of service of users, the latter by means of the minimization of travel and waiting times as well as number of transfers. The entire optimization scheme relies on the availability of computer and communication technology in order to allow real-time optimization and coordination/synchronization between subsystems. Fixed route services in transit without near-the-door pickup and delivery are not very attractive to certain users with poor accessibility to the bus route from their origin or destination, or both; however, fixed route services are recommended in case of some very high-density demand corridors. That is the major reason to propose more flexible alternatives to the user, taking advantages of fixed route (with high capacity vehicles) services on high-demand corridors, in combination with local dial-a-ride systems for low demand portions of the trip. The idea is to combine a traditional public transport service on trunk corridors (big buses operating with established stops along the route) with a more flexible system (reroutable vans or big cars), transferring passengers between systems at specific transfer stations. This type of scheme could become attractive to people who presently prefer the automobile to traditional transit systems for their regular trips.

Future Research work

- New approaches of fuzzy hybrid modelling will be analyzed such as fuzzy clustering that generates both the fuzzy and hard partitions (fuzzy models with hybrid submodels). The stability issues of the proposed fuzzy hybrid modelling will be also studied. Also many dynamic transport systems applications could be solved with this method, from demand predictions, traffic, user behaviours, etc.
- The analytical formulation of HPC based on GA developed in this research can be potentially utilized to fit other numerical methods to solve the dial-a-ride system optimization process.
- The combination of historical data (off-line) with online information could be proposed in a more elaborate model able to capture imminent events in demand distribution that could affect the system performance.

- Other evolutionary algorithms for efficient optimization of HPC, such as PSO, could also be investigated, along with the convergence or trade/off with computation time of those algorithms. A good constraint handling technique is a very important issue in this kind of systems.
- More complex configurations of dial-a-ride systems could explore the inclusion of time windows (hard and soft), transfer points (in bus stops for example or another ad-hoc locations), and a better consideration of operational costs. A sensitivity analysis with regard to parameters of HPC applied to dial-a-ride is also interesting of being further investigated, for two and three-step-ahead problems. It is possible to improve the estimation of tuning variables, such as number of probable calls, future step time prediction (τ) which is unknown, prediction horizon (*N*), service policy, search over different feasible solutions structures, etc. One nice problem could be to solve the version of the problem where the demand is well-known a priori. Heuristic like evolutionary algorithms could be applied to finding a good solution in a reasonable computation time. The trade-off between accuracy and computation time should be considered.
- In addition, less restrictive dispatching rules, for which the analytical formulation approach would be useful, can be adapted within the same methodological scheme. Local heuristics could improve the performance to keep the effect of the *N*-step-ahead predictions. For example, to repair a route without considering the future request could results in myopic assignations.
- A real network configuration (with specific links and nodes) could be considered replacing the generic speed model in space by a velocity distribution model at a link level. This extension requires the coding of a time-dependent shortest path algorithm to compute optimal routes from point to point through the network, with link travel times depending on the time at which vehicles reach the upstream node of such a link. The coding could become harder, however the general framework remains the same. The use of traffic micro-simulation is proposed in order to have a better quantification of the performance of the system in real-time (simulation time). Better velocity models should result in better performance of the HPC scheme. In the case of unexpected incidents, a FDI-FFTC method is proposed. However, the rules can be further improved, sophisticating the way in which the system reacts to the occurrence of the detected fault. One straight extension is to somehow reroute those vehicles whose sequence path fall

into the fault area, even though the associated stops are not inside the affected zone. Besides, the present formulation can be extended to the use of fixed stations monitoring traffic conditions at strategically chosen locations over the urban area, in order to have more data available to better trigger the FDI detection.

- The integrated HPC allows systematizing the formulation of dial-a-ride systems as a control problem, which open more possibilities for using sophisticated techniques, not only to characterize the dynamic problem properly, but also to solve complex dynamic pick-up and delivery configurations unable to be treated without such a framework.
- The multi-objective predictive control design could be further generalized.
- In addition, as the hybrid predictive control optimization problem for the integrated dynamic public transport system is huge at every instant, it is also proposed to study local optimization versus global optimization schemes, under an evolutionary multi-objective optimization predictive control framework. Specific evolutionary algorithms can be developed in order to propose real time optimization of the whole system, properly defining the system cost functions considering the necessity of coordination at transfer points but also considering the operator cost. Some insights regarding the implementation of this type of flexible systems will be provided, which can be incrementally phased or contracted out for private fleet operators. Potential zoning method and heuristics for reducing computational time could also be analyzed.

APPENDIX: Publications generated during the development of the thesis (2006-2009)

International journal papers (ISI).

Cortés, C.E., Sáez, D., Milla, F., **Núñez, A.**, Riquelme, M. "Hybrid Predictive Control for Real-time Optimization of Public Transport System' Operations based on Evolutionary Multiobjective Optimization". Accepted, Transportation Research Part C.

Cortés, C.E., Sáez, D., **Núñez, A.**, Muñoz, D. "Hybrid Adaptive Predictive Control for a Dynamic Pick-up and Delivery Problem". Transportation Science, Volume 43, February 2009, Pages: 27-42.

Núñez, A., Sáez, D., Oblak, S., Škrjanc, I. "Fuzzy-Model-Based Hybrid Predictive Control". ISA Transactions, Volume 48, Issue 1, January 2009, Pages: 24-31.

Causa, J., Karer, G., **Núñez, A.**, Sáez, D., Škrjanc, I., Zupančič, B. "Hybrid Fuzzy Predictive Control based on Genetic Algorithm for the Temperature Control of a Batch Reactor", Computer & Chemical Engineering, Volume 32, Issue 12, December 2008, Pages 3254-3263.

Sáez, D., Cortés, C.E., **Núñez, A.** "Hybrid Adaptive Predictive Control for the Multi-Vehicle Dynamic Pick-up and Delivery Problem based on Genetic Algorithms and Fuzzy Clustering", Computers & Operations Research, Volume 35, Issue 11, November 2008, Pages: 3412-3438.

Cortés, C.E., Sáez, D., **Núñez, A.** "Hybrid Adaptive Predictive Control for a Dynamic Pickup and Delivery Problem including Traffic Congestion", International Journal of Adaptive Control and Signal Processing, Volume 22, Issue 2, March 2008, Pages: 103-123.

Submitted international journal papers ISI

Sáez D., Cortés, C.E., Riquelme, M., **Núñez, A.**, Milla, F., Tirachini, A. "Hybrid Predictive Control Strategy for a Public Transport System with Uncertain Demand". Submitted to Transportmetrica.

Núñez, A., Sáez, D., Skrjanc, I., Torres, P. "Fuzzy Model Identification of Non-linear Hybrid Systems". Submitted to IEEE Transactions on Systems, Man and Cybernetics Part B.

Muñoz-Carpintero, D., Sáez D., Cortés C.E., **Núñez, A.** "Evolutionary Algorithms to solve a Dial-a-Ride System based on a Hybrid Predictive Control approach". Submitted to Computers & Operations Research.

National journal papers (Chile):

Causa, J., Karer, G., **Núñez, A.**, Sáez, D., Škrjanc, I., Zupančič, B. "Control Predictivo Híbrido Difuso basado en Algoritmos Genéticos y su Aplicación al Control de Temperatura de un Reactor Batch (in Spanish)". Revista Chilena de Ingeniería. Anales del Instituto de Ingenieros de Chile, Vol. 120, N°3, December 2008, pp 113-123.

Núñez, A., Sáez, D., Cortés, C.E. "Aplicación de Técnicas de Inteligencia Computacional en un Problema de Ruteo Dinámico de Vehículos (in Spanish)". Revista Chilena de Ingeniería. Anales del Instituto de Ingenieros, Vol. 119 Nº1, April 2007, pp. 21-31.

International conference papers:

Cortés, C.E., Sáez, D., N**úñez**, A., Otárola,G. "Control Inteligente para Optimización en Tiempo Real de Sistemas de Transporte (in Spanish)". IX Congreso Internacional de Estudiantes y Profesionales de Ingeniería Civil: Avances y desafíos de la movilidad y del transporte (ANEIC 2009), Bucaramanga, Colombia, April 22-25, 2009.

Karer, G., Škrjanc, I., Zupančič, B., Causa, J., **Núñez**, A., Sáez, D. "Comparison of Branch and Bound and Genetic Algorithm based Approaches to Hybrid Fuzzy Predictive Control". 15th Zittau East-West Fuzzy Colloquium, Zittau, Germany, September 17-19, 2008.

Cortés, C.E., Sáez, D., N**úñez**, A., Gendreau, M. "Hybrid Predictive Control for the Dynamic Pick up and Delivery Problem with Variable Fleet Size based on an Evolutionary Multiobjective Optimization Approach (EMO)". International Federation of Operational Research Societies Conference, IFORS 2008, Sandton, South Africa, July 13-18, 2008.

Causa, J., Karer, G., N**úñez**, A., Sáez, D., Škrjanc, I., Zupančič, B. "Hybrid Fuzzy Predictive Control of a Batch Reactor using Branch and Bound and a Genetic Algorithm Approach", 17th IFAC World Congress, Seoul, Korea, July 6-11, 2008.

N**úñez**, A., Sáez, D., Cortés, C.E. "Hybrid Predictive Control for the Vehicle Dynamic Routing Problem based on Evolutionary Multiobjective Optimization (EMO)". 17th IFAC World Congress, Seoul, Korea, July 6-11, 2008.

N**úñez**, A., Cortés, C.E., Sáez, D., Milla, F., Riquelme, M. "Hybrid Predictive Control for Real-time Optimization of Public Transport System' Operations based on Evolutionary Multiobjective Optimization". 10th International Conference on Application of Advanced Technologies in Transportation, Athens, Greece, May 27-31, 2008.

Cortés, C.E., N**úñez**, A., Riquelme, M., Sáez, D. "Hybrid Predictive Control for the Dynamic Vehicle Routing Problem based on Evolutionary Multiobjective Optimization (EMO)". International Congress INFORMS Annual Meeting 2007, Seattle USA, November 4-7, 2007.

N**úñez**, A., Sáez, D., Oblak, S., Škrjanc, I. "Hybrid Fuzzy Predictive Control based on Evolutionary Multiobjective Optimization". Eurosim 2007, Ljubljana, Slovenia. September 9-13, 2007.

Cortés, C.E., Sáez, D., Sáez, E., N**úñez**, A., Tirachini, A. “Hybrid Predictive Control Strategy for a Public transport system with uncertain demand”. TRISTAN IV Congress, June 10-15, 2007, Phuket Island, Thailand.

N**úñez**, A., Sáez, D., Oblak, S., Škrjanc, I. “Hybrid Predictive Control based on Fuzzy Model”. IEEE World Congress on Computational Intelligence, FUZZ IEEE 2006, Vancouver, BC, Canada. July 16-21, 2006.

National conference papers (Chile):

Sáez, D., Cortés, C.E., N**úñez**, A., Riquelme, M., Milla, F., Otarola, G. “Hybrid Predictive Control for Real-Time Optimization of Public Transport Systems’ Operations”. Bus Rapid Transit Internacional Workshop, Santiago, Chile, 26-29 August, 2008.

N**úñez**, A., Riquelme, M., Sáez, D., Cortés, C.E. “Control Predictivo Híbrido para el Problema de Ruteo Dinámico de Vehículos basado en Optimización Multiobjetivo Evolucionaria (EMO) (in Spanish)”. XIII Congreso Chileno de Ingeniería de Transporte, Santiago, Chile, October 22-26, 2007.