ABSTRACT

Implementation of a Multi-Agent System in Java Agent Development Framework for a Large-Scale Fossil-Fuel Electrical Power Unit Control

Diego F. Estrada, M.S.E.C.E

Mentor: Kwang Y. Lee, Ph.D.

With technology and a larger-than-ever population, the society demand for energy has achieved records in the last few years. Currently, large-scale fossil fuel power plants, ranging in the hundreds of MW's, produce most of the power in the United States and must do efficiently with more added regulation and more in the short term future. The control for these large-scale power plants can be complex but must still address several different operating objectives.

Implementing advanced control for a large-scale power plant targeting multiple objectives can be computationally intensive as well as difficult when attempted in a centralized control configuration. In order for this control to be more flexible, adaptive and robust, it needs to be coordinated in distributed environments. The method presented in this thesis involves Multi-Agent System (MAS), which addresses the challenges of computing optimal multi-objective control using Java Agent Development (JADE) Framework, a much more communication-efficient, network-oriented environment.

Implementation of a Multi-Agent System in Java Agent Development Framework for a Large Scale Fossil-Fuel Electrical Power Unit Control

by

Diego F. Estrada, B.S., B.A.

A Thesis

Approved by the Department of Electrical and Computer Engineering

_____

Kwang Y. Lee, Ph.D., Chairperson

Submitted to the Graduate Faculty of
Baylor University in Partial Fulfillment of the
Requirements for the Degree
of

Master of Science

Approved by the Thesis Committee

_____

Kwang Y. Lee, Ph.D., Chairperson

_____

Steven R. Eisenbarth, Ph.D.

_____

Eunjee Song, Ph.D.

Accepted by the Graduate School

August 2013

_____

J. Larry Lyon, Ph.D., Dean

Page bearing signatures is kept on file in the Graduate School.

Copyright © 2013 by Diego F. Estrada

All rights reserved

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my gratitude to all who supported me during my graduate career here at Baylor University. In particular, I wish to thank my research advisor Dr. Kwang Y. Lee for his continuous encouragement, and prayers. I also want to thank my committee members, Dr. Steven Eisenbarth and Dr. Eunjee Song, who have shared their valuable time to assist me with my thesis and research.

I wish to thank the members and graduate students of the Power Systems Lab and the Computer Science Department, in particular, Craig Williams, Jason D. Head and Sanjeev Arora, for their help and collective work and contribution towards this project.

I also would like to thank my family for encouraging me and lifting me in prayers when most needed. Lastly, I wish to thank God for giving me strength and looking after me at all times.

Dedicated to the loving memory of my father

CHAPTER ONE

Introduction

This chapter intends to explain the motivation for this thesis project and introduce previous work that has led to this work. This thesis seeks to further, add and implement the methods proposed in the past and serve as reference for future projects. Section 1.1 aims to describe the personal motivation behind this thesis work. Section 1.2 gives the reader a look into the context and background of current energy production along with the main challenges to be dealt with and previous work in the field. Section 1.3 describes the new proposed approach, which is carefully explained throughout the different chapters of this thesis. And Section 1.4 explains how the rest of the thesis work is structured and how the chapters are described and what they contain.

1.1 *Motivation*

Industrialized civilizations exist today due to the ability of mankind to produce and manufacture products, transport and trade in a large scale. As scientific developments have helped humanity obtain means for a more comfortable life by allowing for manufacturing processes to be more cost-effective and accessible to the common person, there has been an increase in the demand for the energy required to produce and transport such products as well as energy consumed in households.

Energy is vital nourishment for society to be able to carry the most basic survival tasks and develop the industry. Energy has always played an important role from technology to the political atmosphere. In the past, energy production played an

important role in world history as the energy crisis in the 1970's reshaped the approach of several fossil-fuel-importing nations as their energy production was limited and an increasing demand meant dealing with foreign energy producers. Before that, one of the main causes for the Japanese attack on Pearl Harbor and subsequent involvement of the U.S. in the Second World War is believed to be the enforcement of a fossil-fuel embargo that directly affected the ability of Japan to carry on with hostilities against its neighbors [1]. The past offers plenty of occasions in which the control of energy and its production has led to conflict and this presents a serious challenge in order to keep peaceful trade among nations. Taking this information into account, it is evident that we must consider the current world energy production conditions and work towards maintaining a favorable world position.

1.2 *Background*

As the world demand for energy increases with emerging manufacturing nations, it is very important to maintain a supply of power that can match this demand while being able to meet different specific objectives that the energy market requires. In addition to a new competitive manufacturing demand for energy, the utility markets must not only be able to respond dynamically to financial conditions but must also be able to address government concerns in the production, transportation and regulation of energy.

Challenges in the Production of Energy Today

In recent years, the role of government has changed, becoming more active in the energy production by imposing taxes on gas emissions, enforcing stringent regulation that aim at relieving social environmental concerns and, most recently, the use of public funds

for private enterprises that seek to present a more clean and green alternative to the current options. Some of the latest regulations proposed by the United States Energy Information Administration [2] examine possible further regulation on the gas emissions for heavy-duty vehicles, addressing the emissions of sulfur dioxide (SO_2) and nitrogen oxides (NO_x) across States on fossil-fuel power plants with a capacity to produce more than 25 MW along with updating States air emission regulations, among other measures intended to address environmental pollution.

To offer a better perspective, in 2011 the United States consumed about 97.301 Quadribillion BTU of Energy in 2011, or equivalently, it spent about 1.205 Trillion dollars in energy consumption. However, domestic production for that same period of time accounted for only 78.096 Quadribillion BTU, or about 80.3% [3]. This data suggests that approximately at least 238 billion dollars is spent in foreign sources of energy that would otherwise remain in the economy and could be used to be reinvested in energy production and perhaps export the surplus of produced energy.

Besides the challenge of producing a higher share of energy consumed domestically, we must also consider the sources of the energy produced domestically. To better understand the sources of energy and the direction of the market we must consider more data of current trends.

Energy Production in the U.S. has grown considerably over the last 60 years, and as Tables I and II show, there is no sign of it slowing down. These tables also serve to display the current trends in the production of energy from fossil fuels compared to total energy as opposed to the production of energy from renewable energy sources.

Table I: Primary Energy Production by Fossil Fuels Sources [2].

	Coal	Natural Gas	Crude Oil	NGPL	Total
1950	14.06	6.233	11.447	0.823	32.563
1960	10.817	12.656	14.935	1.461	39.869
1970	14.607	21.666	20.401	2.512	59.186
1975	14.989	19.64	17.729	2.374	54.733
1980	18.598	19.908	18.249	2.254	59.008
1985	19.325	16.98	18.992	2.241	57.539
1990	22.488	18.326	15.571	2.175	58.56
1995	22.13	19.082	13.887	2.442	57.54
2000	22.735	19.662	12.358	2.611	57.366
2005	23.185	18.556	10.963	2.334	55.038
2006	23.79	19.022	10.801	2.356	55.968
2007	23.493	19.786	10.721	2.409	56.409
2008	23.851	20.703	10.509	2.419	57.482
2009	21.624	21.139	11.348	2.574	56.685
2010	22.038	21.823	11.593	2.781	58.235
2011	22.181	23.506	11.986	2.928	60.601

Table II: Primary Energy Production by Renewable Energy Sources [2].

	Hydro	Geothermal	Solar/PV	Wind	Biomass	Total
1950	1.415	N/A	N/A	N/A	1.562	2.978
1960	1.608	N/A	N/A	N/A	1.32	2.928
1970	2.634	0.006	N/A	N/A	1.431	4.07
1975	3.155	0.034	N/A	N/A	1.499	4.687
1980	2.9	0.053	N/A	N/A	2.476	5.428
1985	2.97	0.097	N/A	N/A	3.016	6.084
1990	3.046	0.171	0.059	0.029	2.735	6.041
1995	3.205	0.152	0.069	0.033	3.099	6.558
2000	2.811	0.164	0.066	0.057	3.006	6.104
2005	2.703	0.181	0.063	0.178	3.104	6.229
2006	2.869	0.181	0.068	0.264	3.216	6.599
2007	2.446	0.186	0.076	0.341	3.461	6.509
2008	2.511	0.192	0.089	0.546	3.864	7.202
2009	2.669	0.2	0.098	0.721	3.928	7.616
2010	2.539	0.208	0.126	0.923	4.341	8.136
2011	3.171	0.226	0.158	1.168	4.511	9.236

Additionally, Fig. 1 shows the evolvement of fossil fuels and renewable energy sources for the last 60 years. As seen, there has been constant growth in the renewable energy sources production, especially in the last few years. This fast growth reflects to some measure the intervention of the government into private enterprises and other subsidies for this kind of energy production. Fig. 1 also displays the fossil fuels energy production with a decreasing trend. It is important to note that the decreasing trend is not quite as fast as the growing one and also that fossil fuels make up for about 77.6% of energy production and it is expected to largely remain the main source of energy production for the next half of the century.

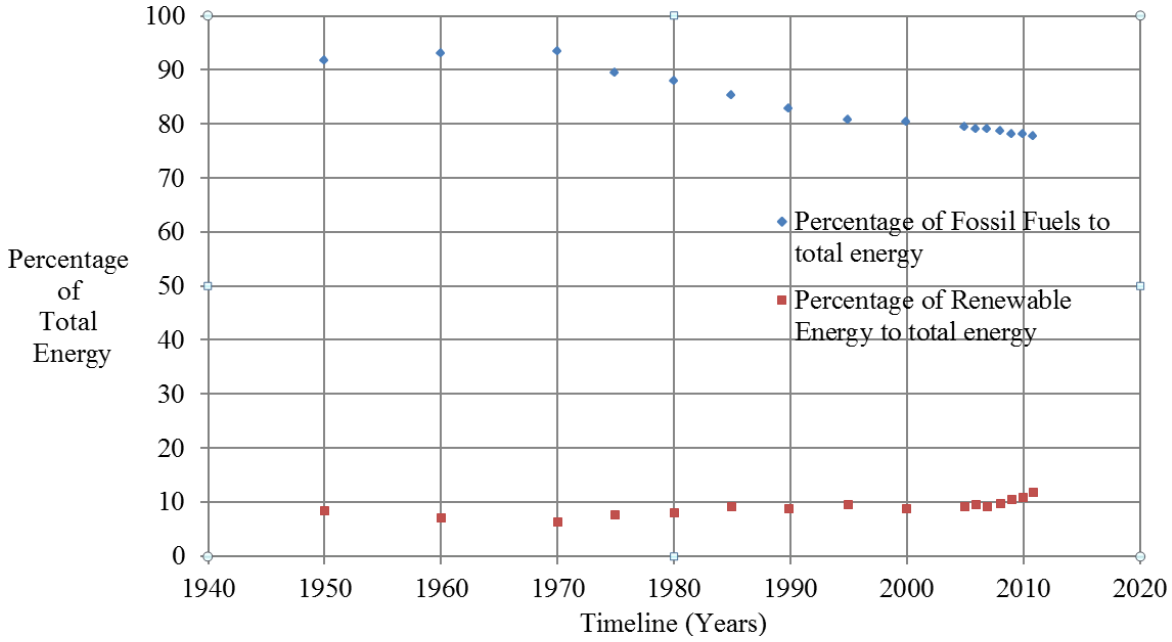

Fig. 1: Comparison of the energy by fossil fuels and renewable sources.

In fact, the U.S. Energy Information Administration has released a report with projections into 2040 shown on Table III [4]. These projections offer very interesting information. The renewable energy sources are expected to grow about 46% in

production as oppose to 22.5% for fossil fuel energy production growth by 2040. Even thou fossil fuels energy production is expected to grow at half the rate of renewable energy, the most significant information is that fossil fuels account for most of the new production of energy by 2040, accounting for over 70% of the new energy produced.

Table III: Energy Production Projections [4].

Sources of Energy	2011	2025	2035	2040
Petroleum	15.05	18.7	17.27	17.01
Natural Gas	23.51	29.22	32.04	33.87
Coal	22.21	22.54	23.6	23.54
Nuclear	8.26	9.54	9.14	9.44
Hydro	3.17	2.86	2.9	2.92
Biomass	4.05	5.27	5.83	6.96
Other Renewable	1.58	2.32	2.91	3.84
Total =	79.02	91.29	94.59	98.46

Since fossil fuels are expected to remain the most important source for energy production in the near future, it is imperative that we seek to provide with more efficient ways to produce energy in larger quantities while still being able to address multiple objectives with increasing constraints.

Electricity demand determines how much generating capacity is needed. When the demand is low, power plants with higher operating costs are taken off line or run at lower intensities, and the financial incentive to build new power plants is greatly reduced, but higher electricity demands, which the current scenario, power plants with higher operating costs are brought into service, increasing the costs for overall operation and therefore rising the price of electricity. Power plants are dispatched primarily on the basis of their variable cost of operation. Plants with the lowest operating costs generally operate continuously. On the other hand, higher variable cost power plants operate as

demand for generation increases. Since fuel prices influence variable costs, changes in fuel prices affect the choice of plants dispatched [2]. For this reason, examining energy production at large capacity power plants is a priority to ensure that they are able to meet the electricity demand in the best possible efficient manner.

Importance of Large-Scale Power Plants

Power plant generation has a wide range of capabilities, going from a few MW of power to several hundred to a few thousands. Large capacity power plants are not often easily described due to their complexity and different interconnected systems. Since larger power plants (500MW – 1200MW) have a much larger impact in the overall energy production in the US, these large power plants must face either be retrofitted to keep up with new regulation or retired facing numerous challenges, and therefore are the subject to this thesis work.

Retiring a power plant is subject to different factors, being the economic one the most important. The owner of the power plant must be able to determine the profitability of a power plant taking into account new environmental regulations, high source-fuel prices, low competitive fuel prices and future demand for energy. It is the case that many of the fossil-fueled power plants are old and do not have the capability to attend to these new environmental regulations. This has caused an ever growing number of power plants to be retired, pushing the ones that remain online to work closer to their peak power capacity.

To address these important issues, the power plant model under examination is a large steam-powered fossil-fuel-electric power unit (FFEPU). This particular drum-type boiler turbine-generator power plant is fueled by oil and is capable of producing up to

600 MW of power. The FFEPU can be modeled as a twenty-third order power plant model with as thirty-three subsystems. The FFEPU consists mainly of an oil-fired, balanced draft, controlled circulation drum boiler, two forced draft fans, six recirculation pumps, two condensate pumps, a 600 MW capable turbine and a generator, which is a 3-phase, 685,600 kVa, 22 kV, 60Hz unit with 0.9 power factor [5], [6].

For this particular type of power plant it has become more and more important to take into account factors as conservation and life extension but at the same time, it must be able to address other optimizing goals such as minimization of load tracking error, minimization of fuel consumption and heat loss rate, maximization of duty life, minimization of pollutant emissions just to name a few. These objectives have to be accomplished taking into account the FFEPU main goal, which is to meet the electric power demand at all times [7]. And in addition to these objectives, we can predict a more intrusive oversight of federal regulations adding to this already long list of required objectives. Multi-objective optimization can target these different goals very efficiently under the correct control methodology since traditional optimization techniques can be considered computationally heavy and burdensome [8]. Therefore, taking a look at a proper control approach is most advantageous.

The Coordinated Control Structure

In order for the FFEPU to provide the desired power demand, there have been techniques developed with different advantages and challenges that are based on essentially three different approaches to the overall control of the power plant. The methods of boiler-following control, turbine-following control and the coordinated control are briefly described below.

In the boiler-following control method the boiler is in charge of the main steam pressure and the turbine controls the power output. In other words, the steam turbine utilizes stored energy in the boiler to provide immediate load response. The boiler must then change its fuel firing rate to bring the steam pressure back to setpoint. The main advantage with this method is that a faster load demand response can be expected because the turbine reacts to the energy stored in steam quickly. On the other hand, since the boiler is incapable of producing steam at the same rate as the turbine converts the steam into energy, the steam pressure slowly reaches steady-state making the pressure less stable.

Similar to this power plant control method, in the turbine-following control the boiler is in charge of controlling the power output of the FFEPU by providing burning fuel as needed, while the turbine is in charge of controlling the main steam pressure and making sure that it is maintained at setpoint. The effects of this control method are the opposite as the boiler-following control method. The advantage is that it provides for a more stable pressure response to changes in the load demand. The main disadvantage, as expected, is that since the boiler is incapable to generate steam at a faster rate, the power generation will be slower.

The coordinated control method is a more practical approach that combines both the boiler-following control and turbine-following control to take advantage of their benefits while trying to minimize their challenges. The main concept behind this type of control is to keep the turbine and boiler operations together by providing them a common setpoint that takes into account the other system. In this manner, the energy input to the boiler is able to be matched to the energy demanded by the turbine. This allows the

system to keep a stable steam pressure but at the same time it makes the power plant capable of responding quickly to power load demand changes [7], [9].

Multi-Agent System and Distributed Control

While the coordinated control method is the particular strategy of this thesis, it is important to note the FFEPU under consideration is a very complex system that has to respond and meet different operating conditions, rendering the approach of centralized control an inadequate one. The complexity of a system that relies so heavily on highly coupled and interconnected subsystems can be severely affected by the failure of a single system [10]. An interesting approach to this is the one of Multi-Agent Systems (MAS). An MAS control structured is composed of several different agents. These agents are computer software programs situated in a given environment that are autonomous in nature, meaning they pose the ability to schedule action based on environmental observation and they work independently for a common goal but have determined tasks in a coordinated-control scheme. The MAS control structure is very efficient for a large-scale power plant model since MAS possesses robustness, adaptability, autonomy, and modularity [11], [12].

One of the main features of MAS is that no one agent has knowledge of the overall system goal, but through cooperation and flexibility they work together to contribute to the overall global result. An MAS can therefore be described as a loosely coupled network of problem solvers, in other words, they may act as individual agents that interact to solve a complex problem with the best information each agent has. This autonomous agent nature and intelligence can help reduce the complexity of the system by reducing the coupling problems among the subsystems [12].

Furthermore, the MAS implementation may be complex if the software used to implement it lacks the ability to perform efficient network communications among the agents. Furthermore, a standard for the development of agents, Foundation of Intelligent Physical Agent (FIPA), has been established and the software must be able to comply with this standard [13]. For this reason, a Java Agent Development (JADE) Framework has been designed (entirely implemented in Java) with the purpose of simplifying the implementing of MAS's by providing a set of graphical tools that support the debugging and deployment phases. The Java agent platform can be implemented across multiple stations, regardless of the underlying operating system, adding as another advantage that computationally intensive tasks are allowed to be distributed [14]. Such a decentralized control approach has to be distributed among different entities and is structured as described below.

Basic Overall MAS Distributed-Control Structure

Since a coordinated control approach is considered, it is necessary to provide a set of points to serve as the common demand. These set-points involve the power demand (E_d), pressure demand (P_d), and both the re-heater (RT_d) and super-heater temperature (ST_d) demands. For this thesis, the coordinated control is distributed among three different entities that have very important tasks at hand. These three entities are the reference governor, the feed-forward controllers and the feedback controllers. The reference governor is in charge of supplying the set-points power demand, the pressure demand and re-heater and super-heater temperatures demand signals, given a set of optimizing objectives that make up the coordinated control approach. Once set-points are calculated, the feed-forward controllers calculate the set of control values that will allow

the desired set-points to be obtained by the FFEPU. Finally, the four feedback controllers, one feedback controller for each of the subsystems, take into account both the desired set-points and the actual power plant output to determine a set of feedback control values to successfully meet the required objectives and preferences illustrated in Fig. 2.

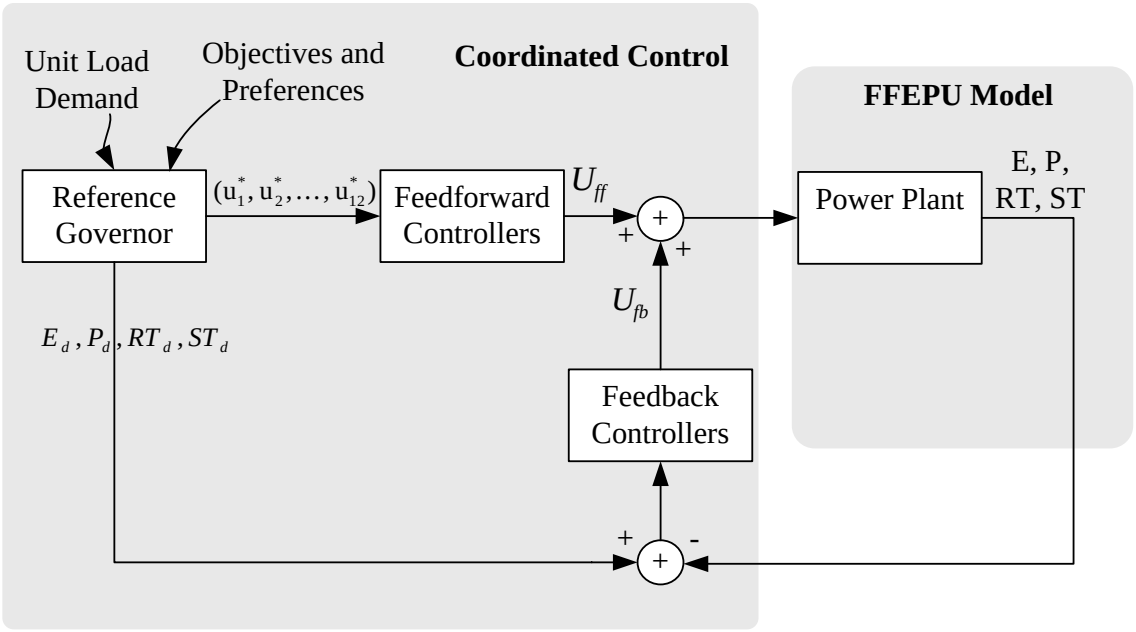

Fig.2: The Coordinated Control for the FFEPU.

This is the basic coordinated control approach and it is possible to address multi-objective goals and preferences by implementing it through an intelligent decentralized control, the Multi-Agent System structure. The MAS allows these three different entities as well defined agents to work independently towards a common goal, making the overall control structure more robust and dynamic.

1.3 *Thesis Contributions*

In the past, different decentralized control system topologies have been at the center of research work, especially for power plant of different magnitudes, different intelligent control systems, and different underlying supporting software. Previous work that support the progress of the research presented here include the development of a large-scale power plant [5], [15], [16], the study of heuristic optimization algorithms for power systems applications [7], [8], development of intelligent feed-forward control for power plants [10],[12], implementation of MATLAB-based MAS system for multiple-scale power plants [6], [19]. The proposed approach in this thesis is to develop MAS for large-scale power plant in the Java Agent Development (JADE) framework and further develop functional robust and dynamic control agents.

1.4 *Scope of Thesis*

This thesis consists of seven chapters. The first chapter presents the motivation to undertake such a research project, gives some background into the current energy production with its challenges, presents previous work in the field and it introduces briefly the proposed approach.

The second chapter describes the particular large-scale power plant model chosen for this project. Each one of the four physical power plant modules are described in detailed, and the FFEPU model is presented.

The third chapter gives an overview of the overall FFEPU coordinated control architecture. The chapter describes the overall FFEPU feed-forward control and its main characteristics, introduces the functionality of a reference governor, use of heuristic search algorithms, and outlines the feed-forward controllers.

The fourth chapter focuses on the proportional integral (PI) feedback control process and the online gain optimization process, which is in charge of tuning PI gains for the feedback controllers to ensure that the power plant response be more optimal and accurate in case of load tracking errors. This chapter explains the motivation and need of an online gain optimization process, describes how heuristic optimization algorithms can address the task of optimization and the implementation within the coordinated control model is explained.

The fifth chapter defines what a Multi-Agent Systems (MAS) is. The chapter begins by introducing the motivation behind the development and implementation of MAS-based control architecture and it outlines its benefits to this type of control. The chapter also outlines background concepts about the MAS, describes the agents architectures and the development of the individual agent and how the agents are organized. The Java Agent Development (JADE) Framework is introduced as the underlying software platform supporting the MAS architecture and the advantages of JADE over other software platforms are outlined. The JADE-based MAS coordinated control is developed, explaining the objectives, the agent hierarchy and all the different type of agents with each of their individual tasks and how they interact with each other and FEEPU. Also, a closer look is taken to the JADE-based communication, which presents a layer of communication between agents, and another layer between the individual agent and the computational intensive software, MATLAB in this case, in charge of performing calculating burdensome tasks.

The sixth chapter takes the JADE-based MAS configuration with all the tools previously described and implemented and presents the results of different stages of the

system at different conditions and it displays the most important advantages of the proposed approach.

Finally, chapter seven makes observation about the implementation, results and it draws conclusions about the system, points out challenges in the implementation and makes suggestions about interesting research work that can be added to the system to further improve the robustness and efficiency of the system.

CHAPTER TWO

Power Plant Model

This chapter describes the overall large-scale fossil fuel power plant system under consideration with detailed insight into its characteristics and the corresponding modules in which the power plant is divided. Section 2.1 describes the overall power plant from which a model is used to implement the proposed control configuration. Section 2.2 describes the characteristics associated with this particular model. Section 2.3 describes the modules and subsystems that conform the large-scale FFEPU under consideration.

2.1 *Large-Scale Fossil Fuel Power Plant Description*

The large-scale power plant under consideration in this thesis is a 600MW oil-fired drum-type boiler-generator fossil-fuel electrical power plant unit (FFEPU).

The steam generating unit is an oil-fired, balanced draft, controlled recirculation drum boiler which can deliver up to 4.2×10^6 (lbs.)/hr of steam at a pressure of 2600 psig and $1005\,^{\circ}\mathrm{F}$ and reheat from $625\,^{\circ}\mathrm{F}$ to $1000\,^{\circ}\mathrm{F}$. The boiler unit also has six recirculation pumps that supply the required recirculation flow with only four of these needed to supply sufficient flow for full load operation for the FFEPU. To address air flow, two forced-draft fans permit the primary air and two induced-draft fans are set to maintain the furnace pressure at a desired pre-set point [5].

Furthermore, in the boiler system two condensate pumps and a combination of combined booster and main boiler feed pumps address the feedwater flow.

The turbine system is a tandem compound triple pressure steam turbine. The system is composed of a high-pressure, intermediate pressure and two double flow low pressure turbines rotating on a common shaft at a rated speed of 3600 rpm, or 60Hz, and exhausting pressure at 2 in. Hq absolute. At maximum load, the throttle steam is designed to meet conditions of 2400 psig and 1000°F, and reheat steam at 1000°F. The turbine system is coupled directly to the generator unit, which is capable of producing 685,600kVa, 3-phase, 60Hz, at 22kV at a power factor of 0.90 [5].

2.2 *System Description and Characteristics*

The FFEPU model represents accurately the boiler-turbine unit and needs to provide not only a physical realistic model of the mechanical power systems but must also incorporate interactions between the mechanical and electrical component of the power plant. This FFEPU model [6] is an improvement to previous models [15], [16] for two very important aspects. Previously, the condensate dynamics and the feedwater dynamics had not been considered nor modeled due to the assumption that they do not have a considerable effect, but these dynamics are explicitly modeled in [6]. Secondly, this model actually accounts for the electrical fans and pumps and how their dependence on voltage and frequency affect the overall system. Other important features that characterize this FFEPU model are:

- The model development is based upon physical processes with model parameters determined from geometry, material properties and manufacturer's data.
- Nonlinearities usually overlooked to obtain simpler models are included to allow the model to be valid over a wide operating range.

- In order to provide physically realistic thermodynamic properties, steam table property fits are used.
- It is possible to verify the power plant model since an actual operating power plant was used as prototype and data can be compared.

Therefore this FFEPU model [6] presents characteristics that are desirable for implementation of the proposed control configuration.

2.3 *Power Plant Modules and Subsystems*

The FFEPU model has been implemented and simulated in MATLAB previously [18]. The FFEPU is a twenty-third order nonlinear dynamic model and it is divided in four different modules, which are the boiler module, the turbine-generator module, the condenser module and the feedwater module. Each of these modules is also composed of a total of thirty three subsystems. Twelve of these subsystems model the boiler module, eleven make the turbine module, two make up the condenser module and eight make up the feedwater module. This distribution is better represented in Table IV. Many of the modules are interconnected and also many of the subsystems are connected to each other. The model has twenty-three state variables distributed throughout the FFEPU and has twelve control valves, seven controlling the boiler module, two controlling the turbine module, one controlling the condenser module and two controlling the feed-water module [5]. These twelve control valves are the only variables that can be manipulated to reach the desired FFEPU response, making it very important to implement not only feedback control but also an intelligent feed-forward control that is capable of producing accurate control valves signals. These control valves are shown in Table V below.

Table IV: FFEPU Modules and Subsystems.

Boiler Module	Turbine Module
Air Preheater	Crosspipe
Downcomers	Generator
Drum	HP Turbine
Forced Draft Fan	Intercept Valve
Furnace Gas	IP Turbine
Induced Draft Fan	LP Turbine
Primary Heater	Re-heater
Primary Super Gas	Re-heater Spray
Secondary Heater	Re-heater Gas
Secondary Super Gas	Steamchest
Spray Heater	Throttle Valve
Waterwall	

Condenser Module	Feedwater Module
Condenser	Boiler Feed Pump
Condenser Pumps	Deaerator
	Deaerator Valve
	Economizer
	Economizer Gas
	Feed Water Valve
	HP Feed Water
	LP Feed Water

Table V: FFEPU Control Valves in each Module.

Boiler Module	Turbine Module
u1: Fuel Flow	u8: Governor Control Valve
u2: Gas Recirculation	u9: Intercept Valve
u3: Induced Draft Fan	
u4: Forced Draft Fan	
u5: Combustor Gun Tilt	
u6: Super-heater Spray Flow	
u7: Re-heater Spray Flow	
Condenser Module	**Feedwater Module**
u10: Deaerator Valve	u11: Feed-water Valve
	u12: Feed Pump turbine Flow

In order to better understand the manner in which the modules interact with each other and where all the subsystems are located within the FFEPU, Fig. 3 [19] displays the FFEPU under consideration with all of the four modules, the thirty-three subsystems within the modules, the set-points to be established by the coordinated control, the location of each of the twelve control valves, and the interconnections of the subsystems to different entities within the FFEPU.

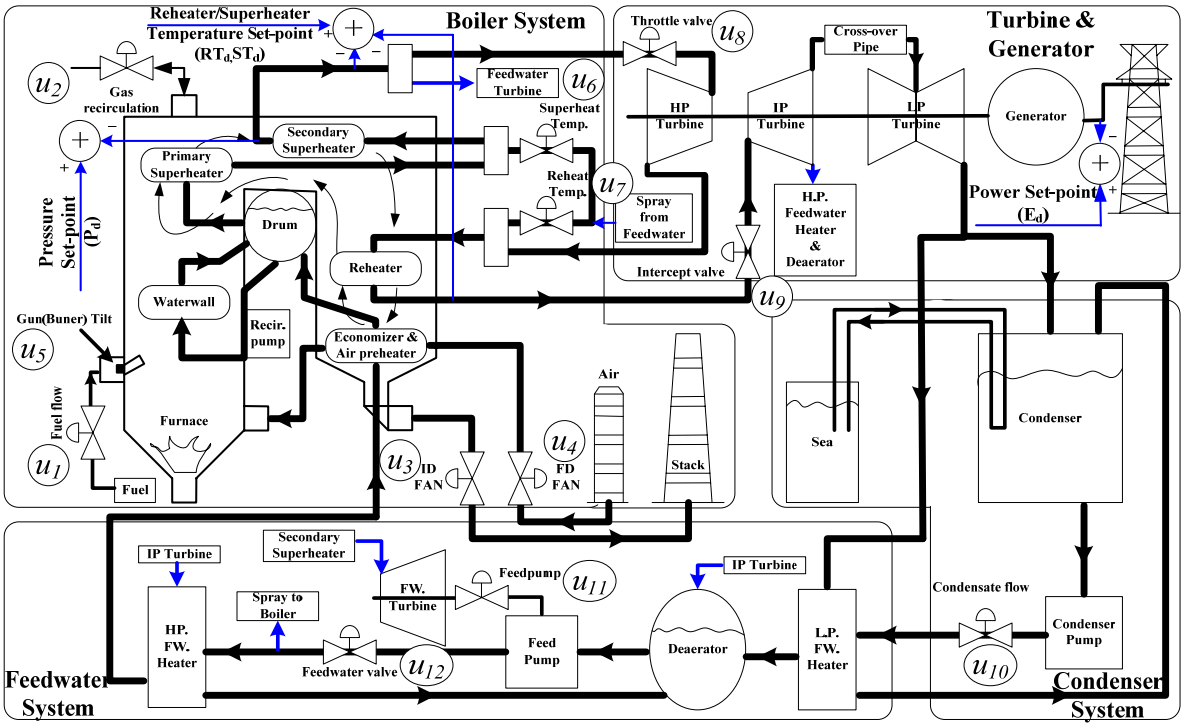

Fig. 3: FFEPU Modules and Subsystems [12].

The mathematical set of differential equations describing how the state variables are defined and how they interact with each other is more accurately presented in Appendix A along with important nomenclature necessary to understanding the dynamic equations.

Boiler Module

The first module we encounter is the boiler system. This boiler is able to produce 4,200,000 lbs. of steam per hour at a temperature of 1005°F and a main steam pressure of 2600 psig. This particular boiler is equipped with a reheat stage that allows cold flow coming from the high pressure turbine (Turbine Module) exhaust to be reheated from a temperature of 625°F to 1000°F. The boiler is composed of twelve different subsystems shown previously in Table IV.

The boiler module works in close conjunction with the feedwater module and the turbine module. The simplest way to describe the interaction between these modules is that the feedwater supplies high pressure hot water for the boiler drum and eventually the hot water will become high pressure steam that will be used by the turbine module to transform the heat energy into electrical power. But it is necessary to take a closer look at the subsystems interaction.

The boiler process starts at the economizer, which is basically a heat exchange subsystem. The economizer receives water coming for the high pressure feedwater heater (Feedwater Module) and feeds it into the drum to be mixed with the water already there to maintain the drum water level. At this stage there is a mixture of saturated steam and water. The saturated water will flow from both the drum and the economizer into six downcomers to the recirculating pumps. This will allow the water flow to go to the waterwall, where the heat produced by the furnace directly causes the saturated water to evaporate and join the saturated steam back into the drum, completing the water circulation cycle in the boiler drum. On the other hand, the mixture of saturated steam

leaves the drum, goes through scrubbers before it reaches the primary superheater, which is the first heat exchanger that is heated by the flue gasses leaving the furnace.

The flue gasses heats by convection the primary superheater first, then the secondary superheater, reaching the re-heater subsystem before heating the economizer, which previously was explained to heat the feedwater before getting to the drum, and finally leaving the FFEPU system through the induced draft fans into the stack out into the atmosphere. In this process it is important to note that the output of the secondary heater is the main steam pressure and temperature (two of the set-points of the coordinated control demand signals) that will head into the turbine module. For this purpose, before the steam reaches the secondary superheater the steam goes through a superheater spray flow that helps control the temperature at its set-point before reaching the throttle valve. Also, the re-heater actually has as input steam that comes out of the high pressure turbine and reheats it before it goes into the intermediate pressure turbine to gain more energy.

Finally, the furnace is supplied oil by five levels of oil guns with four guns in each level. The burner tilt angle also plays an important role as it keeps the reheat temperature at its set-point by controlling the heat exchange radiation distribution. Since the furnace needs air to combust the oil there is a forced draft fan that is fed by air from the atmosphere. Before reaching the furnace, this air goes through the air pre-heater subsystem [5]. To better illustrate this rather complex process, Fig. 4 displays the boiler process and interactions between modules and subsystems.

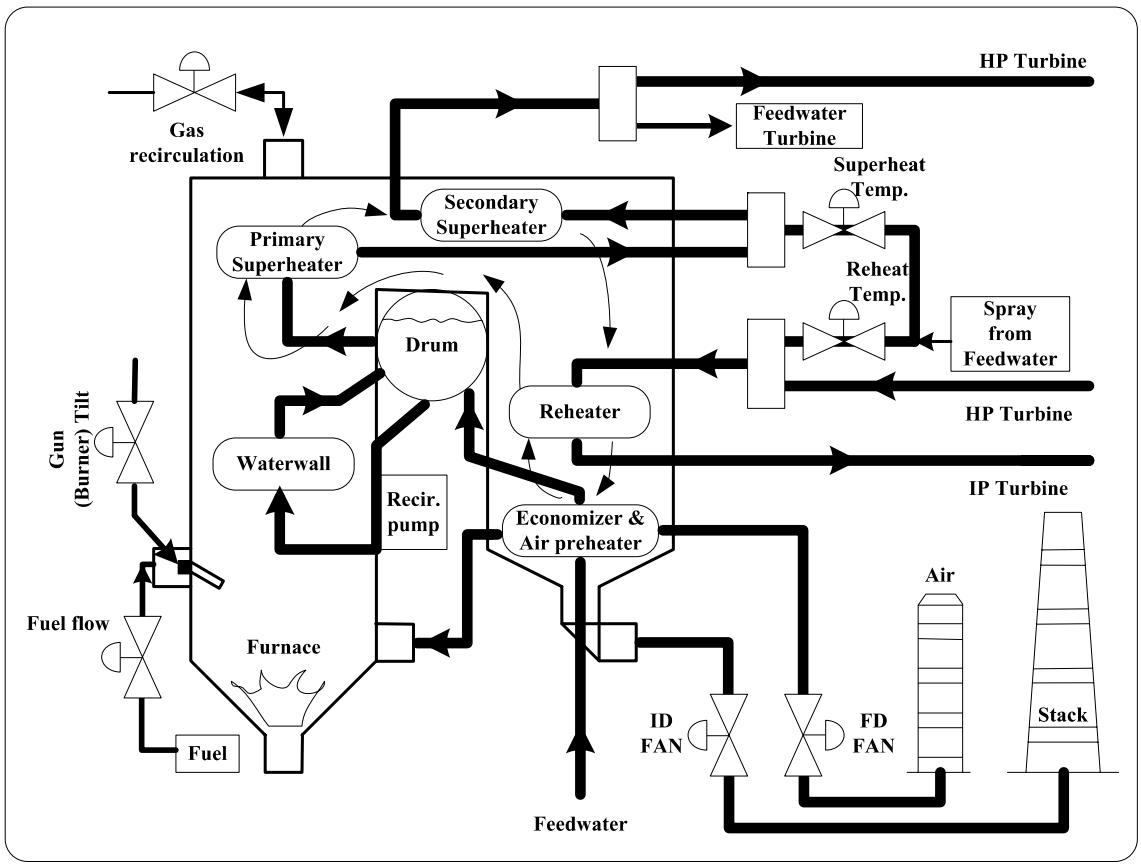

Fig. 4: Boiler Module and Subsystems [6].

Turbine Module

Some of the dynamics of the turbine module are presented already in the boiler module. The turbine is composed eleven subsystems with three of them as turbines, each handing a different level of steam pressure. The turbine module transforms the thermal energy from the hot pressurized steam, transforms it into kinetic energy through throttle valves, converts this energy into mechanical energy in each of the three turbines that share a common shaft and it finally converts the sum of the power from the three turbines into electrical power at the generator, which is capable of 685,600kVa, 3-phase, 60Hz, at 22kV at a power factor of 0.90.

The steam coming from the secondary superheater goes through a governor control valve, which controls the steam velocity and pressure before it goes into the high pressure turbine, the first one to meet. The main function of the governor valve and the intercept valve is to control the steam pressure and velocity to maintain a constant angular speed at the turbine shaft. After the steam leaves this turbine, it goes back into the boiler modules to a re-heater stage where heat transfer increases its pressure and temperature. Afterwards, the steam goes into the intermediate pressure turbine, which outputs steam into a crossover pipe before it reaches the low pressure turbine. The mechanical energy for the shaft rotation is converted in electrical power by the generator and the low pressure steam is now headed to the condenser module [5]. Fig. 5 shows the turbine module process.

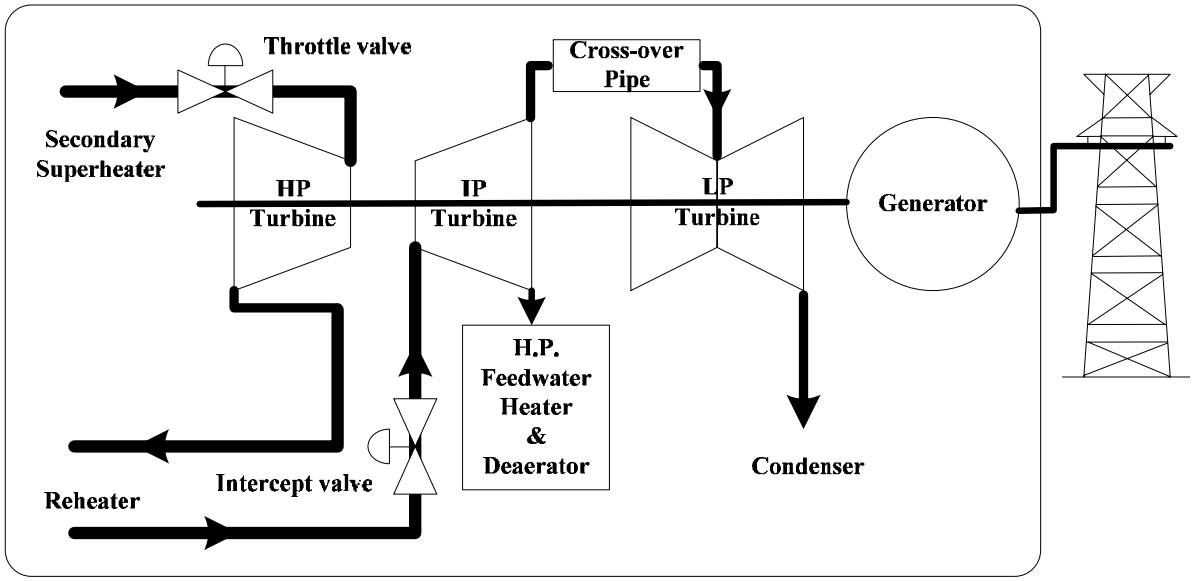

Fig. 5: Turbine Module and Subsystems [6].

Condenser Module

The condenser module is the least complex among the FFEPU mathematical modules. It comprises two subsystems. The main function of the condenser subsystem is to cool down the steam coming out of the low pressure turbine so that it becomes pure distillate water, which is necessary for the feedwater heating subsystem (Feedwater Module). This is done by allowing the steam to flow through cooling pipes from an outside source, which, in this particular example is the sea, as shown in Fig. 6, where the heat is dissipated. The steam condensate is then discharged at a higher than the steam saturation temperature while maintaining an economical condenser pressure. The condenser pumps subsystem is comprised of two pumps which propel the condensate water into the low pressure feedwater heater subsystem [5].

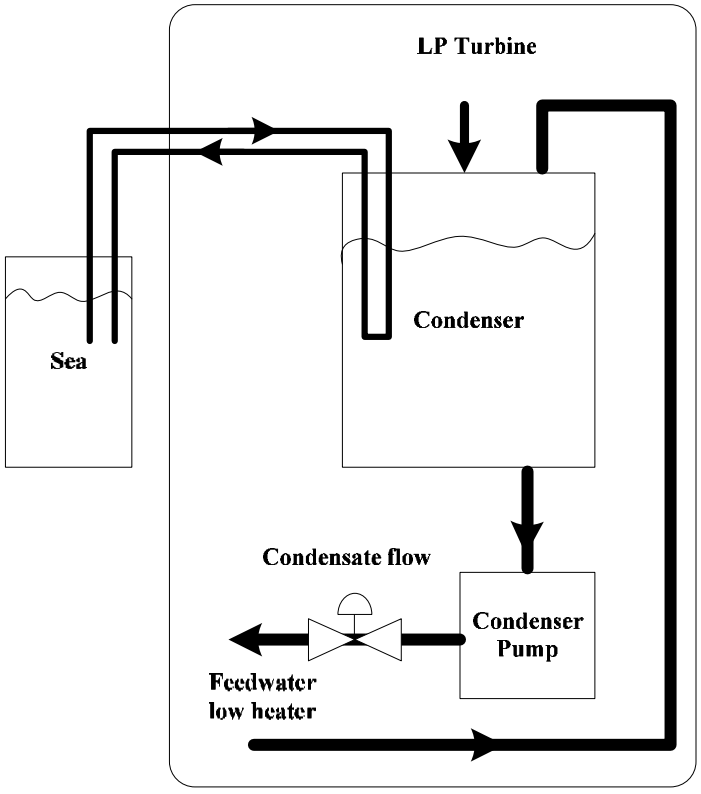

Fig. 6: Condenser Module and Subsystems [6].

Feedwater Module

Lastly, the feedwater module, which supplies water to the boiler module and some subsystems in other modules, is organized in eight different subsystems described in Table IV. The source of water for the feedwater comes from the condenser pumps subsystems, which provides pure distilled water to the first feedwater subsystem, the low pressure feedwater heater. The heating is aided by the supply of steam coming from the low pressure turbine output, which helps attain a water temperature suitable for saturation levels. The output water from the low pressure feedwater heater goes into the deaerator, which is basically a large open heater. In addition, steam from the intermediate pressure turbine helps attain saturated levels temperature for the water. Next, the deaerator supplies water to the boiler feed pump and booster pump, which supply water to the high pressure feedwater heater. These pumps are powered by a feedwater turbine, which uses steam generated at the secondary superheater as its source of energy. Following, the water leaving the pumps meet a feedwater valve before reaching the heater. This valve regulates the flow of water into the boiler module. Before it reaches the high pressure heater, part of the water is pumped into the superheater spray flow and the re-heater spray flow within the boiler module, which are two control valves that help maintain the pressure demand and superheater and re-heater temperature set-points. Finally, at the high pressure feedwater heater, steam is allowed from the intermediate pressure turbine. The heated water from the feedwater module flows again into the economizer to be heated before it reaches the boiler drum, describing the complete cycle of the FFEPU model. Fig. 7 illustrates the whole feedwater modules and subsystems.

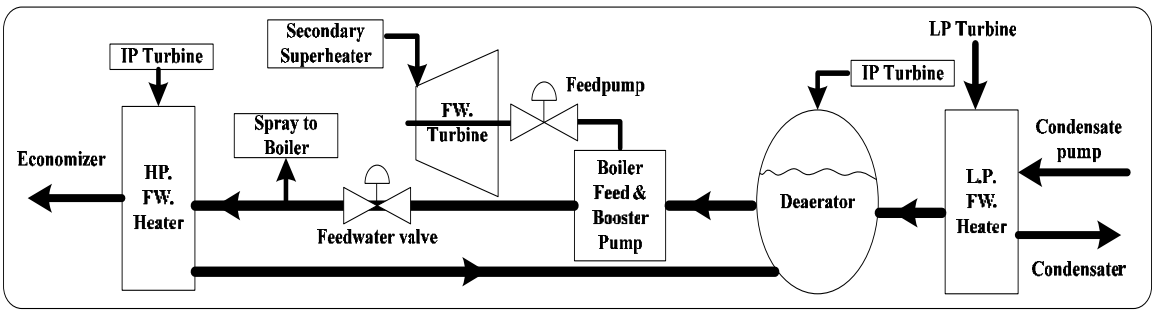

Fig. 7: Feedwater Module and Subsystems [6].

CHAPTER THREE

Power Plant Control System Architecture

This chapter aims to describe the overall Fossil Fuel Electrical Power Unit (FFEPU) control system setup and architecture. Section 3.1 describes the overall control system configuration for the FFEPU and its main characteristics. Section 3.2 describes how the goals of a coordinated control configuration can be achieved through the use of a reference governor. Section 3.3 outlines the process for the feed-forward control component.

3.1 *Overview of the Distributed FFEPU Control System*

In order to obtain a FFEPU system that works in a faster and more stable manner, the control configuration should combine the advantages of the boiler-following control and the turbine-following control described previously in Chapter 1, making the coordinated control configuration ideal since it takes advantage of the fast response of the boiler-following control configuration and takes advantage of the stability provided by the turbine following control configuration [9].

In the coordinated control configuration the unit load demand (E_{uld}) signal is taken as the reference to provide set-points for the boiler module and turbine module to use simultaneously within the FFEPU. These set-points are the power demand (E_d), pressure demand (P_d), re-heater temperature (RT_d), and super-heater temperature demand (ST_d) for the 600 MW FFEPU.

The coordinated control configuration is divided in three basic components: the reference governor, the feed-forward controller and the feedback controller, which are explained in more detail in the following sections. Fig. 8 displays the coordinated control configuration with its three main control components along with the FFEPU system.

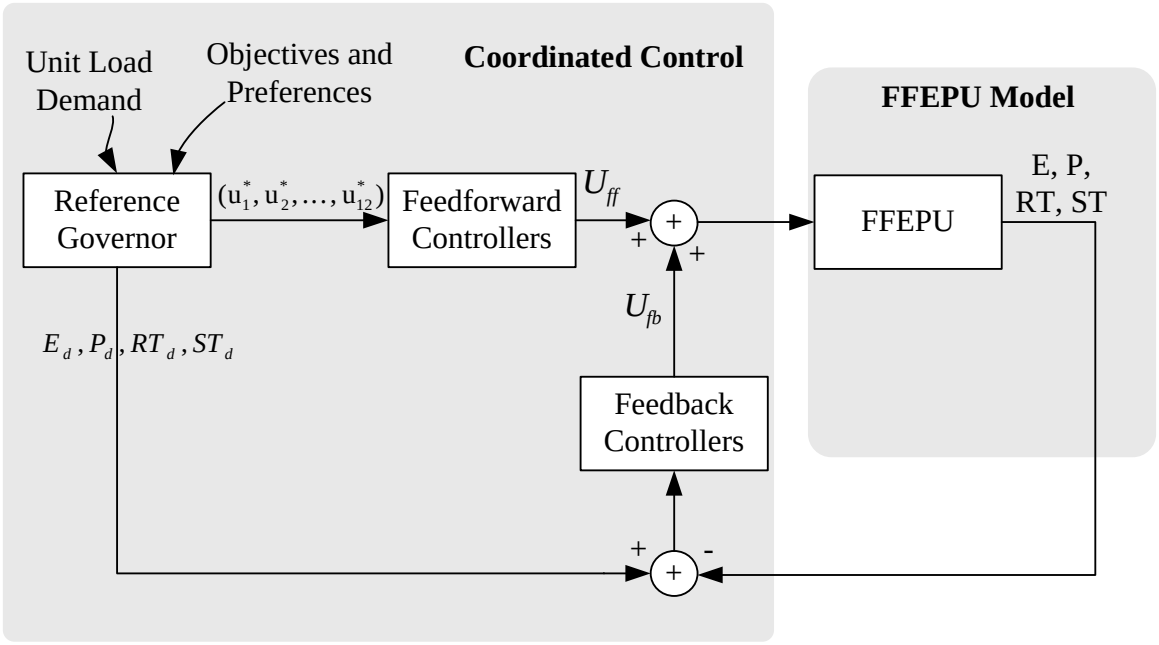

Fig. 8: Distributed Coordinated Control for the FFEPU.

As seen in the graph above, the reference governor obtains a Unit Load Demand (E_{uld}) already established by the user and also given a set of multiple objectives and preferences to generate the set-points for which the feed-forward controllers produces twelve control signals that are distributed among the FFEPU modules. In order to provide a fast enough response and avoid steady- state output tracking error, feedback controllers are implemented to complement the feed-forward control actions. Not shown is the implementation above is the addition of an online feedback gain optimizer process, which will be presented and discuss in Chapter 4.

The task of the reference governor is to optimally map the set-points and a set of corresponding control signals, which sends to the feed-forward controller to calculate control valve signals, for the functioning of the FFEPU by searching the best possible solution to the multiple optimization objectives, many of which can present conflicting requirement.

The feedback controllers are divided among the four modules of the power plant and they are implemented by PI controls that generate feedback control valve corrections to account for errors between the set-points generated by the reference governor and the actual output of the power plant. Table VI illustrates the control valves the feedback controllers aim to correct. In the distributed coordinated control implemented by the MAS, it is possible to tune the feedback PI gains with an online gain optimization process. A JADE-based MAS offers several advantages that will be explained more detailed in Chapter 5.

Table VI: Feedback Control Valves for each FFEPU Modules.

Boiler Module	Turbine Module
u_fb1: Fuel Flow	u_fb8: Governor Control Valve
u_fb2: Gas Recirculation	u_fb9: Intercept Valve
u_fb3: Induced Draft Fan	
u_fb4: Forced Draft Fan	
u_fb5: Combustor Gun Tilt	
u_fb6: Super-heater Spray Flow	
u_fb7: Re-heater Spray Flow	
Condenser Module	**Feedwater Module**
u_fb10: Deaerator Valve	u_fb11: Feed-water Valve
	u_fb12: Feed Pump turbine Flow

3.2 *Reference Governor*

For the 600 MW FFEPU, power demand, the main steam pressure and re-heater and super heater steam temperature are the set-points, and they should be mapped by the reference governor. To do this, the reference governor process is divided in three different tasks. First, the reference governor is given a unit load demand (E_{uld}) curve, for which the reference governor is able to establish a set of possible operating regions, by making use of a Power-Pressure Operating Window, for the twelve control actions (Ω_1, $\Omega_2, \Omega_{13}, \ldots, \Omega_{12}$), which is the upper and lower limit that the control action can be for a given (E_{uld}).

Second, the reference governor is also given a list of multiple optimization objectives and preferences that have to be taken into account to generate twelve normalized control actions. Note that the generated control actions must be normalized to be used by a steady-state model, which is used to evaluate the results of the given heuristic optimization process. To find the adequate control actions, the reference governor must perform an optimization search, which will be discussed with much detail in the next sections.

Finally, the set-points (E_d), (P_d), (RT_d) and (ST_d) can be established from the given normalized control actions by the use of the steady state model of the unit that is capable of generating such set-points from the evaluation of a steady-state model for the FFEPU. Fig. 9 illustrates the processes associated with the reference governor, the inputs and outputs and variables within the reference governor.

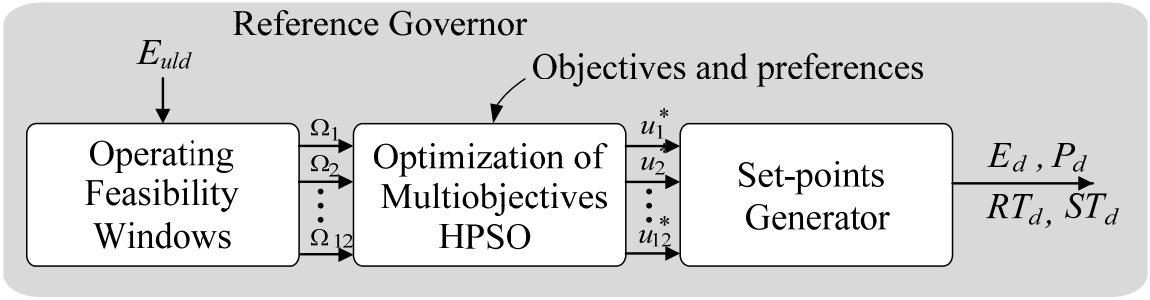

Fig. 9: Processes within the Reference Governor.

Power-Pressure Feasibility Window

The first task at hand by the reference governor is to find the operating regions for each one of the twelve control actions by a given Unit Load Demand (E_{uld}). The operating regions are important for the optimization of the multiple objectives and preferences since in order to run the heuristic algorithm, the reference governor must know the upper limit and lower limits for the control actions. For this purpose, a collection of sets of control actions inputs and steady-state power output, pressure output, re-heater and super-heater temperatures must be established.

Specifically, the reference governor must establish the upper and lower pressure output levels, the upper and lower re-heater temperature levels and the upper and lower super-heater temperature levels for a given unit load demand that will permit the FFEPU power output to be accurate. Along with the acceptable pressure and temperature set-points, the upper and lower limits for the control actions that fall within the category of accurate power output to the given unit load demand must also be established. The collection of these control action inputs and steady-state set-points outputs are operating points, and they are obtained by evaluating them in a systematic process. [12], [19] and [6] present the process to establish an operating window for different set-points. The

proposed approach takes the FFEPU model and runs simulations with different set-point patterns. Specifically, the power demand is swept every 10MW for the region of interest, in this case from 600 MW to 200 MW. Additionally, for every single power demand point the pressure demand is varied from 2400 psia to 1400 psia in steps of 50 psia. Finally, for every single combination of power demand and pressure demand, the temperature demands are swept from $1050\,^{\circ}\mathrm{F}$ to $850\,^{\circ}\mathrm{F}$ in steps of $20\,^{\circ}\mathrm{F}$.

In addition, the criterion for which an operating point is considered is acceptable is determined by two factors. The power output should reach steady-state in less than 500 seconds and it should match the power demand by a steady-state error of no more than 1 MW.

Even though there are four set-points and a four-dimensional operating window would accurately represent all the acceptable operating set-points, we can make a few assumptions about the FFEPU. Since the design and implementation of the re-heater and super-heater are basically the same and the outputs for their systems are very similar, we can assume that their set-points are equal. Furthermore, we can also simplify the operating window from a three-dimensional graph by noting that the re-heater and super-heater operating range for all power demands is between $1000\,^{\circ}\mathrm{F}\,(1459\,^{\circ}\mathrm{R})$ and $900\,^{\circ}\mathrm{F}$ $(1359\,^{\circ}\mathrm{R})$.

These assumptions are helpful to generate a two-dimensional operating window that will aid the reference governor in finding a suitable range for the control actions limits. With the given collection of acceptable set-points, the control actions that meet the criterion are also used to determine the operating control regions $(\Omega_1, \Omega_2, \Omega_{13}, \ldots, \Omega_{12})$ for the FFEPU [6]. Fig. 10 displays the power-pressure operating window showing the

acceptable pressure range for the given power demand and Fig. 11 shows the control actions feasibility regions for the power demand from 450MW to 600MW, which is the range of interest for the FFEPU.

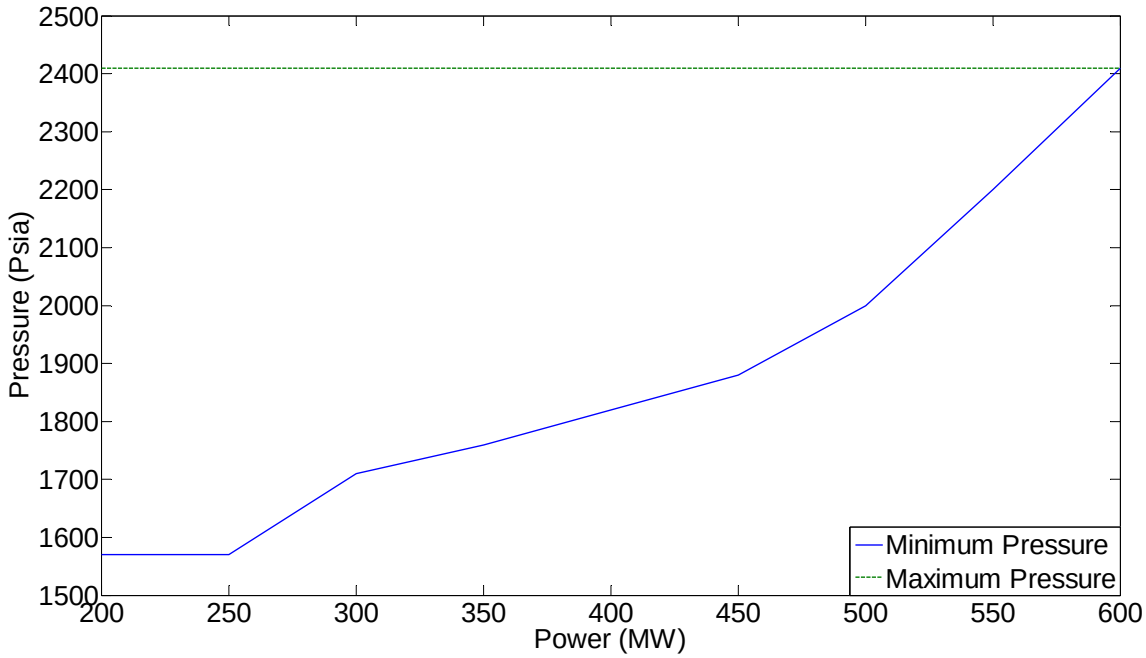

Fig. 10: Reference Governor Power-Pressure Operating Window.

Once the control actions feasibility regions are obtained, the next step in the reference governor is to perform the heuristic optimization algorithm, which will use these regions as their search space and it will generates normalized control action signals.

Multi-Objective Optimization

Obtaining the feasibility regions, shown in Fig. 11, is only the first step. To obtain the actual control actions signals it is necessary to take into account much more than the set-points. Since there are many modules and subsystems and twelve control actions freedom to choose the control actions allows setting objectives that deal not only with

steady-state control of the FFEPU outputs, but also with long-term functioning, atmosphere emissions or fuel consumption.

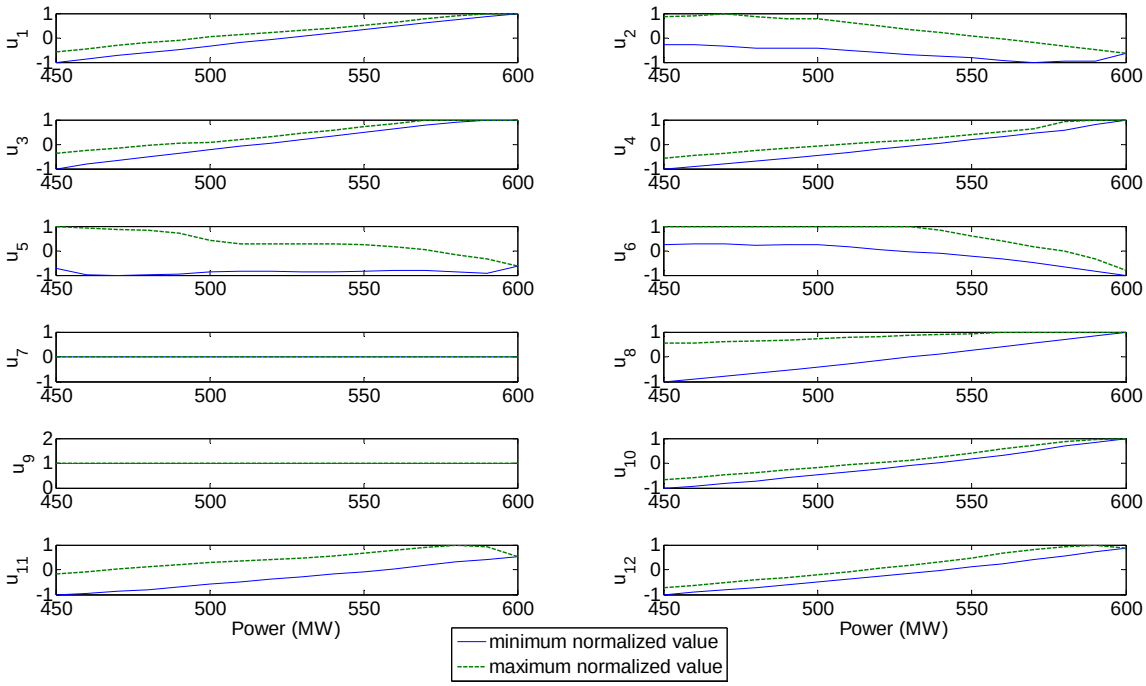

Fig. 11: Control Actions Feasibility Regions ($\Omega_1, \Omega_2, \Omega_{13}, \ldots, \Omega_{12}$).

To obtain the control actions for the FFEPU the multiple objectives must be defined first. These objectives function must target not only the FFEPU output error but the control actions themselves, even if some of them present conflicting requirements. The objective functions are formulated as:

$$J_0(u) = |E_{uld} - E_d|,\ J_1(u) = u_1,\ J_2(u) = -u_2,\ J_3(u) = u_3,$$

$$J_4(u) = -u_4,\ J_5(u) = -u_5,\ J_6(u) = u_6,\ J_7(u) = u_7,\ J_8(u) = -u_8, \qquad (1)$$

$$J_9(u) = -u_9,\ J_{10}(u) = -u_{10},\ J_{11}(u) = -u_{11},\ J_{12}(u) = -u_{12}$$

The objectives defined above are directly derived from the control action inputs and the FFEPU power output difference to the power demand set-point. The goal of

multi-objective optimization is to obtain a set of control actions that will minimize the chosen objective functions (1) by the given preferences. The first and most important of the objective functions is $J_0(u)$, which represents the load tracking error. The rest of the objective functions deals directly with the control valves $(u_1, u_2, \ldots, u_{12})$. The description of each of the objective function is given by Table VII below.

Table VII: Definition of Objective Functions.

Objective Function	Description
$J_0(u)$	Load tracking error
$J_1(u)$	Fuel consumption through the fuel flow valve
$J_2(u)$	Use of flue gas through the gas recirculation valve
$J_3(u)$	Pollutant emission through the induced draft fan valve
$J_4(u)$	Use of air through the forced draft fan valve
$J_5(u)$	Heat distribution through the burner tilt
$J_6(u)$	Heat loss rate through the super-heater spray valve
$J_7(u)$	Heat loss rate through the re-heater spray valve
$J_8(u)$	Use of steam pressure through the governor valve
$J_9(u)$	Use of steam pressure through the intercept valve
$J_{10}(u)$	Use of condensate water through the deaerator valve
$J_{11}(u)$	Use of water through the feed-water valve
$J_{12}(u)$	Use of water through the feed pump turbine valve

These objective functions aim to be minimized, therefore sometimes directly minimizing the control input signal is enough. For instance, to maintain a more efficient of resources it is aimed at using less fuel but still attaining an accurate power output. $J_1(u)$ is defined as the fuel consumption through the fuel flow valve and is directly aim at minimizing the control action u_1. This makes since u_1 actually controls flow of fuel into the furnace. A particular power plant that seeks to save fuel resources but still deliver an accurate power demand may set the preferences to optimize functions $J_0(u)$ and $J_1(u)$

while finding optimal solutions for the rest of the control actions that target the set preference. Another example deals with reducing the emission of pollutants into the atmosphere. This particular objective is important since more government regulations are set on energy production that does not meet certain standards. As seen in (1), the function is, again, directly related with the control action u_3, which is the control signal that controls the induced draft fan. This certainly makes sense since the induced draft fan allows the flue gasses from the furnace into the stacks, which leads the gasses into the atmosphere. Therefore, minimizing this control signal means allowing less of these gasses to escape and reducing the emission of pollutants.

It is important to note, that reducing the control action signals does not target all of the objectives. For some of them the opposite is necessary to be optimized. This is the case of objective function $J_8(u)$, which is defined as the use of steam pressure through the governor valve, which is the subsystem before the high pressure turbine. This objective function targets wear reduction, permitting the subsystems to have a longer equipment life and it is associated with the governor control valve u_8. If this control signal was minimized, meaning it actually allowed less steam through it, the steam pressure would actually increase. Therefore, to prolong life equipment, the control action must do exactly the opposite and aim to maximize its signal. Most of the objectives that are related to the use or equipment life need to be maximized and they carry a minus (-) sign.

These multi-objectives can be targeted all at once, individually or it can be divided in certain groups. The objectives that will be optimized need to be chosen before the reference governor can generate control action signals. This is due to that in the

optimization process, the best set of control actions are selected by a cost function, which combines all of the objective functions in a cost function. It is necessary to clearly state which functions are to be optimized so that the correct preference weight vector, β, can be implemented. Three cases are denoted:

- Case 1: Minimize $J_0(u)$ only.
- Case 2: Minimize $J_0(u),\ J_1(u),\ J_2(u)$.
- Case 3: Minimize $J_0(u),\ J_1(u),\ J_2(u),\ \ldots,\ J_{12}(u)$.

In Case 1, β is defined a vector that puts all the weight on the first objective function, thus β = [10, 0 , 0, …,0]. In Case 2, β must account for other objectives so the preference vector accounts for their weights, thus β = [10, 0.25, 0.25, 0,…,0] and subsequently Case 3 takes into account all of the objective functions and has a preference vector of β = [10, 0.25, 0.25,…,0.25]. It is for this reason that the feasibility regions and the control actions are normalized; otherwise the control actions with largest numerical values would take an undeserved priority.

Before the optimization of these different objectives can begin, we must note that many of the objectives are conflictive to one another and that has to be taken into account since by simply adding the objectives function results and assigning them weights will likely not yield the best result. In previous research, it has been proposed to minimize the maximum deviation (2) to address the conflicts between objectives.

$$\delta_m = \max_{j=1,\ldots,k} \delta_j,\ \delta_j \geq 0 \tag{2}$$

where δ_j is the deviation for any given of the multiple objectives functions and δ_m is the maximum of any of the objective functions deviations.

Yet, the results do not yield a balanced cost function, putting too much priority to some objective functions. [6] proposes that instead of minimizing the maximum deviation for any single objective function, the cost function should minimize the weighted combined sum deviation of the objective functions at hand. This process (3) is shown below:

$$\delta_s = \underset{i=1,\ldots,k}{\text{sum}}\ \delta_i, \quad \delta_i \geq 0$$

$$\delta_i = \beta_i \left| J_i(u) - J_i(u)^* \right|, \quad i = 1, 2, \ldots, k, \quad u \in \Omega \qquad (3)$$

$$J^*_i = \min \{ J_i(u); u \in \Omega \}, \quad i = 1, 2, \ldots, k$$

where J^*_i is the minimum value of the objective function J_i, β_i the preference value for the objective function J_i, δ_s is the combined sum deviation of the multi-objective functions, Ω is the solution space, k is the number of objective functions set by the preference.

The heuristic optimization process performs a hybrid particle swarm optimization (HPSO), described in Appendix B, that searches the given control actions feasibility region for control valves. To evaluate any given set of control actions the HPSO uses an inverse steady state model (ISSMs). In this case, artificial neural networks (ANN) are used to simulate the FFEPU. The ANN process is described in Appendix C.

Set-Points Generation

After the HPSO optimization process delivers a set of normalized control actions that target the given multiple objectives best the last process of the reference governor is to generate the actual set-points that would come out of the FFEPU at steady state. The set-points are obtained by the use of the ISSMs. These ISSM generates the needed set-

points with the input of the twelve control action signals. The ISSM consists of three ANNs and its output is described by (4).

$$E_d = \varphi_{ISSM_E}(u_1, u_2, ..., u_{12})$$

$$P_d = \varphi_{ISSM_P}(u_1, u_2, ..., u_{12}) \quad (4)$$

$$RT_d = ST_d = \varphi_{ISSM_{RT/ST}}(u_1, u_2, ..., u_{12})$$

As previously assumed, since the re-heater and super-heater set-points are very similar, their set-points are set as the same. To implement the reference governor, a power demand curve that represents the daily use of a regular day is presented in Fig. 12. For this curve, the reference governor calculates the solution space ($\Omega_1, \Omega_2, \Omega_{13}, \ldots, \Omega_{12}$) presented in Fig. 13. Finally, the reference governor uses the most optimal set of control action signals found by the HPSO. Fig. 14, Fig. 15, and Fig 16 show the set-points obtained by optimizing the three different combinations preferences for multiple objectives. Case 1 targets only $J_0(u)$, Case 2 targets $J_0(u)$, $J_1(u)$, $J_2(u)$ and Case 3 targets all of the mentioned optimization objectives. The figures compare the power set-points, the pressure set-points and the temperature set-points for each one of the optimizing cases.

As Fig. 14 shows, the set-point trajectories for E_d is nearly the same as E_{uld} and the curves in all three cases is almost the same, since Load Tracking Error is the most important objective in the three cases for multi-objective optimization preferences. This is not the case for pressure demand P_d and temperature demand RT_d / ST_d trajectories.

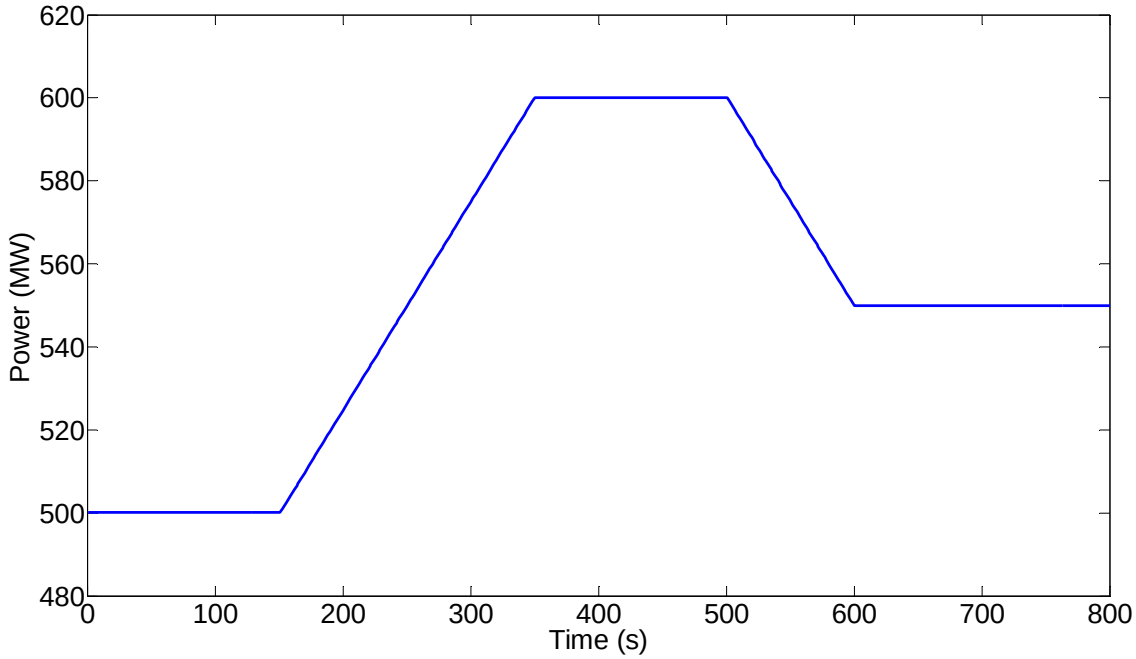

Fig. 12: FFEPU Power Unit Load Demand (E_{uld}) Curve.

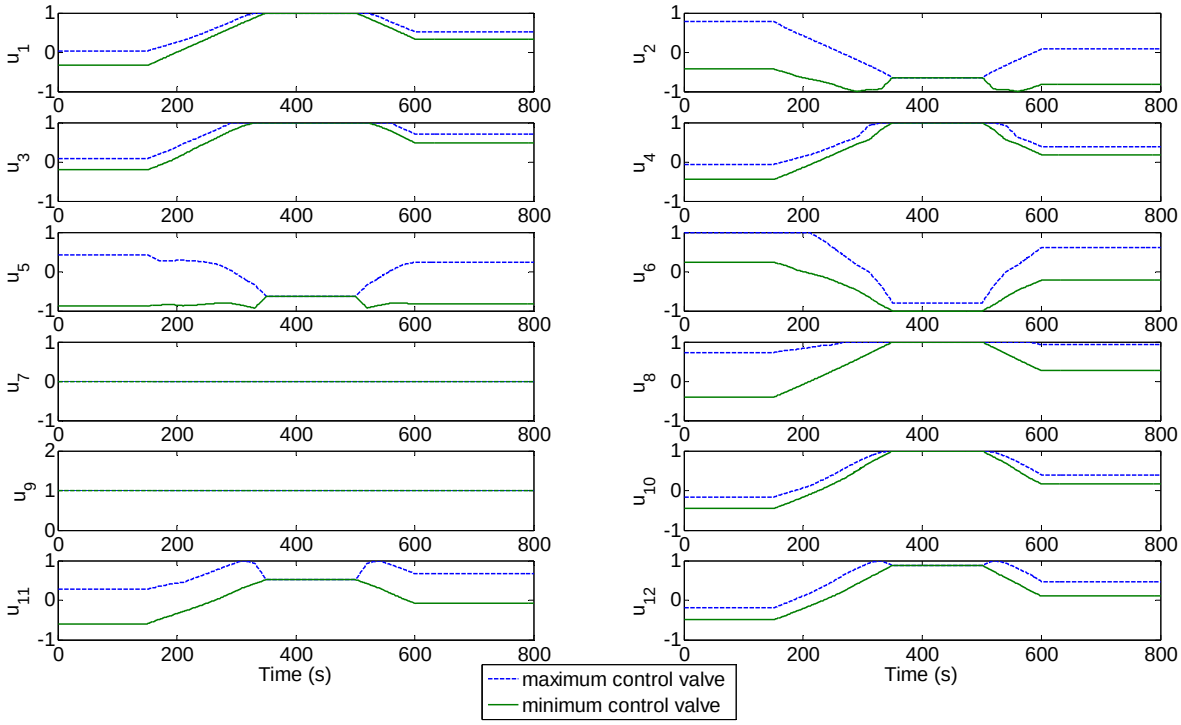

Fig. 13: Solution Space ($\Omega_1, \Omega_2, \Omega_{13}, \ldots, \Omega_{12}$) for given Power Demand Curve.

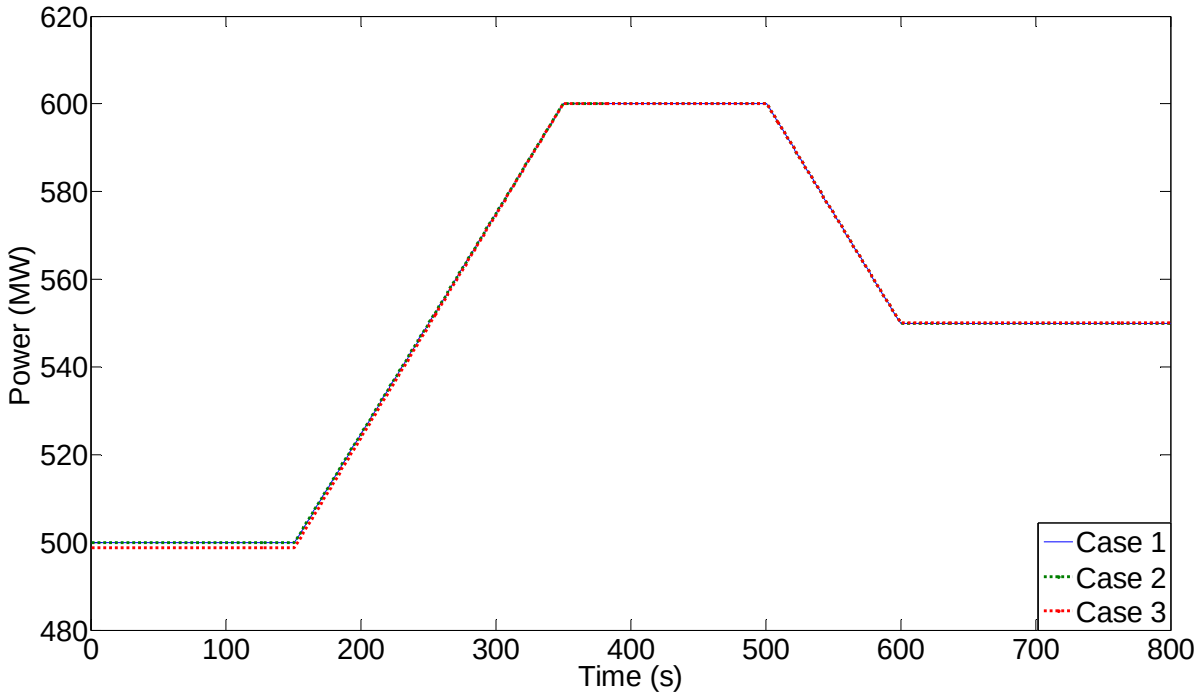

Fig. 14: Power Demand E_d Trajectories for given Power Demand Curve.

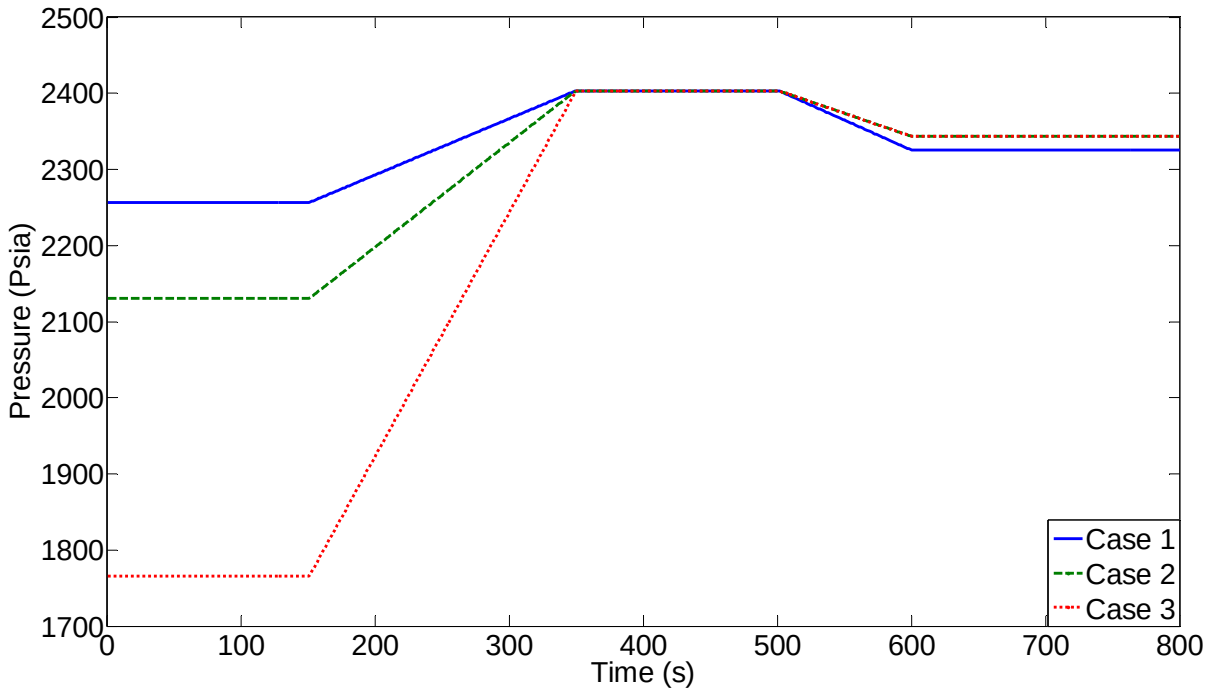

Fig. 15: Pressure Demand P_d Trajectories for given Power Demand Curve.

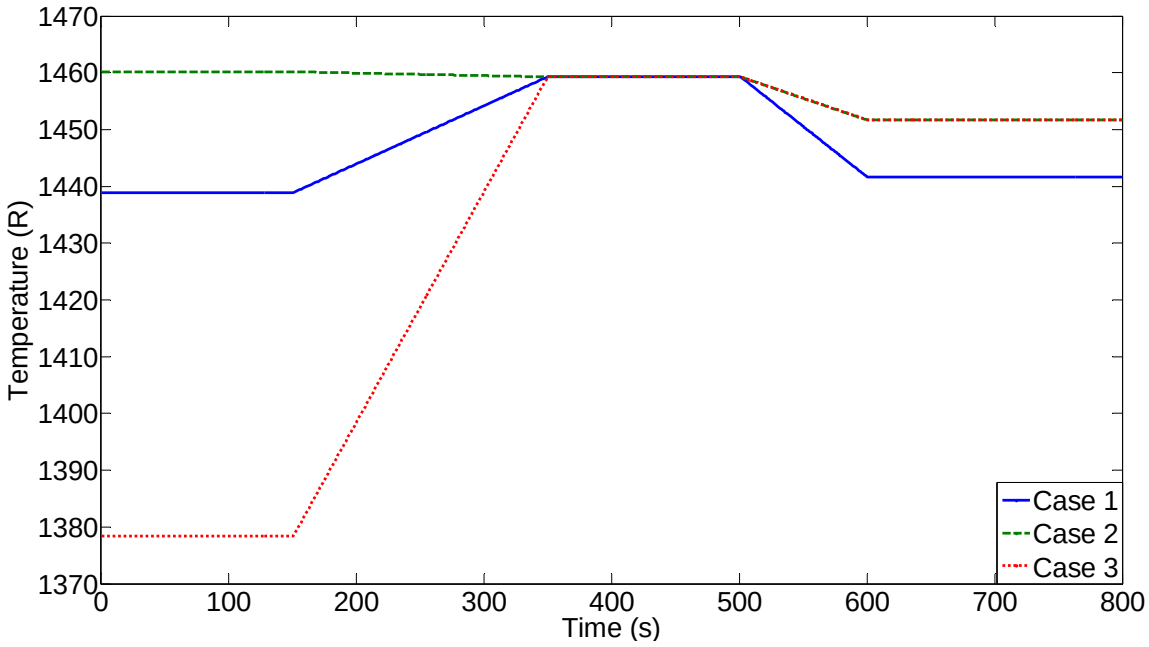

Fig. 16: Temperature Demand RT_d / ST_d Trajectories for given Power Demand Curve.

This is due to that the power-pressure operating window offers a wide range of pressure levels that can generate the same amount of power. Similarly, there is a wide range of temperature set-points that map to the map power set-point curve. In addition, the pressure and temperature trajectories are the result of control action signals that best suit their respective multi-objective preference case.

3.3 *Feed-forward Controllers*

Before the normalized control actions are sent to the FFEPU by the reference governor, they must be scaled back to appropriate control signals so that the FFEPU can use them as valid inputs. As previously mentioned, control actions are normalized so that they can be used by the HPSO, which uses an inverse steady state model (ISSM) based on artificial neural networks (ANN) to evaluate each set of candidate control actions.

Since the ANNs use a tangent-sigmoid function the only inputs acceptable into the ANN are those between -1 and 1.

After scaling the control actions back to its respective range, the feed-forward process uses interpolation to generate a set of control action curves that will be fed into the FFEPU. This is the last step before the feed-forward control action signals $(u_{ff1}, u_{ff2}, \ldots, u_{ff12})$ are added to the feedback control action signals $(u_{fb1}, u_{fb2}, \ldots, u_{fb12})$ and sent to the FFEPU. Fig. 17 illustrates of the Feed-Forward control process. Fig. 18 through Fig 25 shows the un-normalized signals generated by the references governor that are inputs into the feed-forward control process and the un-normalized output control signals that are ready to be fed in the FFEPU.

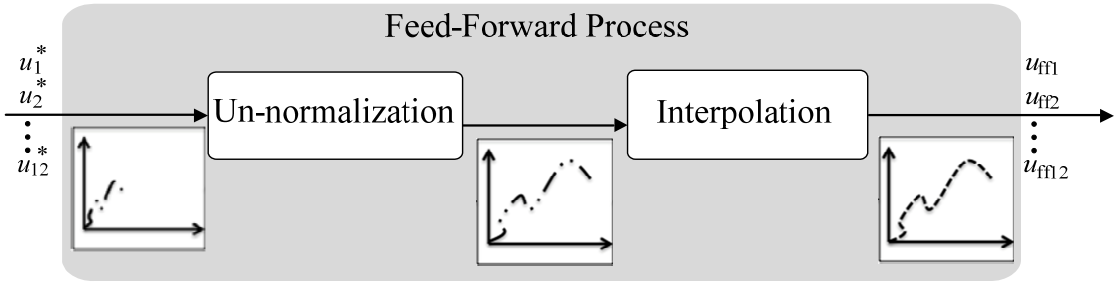

Fig. 17: Feed-Forward Control Process

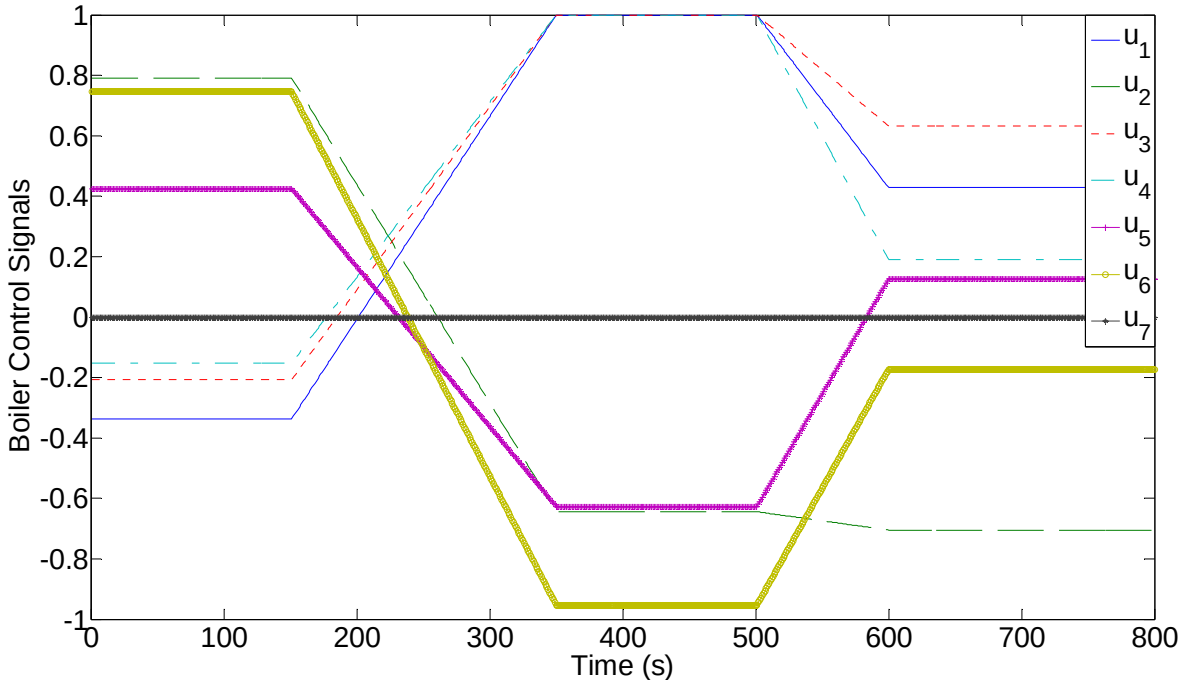

Fig. 18: Boiler Normalized Control Actions

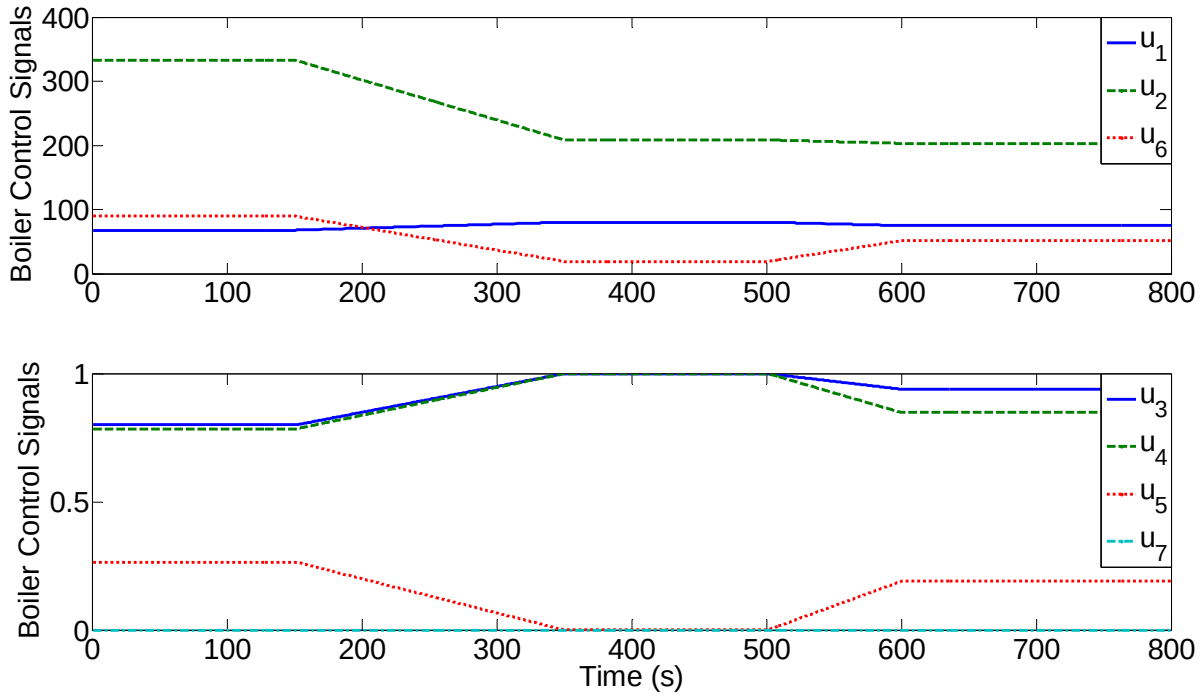

Fig. 19: Boiler Control Valve Action Signals

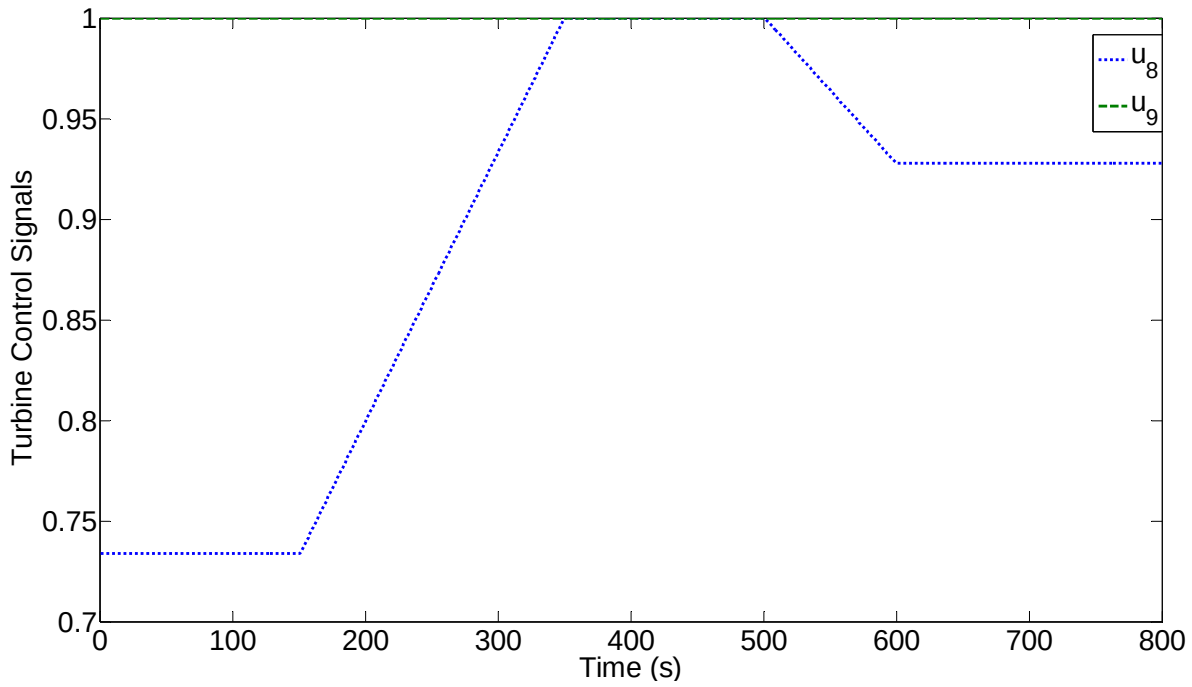

Fig. 20: Turbine Normalized Control Actions

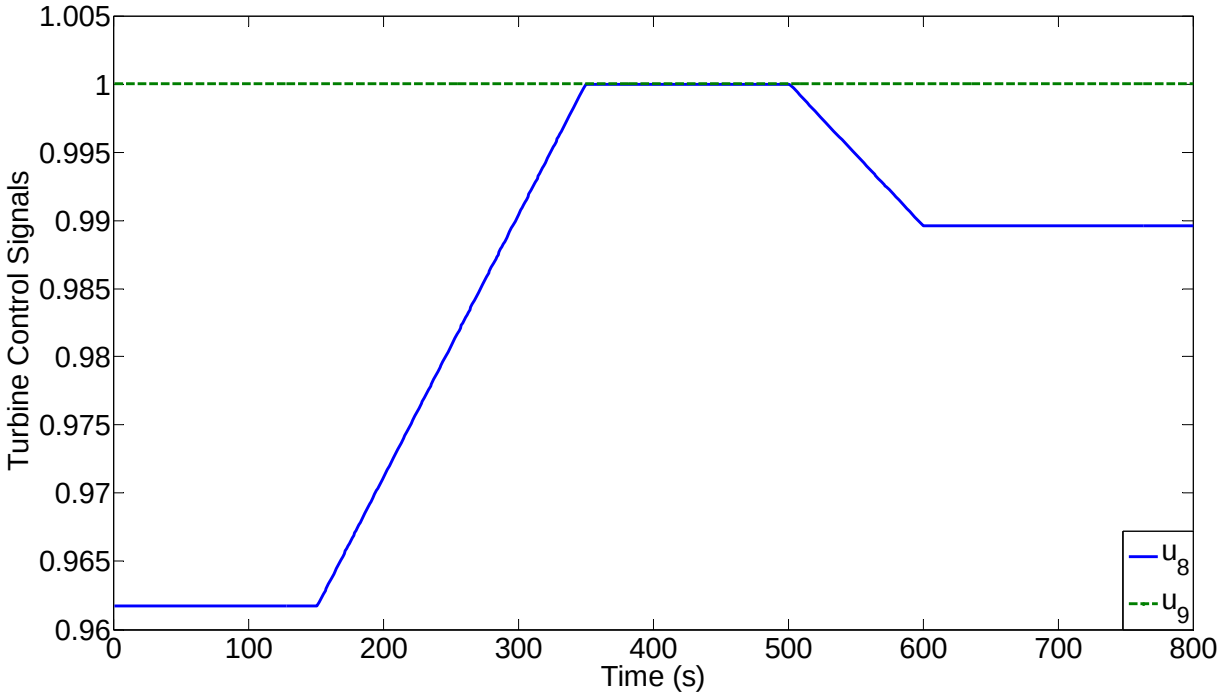

Fig. 21: Turbine Control Valve Action Signals

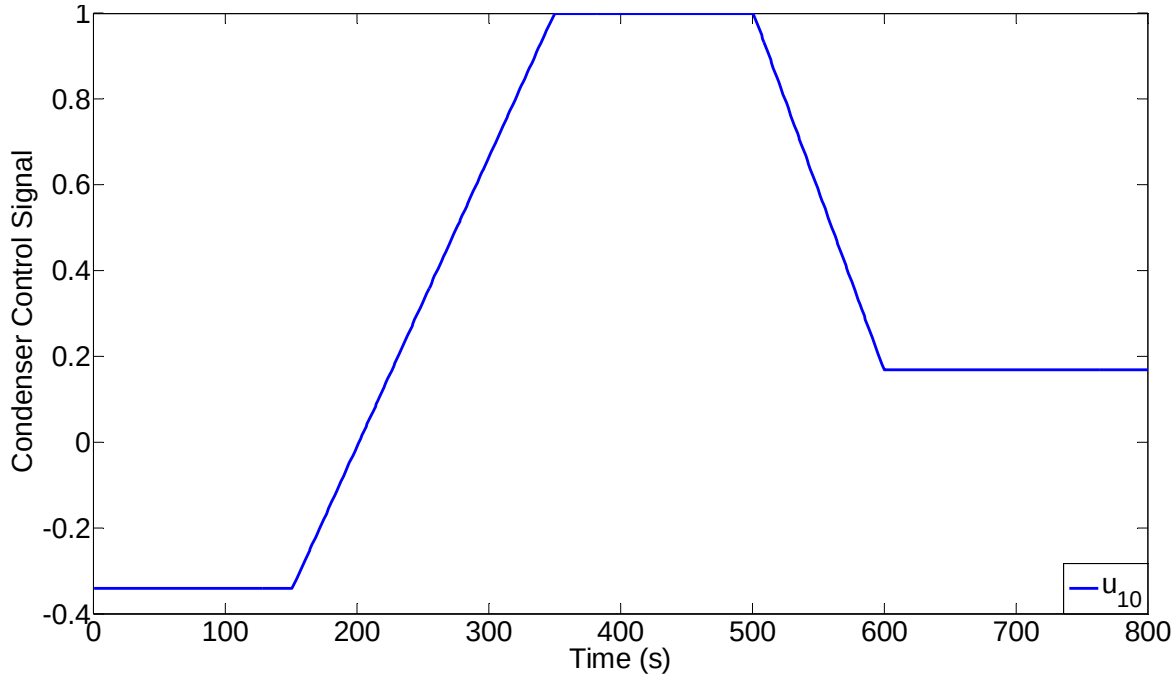

Fig. 22: Condenser Normalized Control Actions

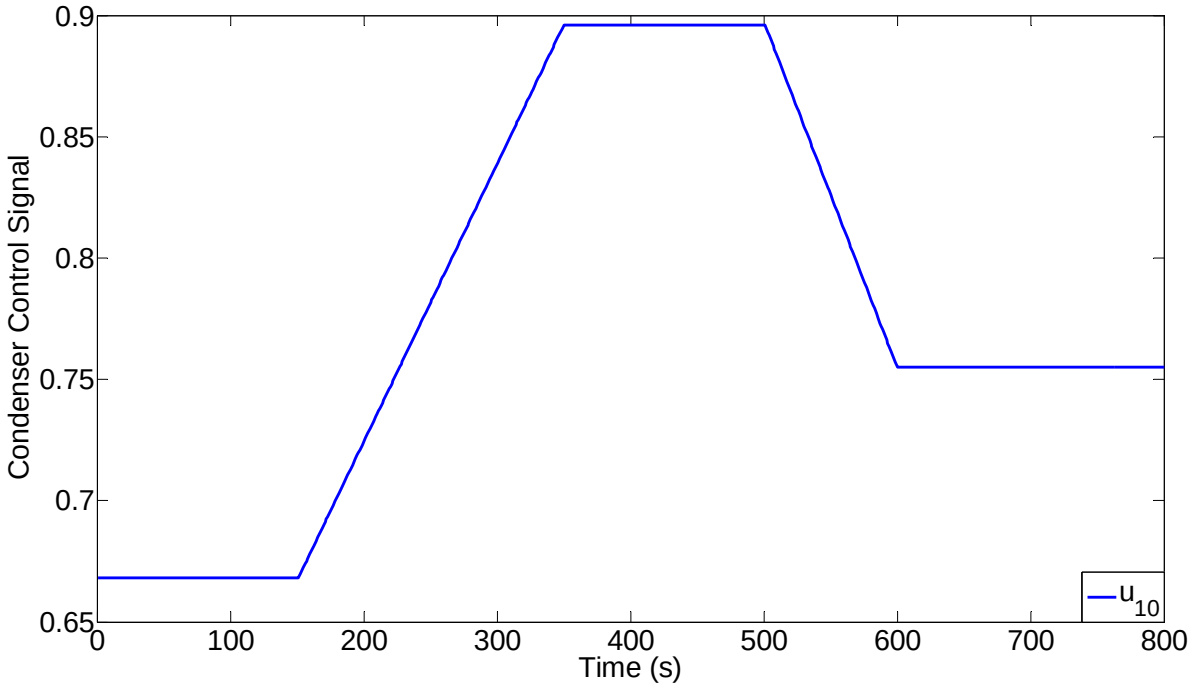

Fig. 23: Condenser Control Valve Action Signals

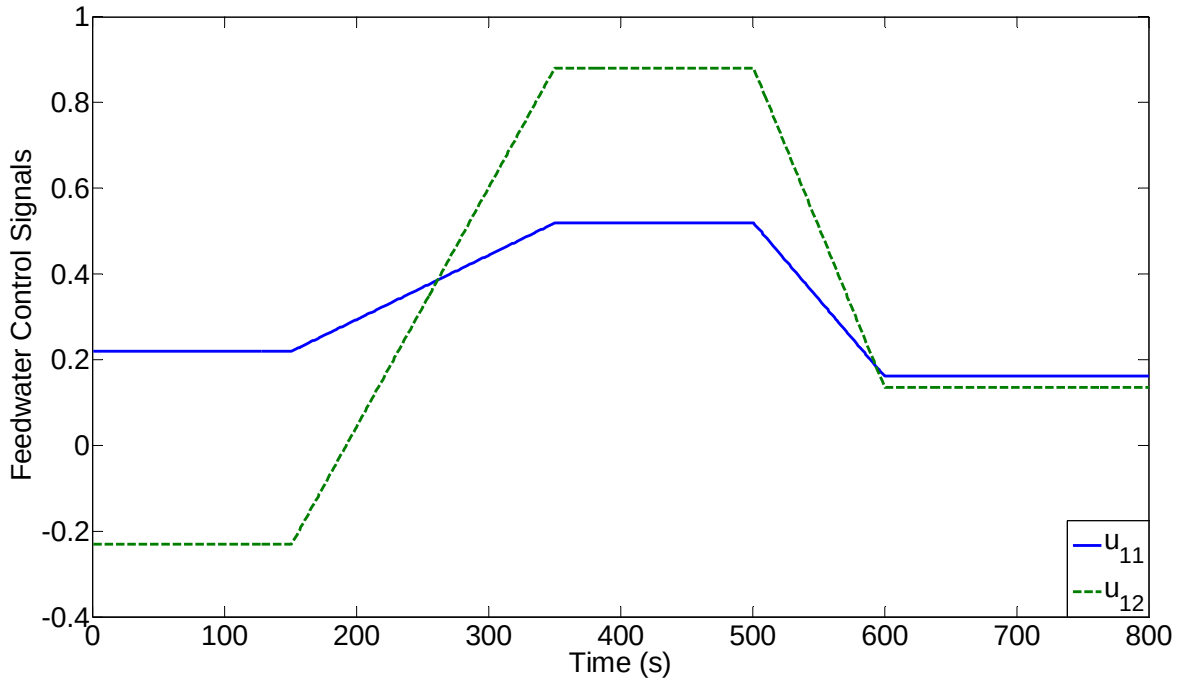

Fig. 24: Feed-water Normalized Control Actions

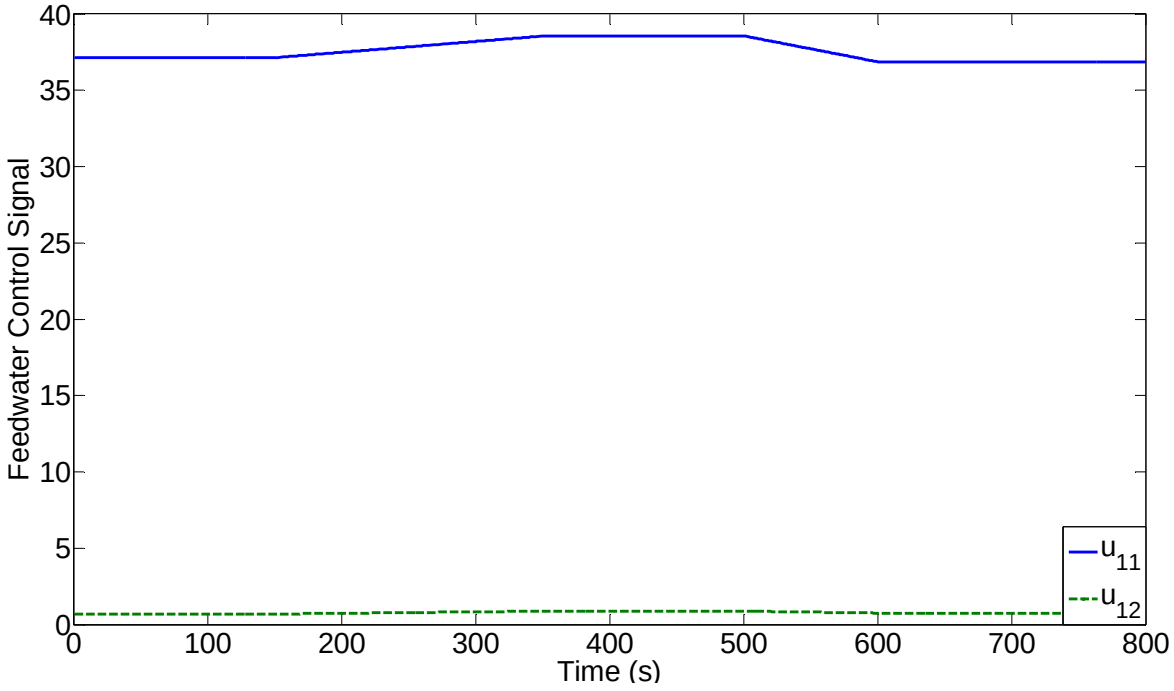

Fig. 25: Feed-water Control Valve Action Signals

CHAPTER FOUR

Feedback Control and Online Gain Optimization

This chapter describes the feedback control implemented for the FFEPU as well as of one of the most important characteristics in the proposed system, the Online Gain Optimization, which is the process by which the PI control gains used by the feedback controllers are tuned to ensure that the power plant response be optimal and dynamic to changes in demand signals. Section 4.1 describes the need for a reliable online gain optimization process that can take into account inefficient feedback PI gains and correct them. Section 4.2 gives an overview and detailed description of the feedback control process for the FFEPU. Section 4.3 gives an overview into heuristic optimization methods and its functionality within the proposed intelligent approach. Section 4.4 describes how the online feedback gain optimization is implemented within the power plant coordinated control and how the heuristic optimization model plays an important role for the system.

4.1 *Motivation*

The use of coordinated control and a reference governor has certainly improved the feed-forward controllers and response of large-scale power plants by providing the best features of boiler-leading and turbine-leading control structures while being able to tackle important multiple objective goals. The use of an intelligent feed-forward control certainly makes the FFEPU control more dynamic and more capable to respond better to changes in preferences by the user.

Most systems implement some feedback control to make sure that the output tracks the desired input as close as possible. Particularly, PI controllers are one of the most used feedback controllers in the industry. They are very popular because not only are they reliable and offer a good performance, but they also are relatively simple and cost-effective [20]. Implementing PI control in different modules of the FFEPU is convenient and efficient in providing with feedback control signals if the gains used for the feedback process generate feedback control actions that minimize steady state error and provides for a fast correction response. Therefore, tuning feedback control gains is important to achieve effectiveness. In order to address this concern, the implementation of a gain optimization process must be established.

Similarly to the implementation of the reference governor process, the online gain optimization procedure uses a heuristic optimization algorithm to find PI control gains that satisfy an accurate FFEPU power response. Unlike the reference governor, which generates set-points and optimized control actions for the FFEPU, the gain optimization process generates new gains for the feedback controllers when their performance becomes poor.

The main task of the gain optimizer is to perform a search among different PI candidate gains, and finding the best set that offers a more accurate FFEPU response for the set-point and control trajectories at the time of implementation. To perform this search, the gain optimizer must evaluate different sets of gains according to a demand curve, which is set by the set-points and control actions generated by the reference governor.

Much like the reference governor, to evaluate these gains, the gain optimization process makes use of a FFEPU model. The main difference with the reference governor is that the model used to evaluate the performance must be a dynamic model to account for FFEPU inputs, outputs, and state variables that play a very important role to find adequate feedback gain values. Adding the online gain optimization feature certainly adds robustness much more to the dynamic FFEPU control. Fig. 26 illustrates the FFEPU coordinated control with the addition of the gain optimization process and the interaction with the rest of the processes.

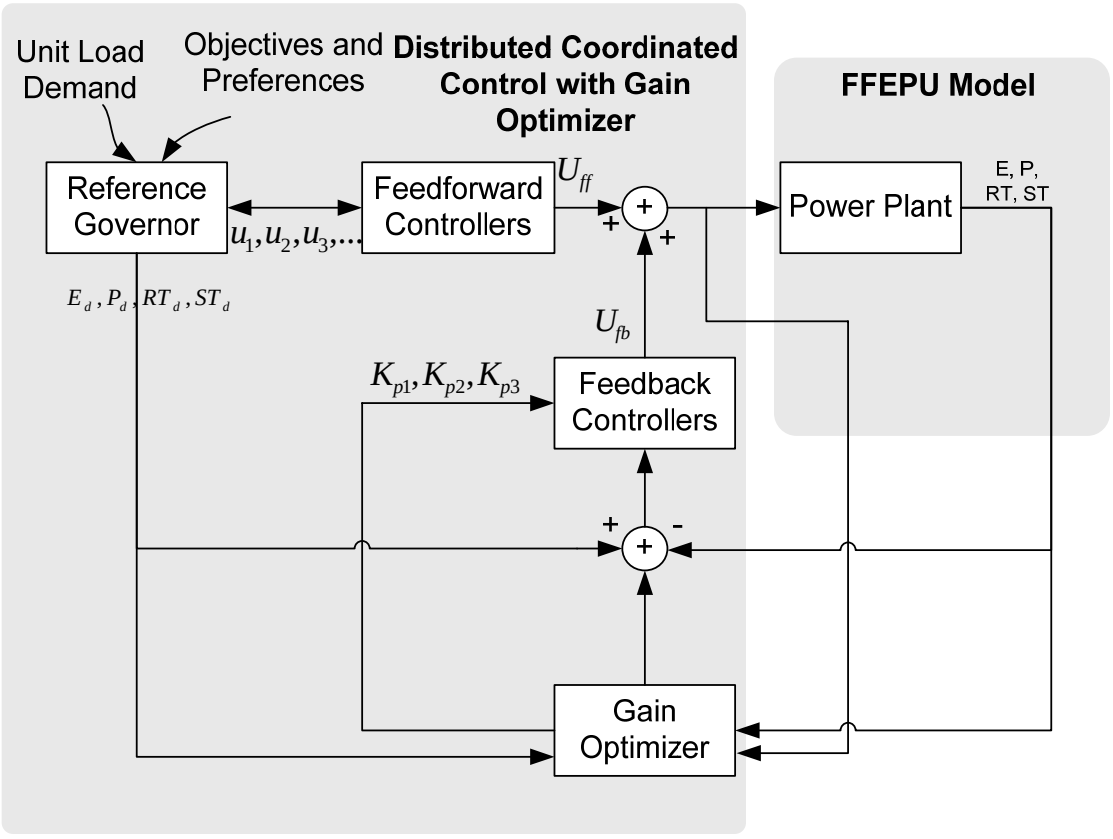

Fig. 26: FFEPU Coordinated Control with Gain Optimization.

The use of an online gain optimization process is presented in this thesis by making us a heuristic optimization algorithm. The approach taken in this thesis uses a

Differential Evolution (DE). DE is an algorithm inspired by biological evolution. It performs a search in a solution space by a population, defined according to a particular mathematical representation of an evolutionary process, and is able to change over different iterations [21].

The decision to use a DE as the heuristic optimization algorithm for the FFEPU is its faster convergence to an optimal solution than other algorithms such as the Particle Swarm Optimization (PSO). This is due to the fact that the DE algorithm requires only one evaluation per particle [21] as opposed the PSO algorithm which requires twice as many. Fast convergence is particularly important for the optimization of feedback gains given that the implementation must be performed online and any time constraining computations may make the gain tuning process ineffective. Therefore, DE is a suitable heuristic optimization algorithm for implementation in an online feedback gain optimization process. The use of a DE as a suitable heuristic optimization algorithm to solve the online gain tuning challenge has been addressed in [12], [6], showing that promising results for the online feedback gain optimization of the FFEPU can be expected.

4.2 *Feedback Control Process*

The main task of the feedback control process is to generate feedback control action signals that, by adding them to the feed-forward control action signals, can compensate for FFEPU output errors and failure to accurately follow the demanded set-points (E_d, P_d, RT_d, ST_d). The feedback control process is part of the overall coordinated control structure and it is divided into four different feedback controllers, one for each of the FFEPU modules. These are the boiler feedback controller, the turbine

feedback controller, the condenser feedback controller and the feed-water feedback controller. The feedback control implemented for the FFEPU is composed of a series of proportional (P) and proportional-integral (PI) control loops.

To implement a feedback controller it is important to bear in mind that the measurements of the different modules and subsystems of the FFEPU come from an actual physical system and in order for them to be processed by the PI controllers they must be transformed into control signal voltages and, likewise, the control signals from the feedback control process must be turned back into a physical signal. For this purpose, two equations represent the linear transducer and the conversion between the physical systems to control signal:

$$c(t) = c_{min} + (c_{max} - c_{min}) \frac{x(t) - x_{min}}{x_{max} - x_{min}} \tag{5}$$

where $c(t)$ is the control signal voltage, $x(t)$ is the time dependent variable, c_{max} and c_{min} are the maximum and minimum voltages for the control signal voltages, x_{max} and x_{min} are the maximum and minimum for the time dependent variable.

Similarly, the transducer conversion back into a time dependent variable from the control signal voltage is dictated by:

$$x(t) = x_{min} + (x_{max} - x_{min}) \frac{c(t) - c_{min}}{c_{max} - c_{min}} \tag{6}$$

In addition to this conversion, another step must be taken to accurately represent the effect of the time delays in an actual physical power plant. The dynamic actuators, which receive control signals from the feedback system, are modeled by actuator simulators with a time delay to account the change in a physical output.

Boiler Feedback Control Process

The boiler module of the FFEPU is controlled by the boiler feedback control process. There are seven control loops: the main steam pressure control, the air flow control, the fuel flow control, the furnace pressure control, the super-heater temperature control, the re-heater temperature control and the gas recirculation control. The first three of these feedback controls make up the combustion control. When the boiler feedback receives the set-points generated by the reference governor, the Boiler Master Demand (BMD) signal is generated from the error between the pressure demand set-point and the actual secondary super-heater pressure and the error between the fixed set-point for turbine speed and the actual turbine speed. Fig. 27 illustrates the process for the BMD signal generation.

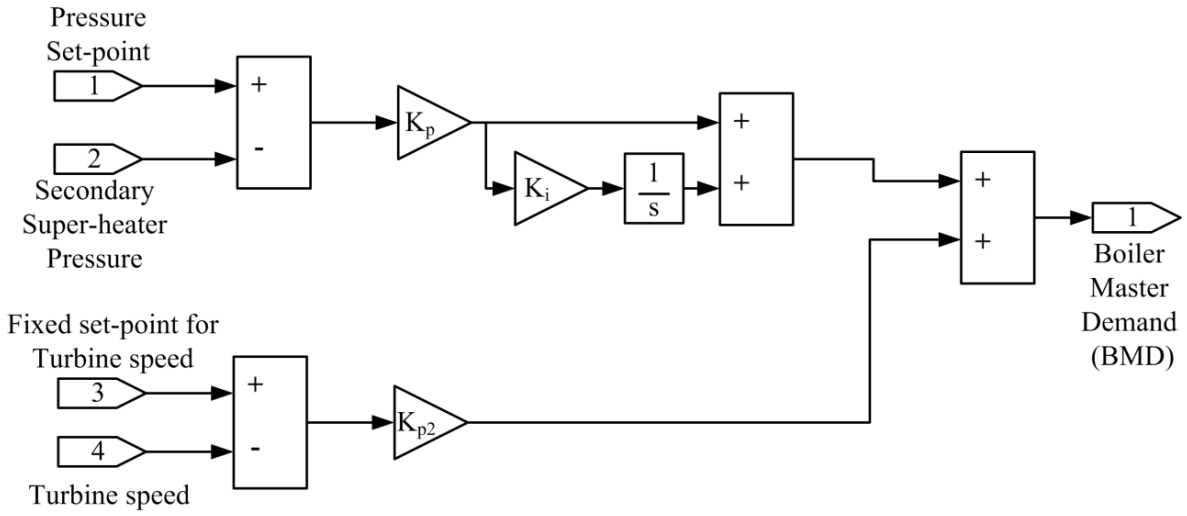

Fig. 27: Process for the Boiler Master Demand [6].

The combustion control process depends mostly on the set-points generated by the reference governor and the actual measured FFEPU outputs. The BMD signal is composed by the sum of the error between the turbine speed and a fixed set-point and the

output of the main steam pressure PI feedback control loop, as illustrated in Fig. 26. The BDM is necessary for the use of the air flow control and fuel control. The fuel flow controller sets the demand for fuel as either the BMD or the total measured air flow, whichever is smaller. The fuel flow control is implemented by a PI controller for the error between BMD and fuel flow. On the air flow control loop side, the BMD signal transmitted to the air flow controller is not allowed to decrease below 25% of its full load value. The air flow controller applies a PI control to the error between air flow demand and measured air flow. The main steam pressure control loop is illustrated in Fig. 28. Fig. 29 illustrates the feedback fuel flow control loop and Fig. 30 illustrates the feedback air flow control loop.

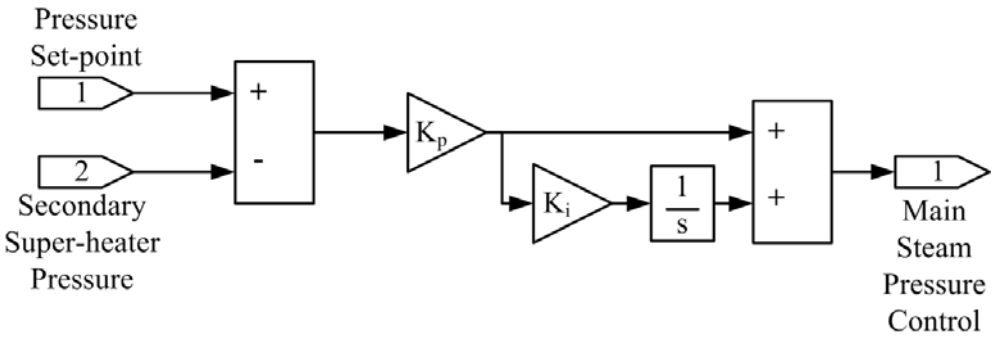

Fig. 28: Main Steam Pressure Control Loop [6].

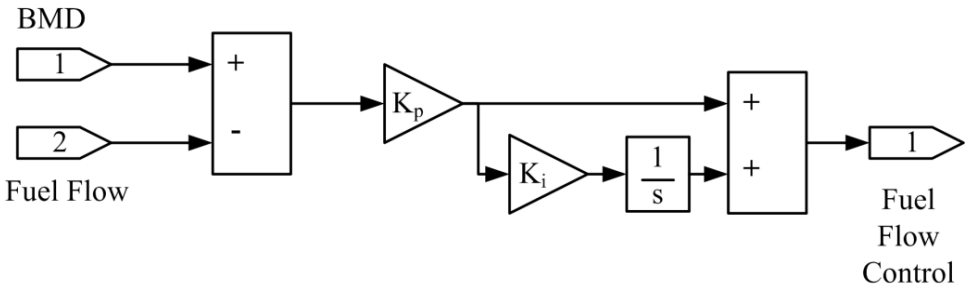

Fig. 29: Fuel Flow Control Loop [6].

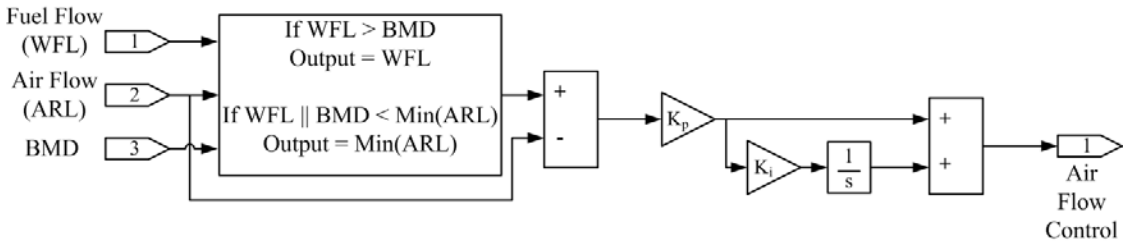

Fig. 30: Air Flow Control Loop [6].

The furnace pressure controller is a PI control on the error between a fixed furnace pressure set-point and the measured pressure, as illustrated in Fig. 31. In addition, measured air flow is included to generate a control signal. The output control signal is converted in induced draft fan inlet vane position that controls furnace pressure.

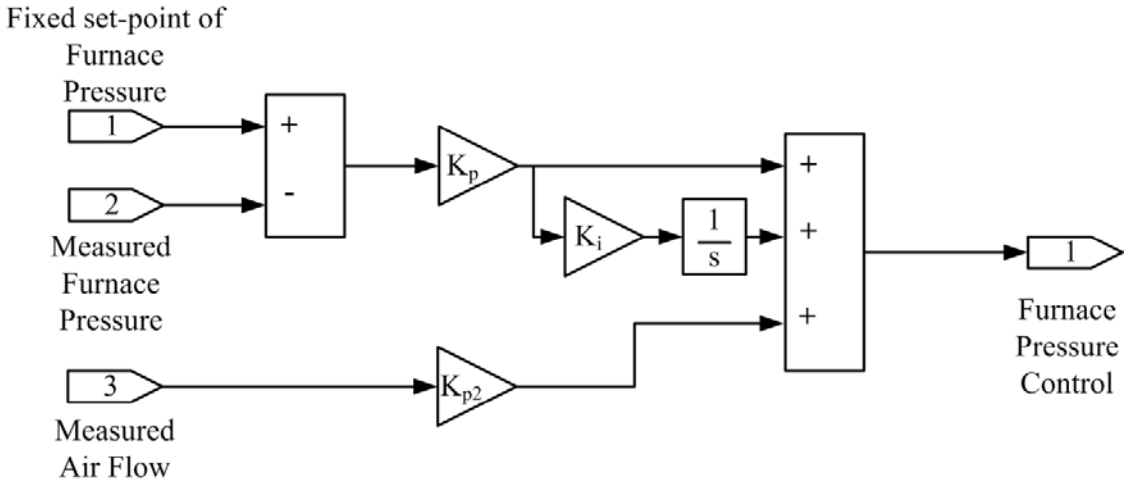

Fig. 31: Furnace Pressure Control Loop [6].

The super-heater temperature controller is a PI control applied on the error between the secondary super-heater set-point and the actual measured super-heater temperature in the FFEPU. Added to the control signals are also proportional control to the change in the first stage pressure and change in the burner gun tilt position. The transducer transforms the output control signal into the super-heater spray flow feedback

which maintains steam temperature exiting the secondary super-heater. Fig. 32 illustrates the super-heater temperature control loop.

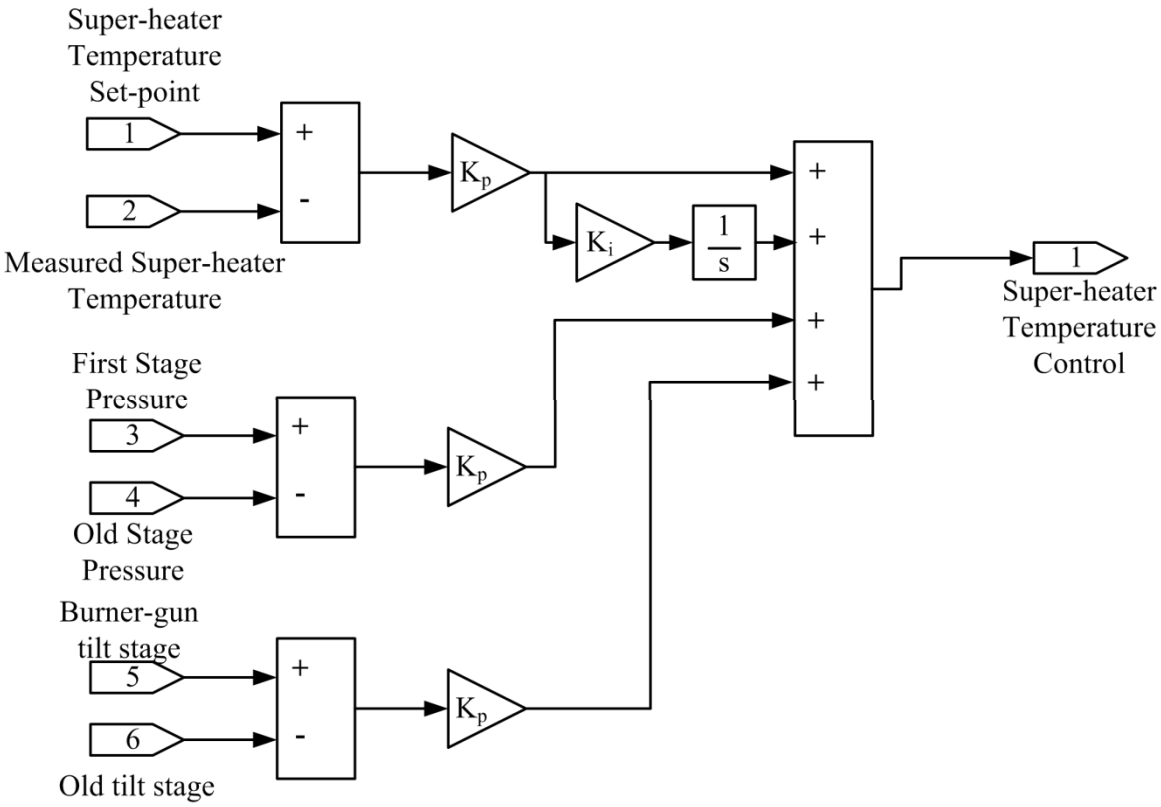

Fig. 32: Super-heater Temperature Control Loop [6].

The re-heater temperature controller aims to maintain re-heater temperature and consists of three different control schemes using the re-heater spray, burner-tilt position and gas recirculation. The re-heater temperature uses a PI control applied to the re-heater temperature set-point and the measure re-heater temperature. In addition, the change of the first stage pressure and the change in measured re-heater temperature are added with a proportional gain. The control signal is sent to two actuator simulators, one of them associated with the delay by the spray valve actuator and the other one associated with the delay by the tilt position actuator. The output control signals are converted into the

burner tilt position and the re-heater spray flow. Fig. 33 illustrates the re-heater spray control loop.

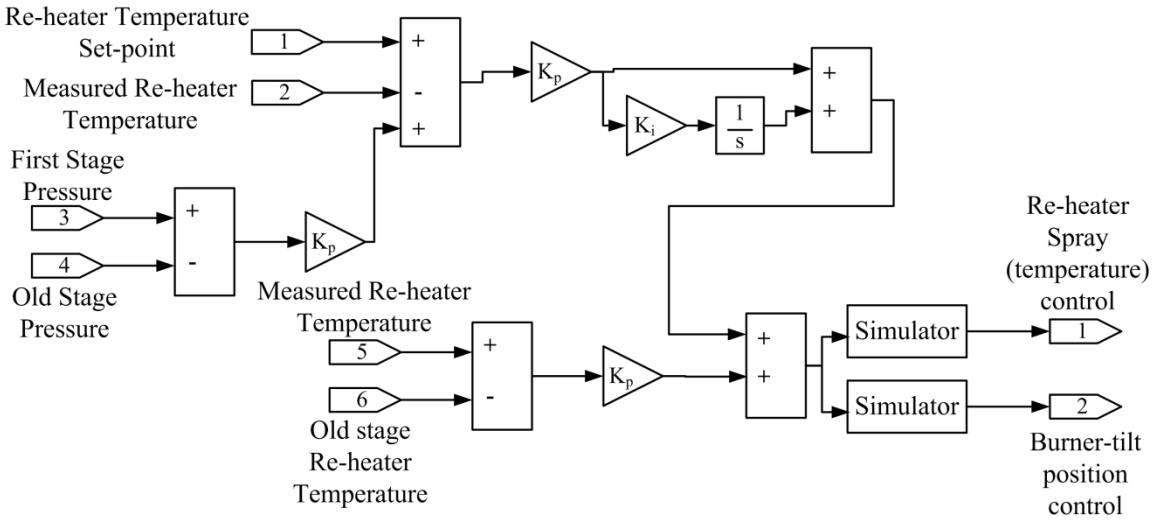

Fig. 33: Re-heater Temperature Control Loop [6].

The gas recirculation flow controller output is determined only by the burner tilt position. The controller uses a track-hold integrator that applies integral control to the error obtained between the burner tilt position and a fixed tilt position. The output of integrator is limited by the measured air flow. If the burner tilt is varied more than $\pm 5°$ from midrange, the gas recirculation flow is changed. The resulting feedback control signal is converted in gas recirculation flow, which is taken from the flow gases at the exit stage of the economizer and let into the furnace. Fig. 34 illustrates the gas recirculation flow control process.

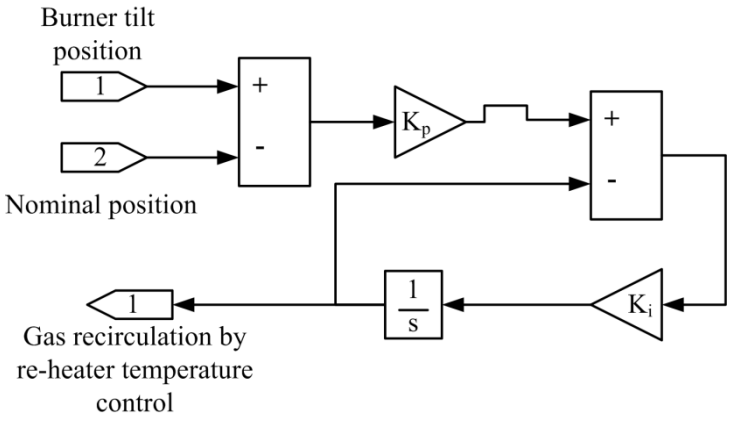

Fig. 34: Gas Recirculation Control Loop [6].

Turbine Feedback Control Process

The main task of the turbine feedback control process is to generate two feedback control actions: the feedback governor control valve and the feedback intercept valve. For this purpose, the Electro-Hydraulic Control (EHC) system, which is basically the turbine feedback controller, is divided into three different areas: the speed control unit, the load control unit and valve control unit.

The speed control unit is in charge of maintaining the turbine speed at its set-point at 60Hz. It does so by using a proportional control applied to the error between the turbine set-point speed and the measured speed. The obtained control signal is then used by the load control unit.

The load control unit uses the output error from the speed control unit and adds it to the Load Demand Computer (LDC), which is obtained from the power unit load demand.

Subsequently, a PI control is applied on the error between the measured power generated and the output signal of the load control unit. The output of this PI control signal is the Desired Load Reference (DLR) signal. The output of the load reference

motor controller, also known as the load reference signal, is compared with DLR and the signal is used as a feedback signal back into the load reference motor. Fig. 35 illustrates the process of the turbine feedback process.

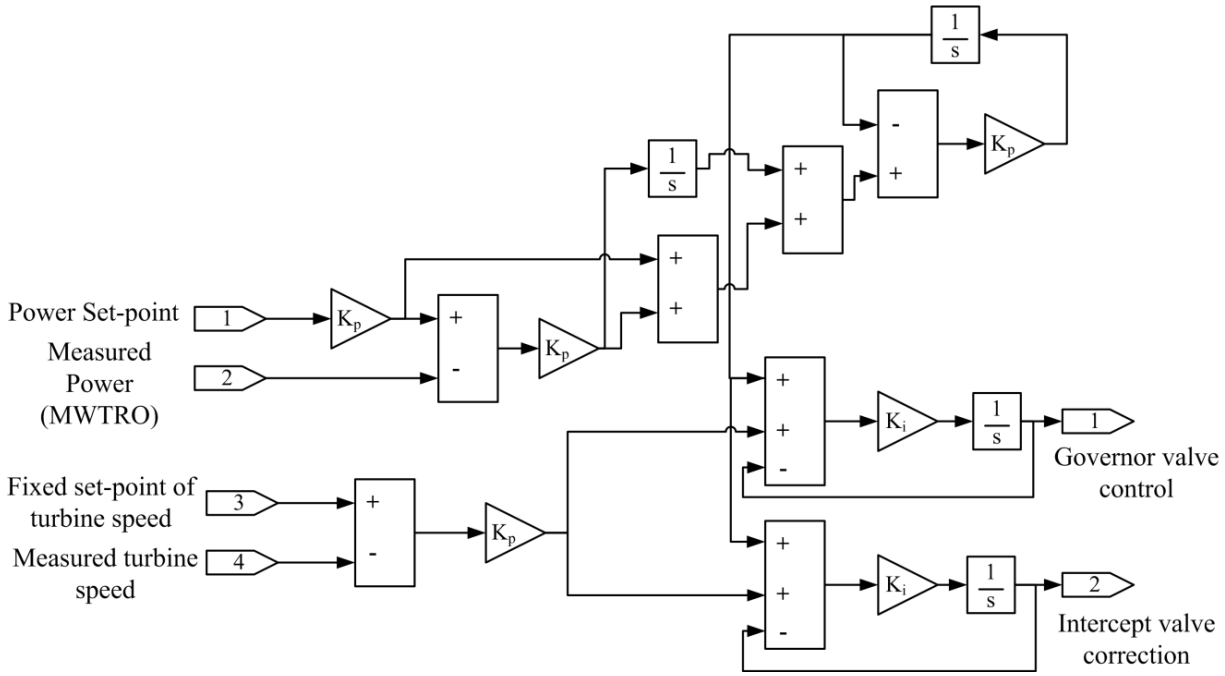

Fig. 35: Turbine Feedback Control Loop [6].

The valve control unit contains two sets of valves: the intercept valve, which is located before the input of steam of the intermediate pressure turbine and the governor valve, which is located before the steam input of the secondary super-heater. The control output signals are then converted into the governor control valve and the intercept control valve.

Condenser Feedback Control Process

The condenser feedback controller uses the steam flow to generate feedback control actions. Steam flow is represented by the measurement of the turbine re-heat

pressure, the measurement of the condensate flow, and the deaerator water level. Fig. 36 illustrates the condenser feedback control loop.

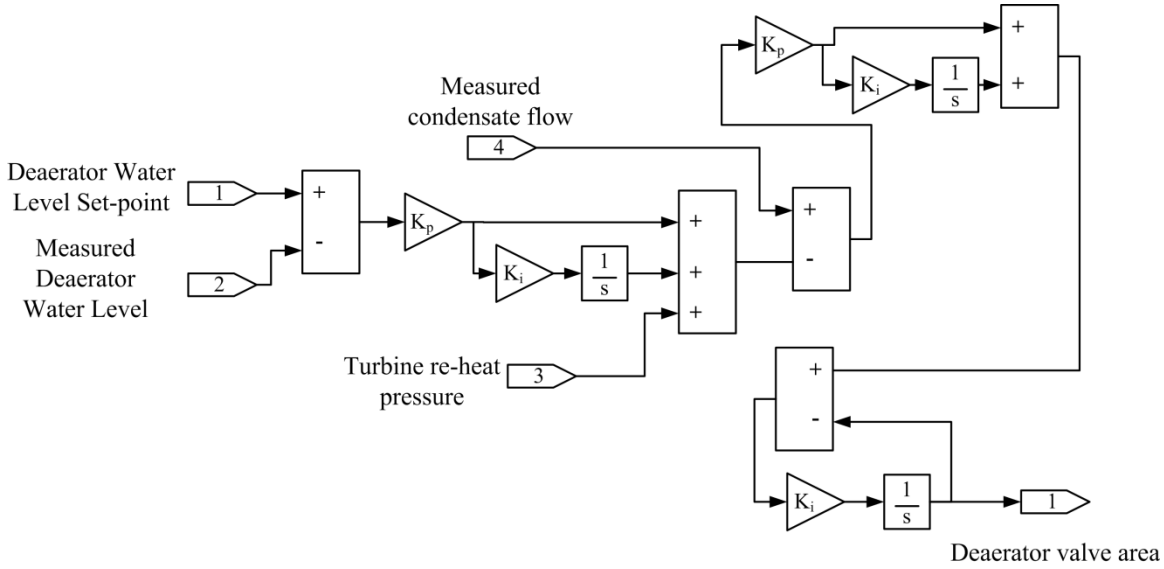

Fig. 36: Condenser Feedback Control Loop [6].

The condenser feedback controller uses PI controls on the error between the water level set-point and the deaerator measured level. The condensate flow demand is then obtained by adding the output of the PI controls to the steam flow signal. Another PI control is applied on the error between the condensate flow demand and the measured condensate flow. The output feedback control signal is converted by the traducer into a deaerator control area that directly controls the condensate flow.

Feed-water Feedback Control Process

The feed-water feedback controllers present three different stages: the first stage pressure, which is used as a measurement of the main steam flow, the feedwater flow, which is composed by the measured feed-water flow, super-heater and re-heat spray flows, and the drum level. The feed-water feedback control process generates the

feedback feedwater valve control signal and the feedback feed pump turbine flow control signal.

The drum unit consists of a PI control that is applied on the error between the measured drum level and a drum level fixed set-point, which will maintain the drum level at a desired value. To obtain the feedwater flow demand, the main steam flow signal is added to the output of the drum level unit. Another PI control stage takes the feedwater flow demand and compares it to the total feedwater flow and the output is sent to the actuator simulator. The output control signal is then transformed into the feedwater valve area, which controls the feedwater flow rate from the feed-pump.

The feed-pump turbine controller generates the feedback steam flow control and maintains the feedwater valve pressure by varying the boiler feed-pump speed. The feed-pump controller consists of a PI control applied to the error between measured pressure drop and the required pressure drop. The output of this PI control process goes into an actuator simulator and the output control signal is converted by the transducer into extraction steam flow that supplies the feed pump turbine. Fig. 37 illustrates the feed-water feedback control process.

4.3 *Online Feedback Gain Optimization*

The implementation of all of these PI feedback controllers certainly is an effective control configuration that leads to accurate FFEPU response to the given set-points, given that the implemented gains are suitable at the time of operation. On the other hand, if gains are not adequate for the particular power unit load demand, the outcome can display a very poor tracking performance.

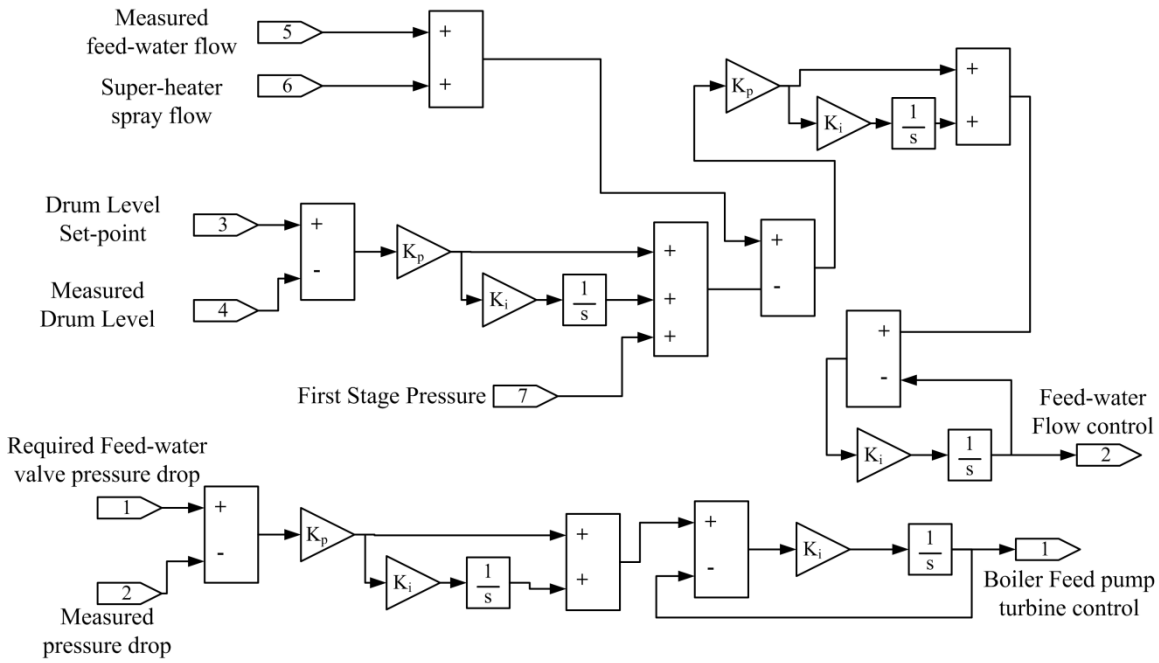

Fig. 37: Feed-water Feedback Control Loop [6].

For this reason, an online gain optimization process must be established to overcome the challenges of tuning feedback gains optimally according to changes in the power unit load demand at the time of implementation. To this end, an effective gain optimization is required to target the four different feedback control processes.

Similarly to the feedback control process, which is divided in four different processes to address the FFEPU four different modules, the gain optimization process is also divided in four different processes to target each one of the feedback control processes. Therefore, the boiler feedback gain optimizer will target the boiler feedback controller and, therefore, affecting the feedback control action signals associated with the boiler feedback modules. The case is the same for the other three modules.

As seen previously in the reference governor, in order to perform the optimization task effectively, it is necessary to establish an efficient heuristic optimization algorithm.

For the reference governor, a Hybrid Particle Swarm Optimization presents a good performance at establishing suitable feed-forward control actions. A more detailed study [19] of the advantages of certain heuristic optimization algorithms over others suggests that a Differential Evolution (DE) is more efficient at optimally finding suitable feedback gains for an online optimization process. This is primarily due to the implementation time superiority of the DE over other algorithms such as the PSO. Specifically, the DE only needs one evaluation per iteration, as opposed to two evaluations per iteration which is the case for the PSO.

Before implementing the gain optimization process, it is necessary to be acquainted with the solution space, or also known as the space of feedback gains that the optimization search will target, and the range for these gains.

PI Controller Gains

The all the PI controllers being implemented by the feedback control processes share basically the same form and Fig. 38 illustrates the PI controller form.

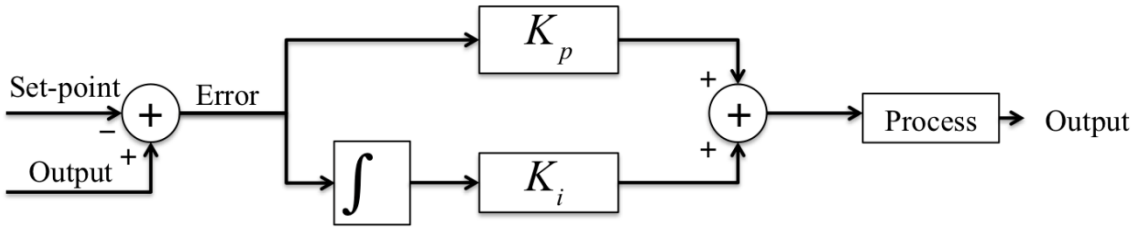

Fig. 38: PI Controller Loop [6].

From Fig. 38 the equation to determine the output as a function of the input error can be simply deduce to the standard form of a proportional and integral gains, as shown below in (8):

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau \right) \tag{8}$$

Where $e(t)$ is the input control error as a function of time, $u(t)$ is the control signal as a function of time, K_p is the proportional gain and T_i is the time step for the feedback controller. The relationship between the integral gain K_i and proportional gain K_p can then be simplified to (9):

$$K_i = \frac{K_p}{T_i}. \tag{9}$$

The relationship in (10) is very useful, since it allows the optimization to target a single value instead of two for the PI controllers, leading to a less computational intensive and faster search. To consider each specific agent, first the target gains must be identified. Table VIII displays the tunable gains within their respective module and range of operation.

Table VIII: FFEPU Tunable Control Gains

Control Loop	PI Control Gain	Lower Range	Upper Range
Boiler:			
Air Flow	KC1AR	2.00	4.00
Fuel Flow	KC1FL	2.00	4.00
Furnace Pressure	KFCFN	-2.25	-0.25
Re-heater Temperature	KC1RH	4.00	15.00
Super-heater Temperature	KC1SY	8.00	15.00
Turbine:			
Load Control	KC1TR	0.00	11.00
Condenser:			
Condensate Control	KC1DV	8.00	12.00
Feed-water:			
Boiler Feed Pump	KC1FT	0.00	2.00
Feed-water Flow	KC1FV	1.00	4.00

To show the functionality of the online gain optimization, Fig. 39 shows an example for a curve that could have been taken for an online performance. Note that the first one hundred seconds of simulation are just settling time and no values are considered for the performance cost. The actual performance cost is evaluated at the last twenty-five seconds of the optimizing input, which in the case of online optimization, this would be the portion of the power curve that is generated by the reference governor and is taken as an input to be followed. In addition to this input, performance cost for the boiler feedback gain optimization and the turbine feedback gain optimization are shown in Fig. 40 and Fig. 41, respectively. Since the performance costs for the condenser and feedwater gain optimization do not take into account any set-points but only the feedback control actions of its own module, they will be ignored.

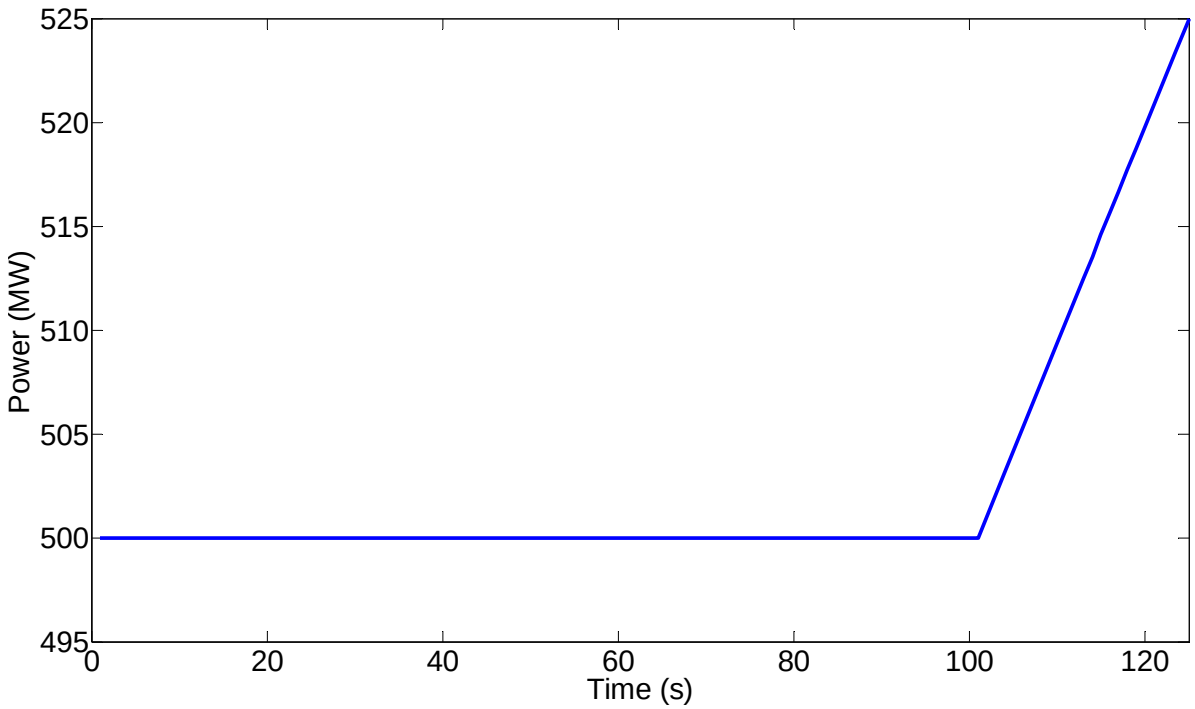

Fig. 39: Gain Optimization Input.

Boiler Gain Optimization Process

As seen in Table VIII, there are five tunable gains that correspond to the feedback control process of the boiler module. As previously stated in (9), since there is a relationship between the proportion gain K_p and K_i, the search does not focus on two different parameters but gains that represent the proportional control. In the case of the boiler feedback tunable gains, they were chosen since they have a direct effect on the set-points for the FFEPU.

Given the gain parameter and the range for the search, a solution space has been established and the heuristic optimization algorithm can start optimizing. As previously discussed, the DE algorithm is used to implement this search. For gain optimization, the cost function is determined by the sum-squared error of error between the pressure demand, the super-heater/re-heater temperature demand, and the size of the feedback control action signals for the boiler module themselves. The cost function aims to minimize all of these, and the best set of tunable gains is considered to be sent to the feedback control process. If the gain optimizing implementation is more accurate than the set of gains currently being used to the feedback control process of the FFEPU, then they are replaced for new optimized gains, if not the gains remain unchanged.

Turbine Gain Optimization Process

In the turbine feedback gain optimization process there only one candidate gain which is the Load Control (KC1TR). For the turbine gain optimizer, the performance function is determined by the sum squared error for the power demand, pressure demand, and temperature demand. Similarly, the values of the feedback control signals are also included in the performance cost.

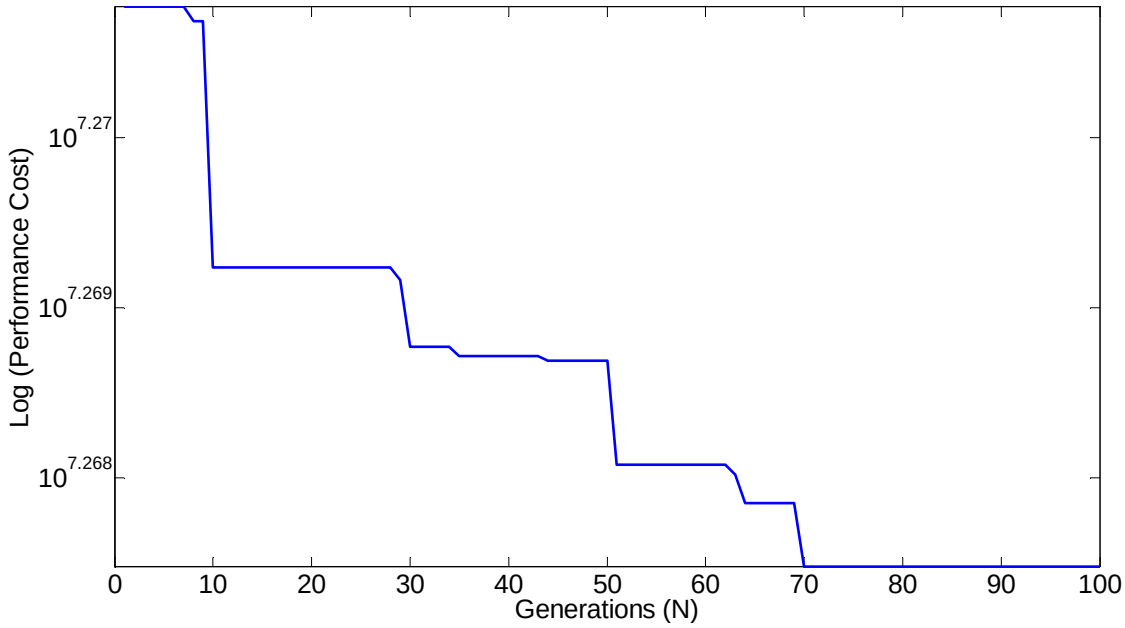

Fig. 40: Boiler Gain Optimization Performance Cost.

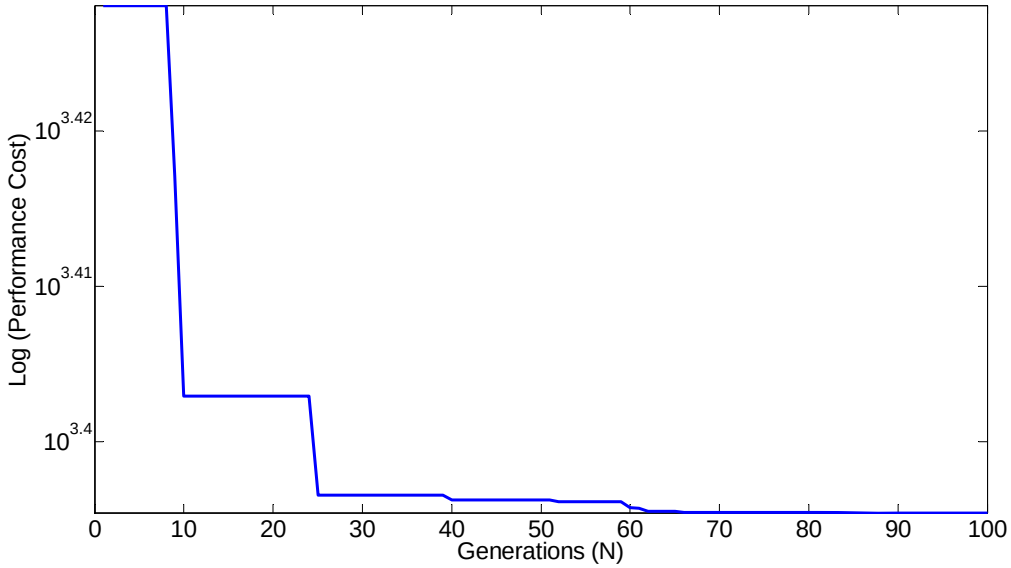

Fig. 41: Turbine Gain Optimization Performance Cost

Condenser Gain Optimization Process

Like the turbine feedback gain optimization, the condenser counterpart is in charge of optimizing only one tunable gain in the condenser feedback process. This

tunable gain is the condensate control (KC1DV) and it addresses the regulation of the error between the set-point water level and the deaerator measured level. Unlike in the previous two gain optimization processes, the condenser gain optimizing process has a different performance function. Since the condenser feedback process does not directly involve any of the set-points generated by the reference governor, these can not be included in the performance function, therefore the performance function is determined only by the squared sum of the condenser feedback control action signals.

Feedwater Gain Optimization Process

The feedwater gain optimization process is similar to the condenser gain optimization process, since the feedwater feedback process does not account for any set-points. Therefore, as in the condenser case, the performance function is to minimize the squared sum of the feedback control action signals. The tunable gains in the feedwater are two: the boiler feed pump (KC1FT) and the feed water flow (KC1FV).

CHAPTER FIVE

Multi-Agent Systems and JADE

In this chapter all of the processes of the distributed coordinated control for the FFEPU previously described are designed for and implemented by Multi-Agent Systems. Section 5.1 provides the motivation for the use of a MAS framework to implement the FFEPU distributed coordinated control. Section 5.2 presents an overview of MAS along with some basic definitions and an overview into the single agent as an entity of a larger control goal and its architecture. Section 5.3 establishes a hierarchy for the FFEPU coordinated control and describes the MAS communication protocol requirements. Section 5.4 presents the Java Agent Development (JADE) framework and states the motivation for its implementation in this particular coordinated control system. Section 5.5 introduces the JADE-based MAS implementation, providing details about the agent structure, the type of agents and their particular tasks and the manner in which the agents communicate to achieve their given task.

5.1 Motivation

As described in the previous chapters, achieving the goal of coordinated control can become complex and computationally burdensome. The FFEPU coordinated control seeks to provide for intelligent feed-forward control by means of a reference governor and at the same time it aims to provide a dynamic feedback control process that requires the use of a gain optimizer process, which must be implemented online to be effective for the FFEPU system.

To maintain the reliability of the distributed coordinated control system, which is highly coupled and interconnected with subsystems, the control system should be operated with new operation requirements. In a large-scale distributed control system different processes must operate in parallel since all systems must communicate in real time and work simultaneously.

It is necessary for such a complex large-scale system to operate at a higher level of automation, being capable of taking operation decisions among different control processes. Control is expected to be flexible with the capability to incorporate more tasks and more processes to the control setup, be robust and fault tolerant with a distributed intelligent control that works independently of other processes, and exhibit a graceful degradation when there are failures in different systems.

Additionally, the response of the FFEPU to the distributed coordinated control framework must become more accurate, increase efficiency and reliability with features between the processes that are capable of cooperation, negotiation and competition.

The proposed approach is the Multi-Agent Systems (MAS) framework to tackle on the challenges of such a large-scale control system. MAS is a framework that is composed of different agents, which are software programs that run in a given environment and are autonomous in nature. This characteristic encourages MAS cooperation and flexibility through the system, since not a single agent possesses knowledge of the overall goal and objectives of the system.

MAS for a large-scale FFEPU exhibit proactive, reactive and robust properties that improve the control of complex power plant systems. There has been plenty of work

done on power engineering applications supported by MAS such as in [8], [10], [12], [22] and more specifically in the MAS for control systems [22], [23], [24].

JADE is a software framework, implemented in Java, designed to develop agents efficiently and to comply with the standards of the Foundation of Intelligent Physical Agent (FIPA) for agent development. JADE exhibits the ability to add and develop agents with small effect in the overall MAS architecture. Among many others, JADE presents features that facilitate the MAS implementation, exploits the advantage of asynchronous messaging protocol, and poses very little overhead structure to implement new agents as needed by the system. The next few sections describe the MAS process, JADE as an agent developing software, and the implementation of the JADE-based MAS for the FFEPU control.

5.2 MAS Definition and the Intelligent Agent

A Multi-Agent System definition can be subject to a wide range of opinions as there is no universal consensus on the matter. Nevertheless, the most accepted feature about the agent is that it is a software-implemented system within an environment and has the ability to exert an autonomous action in the given environment to meet its design objectives [11], [26].

MAS Background and Principles

The main goal of the MAS control approach is that it allows the interaction between agents to improve [19]. The MAS is autonomous, which means that the agents are independent, make their own decisions, and provides for an environment for the agents to be situated in. MAS tend to be used where environment can be challenging and

can be dynamic, which means that the environment may change rapidly; unpredictable, which means that oftentimes the agents may not know complete information about their environment; and unreliable, which means that failure in the system may be beyond an agent's control [11]. For these reasons, the agent must be prepared to face these challenges and account for them.

Single Agent

As previously defined, an agent is defined as a computer software program in a distributed environment that acts autonomously to meet its design objectives. In order to meet these basic requirements, there are some fundamental properties that an agent must exhibit.

Since the environment for the MAS may be dynamic and change rapidly, the agents must be reactive. Reactivity is exhibited in an agent if the agent is capable of responding to perceived changes in its environment in a timely manner and aims to meet its designed goal.

The agent should not only behave to perceived changes in its environment but it must take into account its own goals. A proactive agent is persistent in pursuing its objectives even if failure occurs or if the MAS environment continuously changes over time. It is important to maintain a balance between the reactivity and proactive property of an agent. If the agent is designed to be too reactive, the changes in environment may be the main factor influencing the agent goals and objectives. Should the agent be more proactive, the agent will attempt to pursue its designed goals but it will ignore its environment and the relevancy of such objectives [11], [26].

In addition, for the agent to work efficiently in a large-scale complex distributed coordinated control such as the one for the FFEPU, the agent must be intelligent. An intelligent agent must be situated in an environment and exhibit autonomy, flexibility, robustness and social interactions [11].

Intelligent Agent

The intelligent agent stands different from other software programs that may be designed to carry out similar control processes. The characteristics of intelligent agents prepare them act autonomously and in a given environment. Therefore, the MAS may only be comprised of two or more intelligent agents exhibiting the following properties [6], [10], [11]:

An intelligent agent must be *situated*. Specifically, this means that the intelligent agents must share a common environment which does not constrain the notion of an agent very much.

The intelligent agent must exhibit *autonomy*. For the individual agent, it is important to interact with other intelligent agents and factor in the new information it receives, but ultimately, the agent works independently, meaning that the intelligent agent is not controlled externally.

The intelligent agent must be *robust*. Challenging environments may result in the individual agent's failure to achieve its goal. The intelligent agent must be able to recover from such failures and must adapt to external and internal challenges. A robust agent within the MAS will not exhibit system-wide failure, instead it will be fault tolerant and it will degrade gracefully, a property which will be displayed in its implementation with the distributed coordinated control for the FFEPU system in the simulations in Chapter 6.

The intelligent agent is also *flexible*. For a given goal, the intelligent agent has a range of ways to achieve the goal if a specific plan fails or if other intelligent agents ceased to operate properly. Flexibility is deeply related to robustness and they are two of the most distinguishing agent properties.

Finally, the intelligent agent must be *social*. Communication between agents should not only be focused on the transfer of data, but it should also exhibit agent interaction. Compared to human interaction, the agent interaction should mimic important exchanges such as negotiation, coordination, cooperation and teamwork.

Even though these properties are considered the most essential, the intelligent agent may also display some other properties that aid the implementation of the MAS control configuration [11].

Architecture of the Single Agent

By defining an intelligent agent, we can now describe the architecture of the single agent comprising the MAS for the FFEPU control. The agent and its main tasks, which may be computational algorithms, must first be situated in an environment which will be shared with different agents. The agent requires its perceptor to be aware of its environment and effecter to react to changes, exhibiting its reactive property. Intelligent agents must be social; therefore they have the ability to communicate with each other. By communicating, the agents are able to account for the operation of other agents, making its operation robust. The single agent is flexible and is capable of accounting for different scenarios and must plan accordingly. In different scenarios, there is a decision making process which must be taken by the agent itself. By being autonomous, the agent is in charge and is not controlled externally. Finally, the agent runs the algorithms determined

by the decision-making process while targeting its goals and objectives. The single agent is able to run different algorithms to best suit the environment of the agent. This single agent architecture has been researched previously in [5], [19], [22], [23] and is better illustrated in Fig. 42.

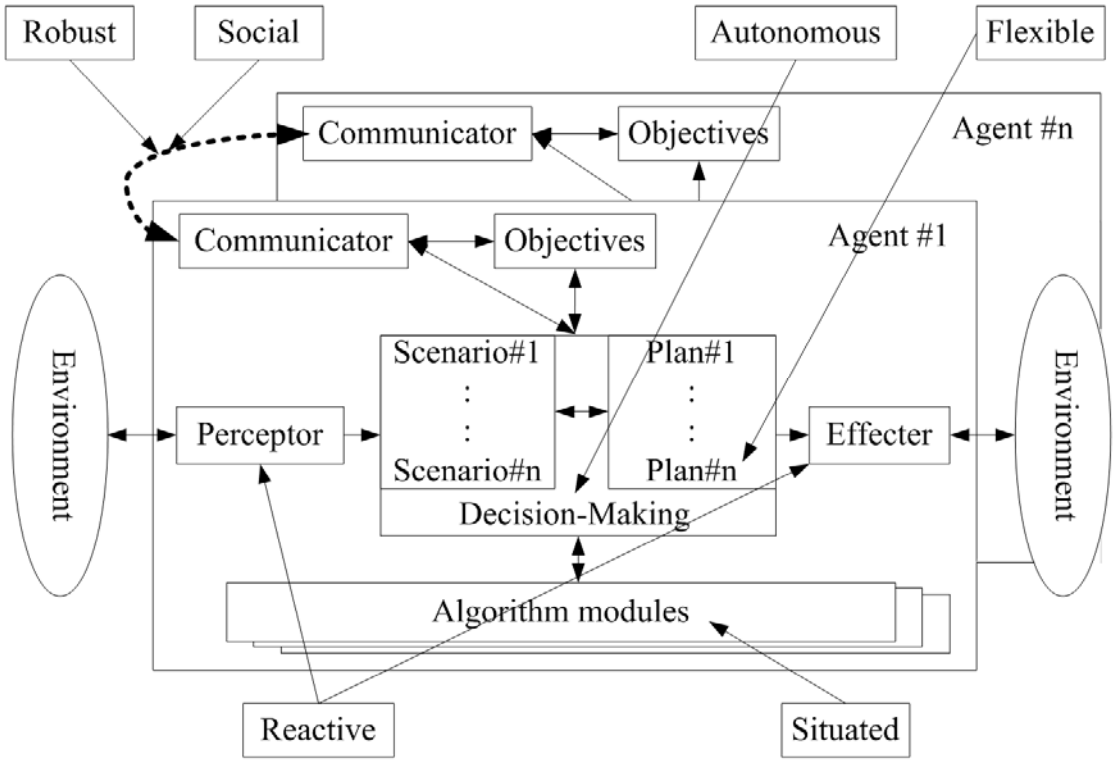

Fig. 42: Single Agent Architecture [19].

The agent architecture displays not only an agent sensitive reactive to its environment, but it also is proactive by recognizing its objectives and taking into account different scenarios in the decision making process. Fig. 42 shows the architecture of the single agent, however how the MAS is organized and the communication process must also be established to fully understand the MAS process.

5.3 MAS Organization

The MAS is based on the principle of autonomous agents working together towards a common objective, without any of the agents having general knowledge of this goal. The agents know their tasks within the overall system and cooperate with each other to attain their respective goal. For this purpose, different MAS hierarchy is presented and described.

MAS Hierarchy

The MAS must present an organization that allows for a performance based on cooperation. It is understood that not all agents perform tasks for the direct control of the FFEPU, which are computationally intensive, and that some agents actually perform management tasks, overlooking other agents and make sure that they are functioning properly.

For this purpose, the MAS agents are organized in a hierarchical three different levels, which are made up of a high level, middle level and low level. The agents in each level have a specific set of tasks and play a role in the MAS society to better control the FFEPU system.

The high level role is different from the lower levels. In this level, the agent is not directly concerned with FFEPU control tasks. It also does not perform management tasks for the middle level agents. Its only role is to interface between the MAS and the rest of the system and the only agent in this organization level is the interface agent.

The middle level agents offer management for the overall FFEPU distributed coordinated control agents. These agents delegate the responsibilities for each one of the intelligent agents and are also in charge of monitoring these agents for correct

performance. Also, the middle level agent communicates with the high level to account for changes in objective by the human operator. The main agents in this level are the delegation agent, which delegates tasks for the low level agents taking into account changes by the interface agent and the monitoring agent, which monitors the activities of the lower level agents to ensure their correct functioning and reports back to the delegation agent when necessary.

The low level agents are responsible for the direct control of the FFEPU. They are assigned tasks for the different processes of the distributed coordinated control. These intelligent agents have highly computationally demanding tasks that may be implemented in different software than that of the agent. The agents in this level are the feed-forward agent, the four feedback agents and the four gain optimizer agents. Fig. 43 illustrates the interaction between the agents in the different levels.

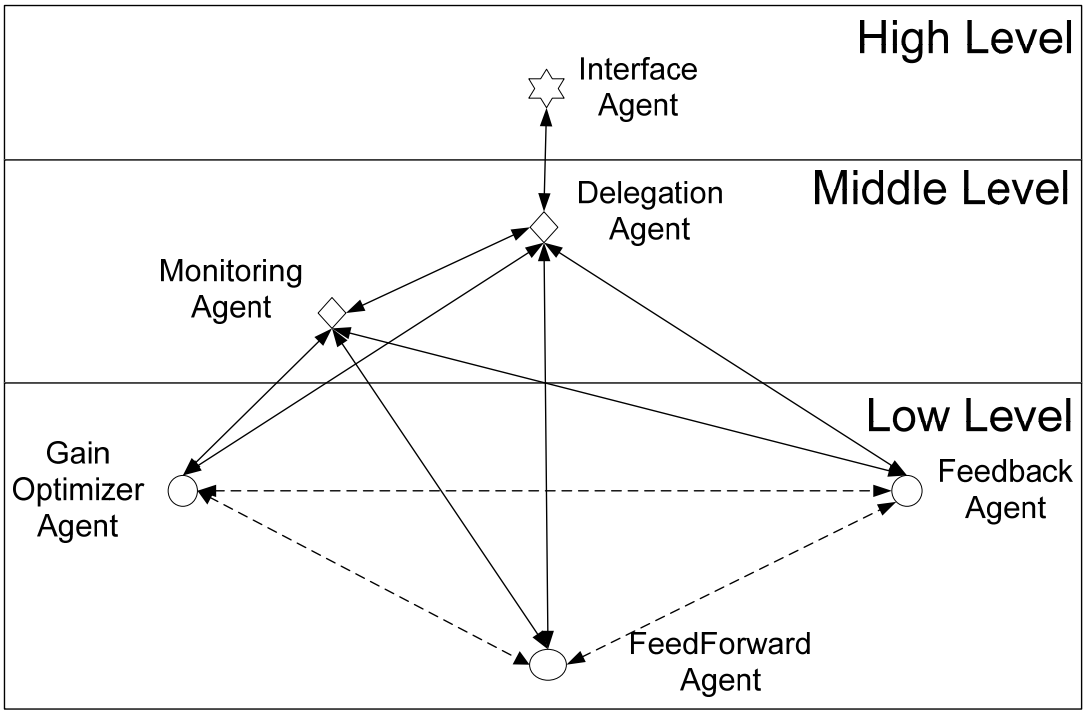

Fig. 43: Overall MAS Hierarchical Structure.

MAS Communication

The MAS communication between agents must be effective and exhibit an efficient interaction between the agents properly. Failing to provide a communication protocol that addresses the flow of agent communication may affect the overall response of the system negatively.

Fortunately, an Agent Communication Language (ACL) standard is developed by the Foundation for Intelligent Physical Agents (FIPA) in [27], [28]. This ACL was created in recognition of a need for a common agent language with a set structure in any platform used for the implementation of the MAS. The FIPA-ACL defines the structure for the agent language used to communicate between agents.

The FIPA-ACL message structure is described by several parameters. The most important and the only required parameter for ACL messages is the performative, which basically denotes the type of the communicative act of the ACL message. Most ACL messages will also include a sender, a receiver and content parameters with a reply-by field implemented optionally [27] , [28].

Even though these five parameters suffice for most implementations of the MAS, FIPA-ACL offers thirteen different parameters for the agents to use. These parameters are used to control the type of communicative act, the participants in the communication, the contents of the message, as well as the language and encoding, ontology, protocol for the conversation, ID for the conversation control and conversation control parameters if necessary. Table IX describes the message parameters and a description [27].

Table IX: FIPA-MAS Message Parameters [27]

Message Parameters	Description
performative	Type of communicative act
sender	Participant in communication
receiver	Participant in communication
reply-to	Participant in communication
content	Content of message
language	Content language
encoding	Encoding of content
ontology	Ontology used
protocol	Protocol for conversation
conversation-id	ID for conversation control
reply-with	Conversation control parameter
in-reply-to	Conversation control parameter
reply-by	Conversation control parameter

Most of these parameters address communication needs that otherwise would be difficult to implement and are often times not required for an efficient implementation of the MAS. The sender and receiver parameters indicated from where and to which agent a message is addressed. The content of the message is intended for the receiver to interpret. The performatives, which are the type of communicative act, are grouped in twenty three different types [29].

For the MAS distributed coordinated control for the FFEPU, it has been suggested in previous research [6], [17] that only six performative types be used for implementation. These performatives are described below:

- The request-performative is used when the sender is requesting the receiver to perform a given action described by the content of the message. The action itself can be requesting data from the receiver, to send data to another agent or even to change the goal at hand.

- The agree-performative is basically the response to a request performative when the agent accepts the request. With this performative, the request-sender agent has knowledge that the agent has plans to comply with the request.
- The refuse-performative is the opposite of the agree-performative. An agent may refuse to carry out the request of another agent if it is occupied with an important task, if the action is not feasible, or if the agent does not have enough knowledge or have insufficient privilege.
- The inform-performative is used to provide information for a request-performative from another agent but it is also used to send important data for the normal functioning of the low level agents. For the FFEPU control system, it is used to transmit data necessary to run the FFEPU and its different control processes.
- The subscribe-performative is closely related to a request-performative, with the exception that the subscribe-performative requests that a piece of information be sent to the sender agent as soon as the information has been changed or updated.
- The not-understood-performative is used by an agent who was not able to process a message sent by another agent. The receiver of this performative had previously send a communicative act to an agent that could not process the content of the message, it may have been expecting a different message or it was not design to process such an act.

These are the most basic performatives necessary for the functioning of the FFEPU MAS [6], [17], [29]. However, to implement the MAS for the FFEPU, a suitable platform must be chosen.

5.4 Java Agent Development Framework

The options to implement MAS can be diverse and there is plenty of freedom to choose a platform more suitable for the goal of the system. Generally, the decision to use platform is divided in three general groups: class 1 platforms, such as PRS, AgentBuilder or JACK, which focus on the internal agent reasoning to support plans and goals; class 2 platforms, such as the Java Agent Development (JADE) Framework, Zeus or OAA, which focus on inter-agent communications and offer infrastructure that conform to the FIPA standards; and class 3 platforms, such as Grasshopper, D'Agents or Aglets, which focus on agents with mobile capabilities [11].

MAS has been implemented previously using more computational intensive software, MATLAB, for the control system involving medium-sized power plant [17], [19], [25] and also for more larger power plants [6], [19]. As pointed out in [6], even though MATLAB is well suited to handle the complex computational operations for the control process for large-scale power plants, MATLAB lacks the ability to offer efficient communication and delay-free processing, affecting the effectiveness of the overall control system. The JADE framework addresses these inefficiencies by providing asynchronous agent communication, a FIPA-complying ACL messaging structure and the ability to be implemented in any set of computers that implement the JAVA virtual machine without other software requirements.

To tackle this particular challenge in the implementation of the MAS, it is practical to aim to improve the communication between that agents and the decision to implement the MAS in a class 2 platform becomes apparent. Specifically, JADE became a very appealing candidate for the implementation of the distributed coordinated control.

JADE is a software framework, implemented in Java, designed to develop agents efficiently and to comply with the standards of the Foundation of Intelligent Physical Agent (FIPA) for agent development previously discussed. Java is the programming language chosen due to its attractive features geared toward object-oriented programming. JADE offers a set of tools that supports the debugging and deployment of the MAS. JADE can be distributed across different machines, which do not even have to share the same operating system. The only system requirement for the machines involved is Java Run Time, version 5 or later. The FIPA communication standards have been implemented and integrated in JADE, offering flexible and efficient messaging [30], [31].

Additionally, JADE is bundled with tools that aim to simplify the administration of the platform and application development. Among these tools there is a remote agent management tool, which acts as a graphical console for the control and management of the agents, a Directory Facilitator, which is a graphical user interface (GUI) that is used as a yellow pages service so that an agent can find other agents providing the services the agent requires, and other convenient tools that facilitate the development of the FFEPU-MAS [30], [31].

JADE exhibits the ability to add and develop agents with small effect in the overall MAS architecture. Among many others, JADE presents features that facilitate the MAS implementation, exploits the advantage of asynchronous messaging protocol and poses very little overhead structure to implement new agents as needed by the system. The FIPA-MAS communication standards have been implemented and fully integrated in JADE, offering flexible and efficient messaging. To fully take advantage of such system, the implementation of FFEPU MAS distributed coordinated control is necessary.

5.5 MAS Implementation in JADE

Motivation

To successfully implement the desired FFEPU MAS distributed coordinated control with JADE, it is necessary to take into account some of the limitations of JADE. The implementation of the controls required by the distributed coordinated approach is too computationally burdensome to be considered in Java. Some of the module algorithms actually are better suited for software that offers useful mathematical tools that would have to be adapted to Java.

The FFEPU model itself is divided into four different modules and thirty three subsystems, making it quite large and complex. Fortunately, the most computationally intensive tasks can be handled by MATLAB. MATLAB has different tools that can address more complex tasks required by the lower level agents.

MATLAB has also been used previously in [6], [17] to address the simulation and computational implementation of MAS distributed coordinated control for power plants. Therefore, it is advantageous to implement the algorithm modules in MATLAB and, at the same time, allow JADE to provide a superior network communication structure since MATLAB presents communication overhead due to the use of TCP/IP communication in the control of a large-scale power plant [6].

The proposed approach utilizes both JADE and MATLAB to implement the MAS distributed coordinated control for the 600MW FFEPU model. This is achieved by allowing JADE to be the platform in which agents interact with each other and having MATLAB take care of the computational tasks for each of the agents.

Agent Shell Architecture

The architecture of the single agent is common for most agents within the MAS. This architecture is called the agent shell and it illustrates the interaction of the agent with its environment and with its own algorithm modules. In this project, the agent shell is divided in two different platforms. The MAS communication container, which carries out the ACL messaging and interaction is implemented in JADE. This approach takes advantage of JADE as the MAS platform. On the other hand, MATLAB is used within the individual agents to take care of the computational tasks. Since MATLAB only does the tasks that each agent requires it to do, the MATLAB modules do not communicate to each other. They only need to communicate to their respective JADE agents. Fig. 44 illustrates the agent shell architecture for the MAS communication.

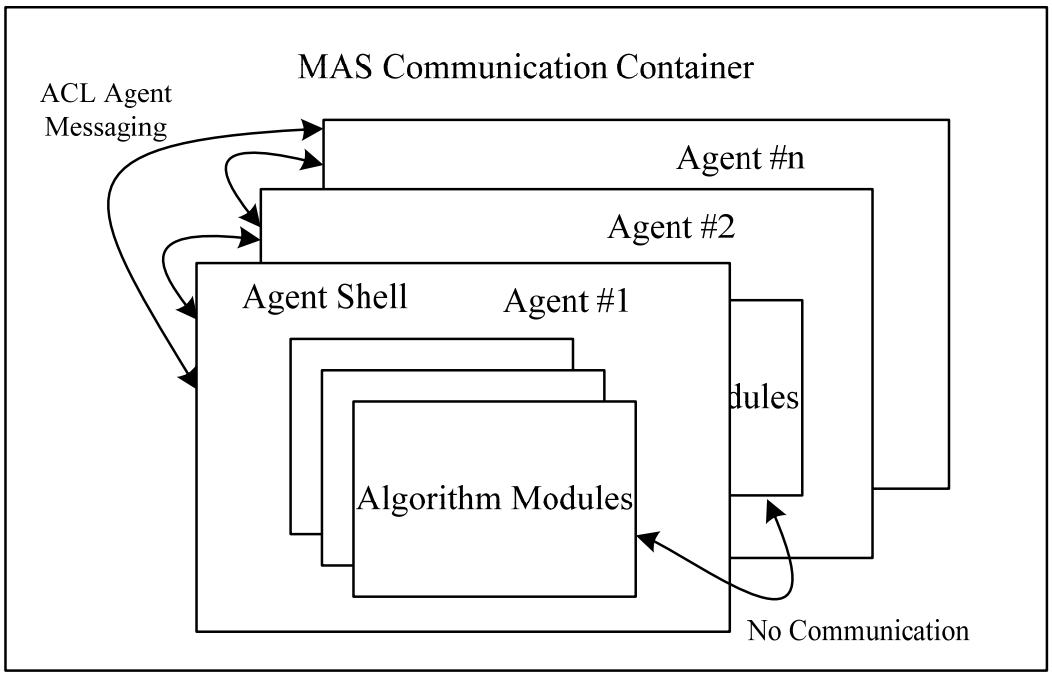

Fig. 44: Agent Shell Architecture.

Agent Types

For the implementation of the JADE-based MAS, different intelligent agents must be implemented to address the overall control needs of the system. As previously asserted, there are three levels of hierarchy in the MAS. The agents are described in the following paragraphs:

Interface agent. The interface agent is the only agent at the high level hierarchy. Its main task is to serve as a the interface between the MAS decision making process and the human operator who may choose to change the system objectives, pursue different control goals, or give priority to different agents.

Delegation agent. The delegation agent is a management agent in the middle level of the MAS hierarchy. The main task of the delegation agent is to create all the different agents of the middle level and lower levels. Since JADE posses the capability of using a built-in Directory Facilitator, the delegation agent makes use of this directory, similar to a yellow pages directory, to keep track of the registry of the agents. With the help of other middle level agents, it is able to make decisions based on the performance of lower level agents. The delegation agent is created by the interface agent at the time that the JADE-based MAS control is implemented.

Monitoring agent. The monitoring agent is another one of the management middle level agents. Its primary task is to monitor closely the performance and status of the lower level agents to aid the delegation agent in its decision making process. It is in constant communication with the delegation agent and it is also possible to use the capabilities of the monitoring agent to collect data from the lower level agents and the

MATLAB FFEPU model to train and update the power plant identifiers and keep them current for changes in the functioning of the power plant.

Feed-forward agent. The feed-forward agent is one of the agents in the low level of the JADE-based MAS hierarchy. The tasks of this agent directly affect the performance of the FFEPU and it is the first agent to interact with the FFEPU. As discussed in chapter three, the tasks performed by the feed-forward agent are computational intensive. The feed-forward agent is in charge of implementing the reference governor process for a given power demand, finding suitable control values that optimize a given set of multiple objectives, determine the set-points for the FFEPU and ultimately sent these control values along with the set-points to the agents of interest.

Additionally, the JADE-based feed-forward agent is given the responsibility of initializing the four feedback agents and the FFEPU system with its respective initial conditions. Without this initial step, the FFEPU and the feedback agents are not able to initialize. To run the feed-forward algorithm modules, the feed-forward agent must, as a first step, start the MATLAB feed-forward control process, which will be in constant communication with its JADE counterpart. Since the FFEPU is modeled in MATLAB, the JADE-based feed-forward agent is also in charge of starting the FFEPU model. Fig. 45 illustrates the JADE-based feed-forward process.

Feedback agents. There are a total of four JADE-based feedback agents and they are also implemented as low level agents. As described previously in Chapter four, the feedback process is divided in four different modules for each one of the FFEPU modules. The four different agents must be initialized by the JADE-based feed-forward

agents and afterwards they run feedback control loops taking the output of the FFEPU and the feed-forward set-points and control values.

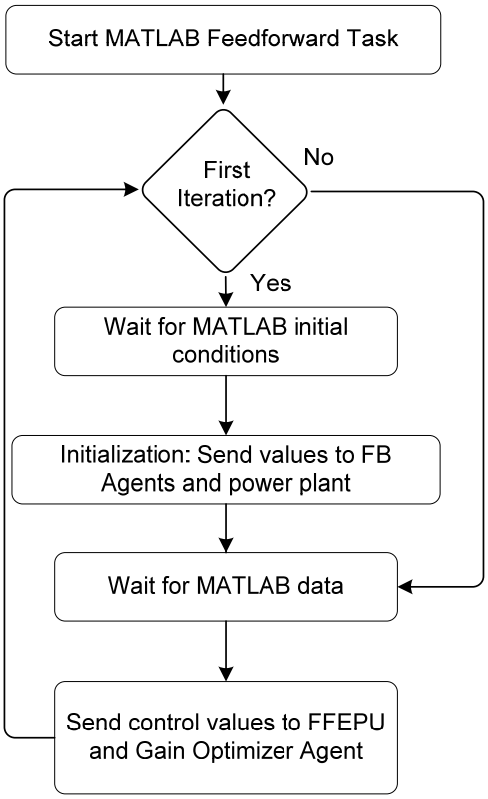

Fig. 45: JADE-based Feed-forward Agent Process.

The feedback agents are the only agents in the lower level hierarchy that actually run the feedback process in JADE. This is made possible due to that the feedback control loops relatively not as computationally intensive, as compared to the feed-forward control loops, which must implement a heuristic optimization algorithm making use of artificial neural networks, the online implementation of the gain optimizer agent, which must perform an heuristic optimization algorithm as well, or the FFEPU which must run thirty three subsystems in the different four modules. Another important factor for the decision to run the feedback agent task in JADE is that, unlike the feed-forward agent, which supplies set-points and control valves for every second, the feedback agents must

implement the control process every 0.1 seconds, just like the MATLAB FFEPU model. Fig 46 illustrates the process for each one of the four feedback agents.

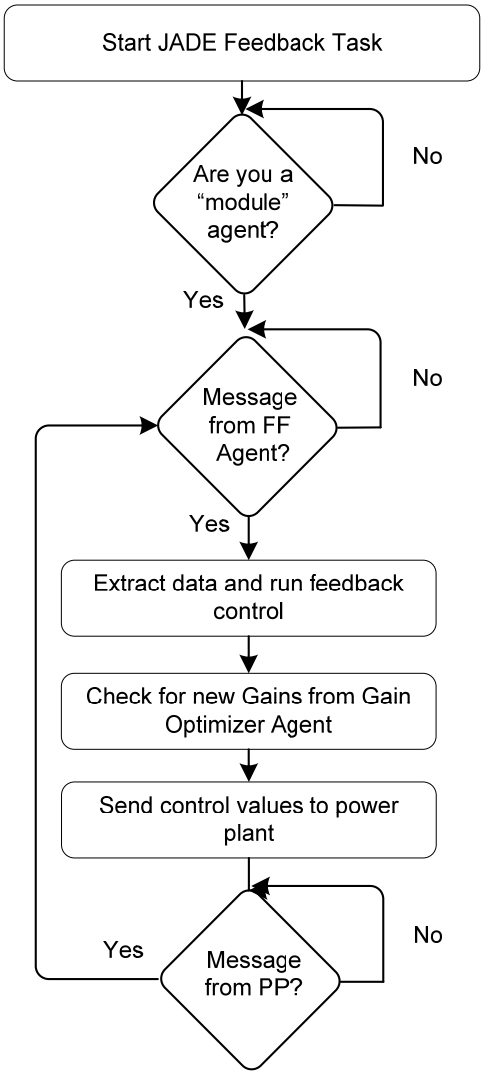

Fig. 46: JADE-based Feedback Agent Process.

Gain optimizer agents. The gain optimizer agents are the also located in the lower level of the MAS hierarchy. Much like the feedback agents, they are also distributed between the four modules of the FFEPU. In contrast to the feedback agents, the JADE-based gain optimizer agents start instances of MATLAB to carry out the

computational intensive tasks. These MATLAB instances remain in constant communication with their JADE-based counterparts. As oppose to the previous agents, the gain optimizer agent is not constantly determining suitable feedback gains. It only performs its gain optimizing tasks if the FFEPU exhibits a poor performance due to unsuitable feedback gains.

The gain optimizer agents receive data from the JADE-based feed-forward agent, from the JADE-based feedback agents and it also examines the output of the FFEPU model. It uses the data from the JADE-based feed-forward agent to determine a gain optimizing curve consisting of the feed-forward control and desired set-points. It examines the data obtained from the MATLAB FFEPU to determine the error signal. If the error signal goes beyond a given threshold, the gain optimizer process is run. Finally, the gain optimization process delivers sets of feedback gains with their respective error costs. The error signal due to the performance of the FFEPU is compared to the new optimized feedback gains. If there is reason to believe that the new feedback gains will improve the FFEPU performance, the JADE-based gain optimizer agents send the gains to the JADE-based feedback agents for implementation. Fig. 47 illustrates the process for all of the JADE-based gain optimizer agents.

JADE-based Agent Communication

The MAS communication for the dual platform approach for this thesis must be comprised of two different kinds of communications protocols; the communication between the JADE-based agents and the communication between the MATLAB instances in charge of performing the computational tasks and the JADE-based agents.

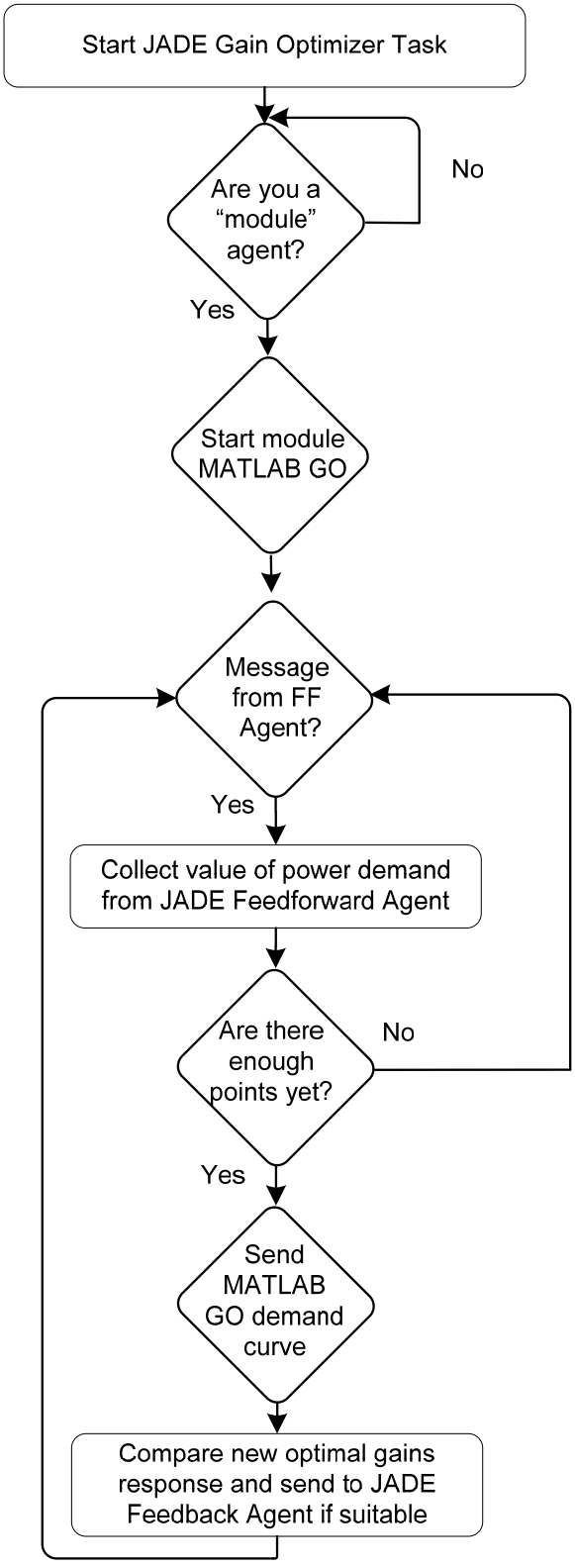

Fig. 47: JADE-based Gain Optimizer Agent Process.

As established previously, the JADE platform abides the FIPA-ACL protocol, implementing communication through the ACL messaging tools provided by JADE. This characteristic of JADE aids for the rapid and concise transmission of messages. Fig. 48 illustrates distributed coordinated control for the FFEPU. The low level agents are differentiated in the JADE-based shaded region and communicate with each other through the ACL messaging protocol provided by JADE.

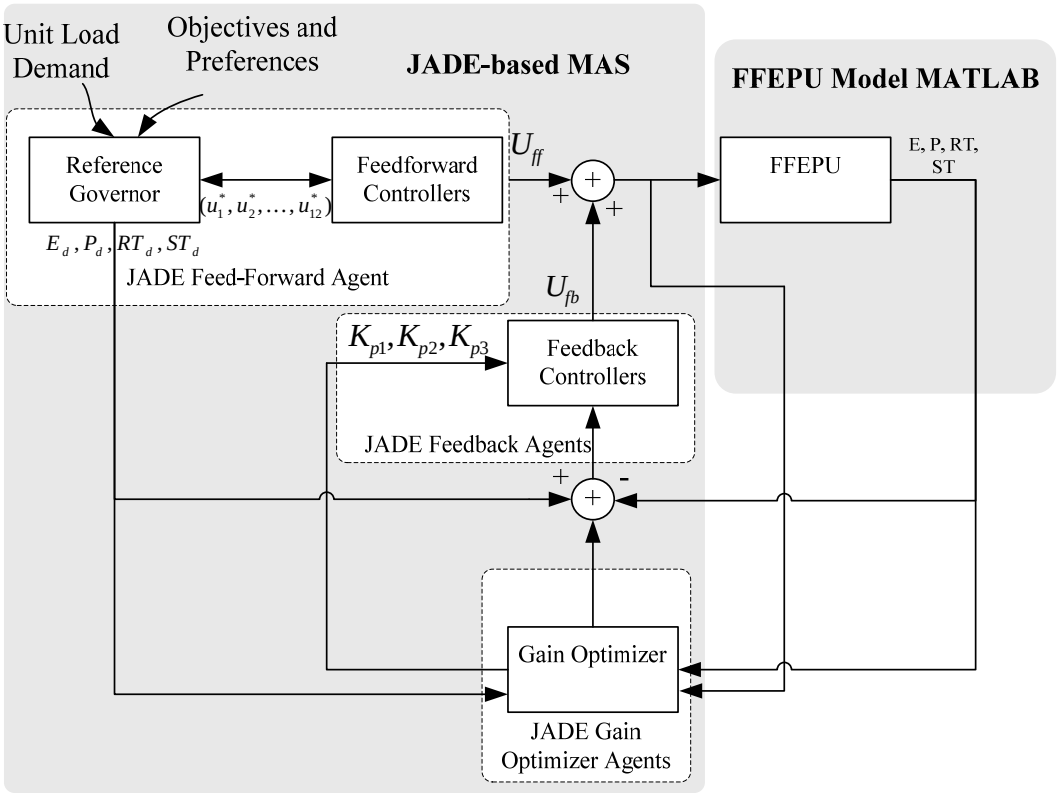

Fig. 48: JADE-based MAS Distributed Coordinated Control.

Additionally, the lower level agents must also establish a protocol to communicate to their MATLAB counterparts. As previously mentioned the feed-forward and gain optimizer agents perform its algorithm modules with the help of MATLAB and the feedback agents implement their tasks in Java. Fig. 49 displays the feed-forward control process with its MATLAB counterpart; Fig. 50 illustrates the operation of the

feedback agents and their operation in JADE; and Fig. 51 displays the operation for the gain optimizer agents and their interaction with their MATLAB counterparts.

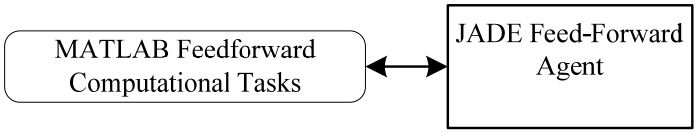

Fig. 49: Feed-Forward Intra-Agent Communication.

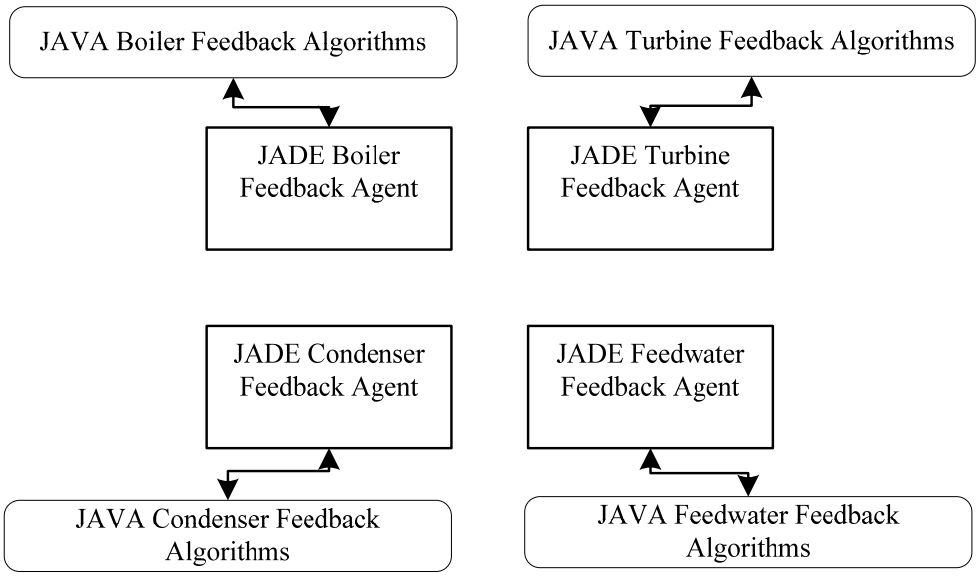

Fig. 50: Feedback Intra-Agent Communication.

The intra-agent communication must be processed in a timely manner and should have the ability to broadcast messages to multiple ports at the same time. Since JADE and MATLAB need to communicate in a network protocol that ensures fast delivery and little process time. For this reason, the decision was made to use User Datagram Protocol (UDP/IP) [32].

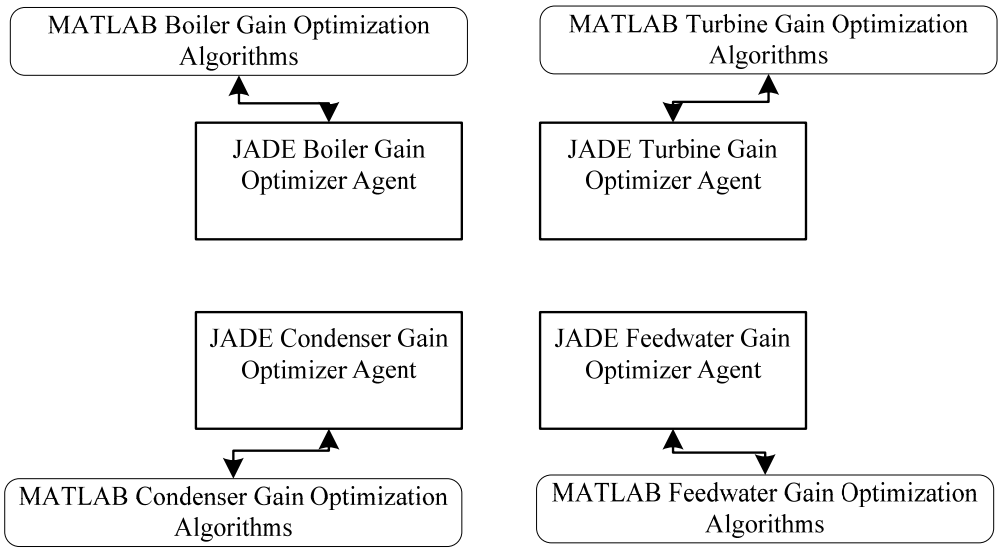

Fig. 51: Gain Optimizer Intra-Agent Communication.

UDP/IP is based on the exchange of datagram packets that are sent over the network to previously determined UDP ports. These datagram packets are sent as a broadcast, and the delivery is fast and it does not need to establish a connection. The opposite is expected from the Transmission Control Protocol (TCP/IP), which must ensure that a connection between the sender and the receiver is established. Even though TCP/IP protocol is considered more reliable, the UDP/IP protocol processing speed and ability to send messages to multiple ports outweighs the advantage of TCP/IP communication [32].

In order to implement UDP/IP communication between the JADE-based Agents and their MATLAB counter parts, the different UDP ports must be established before-hand to ensure that there are no communication issues and that the agents can send data to other agents in a timely manner. Table X enumerates each one of the UDP ports used for the intra-agent communication. There are twenty UDP ports used for sending and

receiving important data. By implementing the feedback agent computational tasks, the number of UDP ports used would increase to twenty eight, which, even with a robust UDP communication protocol, may cause communication delays.

Table X: UDP Ports for Intra-Agent Communication

UDP	Emitter System	Receiver System	Control Process
49994	MATLAB	JADE - MAS	FFEPU Initial Conditions
49996	JADE - MAS	MATLAB - FFEPU	FFEPU Initial Conditions
49998	MATLAB	JADE - MAS	FFEPU Feedforward
49999	JADE-MAS	MATLAB - FFEPU	FFEPU Feedforward
50002	MATLAB -Feedback	JADE - MAS	Boiler Feedback
50002	MATLAB -Feedback	JADE - MAS	Turbine Feedback
50004	MATLAB -Feedback	JADE - MAS	Condenser Feedback
50005	MATLAB -Feedback	JADE - MAS	Feedwater Feedback
50006	JADE - MAS	MATLAB -Feedback	Boiler Feedback
50007	JADE - MAS	MATLAB -Feedback	Turbine Feedback
50008	JADE - MAS	MATLAB -Feedback	Condenser Feedback
50009	JADE - MAS	MATLAB -Feedback	Feedwater Feedback
50020	MATLAB-Gain Optimizer	JADE - MAS	Boiler Gain Optimizer
50021	MATLAB-Gain Optimizer	JADE - MAS	Turbine Gain Optimizer
50022	MATLAB-Gain Optimizer	JADE - MAS	Condenser Gain Optimizer
50026	MATLAB-Gain Optimizer	JADE - MAS	Feedwater Gain Optimizer
50035	JADE - MAS	MATLAB-Gain Optimizer	Boiler Gain Optimizer
50036	JADE - MAS	MATLAB-Gain Optimizer	Turbine Gain Optimizer
50037	JADE - MAS	MATLAB-Gain Optimizer	Condenser Gain Optimizer
50039	JADE - MAS	MATLAB-Gain Optimizer	Feedwater Gain Optimizer

Finally, Fig. 52 illustrates the overall JADE-based MAS distributed control display with both intra-agent UDP/IP communication and the communication between JADE-based agents implemented through the ACL messaging protocol.

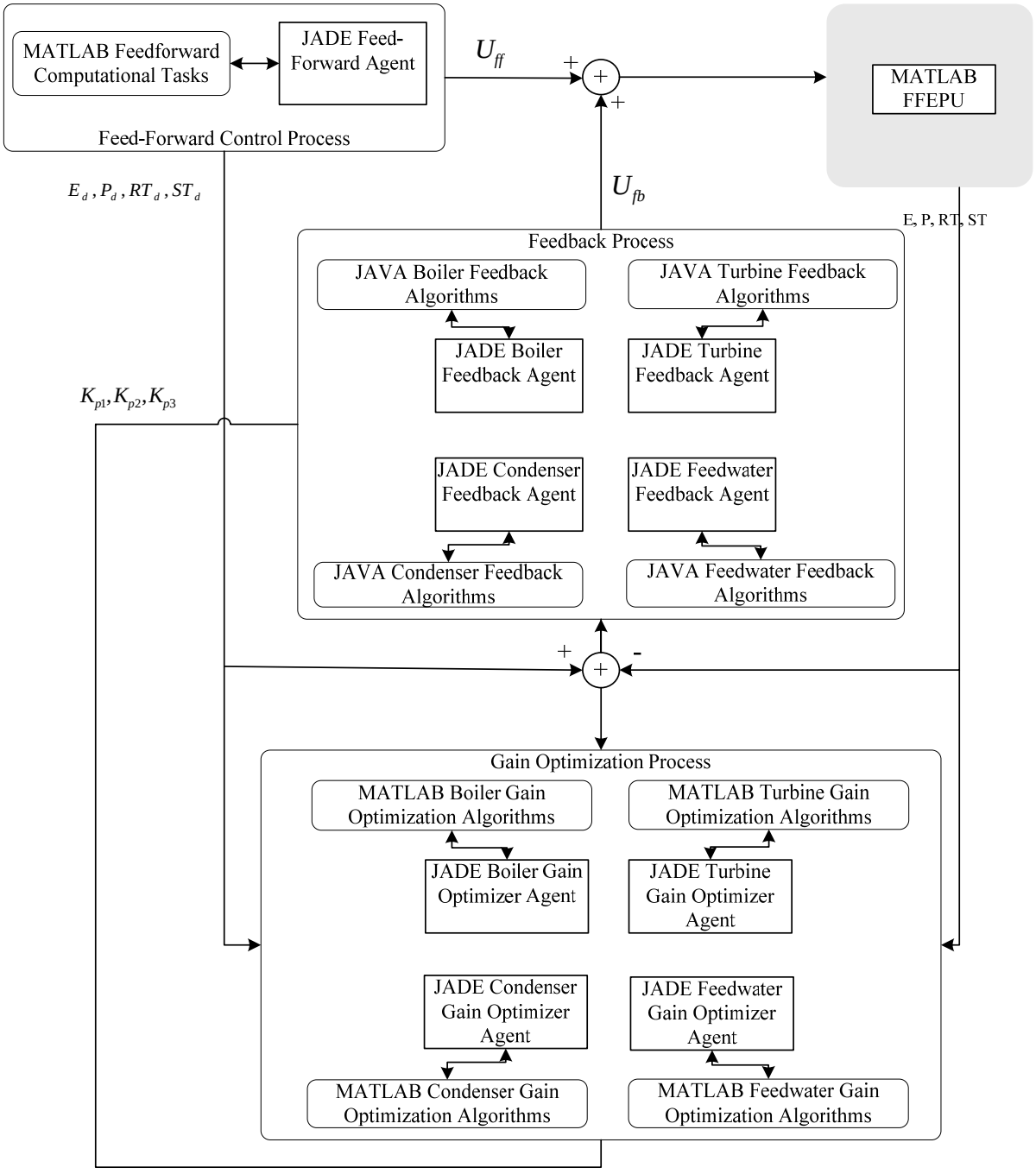

Fig: 52: The Overall JADE-based MAS FFEPU Distributed Coordinated Control.

CHAPTER SIX

Simulation and Results

This chapter presents the simulation and results obtained by the implementation of the JADE-based MAS distributed coordinated control for the FFEPU. Section 6.1 evaluates the proposed system for performance while addressing different multiple objectives by the reference governor. Section 6.2 displays the capabilities of the JADE-based MAS online gain optimization agents and draws a comparison for performance. Section 6.3 presents Fault Tolerance through graceful degradation, one of the main characteristics of the proposed JADE-based MAS.

6.1 *JADE-based MAS Simulations*

To validate the proposed JADE-based MAS control approach it is necessary to evaluate the performance of the system while addressing different groups of multiple objectives. These different multiples objectives were described in Chapter three and the FFEPU simulation and results are presented below. Fig. 53 displays the comparison of the FFEPU power outputs to each of the different sets of multiple objectives. As noted previously, the response is very similar since for all the multiple objectives, power tracking is the most important objective. Fig. 54 displays the FFEPU pressure output for the three different sets of multiple objectives. As previously noted, the pressure set-points differ for each sets of multiple objectives because, as opposed to power tracking as an objective since these are generated by the reference governor. As the simulations display, the JADE-based MAS FFEPU responses exhibit an excellent performance.

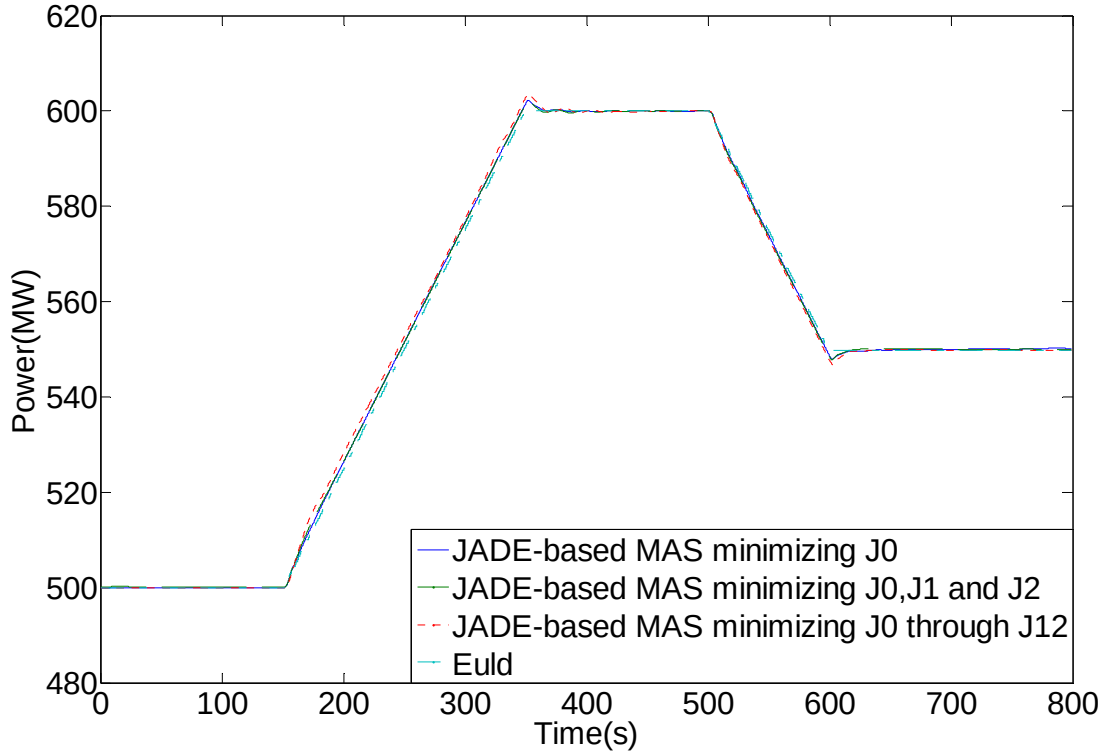

Fig. 53: FFEPU Power Output for Different Multiple Objectives.

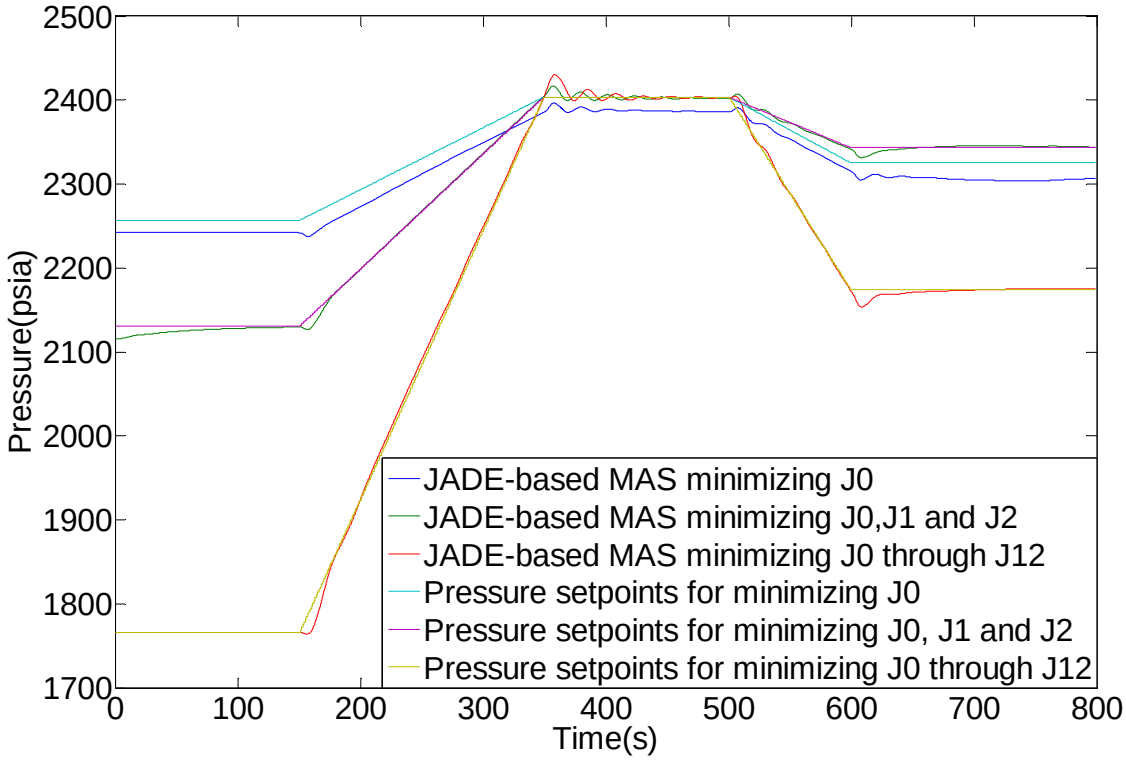

Fig. 54: FFEPU Pressure Output for Different Multiple Objectives.

6.2 *JADE-based MAS Simulations with Online Gain Optimization*

One of the main advantages of implementing a MAS-based distributed coordinated control is the ability to run optimization of the system simultaneously and in parallel to the simulation of the overall system. In the proposed approach, the feedback agents are in charge of providing PI control to the different FFEPU modules. The challenge arises when the PI gains chosen for the implementation of the MAS happen to be inadequate. Fig. 55 displays the FFEPU power output for a simulation with unsuitable feedback gains compared to the response of an FFEPU with the implementation of the online gain optimizer agents to correct the response.

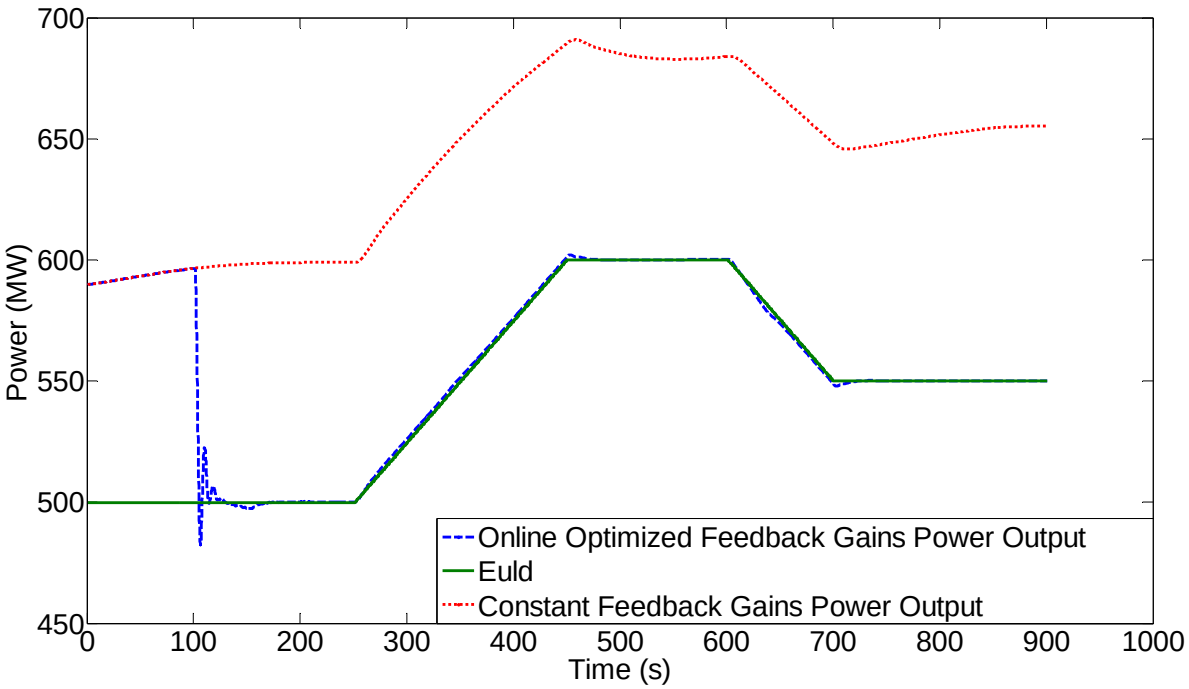

Fig. 55: FFEPU Power Output Comparison of Online Gain Optimization.

Even though Fig. 55 displays a simulation where the feedback gains optimization is evident, the online gain optimizer agents can offer for a faster converging and more stable response even when the response for constant feedback gains seems adequate. Fig.

56 shows the comparison of FFEPU power output with a better set of constant feedback gains to a step response power demand signal.

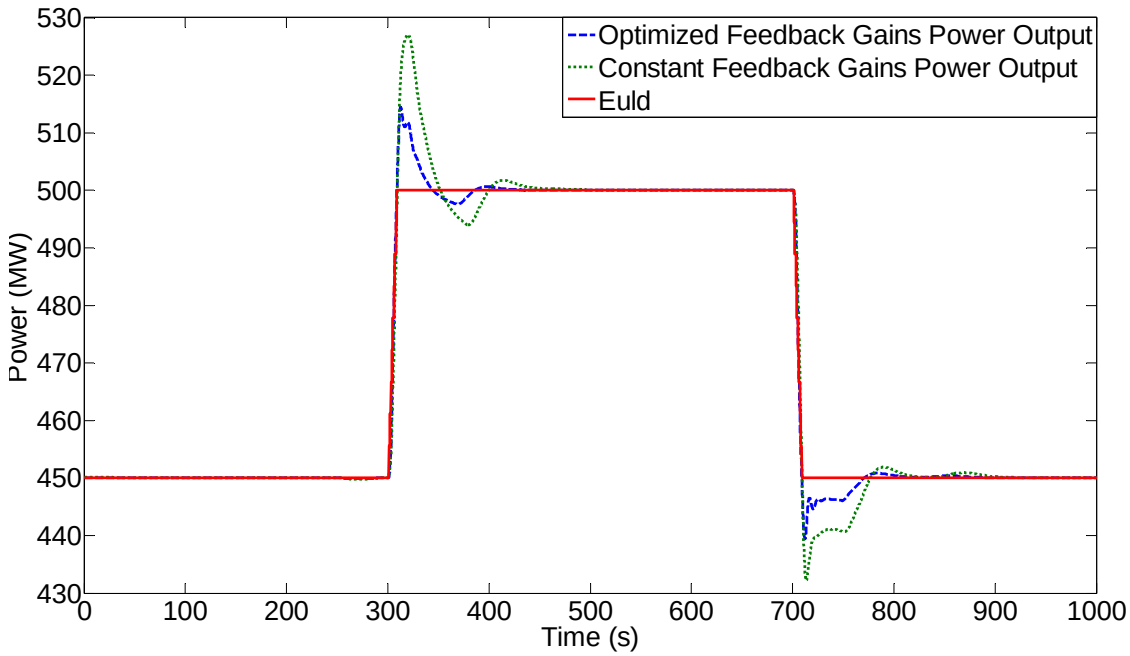

Fig. 56: Step Response Power Demand FFEPU Power Output Comparison.

Furthermore, to validate the performance of the JADE-based MAS Gain Optimizer Agents, the performance is compared with the MATLAB-based MAS implementation. Fig. 57 displays the FFEPU power output of both platforms and Fig. 58 displays the FFEPU pressure output of both platforms. For these simulations, the chosen feedback PI gains are different from those implemented for Fig. 53 and Fig. 54, since the latter ones display a performance that would not allow the gain optimization process to be shown. The gain optimization process is only deemed necessary when there is a power tracking error above a certain threshold and when the proposed optimized gains carry an error cost that is lower than the actual FFEPU power tracking error.

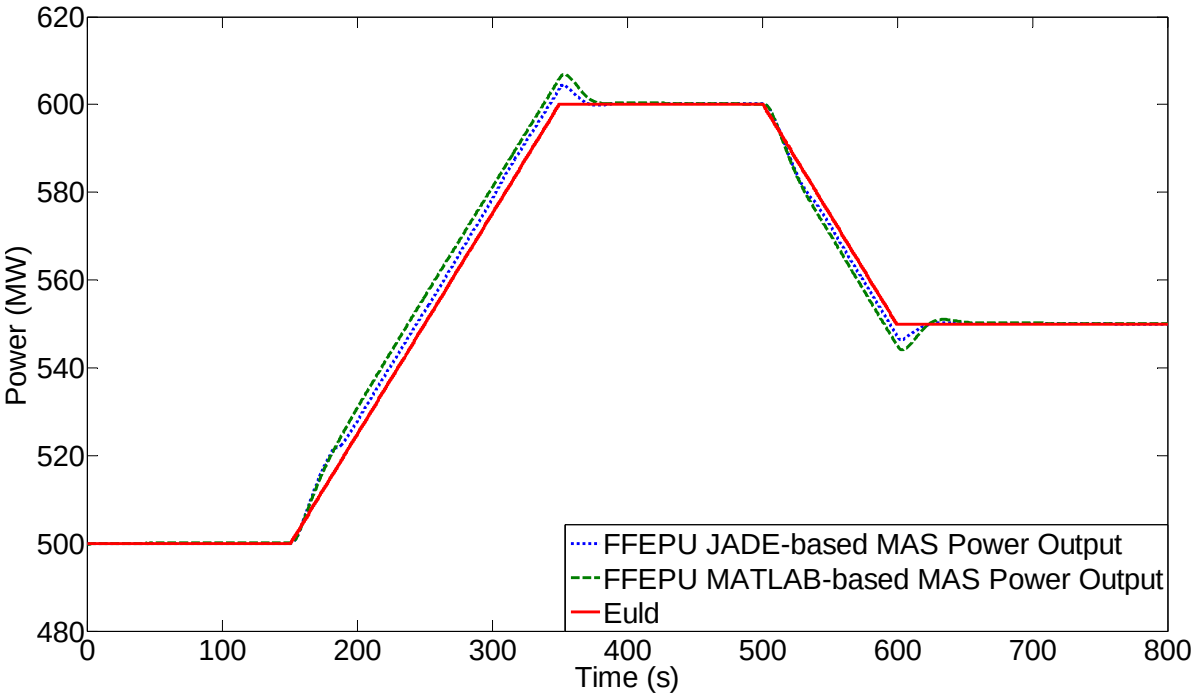

Fig. 57: Different MAS Platforms Gain Optimizer Power Output Comparison.

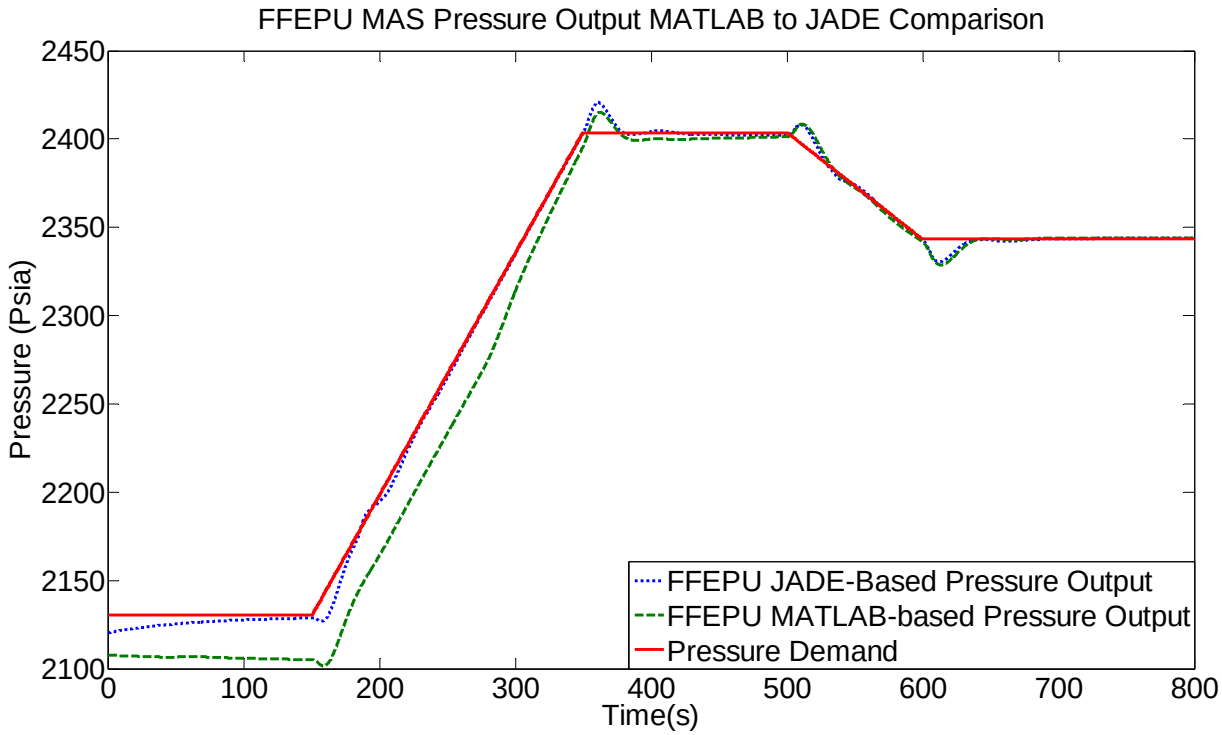

Fig. 58: Different MAS Platforms Gain Optimizer Pressure Output Comparison.

6.3 *JADE-based MAS Simulations Fault Tolerance*

A major advantage of the implementation of JADE-based MAS for the distributed coordinated control for the FFEPU is that the overall system is robust, and therefore fault tolerant. This property, also known as graceful degradation, is better shown in the overall good performance of the FFEPU even when different agents in charge of the direct control progressively fail. To demonstrate this principle several simulations were produced. To simplify the different case scenarios for agent failure, it is assumed that the gain optimization agents cease to function. Secondly, since the lower agents are most important for the direct control of the FFEPU, each one of the feedback agents will be allowed to fail for each case scenario. This will better allow observing how the graceful degradation affects the outputs of the FFEPU.

In the context of the JADE-based MAS, failure is defined as the complete inaction by an agent responsible of carrying out a task of the lower level hierarchy. Fig.59 through Fig. 70 present the FFEPU responses when the four different feedback agents are allowed to fail at approximately three hundred seconds into the simulation. Fig. 59 through Fig. 61 show the FFEPU power, pressure and temperature output, respectively, when the condenser feedback agent is allowed to fail. Fig. 62 through Fig. 64 allows the feed-water feedback agent to fail in addition to the condenser and it is also compared to the previous case scenario. Fig. 65 through Fig. 67 also allows the turbine feedback agent to fail in addition to the previous agent failures and compares the response to the previous case scenarios. Furthermore, Fig. 68 through Fig. 70 show the failure of the four feedback agents which represents the FFEPU response in non-MAS distributed control system. This response is compared to the other case scenarios of graceful degradation. The

simulations prove that the JADE-based MAS graceful degradation is obviously preferable to the Non-MAS control failure.

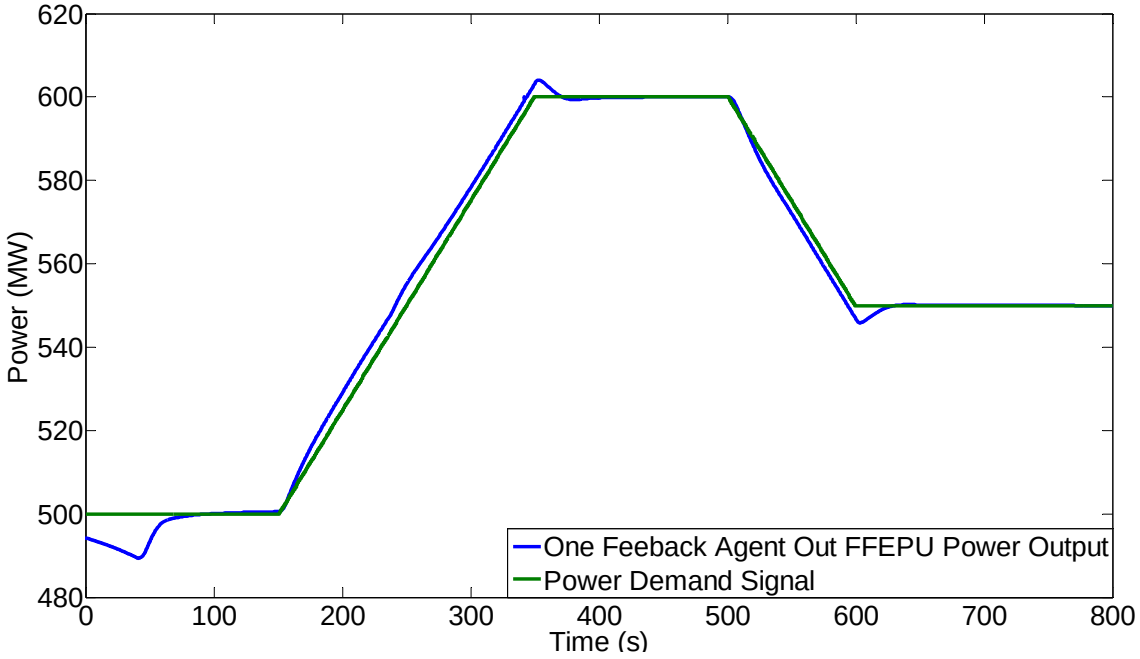

Fig. 59: Condenser Feedback Agent Failure FFEPU Power Output.

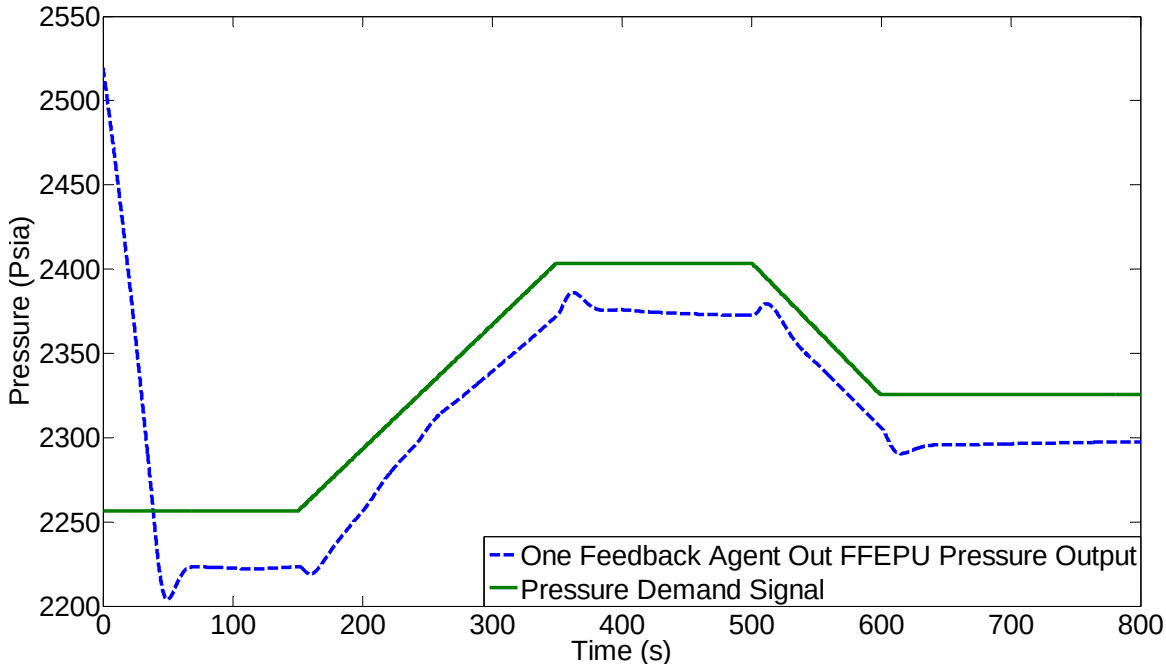

Fig. 60: Condenser Feedback Agent Failure FFEPU Pressure Output.

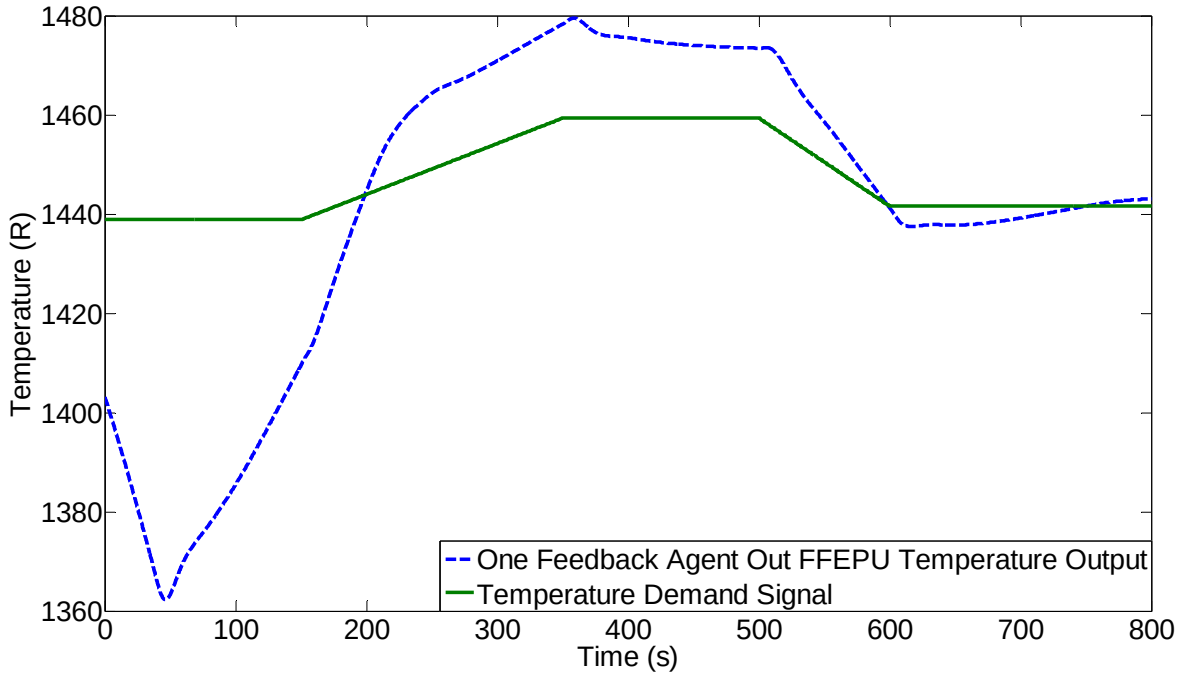

Fig. 61: Condenser Feedback Agent Failure FFEPU Temperature Output.

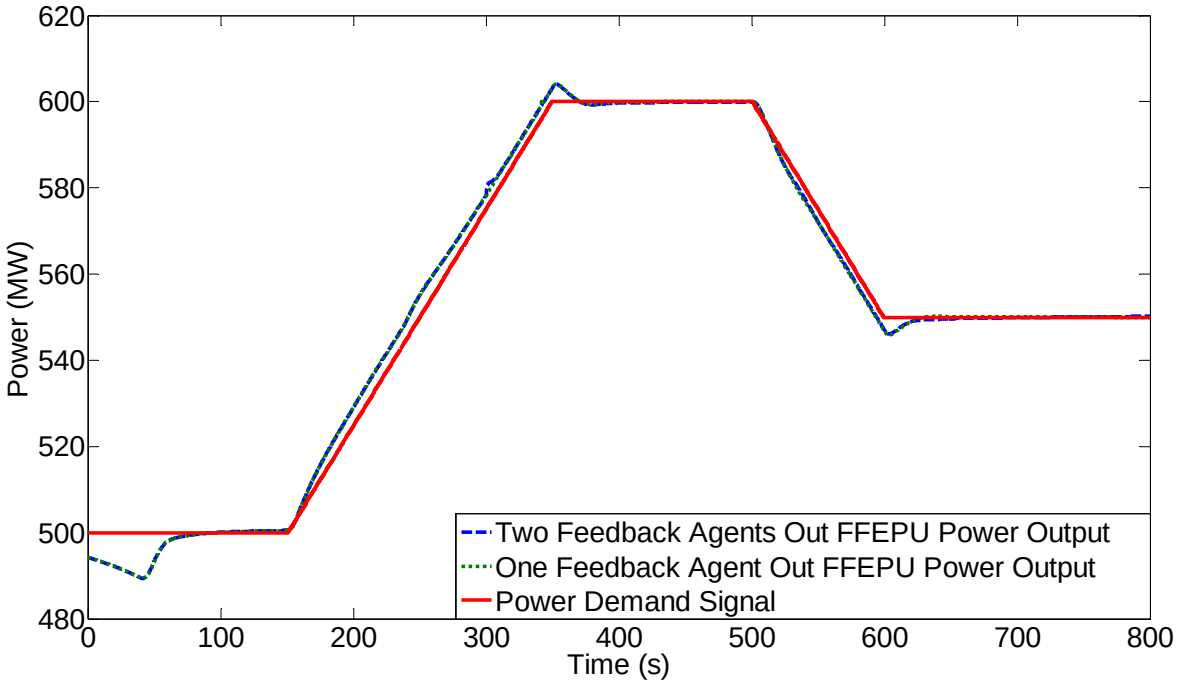

Fig. 62: Adding Feedwater Feedback Agent Failure FFEPU Power Output.

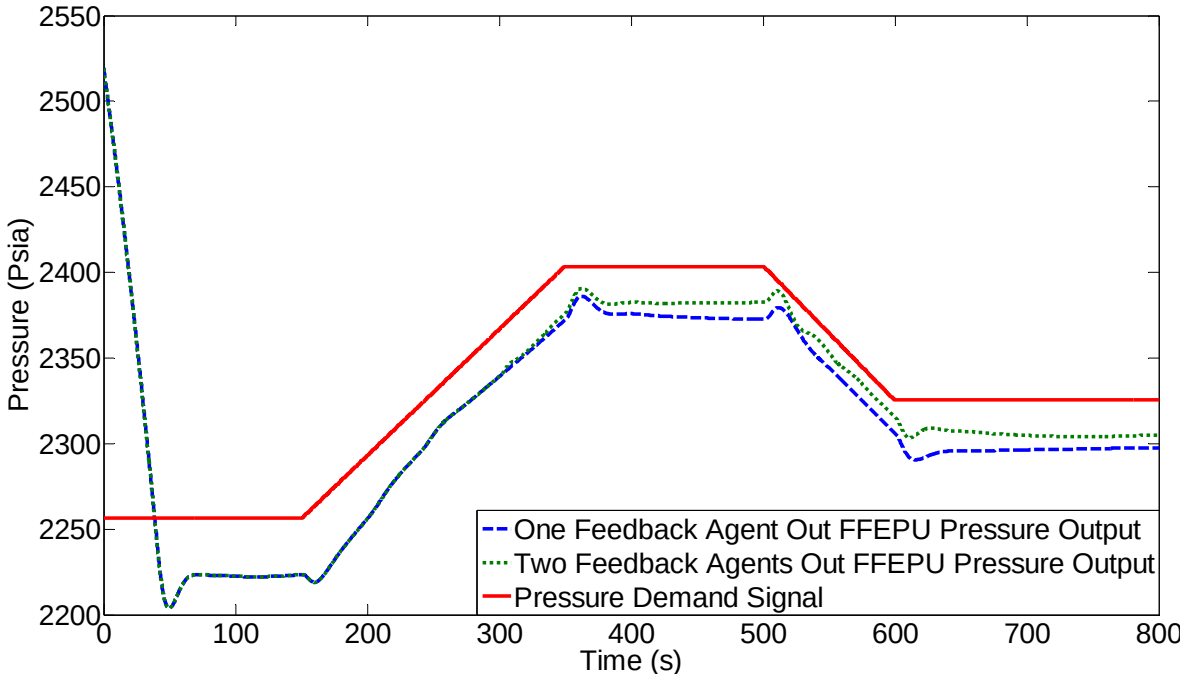

Fig. 63: Adding Feedwater Feedback Agent Failure FFEPU Pressure Output.

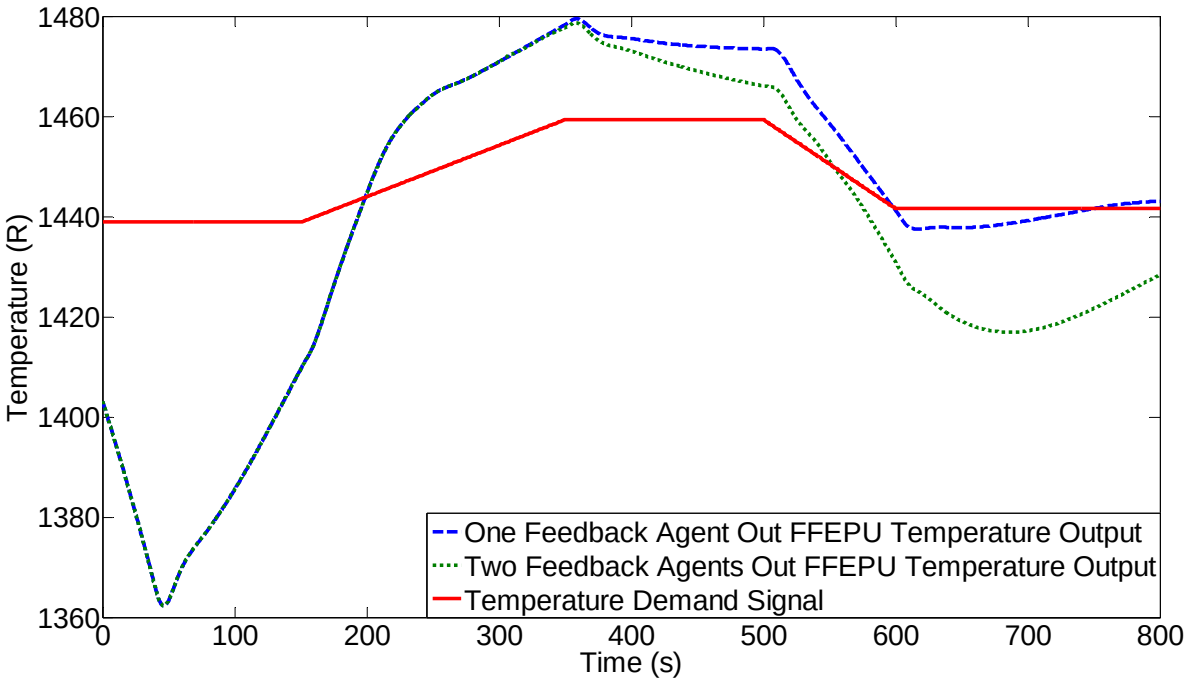

Fig. 64: Adding Feedwater Feedback Agent Failure FFEPU Temperature Output.

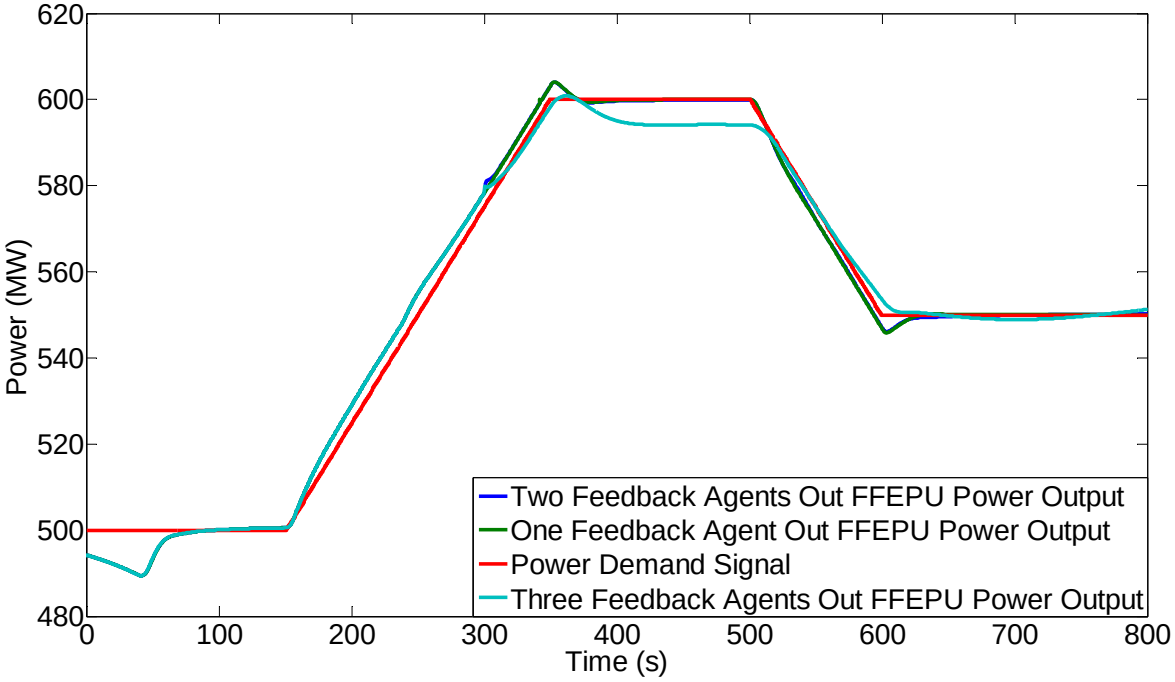

Fig. 65: Adding Turbine Feedback Agent Failure FFEPU Power Output.

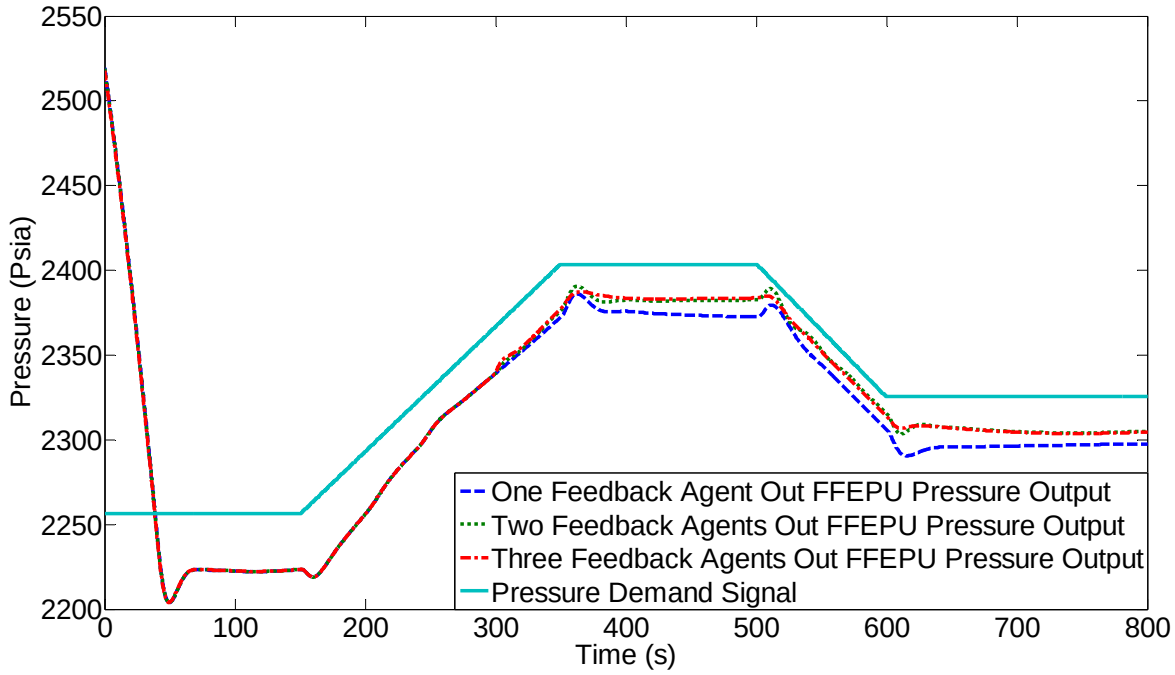

Fig. 66: Adding Turbine Feedback Agent Failure FFEPU Pressure Output.

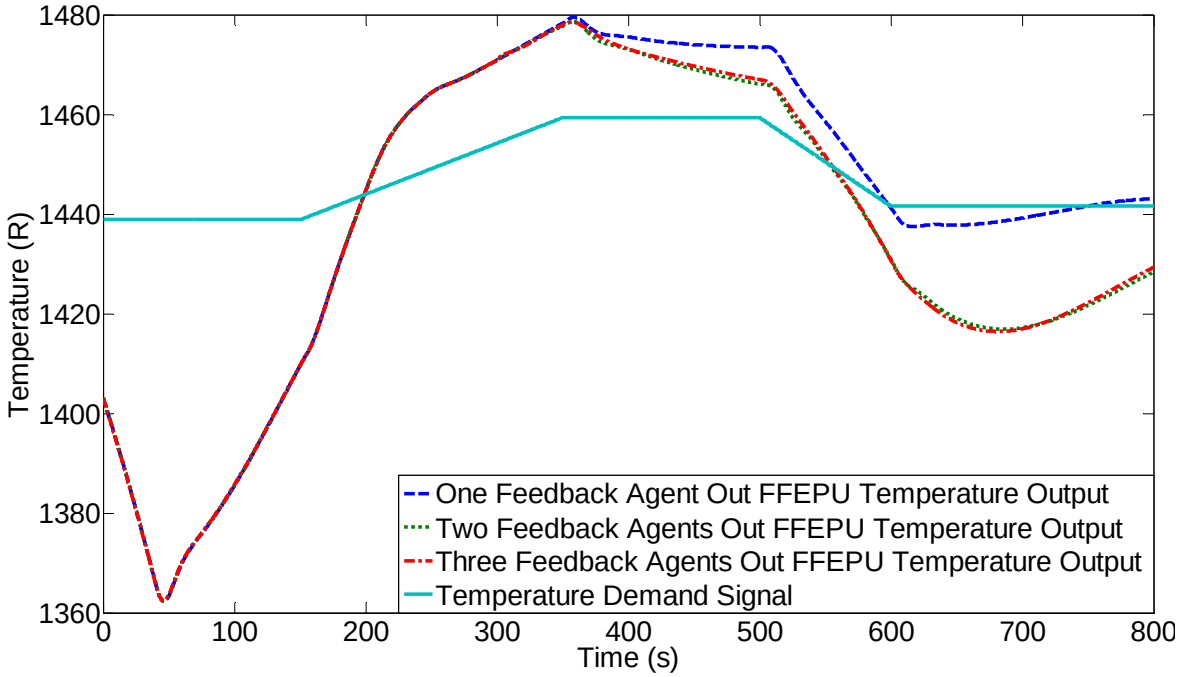

Fig. 67: Adding Turbine Feedback Agent Failure FFEPU Temperature Output.

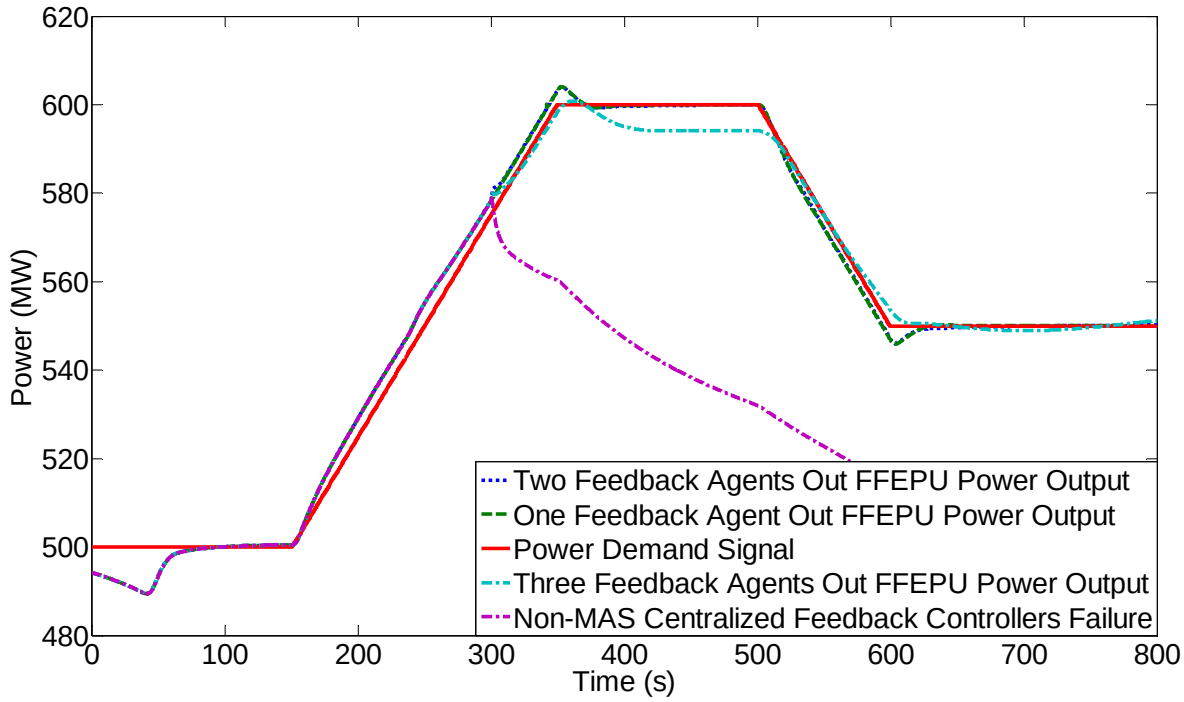

Fig. 68: Comparison to Centralized Control Failure FFEPU Power Output.

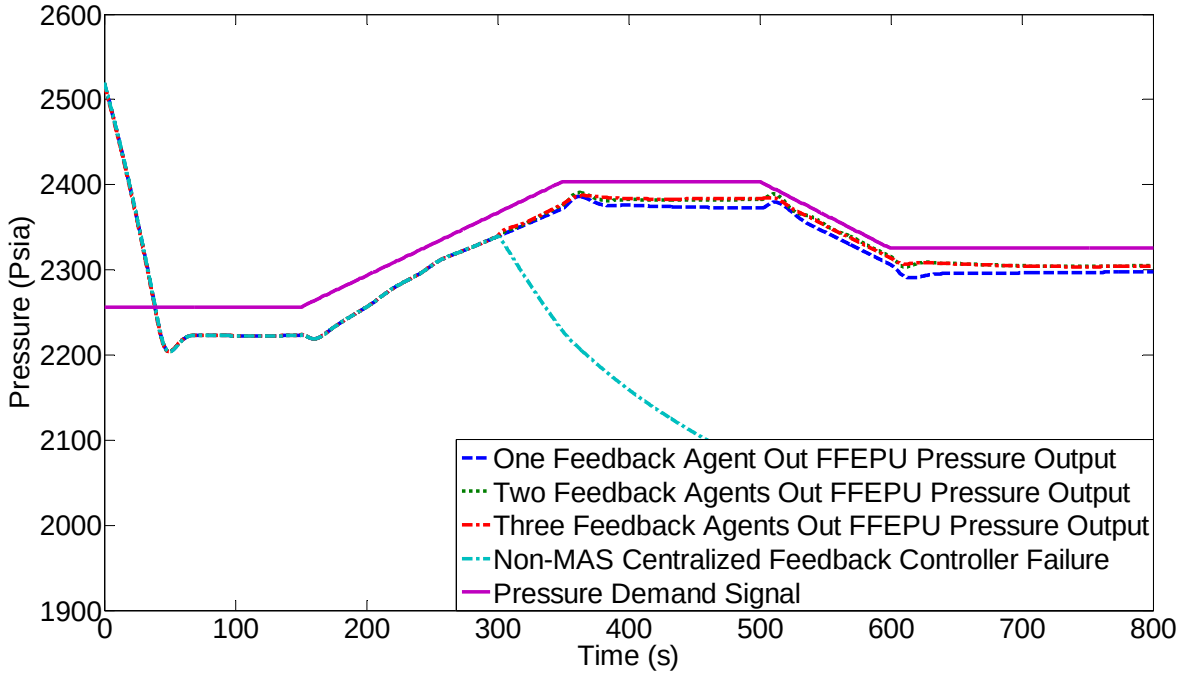

Fig. 69: Comparison to Centralized Control Failure FFEPU Pressure Output.

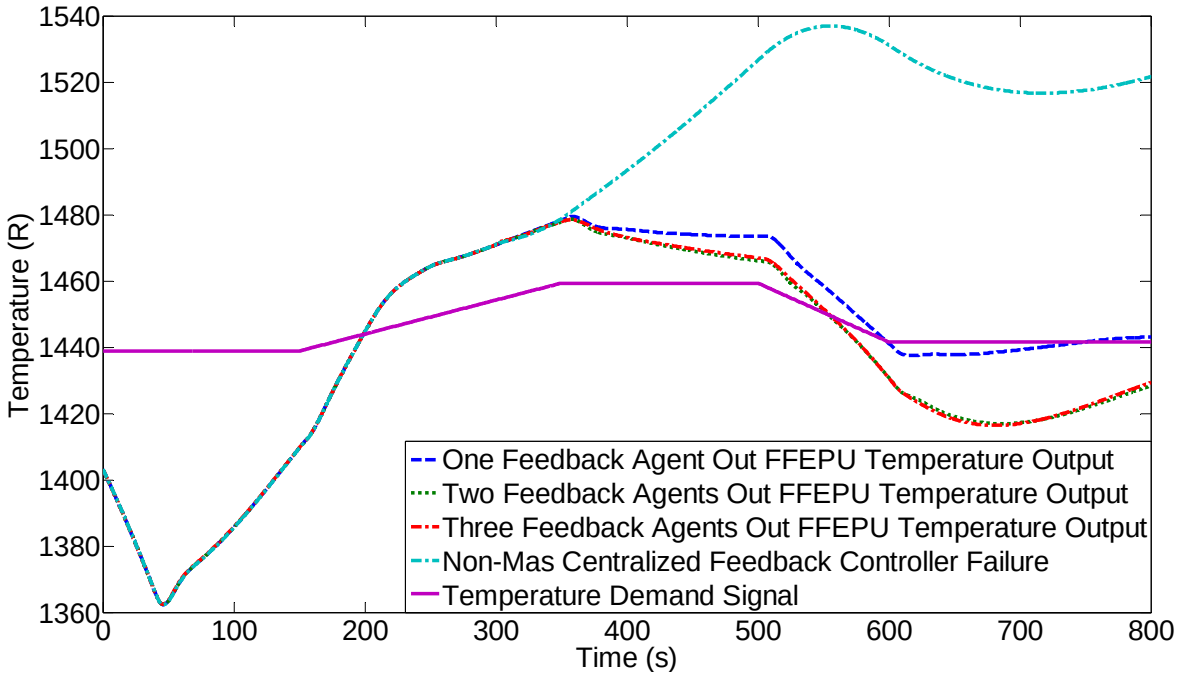

Fig. 70: Comparison to Centralized Control Failure FFEPU Temperature Output.

Finally, to better demonstrate the graceful degradation sequentially and how it affects the FFEPU response, a simulation where several agents are induced into failure must be presented. Specifically, the condenser feedback agent is allowed to fail 150 seconds into the simulation, the feed-water feedback agent fails 50 seconds later and the turbine feedback agent fails 50 seconds later at 250 seconds into the simulation. Additionally, 300 seconds into the simulations, the feed-forward agent is allowed to fail.

Fig. 71 through Fig. 73 display the comparison of sequential agent failure compared the non-MAS FFEPU control implementation. When the Non-MAS system fails, the FFEPU output becomes quickly unstable and is incapable of meet its demand set-points. On the other hand, the JADE-based MAS implementation not only remains relatively stable to the feedback agent failures, but when the feed-forward agent fails, the FFEPU steady state response actually tracks the demand set-points for power, pressure and temperature, showing the robustness and fault tolerance of the overall system.

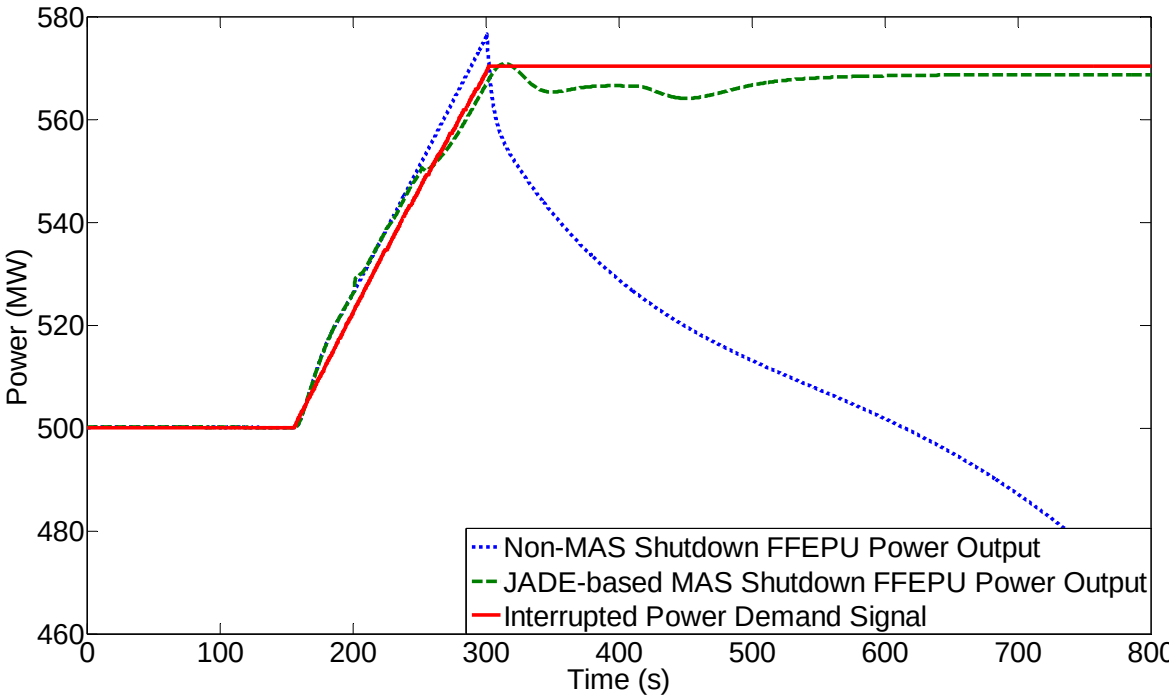

Fig. 71: JADE-based MAS Stage Failure FFEPU Power Output.

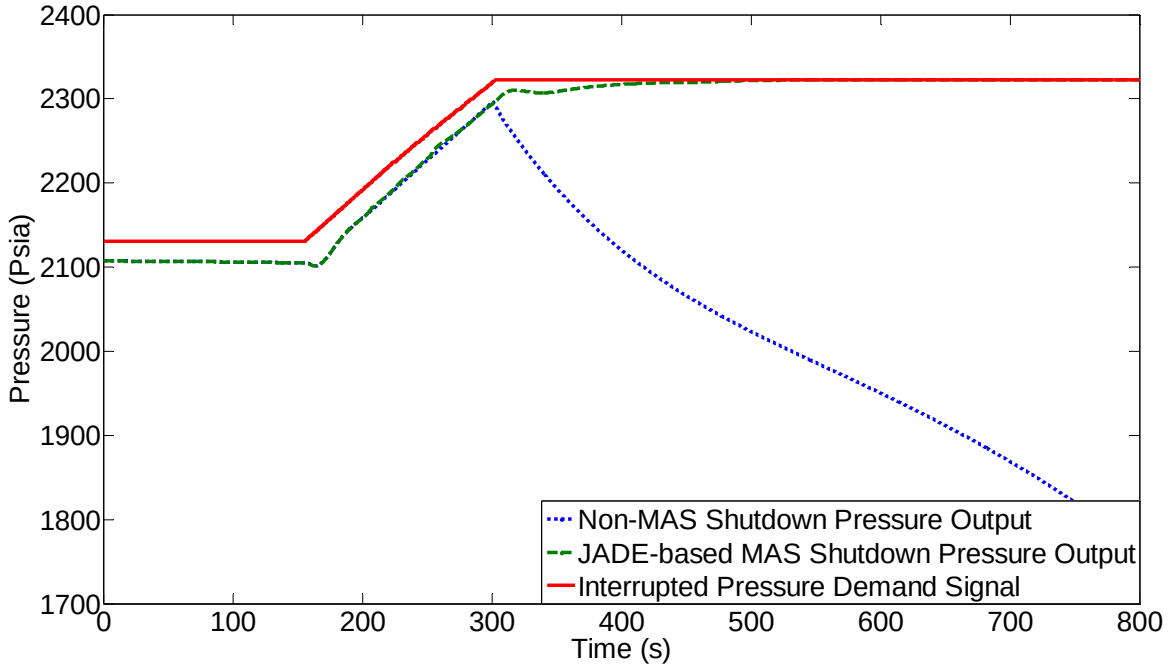

Fig. 72: JADE-based MAS Stage Failure FFEPU Pressure Output.

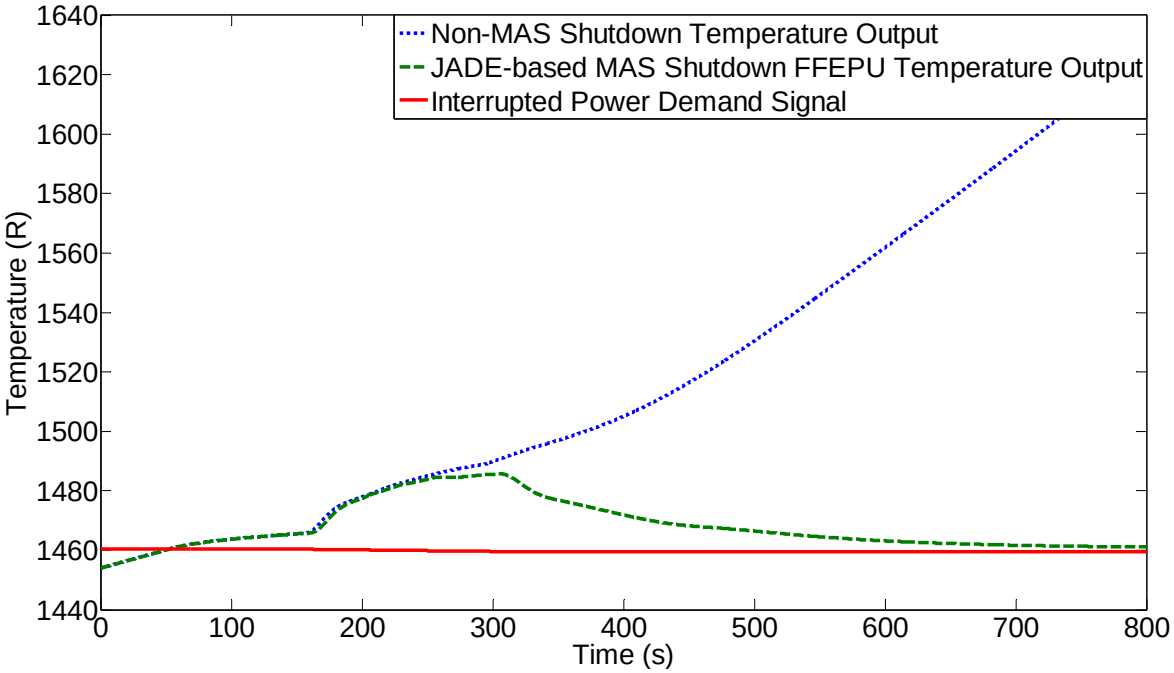

Fig. 73: JADE-based MAS Stage Failure FFEPU Temperature Output.

CHAPTER SEVEN

Conclusions

In this chapter the work and implementations of the JADE-based MAS distributed coordinated control for the FFEPU is summarized, highlighting the most important advantages and conclusions are drawn based on the observations of the performance of the overall system. The chapter concludes with suggestions regarding future research on the topic.

7.1 *Summary and Conclusions*

The main objective of this thesis was to design and implement the control system for a 600 MW FFEPU in a distributed coordinated Multi-Agent System (MAS) in the Java Agent Development (JADE) Framework platform. For such a large complex system, there were several requirements that had to be met. The overall control approach must be robust, flexible, adaptive, autonomous and social. The control system must also be able to implement multiple objectives optimization to address different operating goals that may change with market demand or increasing regulations. The control system is also required to implement online feedback gains optimization to account for unsuitable FFEPU performance.

The 600 MW FFEPU under consideration is described and its model presented. The four modules that comprised and their physical subsystems are presented to better understand the system in question. The need to address multiple objectives is addressed in chapter three. The motivation for a reference governor is explained and the different

steps for its implementation are described. Multi-objective optimization is presented to achieve the goal of tackling of different objectives while still maintain an adequate FFEPU power output response.

The feedback PI controllers and process are described for each of the modules of the FFEPU. The motivation for the optimizing of the feedback PI gains is presented and the gain optimization processes are described within each one of the FFEPU feedback modules.

The motivation for the use of a Multi-Agent System and an overview is given into the MAS and the agent, the basic structure and member of the MAS, is presented. The overall organization of the different agents is described and the requirements for the MAS communication.

JADE is presented as the platform for the implementation of the FFEPU MAS distributed coordinated control. JADE is chosen over other platforms for the focus of the platform on agent communication and the implementation of ACL messaging, which simplifies the communication between the agents and offers a more efficient MAS interaction.

The implementation of the MAS is carried out in a hybrid platform. JADE is the platform used to implement the agent architecture and to perform most of the non-computationally demanding tasks. MATLAB is the software used by the some of the low level agents to perform the computationally burdensome tasks. An intra-agent communication between the JADE agents and its MATLAB counterparts is established by making use of the UDP/IP communication protocol. UDP/IP offers relatively fast communication by broadcasting datagram packets instead of having to establish a

sender/receiver communication, such as the one required by the TCP/IP communication protocol.

Finally, the overall implementation is validated through simulation of the implemented JADE-based MAS system. The proposed approach is able to target different multiple optimization objectives while implement a distributed coordinated control. Furthermore, the implementation of an online gain optimization is successfully illustrated in simulations and is also compared to the response of the implementation of the MAS using MATLAB entirely as a platform. Lastly, the JADE-based MAS is able to exhibit graceful degradation by simulating the failure of different agents that affect directly the control of the FFEPU. By displaying graceful degradation, the JADE-based displays superiority in its autonomy and overall robustness. This is particularly important to maintain an adequate level of cyber security, since the MAS is impervious to sudden failure.

7.2 *Future Work*

The main objective for this thesis is to prove that a JADE-based MAS distributed coordinated control is suitable for a 600 MW FFEPU. To this purpose, it is important to address challenges encountered during the implementation of the proposed approach and consider issues that may need to be address in the near future.

The ability of the JADE-based MAS to be fault tolerant suggests that further steps can be taken to implement a more robust system. JADE has many benefits as a platform, one of which is the relatively few issues with the implementation of new agents. Since JADE already implements the FIPA-ACL, the addition of different agents creates little disturbance in the overall performance of the MAS. It is possible to include the addition

of agents that address the correct functioning of the MAS, such as the implementing of fault diagnosis, elaborate on the capability of agents to remotely operate the processes of the MAS, or even focus on making the overall MAS less prone to cyber attacks.

In this thesis, the agents were implemented in JADE but some of the computational tasks were undertaken in the MATLAB environment. Since the inter-agent communication protocol used is UDP/IP, it is clear that software different than MATLAB can be used to overtake the computational intensive tasks. A particular challenge in the proposed approach was convergence of the feedback gain optimizers to find better PI gain candidates. This is due to the fact that computational intensive tasks are time demanding. The gain optimizing agents might benefit of the implementation of their algorithms in a faster framework. The hybrid approach of using a real-time digital simulator (RTDS) may improve the effectiveness of the gain optimizer agents by allowing a faster convergence to an optimized solution.

Finally, the JADE-based MAS focus solely on the FFEPU power output response with a given unit load demand curve. To make the MAS control more realistic and mimic a real world scenario, it would be interesting to couple the FFEPU power output to the grid. Such implementation would definitely set the FFEPU to a more challenging MAS control structure that would also take into account external power sources as well and a variable power demand.

APPENDICES

APPENDIX A

FFEPU Dynamic Equations

A.1 State Relations [19]

Drum:

$$\rho_{drw} = 49.271 - 2.137\rho_{drs} + 0.0335\rho_{drs}^{\ 2}$$
$$h_{drw} = 526.596 + 31.044\rho_{drs} - 0.621\rho_{drs}^{\ 2}$$
$$h_{drs} = 1241.71 - 21.34\rho_{drs} + 0.21\rho_{drs}$$
$$P_{drs} = 11.188 + 500.27\rho_{drs} - 26.403\rho_{drs}^{\ 2} + 0.469\rho_{drs}^{\ 3}$$
$$T_{drs} = 458.08 + 48.209\rho_{drs} - 3.233\rho_{drs}^{\ 2} + 0.0725\rho_{drs}^{\ 3} + 459.67$$

Downcomer

$$h_{dc} = 658.45 + 11.887\rho_{dc} + 3.95e^{-2}P_{dc} - 5.67e^{-4}\rho_{dc} * P_{dc} - 0.3417\rho_{dc}^{\ 2} - 3.297e^{-7}P_{dc}^{\ 2}$$
$$T_{dc} = 163.6 + 29.199\rho_{dc} + 4.268e^{-2}P_{dc} - 7.263e^{-4}\rho_{dc} * P_{dc} - 0.457\rho_{dc}^{\ 2} + 3.183e^{-7}P_{dc}^{\ 2}$$

Primary and Secondary Super-heaters

$$P_{pso}(P_{sso}) = -291.36 - 964.04\rho_{pso}(\rho_{sso}) + 0.218h_{pso}(h_{sso}) + 1.182\rho_{pso} * h_{pso}(\rho_{sso} * h_{sso})$$
$$T_{pso}(T_{sso}) = -1745.1 + 129.1\rho_{pso}(\rho_{sso}) + 1.811h_{pso}(h_{sso}) - 0.0663\rho_{pso} * h_{pso}(\rho_{sso} * h_{sso})$$
$$S_{pso}(S_{sso}) = 1.3136 - 1.7799e^{-3}\rho_{pso}(\rho_{sso}) + 6.357e^{-4}h_{pso}(h_{sso}) - 8.3591e^{-2}\log\left[\rho_{pso} * h_{pso}\right]$$

Super-heater Spray

$$\rho_{ssl} = -1.903 + 1.386e^{-3}h_{ssl} + 6.767e^{-3}P_{pso} - 3.766e^{-6}P_{pso} * h_{ssl}$$
$$T_{ssl} = -1654.5 + 1.7443h_{ssl} + 0.348P_{pso} - 2.074e^{-4}P_{pso} * h_{ssl} + 459.67$$

High Pressure Turbine

$$\eta_{isen} = E_{hp} = 0.589 + 2.317e^{-4} W_{hp}$$
$$h_{hpol} = -485.23 + 1065.28\, S_{sso} + 0.232 P_{hpo}$$
$$T_{hpo} = -1639.24 + 0.119 P_{hpo} + 1.682 h_{hpo} + 459.67$$

Reheat Spray

$$\rho_{rhl} = -4.66e^{-2} + 3.089e^{-5} h_{rhl} + 6.292e^{-3} P_{hpo} - 3.489e^{-6} P_{hpo} * h_{rhl}$$

$$T_{rhl} = -550.34 - 0.394 h_{rhl} + 1.68 P_{hpo} - 1.136e^{-3} P_{hpo} * h_{rhl} + 9.361e^{-4} {h_{rhl}}^2 - 4.125e^{-4} {P_{hpo}}^2 + 2.085e^{-10} {P_{hpo}}^2 * {h_{rhl}}^2$$

Reheater

$$P_{rho} = 27.061 - 1019.1 \rho_{rho} - 1.735e^{-2} h_{rho} + 1.228 \rho_{rho} * h_{rho}$$
$$T_{rho} = -2013.4 + 189.65 \rho_{rho} + 1.963 h_{rho} - 9.305e^{-2} \rho_{rho} * h_{rho} + 459.67$$
$$S_{rho} = 1.5015 + 3.8306e^{-3} \rho_{rho} + 6.4181e^{-4} h_{rho} - 0.1094 \log\left[\rho_{rho} * h_{rho}\right]$$

Crossover Pipe

$$\eta_{isen} = E_{ip} = 0.814$$
$$h_{ipoi} = -1211.8 + 683.58 \rho_{cro} + 1384.39\, S_{rho}$$
$$h_{ipo} = h_{ipoi} - E_{ip}\left(h_{ipoi} - h_{rho}\right)$$
$$h_{cro} = h_{ipo}$$
$$P_{cro} = -381.05 + 0.2783 h_{cro} + 668.61 \rho_{cro}$$
$$T_{cro} = -2074.92 + 2.004 h_{cro} + 71.326 \rho_{cro} + 459.67$$

Condenser

$$\rho_{cnw} = 62.345 - 0.289P_{cn}$$
$$\rho_{cns} = 2e^{-5} + 3.21e^{-3}P_{cn} - 3.2e^{-4}P_{cn}^{\ 2} + 8e^{-5}P_{cn}^{\ 3}$$
$$h_{cns} = 1075.87 + 29P_{cn}$$
$$h_{cnw} = -25.803 + 365.636P_{cn} - 878.108P_{cn}^{\ 2} + 1305.2P_{cn}^{\ 3} - 1002.14P_{cn}^{\ 4} + 305.01P_{cn}^{\ 5}$$
$$T_{cn} = 6.292 + 364.609P_{cn} - 866.922P_{cn}^{\ 2} + 1311.46P_{cn}^{\ 3} - 1012.18P_{cn}^{\ 4} + 309.494P_{cn}^{\ 5} + 459.67$$
$$h_{lpo} = h_{cnw} + Q_{ylpo}\left(h_{cns} - h_{cnw}\right)$$
$$\rho_{lpo} = \rho_{cnw} + Q_{ylpo}\left(\rho_{cns} - \rho_{cnw}\right)$$

Condenser Pumps

$$h_{cpo} = 2668.5 + 24.79\rho_{cno} - 1.738P_{cpo} + 0.0273P_{cpo} * \rho_{cno} - 1.075\rho_{cno}^{\ 2} + 1.027e^{-4}P_{cpo}^{\ 2}$$
$$T_{cpo} = -7918.5 + 366.8\rho_{cno} - 1.734P_{cpo} + 0.0271P_{cpo} * \rho_{cno} - 3.829\rho_{cno}^{\ 2} + 1.044e^{-4}P_{cpo}^{\ 2} + 459.67$$

Low Pressure Feedwater Heater

$$\rho_{lho} = 62.633 + 2.309e^{-4}P_{lho} - 7.847e^{-3}h_{lho} + 8.453e^{-7}P_{lho} * h_{lho} - 1.308e^{-7}P_{lho}^{\ 2} - 4.417e^{-5}h_{lho}^{\ 2}$$
$$T_{lho} = 31.363 - 3.68e^{-3}P_{lho} + 1.022h_{lho} + 2.258e^{-6}P_{lho} * h_{lho} + 1.311e^{-6}P_{lho}^{\ 2} - 9.836e^{-5}h_{lho}^{\ 2} + 459.67$$

Deaerator

$$\rho_{dew} = 60.458 - 19.612\rho_{des}$$
$$h_{dew} = 118.263 + 1905.65\rho_{des} - 8414.69\rho_{des}^{\ 2} + 15688.04\rho_{des}^{\ 3}$$
$$h_{des} = 1143.5 + 224.56\rho_{des}$$
$$P_{des} = -1.042 + 419.172\rho_{des} + 131.328\rho_{des}^{\ 2}$$
$$T_{des} = 150.489 + 1896.42\rho_{des} - 8447.88\rho_{des}^{\ 2} + 15757.43\rho_{des}^{\ 3} + 459.67$$

Boiler Feedpump

$$h_{fpo} = 9024.9 + 0.1148P_{fpo} - 90.346\rho_{dew} - \frac{2.053e^{-5}}{\rho_{dew}} - 1.81e^{-3}P_{fpo} * \rho_{dew}$$
$$T_{fpo} = -1268.8 + 82.714\rho_{dew} + 0.103P_{fpo} - 1.474e^{-3}P_{fpo} * \rho_{dew} - 0.9707\rho_{dew}^{\ 2} - 1.801e^{-6}P_{fpo}^{\ 2} + 459.67$$

High Pressure Feed-water Heater and Economizer

$$\rho_{hho}\left(\rho_{eco}\right)=71.143-5.09e^{-3}P_{hho}\left(P_{drs}\right)-3.266e^{-2}h_{hho}\left(h_{eco}\right)+1.169e^{-5}h_{hho}*P_{hho}\left(h_{eco}*P_{drs}\right)$$
$$+6.166e^{-7}{P_{hho}}^{2}\left({P_{drs}}^{2}\right)-3.063e^{-5}{h_{hho}}^{2}\left({h_{eco}}^{2}\right)-2.31e^{-12}{h_{hho}}^{2}*{P_{hho}}^{2}\left({h_{eco}}^{2}*{P_{drs}}^{2}\right)$$
$$T_{hho}\left(T_{eco}\right)=146.98-0.0885P_{hho}\left(P_{drs}\right)+0.7972h_{hho}\left(h_{eco}\right)+1.803e^{-4}h_{hho}*P_{hho}\left(h_{eco}*P_{drs}\right)$$
$$+1.048e^{-5}{P_{hho}}^{2}\left({P_{drs}}^{2}\right)-1.823e^{-4}{h_{hho}}^{2}\left({h_{eco}}^{2}\right)-3.812e^{-11}{h_{hho}}^{2}*{P_{hho}}^{2}\left({h_{eco}}^{2}*{P_{drs}}^{2}\right)$$

Low Pressure Feedwater Heater

$$h_{3lhso}=32.5+43.498P_{3lhs}-7.38{P_{3lhs}}^{2}+0.5147{P_{3lhs}}^{3}$$
$$\rho_{3lhs}=62.345-0.2888P_{3lhs}$$
$$T_{3lhs}=64.059+43.602P_{3lhs}-7.402{P_{3lhs}}^{2}+0.5162{P_{3lhs}}^{3}+459.67$$

High Pressure Feedwater Heater

$$h_{3hhso}=126.874+4.154P_{3hhso}-0.0422{P_{3hhso}}^{2}+1.8e^{-4}{P_{3hhso}}^{3}$$
$$\rho_{3hhso}=60.532-0.046P_{3lhs}$$
$$T_{3hhso}=159.096+4.129P_{3hhso}-0.0434{P_{3hhso}}^{2}+1.8e^{-3}{P_{3hhso}}^{3}+459.67$$

A.2 Nomenclature

Prefixes

K ≡ A constant
C ≡ A control system variable
K_c ≡ A control system constant
K_{tc} ≡ A control system time constant
Z ≡ Intermediate variable

Primary Quantity

A,a ≡ Area, fan vane position
E,e ≡ Effeciency
F,f ≡ Friction factor, fraction
G,g ≡ Accleration due to gravity

G_c, g_c ≡ Units conversion factor
H,h ≡ Enthalpy, heating value
J,j ≡ Moment of inertia
L,l ≡ Length
M,m ≡ Mass
MW ≡ Power
N,n ≡ Number, speed, frequency
P,p ≡ Pressure
Q,q ≡ Heat transfer rate
R,r ≡ Density
S,s ≡ Slip, entropy, specific heat
T,t ≡ Torque, temperature; temperature factor
U,u ≡ Heat transfer coefficient, specific internal energy
V,v ≡ Volume, voltage
W,w ≡ Mass flow rate
X,x ≡ Burner tilt, position, length, water level
η ≡ Efficiency
ρ ≡ Density
ϕ ≡ Function
δ ≡ Power Angle

Components

ah ≡ Air heater (glycol)
ap ≡ Air pre-heater
ar ≡ Air
at ≡ Atmosphere
bp ≡ Booster feed pump
cn ≡ Condenser
cp ≡ Condensate pump
cr ≡ Cross-over pipe
cv ≡ Governor control valve
cw ≡ Condensate (water)
dc ≡ Downcomer
de ≡ Deaerator
dr ≡ Drum
dv ≡ Deaerator valve
ec ≡ Economizer
fd ≡ Forced draft fan

fl ≡ Fuel
fn ≡ Furnace
fp ≡ Boiler feed pump
ft ≡ Feed pump turbine
fv ≡ Feed-water valve
fw ≡ Feed-water
gg ≡ Gun
gn ≡ Generator
gr ≡ Gas recirculation
gv ≡ Governor control valve
hp ≡ High pressure turbine
hh ≡ High pressure feed-water heater
id ≡ Induced draft fan
ip ≡ Intermediate pressure turbine
iv ≡ Intercept valve
lh ≡ Low pressure feed-water heater
lp ≡ Low pressure turbine
ps ≡ Primary super-heater
rh ≡ Re-heater
rp ≡ Recirculating pump
rw ≡ Recirculating water
ry ≡ Re-heat spray
sc ≡ Steam chest
sh ≡ Super-heater
ss ≡ Secondary super-heater
st ≡ Stack
sy ≡ Superheat spray
sv ≡ Stop valve
tr ≡ Turbine
tv ≡ Throttle valve
vf ≡ Forced draft fan vane
vi ≡ Induced draft fan vane
ww ≡ Waterwall

Conditions

a ≡ Air
bd ≡ Blowdown
c ≡ Convective heat transfer

d ≡ Difference, drop, change
e ≡ Effective, average
elec ≡ Electrical
g ≡ Gas, gun
i ≡ Inlet condition, isentropic process
isen ≡ Isentropic
in ≡ In, inlet
L ≡ Lower limit
m ≡ Metal, motor
max ≡ Maximum
o ≡ Outlet condition
out ≡ Out, outlet
pu ≡ Per unit
r ≡ Radiation, rated, ratio
s ≡ Steam, supply line, seal
sr ≡ Steam return
u ≡ Upper limit
v ≡ Valve
w ≡ Water
x ≡ Extraction
l ≡ Inlet condition

Exceptions

ldc ≡ Load demand computer signal
q_{vww} ≡ Quality of steam leaving waterwall
q_{ylpo} ≡ Quality of steam leaving low pressure turbine
y_{wgr} ≡ Water to gas ratio of flue gas
w_g ≡ Flue gas flow rate

APPENDIX B

Heuristic Optimization Algorithms

In order to implement the main control modules of the FFEPU control system, such as the reference governor and gain optimizer, heuristic optimization techniques that can solve multi-objective problems and perform multi-dimensional random searches are employed.

Multi-objective problems and approaches to solving them have been around for many years; however, implementing these solutions take time and are not applicable for real-time performance in a large-scale FFEPU [7]. For the FFEPU, in particular, analytic mathematical programming [33] and genetic algorithms [34] were used as solutions to the multi-objective problem facing large-scale power plants but both take a long time to reach convergence and thus are not practical for online implementation [7], [34].

B.1 Particle Swarm Optimization

The PSO algorithm was developed by Eberhart and Kennedy and is based on the analogy of a swarm of birds and school of fish [35], [36]. The algorithm replicates the behavior of the swarm computationally and has been used for solving other nonlinear continuous optimization problems [7].

In a basic two-dimensional PSO algorithm, the position and velocity of each bird or particle is given an x-y component in the two-dimensional space. The goal of the algorithm is to simulate birds flocking towards the minimization of an objective function,

which is in the search space. Each bird or particle in the swarm has three pieces of information at all times, which are its current position, its best position so far, and the best position of the group of all the particles in the swarm. In the algorithm, the particle's best position is known as its personal best (*pbest*) and the best position of the group of all the particles in the swarm is known as the global best (*gbest*). This information is an analogy of the personal experience of each particle, and is what is used by the particle to update its position towards what it thinks is the right path towards the global minimum.

The particle updates its position based on a velocity correction that consists of three terms that each reflects a different portion of the particle's experience in the swarm. The velocity equation below is formulated as the following:

$$v_i^{k+1} = wv_i^k + c_1 \cdot \text{rand}_1 \times (pbest_i - s_i^k) + c_2 \cdot \text{rand}_2 \times (gbest - s_i^k) \tag{9}$$

where v_i^k is the velocity of particle i at iteration k, w is a weighting function, c_1 and c_2 are weighting factors, rand_1 and rand_2 are random numbers between 0 and 1, s_i^k is the current position of particle i at iteration k, $pbest_i$ is the *pbest* of particle *i*, and *gbest* is the best value of all the *pbests* in the swarm [21]. The first term allows for change in the search and reflects the particle's tendency to explore new areas in the search space. The second and third terms reflect the particle's tendency to move in the best direction towards the global minimum, a combination of its *pbest* and *gbest* at the current iteration. The three terms in (4) have been shown to be a well-balanced approach to updating particles in the search procedure [15], [36], [37]. The weighting function *w* used in (9) is often formulated as the following:

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \tag{10}$$

where $w_{\max}$ is the maximum initial weight given by the user, $w_{\min}$ is the minimum initial weight given by the user, $iter_{\max}$ is the maximum iteration number, and *iter* is the current iteration number. Using (9) and (10), the particles' velocities during the search procedure are updated with the most current information and the particles' positions are updated with the following equation:

$$s_i^{k+1} = s_i^k + v_i^{k+1} \tag{11}$$

During the optimization, with the use of (9), (10), and (11), the particles are brought closer to their personal best values and the global best value [7], [34].

As a result, with online implementation in mind, the introduction of algorithms such as particle swarm optimization (PSO) and other evolutionary computation, have been shown to be superior to traditional methods in solution quality, convergence rate, and computational complexity [7], [34]-[38].

B.2 Hybrid Particle Swarm Optimization

In an ordinary PSO, the particles' position updates depend heavily on the particles' personal best positions (*pbests*) and swarm's global best position (*gbest*). Because of this, the search area can often be reduced, increasing the likelihood of particles getting trapped in local minima. The HPSO seeks to remedy this problem by combining elements from the original PSO method with elements from evolutionary computation (EC) [7], [34]. With the addition of a natural selection mechanism to particles' position updates, the dependence on *pbest* and *gbest* is gradually reduced as the population of particles evolves during the search, equivalently expanding the overall search area [7]. In this natural selection mechanism, half of the particles with the best

performance replace the other half of the particles with the worst performance. Each particle, however, still retains information about their personal best location (*pbest*) so far during the search regardless of the natural selection mechanism. The performance is determined through user-defined objective functions that are minimized by the PSO. In the FFEPU case, there are thirteen objective functions which reflect several conflicting operating requirements of the plant. With this hybrid approach, the particles have information that allow them to search effectively in their current area and at the same time retain information about a past high-performance search area [5], [7]. This HPSO search procedure is applied to the multi-objective optimization problem for the large-scale FFEPU, in which thirteen objectives have to be minimized in order for the reference governor to generate optimized set-points for power, pressure, re-heater and super-heater temperatures.

APPENDIX C

Power Plant Identifiers

The reference governor, one of the main control elements in the FFEPU control system, requires a steady-state mathematical representation of the plant in order to generate optimized set-point trajectories for power, pressure, re-heater temperature and super-heater temperature.

C.1 Artificial Neural Networks

Artificial neural networks (ANNs) are primarily used as tools to model complex system relationships between inputs and outputs. They were originally developed as tools for one of the main goals of artificial intelligence, which was to develop artificial systems that could perform intelligent tasks like learning similar to tasks performed by the brain [39], [40]. In an effort to replicate the intelligent learning process of the brain, ANNs were developed with the concept of the structure of the brain in mind. In terms of structure, ANNs mimic the brain in the following ways:

- Each neural network consists of multiple layers of hidden artificial neurons similar to what we consider to be the large decentralized network of neurons that make up the human brain. Within these hidden layers of artificial neurons, are an input layer that perceives information, and an output layer;
- Within the hidden layers, each artificial neuron is interconnected with the other artificial neurons that make up the neural network. This structure models the way neurons in the brain communicate with each other through neurotransmitters;

- The strength of each interconnection is determined through interconnection weights that represent the size or area of synapses in the human brain.

To mimic the process of learning, an ANN first senses information through neurons that make up the input layer [39], [40]. This information is usually presented as a set of inputs with a corresponding set of desired outputs [19]. The ANN then processes that information through the various interconnections of neurons that make up the hidden layer and output layer. The network constantly modifies the synaptic weights between the neurons until the ANN has "learned" the relationship between the inputs and outputs.

There are many different types of ANNs depending on the desired application. The many different variations differ from each other based on the interconnection pattern between the layers of neurons, the learning process for updating the neurons' interconnection weights, and the transfer function that converts the network's input to its corresponding output [39]. The Feed-forward neural network (FFNN) is used to obtain a steady-state model of the FFEPU, which is needed by the reference governor to generate optimized set-point trajectories.

Neuron Model

The artificial neurons are an analogous representation of the biological neurons that makeup the human brain. As such, the artificial neuron is the basic building block from which the entire ANN is built. Mathematically, each artificial neuron can be represented as the result of three distinct functional processes applied to a single input. These processes that make up the mathematical representation of an artificial neuron consist of a multiplication operation, addition operation, and a scalar operation [40], [6].

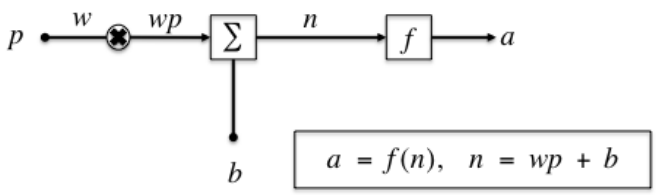

Fig. C.1: Architecture of Single Neuron [6].

When presented with a scalar input *p*, the first operation takes place with the weight *w* to form the scalar product *wp* [40]. In the second operation, *wp* is added to the scalar bias, *b*, to form the net input *n*. The scalar bias, *b*, is constant and equal to 1 and functions as a shift value to the transfer function *f*. In the third operation, the net input *n* is processed through the transfer function *f*, which produces the scalar output *a*. These three operations are referred to as the weight function, net input function and transfer function, respectively. The weight function, net input function, and transfer function differ depending on the type of network and desired network function.

The neuron shown in Fig. 74 is a representation of a single neuron with one input. However, a typical neuron in an artificial neural network can have many inputs. Moreover, a typical ANN usually has several layers of many neurons that are interconnected. Consequently, to better understand the mathematical representations of the neuron model in the larger context, Fig. 75 is used as the general neuron model [6].

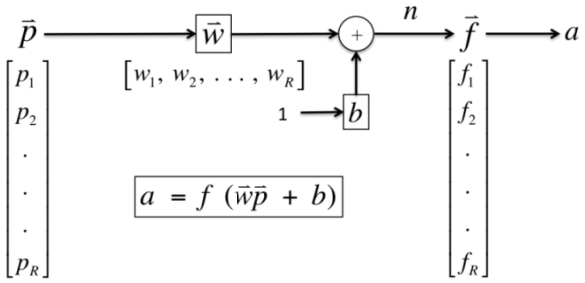

Fig. C.2: General Neuron Model [6].

In Fig. 75, the input $\vec{p}$ is a *R*-dimensional row vector and $\vec{w}$ is a *R*-dimensional column vector, where *R* represents the number of elements in the input vector $\vec{p}$. The variable *n* is the net input, which is formed according to the following:

$$n = \vec{w}\vec{p} + b \tag{12}$$

Fig. 75 also represents the architecture of a single layer of neurons. In a layer representation, the dimensions of the inputs, weights, biases, net inputs, and output vectors change. For a *R*-dimensional row vector $\vec{p}$, input weights (*IW*) are represented by a *S* x *R* matrix, and biases, net inputs, and outputs are each represented by a *S*-dimensional row vector, where *S* is for the number of neurons.

ANN Multiple-Layer Architecture

Artificial neural networks usually consist of multiple layers, each with a weight matrix (*LW*), bias vector $(\vec{b})$, and output vector $(\vec{a})$, as described in the section above.

The feed-forward neural network is one of the simplest neural networks, with a multi-layered architecture. Input weights (*IW*) are weights directly connected to the input vector $\vec{p}$, whereas layer weights (*LW*) are weights directly connected to the output vector $\vec{a}$ of the previous layer. For each weight, *IW* or *LW*, an index is assigned to denote the source and destination of the corresponding weight. The first index denotes the destination of the weight, and the second index denotes the source of the weight. Likewise, in order to distinguish the element variables in the different layers, each element variable is assigned a superscript to denote the corresponding layer number it belongs to [40]

Shown in Fig. 76, the feed-forward neural network can consist of one or more hidden layers and an output layer, though only one hidden layer is usually needed. In a FFNN, information flows in only one direction; there are no feedback inputs or recursive loops. For function approximation, the transfer function of the hidden layer(s) is usually a hyperbolic tangent sigmoid function and the transfer function of the output layer is usually a linear transfer function. With these transfer functions, the feed-forward network has been shown to be very good at non-linear function approximation [6], [32], [41]-[42].

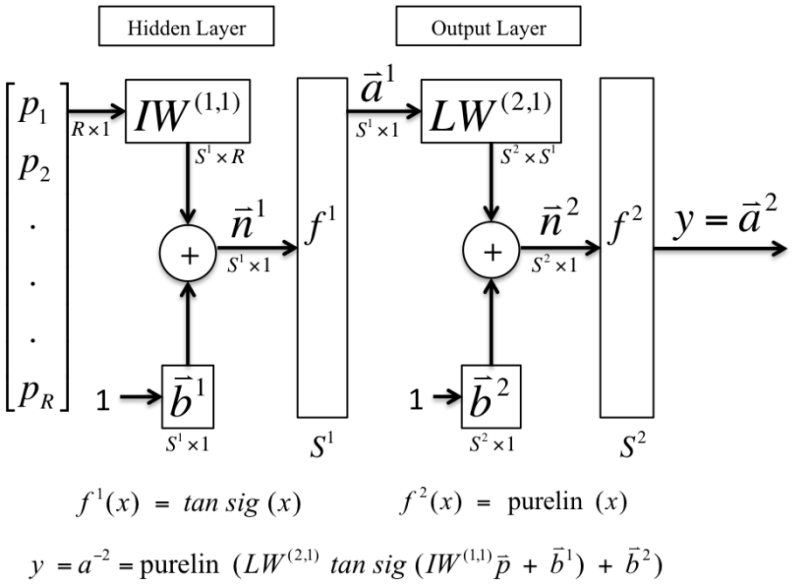

Fig. C.3: Feed-forward Neural Network (FFNN) Architecture [6].

C.2 FFEPU Steady-State Model

The need for a steady-state model is reflected in the overall design goal of the FFEPU control system, which is based in a coordinated boiler-turbine control scheme. Under this scheme, the boiler and turbine are controlled simultaneously through a common unit load demand (E_{uld}). To meet the requirements of this scheme, the reference

governor generates four set-points (power, pressure, re-heater temperature, and super-heater temperature) that reflect the common E_{uld}. However, these four set-point demands only reflect the desired output of the FFEPU in steady-state. The reference governor is also responsible for generating the twelve control valve actions $(u_1, u_2, \ldots, u_{12})$ that correspond to the four set-point demands (E_d, P_d, RT_d, ST_d). In order to do this, the reference governor needs to be able to map control inputs $(u_1, u_2, \ldots, u_{12})$ to steady-state plant outputs (E_d, P_d, RT_d, ST_d). Therefore, in order to successfully map inputs to outputs, a steady-state model of the plant is needed each for power, pressure, re-heater temperature, and super-heater temperature.

Four FFNNs are used to develop steady-state models that correspond to power, pressure, re-heater temperature, and super-heater temperature set-points. Each FFNN has a similar architecture to the general FFNN shown in Fig. 76, and consists of two layers, one hidden and one output. For each FFNN, the number of neurons in the hidden layer was determined by:

$$S^{hidden} = 2 \cdot R + 1, \tag{13}$$

which was proposed in [43], [44] as the minimum number of neurons necessary to obtain good performance in a typical FFNN. As a result of (13), each FFNN effectively has the same configuration with twelve inputs, twenty-five neurons in the hidden layer, one neuron in the output layer, and one output corresponding to one of the four respective set-point demands. The number under input, and output denotes the number of elements. Likewise, the number under hidden layer and output layer denotes the number of neurons located in each layer.

APPENDIX D

User's Manual

This document gives insight into the set up and implementation of the JADE-based MAS. The manual helps the user and it describes the steps of the process to run the program already described.

D.1 JADE Set up

In order to set JADE it is necessary to download the compressed JADE files provided on the http://jade.tilab.com website. For the implementation of the JADE-based MAS it is necessary to choose an Integrated Development Environment (IDE). The IDE chosen was Eclipse.

After downloading JADE-all-xxx.zip, where xxx is the version number, user should extract the files contained in it to some path on disk drive. JADE's all-zip files contain a lot of folders, the one we are interested in is the one named as JADE-bin-xxx. The path that we are going to use as JADE jar file-path is '*<PATH_ON_THE_DISK_DRIVE>\JADE-bin-xxx\jade\lib*', it may also be referenced as JADE classpath. This file-path/classpath contains the necessary jade.jar file that we are going to reference while compiling our code.

In order to follow the setup instructions, it is assumed that the user has downloaded the Eclipse from its website and installed it and have installed the minimum necessary JRE-version to be used with above downloaded JADE version. The following

steps will help setup the project to use the JADE files for development. These steps have been written based on the options provided in Eclipse Helios:

1. Open Eclipse
2. In the File menu, choose New -> JavaProject
3. Give a name to the project e.g. MAS
4. Click Next button at the bottom
5. Click on Libraries tab
6. Click on ‘Add External Jars’ button
7. Select the location of jade.jar file from the classpath (specified previously)
8. Click on Order and Export tab and then -> Select jade.jar -> click on Finish button

The above steps will create a project by name as provided in step-3, this project will now have the necessary reference to the jade.jar file. Optionally when required the commons-codec-xxx.jar can be added from the same classpath by repeating steps 6 to 8 with this .jar file. Now, we need to create necessary configurations to launch the Agent Management System from within Eclipse. The following steps will help set those instructions:

1. In the Run menu, choose Run Configurations
2. Double-click on Java Application option. This will give the new options on the right pane.
3. In the Main Class option type jade.Boot
4. Then select Include System Libraries When Searching For a Main Class option
5. Click on the Arguments tab

6. Type –gui <NAME_OF_AGENT>:<AGENT_CLASS_NAME>, where <NAME_OF_AGENT> should be string that you want to name your agent object and <AGENT_CLASS_NAME> should be the class name of the agent class.

7. Click on Apply and then click on Run button

The above Run button click will start the jade system and then will automatically launch the Remote Agent Mangement GUI.

To run the JADE-based MAS distributed coordinated control, the source code needs to be added to the workspace. To do this, the associated documents described in the next section need in the .java format and added to the source file. The general location is 'H:\JADE_MAS\Workspace\MAS\src\edu\baylor\ece\mas' or depending on the main file location.

The purpose of the Baylor ECE MAS project is to handle all the agents via the delegation agent. This means that one agent needs to create/assign tasks to other agents. But it was not possible to refer to an agent object once that agent has been created. To overcome this situation another abstract class Agent has been created in the package *edu.baylor.ece.mas.agent*. This agent class extends the *jade.core.Agent* class and modifies its function in such a way that at the time of initialization, the agent registers its own reference in the *edu.baylor.ece.mas.agent.AgentFactory* class. This Agent class provides a method agentAction() which needs to be implemented in order to get the desired agent behaviour. To run the MAS it is required to start the delegation agent, which is created by the interface agent. Therefore, in step 6, the argument to be run is actually of the form *-gui dd:edu.baylor.ece.mas.agent.InterfaceAgent.*

For Baylor ECE's MAS project, it's recommended that the Agents should be created with the provided Agent Framework classes i.e. the Agent class provided in *edu.baylor.ece.mas.agent* namespace.

BIBLIOGRAPHY

[1] *Peace and War, United States Foreign Policy 1931–1941*, Washington D.C.: United States Government Printing Office, 1943, retrieved 2007-12-08

[2] Annual Energy Outlook 2012 with Projections to 2035, United States (U.S) Energy Information Administration (EIA), 2012. Available: http://www.eia.gov/forecasts/archive/aeo12/index.cfm

[3] Annual Energy Review 2011, United States (U.S) Energy Information Administration (EIA), 2011. Available: http://www.eia.gov/totalenergy/data/annual/pdf/aer.pdf

[4] Annual Energy Outlook 2013 Early Release Overview, United States (U.S) Energy Information Administration (EIA), 2012. Available http://www.eia.gov/forecasts/aeo/er/executive_summary.cfm

[5] P. B. Usoro, "Modeling and Simulation of a Drum Boiler-Turbine Power Plant under Emergency State Control," M.S. Thesis, Dept. Mech. Eng., Massachusetts Institute of Technology, Cambridge, MA, 1977.

[6] C.S. Williams, "Design and Implementation of a Multi-Agent Optimized Control System for a Large-Scale Fossil-Fuel Electrical Power Unit," M.S. Thesis, Dept. Electrical and Computer Engineering, Baylor University, Waco, TX, 2011.

[7] J. S. Heo, K. Y. Lee and R. Garduno-Ramirez, "Multiobjective Control of Power Plants Using Particle Swarm Optimization Techniques," *IEEE Trans. Energy Conversion*, vol. 21, no.2, pp. 552-561, Jun. 2006.

[8] J. S. Heo and K. Y. Lee, "A Multi-Agent System-Based Intelligent Heuristic Optimal Control System for a Large-scale Power Plant," *IEEE Congress on Evolutionary Computation*, Vancouver, BC, Canada, pp. 5693-5699, Jul. 2006.

[9] H. C. Gery, "The Evolution of Coordinated Control," Inst. Soc. Amer. Power Symp., St. Petersburg, FL, paper no: 88-0417, pp. 109-112, 1988.

[10] J. S. Heo and K. Y. Lee, "A Multi-Agent System-Based Intelligent Control System for a Power Plant," *IEEE Power Engineering Society General Meeting*, San Francisco, CA, paper code: CD PESGM2005-000858.pdf, 2005.

[11] L. Padgham and M. Winikoff, *Developing Intelligent Agent Systems*, John Wiley & Sons, 2004.

[12] J. S. Heo and K. Y. Lee, "A Multi-Agent System Based Intelligent Reference Governor for Multi-objective Optimal Power Plant Operation," *IEEE Transactions on Energy Conversion,* vol. 23, no. 4, 2008.

[13] K. I. Wang, W. H. Abdulla, and Z. Salcic, "A Multi-Agent System for Intelligent Environments Using JADE," IEEE Seminar on Intelligent Building Environments. Colchesters, UK, 28 June 2005.

[14] M. Nikraz, G. Caire, and P. Bahri, A Methodology for the Analysis and Design of Multi-Agent Systems using JADE, Telecom Italia Lab, May 2006.

[15] J. P. McDonald, H. G. Kwatny, and J. H. Spare, "A Non-Linear Model for Reheat Boiler-Turbine Generator Systems: Part I – General Description and Evaluation," *in Proc. 12th Joint Automatic Control Conf.*, pp. 219-226, 1971.

[16] H. G. Kwatny, J. P. McDonald, and J. H. Spare, "A Non-Linear Model for Reheat Boiler-Turbine Generator Systems: Part II – Development," *in Proc. 12th Joint Automatic Control Conf.*, pp. 227-236, 1971.

[17] J. D. Head, "Development and Implementation of a Multi-Agent System for Intelligent Optimized Power Plant Control" M.S. Thesis, Dept. Electrical and Computer Engineering, Baylor University, Waco, TX, 2012.

[18] S. J. Al-Nasurs, "Simulation of a Fossil-fuel Power Plant Unit in Matlab Environment," M.S. Thesis, Dept. Elect. Eng., Pennsylvania State Univ., University Park, PA, 2001.

[19] J. S. Heo, "A Multi-Agent System-Based Intelligent Control for Large-Scale Power Plants," Ph.D. dissertation, Dept. Elec. Eng., Pennsylvania State University, University Park, PA, 2006.

[20] W. Junpu, C. Hao, X. yang, L. Shuhui, "An Architecture of Agent- Based Intelligent Control systems," in *Proc. 2000 the 3rd World Congress on Intelligent Control and Automation,* pp. 404-407.

[21] K. Price, R. Storn, and J. Lampinen, *Differential Evolution – A Practical Approach to Global Optimization (Natural Computing Series).* Springer, Berlin Heidelberg, 2005.

[22] J. S. Heo and K. Y. Lee, "A Multi-agent System Based Intelligent Steady-State Model for a Power Plant," *in Proc. 13th International Conf. Intelligent Systems Application to Power Systems,* Washington D.C., pp. 419-424, 2005.

[23] J. D. Head, J. R. Gomes, C. S. Williams, and K. Y. Lee, "Implementation of a Multi-Agent System for Optimized Multiobjective Power Plant Cnotrol," *in Proc. of the 2010 North American Power Symposium,* Arlington, Texas, Sept. 2010.

[24] J. R. Velasco, J. C. Gonzalez, L Magdalena, and C. A. Iglesias, "Multiagent-Based Control Systems: A Hybrid Approach to Distributed Process Control," *Control Engineering Practice,* pp. 839- 845, 1996.

[25] R. Garduno-Ramirez, "Overall Intelligent Hybrid Control System for a Fossil-Fuel Power Unit," Ph.D dissertation, Dept. Elec. Eng., Pennsylvania State University, University Pak, PA, 2000

[26] M. Wooldridge, G. Weiss, Ed., "Intelligent Agents," in *Multi-agent Systems*. Cambridge, MA: MIT Press, Apr. 1999, pp. 3-51.

[27] Foundation for Intelligent Physical Agents (FIPA), *FIPA ACL Message Structure Specification,* 2002. [Online]. Available: http://www.fipa.org/specs/fipa00061/SC00061G.html.

[28] Foundation for Intelligent Physical Agents (FIPA), *FIPA Content Language Specifications.* 2003. [Online]. Available: http://www.fipa.org/repository/cls.php3.

[29] Foundation for Intelligent Physical Agents (FIPA), *FIPA Communicative Act Library Specification,* 2002. [Online]. Available: http://www.fipa.org/specs/fipa00037/SC00037J.html.

[30] *Java Agent Development Framework (JADE),* [Online]. Available: http://jade.cselt.it/.

[31] F. Bellifemine, G. Caire, T. Trucco, G. Rimassa *JADE Programmer's Guide*, 2010 [Online]. Available: http://jade.tilab.com/doc/programmersguide.pdf

[32] R. Garduno-Ramirez and K. Y. Lee, "Multiobjective Optimal Power Plant Operation through Coordinate Control with Pressure Set-point Scheduling," *IEEE Trans. Energy Conversion,* vol. 16, no. 2, pp. 115-122, Jun. 2001.

[33] J. Chang, K. Y. Lee, and R. Garduno-Ramirez, "Optimal task decomposition agents of the multiagent power plant control system," *in Proc. Int. Conf. Intelligent Syst. Appl. Power Syst.,* 2003, CD ISAP03-085.pdf.

[34] J. S. Heo and K. Y. Lee and R. Garduno-Ramirez, "Multiobjective optimal power plant operation using particle swarm optimization technique," *in Proc. IFAC Congress,* Prague, 2005, paper code: 04833.pdf, Tu-M06-TO/4.

[35] L. Kennedy and R. Eberhart, "Particle swarm optimization," *in Proc. IEEE International Conf. Neural Networks,* Perth, Australia, vol. IV, pp. 1942-1948, 1995.

[36] E IEEE Catalog Number 02TP160, *Tutorial on Modern Heuristic Optimization Techniques With Applications to Power Systems*, K. Y. Lee and M. A. El-Sharkawi, Eds. Piscataway, NJ: IEEE Power Engineering Society, 2002.

[37] P. Angeline, "Using selection to improve particle swarm optimization," in *Proc. IEEE International Conference on Evolutionary Computation*, pp. 84-89, 1998.

[38] J. –B. Park, K. –S. Lee, J. –R. Shin, and K. Y. Lee, "A particle swarm optimization for economic dispatch with non-smooth cost functions," *IEEE Trans. Power Syst.*, vol. 20, no.1, pp. 34-42, Feb. 2005.

[39] *Intro to Neural Networks*, Neuro-Solutions, [Online]. Available: http://www.neurosolutions.com/products/ns/whatisNN.html.

[40] *Artificial Neural Network*, Wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Artificial_neural_network

[41] R. Garduno-Ramirez and K. Y. Lee, "A Multiobjective-Optimal Neuro-Fuzzy Extension to Power Plant Coordinated Control," *Trans. The Institute of Measurement and Control*, vol. 24, no.2, pp. 159-192, 2002.

[42] H. Ghezelayagh and K. Y. Lee, "Intelligent Predictive Control of A Power Plant with Evolutionary Programming Optimizer and Neuro-Fuzzy Identifier," *in Proc. 2002 Congress on Evolutionary Computation*, vol.2, pp. 1308-1313.

[43] C. C. Ku, K. Y. Lee, and R. M. Edwards, "Improved Nuclear Reactor Temperature Control Using Diagonal Recurrent Neural Networks," *in IEEE Transactions on Nuclear Science*, vol. 39, no. 6, pp. 2298-2308, Dec. 1992.

[44] C. Ku and K. Y. Lee, "Diagonal Recurrent Neural Networks for Dynamic Systems Control," *in IEEE Transactions on Neural Networks*, vol. 6, no.1, pp. 144-156, Jan. 1995.