INTEGRATED APPROACHES TO THE OPTIMIZATION OF PROCESS-UTILITY SYSTEMS

A Dissertation

by

NASSER AHMED AL-AZRI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2008

Major Subject: Chemical Engineering

INTEGRATED APPROACHES TO THE OPTIMIZATION OF PROCESS-UTILITY SYSTEMS

A Dissertation

by

NASSER AHMED AL-AZRI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Mahmoud El-Halwagi
Committee Members,	Guy Curry
	Juergen Hahn
	M. Sam Mannan
Head of Department,	Michael Pishko

December 2008

Major Subject: Chemical Engineering

ABSTRACT

Integrated Approaches to the Optimization of Process-Utility Systems. (December 2008)

Nasser Ahmed Al-Azri, B. Eng. Sultan Qaboos University;

M.S. Texas A&M University;

M. Eng. Texas A&M University

Chair of Advisory Committee: Dr. Mahmoud M. El-Halwagi

The goal of this work is to develop a conceptual framework and computational tools for the optimization of utility systems in the process industries. The emphasis is devoted to the development of systematic design techniques aimed at identifying modifications to the process and the associated utility-systems to jointly optimize the process and the utility system. The following contributions describe the specific results of this work:

- Development of shortcut methods for modeling and optimizing steam systems and basic thermodynamic cycles with the objective of using these methods in the optimization of combined heat and power. To enable efficient mathematical programming formulations, simple yet accurate correlations have been developed for the thermodynamic properties of steam in the utility system.
- Optimization of multi-level steam system for combined process requirements and power cogeneration. A general procedure is developed to determine rigorous cogeneration targets and the optimal configuration of the system with the associated design and operating variables.

- Graph theory methods are also used to optimize the pipeline layout in the plant for the distributing the utilities.
- Finally, because of the nonconvex nature of much of the developed optimization formulations, a global optimization method has also been suggested by using interval analysis and simulated annealing.

The techniques proposed in this work are compared to previous works and their applicabilities are presented in case studies. These techniques outperform previously suggested ones in terms of the accuracy, computational efficiency and/or optimality.

DEDICATION

To my parents…

Fatima Khalid and Ahmed Zahran

ACKNOWLEDGEMENTS

I would like to thank my parents for their care. Being away from them is the only thing that my education could never offset.

I owe my deepest gratitude to my advisor Dr. Mahmod El-Halwagi whose support was not limited to his guidance in this work; my words will never convey my deep gratitude.

I also want to thank the committee members Dr. Guy Curry, Dr. Juergen Hahn and Dr. M. Sam Mannan for their time and input on this work. I thank Dr. Sergiy Butenko for substituting for Dr. Curry in the defense of this work and for his help and feedback on chapter VIII of this dissertation.

I am very thankful to the members of the Process and Integration Group and special thanks to my office mates: Metab Al-Otaibi, Musaed Al-Thubaiti, Abdullah Binmahfouz and Benjamin Cormier.

I am very grateful to Sultan Qaboos University for sponsoring my education.

My thanks and gratitude would have gone at length if the space permitted. Even if the many others are not mentioned here by name, my thoughts never missed imagining how things could have been without their support and help.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

LIST OF TABLES

CHAPTER I

INTRODUCTION

In the chemical process, the utility system is the heart of the process without which the whole process would be totally paralyzed. Consequently, the design and construction of the utility system possess a major consideration. Furthermore, the utility system has a substantial impact on the environment, and the economics and capacity of the plant. These factors necessitate a design that is setup and evaluated very carefully.

Almost every single unit in a chemical plant requires energy to operate. The design of the utility system is a complicated task. The plant usually has different kinds of energy-driven units working at different loads and operation scheduling. The usage of energy may not be steady throughout the year or even the day and so the design of such a system needs a very careful planning that counts for all these variances. An optimum design is that which squares with the exact needs of a plant without shortage or excess. This scenario is almost impossible due to the high unpredictability involved in chemical plants. Risk management under uncertainties is an indispensable approach to alleviate any negative consequences. This will necessitate introducing standby units which will add to the cost.

Owing to the global calls for energy conservation and awareness on depletion of the natural resources, energy integration has proved being one of the most reliable solutions. Not only can energy integration save energy, it also contributes to

This dissertation follows the style and format of *Journal of Energy Engineering*.

environmental preservation by reusing the energy that would contribute to pollution if not reused.

The energy needed by the process can be in the form of power, heating and cooling. These three components are the master players in the energy balance of the process. Since the most substantial cost of running a typical process is owed to the energy requirements, designers has recently started integrating and interchanging the energy components in its different forms so as to reduce the external needs.

There are many forms of energy integration; the most prevalent are heat and power. Heat integration was first proposed by Linnhof and Hindmarsh (1983) to who the advent of this concept is attributed. Heat integration is very fundamental to this work; it is the step preceding the design of the utility system. In heat integration, instead of purchasing cooling and heating utilities separately, the heating/cooling processes are carried out from within by exchanging heat between the process streams. The synthesis of heat exchange network (HEN) is a systematic procedure that achieves the optimum matching between the process streams. After the optimum has been achieved, the rest will be the minimum possible utility needed by the process. Also, due to safety and geographical constrains, integrating two streams may not be a viable solution and hence some streams will be excluded from the integration procedure or might be integrated as another separate group.

Energy integration may also involve heat streams. Heat is one kind of energy and it can be transformed into work, through steam turbines for instance. The idea of reusing heat in producing power has introduced a new concept in energy conservation, the so-called cogeneration. Cogeneration has recently gained tremendous attention especially in countries where energy resources are scarce. It allows production of power through shaft-work from waste heat. Through utilization of waste heat, cogeneration upgrades the plant efficiency to 90 % or more and electricity saving of 15-40 % (EDUCOGEN and INESTENE 2001) .

In addition to the economical benefit of cogeneration, it also has an environmental impact which is a lower emission of gases especially CO_2.

Cogeneration was first introduced in early 20th century. In spite of its relatively slow growth, it is a very promising alternative to the conventional systems both in terms of environmental protection as well as energy saving. In addition, the usually unexpected oil price crises have prompted governments, especially in the European Union, to seek efficient energy generation systems and hence promoting cogeneration as a substitute. In 2001, the share of electricity produced by cogeneration in Europe was about 10%. A share of 18% is targeted by 2010. The belief in the achievement of this target is supported by the fact that three European countries, namely the Netherlands, Finland and Denmark had cogeneration as a share of national power production in 2001 of 40, 35 and 50 % respectively (EDUCOGEN and INESTENE 2001).

The methods discussed in this work are all based on the thermodynamic properties of steam. Having accurate correlations of the steam properties can increase the programmability and the traceability of the procedure in optimizing and targeting utility systems. Moreover, using the thermodynamic correlations will help visualize the system more clearly on Mollier (T-s) diagram and will give more accessibility to checking on the quality of the system at the utility outlet. The thermodynamic properties of steam will be developed for the operable range of the steam turbine within an acceptable accuracy compared to the customary correlations developed by ASME (IFC-ASME 1967).

Integration lifts the concept of efficiency to a more process-centered paradigm. Instead of looking at the efficiency of each unit, the deficiencies (wastes) of the individual units could be utilized and used within the process in order to have a better overall efficiency of the whole process. Rankine cycle consists of four main blocks, namely a boiler, a steam turbine, a condenser and a pump. The overall efficiency of Rankine cycle is about 45% at best. A substantial loss is due to the heat discharged at the condenser. This heat can be reused by supplying it to the process and hence will be counted in the overall process efficiency.

When cogeneration is implemented, the steam is produced at high pressure levels, let down through different steam mains at lower pressure values while producing power during the expansion, and finally they are exported to the process. The essential purpose of the steam in this case is to satisfy the process requirement, having expanded the steam through the steam turbines, however, will alter the thermodynamic properties of the steam at the exit of the utility system. The alterations are due to the expansion and the mechanical deficiencies of the turbine. Consequently, deciding the right amount of steam to be produced in the boiler will become a hectic task. The steam quality required by the process will be the one at the exit of the utility. Thus, the steam in the boiler needs to be produced so as to count for the alteration and deliver it at the exit of the utility just as required by the process.

Raissi (1994) developed the temperature-enthalpy (T-H) method. In this model, for any given inlet pressure, the heat load at the exit of the turbine is assumed constant. This assumption might be reasonable at low exit pressure values but it gets so unreliable at higher values. Moreover, the T-H model did not account for the variability of the turbine efficiency with the load. Mavromatis and Kokossis (1998) developed the turbine hardware model. It is an iterative procedure that helps estimating the mass requirement and accounts for efficiency variations. Although it is better than Raissi's (1994), the error, as will be discussed in a later section, can reach 10%. Moreover, the procedure doesn't give a direct check on the quality of the steam at the exit of the turbine.

Harell (2004) used the concept of extractable energy. A heat balance for the different mains is performed where the surplus hearers provide steam to the deficit ones wile power is being produced by the turbine while transferring the steam. Harell (2004) suggested a pinch diagram similar to the thermal one, in which the maximum extractable power is decided.

Targeting for the utility system takes place at definite pressure levels where the mass flow rates need to be identified. The pressure levels might not however be the optimal in terms of utilizing the system to generate heat and produce power. Moreover, deciding the pressure level should conform to the requirement of the steam needs by the process. Thus, optimizing the pressure levels needs evaluating the process requirements of steam at the different pressure levels. That task needs performing heat integration.

Before the design of the utility system, minimum utility targets need to be determined using the pinch diagram. In the pinch diagram (El-Halwagi 2003) , for each hot streams the enthalpy exchange represented by equation (1.1.a) is calculated. In a heat-versus-temperature diagram, an arrow whose tail is located at supply temperature (T^s) and head at target temperature (T^t) is drawn with a slope equals to the product of the stream flowrate and the specific heat ($f\ C_p$). These arrows are then arranged in the ascending order of their target temperature i.e. by sliding the arrows vertically while maintaining their vertical position.

$$HC_u = f_u C_{p,u} (T_u^s - T_u^t) \qquad 1.1.a$$

$$HC_v = f_v C_{p,v} (T_v^t - T_v^s) \qquad 1.1.b$$

In the same manner the cold streams are arranged by using equation1.b. For both types of streams, if two streams overlap in their horizontal span, the overlapping segments will be represented by a single segment extending from the lowest point in the bottom segment to the highest point in the top segment as shown in Figure 1.1.

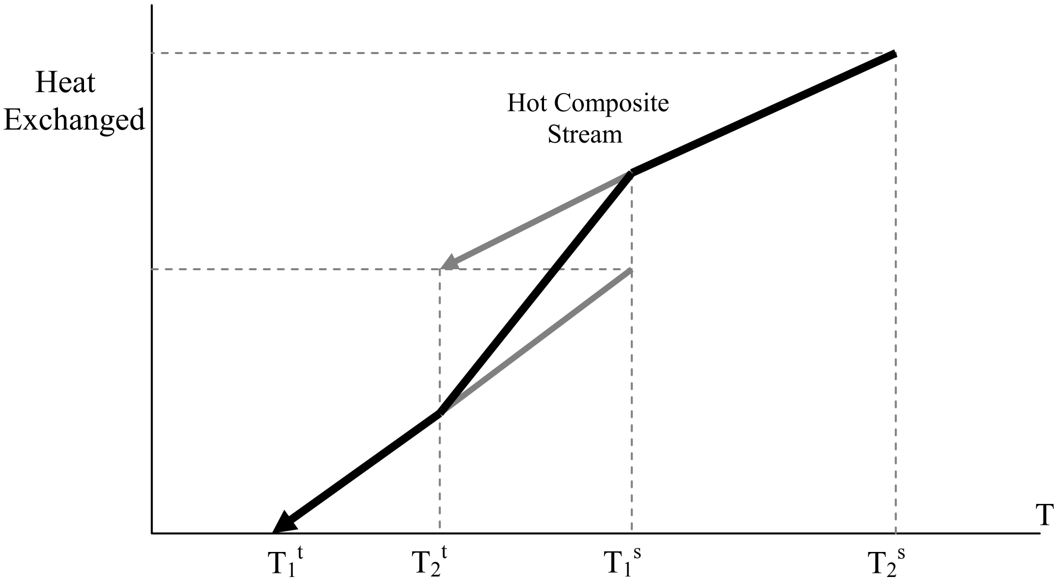

Figure 1.1 Constructing a hot composite stream using superposition (Linnhoff and Hindmarsh 1983)

After both group of streams (hot and cold), are constructed, the cold stream composite will then be slid until it touches the hot composite at a point, without overlapping as shown in Figure 1.2.

Above the pinch point, no cooling utility should be used and below that point, no heating should be used. The minimum heating utility is the one that will be counted for as the heat required by the process in the utility system.

The amount of heat at each level and the minimum temperature are determined from the grand composite curve (GCC) (Figure 1.3). The construction of this curve is described by El-Halwagi (2006).

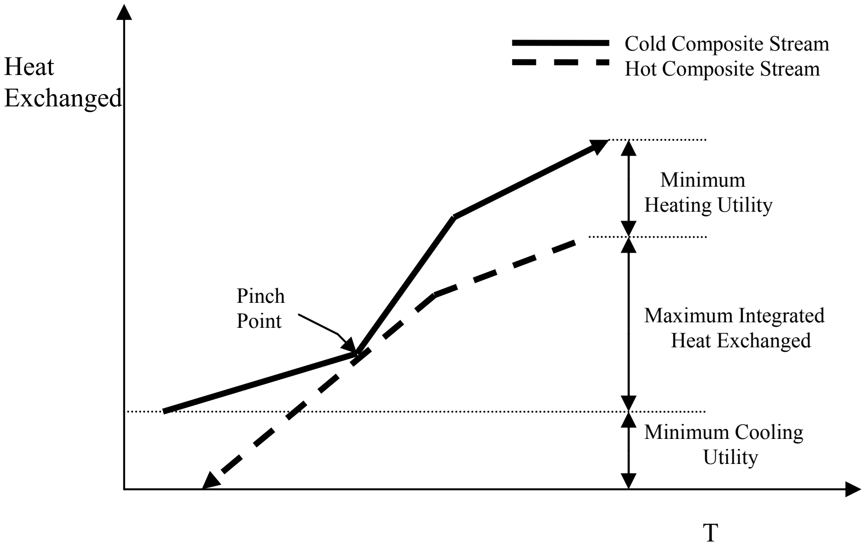

Figure 1.2 Thermal pinch diagram (Linnhoff and Hindmarsh 1983)

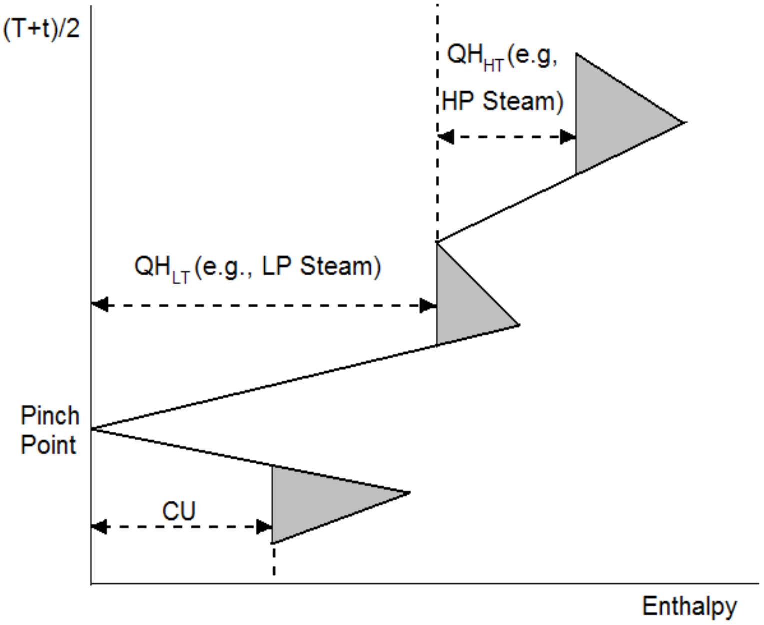

Figure 1.3 The grand composite curve (Linnhoff et al. 1982)

The steam requirement for the non-heating purposes is introduced to the problem after the integration of the heat exchangers. The steam requirement for the non-heating purposes may itself undergo an integration procedure using the mass exchange network. This network is, in many aspects, analogous to the thermal pinch (El-Halwagi 2006).

In order to optimize the steam levels, the necessary heat requirement at each level needs to be identified by using the grand composite curve (Smith 2005). The grand composite curve will serve as the source from which the constraint on the minimum heat requirement is established. Optimizing the steam mains is performed by using a mixed-integer linear/nonlinear programming e.g. (Papoulias and Grossman 1983) and also exhaustive enumeration (Mavromatis and Kokossis 1998).

CHAPTER II

PROBLEM STATEMENT

The overall goal of this work is to develop a conceptual framework and computational tools for the optimization of utility systems in conjunction with the process. The emphasis is on the development of systematic design techniques aimed at identifying process and utility-systems modifications to jointly optimize the process and the utility system. In particular, systematic procedures will be developed to address the following issues:

- Development of shortcut methods for modeling and optimizing steam systems and basic thermodynamic cycles with the objective of using these methods in overall optimization of the utility system and the process. In a single cycle, the attention will be paid to the power produced by the steam as well as the heat discharged by the condenser. Consequently, the heat will be regarded as a product instead of waste. The heat will be dictated by the process and the design will be to optimize the tradeoff between the fuel consumed at the boiler with the power produced by the turbine.
- Optimization of multi-level steam system for combined process requirements and power cogeneration. The power produced by the steam turbine is a function of the mass flowrates, efficiency and the pressure ratio. For a given heat outlet at each steam main, the mass flow at the boiler needs to be determined so as to deliver the exact amount of heat required by the process. A general procedure will be developed which will be flexible so as to incorporate the steam

discharged by the reprocess, assign it to the proper steam main and then calculate the appropriate turbine size and boiler operation parameters.

- Process synthesis of energy-induced systems (e.g., vapor-compression distillation networks). Once the targeting procedure has been developed, the design problem will broaden to include the process. The incorporation of the process in the optimization problem will necessitate optimizing the steam mains' levels in the utility system. The optimization procedure will be outlined by the process requirement of steam on the grand composite curve as well as the requirement of steam for non-heating purposes. After the steam distribution destination has been established, an optimization procedure for the piping network will be presented.

Before tackling these issues, the steam thermodynamic properties will be used. Instead of using the steam tables, the correlation of the properties will be developed for the operable range of the steam turbine within an acceptable error compared to the commonly used ASME correlations (IFC-ASME 1967).

In addition to addressing these issues, a global optimization algorithm will be suggested. This algorithm will be based on interval analysis and heuristics.

CHAPTER III

CORRELATIONS OF STEAM PROPERTIES

Introduction

The main parameters in the design of the utility system are the pressure of each level, mass flowrates and the number of turbines. Before determining these parameters, the optimum requirement of heat needs to be determined through heat integration. Steam has numerous uses in a chemical plant. Heating might be the leading function but its uses extend to other uses that include (Smith 2005):

- team cracking and stripping;
- cleaning vessels;
- lowering the viscosity (to avoid solidification) using steam tracers;
- direct heating of water using steam injection;
- creation of vacuum in steam ejectors;
- combustion processes to atomize fuel oil;
- lowering NO_x emission through reduction in the flame temperature;
- injection into flares to assist the combustion;
- power generation in steam turbines.

Not only is steam a cheap and available substance. It possesses other features that make it superior over other alternatives. In addition to its non-toxicity, it has a wide range of temperature (high critical temperature) and high heat content (high latent heat) which will result in less flowrate inside the heat exchanger.

This fact is depicted in Table 3.1 where steam is compared to other common organic material. In these calculations, a simple heat pump cycle is assumed using isentropic turbine and compressor with the later having a pressure ratio of four.

Table 3.1 Comparison of Steam Performance with Other Common Organic Material in a Simple Heat Pump

	T^c	P^c	Q_{cond}	Q_{Evap}	W_{net}
Material	**(^oK)**	**(bar)**	**(kJ/kg)**	**(kJ/kg)**	**(kJ/kg)**
Steam	674.	221.20	-1949.	1716.	233
Ethylene	283.	50.97	-630.	244.	386.27
Ethane	305.	48.71	-280.	221.	59.29
Methane	191.	45.99	-349.	265.	83.86
Methanol	513.	81.03	-722	667.	54.94

The heat loads in the steam heat exchangers are extraordinarily superb compared to the other materials'. Steam with its superior qualities makes it essential to a chemical plant like water to live species.

The Importance of the Correlations

In the utility system design, the steam correlations are needed for a very limited range. Customarily, designers refer to the steam tables or use complicated correlations. Simple correlations will make the optimization procedure faster and more robust.

Other correlations in use are usually dependent on different variables and also valid for a wide range of pressure and temperature values than needed for a utility system. This feature has rendered them to be very complicated in order to stay accurate. In order to solve for some of the variables, the user will have to find a number of roots to the correlation and then validate one of the roots. This procedure would require additional computational effort to solving the original design problem. These disadvantages urge the need to develop the correlations presented herein.

Not only do the presented correlations void the need for the steam tables, they, in addition, are expressed directly in terms of the minimum possible independent variables with a very acceptable accuracy.

Steam Properties Correlations

According to Gibbs rule, the number of independent variables that must be fixed in order to establish the intensive state of a system is:

$$F=2-\pi+N \tag{3.1}$$

where π is the number of phases, N is the number of chemical species and F is the degree of freedom of the system i.e. the number of independent variables that must be fixed.

Since steam which is virtually single-phase and single-substance is used here, at least two independent variables need to be known at a given state to determine the other variables. One variable that is already provided in the problem statement, is the head pressure or more practically here, the saturation temperature. The additional variable will be either enthalpy or entropy.

In the case of properties on the saturation line, the saturation temperature will suffice in order to determine the required property. Inside the saturation region, however, the knowledge of at least one extensive property (specific volume, internal energy, enthalpy or entropy) and one intensive property (temperature or pressure) is needed. The extensive property is needed to determine the quality of the steam. The quality will then be used with the saturation line values of the required property at the given intensive property.

Computationally-efficient steam-table correlations are central to the effectiveness of any steam-system design and optimization procedure. Correlations of steam properties are developed and compared to the steam tables developed by ASME (IFC-ASME 1967).

It is worth mentioning that these correlations fit the best in the operation range of steam turbines which is up to about 2000 psi and 1000 ^{o}F of superheat.

Since only two independent variables at most are dealt with here, each correlation was performed by first correlating the dependent values with the first independent variable and for different constant values of the second independent variable. Then the correlation constants were correlated with the second independent variables. In most cases, different correlations were tested and then refined by comparing their sums of mean square and relative error with the ASME steam tables.

The correlation procedure was performed using Microsoft® Excel 2003 and Polymath® 5.0.

Enthalpy

The enthalpy in the superheated region is expressed as a function of the saturation temperature of the steam main and the entropy.

$$h = 0.2029\ T_{sat}\ s^{3.647} + 817.35 \qquad (3.2)$$

whereas h in Btu/lb, s in Btu/(lb.R) and T_{sat} in °F. The relative error is about ± 0.7 % (Figure 3.1 and Figure 3.2), and is valid in the range between atmospheric pressure and 2500 psi and a superheat of up to 1000 °F.

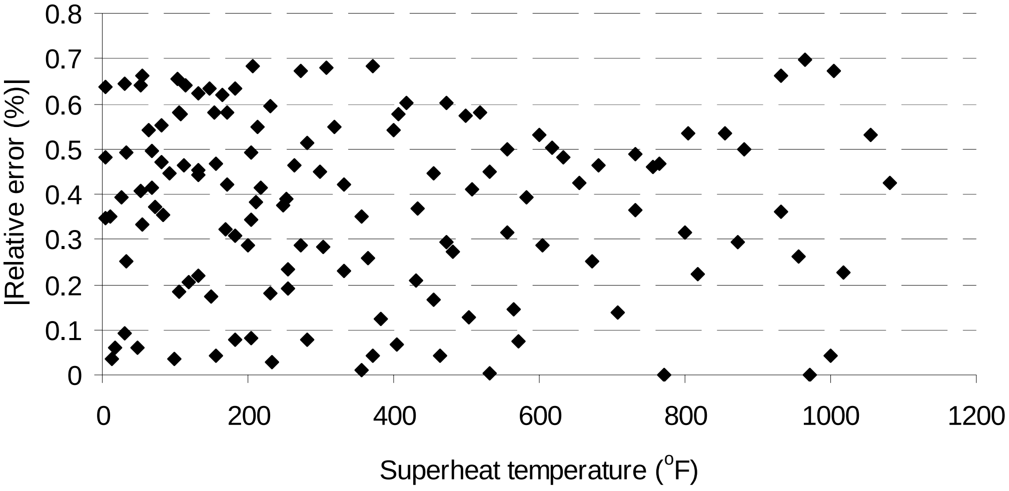

Figure 3.1 The absolute relative error of the enthalpy function at the different superheat temperatures

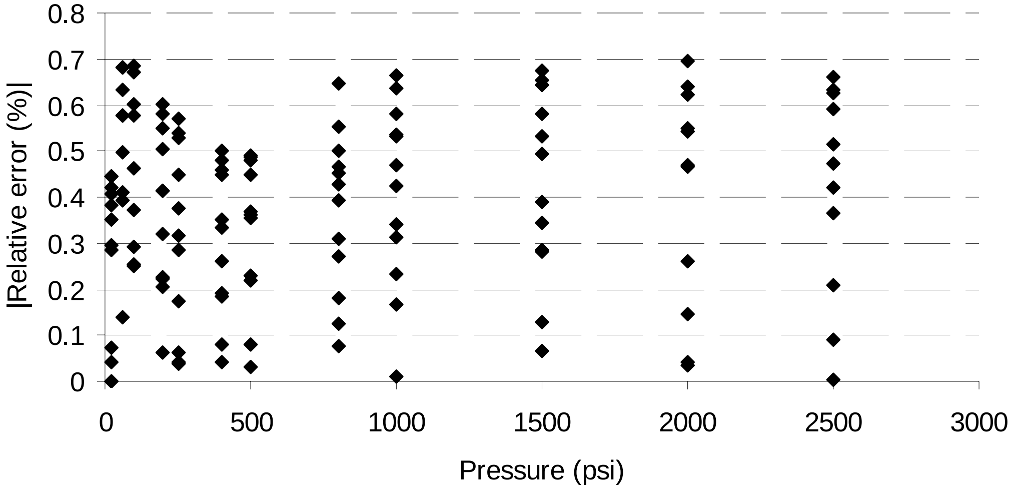

Figure 3.2 The absolute relative error of the enthalpy function at the different pressure values

This correlation performs better at high pressure values, above 100 psi. As will be shown in chapter VI, a correlation of the enthalpy as a function of the steam temperature and saturation temperature will prove useful in observing the effect of the inlet temperature on the specific power. This correlation is:

$$h = A_T\, T + B_T \tag{3.3}$$

$$A_T = 6\times10^{-7} T_{Sat}^2 - 0.0002\, T_{sat} + 0.537$$
$$B_T = -0.0014\, T_{sat}^2 + 0.6843\, T_{sat} + 941.73$$

This correlation is valid for a pressure range between atmospheric and 2000 psi and a superheat of up to 1000 ^{o}F with a relative error less than ±2.0% (Figure 3.3).

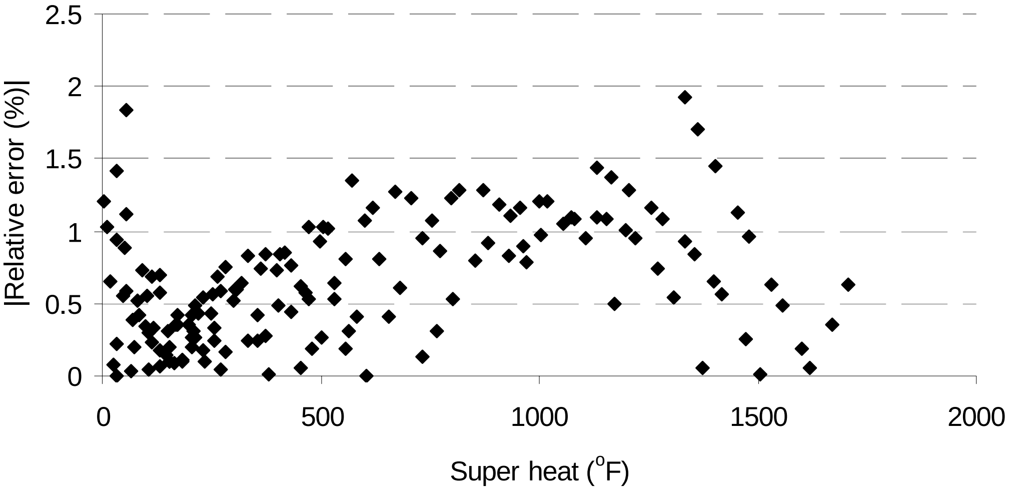

Figure 3.3 The absolute relative error of the enthalpy as a function of steam temperature and saturation temperature, for different superheat temperature values

This correlation tends to have higher error at very low superheat temperature and too high temperature values. In the most practical region it maintains a relative error of ±1.5%.

The saturated liquid enthalpy is expressed as a function of the saturation temperature (T_{sat}).

$$h_f = 0.0005\,T_{sat}^2 + 0.7159\,T_{sat} + 8.8982 \tag{3.4}$$

whereas h_f in Btu/lb and T_{sat} in °F. This correlation is developed for a range of pressure between the atmospheric value and 2000 psi. The maximum relative error is ± 1.6% which is located at the atmospheric pressure and it is less than ±1.0 % elsewhere.

The latent heat is calculated from:

$$h_{fg} = -0.0022\ T_{sat}^{2} + 0.7319\,T_{sat} + 901.69 \qquad (3.5)$$

whereas h_f in Btu/lb and T_{sat} in ^{o}F. The maximum relative error is ± 2.7 % located at 2000 psi and it is less than ±2.0% elsewhere.

The saturation vapor enthalpy is simply the sum of the saturation liquid and the latent heat.

$$h_g = h_f + h_{fg} = -0.0017\,T_{sat}^{2} + 1.4475\,T_{sat} + 910.5882 \qquad (3.6)$$

Figure 3.4 shows the relative error of these correlations when compared to the steam tables.

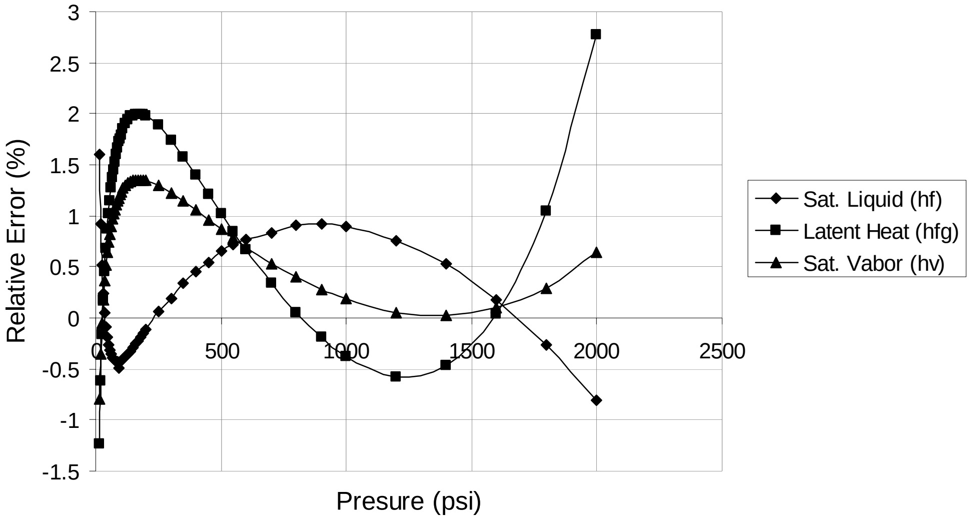

Figure 3.4 The relative error of the correlations developed for the saturation enthalpies

Entropy

One more important correlation is the entropy as a function of enthalpy. This correlation is very crucial. It is used to correct the entropy of the next steam main after the non-isentropic expansion of steam. This correlation is given in equation (3.7).

$$s = \left(\frac{h - 817.35}{0.2029 T_{sat}} \right)^{0.2742} \qquad (3.7)$$

whereas s in Btu/(lb.R) and T_{sat} in ^{o}F. This correlation is derived from equation 3.2 and it is suitable for the same range.

When the boiler condition is defined by the amount of superheat at a certain pressure, the following correlation is used:

$$s = (-0.5549 \ln(T_{sat}) + 3.7876) T^{\,(0.1001 \exp(0.0017 T_{sat})} \qquad (3.8)$$

whereas s is the entropy of the superheat in Btu/(lb.R), T is the inlet temperature in ^{o}F and T_{sat} is the saturation temperature, corresponding with the inlet pressure, in ^{o}F . This correlation is developed for a range of pressure values up to 2500 psi with superheat temperature up to 1500 ^{o}F. The relative error of this correlation is about ±5% but mostly ±3.5% (Figure 3.5 and Figure 3.6).

The saturation region of the entropy is important for determining the quality of steam at the exit of the turbine before it mixes with the streams coming into the steam main. The saturated liquid enthalpy is

$$s_f = 0.0013 T_{sat} + 0.052 \qquad (3.9)$$

while the saturation latent region entropy is

$$s_{fg} = -0.0023 T_{sat} + 1.9071 \qquad (3.10)$$

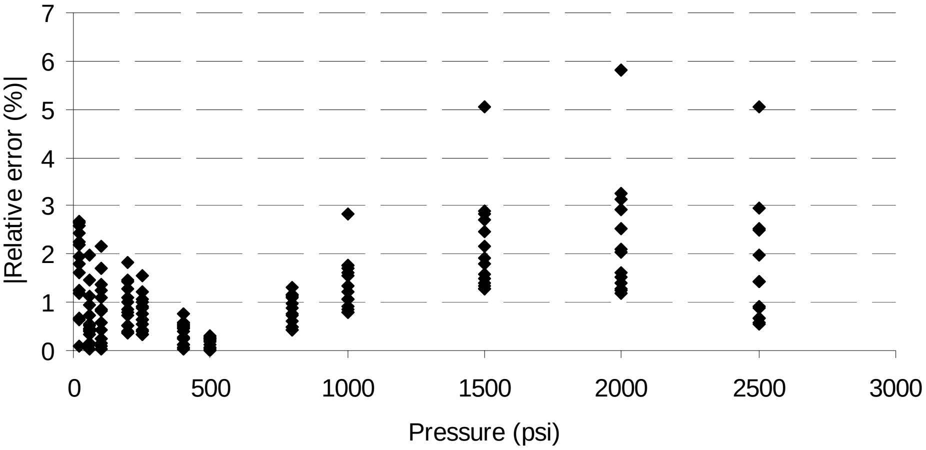

Figure 3.5 The absolute relative error of the entropy as a function of steam temperature and saturation temperature at different pressure values

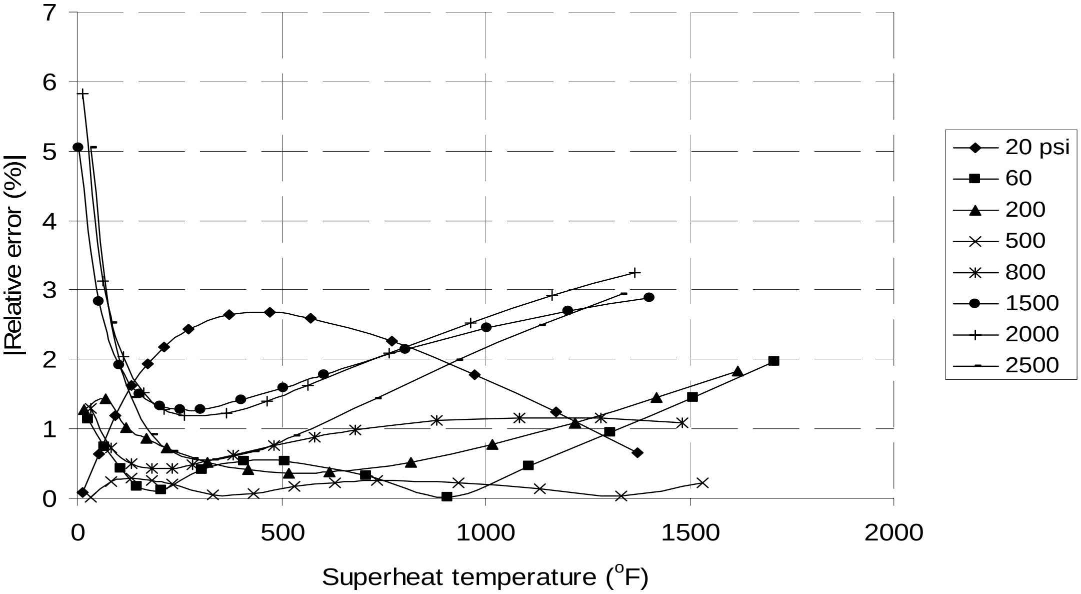

Figure 3.6 The absolute relative error of the entropy as a function of steam temperature and saturation temperature at different superheat temperatures

The saturated vapor enthalpy is then obtained from the sum of the saturated liquid and the latent region entropies and is given by

$$s_g = s_f + s_{fg} = -0.001T_{sat} + 1.9591 \qquad (3.11)$$

These correlations are valid for pressure values up to 2000 psi with a relative error of ±5.0% and less than 3.0±% in most of that range as shown in Figure 3.7.

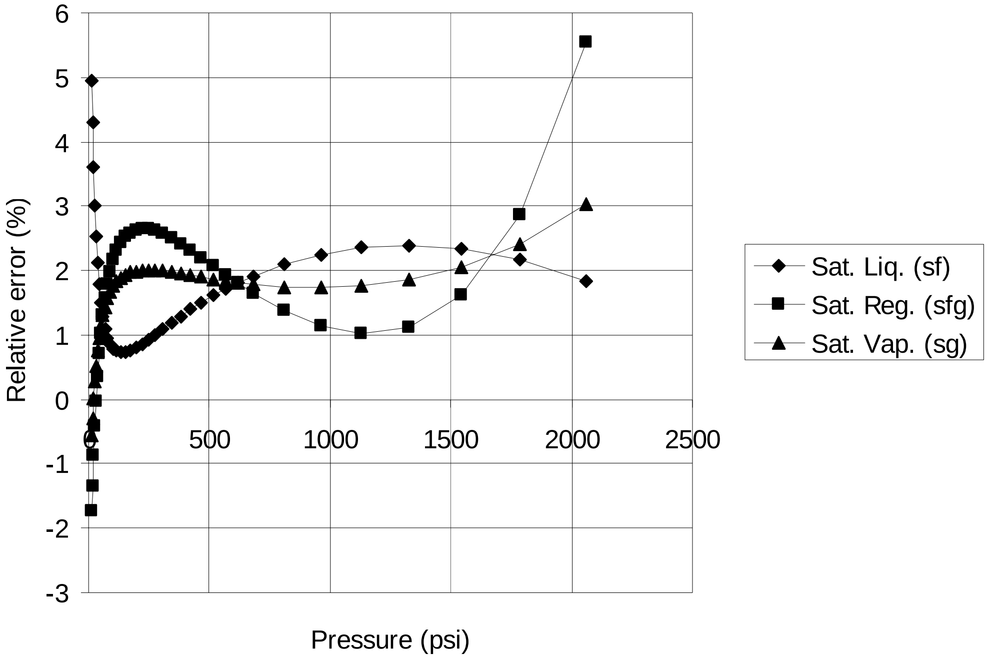

Figure 3.7 The relative error of the entropy correlations at different pressure values

Temperature

In the case of a single turbine optimization, the temperature at the exit of the turbine needs to be evaluated as a function of enthalpy and entropy, the following correlation is used for that purpose:

$$T = (h - B_s) / A_s \tag{3.12}$$

$$A_S = -0.7918S^3 + 3.4575S^2 - 4.5513S + 2.1267$$
$$B_S = 710.22S^3 - 3910.6S^2 + 7117.3S - 3253.5$$

This correlation is valid for a superheat of up to 500 ^{o}F with a relative error less than ±5%. The complexity of this correlation will not entail any computational difficulty since it is only used to check the amount of superheat at the exit of a turbine before it is exported to the chemical process. Moreover, in almost all cases the amount of superheat required is limited, the range of validity (500 ^{o}F) is even exaggerated. The error plot for different pressure values is given in Figure 3.8.

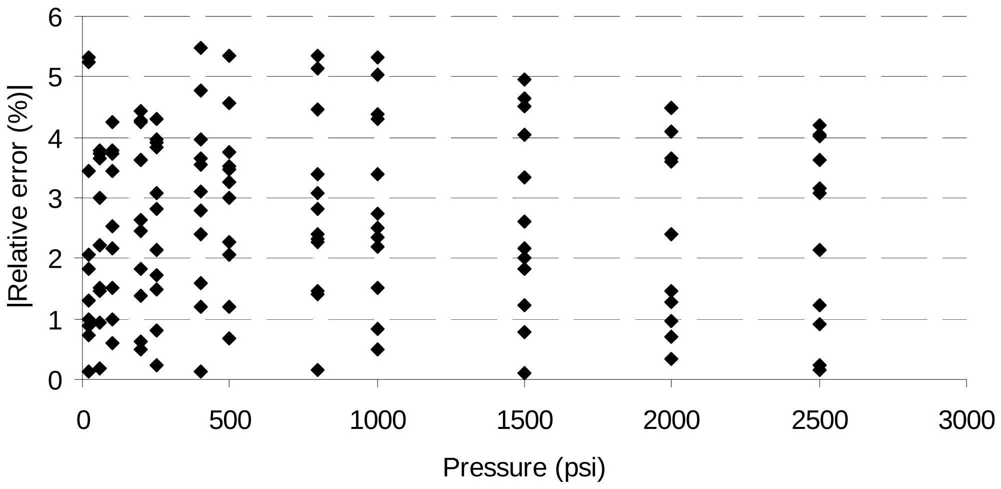

Figure 3.8 The absolute relative error for the temperature as a function of entropy and enthalpy at different pressure values

CHAPTER IV

PROCESS INTEGRATION WITH POWER

Introduction

Due to the depletion and unpredictable price rise of the conventional energy resources, tremendous efforts on different frontiers have been devoted to regulate wise usage and saving of these resources. In this respect, process optimization and integration have proven indispensable. In this paper, the chemical process is integrated with the power cycle to save on the heating requirement and meanwhile simultaneously produce some power. Such an optimization is implemented on Rankine cycle coupled with a process that needs to be provided with heat at a certain pressure.

Steam Usage in Chemical Plants

Steam is one of if the mostly used commodities in the chemical plant. It is used for heating and non-heating purposes. It is used as a heating mean due to many advantages including its high latent heat content at a wide range op operating temperatures (Smith 2005). The heating usage of steam can be directed, e.g. through live steam injection, or indirect e.g. through heat exchanger. Other uses of steam include reduction of partial pressure, lowering NO_x emissions, assisting combustion and power generation.

Saving of steam takes different ways depending on how steam is used in the chemical process. After a non-heating usage of steam, steam can be recovered using mass integration. When steam is used for heating, the saving is achieved through heat integration and the grand composite curve (GCC). In heat integration, instead of providing external cooling or refrigeration to the hot streams to be cooled and external cooling to the hot streams to be cold, the cold and hot streams are internally matched so as to allow the minimum possible amount of external heating and cooling (El-Halwagi 2003; El-Halwagi 2006).

The Utility System Optimization

A huge portion of a plant's operation cost comes from the energy consumed by the plant in both of its forms, heat and power. Moreover, a huge portion of an industrial country's usage of energy is allocated to industrial energy-consuming industries such as refineries. A slight improvement in the performance of a single component e.g. a boiler can reap thousands of dollars of saving every year.

Optimizing the utility system is economically and environmentally beneficial. Papandreou and Shang (2008) used a multi-objective mixed integer linear programming (MILP) approach to optimize the chemical plant by setting up economic and environmental criteria in their objectives. They included binary variables to select the different design and operating conditions. Rodríguez-Toral et al. (2001) used a sequential quadratic programming to solve the same problem. They used an equation-oriented optimization that involves the steam thermodynamic properties. Grossman (1985) and Papoulias and Grossman (1983) introduced a MINLP approach for the superstructure utility system optimization. Hul and Natori (1996) introduced a mixed-integer approach for multi-period utility system. Maia and Qassim (1997) used simulated annealing which is a metaheuristic method to optimize a multi-period utility system. In addition to environmental and economical concerns, cogeneration is another criterion that is included in the optimization of the utility system (Dhole and Linnhoff 1993; EDUCOGEN and INESTENE 2001).

Wilkendorf et al. (1998) introduced synthesis of the utility system by satisfying the process requirement at the minimal possible cost. In Rankine cycle, attention is mainly paid to the expander (steam turbine) while the other three blocks in the cycle are viewed as supporting equipments. A more rewarding approach is to versify the uses of this cycle by taking advantage of the heat discharged from the condenser. This will significantly contribute to energy conservation and environmental thermal pollution. However, in such an approach, the lower pressure level has to be dictated to the cycle by an external party, a chemical process in this work. Consequently, energy production from the turbine side can be increased only by increasing the boiler pressure which is highly dependent on safety and economical factors. In this problem, for a given heat required by a process at a certain pressure level, the optimum high pressure level is sought. The change of this value with changes in other parameters of the optimization problem is further studied.

The Heat Engine

The principle of the heat engine is to produce power by passing high pressure steam through an expander, Figure 4.1.

The basic Rankine cycle is composed of the following thermodynamic processes that take place in a preodic manner:

- 1-2 Constant pressure heat addition in a boiler
- 2-3 Isentropic expansion in a turbine
- 3-4 Constant pressure heat rejection in a condenser
- 4-1 Isentropic Compression in a pump

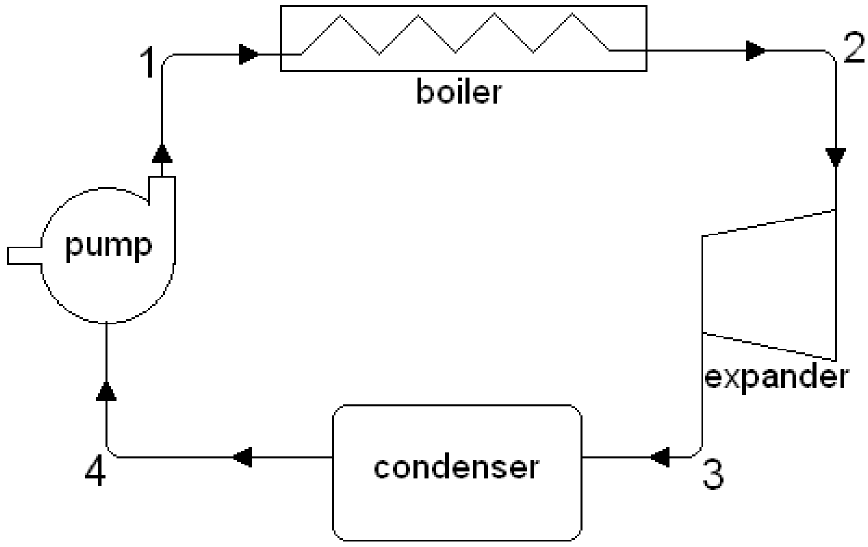

Figure 4.1 Rankine power cycle

These four processes are better described with respect to Mollier (T-S) Diagram T-S (Figure 4.2). Fresh treated water is first fed to the boiler which is fueled by combustions gases or nuclear reactor and operates at a constant pressure. The boiler section with the superheated section is termed as *steam generator* (Process 1-2). The high pressure steam is then fed to the turbine (expander), the high pressure energy stored as high enthalpy is used to rotate the turbine blades and hence produce power (Process 2-3). This process is ideally isentropic, but due to the mechanical deficiencies, the process line deviates towards the left on the T-S diagram causing less enthalpy difference than the idea process.

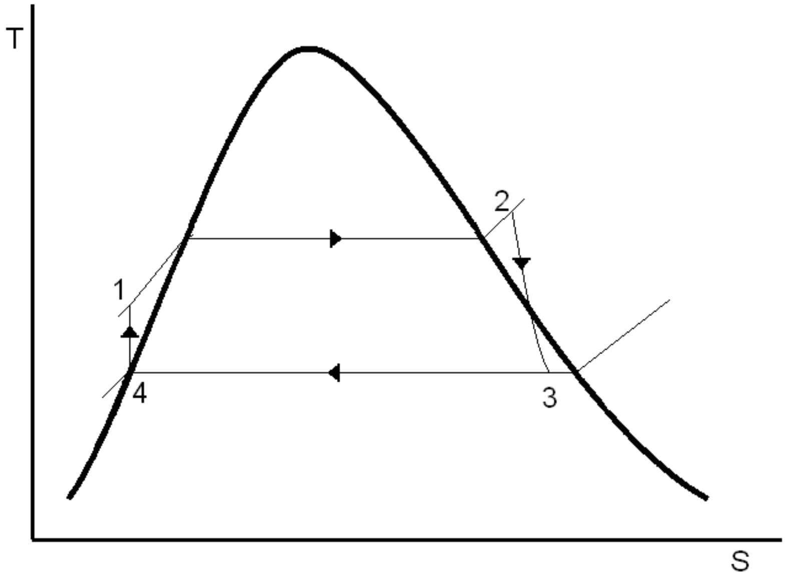

Figure 4.2 Mollier (T-s) diagram

At the exit of the turbine the working fluid can either superheated steam (back-pressure turbine) or saturated steam (condensing turbine) with quality of 90% or more to avoid erosion of the turbine hardware.

The steam at the exit is then fed to a condenser (Process 3-4). In order to pump the steam back to the boiler, the steam is condensed and heat is discharged to the ambient in a condenser. At the exit of the condenser, the liquid is fed to a water pump (4-1) to elevate the pressure to the boiler's level in order to start the process over.

Integrating the Heat Engine with the Process

By investigating the Rankine cycle, in addition to the mechanical losses, a major loss in the overall cycle is the heat discharged to the ambient from the condenser. Work is input to the cycle in the pump and the boiler. Power is produced from the turbine and the rest is losses in the forms of heat in the condenser and mechanical losses in the turbine. In order to improve the overall efficiency of the cycle, the heat discharge to the ambient can instead be utilized by the chemical process as shown in Figure 4.3.

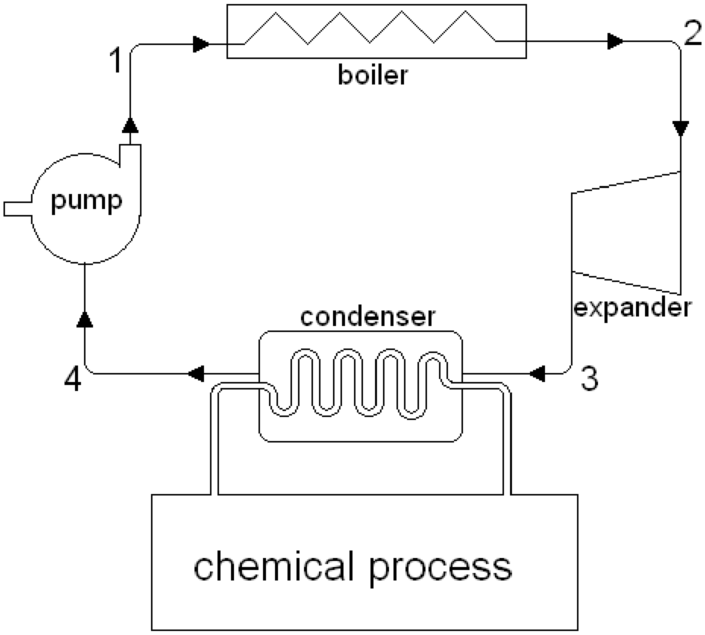

Figure 4.3 Integrating the chemical process with the heat engine

Looking at the upper picture from an economic viewpoint, the heat discharged at the condenser is no longer a loss. This approach is not only viable to utilize that heat, a reverse approach can also be implemented, i.e. determining the cycle's parameters as dictated by the process requirement. This approach is practical when producing steam at high pressure and cogenerating power is more economical than producing steam at low pressure.

In this work, it is assumed that the boiler operates at relatively high pressure levels. The pressure of the boiler is set in conjugation with the process need in such a way that will optimize the tradeoff between the cost and the output of the process.

Instead of sticking to a specific equipment model, the approach is made open to be utilized by any real case problem. The cost and performance estimations can be incorporated either as a continuous function in terms of other parameters, e.g. cost of turbine as a function of its load, or as discrete integers e.g. a set of possibilities of different turbines at different costs. In the objective function of the formulation, the performance and cost are expressed using the thermodynamic properties (Figure 4.4) in order to define the optimal operation.

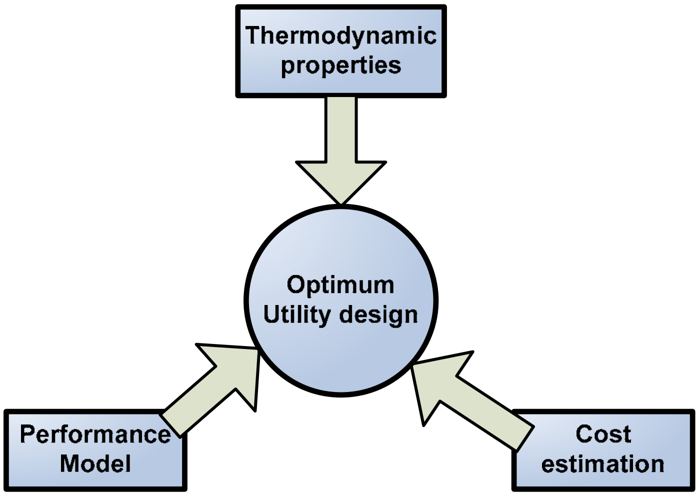

Figure 4.4 The optimum utility design’s considerations

Applications

One of the most utility-consuming units is the distillation column. In order to decrease the utilities required for the process, different ways are suggested as shown on Figure 4.5.

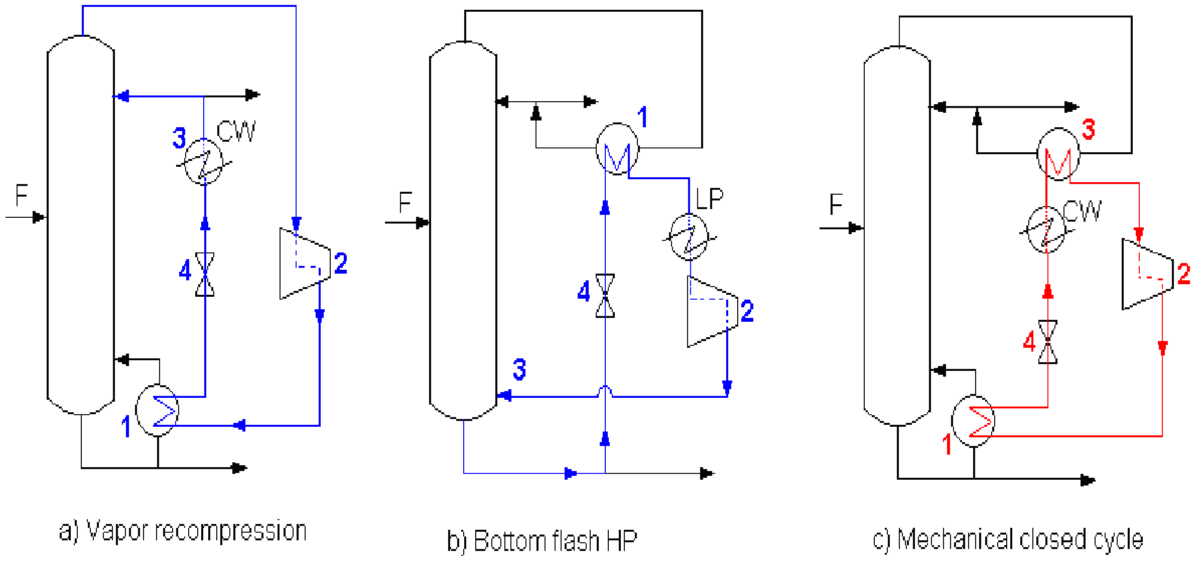

Figure 4.5 Proposed ways to save on the distillation utility requirements

The first method (Figure 4.5.a) is vapor compression. It is the most popular of the three. When applicable, the saving can be as low as one tenth of the conventional column (Sloley 2001). This method is feasible when the overhead vapor possesses the desirable thermodynamic features of a heat pump working fluid. This method, however, has the disadvantage of the overhead product getting contaminated by the lubricating oil of the compressor (Supranto et al. 1986).

The second method (Figure 4.5.b) is called the bottom flash heat pump. This method is feasible when the bottom product has the thermodynamic characteristics needed for the working fluid of a heat pump. It is more applicable when the column is operated at high pressures in order to make expansion feasible (Supranto et al. 1986).

The third method is the mechanical closed cycle (Figure 4.5.c). This method requires a feasible pressure difference between the reboiler and the condenser in order to make installing a turbine a feasible solution. Despite the early advent of this type of integration, there is no systematic or reproducible model that could be located in literature.

The fact that distillation columns need enormous heat load makes integrating the steam turbine with the distillation column an attractive solution. The interaction in this case will not be as limited and circumstantial as the methods described above. Due to the high amount f heat required by the column, makes delivering it from the power cycle condenser less costly and more accessible.

The Optimization Problem

The developed correlations in addition to the cost and efficiency estimations can be used to solve a variety of optimization problems in steam turbine cycle.

The first case is on a process that requires heat to be delivered at pressure P_2 with the minimum temperature be slightly above the saturation temperature and at P_2. Available to the process, a boiler that can operate at a maximum pressure P_{1max} and also a steam turbine to be purchased assuming that it operates at or below a maximum load

M

M_{max} and has a maximum inlet temperature T_{1max}. The formulation of the problem is given in Figure 4.6.

$$
\begin{aligned}
&\textit{objecive} \quad \textit{function}: \\
&\text{min cost} = \text{Turbine cost} + \text{Boiler cost} + \text{ Fuel cost} + \text{operation cost - power price} \\
&\textit{subject to}: \\
&\text{Turbine cost} = f(power) \\
&\text{Boiler cost} = f(Q_{boiler}, P_1, T_1) \\
&\text{Fuel cost} = f(Q_{boiler}, \eta_{boiler}) \\
&\text{power price} = f(shaft\ work, \eta_{generator}) \\
&shaft\ work = f(h_1, h_{2a}, m) \\
&P_2 < P_1 \le P_{1\max} \\
&T_{sat,1} < T_1 \le T_{1\max} \\
&T_{sat2} = f(P_1) \\
&T_{sat1} = f(P_2) \\
&s_1 = f(T_{sat1}, T_1) \\
&h_1 = f(T_{sat1}, s_1) \\
&\eta_{turbine} = f(P_1, load) \\
&h_{2,is} = f(s_1, T_{sat2}) \\
&h_{2,a} = f(\eta_{turbine}, h_1, h_{2,is}) \\
&h_{2a} > h_{g2} \\
&s_2 = f(h_{2a}, T_{sat2}) \\
&T_2 = f(s_2, h_{2a}) \\
&m = \frac{Q_{process}}{h_{2a} - h_{f2}} \le m_{\max} \\
&Q_{boiler} = f(h_1, f_{f2}, m) \\
&\eta_{boiler} = f(fuel, P_1, load)
\end{aligned}
$$

Figure 4.6 The optimization problem of integrating power with the process

The Equipment Performance

A very case-dependent criterion in the optimization problem is the equipment performance. The wide variety of the different generic estimations may affect the final answer. A much better alternative is to use the actual prices of all possible choices. For the demonstration purpose, however, textbook cost estimation will be used along with a set of developed correlations of the thermodynamic properties that could be used with any optimization setup.

Turbine Efficiency

The steam turbine has a mechanical efficiency as well as isentropic efficiency (Figure 4.7). The overall efficiency is the product of both. The mechanical efficiency is highly dependent on the turbine structure and manufacturing and it can be higher than 95% for the today's steam turbine. The isentropic efficiency, however, is more dependent on the operation parameters. It exhibits a high sensitivity to the inlet pressure and steam load for example.

The turbine efficiency can be expressed in terms of the inlet pressure as well as the load (Evans Jr. 1971; Fronseca Jr. 2003). The efficiency is given as a function of the mass or power load and produced for a family of inlet pressure values.

These graphs correlate well with the following empirical equation:

$$\eta = \frac{a\,P}{bP + P_{\max}} \tag{4.1}$$

where P is the power, P$_{\text{max}}$ is the maximum power at that inlet pressure and, *a* and *b* are functions of the inlet pressure.

Mavromatis and Kokossis (1998) established another correlation by modeling the steam turbine using Willan's line and came up with the following correlation:

$$\eta_{is} = \frac{6}{5B}\left(1 - \frac{3.41443\times10^{6}\ A}{\Delta h_{is}\ M^{\max}}\right)\left(1 - \frac{M^{\max}}{6M}\right) \tag{4.2}$$

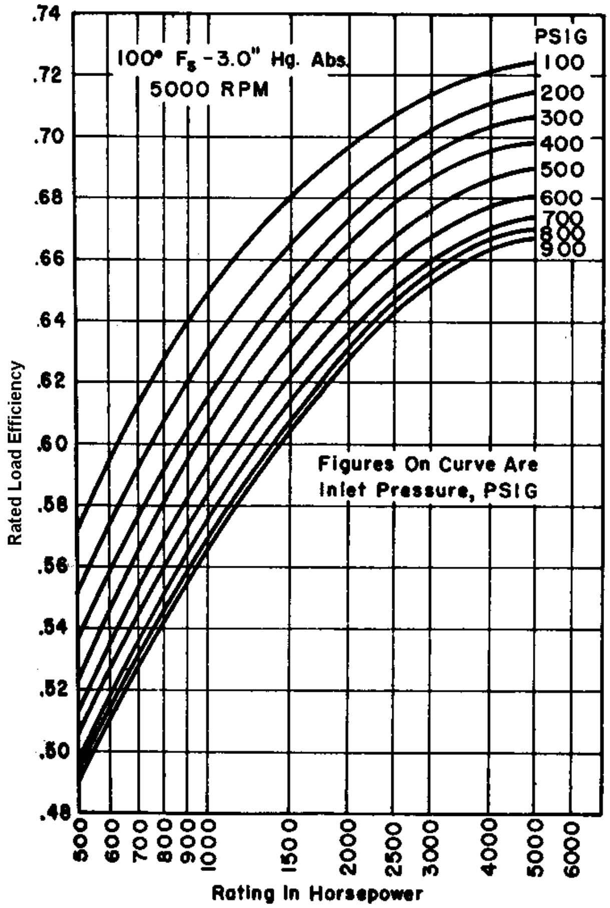

Figure 4.7 A typical performance chart (Evans Jr. 1971)

whereas M and M^{max} (in lb/hr) are the mass flow and the maximum mass flow respectively, Δh_{is} is the isentropic enthalpy difference (in Btu/lb) and, *A* and *B* are functions of the saturation temperature of the inlet pressure given in appendix A. It should be noted that the coefficient of this equation is changed to the American customary system here while it is used in different units in the original reference.

The A and B factors in equation (4.2) correlation change with the power output as explained in appendix A. This expression very closely matches the one given by (4.1). In this works, Mavromatis and Kokossis' correlation is used without losing generality.

Most models suggest a generic form regardless of the manufacturer but there is a model that involves a parameter that requires to be estimated using experimental runs of a specific turbine (Varbanov et al. 2004). This model is given by the following equation:

$$\eta_{is} = \frac{n.m - W_{INT}}{m.\eta_{mech}.\Delta H_{is}} \tag{4.3}$$

whereas m, η_{mech} and DH_{is} are the mass flow rate, mechanical efficiency and isentropic enthalpy difference respectively. W_{int} and n are two parameters, given in appendix A, that are calculated in terms of the mass flowrate, isentropic enthalpy difference, saturation temperature difference, type of turbine (backpressure versus condensing) and a parameter L that needs to be defined by correlating the reading of the specific turbine.

Boiler Efficiency

The boiler efficiency is basically the heat delivered to the steam to the heat contained in the fuel.

The typical boiler efficiency is somewhere between 75 to 80% and can reach to over 90%. The boiler efficiency depends on the material from which it is constructed and the fuel type in addition to the operating and ambient conditions. Due to the fuel cost, that is, with the maintenance cost, is much higher than the capital investment, a slight improvement in the boiler efficiency may result in substantial savings. The evaluation of the boiler efficiency is more complicated than that of the steam turbine and the cost is more sensitive to the boiler efficiency. A common way to predict the boiler efficiency is the Power Test Code for Steam Generating Units developed by the American Society of Mechanical Engineers (ASME,1998).

The Cost Function

The generation cost is the sum of all costs of operations and materials during production. For the optimization problem, the generation cost of steam will be calculated using the following equation (Kumana and Associates 2003):

$$C_g = C_f + C_w + C_{BFW} + C_p + C_a + C_b + C_d + C_e + C_m \qquad (4.4)$$

where

C_f : Fuel cost ($/klb)

C_w: Raw water supply cost($/klb)

C_{BFW}: Boiler feed water treatment cost including clarification, softening, and demineralization ($/klb)

C_p: Feed water pumping power cost ($/klb)

C_a: Combustion air fan power cost ($/klb)

C_b: Sewer charges for boiler blowdown (\$/klb)

C_d: Ash disposal cost (\$/klb)

C_e: Environmental emission control cost (\$/klb)

C_m: Maintenance materials and labor (\$/klb)

In this work however, to maintain a traceable formulation, a consistent set of units will be used. Instead of \$/klb for the operation cost, \$/MMBtu of heat delivered by the boiler will be used.

The fuel cost contributes substantially to the generation cost. It contributes with over 90% of the generation cost. The fuel cost is calculated as

$$C_f\ (\$/hr)= a_f\ Q_b\ 10^{-6}/\eta_b \qquad (4.5)$$

where

a_f : fuel cost (\$/MMBtu)

Q_b: the amount of heat transferred to the steam at the boiler (Btu/hr)

$Q_b= m\ (h_1 - h_{1f})$

h_1: enthalpy of steam at turbine inlet (Btu/lb)

h_f: enthalpy of boiler feedwater assumed at the saturated liquid point (Btu/lb)

m: steam flowrate (lb/hr)

η_b: overall boiler enthalpy

The maintenance cost of the boiler is about 30% of the fuel cost (Kumana and Associates 2003). In order to account for the increase in pressure, a flexible factor is introduced.

$$C_g = 1.3\ F_p\ C_f \qquad (4.6)$$

where F_p: a factor accounting for the increase in pressure. This factor can be unity if the increase in pressure is insensitive to the operation cost but sometimes there are additional costs incorporated with higher pressure values such as the minimum allowed concentrations of the water composition in the boiler which gets less and more costly to remove as the pressure values get higher (Branan 2002).

The electric power price ($/kWh) is

$$P_e\ (\$/hr)= a_e P_t\ \eta_g\ /3413 \qquad (4.7)$$

where

a_e: electrical power price ($/KWh)

P_t: the turbine shaft power output (Btu/hr); P_t=m $(h_1 – h_{2a})$

h_1 : enthalpy of steam at turbine inlet (Btu/lb)

h_{2a}: actual enthalpy at steam outlet (Btu/lb)

η_g: generator efficiency

The capital costs of the equipment are the most unstable element of the formulation. This cost can be introduced as a function of other operation variables or as integers of a set of possible selections with their operation capacities. For the illustration purposes in this work, cost equations are improvised by comparing different cost estimations (Seider et al. 2004), moreover, the most significant cost is assumed contributed by the turbine and the boiler. The variation in the pump cost between the different operation variables is insignificant.

The boiler is assumed a water-tube boiler fueled with oil or gas with the following cost estimation

$$C_{boiler}\ (\$) = 3\ N_p\ N_T\ Q_b^{0.77} \qquad (4.8)$$

where

Q_b : The amount of heat transferred to the steam (Btu/hr)

N_p : a factor accounting for the operation pressure and is given by:

$N_p = 7\times10^{-4}\ P_{1g} + 0.6$; P_{1g} is the gauge pressure (psig) of the boiler.

N_T: a factor accounting for the superheat temperature and is given by:

$N_T = 1.5\times10^{-6}\ T_{sh}^2 + 1.13\times10^{-3}\ T_{sh} + 1$; T_{sh} is the superheat temperature (°F).

$T_{sh} = T_1 - T_{sat1}$

The turbine is assumed a non-condensing turbine with its cost equation derived the same way

$$C_{Turbine}(\$) = 475\ P_t^{0.45} \qquad (4.9)$$

where

P_t: the turbine shaft power output (Btu/hr); $P_t = m\ (h_1 - h_{2a})$

To find the optimum design of a grass root utility, the objective function will be expressed in terms of the annual cost. In this case, the annualized capital cost of the equipment will be used.

$$\text{The annual cost} = 1.3\ F_p\ C_f\ t + AC_{boiler} + AC_{turbine} - P_e\ t \qquad (4.10)$$

where

t: the annual operation time (hours/year)

AC_{boiler}=the annualized capital cost of the boiler (\$/year)

AC_{boiler}= (C_{boiler} – salvage Price)/years of service

$AC_{turbine}$=the annualized capital cost of the turbine (\$/year)

$AC_{turbine}$= ($C_{turbine}$ – salvage Price)/years of service

To determine the optimum operation parameters for some existing equipments and to confine the tradeoff between the electricity produced and the fuel consumed, the objective function will reduce to

$$\text{The annual cost}= 1.3\ F_p\ C_f\ t\ -P_e\ t \tag{4.11}$$

Correlations of the Thermodynamic Properties

To find the optimal values of the Rankine cycle parameters, the whole problem needs to be formulated and stated as an optimization problem with an objective function and constrains. Consequently, the steam tables usually used for thermodynamic cycle calculations will be replaced by mathematical equations.

The thermodynamic properties will be calculated from the correlations developed in chapter III.

In terms of the pressure, the saturation temperature will be calculated using the following correlation:

$$T_{sat} = 112.72\, P_1^{0.2289} \tag{4.12}$$

whereas P_1 is the pressure in psi and T_{sat1} is the saturation temperature in ^{o}F. The relative error is ±1.87% and average relative error is 0.64%.

For a given pressure and temperature inlet, the entropy at the inlet of the turbine can be calculated from equation (3.8):

$$s = (-0.5549\ln(T_{sat}) + 3.7876)T^{\ (0.1001\exp(0.0017T_{sat})} \tag{3.8}$$

whereas s is the entropy of the superheat in Btu/(lb.R), T is the inlet temperature in °F and T_{sat} is the saturation temperature, corresponding with the inlet pressure, in °F . As mentioned in chapter III, this correlation is developed for a range of pressure values up to 2500 psi with superheat temperature up to 1500 °F. The relative error of this correlation is ±3.5%.

The enthalpy is then obtained as a function of the saturation temperature of the steam main and the entropy as given in equation (3.2).

$$h = 0.2029\ T_{sat}\ s^{3.647} + 817.35 \tag{3.2}$$

whereas h in Btu/lb, s in Btu/(lb.R) and T_{sat} in °F. The relative error is ± 0.6% and is valid in the range between atmospheric pressure and 2000 psi.

The turbine efficiency will be expressed using one of the above correlations or a better expression according to the data provided by the manufacturer. The isentropic efficiency at the outlet of the turbine will be expressed using the same equation of enthalpy but in terms of the exit saturation temperature and the inlet entropy. In terms of the turbine efficiency and the inlet enthalpy and the outlet isentropic enthalpy, the actual enthalpy will be

$$h_{2,a} = h_1 - \eta_{turbine}(h_1 - h_{2,is}) \tag{4.13}$$

The outlet temperature will be expressed in terms of the entropy and the enthalpy. Given the saturation temperature and actual enthalpy, equation (3.7) is used to find the entropy.

$$s = \left(\frac{h - 817.35}{0.2029\, T_{sat}} \right)^{0.2742} \tag{3.7}$$

In terms of the entropy and the enthalpy, the outlet temperature is found using equation (3.12):

$$T = (h - B_s) / A_s \tag{3.12}$$

$$A_S = -0.7918\,S^3 + 3.4575\,S^2 - 4.5513\,S + 2.1267$$
$$B_S = 710.22\,S^3 - 3910.6\,S^2 + 7117.3\,S - 3253.5$$

This correlation is valid for a superheat of up to 500 °F with a relative error less than ±5%.

The boiler and condenser loads are assumed to be the latent heat added to the superheat. The saturated fluid enthalpy is calculated from equation (3.4).:

$$h_f = 0.0005\,T_{sat}^2 + 0.7159\,T_{sat} + 8.8982 \tag{3.4}$$

whereas h_f in Btu/lb and T_{sat} in °F. This correlation is developed for a range of pressure up to 2000 psi. The maximum relative error is less than ± 1.6%.

The mass flowrate will be expressed in terms of the heat required by the process, the actual enthalpy at the exit of the turbine and saturated fluid enthalpy of the exit pressure.

$$m = \frac{Q_{process}}{h_{2a} - h_{f2}} \tag{4.14}$$

The heat output of the boiler will be expressed in terms of the mass flowrate, the enthalpy at the inlet of the turbine and the saturated fluid enthalpy of the exit pressure.

$$Q_{boiler} = m(h_1 - h_{f1}) \tag{4.15}$$

The above formulation gives a universally solvable optimization problem for the combined heat and power in a steam turbine. An optimization software such as Lingo 10.0 that is used in the following case studies can obtain the global solution of the problem. An incremental loop of pressure with a nested loop of superheat temperature can also be used in any programming language. The increment can be set reasonably such that the runtime is reasonable. A proof of convergence of this loop is given in appendix B.

Case Study I

A distillation column (Figure 4.8) requires 20 MMBtu/hr heat at 100 psi. The turbine used in the facility can operate at a maximum inlet temperature of 450 ^{o}F and maximum flowrate of 25,000 lb/hr. The boiler efficiency is 40% and it operates at a maximum pressure of 280 psi. The minimum superheat temperature at the exit of the turbine is 5 ^{o}F. The power price is 0.05 \$/kWh and the fuel price is 2.80 \$/MMlb, the generator efficiency is 40%, the pressure factor in the generation cost is unity.

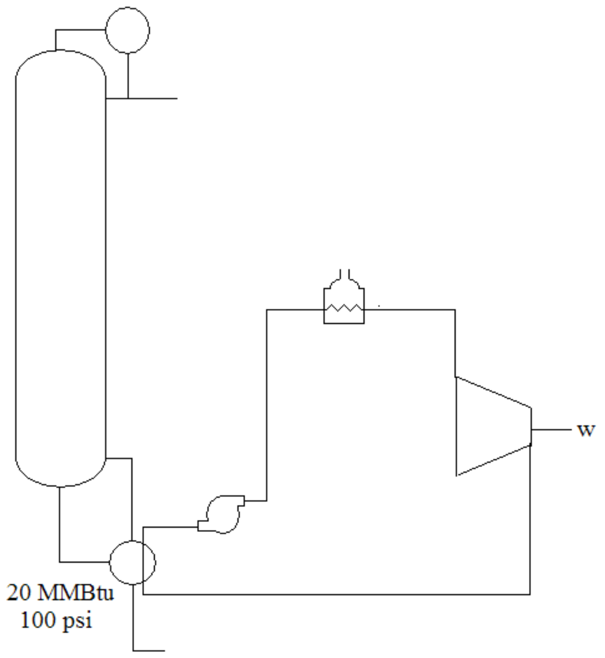

Figure 4.8 Integrating the distillation column with power

In order to determine the optimum working conditions, the isentropic efficiency of the turbine is estimated using Mavromatis and Kokossis (1998) model. Solving the problem can be either performed using a nonlinear programming technique to solve the optimization problem using a global solver like Lingo^{TM} 10.0 or it can also be solved by using an iterative algorithm that tests the optimization variables at reasonable increments.

The optimum solution of this problem is located at an inlet pressure of 202 psi and inlet superheat temperature of 54 ^{o}F with the superheat at the turbine exit 5.8 ^{o}F. The flow rate is 22,802 lb/hr. The turbine isentropic efficiency is 57 % and the maximum efficiency is estimated around 59%.

Although the pressure factor is an assumed one, there is a clear tradeoff between power production and fuel consumption.

The developed optimization problem can be used for a wide range of applications. The applications are not only limited to operation parameters but also to equipment choices by establishing an MINLP that includes selecting the proper equipment along with the optimal operation parameters as shown in the next case study.

Case Study II

A chemical process that requires an amount of heat that is 20 MMBru/hr at 100 psi pressure; available selections of turbines and boilers are given as follows:

All turbines tolerate the maximum pressure of the boilers. The minimum superheat at the exit of the temperature should at least be 5 ^{o}F. The generator efficiency is 50%. The pressure factor is taken as unity. Other specifications are given by Table 4.1 and Table 4.2.

Table 4.1 Boilers' Specifications in Case Study II

	Fuel cost ($/MMBtu)	Efficiency	Maximum operating pressure (psi)
Boiler 1	5.50	0.8	250
Boiler 2	3.40	0.7	230
Boiler 3	2.80	0.4	280

Table 4.2 Turbines' Specifications in Case Study II

	Maximum inlet temperature (^{o}F)	Maximum steam load (lb/hr)
Turbine 1	500	25,000
Turbine 2	450	25,000
Turbine 3	600	30,000

This price equation is to account for the expense resulting from producing the same amount of heat at higher pressure values. The power price is 0.05 $/kWh.

The optimum of the nine possible combinations was found to be Boiler 2 and Turbine 3. The inlet pressure is chosen to be 230 psi and a superheat of 60 ^{o}F at inlet and 5.2 ^{o}F at exit and flow rate of 22,811 lb/hr. The turbine isentropic efficiency is 59 % and its maximum efficiency is 63 %.

Conclusion

The inter-dependent nature of the Rankine cycle variables makes optimization a viable for design consideration. The most expensive term in the cost function is the fuel, the above examples show that the solution usually lean towards the boiler with the cost ($/MMBtu) to efficiency ratio.

Other factors that can only be defined by the solution of the optimization problem are the optimum superheat temperature and flow rate.

CHAPTER V

TARGETING OF THE UTILITY SYSTEM

Introduction

The chemical process usually requires steam at different pressure and temperature values for heating and non-heating purposes. In order to provide steam at the required condition, the designer has to decide whether to provide steam at the extreme condition and then let it down to the different levels or produce steams separately at different boilers. The second option is mostly viable if the capital cost involved is not very high and also when the steam quantities are not very high which is the case in small plants.

When the steam range is wide with relatively high quantities of steam required, it might be lucrative to produce steam at the extreme condition and then let it down to the lower pressure values through steam turbines. This process however requires an iterative procedure due to the interdependency of the variables to be determined by the designer. This chapter is meant to design a general procedure for targeting the utility system's parameters.

The Utility System

The steam utility system (Figure 5.1) is mainly composed of boiler feedwater treatment, steam boilers and steam turbines. The boiler feedwater treatment is important in that it purifies the feedwater from any suspended solids, dissolved solids, dissolved salts and dissolved gases. These substances are very detrimental to the utility system as they may cause fouling in the boiler and corrosion in the turbines and piping system. The feedwater treatment is a multi-unit system where the removal of the above substances is carried out.

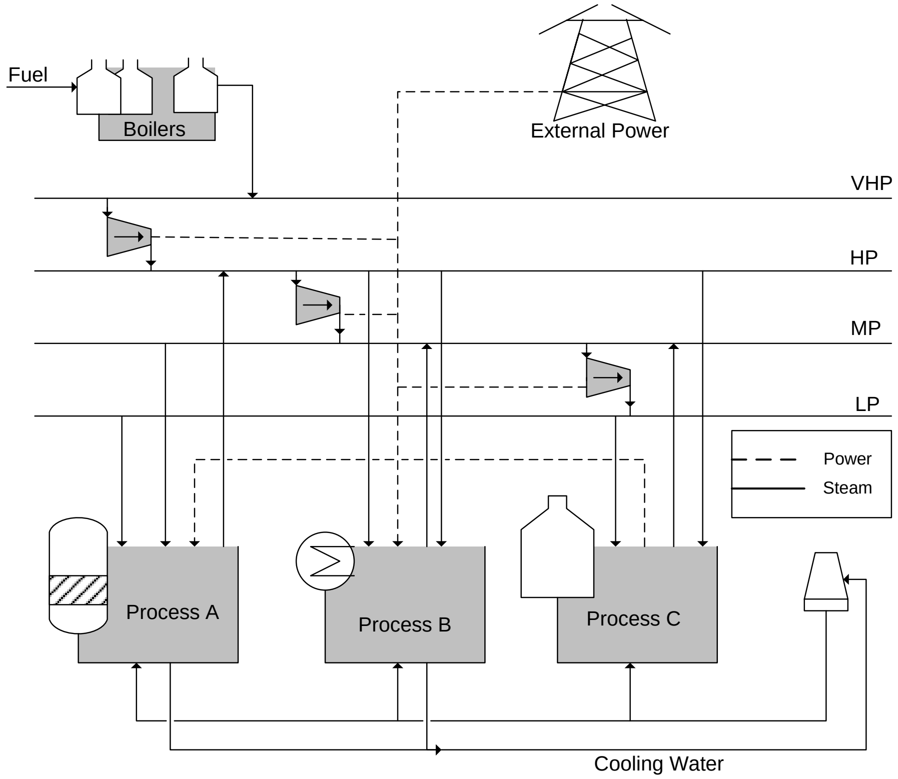

Figure 5.1 A typical site utility system flowsheet

There are many types of steam boilers depending on the steam pressure, steam output and type of fuel (Smith, 2005). The highest pressure is the one used for power generation and it is 100 barg (about 1470 psi). The steam is normally distributed at a pressure between 10 and 40 barg with the lowest approaching 2 barg.

Steam turbines are categorized based on their output pressure. Back pressure turbines are those that outlet reusable steam while condensing turbines are the ones that outlet condensate. Some turbines can also outlet or receive steam in their intermediate stages.

The placement of the steam turbines is a main issue in optimizing the performance of a utility system. Thermodynamic insights and understanding of the performance of the steam turbine is very essential.

Determining the operation conditions is somehow dependent on the steam requirement. Physical insight is also very helpful in discriminating the practical alternatives from the unviable ones. For example, it is always better to letdown pressure and produce energy at the highest possible pressure gab instead of passing steam through an intermediate pressure level when steam is not needed at that level. It is also practical to import steam to the steam main whose pressure is equal or just below the steam pressure while importing heat required at a minimum temperature from a steam main whose temperature is just above that minimum temperature. Other issues need to be tested to locate the suitable operation conditions, this check will be done either through algorithms, solving optimization problems or by exhaustive enumeration when such a solution is practical, in the hope of finding out a general and consistent solution to similar problems.

As long as steam is needed at the exit pressure of the turbine, a back-pressure turbine will then be used. Moreover, the efficiency of steam is proportional with its flowrate load and so the size of the turbine must be selected accordingly.

For a given system of steam mains with heating and non-heating requirements at each level, the calculation of the optimum steam flowrates is not straightforward. The potential for power cogeneration introduces additional complexity to the design procedure. The followings are the design challenges:

- What are the optimum sources of steam (e.g., type of boiler, pressure and temperature of steam)?
- Should turbines be placed among steam mains? Which steam main? What types of turbines (e.g., condensing, extraction)? At what capacities?

When steam turbines are used, their efficiencies are not determined beforehand. Indeed, the efficiency at each level is a function of the mass flowrate and the turbine capacity. On the other hand, the mass flowrate is a function of the heat required as well as the actual enthalpy at each level which is, in turn, a function of the efficiency. This mutual dependency necessitates developing an iterative procedure.

Previous Work

The procedure developed here is comparable to previous works in this field. Raissi (1994) developed the T-H model which was based on an observation by Salisbury and Schenectady (1942). They noticed that for fixed inlet conditions of a steam turbine:

$$h_{@\text{ exit conditions}} - h_{\text{saturated water @ exit pressure}} = \text{Constant} \tag{5.1}$$

This enthalpy line is depicted in Figure 5.2. In this figure, the straight line shown on the right is the outlet of a steam turbine at different saturation temperature with the inlet condition assuming 1.65 Btu/lb.^{o}F entropy and 500 ^{o}F inlet saturation temperature, assuming 70 % efficiency. The two arrows show the constant heat load assumption by Salisbury and Schenectady (1942) and the actual heat load.

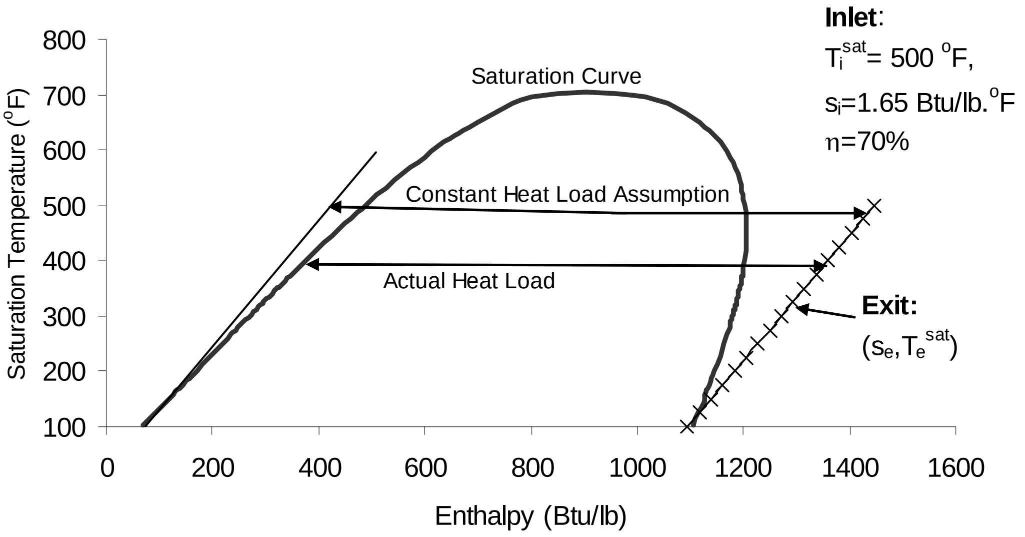

Figure 5.2 Steam loads for a given turbine inlet conditions

Using that observation, Raissi (1994) suggested the T-H model to calculate the shaftwork power output which was oversimplified but motivated an improvement introduced by Mavromatis and Kokossis (1998) in the so called Turbine Hardware model. Although the results of that technique are more realistic, initializing the mass flow rates for the first iteration is a cumbersome simulation procedure based on Raissi's model. Their procedure iterates by correcting the losses through heat-energy balance equation.

Previous works have some limitations to their usability and applicability. Raissi's (1994) approach is over simplified. Based on the observation of Salisbury and Schenectady (1942), Raissi (1994) assumed that the heat load is for a given exit temperature is the same regardless of the saturation temperature. The evaluation of this constant heat load is cumbersome, as for a given exit temperature, the load at different saturation temperatures is calculated and the average is taken as the constant heat load with variation of ±4%. In that study, a constant factor ε which is the ratio of the specific power to the difference in the inlet and exit temperature difference is defined as shown in equation (5.2).

$$w = \varepsilon\,(T_{in}^{sat} - T_{out}^{sat}) \tag{5.2}$$

This constant is calculated by simulating a turbine with a given efficiency and plot its specific power output against different saturation temperature difference.

Mavromatis and Kokossis (1998) developed a better model. Their model is initiated by the results of Raissi (1994) and improves iteratively using heat and energy balance equation. They also assumed the incoming flow to the steam main as saturated vapor which is not always the case.

The algorithm developed in this work is based on correlations of thermodynamic properties. The main difficulty in such procedures is how to count for irreversibilities to calculate the outlet enthalpies after subtracting the delivered power. While in previous works (Mavromatis and Kokossis 1998) these losses were counted for by using heat-energy balance, in this work correlations of entropies have proved very useful. Moreover, this work is applicable to more realistic problems with different pressure levels and heat is required or extracted to each level at a minimum temperature.

Targeting by Using Steam Properties

This work is mainly dependent on utilizing the concept of entropy and Gibbs rule in determining thermodynamic properties. This approach will furnish more accurate estimation of the thermodynamic properties without the cumbersome simulation of determining the constants used in previous works. The isentropic efficiency can also be expressed as the change of enthalpies which are, in turn, expressed as functions of entropy and saturation temperatures.

Figure 5.3 shows a thermodynamic expansion on a T-S diagram. The step s_1-s_2' shows an isentropic expansion. An isentropic process is an ideal case where thre is not any kind of irreversibilities such as mechanical friction and heat losses. Step s_1-s_2' is a better representation of what happens in reality. The outlet of the expander (turbine) is shifted to the right which indicates increase of entropy (state of disorder) caused by losses. The isentropic efficiency is basically the ratio of the enthalpy difference of step s_1-s_2' to that of step s_1-s_2. If the enthaly is expressed as a function of entropy, the efficiency will then be able to be embeded in the calculations of the output properties of the turbine.

The actual enthalpy is calculated from the knowledge of the isentropic enthalpies and efficiency.

$$h_{l,actual} = h_{l-1,isentropic} - \eta\,(h_{l-1,isentropic} - h_{l,isentropioc}) \tag{5.3}$$

The isentropic enthalpy will be calculated using equation (3.2).

$$h = 0.2029\ T_{sat}\ s^{3.647} + 817.35 \tag{3.2}$$

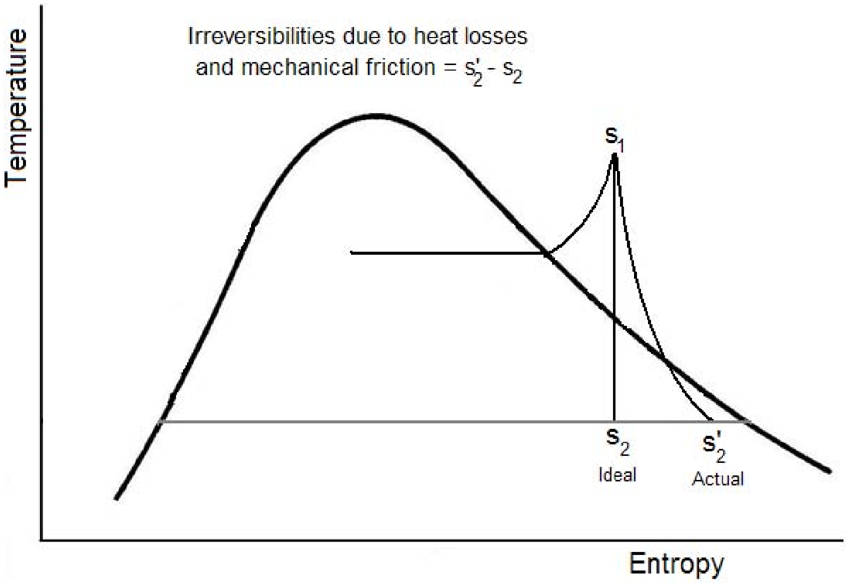
Irreversibilities due to heat losses
and mechanical friction = s'2 - s2
Temperature
s1
s2
Ideal
s'2
Actual
Entropy

Figure 5.3 The T-s diagram of a thermodynamic cycle

At the boiler, for a given saturation temperature and steam temperature, the entropy can be obtained with the aid of steam tables or simply by using (3.7). At lower levels, with the knowledge of the enthalpy of the previous expansion zone, equation (3.3) can be used to find the entropy:

$$s = \left(\frac{h - 817.35}{0.2029\, T_{sat}} \right)^{0.2742} \tag{3.7}$$

The enthalpy here is the actual enthalpy at the exit of the turbine in the previous expansion zone or simply mean enthalpy if the steam main has some streams induced from the process.

In the case of induction, the enthalpy of the steam main's outlet stream will be:

$$h_{l,out} = \frac{h_{l,in}\, M_z + h_{1,out} m_l}{M_z + m_l} \tag{5.4}$$

where M_Z is the mass flowing through the previous zone and m_l is the mass induced to the steam main (Figure 5.4).

It is very important to bear in mind that each zone has the same index of the outlet steam main. This is very important in tracing the algorithm as each zone has to be expressed relative to the neighboring steam mains or vice versa. Of course, in this case zone 1 is located between the boiler and the highest temperature steam main and it is a flat zone (has no expansion).

Figure 5.4 shows a schematic of the utility system layout. There is an expansion zone between two pressure levels. The number of steam mains usually does not exceed four.

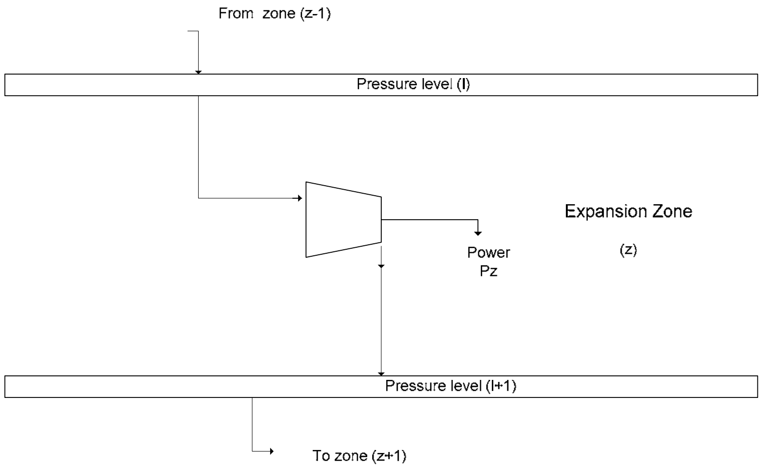

Figure 5.4 The expansion zone located between two pressure levels

The isentropic efficiency is a function of the load and for fixed values of flowrates, it would be better to consider the highest efficiency assuming using turbines for which the calculated flowrate will be the full load. As pointed out in chapter IV, the efficiency correlation is different for different turbines. Without any loss of generality, the efficiency expression suggested by Mavromatis and Kokossis (1998) will be sued here.

This expression is based on Willan's line that correlates the load to the power output. The isentropic efficiency depends on the turbine and the manufacturer. Different expressions are in the literature. The one used here is taken from Mavromatis and Kokossis (1998) and is explained in appendix A.

$$\eta_{is} = \frac{6}{5B}\left(1 - \frac{A}{\Delta h_{is}\, M^{\max}}\right)\left(1 - \frac{M^{\max}}{6M}\right) \qquad (A.1)$$

For full load efficiency, this equation will be

$$\eta_{is} = \frac{1}{B}\left(1 - \frac{A}{\Delta h_{is}\, M^{\max}}\right) \qquad (A.2)$$

where A and B are constants that are dependent on the turbine and are functions of the inlet saturation temperature. The flowrate is measured in lb/s. A good approximation, within 2%, is given by the following straight line segments (Mavromatis and Kokossis 1998; Varbanov et al. 2004):

$$A = a_0 + a_1\, T^{sat} \qquad (A.3.a)$$

$$B = a_2 + a_3\, T^{sat} \qquad (A.3.b)$$

The values of these constants are given in appendix A.

Extracted Mass

The steam can be either induced or extracted at a steam main i.e. exported for heating, the quantities are shown in the schematic diagram of the steam main in Figure 5.5. The calculation of the extracted steam is relatively easier than that of the induced steam for the extracted steam takes the properties of the source expansion zone while in the case of the induced steam the properties of the steam main properties are disturbed with the mixing of induced stream.

From the knowledge of the entropy of the previous steam main and the efficiency of the preceding zone, the input actual and isentropic enthalpies to the steam mains are calculated. The heat load will be the difference between the actual enthalpy and the saturated liquid at that level.

The (l) indicates the pressure level while (z) indicates the expansion zone. In the algorithm, the iteration loop could go from zone to zone and the level number is expressed in terms of the zone number or vice versa. In general, only the efficiency and the power produced are zone properties while other quantities are steam main properties.

The actual input enthalpy (h_{ina}(l)) and entropy (S_{in}(l))of steam main (l) are usually provided from the calculations of the previous steam main. The input isentropic enthalpy is calculated from equation (3.2). Then the efficiency which is a function of the difference in isentropic enthalpies is calculated. The actual enthalpy (which will serve as the input enthalpy for the next zone) is then calculated using the isentropic enthalpies and efficiency. The input entropy of the next zone is also calculated as a function of enthalpy and saturated temperature using equation (3.7).

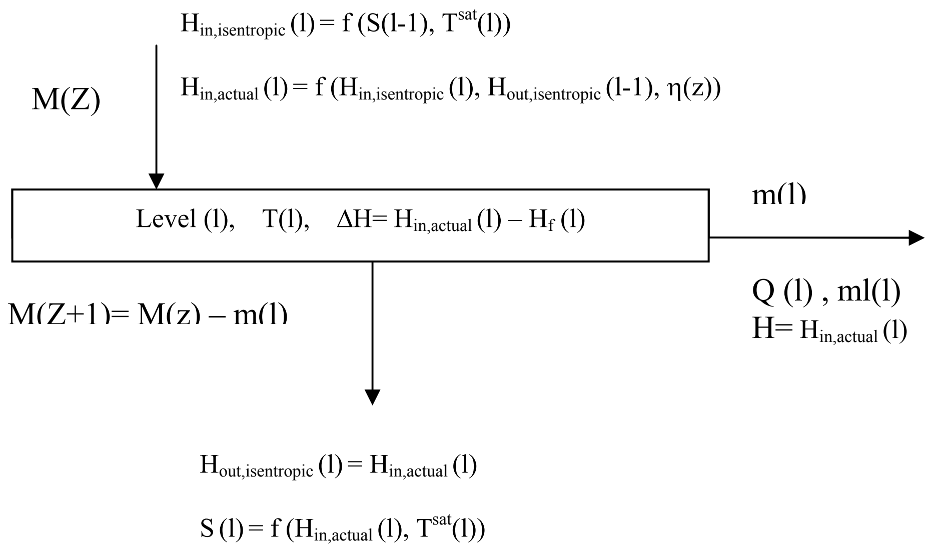
$H_{in,isentropic}$ (l) = f (S(l-1), T^{sat}(l))
$H_{in,actual}$ (l) = f ($H_{in,isentropic}$ (l), $H_{out,isentropic}$ (l-1), η(z))
M(Z)
m(l)
Level (l), T(l), ΔH= $H_{in,actual}$ (l) – H_f (l)
Q (l) , ml(l)
H= $H_{in,actual}$ (l)
M(Z+1)= M(z) – m(l)
$H_{out,isentropic}$ (l) = $H_{in,actual}$ (l)
S (l) = f ($H_{in,actual}$ (l), T^{sat}(l))

Figure 5.5 A steam main at which steam is extracted

Induced Mass

In the case where there is induced stream (Figure 5.6), the calculation is different. When a stream is split, the sub-streams carry the same intensive properties as those of the main one while in mixing, the properties of the mixture take intermediate values of the properties of the united streams.

The induced stream is provided at a certain condition and is mixed with the inlet stream from the previous zone. The outlet enthalpy (of the mixed streams) from the steam main is calculated using equation (5.4) and the outlet entropy will be calculated using equation (3.7). Then the same procedure used in the extraction steam main will be followed i.e. finding the isentropic enthalpy difference and efficiency. The actual enthalpy and entropy of the inlet stream of the next steam main is also calculated.

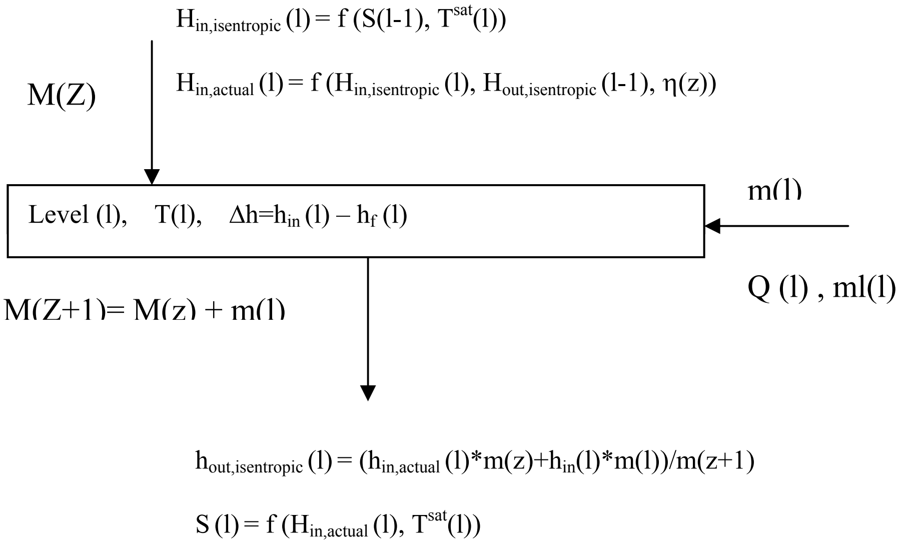
$H_{in,isentropic}$ (l) = f (S(l-1), T^{sat}(l))
M(Z)
$H_{in,actual}$ (l) = f ($H_{in,isentropic}$ (l), $H_{out,isentropic}$ (l-1), η(z))
Level (l), T(l), $\Delta h = h_{in}$ (l) – h_f (l)
m(l)
Q (l) , ml(l)
M(Z+1)= M(z) + m(l)
$h_{out,isentropic}$ (l) = ($h_{in,actual}$ (l)*m(z)+h_{in}(l)*m(l))/m(z+1)
S (l) = f ($H_{in,actual}$ (l), T^{sat}(l))

Figure 5.6 A steam main at which steam is induced

Non-Heating Steam Requirement

The steam required by the process for non-heating purposes is usually demanded as a mass flows (m_p) at certain pressure values. These values will be simply added to the steam exported to the process. If there is no steam main having the same pressure as the one required by the process, the steam will be taken from the steam main that has a pressure value just above the one required by the process.

The Targeting Algorithm

Before experiencing a practical case study, the reader may find the following account of the algorithm a little vague. It could be more tactical if this description is read in parallel with the following case study so as to have more sense of how the procedure works.

The layout shown in Figure 5.7 should be referred to in order to understand how the indices of the different levels and zones interrelate.

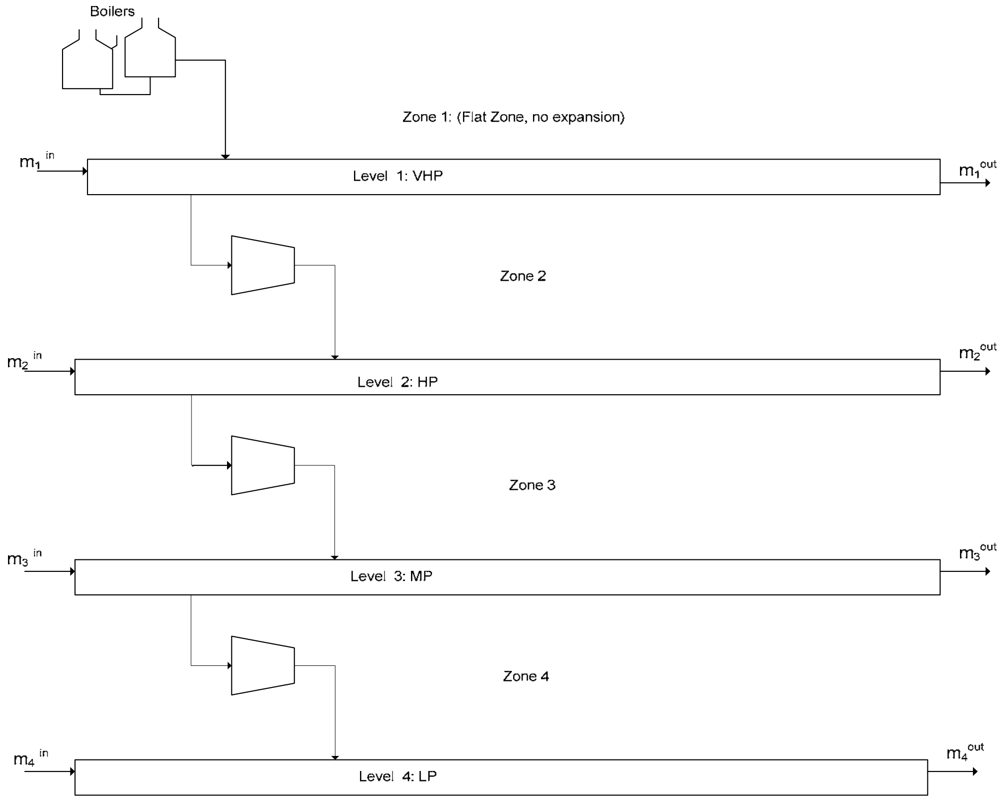

Figure 5.7 A typical utility system with four levels and three expansion zones

Figure 5.8 shows the main steps in the algorithm. After applying the integration techniques, the minimum heat requirements along with the hot streams conditions coming from the plant are input to the algorithm. Then these streams are allocated to the suitable steam main level. An initial guess of the mass flowrates coming out of the steam mains is estimated assuming isentropic expansions throughout the levels. Then the efficiency is corrected and better estimates of the mass flowrates are evaluated. Next, the stopping criterion is checked to decide on terminating or looping. These steps are detailed in the following account.

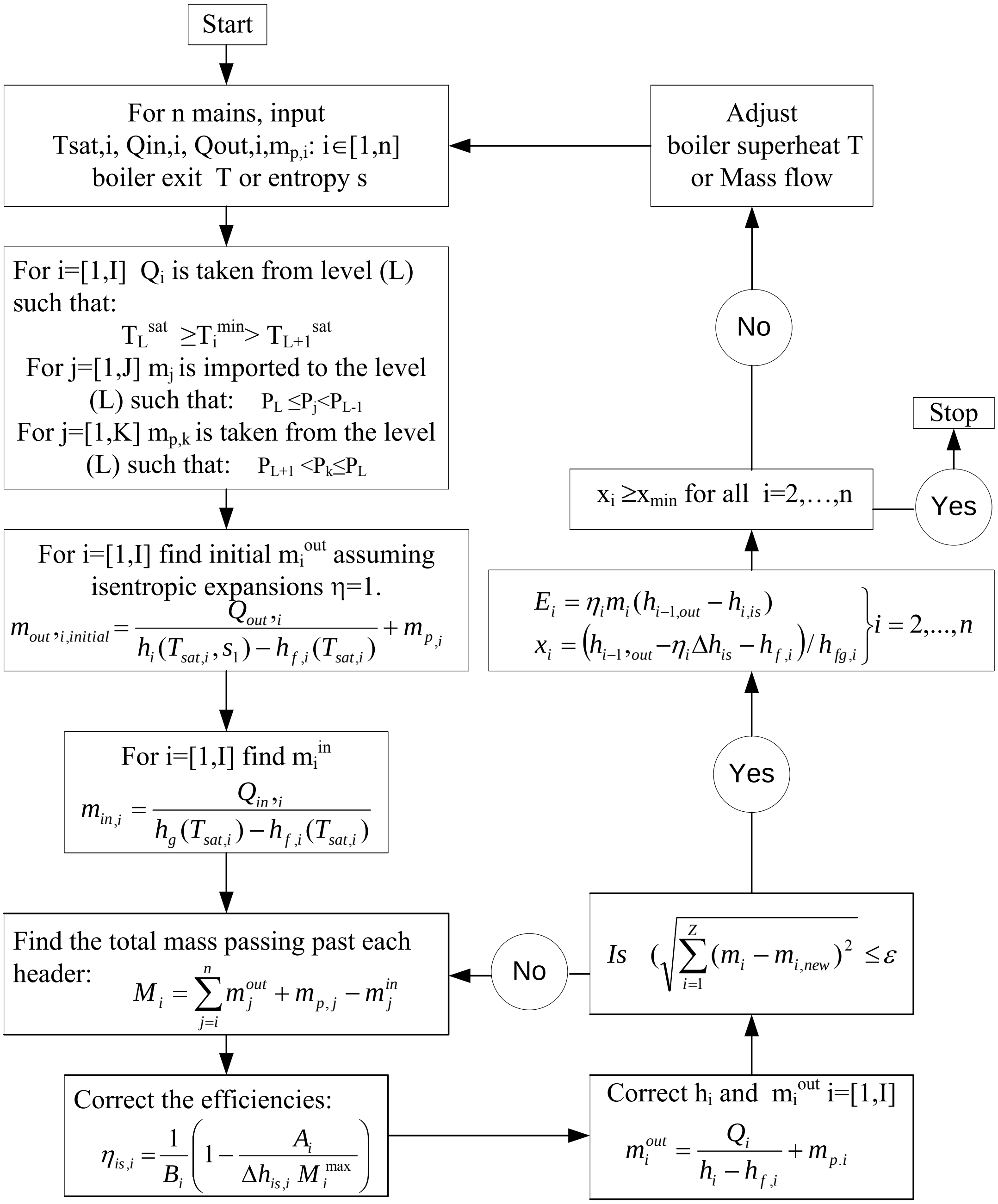
Start
For n mains, input
Tsat,i, Qin,i, Qout,i,$m_{p,i}$: i∈[1,n]
boiler exit T or entropy s
For i=[1,I] Q_i is taken from level (L) such that:
$T_L^{sat} \geq T_i^{min} > T_{L+1}^{sat}$
For j=[1,J] m_j is imported to the level (L) such that: $P_L \leq P_j < P_{L-1}$
For j=[1,K] $m_{p,k}$ is taken from the level (L) such that: $P_{L+1} < P_k \leq P_L$
For i=[1,I] find initial m_i^{out} assuming isentropic expansions η=1.
$m_{out},_{i,initial} = \frac{Q_{out},_i}{h_i(T_{sat,i}, s_1) - h_{f,i}(T_{sat,i})} + m_{p,i}$
For i=[1,I] find m_i^{in}
$m_{in,i} = \frac{Q_{in},_i}{h_g(T_{sat,i}) - h_{f,i}(T_{sat,i})}$
Find the total mass passing past each header: $M_i = \sum_{j=i}^{n} m_j^{out} + m_{p,j} - m_j^{in}$
Correct the efficiencies:
$\eta_{is,i} = \frac{1}{B_i}\left(1 - \frac{A_i}{\Delta h_{is,i} M_i^{max}}\right)$
Correct h_i and m_i^{out} i=[1,I]
$m_i^{out} = \frac{Q_i}{h_i - h_{f,i}} + m_{p.i}$
Is $(\sqrt{\sum_{i=1}^{Z}(m_i - m_{i,new})^2} \leq \varepsilon$
No
Yes
$E_i = \eta_i m_i (h_{i-1,out} - h_{i,is})$
$x_i = (h_{i-1},_{out} - \eta_i \Delta h_{is} - h_{f,i}) / h_{fg,i}$
$i = 2,...,n$
$x_i \geq x_{min}$ for all i=2,…,n
Yes
Stop
No
Adjust
boiler superheat T
or Mass flow

Figure 5.8 Flow diagram of the algorithm of targeting the mass flow rates

Input

The number of the pressure levels (L) = number of expansion zones (Z)

The saturation temperature at each pressure level

$T_l : l \in [1,L]$

The process heat requirement with the minimum temperature for each requirement

$Q_{i,}$, T_i^{min}

The mass flowrate and state of each heat stream discharged from the process:

M_i, P_i, T_i

The mass flowrates of the steam required by the process for non-heating purposes with the required pressure values: M_{pi}, P_{pi}

The entropy of the steam produced in the boiler s_l=$s_{1.}$

The amount of superheat to be used in the boiler, ΔT_{sh}

Procedure

Step 1: The incoming streams are distributed such that each stream will go to the steam main whose pressure is the same or just below the pressure of the stream.

Step 2: The outgoing streams will be extracted from the steam mains whose saturation temperature is the same or just above the temperature of the outgoing stream.

Step 3: The steam flowrates required by the process for non-heating purposes will taken from the steam main that has the same pressure value as the one required by the process or just above it.

Step 4: Find initial estimates of mass flowrates assuming isentropic expansions and a heat load which is the difference between the steam main isentropic enthalpy and saturated liquid enthalpy. Notice that in this initial step the incoming flows are being ignored and so the mass flowrate of the outgoing flows are calculated as if the incoming flows are inexistent.

$$m_{l,initial} = \frac{Q_l}{h_l(T_{sat,l}, s_1) - h_{f,l}(T_{sat,l})} \quad l \in [1, L] \tag{5.5}$$

Step 5: Calculate the steam passing by each zone using the equation:

$$M_z = \sum_{n=z}^{L} m_n^{out} + \sum_{n=z}^{L} m_{p,n} - \sum_{n=z}^{L} m_n^{in} \tag{5.6}$$

whereas m_i^{out} is the mass flowrate of the steam leaving the steam main for the process, m_i^{in} is the mass flowrate of the steam leaving the process to the steam main and $m_{p,I}$ is the mass flowrate required by the process for non-heating purposes.

Step 6: Correct the efficiency by using equation (A.2) or (A.3) if the maximum load is used. The procedure depends on wither mass is extracted or induced.

Step 7: From the second iteration through convergence, the steps are repetitive in manner until they meet similar results in two consecutive iterations as described below.

```
$for\ z = 1$
This zone is a flat zone (has no expansion), it is connecting the boiler to
 the first header whose pressure  is the same as that of the boiler. The effciency
 of this zone is assumed 100%.
The enthalpy at the first level
$h_{1,out} = \frac{M_1 h_{1,out} + m_1^{in} h_1^{in}}{M_1 + m_1^{in}}$
$s_{1,out} = f(h_{1,out}, T_1^{sat})$
$m_1^{in} = \frac{Q_1^{in}}{h_{1,out} - h_{f,1}}$

$for\ z \in [1, z]$
        $s_{z,in} = s_{z-1,out}$
        $h_{s,z,in} = f(T_{z-1}^{sat}, s_{z,in})$ using (3.2)
        $h_{s,z,out} = f(T_z^{sat}, s_{z,in})$ using (3.2)
        $\Delta h_{s,z} = h_{s,z,in} - h_{s,z,out}$
        $\eta_z = f(M_z, \Delta h_{s,z})$ using (A.1)
        $h_{a,z,out} = h_{s,z,in} - \eta_z \Delta h_{s,z}$ using (5.3)
        $h_{z,out} = \frac{M_z h_{a,z,out} + m_z^{in} h_z^{in}}{M_z + m_z^{in}}$
        $s_{z,out} = f(h_{z,out}, T_z^{sat})$ using (3.7)
        $m_z^{in} = \frac{Q_z^{in}}{h_{z,out} - h_{f,z}}$
$end$
```

Figure 5.9 Finding the thermodynamic properties of the different levels in a utility system

Step 5: Test the stopping criterion.

```
if ( $\sqrt{\sum_{i=1}^{Z}(m_i - m_{i,new})^2} \le \varepsilon$
        The net power $E_z = \eta_z \, m_z \, (h_{z-1,out,isentropic} - h_{z,in,isentropic}) \quad z \in [2, Z]$        (5.7)
        STOP
else
        GO TO Step 4
```

When the algorithm terminates, the steam quality is checked at each turbine's exit. If it falls below the allowed minimum, the superheat temperature or mass flow is increased in the boiler. The calculations for each zone are given in Figure 5.9.

Case Study

In a utility system with the following steam mains:

P_1=200 psi T_{sat1}=382 ^oF

P_2=100 psi T_{sat2}= 328 ^oF

P_3=40 psi T_{sat3}= 267 ^oF

P_4= 15 psi T_{sat4}= 213 ^oF

The following streams are coming from the process:

m_1=14,000 lb/hr at P_1 =60 psi and T_1=318 ^oF (with superheat) h_1=1189 Btu/lb

m_2=11,000 lb/hr at P_2 =110 psi and T_2=335 ^oF as saturated vapor h_2=1188 Btu/lb

The followings are the streams needed by the process:

Q_1=10 MMBtu/hr at T_1=250 ^oF

Q_2=30 MMBtu/hr at T_2=300 ^oF

Q_3=20 MMBtu/hr at T_3=370 ^oF

Q_4=15 MMBtu/hr at T_4=200 ^oF

The process requires the following mass flowrates for the non-heating uses of steam:

$m_{p,1}$ =1000 lb/hr at 150 psi

$m_{p,2}$ =5000 lb/hr at 100 psi

$m_{p,3}$ =3000 lb/hr at 75 psi

The steam quality at the exit of each turbine is not to fall below 0.9 and maximum mass flow rate is assumed i.e. equation (A.2) will be used for the efficiency.

Step 1: The incoming streams will be distributed as follows:

m_2=700 lb/hr at P_2 =110 psi and T_2=335 ^oF will go to the second steam main.

m_1=500 lb/hr at P_1 =60 psi and T_1=318 ^oF will go to the third steam main.

Step 2: The outgoing streams will be distributed as follows:

Q_3=300,000 Btu/hr at T_3=370 ^oF will be extracted from the first steam main

Q_2=500,000 Btu/hr at T_2=300 ^oF will be extracted from the second steam main

Q_1=700,000 Btu/hr at T_1=250 ^oF will be extracted from the third steam main

Q_4=400,000 Btu/hr at T_4=200 ^oF will be extracted from the fourth steam main

Step 3: The non-heating mass flowrates required by the process will be imported from the steam main that has the same pressure required by the process or just above it:

$m_{p,1}$ =1000 lb/hr at 150 psi from the first steam main

$m_{p,2}$ =5000 lb/hr at 100 psi from the second steam main

$m_{p,3}$ =3000 lb/hr at 75 psi from the second steam main

Step 4: (First Iteration) Find the initial estimates of the mass flowrates of the outgoing steam (Table 5.1) assuming isentropic expansion throughout the utility system while ignoring the incoming streams.

Table 5.1 The Initial Estimates of the Mass Flowrates

P	**T_{sat}**	**S**	**h**	**h_f**	**Q**	**m^{out}**
(psi)	**(^oF)**	**(Btu/lb.^{o}F)**	**(Btu/lb)**	**(Btu/lb)**	**(MMBtu/hr)**	**(lb/hr)**
200	382	1.5668	1216.0	355.33	10	23,248
100	328	1.5668	1159.6	297.51	30	34,798
40	267	1.5668	1096.0	235.69	20	11,619
15	213	1.5668	1039.6	184.07	15	17,533

Step 5: Calculate the steam passing by each zone (Table 5.2).

Table 5.2 The First Iteration of the Mass Flowrates Passing by Each Zone

Zone (z_i)	**m_z^{in} (lb/hr)**	**m_z^p (lb/hr)**	**m_z^{out} (lb/hr)**	**M_z (lb/hr)**
1 (boiler-200 psi)	0	6,000	23,248	71,198
2 (200-100)	11,000	3,000	34,798	41,950
3 (100-40)	14,000	0	11,619	15,152
4 (40-15)	0	0	17,533	17,533

Step 7: (Second iteration) Correct the efficiency by using equation (A.2). The procedure depends on whether mass is extracted or induced (Table 5.3).

Table 5.3 The Corrected Efficiencies after the First Iteration

	T_1^{sat}	T_2^{sat}	$S_{z-1,out}$	Δh_s				Power
Zone	(°F)	(°F)	(Btu/lb.F)	(Btu/lb)	A	B	η	(MMBtu/hr)
2 (200-100)	382	328	1.5668	55.75	0.0965	1.2846	0.669	1.564
3 (100-40)	328	267	1.5877	66.81	0.0614	1.2390	0.640	0.648
4 (40-15)	267	213	1.6559	68.95	0.02175	1.1874	0.790	0.955

Step 6: The steps descried in the above algorithm are performed to correct the mass flowrates as given in Table 5.4.

Table 5.4 Recalculation of the New Mass Flowrates of the Second Iteration

		T_1^{sat}	T_2^{sat}	M_z		$S_{z,in}$
Zone/Level	Zone	(°F)	(°F)	(lb/hr)	η	(Btu/lb.F)
1	R-VH	382	382	68,236	1	1.5668
2	VH-H	382	328	38,998	0.660	1.5668
3	H-M	328	267	13,027	0.613	1.5883
4	M-L	267	213	16,236	0.787	1.6598

The algorithm will keep iterating until convergence. Table 5.5 shows the final results.

Table 5.5 The Final Result of the Case Study

Zone/Level	Zone	T_1^{sat} (°F)	m_z^{out} (lb/hr)	M_z (lb/hr)	η	Power (MMBtu/hr)
1	R-VH	382	23,239	68,130	1	0
2	VH-H	382	33,952	38,891	0.660	1.431
3	H-M	328	10,754	12,940	0.612	0.530
4	M-L	267	16,185	16,185	0.787	0.886

The algorithm converges in the fourth iterations. The steam qualities, at the exit of each zone, are found to be 0.97414, 0.96196 and 0.96762 respectively. This algorithm is a very general one. in the next chapter, it will be used with the grand composite curve and then its accuracy will be compared with a previous work.

Conclusion

The algorithm developed here provides a consistent, general procedure for determining the mass flowrates and the efficiencies of the turbines used. This algorithm utilizes the relationship of the entropy with the enthalpy and the isentropic efficiency. Although the algorithm is initiated assuming 100% efficiency, it shows a very rapid convergence. It is superior to previous works in that it does not require cumbersome simulation for initiation, accurate and it can be traced easily which enhance its programmability.

CHAPTER VI

OPTIMUM LEVELS OF STEAM MAINS

Introduction

When cogeneration is involved, locating the different steam mains at the optimum pressure levels becomes a very challenging problem. This is so because of the big number of variables and the different tradeoffs which that result in different possibilities of objective criteria, namely minimum fuel and maximum power. The complexity of this problem stems from its high nonlinearity. This problem reportedly was first investigated by Nisho (Nisho 1977; Nisho and Johnson 1979). An absolute global solution to this problem needs the solver to look into the hardware performance, process needs and accurate energy and mass balances. Due to its complexity, most of the approaches used follow exhaustive enumeration paths or relaxing some of the variables.

The early optimization approaches disregarded the variation in the pressure levels. Papoulias and Grossman (1983) used a mixed integer linear (MILP) approach to optimizing the utility system without accounting for the different levels.

A very important aspect of the cogeneration in the utility system is to provide specific amounts of heat required by the process at each level. In order to further the profitability, heat integration is first performed. The advent of heat integration is attributed to Linnhoff et al. (Hohman 1971; Linnhoff and Hindmarsh 1983; Umeda et al. 1979). In heat integration, cold streams and hot streams are matched so as to minimize the external utilities. This is done by using the thermal pinch diagram. This technique however defines the minimum utility but a more suitable way of defining the optimum pressure levels of the utility is the grand composite curve (GCC) (Smith 2005), the GCC was introduced by Morton and Linhoff (Morton and Linnhoff 1984) which defines how much heat of the external utilities should be provided or extracted from which temperature range. Raissi (1994) developed the temperature enthalpy (T-H) model for the utility system. In that model, Raissi (1994) assumed a constant specific heat load at the turbine outlet, an assumption that can lead to up to 30% error (Mavromatis and Kokossis 1998). In optimizing the steam main level, Raissi defines two extremes, namely the minimum fuel requirement (MFR) and the minimum utilities cost (MUC). In the first case, the fuel consumption is minimized with no cogeneration. That case however may not insure a minimum utilities cost (accounting for power and heat) and hence the second case which involves cogeneration with additional fuel consumption. Raissi however did not give any rigorous optimization method of the steam mains' pressure values. Mavromatis and Kokossis (1998) used the turbine hardware model for the targeting of the utility system. Their module is initiated by the T-H model and then an iterative procedure is followed. To optimize the steam levels, Mavromatis and Kokossis (1998) used the same approaching with exhaustive search. Shang and Kokossis (2004) used the pinch diagram. The same method is ued as in the cascade procedure in the GCC but by involving the steam main within the interval balance. The optimum location is assumed to be at one of the stream target or supply temperatures. This method apparently is exhaustive when a large number of streams are involved especially in the case of multiple steam mains.

In this work, the grand composite curve is used with a mathematical algorithm to locate the optimum steam mains' levels.

Problem Statement

After performing heat integration for a set of hot and cold streams, the minimum utilities are identified to be supplied by external sources. The thermal pinch is used to check the minimum heating and cooling utilities, this diagram however only matches the total heat required by the cold streams to the total heat to be recovered from the hot streams but it doesn't specify how much heat is needed for each specific temperature range. The grand composite curve (GCC) is built by analyzing the heat deficit or surplice at each temperature interval (specified by the streams supply and target temperatures) and accordingly. The composition of the GCC is based on how much heat is needed or to be removed at each of the temperature intervals and so the temperatures to which the external utilities are allocated are known. For cogeneration to be incorporated, determining the location of the steam mains becomes a challenging task. The designer would have to do some tradeoffs among cost of fuel, power produced and capital.

In this work, a mathematical algorithm is used to locate the optimum location of the steam main based on the grand composite curve.

Observations

The power generation through steam turbines is a function of many parameters including the pressure difference, the superheat temperature, the turbine efficiency and the steam load. Before trying to optimize the steam mains pressure values, the effect of these parameters is tested. Testing the relationship between these parameters and the amount of power generated from the turbine will help reduce the size of the optimization problem. In this investigation, the change of such parameters with the main pressure values is performed using the developed correlation of steam properties.

The Effect of the Outlet Pressure Value

The power produced by a stem turbine is proportional to the pressure difference across the turbine.

The specific power for a given isentropic efficiency between two steam mains is: P=η (h_1-h_{2s}) the isentropic enthalpy difference of the backpressure turbine can be calculated by using equation (3.2):

$$\Delta h_{is} = 0.2029\ (T_{sat1} - T_{sat2})\, s_1^{\ 3.647} \qquad (6.1)$$

and hence the specific power will be:

$$p = 0.2029\ \eta (T_{sat1} - T_{sat2})\, s_1^{\ 3.647} = [-0.2029\ \eta\ \ s_1^{\ 3.647}]T_{sat2} + [0.2029\ \eta\ \ s_1^{\ 3.647} T_{sat1}] \quad (6.2)$$

The right hand side form clearly shows that the change of the specific power for a turbine at given inlet condition, with the outlet pressure has a linear relationship, with the slope and the vertical-axis intercept given by the first and second square brackets, respectively.

This relationship is depicted in Figure 6.1. Using that equation, a family of lines for a group of inlet pressure values can be easily produced. It also helps calculate the outlet power for a given turbine and fixed boiler operation parameters.

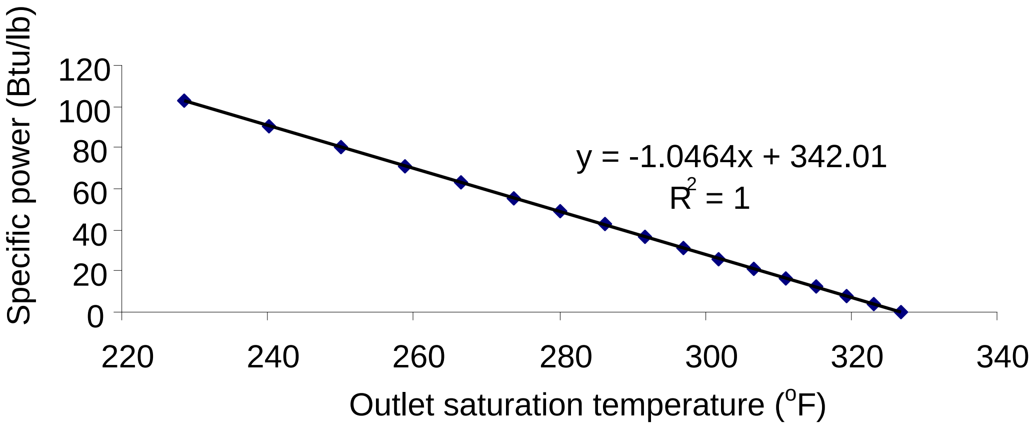

Figure 6.1 The change of the specific power with the outlet saturation temperature (for P_{in}=100 psi (T_{sat}=327 °F), T_{in}=550 °F and η=70%)

The Change of the Specific Power with the Inlet Temperature

Having a high superheat temperature at the turbine's inlet increases the inlet enthalpy and consequently it also incresaes the power produced by the turbine.

By using the correlations of the thermodynamic properties of the saturated steam, the specific power for a given isentropic efficiency between two steam mains is P = η (h_1-h_{2s}). The isentropic enthalpy difference of backpressure turbine is calculated from

$$h = 0.2029\ T_{sat}\ s^{3.647} + 817.35 \qquad (3.2)$$

and also from (3.3) :

$$h = A_T\, T + B_T \tag{3.3}$$

$$A_T = 6\times10^{-7} T_{Sat}^2 - 0.0002\, T_{sat} + 0.537$$
$$B_T = -0.0014\, T_{sat}^2 + 0.6843\, T_{sat} + 941.73$$

Substituting these equations in the expression will yield

$$p = \eta(A_T\, T + B_T - (\frac{T_{sat2}}{T_{sat1}}(A_T\, T + B_T - 817.35) + 817.35))$$
$$p = [\eta\, A_T(1-\frac{T_{sat2}}{T_{sat1}})]\ T\ + [\eta\, B_T(1-\frac{T_{sat2}}{T_{sat1}})\ - 817.35\eta(1-\frac{T_{sat2}}{T_{sat1}})] \tag{3.4}$$

This relationship is linear (Figure 6.2) with the slope and the intercept with the vertical axis given by in the square brackets respectively.

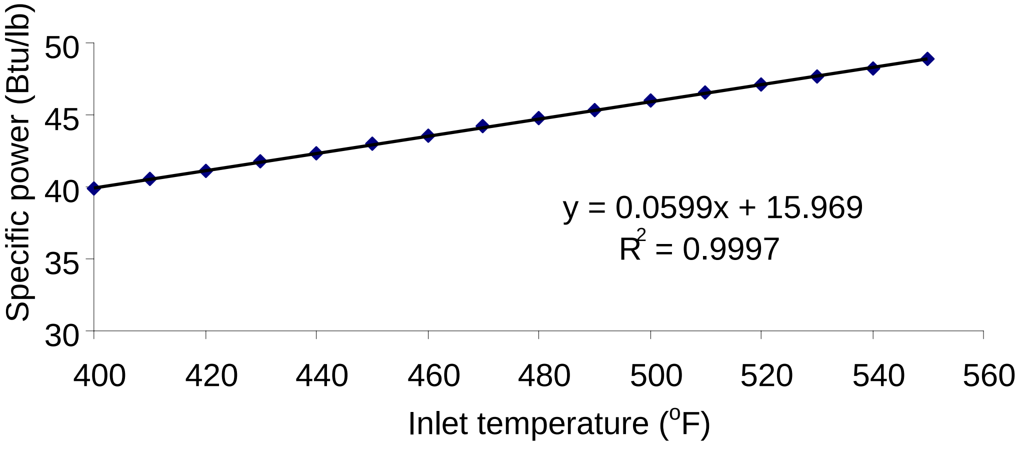

Figure 6.2 The change of the specific power with the inlet temperature (for P_{in}=100 psi, P_{out}=50 psi and η=70%)

Intermediate Steam Mains

In a multi-steam mains utility, the steam is passed on from the boiler to the different steam mains through steam turbines. These turbines can have different arrangements, the same amount of steam can be passed on from the high pressure steam main (Figure 6.3) to the low pressure steam main either directly or through the medium pressure steam main.

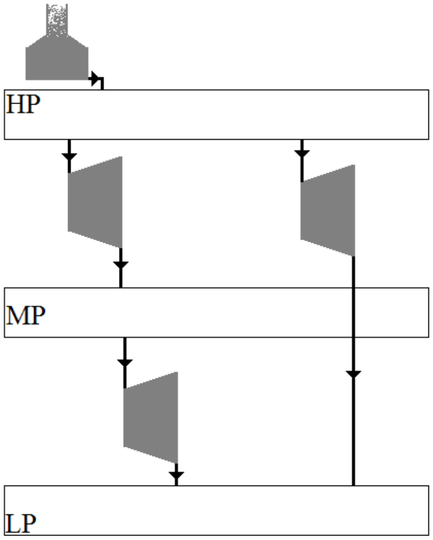

Figure 6.3 The different arrangement of steam turbines

The different arrangements were checked for different pressure values using the algorithm that will be discussed later on. Assuming a constant efficiency of the steam turbine, i.e. the efficiency does not change with load, both arrangements shown on Figure 6.3 will give similar results, this is due to the fact that the power change linearly with the outlet pressure and so the intermediate steam main will serve just like any intermediate point on the path between the high pressure and the lower pressure mains. In reality, however, the efficiency is not constant. It is a function of the inlet pressure, the pressure difference and the mass flowrate. For a constant load (mass flow rate), different position of the intermediate pressure steam mains were tested, the deviation of the total power produced from each arrangement from the average is very small, provided that the pressure difference is wide enough to ensure feasibility. However, for a constant mass flow, having a number of intermediate turbines produces more power than having one single turbine that expands the steam from the high pressure steam main to the low pressure steam main. This is justified by the fact that for a constant load, the efficiency tends to increase by decreasing the steam inlet pressure. Given that, by letting the steam down through the intermediate turbines will improve the efficiency downwards provided that the difference in pressure between the steam mains is wide enough to maintain a less deviant isentropic enthalpy difference. This fact, however, does not absolutely impose this arrangement as the always feasible, simplicity of design, increase of power produced compared to the increase in capital cost investment associated with this arrangement, also plays an important role.

As will be discussed later, in optimizing the steam main levels, heat will always have to be supplied at each steam main and hence the same number of turbines will be needed; working either in parallel or in series. The optimization procedure, however, will assume operation in series. That decision should not endure in loss in generality since operation in parallel would entails nothing more than performing the same procedure for single expansion zones instead of doing it sequentially all at once.

Heat Integration

In order to save energy, the first step is to perform heat integration by finding the optimum matching of the hot streams with the cold ones. The thermal pinch diagram is used to identify the most possible saving. The shape of the thermal pinch diagram is very important for the investigation. A computer code is used here which takes in the supply temperature, target temperature and heat content and then produces the pinch diagram, grand composite curve and the pinch point and the minimum utilities required.

The perfect matching of the pinch diagram takes place when all streams, hot and cold, have normally distributed ranges with almost the same average with their heat content having the same average and are also normally distributed.

Figure 6.4 shows the thermal pinch diagram of 96 hot streams and 88 cold streams. The temperature of each set ranges from 5 degree to 400 degree with an average heat content of 225 unit heat per temperature degree. There is no single point for the pinch but the two curves meet in a straight line. Streams having that kind of distribution retain the $\int$-shaped curve; being almost perfectly parallel insures better integration potential, with the external utilities required being a small fraction of the integrated portion.

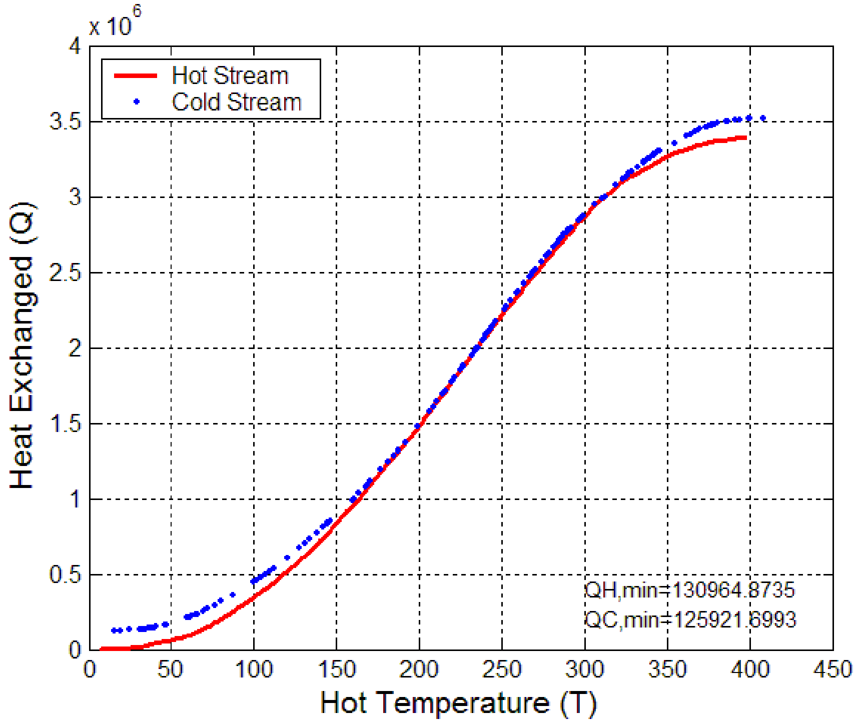

Figure 6.4 The thermal pinch diagram of normally distributed stream parameters

The construction of the pinch diagram is basically done by accumulating the amount of heat transferred between two temperature values for all streams, hence the similarity between the curve above with the accumulative distribution function.

Cogeneration is not always a viable option. This is because the power produced may not offset the capital and operation costs entailed by the incorporation of the cogeneration accessories. Accordingly, the temperature range, heat capacities and the amount of external utilities dictate whether or not cogeneration is feasible.

The scenario looks more promising to invest on cogeneration when the magnitude of the external duties required after integration is high enough and also the temperature range is wide enough. This situation is met when there is a divergence between the two curves. This situation comes by when the hot and cold stream average ranges do not overlap and also when their average heat duties are reasonably different. For a group of streams that have fairly closer ranges, external utilities will be needed if the average hot streams heat load is smaller than that of the cold. When the temperature ranges of both streams overlap in a smaller range, the situation will call for more utilities to be supplied to the streams outside the overlap.

Locating the Optimum Steam Levels

Cogeneration usually entails additional fuel consumption. This is because generating steam at high pressure and expanding it to the required pressure values will involve additional fuel consumption. Expanding the steam beyond the pinch point will even add the cost due to the shift of the two streams in order to allow expansion across the pinch.

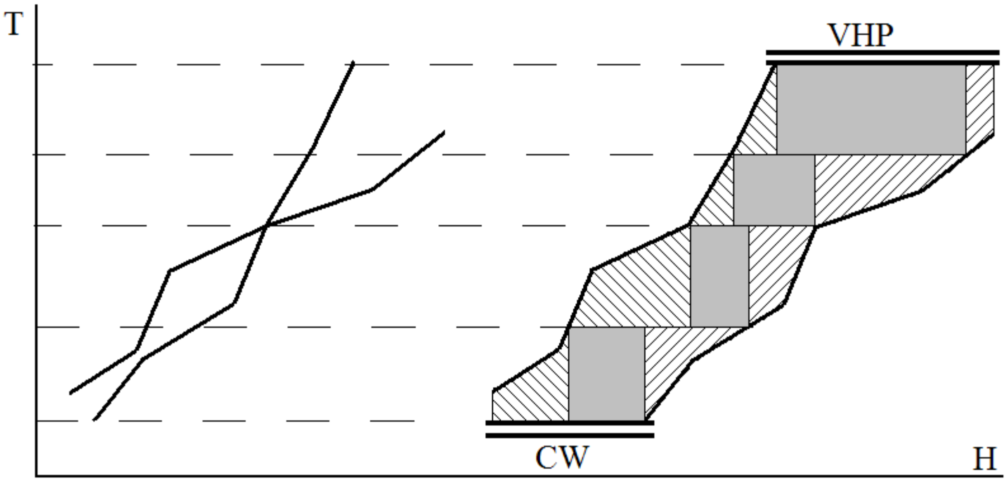

Figure 6.5 Steam utility with cogeneration

In Figure 6.5, the pinch diagram on the left does not involve any cogeneration unless an infinite number of mains are introduced which is an idealistic case. In order to involve cogeneration, the diagram on the right has the cold and hot composites split apart at the pinch in order to accommodate the limited number of mains. The more mains introduced, the smaller shifting is required at the pinch. The shape of the composite curves however, plays an important role (Raissi 1994; Smith 2005). As described before, if the cold and hot composite curves diverge around the pinch, the system can then easily accommodate a limited number of mains with a reasonable shifting of the composites at the pinch.

In Figure 6.5, steam is produced in the boiler at a very high pressure value and is then let down to the rest of the mains through steam turbines. The backward hatching implies the steam extracted from the hot composite to the mains, the gray area accounts for the power produced while the forward hatching implies the steam to be delivered to the cols composite from the mains. Since the shifting of the two composite curves is parallel, it is easily noticed that the amount of shifting at the pinch is the same as the increase the heating and the cooling utilities and hence the additional fuel consumption.

As mentioned before, the composite curve is not a robust tool for the selection of the utility levels. It shows the heating and cooling insufficiency without specifying at which temperature these utilities are to be delivered. The grand composite curve will be used for this purpose.

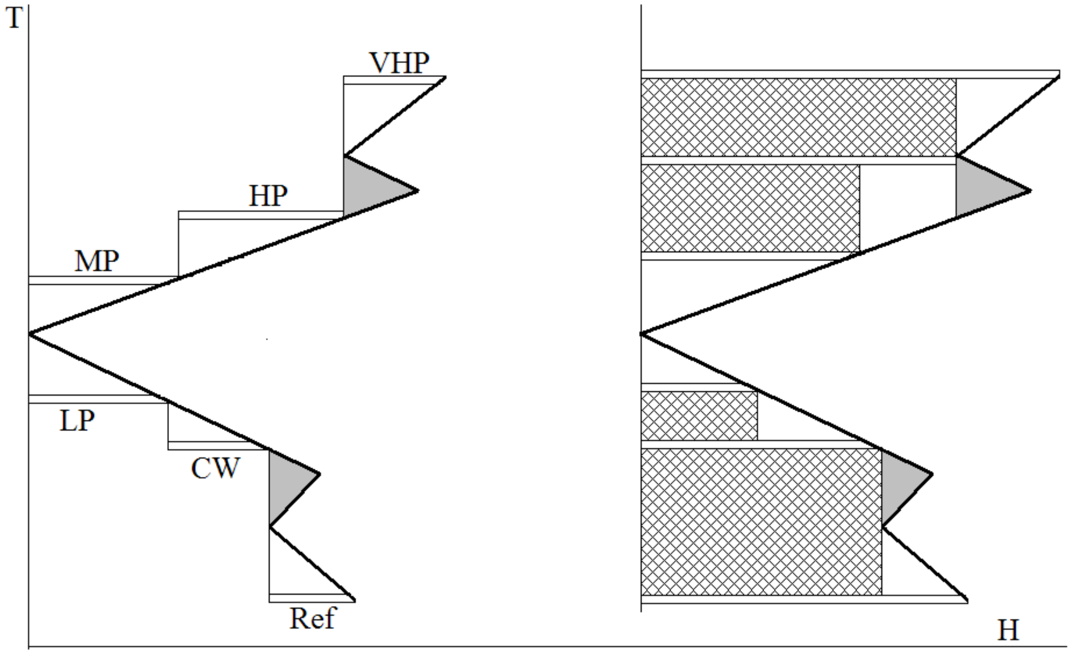

Figure 6.6 Using the GCC to select the pressure mains

Figure 6.6 shows the grand composite curve of a process. Each segment of the curve represents a deficit or a surplus at each temperature interval. The parts with a negative slope indicate a surplus of heat while the ones with positive slope indicate a deficiency of heat. The grey areas are called pockets and are the locations where parts or whole of some adjacent intervals are integrated. The left hand side GCC only includes utilities at different pressure levels with no cogeneration while the one on the left includes cogeneration. Above the pinch, steam is produced at the very high pressure level and then it is let down to the different steam mains while producing power and satisfying the process needs for heat. Below the pinch, the cooling fluid is let down to the different steam mains from the process. The coolant can be refrigerant, cooling water or even steam if cooling is required at relatively high temperature values. The hatched areas indicate power production. No steam main will be used across the pocket unless the temperature range is wide enough to locate a steam main.

In order to optimize the steam mains' locations, the targeting algorithm will be used in an exhaustive manner. Later on, the results will be compared to the area of the production potential to extract a less costly algorithm. The targeting algorithm is made very general. It can take multiples streams into and out of the steam main. Using the GCC will render each steam main either importing or exporting streams. Moreover, the inflows and the outflows will be taken as heat at the steam main temperature. The modified algorithm is shown in Figure 6.7.

Mavromatis and Kokossis (1998) developed a similar procedure to solve this problem. Their procedure is initiated with Raissi's (1994) method and then runs iteratively. Unlike the algorithm presented here, which depends on mathematical correlations of steam thermodynamic properties, Mavromatis and Kokossis' (1998) procedure is heavily dependent on heat and energy balance. In order to check the performance of both methods, a case study presented Mavromatis and Kokossis (1998) will be used.

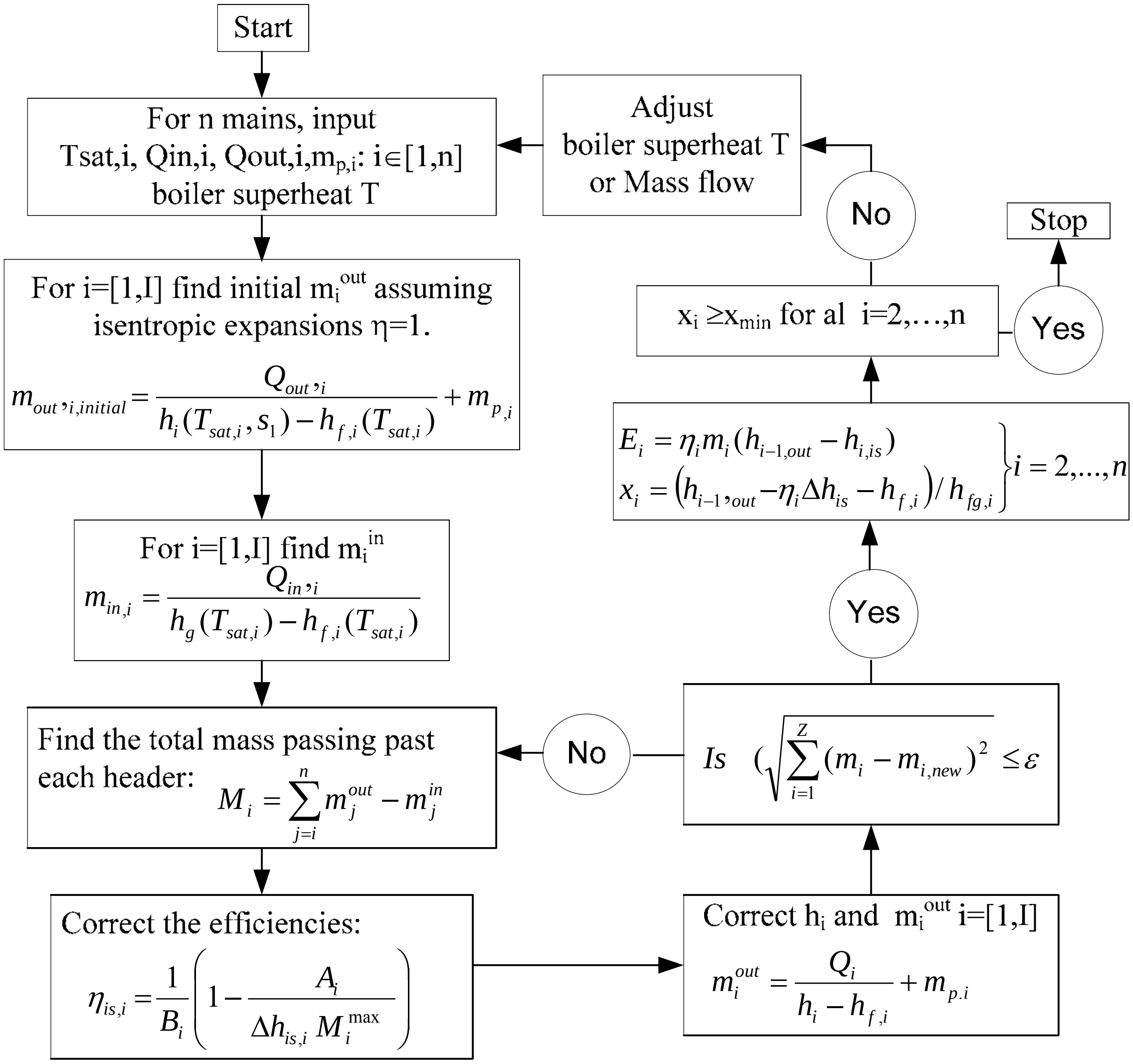
Start
For n mains, input
Tsat,i, Qin,i, Qout,i,$m_{p,i}$: i∈[1,n]
boiler superheat T
Adjust
boiler superheat T
or Mass flow
No
Stop
For i=[1,I] find initial m_i^{out} assuming
isentropic expansions η=1.
$m_{out,i,initial} = \frac{Q_{out,i}}{h_i(T_{sat,i}, s_1) - h_{f,i}(T_{sat,i})} + m_{p,i}$
$x_i \geq x_{min}$ for al i=2,…,n
Yes
$E_i = \eta_i m_i (h_{i-1,out} - h_{i,is})$
$x_i = \left(h_{i-1,out} - \eta_i \Delta h_{is} - h_{f,i}\right) / h_{fg,i}$
i = 2,...,n
For i=[1,I] find m_i^{in}
$m_{in,i} = \frac{Q_{in,i}}{h_g(T_{sat,i}) - h_{f,i}(T_{sat,i})}$
Yes
Find the total mass passing past
each header: $M_i = \sum_{j=i}^{n} m_j^{out} - m_j^{in}$
No
Is $(\sqrt{\sum_{i=1}^{Z} (m_i - m_{i,new})^2} \leq \varepsilon$
Correct the efficiencies:
$\eta_{is,i} = \frac{1}{B_i}\left(1 - \frac{A_i}{\Delta h_{is,i} M_i^{max}}\right)$
Correct h_i and m_i^{out} i=[1,I]
$m_i^{out} = \frac{Q_i}{h_i - h_{f,i}} + m_{p.i}$

Figure 6.7 The modified algorithm for targeting a utility system

Case Study

A utility system has four temperature levels, very high pressure (VHP), high pressure (HP), medium pressure (MP) and low pressure (LP) (Figure 6.8). The steam is produced at 500 °C (932 °F) and the pressure values are 90 bar (T_{sat}=578.0), 46 bar (T_{sat}=497.8), 15.5 bar (T_{sat}=391.7) and 2.7 bar (T_{sat}=266.0). The process is discharging heat of 10.63

MW at the high pressure main and is in need for heat of 6.88 MW and 16.25 MW at the middle pressure and low pressure mains, respectively.

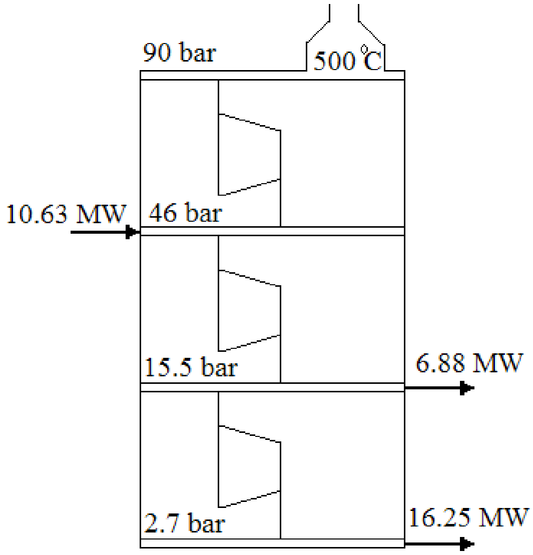

Figure 6.8 Example on targeting a utility system

The targeting procedure will determine the mass flows needed to be produced at the boiler and distributed to the steam mains along with the power produced at each turbine in a way that conforms to the process' heat requirement. This example is adopted from Mavromatis and Kokssis (1998). Table 6.1 shows the result of the proposed algorithm and the one suggested by Mavromatis and Kokossis (1998).

Table 6.1 A Comparison between the Proposed Method and a Previously Reported Method

Expansion Zone	Reported Method			Proposed Method		
	m (t/hr)	η	Power MW	m (t/hr)	η	Power MW
VHP-HP	18.0	0.53	0.52	19.3	0.53	0.53
HP-MP	41.2	0.68	1.98	44.6	0.68	1.78
MP-LP	28.6	0.74	1.87	31.0	0.74	1.75

In order to check the robustness of the two procedures, using the mass flowrates, thermodynamic calculations are performed using the steam tables (Çengel and Boles 1999). The heat induced to the main or delivered to the process, from the steam tables calculations, is then compared to that given in the problem.

Table 6.2 A Comparison Between the Errors in the Proposed Method Compared to the Previously Reported Method

Process Heat (MW)	Reported Method		Proposed Method	
	Heat (MW)	% error	Heat (MW)	% error
10.630	9.741	-8.36	10.630	0.00
6.880	6.294	-8.52	6.844	-0.52
16.250	14.784	-9.02	16.064	-1.14

From Table 6.2, the current algorithm shows a much better accuracy than the reported one. There is no absolute margin of tolerance here, the decision is almost entirely economical; for large amount of heat as is the case in large plants, even a small percentage in excess or deficit will have a significant impact on the process and the operation cost. Moreover, the error tends to increase towards the lower mains. This is due to the recursive nature of the calculations. The highest steam main's estimations are dependent only on the input boiler's condition while lower steam mains depend on the estimated values of the higher steam mains.

In addition to its accuracy, the sequential mathematical nature of the algorithm makes it more flexible with the different operation scenarios. An important aspect that the reported algorithms overlooked is the liquid content at the exit of the turbine. This algorithm can check the quality of the steam and if it is found to fall beyond the tolerated range, the algorithm sends a feedback to the boiler to alter the superheat temperature.

The same algorithm will be used to locate the optimum steam mains' locations, initially in an exhaustive manner. The optimal mains locations were specified as the ones that would result in the highest power produced. The proposed algorithm is coded in Matlab^{TM}, the run time was mainly passed on saturated temperature increment as well as the number of steam mains. For a four steam main utility system with a unit increment in the saturation temperature, the algorithm took a few minutes.

As discussed before, the power produced at each zone increases linearly with the superheat temperature at the inlet of the turbine, hence the superheat temperature was chosen so the heat at each main is satisfied and the quality at the exit of each turbine exit is within the permissible limits.

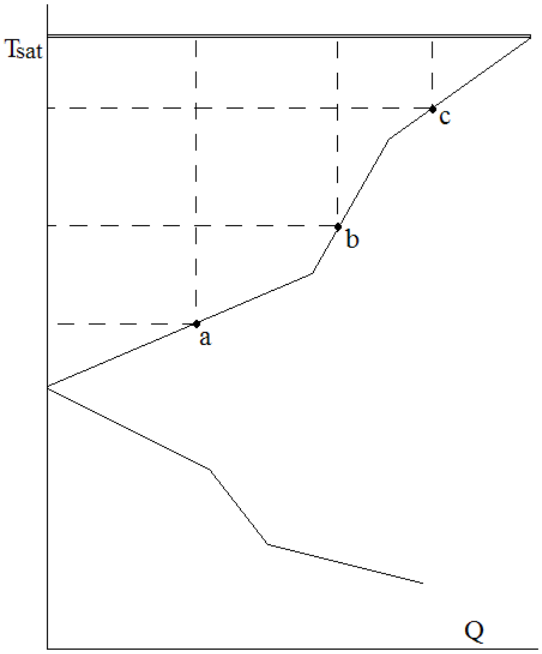

Figure 6.9 Tradeoff between load and pressure ratio

There are many factors that affect the turbine shaftwork; the mass flowrate, the pressure ratio, the isentropic efficiency, the mechanical efficiency and inlet conditions. The complicated nature of the algorithm makes it very hard to determine which of these factors contributes the most to the optimum main selection. On the grand composite curve, the two main factors that make a clear tradeoff are the mass flowrate and the pressure ratio. On Figure 6.9, if the difference between the two steam mains is wide, the heat at the outlet of the turbine tends to get smaller until it reaches zero at the widest possible range i.e. between the highest pressure value and the pinch point. Likewise, when the pressure range is small, the lower main will have a higher duty and hence more load to be delivered through the turbine that will increase the shaft work output. As the steam main is mover up, the mass flow required gets higher until it hits the higher pressure main at which the pressure difference is zero. The tradeoff between the mass flow and the pressure difference can be compared to the area of the rectangle formed by the heat load (width) and the pressure difference (height) as shown on Figure 6.9.

In order to check for any correlation between the area and the optimum power produced, the steam mains' locations at which the area (the hatched rectangles in Figure 6.9) is maximum, are compared to those at which the shaft work produced is maximum. A perfect match between the two sets has been found.

Raissi (1994) stated that the area is approximately linear with the power produced, i.e. the power can be estimated by multiplying the area by a constant, called the conversion factor (cf). This approximation however could predict an erroneous estimation of up to 30% of relative error as reported (Mavromatis and Kokossis 1998).

In order to establish a proper relationship between the area of the rectangle on the GCC and the turbine shaft work, the profile of both have been matched and compared. When the tow profiles are superimposed, their maximum values matches, i.e. the maximum area of the rectangle and the maximum power are both located at the same saturation temperature and the two profiles start diverging gradually away from the maximum point.

Having established this observation, the exhaustive enumeration procedure can now be reduced down from running the iterative algorithm to locating the maximum area made by the steam mains on the grand composite curve. This procedure has been executed for different grand composite curves with a number of steam mains up to seven. The steam mains at which the area is maximum and the power is also maximum match invariably. Moreover, since the calculating the area is much simpler than calculating the power, retrieving the optimum steam mains by using the area search was much faster.

Visualizing the similitude between the area and the power profiles can be obtained in the case of two and three steam mains. Figure 6.10 and Figures 6.11.A to 6.11.C show such profiles for two and three steam mains respectively. In both cases the highest saturation point is taken at the highest pressure level.

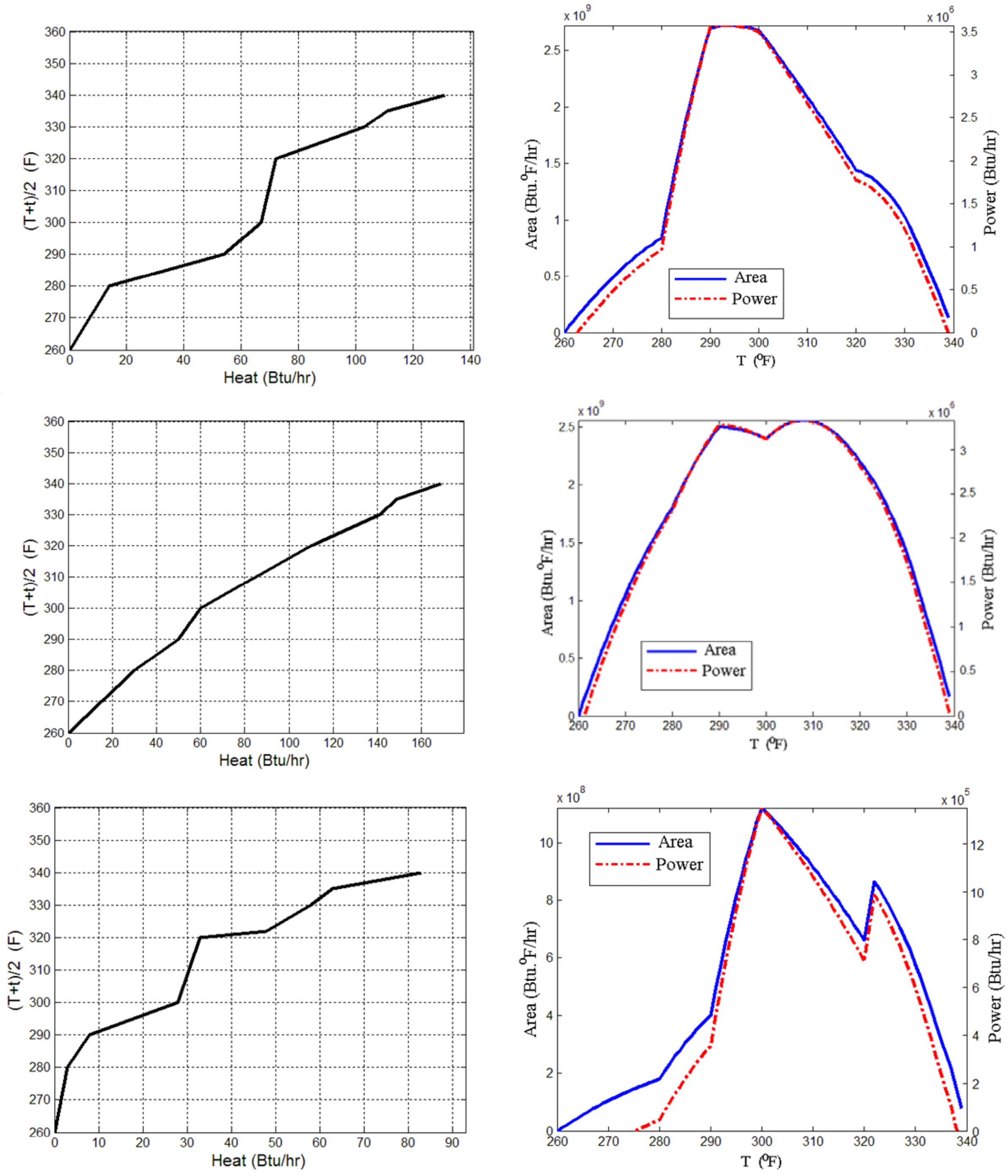

Figure 6.10 The area and power curves for different GCC scenarios

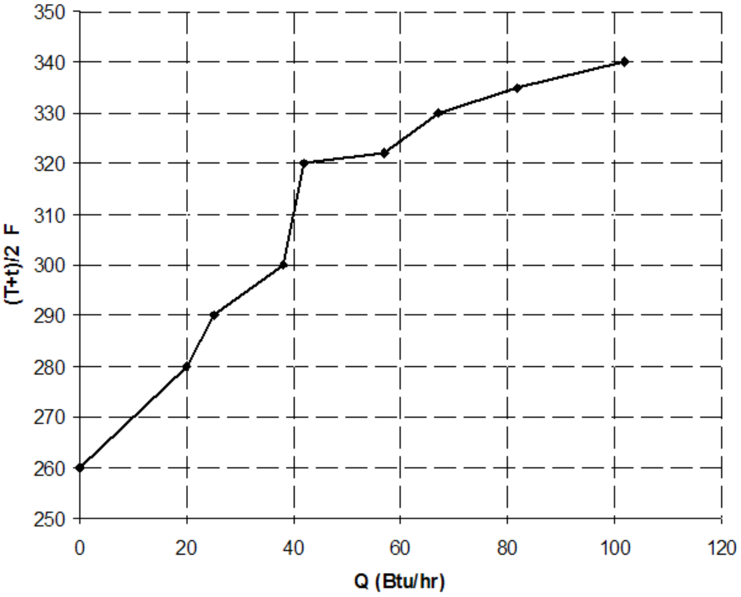

Figure 6.11.A The GCC for a three steam mains scenario

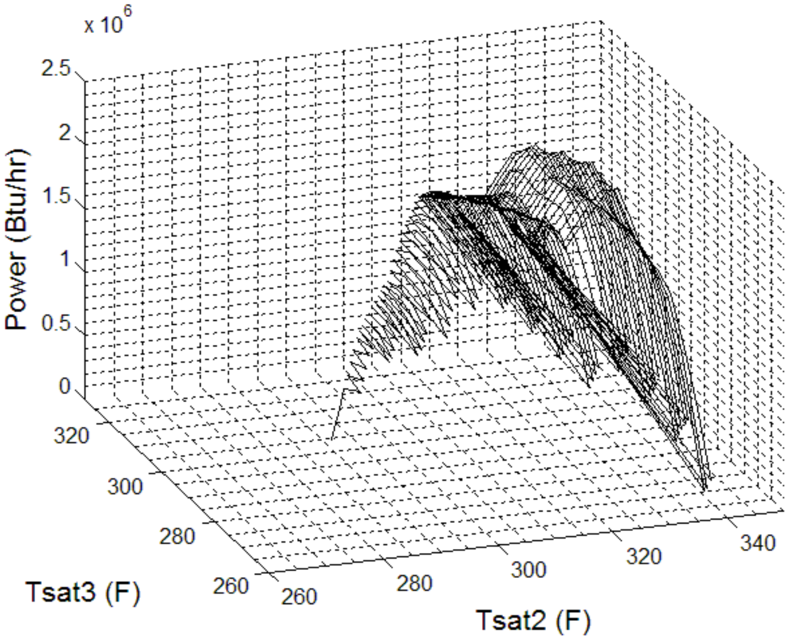

Figure 6.11.B The power plot for a three steam mains scenario

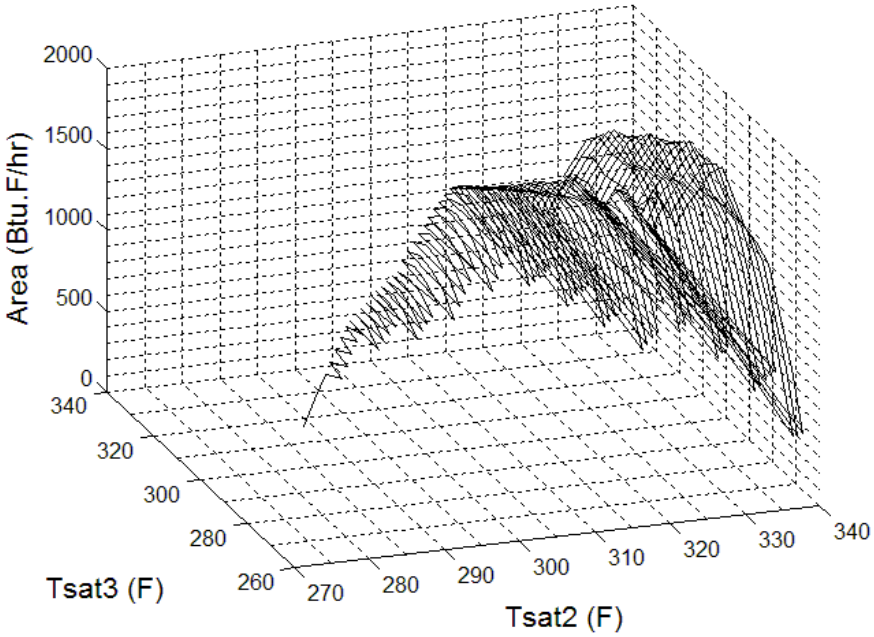

Figure 6.11.C The area plot for a three steam mains scenario

The maximum point in Figure 6.11.B and Figure 6.11.C is located at T_{sat2}=300 and T_{sat3}=280.

Conclusion

Optimizing the location of mains is considered a difficult task due to the interdependency of the many variables involved in the procedure. Establishing some observation has abated the difficulty of the task to optimizing the steam mains' level to a number of turbines in series with a boiler producing steam at sufficiently high temperature. The effect of the high temperature was found to be linear on the turbine's shaft work. Moreover, turbines in series are observed to be more efficient from a thermodynamic view point.

The task of optimizing the steam mains level has also been reduced to only maximizing the area in the grand composite curve which has made the procedure computationally efficient.

CHAPTER VII

OPTIMIZING THE PIPELINE STRUCTURE

Introduction

A very important aspect of chemical plant design is the involvement of the several networks of pipelines and other carriers. In most cases, the design of such networks has several possible layouts; and in most of these cases, there is one layout that is optimum in terms of the cost, safety and maintenance.

The complexity of identifying the optimum layout depends on the design specifications in the first place; the more parameters are involved, the more complex the problem is.

Alandi et al. (2007) developed a procedure that took account both the layout and the pipe size. Their procedure starts by identifying the possible alternatives and retaining the most inexpensive branches. Ito (1999) developed a genetic algorithm approach in planning interactive pipeline routs. The GA method is based on starting with an initial, presumably good, solution and then the algorithm alters that solution until it evolves to a better or best solution. Guirardello and Swaney (2005) developed a mixed-integer linear programming (MILP) model for a 3D problem considering all fitting pipes and fittings cost for the shortest path problem.

In this paper, a combinatorial method is used to find the optimum layout of a pipeline in a chemical plant using a uniform cross-sectional area.

Graph Theory

Graph theory studies the pairwise relationship between nodes connected by arcs in a given network. This field has many applications in real life; it is useful whenever a given problem can be modeled as nodes and arcs, for example, in transportation, graph theory is used to design the shortest routes passing by a number of cities. Graph theory is used here to model a layout of pipeline network representing each pipe segment as an arc and the source and all consumption points as nodes.

The Traveling Salesman Problem

A classical problem in networks design is the traveling salesman problem (TSP). For a given network of nodes interconnected with arcs, it is required to find the shortest route that stops at each node only once and returns back to the first node as shown in Figure 7.1.

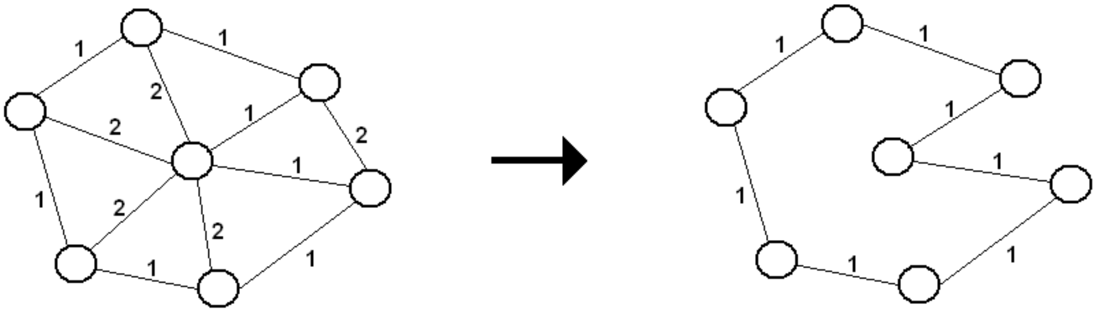

Figure 7.1 The traveling salesman problem

The main objective of the traveling salesman problem is to minimize the distance (the cost) of the connecting route. T. For a number of nodes N with cost of c_{ij} for an arch connecting node (i) to node (j), the traveling salesman problem takes the following formulation (Figure 7.2) (Winston 1994):

$$\text{Minimize } \sum_{\mathrm{i}}^{N}\sum_{\mathrm{j}}^{\mathrm{N}} c_{ij}x_{ij}$$

$$s.t. \ \sum_{i=1}^{N} x_{ij} = 1 \ \ (\mathit{for}\ j = 1,2,...,N)$$

$$\sum_{j=1}^{N} x_{ij} = 1 \ \ (\mathit{for}\ i = 1,2,...,N)$$

$$u_i - u_j + Nx_{ij} \le N - 1 \ (\mathit{for}\, i \ne j; i = 2,3,...,N; j = 2,3,...,N)$$

$$x_{ij} = \{0,1\} \ \ \mathit{for\,all}\, i, j \ \ \mathit{and}\ \ u_j \ge 0 \ \ \mathit{for\,all}\ j$$

Figure 7.2 Formulating the traveling salesman problem

The variable x_{ij} is a binary one, it is true i.e. one if the arc connecting node i to node j is included in the solution of the problem and it is false i.e. zero, otherwise.

The third constraint is required to avoid having all cities connected by more than one route.

This formulation is valid when the number of nodes is relatively small, however, when the number of nodes is large (can be 100 and above) heuristics is more viable to solve the problem.

To ensure the robustness of this method, an example of such a network is solved by exhaustive enumeration and is then solved by this method for comparison.

Figure 7.3 shows a plant where the circulation needs to take place through eight stations with the cost matrix given by Table 7.1.

Table 7.1 The Cost Matrix of the Different Possible Routes

Nodes	1	2	3	4	5	6	7	8
1	0	14	6	17	26	M	M	M
2	14	0	M	10	M	M	24	M
3	6	M	0	10	11	12	M	M
4	17	10	10	0	M	15	19	M
5	26	M	11	M	0	9	M	12
6	M	M	12	15	9	0	11	6
7	M	24	M	19	M	11	0	14
8	M	M	M	M	12	6	14	0

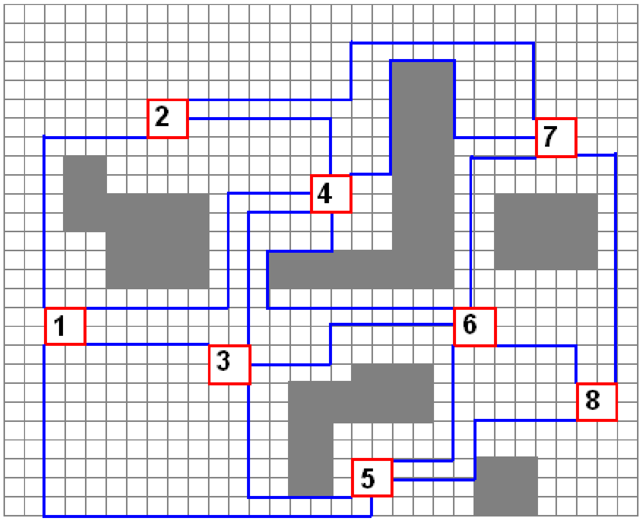

Figure 7.3 Application of traveling salesman problem

The gray areas represent obstacles through which the circulation line cannot pass while the solid lines represent all possible candidates for the circulation route.

With eight nodes, exhaustive enumeration requires testing a number of solutions equivalent to the factorial of eight (8!=40,320). Using MatLab, the best solution was found by following the route: 1-3-5-6-8-7-4-2-1 with an optimum cost of 89 units. Using the above-described formulation with Lingo, the same solution was retrieved, Figure 7.4,

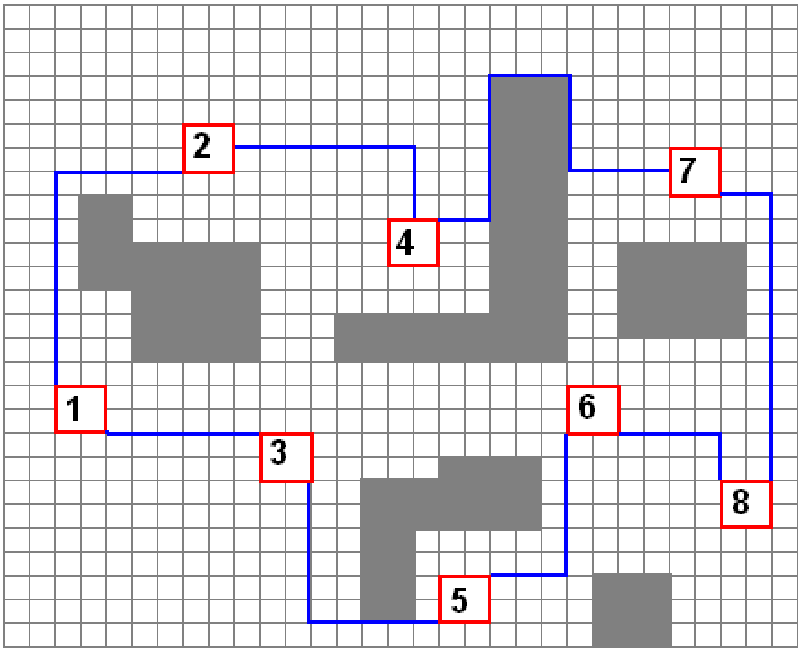

Figure 7.4 The optimum route using the traveling salesman problem

Industrial Applications

A dowtherm is being circulated across a number of heaters and is then recharged with heat in a furnace assuming. The nodes represent the process where heating take place, and each arc represents a tentative line connecting nodes. Fortunately, in most cases, the number of nodes is not very big. For N heat exchangers, the costs are introduced as a matrix of size N×N, with c_{ij}=M when i=j, where M is a very large number and also c_{ij}=M when the arc connecting i and j, is not a realistic solution.

As a case study, Figure 7.5 shows a set of 20 heaters with a set of possible arcs. It is required to find the optimal route. Figure 7.6 shows the set of all possible routes.

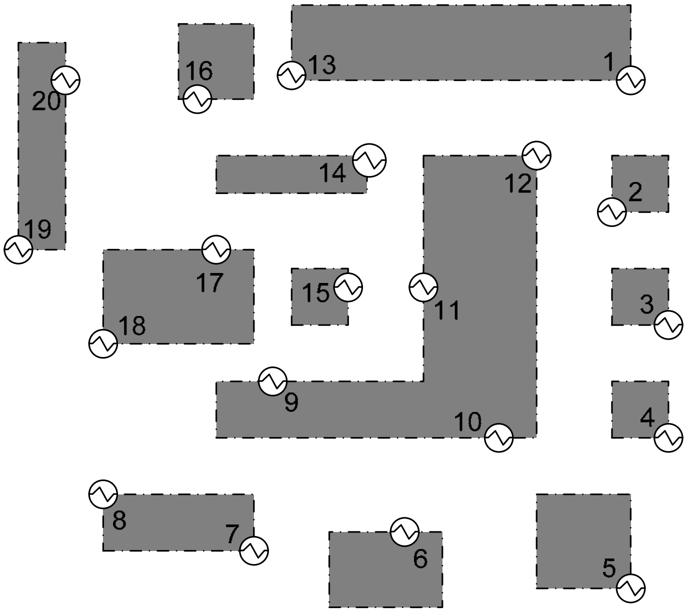

Figure 7.5 Set of 20 heaters

The grey areas are repressing obstacles that pipelines (arcs) cannot overlap.

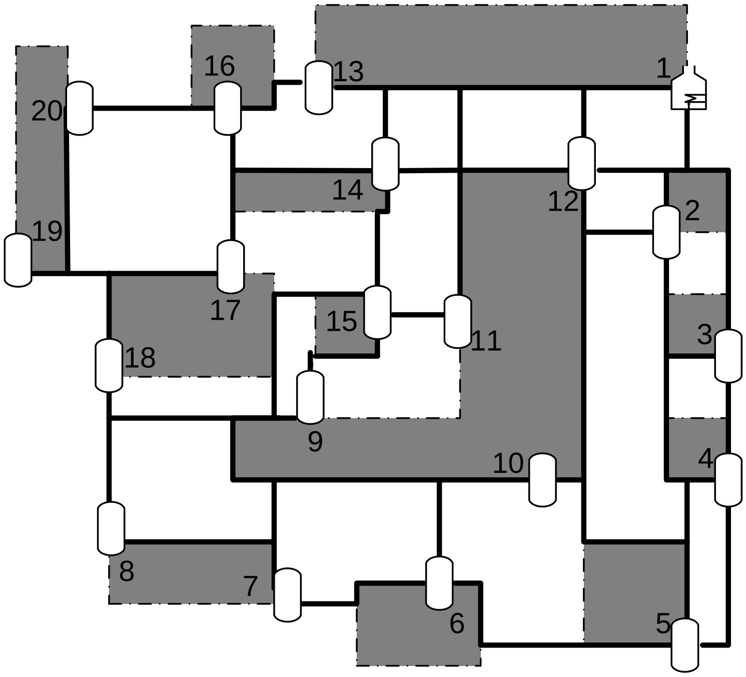

Figure 7.6 Set of all possible routes interconnecting the heat exchangers

The distances between each pair of nodes are given in the cost matrix **C** in Table 7.2 where M indicates an infinitely large number. The above formulation was solved by using Lingo^{TM} 10.0 .

Table 7.2 The Cost Matrix of the Possible Circulation Layout

Nodes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	M	8	15	M	M	M	M	M	M	M	20	9	18	M	M	M	M	M	M	M
2	8	M	9	M	21	M	M	M	M	18	M	7	M	M	M	M	M	M	M	M
3	15	9	M	9	M	21	M	M	M	M	M	M	M	M	M	M	M	M	M	M
4	M	M	9	M	10	M	M	M	M	9	M	M	M	M	M	M	M	M	M	M
5	M	21	M	10	M	15	M	M	M	15	M	28	M	M	M	M	M	M	M	M
6	M	M	21	M	15	M	9	M	M	10	M	M	M	M	M	M	M	M	M	M
7	M	M	M	M	M	9	M	11	14	19	M	M	M	M	M	M	M	M	M	M
8	M	M	M	M	M	M	11	M	15	M	M	M	M	M	M	M	M	8	M	M
9	M	M	M	M	M	M	14	15	M	21	M	M	M	M	9	M	10	11	M	M
10	M	18	M	9	15	10	19	M	21	M	M	M	M	M	M	M	M	M	M	M
11	20	M	M	M	M	M	M	M	M	M	M	12	M	M	4	M	M	M	M	M
12	9	7	M	M	28	M	M	M	M	M	12	M	17	9	M	M	M	M	M	M
13	18	M	M	M	M	M	M	M	M	M	M	17	M	8	M	6	M	M	M	M
14	M	M	M	M	M	M	M	M	M	M	M	9	8	M	8	12	13	M	M	M
15	M	M	M	M	M	M	M	M	9	M	4	M	M	8	M	M	9	M	M	M
16	M	M	M	M	M	M	M	M	M	M	M	M	6	12	M	M	9	M	M	8
17	M	M	M	M	M	M	M	M	10	M	M	M	M	13	9	9	M	11	10	17
18	M	M	M	M	M	M	M	8	11	M	M	M	M	M	M	M	11	M	10	17
19	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	10	10	M	12
20	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	8	17	17	12	M

Using the above mixed integer linear programming; Figure 7.7 shows the optimal layout.

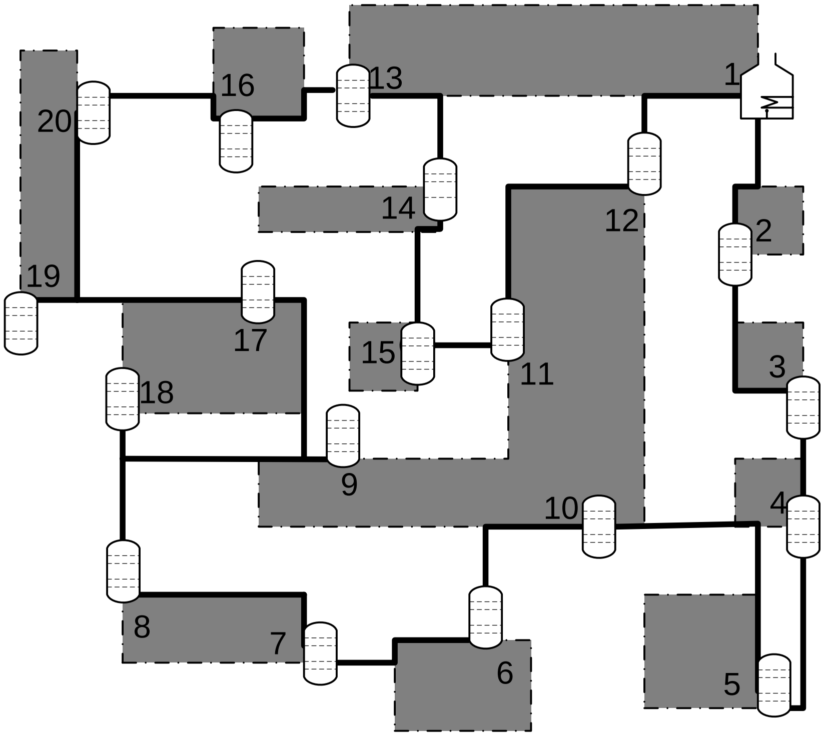

Figure 7.7 The minimum route connecting the 20 heaters

The minimum route passes by all nodes in the order 1-12-11-15-14-13-16-20-19-17-9-18-8-7-6-10-5-4-3-2-1 with a total cost of 187 unit distance.

The Shortest Path Problem through a Set of Nodes

Another very frequent scenario is when the path is expected to start from a node and terminate at another passing by a given set of intermediate nodes. If the problem only requires the shortest path without having to pass by a set of nodes, then Dijkstra algorithm can be used (Winston 1994) and such a problem is usually easy to solve by inspection, but with the above requirement, a different approach is used.

To define the shortest path from node A to node B through a set of N nodes, the problem formulation is shown in the Figure 7.8.

$$\text{Minimize } \sum_{\text{i}}^{N}\sum_{\text{j}}^{\text{N}} c_{ij}x_{ij}$$

$$s.t.\ \sum_{i=1}^{N} x_{ij} = 1 \ \ (\mathit{for}\ \ i = 1,2,...,N;\ i \neq B)$$

$$\sum_{j=1}^{N} x_{ij} = 1 \ \ (\mathit{for}\ j = 1,2,...,N;\ j \neq A)$$

$$u_i - u_j + Nx_{ij} \leq N - 1 \ (\mathit{for}\, i \neq j; i = 2,3,...,N;\ j = 2,3,...,N)$$

$$x_{ij} = \{0,1\}\ \ \mathit{for\, all}\, i, j\ \ \mathit{and}\ \ u_j \geq 0\ \ \mathit{for\, all}\ j$$

Figure 7.8 Formulating the shortest path problem

As an example, the application of the shortest path problem can be applied in the case when the dowtherm circulation needs to be first assigned to a number of heaters with high temperature requirements. The flow will be directly assigned to the high-temperature heaters and then the shortest path problem will be solved between the last of these heaters to the furnace passing by all other low-emperature heaters.

In the above network of 20 heaters, the dowtherm needs first to be assigned to high temperature heaters in the order 2, 12 then 3. The task now is to find the shortest path linking heater 3 to the furnace passing by all other heaters.

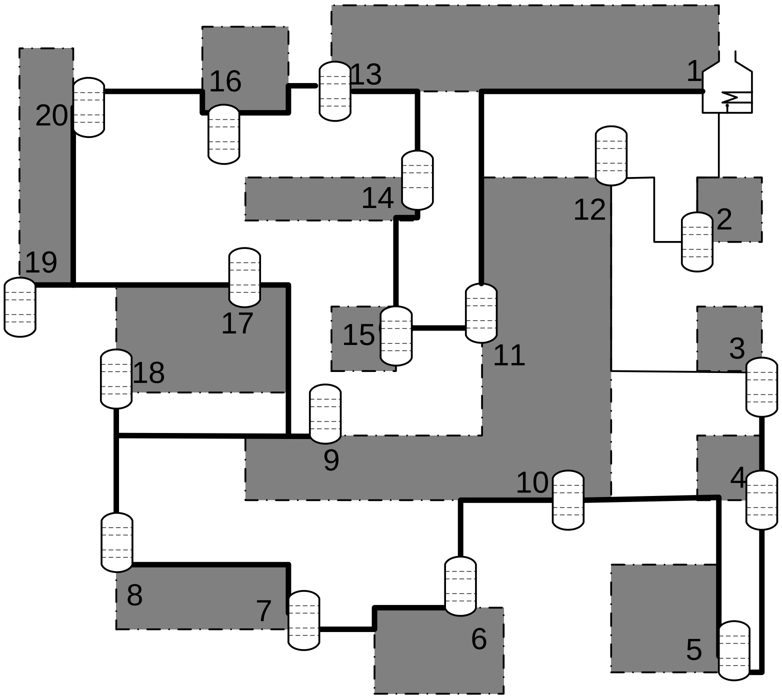

Figure 7.9 The shortest path through a set of heaters

The optimum path (Figure 7.9) will follow the heaters: 3-4-5-10-6-7-8-18-9-17-19-20-16-13-14-15-11-1 with a total cost of 169.

The Minimum Spanning Tree

Another problem that can be solved using graph theory is the situation where the fluid is carried in a branching network to different destinations. Known algorithms for solving this problem are Kruskal's algorithm and Prim's algorithm (Cook et al. 1998). For a given network of N nodes, the minimum spanning tree (MST) is a group of n-1 arcs that connects all nodes of the network without creating a loop, Figure 7.10.

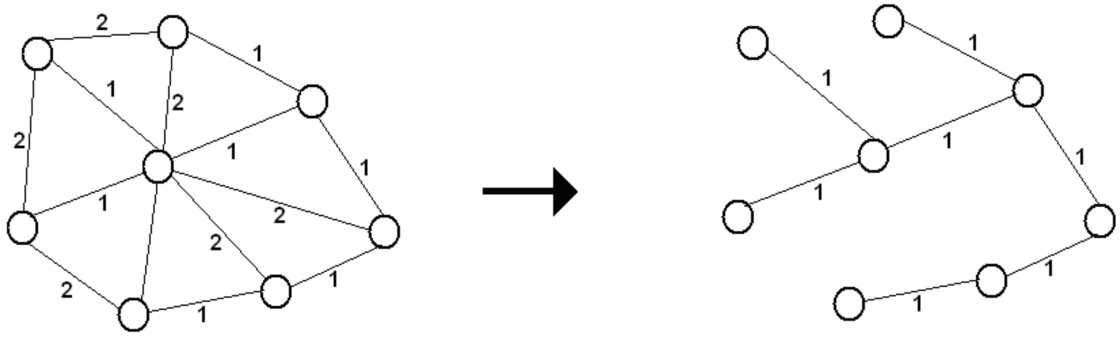

Figure 7.10 The minimum spanning tree

For a network of N nodes the following procedure is followed to determine the minimum spanning tree:

Step 1: Start from any node in the network, say node i, investigate all arcs connecting node i with the rest of the network and choose the arc with the lowest cost, call it arc ij connecting node i to node j. By now, a subset tree composed of two nodes, namely i and j has been built.

Step 2: Investigate the arcs connecting the existing tree with the rest of the nodes and select the arc with the lowest cost to be added to the existing tree. If the addition of that arc will create a loop, then move to the second lowest and so forth.

Step 3: Expand the existing tree by repeating step 2 until no more arcs can be added without creating a loop. At that stage, the minimum spanning tree has been attained.

By applying this procedure for the set of 20 consumption stations given in the previous example, the optimal solution shown in Figure 7.11 was obtained, with a total cost of 161 unit distance.

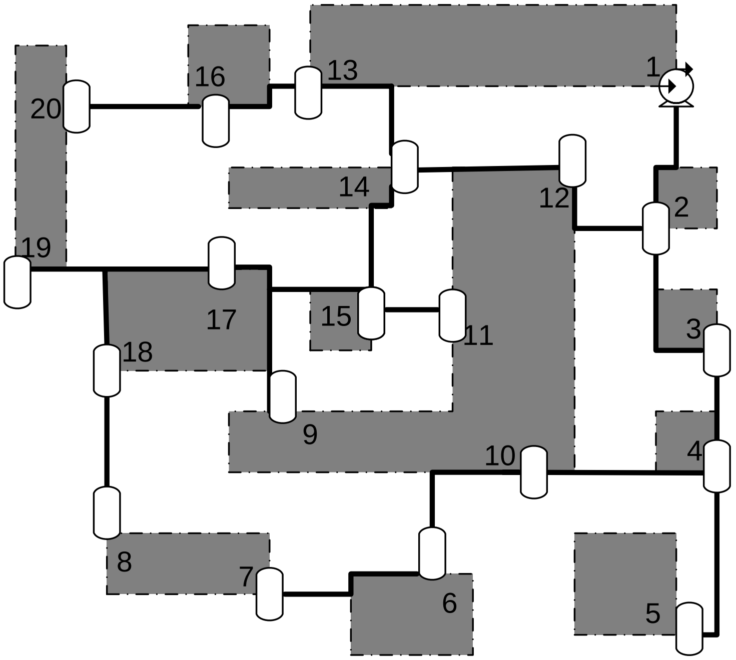

Figure 7.11 The minimum spanning tree for a set of 20 heat exchangers

The proof to the validity of this method is self-evident. The most inexpensive arcs are picked up in order, unless an inexpensive arc will create a cycle, until the spanning tree is complete.

Conclusion

To conclude, the use of combinatorial procedures in graph theory had proved very useful in the optimization of the pipeline network routing. Although these methods are limited in many other areas, the number of nodes in a chemical plant usually small enough that these methods are practical.

CHAPTER VIII

HYBRID GLOBAL OPTIMIZATION ALGORITHM

Introduction

Interval arithmetic is a useful bounding technique that was initially invented to control bounding computing error (Boche 1963; Moore 1962; Moore 1965; Moore 1966). It is extensively used as a mean to control rounding errors (Kreinovich 2007; Moore 1966; Neumaier 1990) but their strength has also been exploited in controlling nonlinearities in an algorithmic manner (Neumaier 1990). Interval analysis can also be utilized to achieve the global solutions of the general MINLP's (Vaidyanathan and El-Halwagi 1996).

Figure 8.1 shows a general procedure of how interval analysis is utilized in optimization. With its ability to control nonlinearity and nonconvexity, interval analysis has become a practical tool in global optimization (Byrne and Bogle 1995; Byrne and Bogle 1996; Vaidyanathan and El-Halwagi 1994; Vaidyanathan and El-Halwagi 1996).

In its simplest form, interval analysis standalone converges very slowly to the global solution. That fact necessitated reasonably bounded domain that included the global solution (Ratschek and Rokne 1991).

Newly developed techniques in global optimization, using interval analysis, focus on how to accelerate convergence using an auxiliary accelerator or a solver (Figure 8.1). Two important devices for accelerating convergence are: first for a function that is twice continuously differentiable, with the Hessian matrix inclusion exists, the boxes that contain local minimizers could be selected (Ratschek and Rokne 1991). The second important device is the monotonocity test which is conducted with the use of the inclusion of the gradient of the objective function. In addition to these two tests, Vaidyanathan and El-Halwagi (1994; 1996) used the upper infeasibility test and the lower bound test along with the upper bound test as will be described in more details later on.

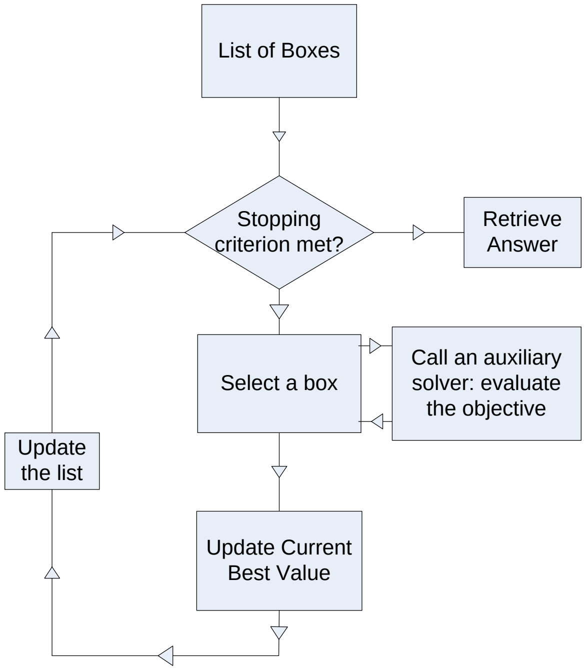

Figure 8.1 Global optimization using interval analysis

With its good performance yet non-guaranteed optimality, simulated annealing can also be invested as a tool in speeding up convergence. This method is a metaheuristic tool used to find near optimal solutions for an optimization problem. Simulated annealing surpasses other deterministic optima locators in that it tries to avoid being confined to local optima. This is achieved by relocating the current solution not just when a better one is found; but may also take place if an inferior solution is found.

However, the latter decision of relocation is dependent on a probability function that is dependent on the difference between the function value of the current iteration and the retained best value as will be described herein.

The advent of this technique is attributed to Kirkpatrick et al. (1983) from an earlier work by Metropolis et al. (1953). This technique is an imitation of another used in quantum mechanics. When the substance is at high temperature, it will be in a state of chaos or disorder and when it is gradually cooled down (annealed), it tends to reorganize itself in a crystalline structure at which the energy of the substance is at its lowest ranks (Laarhoven and Aarts 1988).

Despite its good performance, using simulated annealing requires problem-specific decisions which effect the quality of the function and the execution time (Boche 1963). These decisions are on the selection of the annealing (cooling) rate c, the number of iterations per temperature k and, the maximum and the minimum temperatures. Having too high or too low for any of these parameters might be very detrimental to either the final solution or the execution time.

The linear cooling schedule is the mostly used. However, logarithmic cooling schedule can also be found in the literature (SZU and Hartley 1987).

For the number of iterations per temperature, Lundy and Mees (1986) suggested using one iteration per temperature but to use a very slow cooling rate .i.e. increasing the number of temperatures tried. Connolly (1990) suggested using the opposite, that is one temperature with a number of iterations, yet the difficulty will still be which temperature to select.

Dowsland (1995) suggested starting with a very high temperature and then cooling it rapidly when worse solutions are accepted at a certain probability. Rayward-Smith et al. (1996) suggested this probability at 0.6.

The most critical decision when applying simulated annealing to continuous problems is defining the neighborhood criterion (Goldstein and Waterman 1988), which is usually a swap of two elements in the case of combinatorial problems. A normal selection of this function would be by slightly displacing the current point in one direction (Goffe et al. 1994). Another approach is to make the displacement in all directions (Bohachevsky et al. 1986). A more complicated way to determine the neighboring point is by adding the current one to a vector which is a product of a normally distributed random numbers and an n×n matrix that controls the step size distribution (Vanderbilt and Louie 1984). This method however can be computationally exhaustive when solving a high dimensionality problem.

Simulated annealing has reputedly proved to be a very efficient tool in global optimization (Cardoso et al. 1997; Choi et al. 1999). When simulated annealing is used, the search procedure has to be confined within the feasible domain and then proceeds by updating the objective function value using other auxiliary criteria.

In this work, simulated annealing is incorporated with interval analysis to speed up convergence by a faster upper bound update as well as detecting superior infeasible values of the objective functions and hence deleting them from the original search domain.

Despite its simple algorithm, simulated annealing can retrieve the global optimum regardless of the complexity of the topography of the given function. Nevertheless, the algorithm parameters e.g. temperature schedule, maximum number of iterations and initial points, need to be very well-selected for a robust performance of the algorithm.

Interval Analysis

Vaidyanathan and El-Halwagi (1996) suggested an algorithm to solve a general nonconvex MINLP. For a given problem with minimum and maximum bounds on each of the ***n*** variables, the global solution is sought. The essence of that algorithm is to handle the box defined by the ***n*** intervals in the n-dimensional space by deleting the suboptimal. Their algorithm is based on an interval analysis method which depends on evaluating the inclusions of the objective functions, its first and second derivatives, and those of the constraints. Since the initial space can be very large and in absence of any criterion to schedule the next search, this procedure can be very time-consuming especially if a large number of local minima exists.

In order to speed up convergence to global solution and in order to escape local minima, a heuristic search method, namely Simulated Annealing (SA) is incorporated in the Interval Analysis Algorithm. Simulated annealing, if implemented properly, is very beneficial in escaping local minima as well as speeding up upper bound update. Moreover, instead of checking sub-boxes currently enlisted in the feasible space, whenever the upper bound is updated, the whole list is rechecked and non-promising sub-boxes are removed immediately without conducting a separate iteration for each sub-box.

Before describing the heuristics to be incorporated with this method, a general description of interval analysis is presented. This method is intended for the general mixed-integer nonlinear problem with a given bounds for each of the ***n*** variable in the n-dimensional space, problem (8.1).

$$\begin{aligned}
&\min f(x,y) && && (8.1.a)\\
&s.t. && && \\
&p_i(x,y) \le 0 \quad i = 1,2,...,m && && (8.1.b)\\
&a_i \le x_i \le b_i \quad i = 1,2,...,k && x \in R^n && (8.1.c)\\
&c_i \le y_i \le d_i \quad i = 1,2,...,n-k && y \in I^{n-k} && (8.1.d)
\end{aligned}$$

A very essential tool in this algorithm is the use of interval arithmetic. An interval (inequalities 8.1.c and 8.1.d) is expressed as X_i=[a_i,b_i] and will be denoted by a capital letter where a and b, a≤b, are scalars bounding the range of the variable x, denoted by small letter. Interval analysis aims at determining the bounds of a given function using the given intervals of each of its variables. The interval bounding the given function is called the function inclusion. This inclusion is determined by some intuitive internal operators such us the ones used for the four basic operations.

For the two intervals X=[a,b] and Y=[c,d], the four basic operations are performed as follows:

Addition

$$X+Y=[a,b]+[c,d]=[a+c,b+d] \qquad (8.2.a)$$

Negation

$$X-Y=X+(-Y)=[a,b]+[-d,-c]=[a-d,b-c] \tag{8.2.b}$$

Multiplication

$$X*Y=[a,b]*[c,d]=[\min(ac,ad,bc,bd),\max(ac,ad,bc,bd)] \tag{8.2.c}$$

Division

$$X/Y=[a,b]/[c,d]=[\min(a/c,a/d,b/c,b/d),\max(a/c,a/d,b/c,b/d)] \tag{8.2.d}$$

$0\notin$(Bohachevsky et al. 1986) if **c** is zero then $a/c = \infty$ if **a** is positive and $-\infty$ if **a** is negative.

These operations are necessary in the evaluation of the function inclusion. The inclusion of a function for a given interval is an interval that bounds all possible values of that function. There are many methods for evaluating the inclusion of a function; their goodness and complexity depend on how accurate the inclusion is, as some methods like natural inclusion may sometime give an interval which is wider than the actual one. However it is the easiest and also is the most exact in many cases as described in the following section.

An inclusion of a function is said to be minimal if the function interval is the smallest possible for some given variables' intervals and is said to be convergent if narrowing the variables' intervals yields a better inclusion in terms of bounding the function's values.

Inclusion Functions

There is no single method that gives a guaranteed and exact inclusion for all functions. All methods depend on three main things: the topography of the function, the width of the variables' intervals and the format in which the function is expressed.

Natural Inclusion

This inclusion is evaluated by replacing each real variable x_i in the function by its interval and each operator by its interval counterpart (Jaulin et al. 2001; Ratschek and Rokne 1988). This method is the one described earlier for the four arithmetic operations.

This method is not always minimal, however minimality can be guaranteed for a continuous function in which, each variable appears only once. Note however, that the same function may take several forms and their inclusions may also vary accordingly.

Centered Inclusion

This method is suitable for small intervals. In this method, the function is expressed using the mean value theorem. For an interval [x] with m being the midpoint of that interval: $F(x)=f(m)+g^T(x)\ (x-m)$

where g is the gradient of f with respect to x.

Other methods are the Mixed Centered Inclusion and Taylor Inclusion which are based on the same concept of re-expressing the function using Taylor series.

For the purpose herein, any inclusion will serve the need except that a minimal inclusion will help speed up the convergence of the solution. However, as noted before no method will guarantee minimality except the natural inclusion for a continuous function in which the variable appears only once.

Optimality Tests

There are several optimizations methods based on interval analysis; this one, however, is based on conducting four tests, if any sub-box of the feasible space fails one or more of these tests, that sub-box is removed from the space.

The Upper-Bound Test

Before conducting this test; an upper bound of the global minimum needs to be established. Initially, the upper bound, denoted here by *upbd*, is the objective function value of any feasible point. Each time a better value is found, the upper bound is updated to the new value. The updating process will continue until convergence is attained.

Consider a given box defined by the intervals $\{X_1,X_2,..X_n\}$ has its function inclusion defined by the interval $F(\{X_1,X_2,..X_n\})=[LBX,UPX]$. If

$$LBX>upbd$$

then, that box can be deleted for its known that the minimum value of the objective function in that box is no better than the upper bound.

The Infeasibility Test

For each of the constraints p_i (x) i=1,..,m and a box defined by the intervals $\{X_1,X_2,..X_n\}$, P_i $(\{X_1,X_2,..X_n\})=[LP_i,UP_i]$ is the inclusion of that constraint for the given box. If

$$LP_i>0 \text{ for some } i=1, 2\ldots m$$

then, the box can be deleted for its known that the constraint p_i can never be satisfied by any point in that box. Of course, before applying this tests, all constraints must be presented in the standard form where they have to be less than or equal to zero.

The Monotonicity Test

Let $g_i(x)$ denote the gradient of the objective function with respect to the variable x_i then for a given box defined by the intervals $\{X_1,X_2,..X_n\}$, G_i $(\{X_1,X_2,..X_n\})=[LG_i,UG_i]$ is the inclusion of that gradient function for the given box. If

$0\notin[LG_i,UG_i]$ for all variables of a strictly feasible box then that box can be deleted except for its end points.

Note that this test necessitates strict optimality and a box that survive the deletion at the infeasibility test will only be tested if the upper bound of all constraints' inclusions is less than or equals to zero. i.e. $UP_i\leq 0$ for all i=1,2,…m.

The Nonconvexity Test

Let $h_i(x)$ denote the Hessian of the objective function with respect to the variable x_i then for a given box defined by the intervals $\{X_1,X_2,..X_n\}$, H_i $(\{X_1,X_2,..X_n\})=[LH_i,UH_i]$ is the inclusion of the diagonal element of Hessian, i.e. the second derivative of the objective function with respect to x_i for the given box.

If $UH_i < 0$ for any of the variables of a strictly feasible box, then the interior of that interval is deleted because the function has to be positive semi-define over the entire box. Note that this test also requires strict optimality of a box.

Interval Analysis Algorithm

In the algorithm by Vaidyanathan and El-Halwagi (1996)(Vaidyanathan and El-Halwagi 1996), the stopping criterion is reached when a box of a very small width is left at the end of the procedure. Initially, an upper bound and a lower bound is determined. Then the algorithm proceeds by selecting the box with the maximum width for investigation. The four tests are then conducted on the box. If any of the tests are failed, the box will be deleted otherwise the local optima of the unconstrained function over the box is evaluated. If that point is feasible, the box is split across the widest direction. However, if the box is infeasible at that point, the distrust region method will then be applied. The distrust region is to extend the infeasible point to an infeasible inner box by performing the following nested optimization problem:

$$Max\ \sigma \qquad (8.3.a)$$

$$s.t{:}\ \ P_i([\underline{x}-\sigma I,\ \underline{x}+\sigma I]) > 0\ \ for\ some\ i=1,2,...,m \qquad (8.3.b)$$

$$\sigma \geq 0 \qquad (8.3.c)$$

The optimum point can even be obtained by gradually increasing σ until the constraint is satisfied. The expanded box from the infeasible point needs to be deleted. The deletion procedure will be described in a later section.

Simulated Annealing (SA)

Before describing the modified algorithm, a brief account of simulated annealing is presented here. Simulated annealing is a metaheuristic tool used to find near optimal solutions for an optimization problem. Simulated annealing surpasses other deterministic optima locators in that it tries to avoid being confined to local optima. This is achieved by relocating the current solution not just when a better one is found; but may also take place if an inferior solution is found. However, the latter decision of relocation is dependent on a random probability function that is dependent on the difference between the function value of the current iteration and the retained best value as will be described herein.

The advent of this technique is attributed to Kirkpatrick et al. (1983). This technique is an imitation of another technique used in quantum mechanics. When the substance is at high temperature, it will be in a state of chaos or disorder and when it is gradually cooled down (annealed), it tends to reorganize itself in a crystalline structure at which the energy of the substance is at its lowest ranks (Laarhoven and Aarts 1988).

Simulated Annealing in Optimization

The concept of annealing in quantum mechanics can be exploited in optimization problems; however the design of the parameters is problem specific which is a feature that makes this technique an art more than a science.

Figure 8.2 shows the simulated annealing algorithm. The algorithm starts with a random or a predetermined point v_c in the feasible space S. This point is initially designated as the current point and its corresponding function value is designated as the current function value f_c. An initial temperature which is relatively high is also determined. This temperature will be decreased depending on an iteration index k initially set as 1.

For each temperature value, a number of iterations, k_{max}, is executed. A point in the feasible space in the neighborhood of the current point is selected and its corresponding function value is calculated. If the new value is better (less in the case of the minimization problem described in the algorithm of Figure 8.2), then it will replace the current value. If the new value is worse than the current then it has a chance to replace the current one if the exponential value of the difference between the two function values divided by the temperature i.e. $\exp((f_c\text{-}f_n)/T)$ is greater than a random between zero and unity. This value is the probability of accepting an inferior solution over a better one.

At each temperature value, a number of iterations $\mathrm{k}_{\mathrm{max}}$ will be tried before the temperature is slightly decreased. The nested loop of iterations will be repeated for each temperature value until the lowest temperature is reached.

```
Initialize
        $t = 0$
        initial temperature $T_0$
        select a random $v_c$
        calculate $f(v_c)$
while $T > T_{\min}$
        {$k = 1$
        while $k \le k_{\max}$
                {randomly select $v_n \in N(v_k)$
                calculate $f(v_k)$
                if $f(v_k) \le f(v_c)$
                        $f(v_c) = f(v_k)$
                        $v_c = v_k$
                elseif $random(0,1] < \exp((f(v_c) - f(v_k))/T)$
                        $f(v_c) = f(v_k)$
                        $v_c = v_k$
                end
                $k = k + 1$}
        $t = t + 1$
        $T = T(t)$}
Return $v_c, f(v_k)$
```

Figure 8.2 Simulated annealing algorithm

Notice that the difference between the function values is negative in this case and so the exponential function or the probability must be a positive fraction. We can notice that when the temperature is high, the probability of accepting an inferior objective value is higher than the case when the temperature is low i.e. at late stages of the runtime. This means that initially the algorithm tends to be less selective and it would accept significantly inferior values. This trick will allow the algorithm to survey the solution space at a wider range in the initial stages, as the temperature is decreased; the algorithm gets more selective in accepting inferior values of the objective function until the temperature reaches almost zero, then inferior values will be accepted at an extremely low probability.

To allow the algorithm to locate a better value of the objective function, it is advised to decrease the temperature at a very low rate.

It can be proved that simulated annealing algorithm converges to the global optima (Laarhoven and Aarts, 1988), however achieving the global optima or reasonable near global-optima solution is dependent on the size of the problem as well as on the budget available i.e. time and cost of execution. For that matter, the intuitive practice of storing the best found objective value may not be commonly seen in simulated annealing, yet it is advisable if the problem size is small or if it is within the capacity of the available budget.

In addition to its simplicity, simulated annealing has some problem-specific parameters whose setting is left at the discretion of the programmer. Of these parameters is the initial temperature which ought to be high. The initial high temperature will prevent the algorithm from being prematurely entrapped in a local minima (Eglese 1990). The magnitude of this high temperature is, however, dependent on the available budget and the nature of the problem. Another issue is the rate at which the temperature is decreased or the so called "*Temperature Scheduling*." There are different versions of this scheduling. Very classical versions are $T=c\ T_0$ whereas c is usually between 0.8 and 0.99, and $T=T_0/\log(t+1)$ where T_o is the initial high temperature and t is an index that indicates the temperature change.

Stopping criterion of each temperature and halting criterion of the whole algorithm are also left at the programmer’s discretion.

Simulated Annealing in Interval Analysis

In this work, simulated annealing is incorporated with interval analysis to speed up convergence by a faster upper bound update as well as detecting superior infeasible values of the objective functions and hence deleting them from the original search domain.

Despite its simple algorithm, simulated annealing can retrieve the global optimum regardless of the complexity of the topography of the given function. Nevertheless, the algorithm parameters e.g. temperature schedule, maximum number of iterations and initial points, need to be very well-selected for a robust performance of the algorithm.

Defining the Neighborhood

An additional complexity arises when applying this algorithm to a continuous function is the definition of the neighborhood and the determination of the initial point. This parameter should be chosen while considering the nature of the problem in hand as well as the mechanism of the algorithm.

In combinatorial problems, the neighborhood can be defined as loosely as any swap between any two vertices. The maximum number of iterations can then be determined based on the size of the graph being handled and also the annealing rate. For a static annealing rate of the form $T_i=cT_{i-1}$, the number of the different temperatures will be

$$n = \frac{\log(T_{\min}/T_{\max})}{\log(c)} + 1 \tag{8.4}$$

and the total number of iterations will be

$$N_{total} = n.N_{perTemperature} \tag{8.5}$$

This number, in the case of combinatorial problems, can be set as a reasonable portion of the total number of possible swaps and the hope is set on the randomness that will lead to a new global optimum without exhaustively enumerating all possible swaps.

Unlike the case of combinatorial problems, with continuous variables, the number of possible searches can be unlimited. Moreover, the definition of the neighborhood has to be close enough to the current point so as not to miss out better solutions by making wider leaps. The following function is a reasonable definition of a neighbor in the continuous space.

$$x_{i+1} = \left(x_i + \varepsilon\ u_{perm}\right) \quad u_{perm} = (0,0..0,\pm 1,0..0) \tag{8.6}$$

whereas u_{perm} is a random unit vector along one of the ***n*** axes in the n-dimensional space and ε is a very small number greater than zero. This function suggests that the next point is a slight displacement of the current point in one of the directions. Therefore, for a large feasible space, the selection of the initial point plays a very important role in retrieving the optimum point. If the initial point is far away from where the optimum is located, the total number of iterations might be exhausted before even getting closer to the optimum. This problem, however, cannot be cured by simply increasing the maximum number of iterations and increasing the annealing rate since the space can still be large enough to miss out the global optimum in the absence of a hint that presumes the neighborhood of the global optimum. In addition to the technical difficulty, a computationally robust algorithm should as well be sought. Simulated annealing, after all, is a way to escape exhaustive enumerations and other time-consuming search techniques.

An attractive preclusion from using an exhaustively large number of iterations and also from diverging from the global optimum is to use a dynamic neighborhood function.

By decreasing the range of the leap as the temperature decreases, the search process will be allowed to perform an overall survey of promising points in the early stages of the algorithm. When the algorithm gets more selective in later stages, the width of the leap is minimal, hoping that the search process will be confined to the vicinity of the global optimum.

The suggested neighborhood function is a function of a dynamically decreasing ratio multiplied by a scalar which is a function of the average side of the box being searched. For a given box to be searched, defined by the intervals {[x_{L1}, x_{R1}], [x_{L2}, x_{R2}],.., [x_{Ln}, x_{Rn}]} This function is:

$$\begin{aligned} x_{i+1} &= (x_i + u_{perm}(\beta S)\, q^p) \qquad (8.7) \\ u_{perm} &= (0,0...0,\pm 1,0...0) \quad 0 < \beta < 0.5 \end{aligned}$$

S is the average side of the box being searched; it can be calculated as the average of the *n* sides of the box:

$$S = \frac{\sum_{i=1}^{n} d_i}{n} \qquad (8.8.a)$$
$$d_i = x_{Ri} - x_{Li}$$

whereas d_i is the side length of the box in direction *i*, or in other words, the width of the interval of the variable *i*.β is a fractional number and so βS is the maximum allowable leap which will take place when the temperature is at the initial maximum value.

The choice of β depends on the size of the problem in hand. When tested on different problems of different initial boxes, the optimal b was found most of the time between 0.001 and 0.05. For the same number of iterations used, small values of β is best for small initial boxes. When large values of β is used with small boxes, the runtime tends to be longer since the leap will then be wider and more likely to jump outside the box and hence the neighboring point will be rejected and another random neighbor is selected. Moreover, if β is too small for a large box, the number of iterations will be exhausted before survey the whole space and hence the final result will be too far from optimal.

Also, q is a fraction and p is zero at $T=T_{max}$ and increases by unity as T decreases. "q" is set in such away that allows a leap equivalent to βS at the maximum temperature and a leap that is within the required precision (δ) at the minimum temperature. This value of q is calculated by the following equations:

$$n = \frac{\log(T_{\min} / T_{\max})}{\log(c)} + 1 \qquad (8.8.b)$$

whereas c is the annealing ratio which is about 0.80 to 0.95.

The value “n” is basically the number of decrements needed to reach the minimum temperature and “q” is then calculated from:

$$q=\left(\frac{\delta}{\beta S}\right)^{1/n} \tag{8.8.c}$$

With this function, instead of feeding an initial point to start with, a box that defines the search domain needs to be provided to the algorithm. Moreover, the initial point will always be the center point of the box.

$$x_o=(\frac{x_{L1}+x_{R1}}{2},\frac{x_{L2}+x_{R2}}{2},...,\frac{x_{Ln}+x_{Rn}}{2}) \tag{8.9}$$

Figure 8.3 and Figure 8.4 show the modified simulated annealing algorithm for a general unconstrained problem with an initial search domain defined by a box.

This algorithm is checked on different problems with different variables and initial box size. The simulated annealing algorithm used had $T_{maximum}$=1 and $T_{minimum}$=10^{-8} with a static cooling schedule T_{i+1}=0.8 T_i and a maximum number of iterations of 300 per temperature. β is selected to be between 0.001 and 0.05.

In interval analysis with the continuing splitting of the initial box into small boxes, it might be computationally practical to set up the number of iterations relative to the volume of the sub-box being searched. This will allow a robust utilization of the computation time; the algorithm will then give more time to larger spaces to be searched and avoid spending superfluous number of iterations on smaller spaces.

```
INPUT:
        $box = [x_{L1}\ \ x_{R1}, ..., x_{Ln}\ \ x_{Rn}]$   : The space to be searched
        $f(x_1, ..., x_n)$                    :A function to be optimized
        $T_{\min}\ , T_{\max}\ , K, c, \delta, \beta$          : Algorithm parameters; temperatures,
                                              max iterations per temperature (K)
                                              cooling rate (c) final leap ($\delta$) and
                                              the fraction of the initial leap ($\beta$)

INITIATIOIN:
        $x_0 = (\frac{x_{L1}+x_{R1}}{2}, \frac{x_{L2}+x_{R2}}{2}, ..., \frac{x_{Ln}+x_{Rn}}{2})$
    $S = \frac{\sum_{i=1}^{n} x_{Ri} - x_{Li}}{n}$ ,  $n = \frac{\log(T_{\min}/T_{\max})}{\log(c)} + 1$,  $q = \left(\frac{\delta}{\beta S}\right)^{1/n}$
$f(x_c) = f(x_o)$ ,  $p = 1$ , $T = T_{\min}$
SEARCH:
$while\ T \le T_{\min}$
        $\{k = 1$
        $while\ \ k \le k_{\max}$
                $x_k = \left(x_c + u_{perm}\ (\beta S)\ q^p\right)$
                $calculate\ f(x_k)$
                $if\ \ f(x_k) \le f(x_c)$
                        $f(x_c) = f(x_k)$
                        $x_c = x_k$
                $elseif\ random(0,1] < \exp((f(x_c) - f(x_k))/T)$
                        $f(x_c) = f(x_k)$
                        $x_c = x_k$
                $end$
                $k = k + 1\}$
        $p = p + 1$
        $T\ = c.T\ \}$
Return $x_c, f(x_c)$
```

Figure 8.3 The modified simulated annealing algorithm

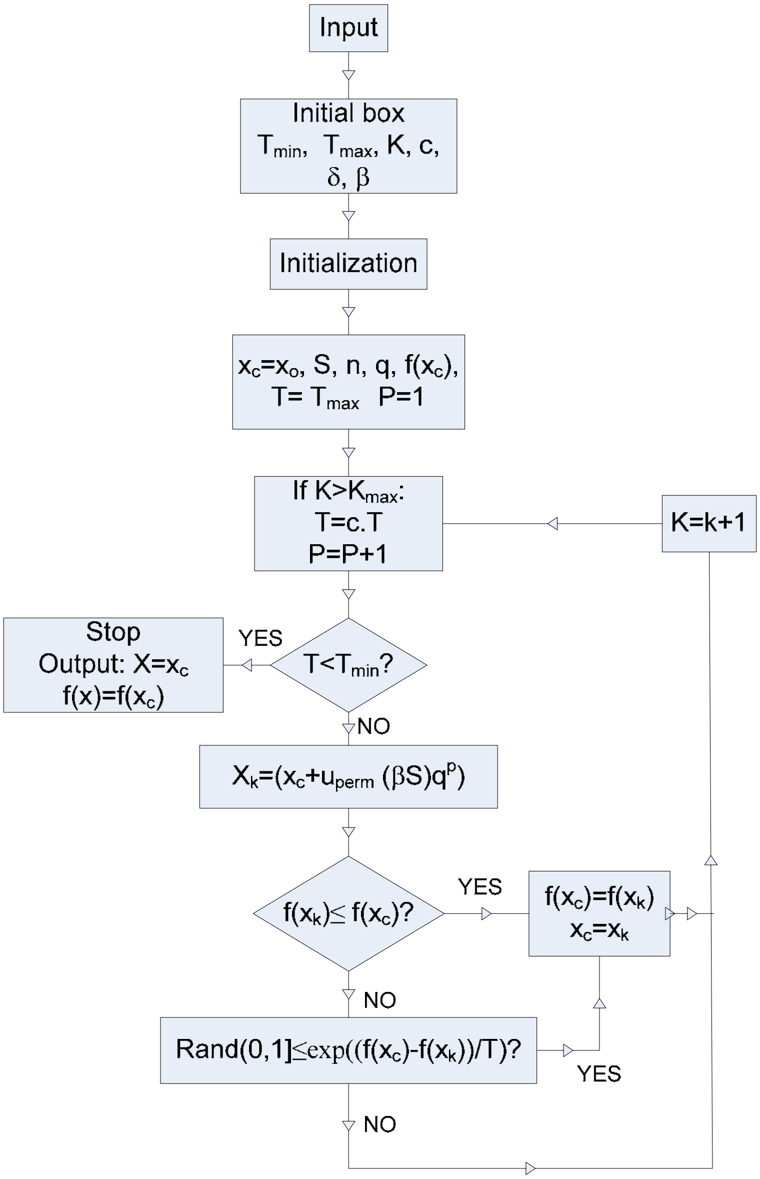

Figure 8.4 Block diagram of the modified simulated annealing algorithm

In order to check the reliability of this neighborhood function, a test is conducted on the famous test problem, the so called six-hump camel back function:

$$f(x) = (4 - 2.1x_1^2 + x_1^{4/3})x_1^2 + x_1x_2 + 4(x_2^2 - 1)x_2^2 \qquad (8.10)$$

This function has two global minimum namely:

f(-0.0898,0.7126) = f(0.0898,-0.7126) = -1.0316.

The simulated annealing algorithm used had $T_{maximum}=1$ and $T_{minimum}=10^{-8}$ with a static cooling schedule $T_{i+1}=0.8\ T_i$ and a maximum number of iterations of 300 per temperature. β is selected to be 0.05 and so the initial leap will be 0.05S.

With conventional algorithm, the global optimum could be easily retrieved using an initial point close enough to the global optimum (Figure 8.5). However, when the static neighborhood function was used with $\varepsilon=0.01$, the global optimum couldn't be retrieved when the initial guess is more than 7 units away from the global optimum (Figure 8.6). However, with the dynamic neighborhood function, the optimum could be retrieved from as far as 200 units (Figure 8.7). These observations were recorded invariably for multiple runs of the algorithm.

As shown in Figure 8.7, the algorithm converges much faster to the global optimum when the neighborhood is made dynamic and wider leaps are allowed initially. In Figure 8.5, the initial point is (1,1) which is close enough to both (0.0898,-0.7126) and (-0.0898,0.7126) while in Figure 8.6, an initial guess of (7,7) was used. In Figure 8.7 however, a box is fed to the algorithm defined on the x-axis with the interval [-100,300] and the same on the y-axis. The center of this square which served as the initial guess for the algorithm was the point (200,200).

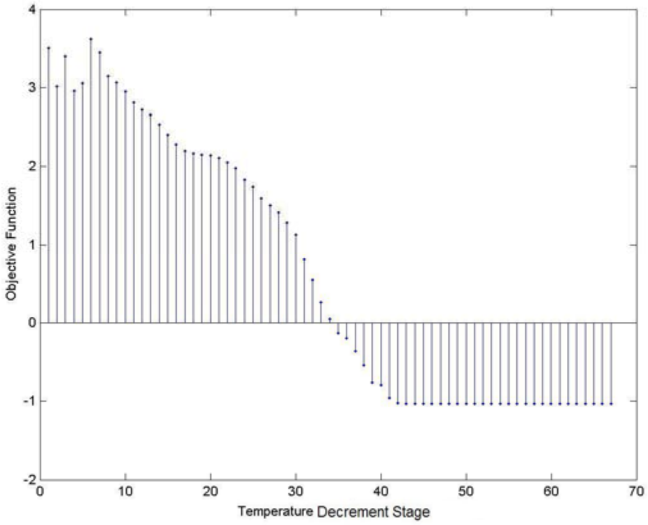

Figure 8.5 The SA performance on the six-hump camel back function using an initial guess close to the global optimum

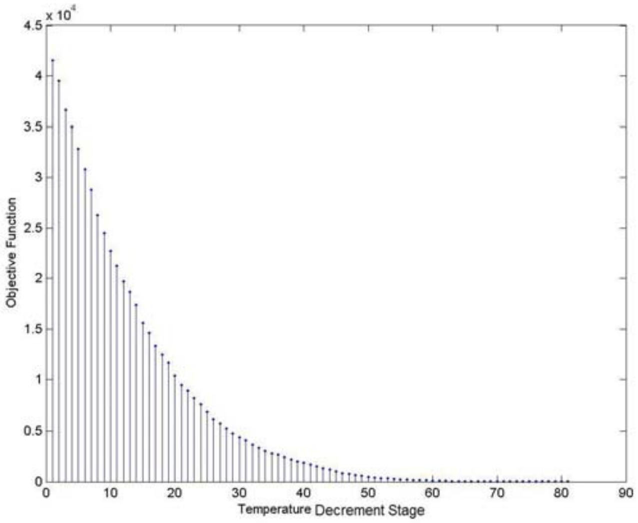

Figure 8.6 The SA performance on the six-hump camel back function using an initial guess slightly far from the global optimum

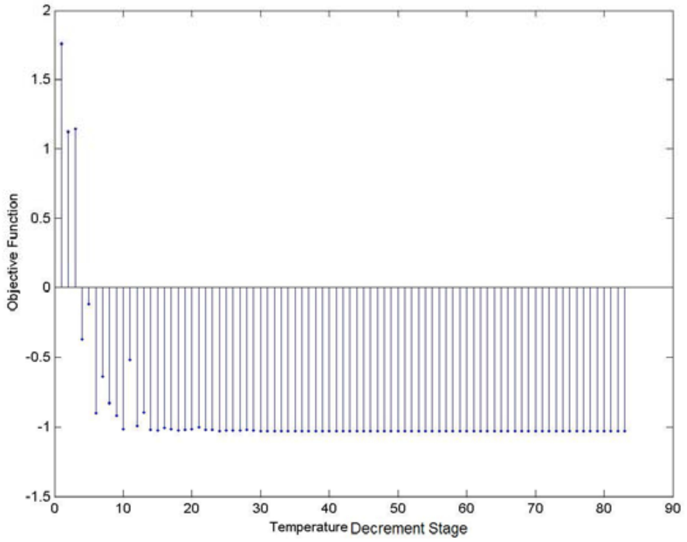

Figure 8.7 The SA performance on the six-hump camel back function using a box with its center serving as an initial point is 200 units far from the global optimum

In all of the above plots, the function value is only plotted at the beginning of each temperature change. Only 20 consecutive acceptances of the new functions are allowed and a maximum of 1000 rejections is allowed. This is to allow diversification.

In interval analysis with the continuing splitting of the initial box into small boxes, it might be computationally practical to set up the number of iterations relative to the volume of the sub-box being searched. This will allow a robust utilization of the computation time; the algorithm will then give more time to larger spaces to be searched and avoid spending superfluous number of iterations on smaller spaces.

Simulated annealing with the proposed neighborhood function is compared with other solutions in the literature (Shang et al. 2007). The reported method (Table 8.1) is programmed in Fortran 95 for working on the windows XP system with Intel c1.7G CPU and 256M RAM. The current work is programmed in Matlab on Windows XP system with Intel processor 1.79 GHz and 1.00 GB of RAM.

For these functions, simulated annealing with the proposed neighborhood function is about substantially faster than the reported results. It is worth-mentioning here that faster performance still obtained even when the initial point is not the center of the box where it happens to be the global minimum as in the second example. Moreover, when expanding the initial box up to ten times, simulated annealing still gives significantly faster results. The main drawback with simulated annealing will still be the fact that it is a metaheuristic method.

Central Point Evaluation

Simulated annealing will be used to find a good solution of the objective function inside a box and then will be compared to the upper bound. When the box is very small then it is computationally more feasible to, instead of using simulated annealing, the central point of the box is evaluated and treated the way a point by simulated annealing is treated. The time needed to evaluate the central point and splitting the box is much faster than applying simulated annealing.

Table 8.1 Comparing the Performance of Simulated Annealing with a Reported Method

Function	Reported computation Time (s)	SA method Computation time (s)
$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 - x_1x_2 - 4x_2^2 + 4x_2^4$ $-3 \le x_1, x_2 \le 3$ $x^* = (0.0898, 0.7127)\ or\ (-0.0898, -0.7127) \qquad f(x^*) = -1.0316$	26.3325	0.2180
$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2)$ $-1 \le x_1, x_2 \le 1$ $x^* = (0.0000, 0.0000) \qquad f(x^*) = -2.0000$	33.7326	0.2650
$f(x) = [1 - 2x_2 + 0.2\sin(4\pi x_2) - x_1]^2 + [x_2 - 0.5\sin(2\pi x_1)]^2$ $-3 \le x_1, x_2 \le 3$ $x^* = (0.3061, 0.4677) \qquad f(x^*) = 0.0000$	38.2165	0.4510
$f(x) = \frac{\pi}{10}\left\{10\sin^2(\pi x_1) + \sum_{i=1}^{9}\left[(x_i - 1)^2(1 + 10\sin^2(\pi x_{i+1}))\right] + (x_n - 1)^2\right\}$ $-10 \le x_i \le 10 \quad i = 1,2,...,10$ $x^* = (1.0000, 1.0000, ..., 1.0000) \qquad f(x^*) = 0.0000$	44.2193	1.3430

Search Regions' Bookkeeping

Another subject that needs to be dealt with is the organization of the unsearched regions. In its initial shape, the feasible domain can be easily defined as a box in n-dimensional space with the initial ***n*** intervals. However, once the algorithm starts chopping off the suboptimal regions, the definition of the remaining part is not as easy. A handy expression is required not only to define that part but also to be computationally practical.

A practical way to express the remaining part of the box is to split it into smaller boxes. However, the splitting has to be efficient by minimizing the number of the resulting parts of the original. This is because each part will be listed as a sub-box in the search list. The resulting number of parts is a function of the dimension (variables) of the problem and hence the resulting number of parts can be substantial if the method is not selective enough.

The best way is to keep the resulting sub-boxes as large as possible to minimize the number of items in the list to be searched. This is done as shown in Figure 8.8. In each axis, the remaining part is divided on both sides of the chopped cube, the same is done to the next axis and so on. This will result in the creation of ***2n*** elements to be added to the list. Although it seems massive to add ***2n*** sub-boxes after every iteration, these sub-boxes will be directly investigated using the four tests before they are included in the list. If a sub-box fails any of the tests, it will not be included in the list. Moreover, the list will be updated each time the upper bound is updated.

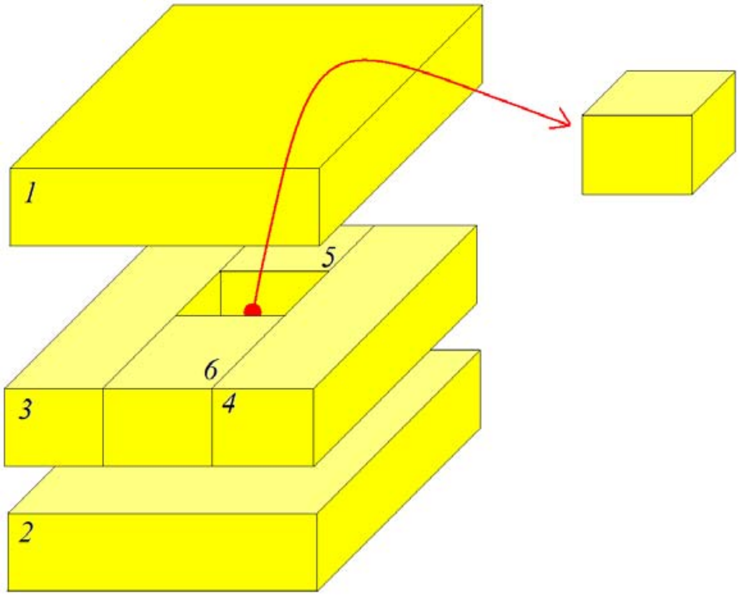

Figure 8.8 Splitting the remaining space into 2***n*** sub-boxes

The chopped sub-box is defined by the intervals, $\{X_{C1},X_{C2},..,X_{Cn}\}$, where X_{ci} denotes the interval defining the side of the box in direction ***i***. Each interval of the original box will have two subintervals, one on each side of the chopped interval (Figure 8.9).

$$X_i=[X_{Li},X_{Ci},X_{Ri}]$$

whereas X_{Li} denotes the subinterval on the left of the chopped one and X_{Ri} denotes the subinterval on the left side. The resulting ***2n*** sub-boxes can be defined as follows:

$$[X_1 \;\; X_2 \;\; ...X_n]-[X_{C1} \;\; X_{C2} \;\; ...X_{Cn}]=\begin{bmatrix} X_{L1}\,X_2\,X_3\,..........\,X_n \\ X_{C1}\;\,X_{L2}\,X_3\,........X_n \\ ..... \\ X_{C1}\,X_{C2}\,X_{C3}\,.......X_{Ln} \\ X_{R1}\,X_2\,X_3\,..........\,X_n \\ X_{C1}\;\,X_{R2}\,X_3\,........X_n \\ ..... \\ X_{C1}\,X_{C2}\,X_{C3}\,.......X_{Rn} \end{bmatrix} \qquad (8.11)$$

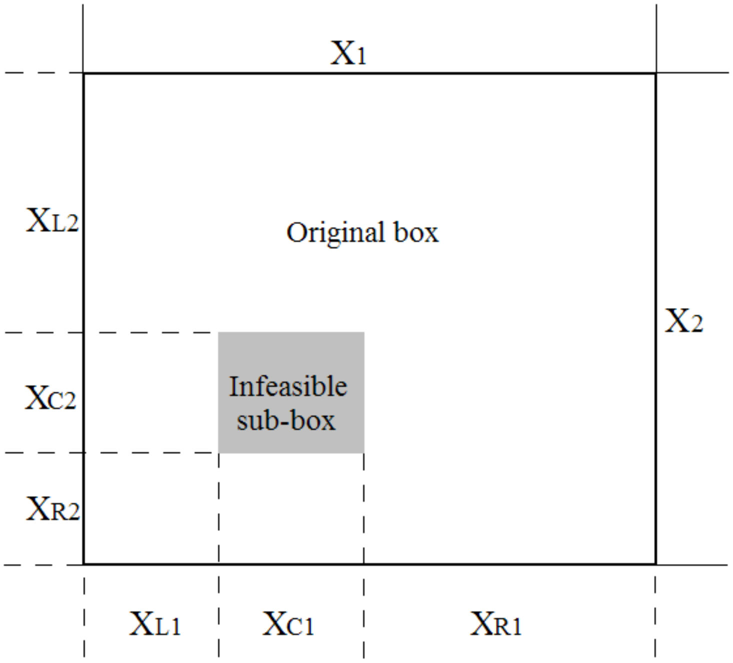

Figure 8.9 Defining infeasible sub-boxes taken from the original box

On the right hand side of the equation, each X denotes an interval and each row defines a new sub-box to be investigated. In a matrix format, the ***2n*** sub-boxes are expressed in two symmetrical matrices as shown in the left hand side of the above equation. The upper ***n***×***n*** matrix has its diagonal being the interval X_{Ri} for ***i***=1,..***n***. Above the diagonal, the whole original interval X_j for $\boldsymbol{j}$=$\boldsymbol{i}$+1…n is presented in each column $\boldsymbol{j}$. While below the diagonal, the interval of the chopped sub-box X_{cj} for $\boldsymbol{j}$=1,..$\boldsymbol{i}$-1 is presented in each column $\boldsymbol{j}$.

Example

To define the sub-boxes resulting from removing the sub-box X_c={[2,4],[5,8],[4,5]} from the parent-box X={[1,10], [1,10], [1,10]}, the following list is given:

$$X - X_C = \begin{bmatrix} [1,2]\,[1,10]\,[1,10] \\ [2,4]\,[1,5]\,[1,10] \\ [2,4]\,[5,8]\,[1,4] \\ [4,10]\,[1,10]\,[1,10] \\ [2,4]\,[8,10]\,[1,10] \\ [2,4]\,[5,8]\,[5,10] \end{bmatrix}$$

The matrix defines the resulting six boxes from removing a sub-box from the parent box in the 3D space.

The Hybrid Algorithm

The developed algorithm is based on using simulated annealing to search in the list of boxes. As the algorithm proceeds, tiny boxes will be produced to the splitting and there testing the central point of the box is more viable than conducting simulated annealing. Sometimes, the problem size suggests using the central method in lieu of simulated annealing throughout the whole procedure.

When integer variables are involved, the integrality requirement will be relaxed when the objective function is evaluated; either by simulated annealing or central point.

Then, if the point is feasible, the variable in the widest direction, if supposed to be integer, will be rounded to the nearest best integer and the new value of the objective function will be compared. If that point is, however, infeasible, it will be taken as to the distrust region subroutine to output an infeasible box. In the infeasible box, the integer side of the infeasible box will be expanded to the neighboring integers. So if the following infeasible box (811.a) with the variable x_i as an integer, is being dealt with:

$$[x_{L1}\ \ x_{R1},...,x_{Li}\quad x_{Ri}\ ,...,x_{Ln}\ \ x_{Rn}] \tag{8.11.a}$$

The following box will be removed from the original one:

$$[x_{L1}\ \ x_{R1},...,\lfloor x_{Li}\rfloor\ \lceil x_{Ri}\rceil,...,x_{Ln}\ \ x_{Rn}] \tag{8.11.b}$$

where $\lfloor . \rfloor$ and $\lceil . \rceil$ denote the floor and ceiling operators, respectively.

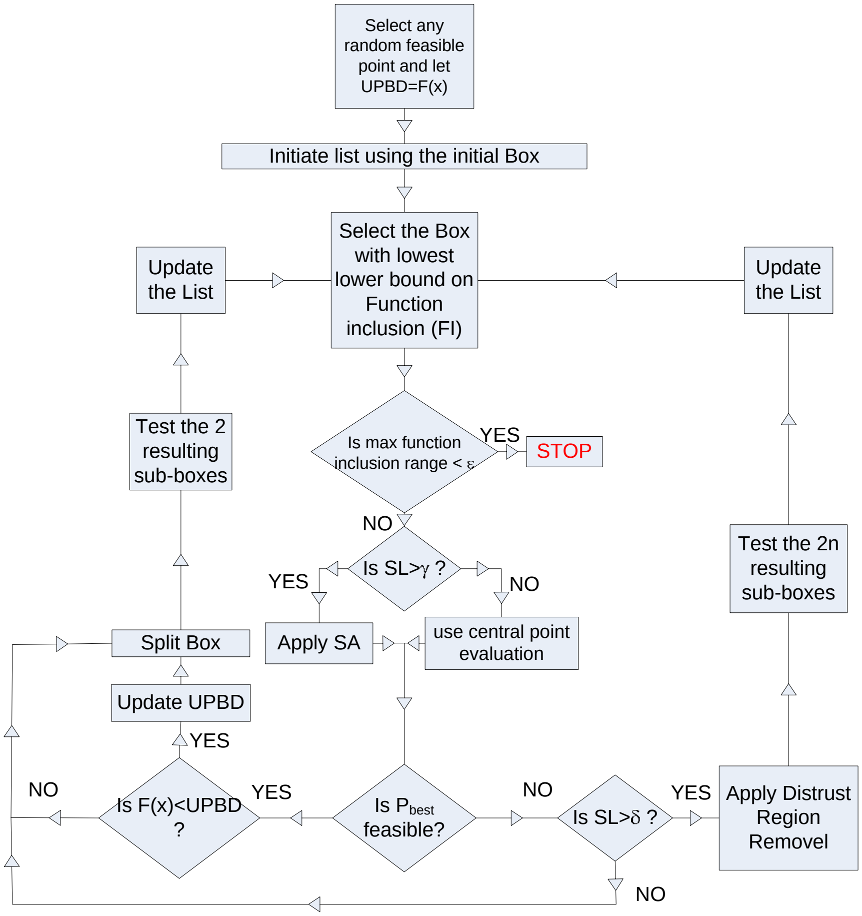
Select any random feasible point and let UPBD=F(x)
Initiate list using the initial Box
Select the Box with lowest lower bound on Function inclusion (FI)
Update the List
Update the List
Is max function inclusion range < ε
YES
STOP
NO
Test the 2 resulting sub-boxes
Is SL>γ ?
YES
NO
Apply SA
use central point evaluation
Test the 2n resulting sub-boxes
Split Box
Update UPBD
YES
NO
Is F(x)<UPBD ?
YES
Is P_best feasible?
NO
Is SL>δ ?
YES
Apply Distrust Region Removel
NO

Figure 8.10 The hybrid algorithm using simulated annealing

The algorithm block diagram is shown in Figure 8.10. Updating the list is included as a black box and will be discussed in the next section.

The hybrid algorithm using simulated annealing runs as follows:

1. Select the stopping criterion for the final maximum width of the objective function inclusion (ε) and the minimum length (γ) for the side of the interval for applying simulated annealing and distrust region.

2. Find a local minimum for the unconstrained problem using simulated annealing and use that point as the initial search point.
3. Check the maximum width of the objective function inclusion in the list, if it is less than ε then STOP, otherwise select a box from the list.
4. Check the total lengths of the box's side, if it is greater than γ do, simulated annealing, otherwise evaluate the object function at the center of the box.
5. Check the feasibility of that point.
6. If the point is feasible, compare the objective function's value with the upper bound. Update the upper bound if the new point is better. If an integer variable is involved and if the widest side is on an integer direction then the value in that direction will be rounded to the next integer that will give a better objective function, if none is feasible then the point will be regarded infeasible.
7. If the point is feasible, then split the box across the widest direction at the feasible point. If that direction is integer, then round the point's component in that direction to the nearest two integers. So if the feasible point component in the side $[x_{iL}\ x_{iR}]$ is x_i such that $x_{iL} \leq x_i \leq x_{iR}$, the box will be split at this direction with the two new boxes having the sides $[x_{iL}$, floor(x_i)] and [ceil(x_i), $x_{iR}]$ in the direction i.
8. If the point is infeasible and if the maximum sides length is greater than δ, then the distrust region will be applied. The infeasible point will be expanded in all directions until the largest strictly infeasible box is obtained. A new set of boxes will result from removing this box from the original one as described before. The

integer directions will also be treated as described in equation (8.11.b). If the total sides' length is less than δ, the box is split at that point as described before. The new set of boxes will be tested before being added to the list.

Interval Tests and Updating the List

For each of the resulting boxes, the following procedure (Figure 8.11) is conducted to test each box:

1. The upper bound test is first conducted. If the lower bound of the box LBX is larger than the upper bound then the whole box is deleted.
2. If the upper bound test did not delete the box, the infeasibility test is then conducted. If the inclusion of any of the constraints is totally positive then the box is deemed strictly infeasible and so is deleted immediately.
3. If the box is not strictly infeasible it will then be added to the list.
4. If the box is strictly feasible, it will be qualified for the other two tests.
5. In the monotonocity test, if zero is not included in any of the gradients inclusions, the objective function is tested at the boundaries of the box and compared to the current upper bound and then the whole box is deleted.

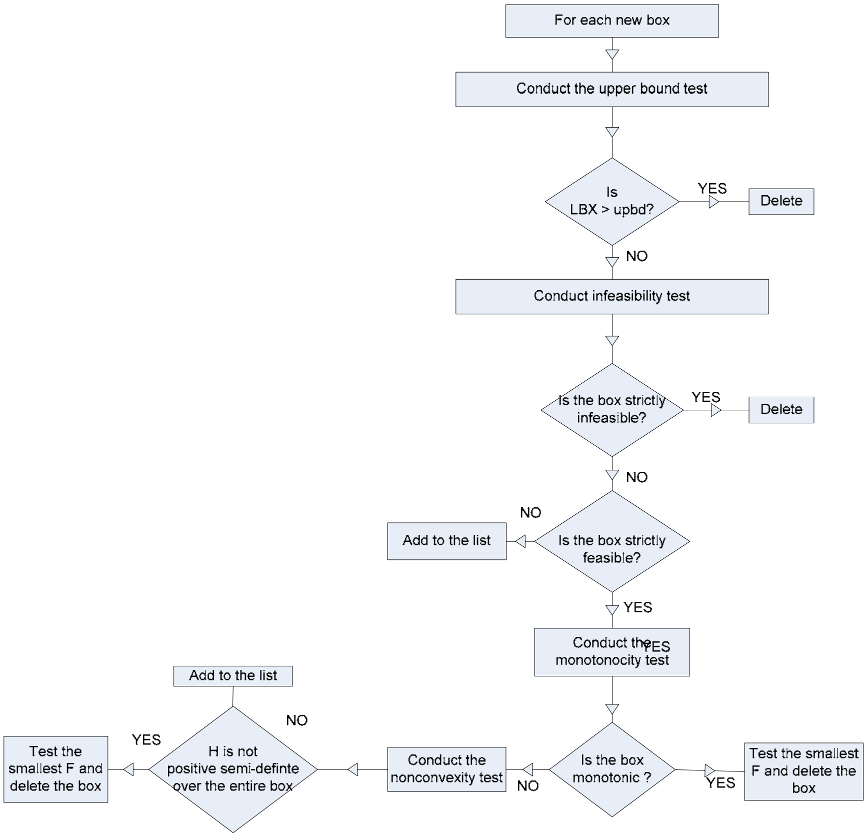

Figure 8.11 The optimality tests conducted on the new sub-boxes

6. If the box survived the monotonoicity, the non-convexity test will then be conducted. If the inclusion of the Hessian diagonal is strictly negative for any of the variables then the objective function is tested at the boundaries of the box and compared to the current upper bound and then the whole box is deleted.

   The surviving boxes will be added to the list and the procedure will repeat. The population of the boxes in the list will increase and will also be updated when ever a better upper bound if found. The algorithm will continue until the boxes in the list have a maximum range of objective function inclusion ε which is the error of the final optimum objective value.

Accelerating the Search

The chance of slow convergence and ridiculously lengthy runtime are very likely for many problems if they are directly fed to the algorithm. In order to overcome this situation, the algorithm can be accelerated by implementing preliminary breaking of the search domain before applying the proposed algorithm.

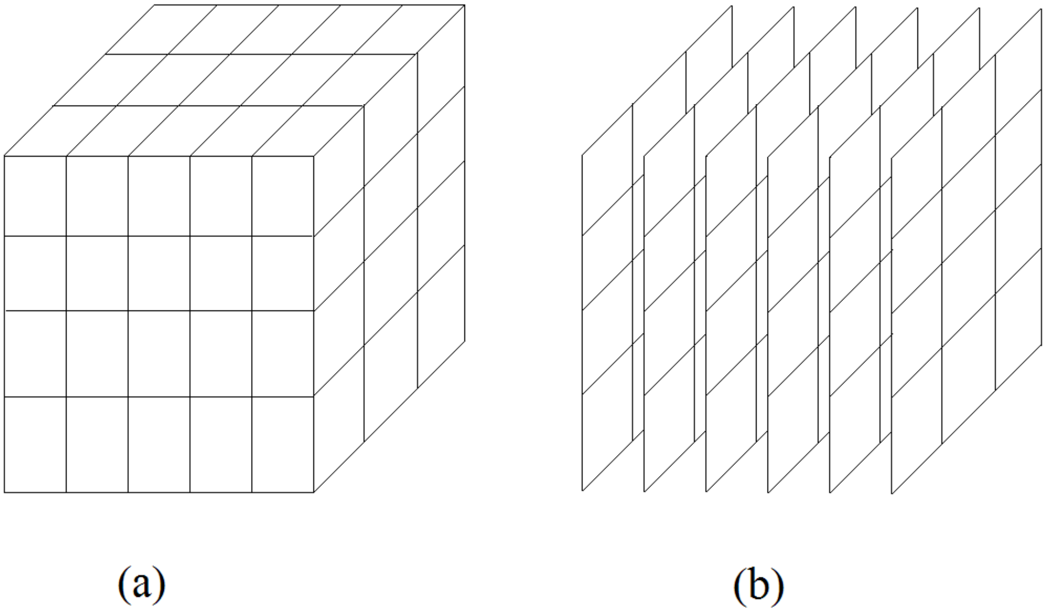

Figure 8.12 Initial partitioning to accelerate the search

A very efficient way to do that is by splitting the initial box into equally partitioned sub-boxes (Figure 8.a) and applying the four tests on them. If the range of an integer variable is not too large, discretizing that range will help reduce the search domain (Figure 8.b).

Applying the four tests on these sub-boxes, after determining an upper bound, will help getting rid of all strictly infeasible portions as well as the strictly feasible ones that can be deleted using the nonconvexity test and the monotonocity test. This preparation can delete a huge portion of the initial search as will be discussed in some of the numerical examples.

For those boxes that survive the preparation step and are eligible to be fed to the proposed algorithm can have their central point evaluated and checked if feasible. These boxes are then sorted ascendingly starting from the lowest feasible central point upwards and so the box with the lowest feasible central point will be searched first.

Numerical Examples

In order to test the validity of the algorithm, numerical examples are presented herein. These examples were programmed in Matlab on Windows XP system with Intel processor 1.79 GHz and 1.00 GB of RAM.

Example 1

The following example (Cardoso et al. 1997; Kocis and Grossmann 1998) has two variables, one is integer and the other is continuous.

$$\min\ 2x + y$$

$$\begin{aligned} s.t. \quad 1.25 - x^2 - y &\le 0 \\ x + y &\le 1.6 \\ 0 \le x &\le 1.6 \\ y &= \{0,1\} \end{aligned}$$

The initial box used in this example is relatively small. It is small enough that central point evaluation converges faster than when simulated annealing is used. Simulated annealing outperforms the central point method when the box is relatively large.

This problem has a local minima at (x,y)=(1.118,0) with an objective function value of 2.236. The global solution was found to be at (x, y) = (0.5, 1) with an objective value of 2. The computation time is 0.16 seconds without the acceleration step.

Example 2

This example is taken from (Floudas 1995; Kocis and Grossmann 1998)

$$\min - 0.7y + 5(x_1 - 0.5)^2 + 0.8$$

$$\begin{aligned} s.t. \quad -\exp(x_1 - 0.2) - x_2 &\le 0 \\ x_2 + 1.1y &\le -1 \\ x_1 - 1.2y &\le 0.2 \\ 0.2 \le x_1 &\le 1 \\ -2.22554 \le x_2 &\le -1 \\ y &= \{0,1\} \end{aligned}$$

When solving this problem, central point evaluation is only allowed when the range of the objective function inclusion, or the total length of the box’s sides are less than one. The initial upper bound used was f(x$_1$,x$_2$,y)=f(0.2.-1,0)=1.25. The global solution was found at f(x$_1$,x$_2$,y)=f(0.9419.-2.1,1)=1.0765. The computation time was 0.73 seconds without the acceleration step.

Example 3

This example is taken from (Floudas et al. 1999).

$$\min\ 0.5t_1t_2^{-1} - t_1 - 5t_2^{-1}$$

$$0.01t_2t_3^{-1} + 0.01t_1 + 0.0005\,t_1\,t_3 - 1 \le 0$$

$$1 \le t_1, t_2, t_3 \le 100$$

The initial feasible point was selected at t= (1,1,1) and hence an initial upper bound f(t)=5.5. By setting the maximum width of the objective function inclusion at 0.001, the global solution of the objective function is -83.256 at t = (88.353, 7.6857, 1.3161). The computation time was 47.1 seconds.

Example 4

The following problem is adopted from (Kocis and Grossmann 1998).

$$\min f(x,y) = (y_1-1)^2 + (y_2-2)^2 + (y_3-1)^2 - \ln(y_4+1) + (x_1-1)^2 + (x_2-2)^2 + (x_3-3)^2$$

$$\begin{aligned} s.t. \qquad p_1(x,y) &= y_1 + y_2 + y_3 + y_4 + x_1 + x_2 + x_3 - 5 \le 0 \\ p_2(x,y) &= y_3^2 + x_1^2 + x_2^2 + x_3^2 - 5.5 \le 0 \\ p_3(x,y) &= y_1 + x_1 - 1.2 \le 0 \\ p_4(x,y) &= y_2 + x_2 - 1.8 \le 0 \\ p_5(x,y) &= y_3 + x_3 - 2.5 \le 0 \\ p_6(x,y) &= y_4 + x_1 - 1.2 \le 0 \\ p_7(x,y) &= y_2^2 + x_2^2 - 1.64 \le 0 \\ p_8(x,y) &= y_3^2 + x_3^2 - 4.25 \le 0 \\ p_9(x,y) &= y_2^2 + x_3^2 - 4.64 \le 0 \end{aligned}$$

$$y = \{0,1\}^4$$
$$0 \le x_i \le 3 \quad : i = 1,2,3$$

The initial feasible point was selected at ($x_1, x_2, x_3, y_1, y_2, y_3, y_4$)=(0,0,0,0,0,0,0) and hence an initial upper bound f(x,y)=20. By setting the maximum width of the objective function inclusion at 0.00001, the best answer was located at ($x_1, x_2, x_3, y_1, y_2, y_3, y_4$)= (0.2, 0.8, 1.9079, 1, 1, 0, 1) with an objective function value of 4.5796.

Using the acceleration step in this problem has reduced the searched domain to less than 15%. The computation time is 113.97 seconds.

Conclusion

A hybrid algorithm has been introduced for the global solution of MINLPs. The approach couples interval-based techniques with simulated annealing. Several interval arithmetic tools are used. These include interval-based tests for upper bound, feasibility, monotonicity, and non-convexity. Additionally, when an infeasible point is located, a distrust region approach is used to maximize the size the eliminated intervals. Simulated annealing is used to accelerate the search by identifying near-optimal solutions that can be used to quickly update upper bounds used in eliminating search spaces. A neighboring function is used in the context of a revised simulated annealing algorithm. Also, new interval decomposition and indexing techniques are introduced to keep track of the unsearched boxes. This hybrid algorithm provided a handy programmable algorithm that can solve the general non-convex mixed integer nonlinear problems. The algorithm is however dependent on the topography of the feasible domain, the volume of the initial box and the number of variables and their type being integer or continuous.

CHAPTER IX

CONCLUSIONS AND RECOMMNDATIONS

Conclusions

This work is aimed at introducing new methods to the design and analysis of the utility system. Employing the thermodynamic properties in the suggested approaches has proved very efficient and more accurate compared to the literature.

An optimization problem has been formulated to optimize the power cycle in conjunction with the chemical process. By involving an expression of the performance as well as the equipment cost, the problem is to solve for the best economic option by trading off the power produced with the fuel spent at the boiler.

The work in the literature on targeting the utility system tends to be off the required accuracy from an economic viewpoint. This deviation is in great part attributed to the interdependency of the involved variables. A general algorithm has been developed using the thermodynamic properties' correlations. This algorithm is made general enough to involve streams coming from the process and others sent to the process. The accuracy of this algorithm is much better than the literature work.

The next step was to optimize the pressure levels. Optimizing the pressure level is a cumbersome task. The developed algorithm was fast enough to run an exhaustive search for the optimum method. The maximum shaft work produced and the maximum area on the grand composite curve, were found to locate at the same pressure values. This observation has shifted the complexity of the problem from optimizing the formulated problem to optimizing the area on the grand composite curve.

The pipeline layout has been optimized by employing the optimization techniques in graph theory. The major limitation of these techniques is the computational time but they are very efficient for small scale problems such as the case of the utility system.

The last part of this work was devoted to developing a global optimization technique using a heuristic method. Simulated annealing is a widely used heuristic method that is more common in integer problem. An approach has been developed to use that method in the case of the contentious problem. The approach has proven very efficient compared to the literature. An optimization algorithm has been developed to use simulated annealing for targeting global solution using interval analysis.

Recommendations for Future Work

A further advancement to this work is to incorporate the material management in the chemical process with the utilities. Many chemical substances can serve as working fluid in heating and refrigeration cycles. These substances could exist as products and byproducts in a chemical plant. Using the cooling/heating potential of these substances in conjunction with heat integration and power requirement of the chemical process will be a useful work in optimizing the plant economy.

The current work tackles the design problem of the utility system assuming steady operation. A further development is to solve the same problem for the case of unsteady operation and fluctuating process requirement of heat and power.

REFERENCES

Alandi, P. P., Alvarez, J. F. O., and Martin-Benito, J. M. T. (2007). "Optimization of irrigation water distribution networks, layout included." *Agricultural Water Management*, 88, 110-118.

Boche, R. E. (1963). "An operational interval arithmetic." *Proc., IEEE National Electronics Conference,* Chicago, 1-14.

Bohachevsky, I. O., Johnson, M. E., and Stein, M. L. (1986). "Generalized simulated annealing for function optimization." *Techonometrics*, 28(3), 209-217.

Branan, C. (2002). *Rules of thumb for chemical engineers*, Gulf Professional Publishing, Houston.

Byrne, R. P., and Bogle, I. D. L. (1995). "An accelerated interval method for global optimisation." *Computers and Chemical Engineering*, 20, S49-S54.

Byrne, R. P., and Bogle, I. D. L. (1996). "Solving nonconvex process optimisation problems using interval subdivision algorithms." *Global optimization in engineering design,* I. E. Grossmann, ed., Kluwer Academic Publishers, Dordrecht, The Netherlands, 155-174.

Cardoso, M. F., Salcedo, R. L., Azevedo, S. F. d., and Barbosa, D. (1997). "A simulated annealing approach to the solution of MINLP problems." *Computers and Chemical Engineering*, 21(12), 1349-1346.

Çengel, Y. A., and Boles, M. A. (1999). *Thermodynamics: An engineering approach*, McGraw Hill, New York.

Choi, S. H., Ko, H. W., and Manousiouthakis, V. (1999). "A stochastic approach to global optimization of chemical processes." *Computers and Chemical Engineering*, 23, 1351-1356.

Connolly, D. T. (1990). "An improved annealing scheme for the QAP." *European Journal of Operational Research*, 64(1), 93-100.

Cook, W. J., Cunningham, W. R., Pulleyblank, W. R., and Schrijver, A. (1998). *Combinatorial optimization*, John Wiley & Sons, New York.

Dhole, V., and Linnhoff, B. (1993). "Total site targets for fuel, co-generation, emissions and cooling." *Computers and Chemical Engineering*, 17, S101-S109.

Dowsland, K. A. (1995). "Simulated annealing." *Modern heuristic techniques for combinatorial problems*, C. R. Reeves, ed., Blackwell Scientific Press, Oxford, 20-69.

EDUCOGEN, and INESTENE. (2001). "A guide to cogeneration." http://www.cogen.org/Downloadables/Projects/EDUCOGEN_Cogen_Guide.pdf (November 1, 2006)

Eglese, R. W. (1990). "Simulated annealing: A tool for operational research." *European Journal of Operational Research*, 46, 271-281.

El-Halwagi, M. M. (2003). *Pollution prevention through process integration*, Academic Press, San Diego.

El-Halwagi, M. M. (2006). *Process integration*, Academic Press, San Diego.

Evans Jr., F. L. (1971). *Equipment design handbook for refineries and chemical plants*, Gulf Publishing, Houston, TX.

Floudas, C. A. (1995). *Nonlinear and mixed-integer optimization- fundamentals and applications.*, Oxford University Press, Oxford, U.K.

Floudas, C. A., Pardalos, P. M., Adjiman, C. S., Esposito, W. R., Gümüs, Z. H., Harding, S. T., Klepeis, J. L., Meyer, C. A., and Schweiger, C. A. (1999). *Handbook of test problems in local and global optimization*, Kluwer Academic Publishers, Dordrecht, The Netherlands.

Fronseca Jr., J. G. (2003). "Análise energética e exergética de um ciclo rankine com aquecimento distrital: Estudo de uma planta termoelétrica." Ph.D. dissertation, Dept. of Mechanical Engineering, The Federal University do Rio Grande do Sul, Porto Alegre, Brazil (in Portuguese).

Goffe, W. L., Ferrier, G. D., and Rogers, J. (1994). "Global optimization of statistical functions with simulated annealing." *Journal of Econometrics*, 60, 65-99.

Goldstein, L., and Waterman, M. (1988). "Neighborhood size in the simulated annealing algorithm." *American Journal of Mathematical and Management Sciences*, 8(3,4), 409-423.

Grossman, I. E. (1985). "Mixed-integer programming approach for the synthesis of integrated process flowsheets." *Computers and Chemical Engineering*, 9(5), 463-482.

Guirardello, R., and Swaney, R. E. (2005). "Optimization of process plant layout with pipe routing." *Computers & Chemical Engineering*, 30, 99-114.

Harell, D. A. (2004). "Resource conservation and allocation via process integration." Ph.D. dissertation, Dept. of Chemical Engineering, Texas A&M University, College Station, Texas.

Hohman, E. C. (1971). "Optimum networks for heat exchanger." Ph.D. dissertation, Dept. of Chemical Engineering, University of Southern California, Los Angeles, California.

Hul, C.-W., and Natori, Y. (1996). "An industrial application using mixed-integer programming technique: A multi-period utility system model." *Computers and Chemical Engineering,* 20, S1577-S1582.

IFC-ASME. (1967). "The 1967 IFC formulation for industrial use." *Proc., International Formulation Committee of the 6th International Conference on the Properties of Steam*, Düsseldorf, Germany, 1-50.

Ito, T. (1999). "A genetic algorithm approach to piping route path planning." *Journal of Intelligent Manufacturing*, 10, 103-114.

Jaulin, L., Kieffer, M., Didrit, O., and Walter, É. (2001). *Applied interval analysis*, Springer, New York.

Kirkpatrick, S., C. D. Gelatt, J., and Vecchi, M. P. (1983). "Optimization by simulated annealing." *Science*, 220, 671-680.

Kocis, G. R., and Grossmann, I. E. (1998). "Global optimization of nonconvex mixed-integer nonlinear programming (MINLP) problems in process synthesis." *Industrial Engineering Chemistry Research*, 27, 1407-1421.

Kreinovich, V. (2007). "Interval computations and interval-related statistical techniques: Tools for estimating uncertainty of the results of data processing and indirect measurements " *Rep No. UTEP-CS-07-53*, University of Texas at El Paso, El Paso, Texas.

Kumana and Associates. (2003). "How to calculate the true cost of steam." *Rep No. DOE/GO-102003-1736*, U.S. Department of Energy, Washington D.C.

Laarhoven, P. J. M. v., and Aarts, E. H. L. (1988). *Simulated annealing: Theory and practice*, D. Reidel Publishing Company, Boston.

Linnhoff, B., and Hindmarsh, E. (1983). "The pinch design method for heat exchanger networks." *Chemical Engineering Science*, 38(5), 745-763.

Linnhoff, B., Townsend, D. W., Boland, D., Hewitt, G. F., Thomas, B. E. A., Guy, A. R., and Marsland, R. H. (1982). *User guide on process integration for the efficient use of energy*, IChemE, Warwickshire, UK.

Lundy, M., and Mees, A. (1986). "Convergence of an annealing algorithm" *Mathematical Programming*, 34, 111-124.

Maia, L. O. A., and Qassim, R. Y. (1997). "Synthesis of utility systems with variable demands using simulated annealing." *Computers and Chemical Engineering*, 21(9), 947-950.

Mavromatis, S. P., and Kokossis, A. C. (1998). "Conceptual optmisation of utility networks for operational variations I. Targets and level optimisation." *Chemical Engineering Science*, 53(8), 1585-1608.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). "Equation of state calculation by fast computing machines." *Journal of Chemical Physics*, 21, 1087-1091.

Moore, R. E. (1962). *Interval arithmetic and automatic error analysis in digital computing,* Applied Mathematics and Statistics Lab., Stanford University, Stanford, California.

Moore, R. E. (1965). "The automatic analysis and control of error in digital computation based on the use of interval numbers." *Error in digital computation,* L. B. Rall, ed., John Wiley & Sons. Inc., New York, 103-112.

Moore, R. E. (1966). *Interval analysis*, Prentice-Hall, Englewood Cliffs, NJ.

Morton, R. J., and Linnhoff, B. (1984). "Individual process improvements in the context of site-wide interactions." *Proc., IChemE Annual Research Meeting*, Bath, UK, 101-124.

Neumaier, A. (1990). *Interval methods for systems of equations,* Cambridge University Press, Cambridge, U.K.

Nisho, M. (1977). *Computer aided synthesis of steam and power plants for chemical complexes*, The University of Western Ontario, London, Canada.

Nisho, M., and Johnson, A. I. (1979). "Strategy for energy system expansion." *Chemical Engineering Progress*, 75(7), 73-79.

Papandreou, V., and Shang, Z. (2008). "A multi-criteria optimisation approach for the design of sustainable utility systems." *Computers and Chemical Engineering*, 32, 1589-1602

Papoulias, S. A., and Grossman, I. E. (1983). "A structural optimization approach in process synthesis. I. Utility systems." *Computers and Chemical Engineering*, 7(6), 695-706.

Raissi, K. (1994). "Total site integration." Ph.D. thesis, Dept. of Process Integration, University of Manchester Institute of Science and Technology, Manchester, UK.

Ratschek, H., and Rokne, J. (1988). *New computer methods for global optimization*, John Wiley & Sons, New York.

Ratschek, H., and Rokne, J. (1991). "Interval tools for global optimization." *Computers Math. Applic*, 21(6/7), 41-50.

Rayward-Smith, V. J., Smith, I. H., Reeves, C. R., and Smith, G. D. (1996). *Modern heuristic search methods*, John Wiley & Sons Ltd, Chichester, UK.

Rodríguez-Toral, M. A., Morton, W., and Mitchell, D. R. (2001). "The use of new SQP methods for the optimization of utility systems." *Computers and Chemical Engineering*, 25(2-3), 287-300.

Salisbury, J. K., and Schenectady, N. Y. (1942). "The steam-turbine regenerative cycle - an analytical approach." *Transactions of the ASME*, 64(4), 231-245.

Seider, W. D., Seader, J. D., and Lewin, D. R. (2004). *Product & process design principles*, Wiley & Sons, New York.

Shang, Y. L., Pu, D. G., and Jiang, A. P. (2007). "Finding global minimizer with one-parameter filled function on unconstrained global optimization." *Applied Mathematics and Computation*, 191, 176-182.

Shang, Z., and Kokossis, A. (2004). "A transhipment model for the optimisation of steam levels of total site utility system for multiperiod operation." *Computers and Chemical Engineering*, 28, 1673-1688.

Sloley, A. W. (2001). "Distillation column operations." http://www. distillationgroup. com/ distillation/H003/H003.htm (March 1, 2006)

Smith, R. (2005). *Chemical process design and integration*, John Wiley & Sons Ltd., London.

Supranto, S., Chandra, I., Unde, M. B., Diggory, P. J., and Holland, F. A. (1986). "Heat pump assisted distillation I: Alternative ways to minimize energy consumption in fractional distillation." *Energy Research*, 10, 145-161.

Szu, H., and Hartley, R. (1987). "Fast simulated annealing " *Physics Letters A*, 122(3,4), 157–162

Umeda, T., Itoh, J., and Shiroko, K. (1979). "A thermodynamic approach to the synthesis of heat integration systems in chemical processes." *Computers and Chemical Engineering*, 3, 273-282.

Vaidyanathan, R., and El-Halwagi, M. (1994). "Global optimization of nonconvex programs via interval analysis." *Computers and Chemical engineering*, 18, 889-897.

Vaidyanathan, R., and El-Halwagi, M. (1996). "Global optimization of nonconvex MINLP's by interval analysis." *Global optimization in engineering design*, I. E. Grossmann, ed., Kluwer Academic Publishers, Dordrecht, The Netherlands, 157-193.

Vanderbilt, D., and Louie, S. G. (1984). "A Monte Carlo simulated annealing approach to optimization over continuous variables." *Journal of Computational Physics*, 56, 259-271.

Varbanov, P. S., Doyle, S., and Smith, R. (2004). "Modelling and optimization of utility systems." *Trans IChеE*, 82(A5), 561-578.

Wilkendorf, F., Espuña, A., and Puigjaner, L. (1998). "Minimization of the annual cost for complete utility systems." *Chemical Engineering Research and Design*, 76(3), 239-245.

Winston, W. L. (1994). *Operations research, applications and algorithms*, International Thomson Publishing, Belmont, CA.

APPENDIX A

TURBINE EFFICIENCY MODELS

For a given turbine, the isentropic efficiency is a function of the flow load and hardware parameters provided by the manufacturer. This expression is based on Willan's line that correlates the load to the power output. The isentropic efficiency is expressed by equation (A.1) (Mavromatis and Kokossis, 1998).

$$\eta_{is} = \frac{6}{5B}\left(1 - \frac{3.41443 \times 10^{6}\ A}{\Delta h_{is}\ M^{\max}}\right)\left(1 - \frac{M^{\max}}{6M}\right) \qquad \text{(A.1)}$$

and when M is at the maximum:

$$\eta_{is} = \frac{1}{B}\left(1 - \frac{3.41443 \times 10^{6}\ A}{\Delta h_{is}\ M^{\max}}\right) \qquad (A.2)$$

where A and B are constants dependent on the turbine and are functions of the inlet saturation temperature and the flowrate is measured in lb/hr. A good approximation, within 2%, is given by the following straight line segments (Mavromatis and Kokossis 1998; Varbanov et al. 2004):

$$A = a_0 + a_1\ T^{sat} \qquad \text{(A.3.a)}$$

$$B= a_2 + a_3\ T^{sat} \qquad \text{(A.3.b)}$$

Table A.1 The Regression Coefficients Used i=n the Isentropic Efficiency Equation

	Back Pressure Turbines		Condensing Turbines	
	$W^{max} < 4.10$ MMBtu/s	$W^{max} > 4.10$ MMBtu/s	$W^{max} < 5.12$ MMBtu/s	$W^{max} > 5.12$ MMBtu/s
a_0 (Btu/s)	-0.1518	-1.038755556	-0.115877778	-0.062488889
a_1 (Btu/s . $^{o}F^{-1}$)	0.00065	0.003461111	0.000555556	0.000777778
a_2	0.961977778	1.111644444	1.195233333	1.166466667
a_3 ($^{o}F^{-1}$)	0.000844444	0.000261111	0.000333333	0.000166667

The coefficients in Table A.1 are taken from Varbanov et. al (2004) and Mavromatis and Kokosis (1998). They exist in the SI units in those references but converted here into the American customary system for consistency with the other examples and quantities in this work.

Another expression of the isentropic efficiency developed by (Varbanov et al. 2004) is:

$$\eta_{is} = \frac{n.m - W_{INT}}{m.\eta_{mech}.\Delta H_{is}} \qquad (A.4)$$

whereas n and W_{int} are the slope and the intercept of the linear Willan's line respectively and given by this equation:

$$W_{\text{int}} = \frac{L}{b}(\Delta H_{is} - \frac{a}{m_{\max}}) \qquad (A.5)$$

$$n = \frac{L+1}{b}\left(\Delta H_{is}\, m_{\max} - a\right) \qquad (A.6)$$

with L is to be estimated by correlating the performance of each specific turbine. The

parameters a and b depends on the type pf pressure, maximum power load and the saturation temperature differences as Table B.1 (Smith 2005).

$$a = a_0 + a_1 \Delta T^{sat} \tag{A.7.a}$$

$$b= a_2 + a_3 \Delta T^{sat} \tag{A.7.b}$$

Table A.2 The Regression Coefficients Used in the Isentropic Efficiency Equation

	Back Pressure Turbines		Condensing Turbines	
	W^{max} < 2000 kW	W^{max} >2000 kW	W^{max} < 2000 kW	W^{max} >2000 kW
a_0 (kW)	0	0	0	-463.
a_1 (kW/^{o}C)	1.08	4.23	0.662	3.53
a_2	1.097	1.155	1.191	1.220
a_3 (^{o}C^{-1})	0.00172	0.000538	0.000759	0.000148

APPENDIX B

PROOF OF CONVERGANCE

The main problem that the algorithm is iterating on, is to determine the efficiency at the given flowrate. The convergence criterion is measured by comparing the output of two consecutive mass flowrates going to the process.

To prove the convergence, a single turbine is assumed and hence a single expansion zone of a process. This should not cause any lose of generality as will be explained later.

For a given single expansion zone, the steam is produced at the boiler at a given fixed enthalpy. The efficiency is then calculated as a function of the mass flowrate and the isentropic efficiency. However, that mass flowrate has to conform to the prescribed heat flow at the exit of the expansion zone. This makes the mass flowrate the most convenient variable for checking convergence. Before proofing convergence, the mass flowrate is expressed in a recursive function that performs fixed-point iterations.

Although the Mavromatis and Kokossis (1998) model is used here, the same procedure should work on other models.

$$m_z^{in} = \frac{Q_z^{in}}{h_{z,out} - h_{f,z}} \qquad (B.1)$$

$$h_{a,z,out} = h_{s,z,in} - \eta_z \Delta h_{s,z} \qquad (B.2)$$

$$\eta_{is} = \frac{1}{B}\left(1 - \frac{A}{\Delta h_{is}\, M^{\max}}\right) \qquad (B.3)$$

For a single expansion zone, the mass flowrate passing the expansion zone, M is the same as the mass flowrate discharged to the process, m.

Equation (B.1) is recursively dependent on (B.2) and (B.3). Equation (B.1) can be expressed in terms of the other two. The following equation is obtained for such substitutions where all the subscripts are removed for simplicity and since a single expansion zone is used.

$$g(m_i) = m_{i+1} = \frac{Q}{h_{in} - \frac{1}{B}(1 - \frac{A}{\Delta h_{is} m_i})\Delta h_{is} - h_f} \qquad (B.4)$$

Before proving, the absolute value of the derivative of g(m$_i$) with respect to m$_i$ needs to be shown, denoted by g'(m$_i$) is less than or equal to a number K which is a less than unity i.e.:

$$\left| g'(m_i) \right| \le K < 1$$

$$g'(m_i) = \frac{dg}{dm_i} = \frac{QA}{Bm_i^2} \frac{1}{\left(h_{in} - \frac{1}{B}(1 - \frac{A}{\Delta h_{is} m_i})\Delta h_{is} - h_f \right)^2} \qquad (B.5)$$

Reorganizing the equation and replacing Q with m$_i$ (h$_{out}$ – h$_f$)

$$g'(m_i) = \frac{A}{Bm_i^2} \frac{m_i\,(h_{out} - h_f)}{\left(h_{out} - h_f\right)\left(h_{in} - \frac{1}{B}(1 - \frac{A}{\Delta h_{is} m_i})\Delta h_{is} - h_f \right)}$$

Simplifying this equation:

$$g'(m_i) = \frac{A}{Bm_i}\frac{1}{\left(h_{in} - \frac{1}{B}(1 - \frac{A}{\Delta h_{is} m_i})\Delta h_{is} - h_f\right)} = \frac{A}{BQ} \qquad (B.6)$$

Since A<BQ, it is now easy to notice that $g'(m_i) \leq K < 1$.

Having proved that fact, the convergence of the algorithm is now ready for provig .Recall that the algorithm of the single expansion zone works on the fixed-point iteration given in (B.4):

$$g(m_i) = m_{i+1} = \frac{Q}{h_{in} - \frac{1}{B}(1 - \frac{A}{\Delta h_{is} m_i})\Delta h_{is} - h_f} \qquad (B.4)$$

Assume there is a solution of (B.4) which is s=g(s), using the mean value theorem of differential calculus, there is a value t between s and m such that:

$g(m) - g(s) = g'(t)(m - s)$

Since g(s) =s and from equation (4), $m_1= g(m_0)$, $m_2=g(m_1)$,$m_3=g(m_2)$ …$m_n=g(m_{n-1})$:

$|m_n-s|=|g(m_{n-1})- g(s)| = |g'(t)|\ |m_{n-1} - s|$

But $g'(t_1) \leq K \rightarrow |m_n-s| \leq K\ |m_{n-1} - s|=K\ |g'(t_1)|\ |m_{n-2} - s| \leq K^2\ |m_{n-2} - s|\ldots \leq K^n\ |m_0-s|$

Since K<1, $K^n \rightarrow 0$ as $n\rightarrow\infty$ which means $|m_n-s| \rightarrow 0$ as $n \rightarrow\infty$ and that concludes the proof of the convergence of the algorithm for a single expansion zone.

APPENDIX C

MATLAB CODES

Code No.	**Description**
1	Finding the optimum steam turbine inlet conditions (Chapter IV)
2	The very general utility system targeting (Chapter V)
3	This code is to perform targeting the utility system using the GCC (Chapter VI)
4	To perform an exhaustive search in order to locate the optimum pressure levels using the iterative algorithm with the thermodynamic properties (Chapter VI)
5	To perform an exhaustive search in order to locate the optimum pressure levels using the area on the GCC (Chapter VI)
6	To produce the mass exchange pinch diagram
7	The same as Code 6 but it also consider external lean stream calculations
8	To produce the material recycle pinch diagram
9	To produce the thermal pinch diagram and the grand composite curve
10	The Code of Example 1 in Chapter VIII

Code 1

```
%Refer to Chapter IV of the Dissertation, This Code is used
%to find the optimum steam turbine's inlet conditions for a
%given process heat requirement. In the cost analysis a
%cost index of (470/400) is assumed for 2008, The index
%will affect the cost but not the %optimal selection of
%variables
format short g
%The Heat and pressure requirement of the process
Q=20e6;
P2=100;
Ts2=117.42*P2^0.2289;
hf2=0.0005*Ts2^2+0.7159*Ts2+8.8982;
bestcost=inf;
i=1;
results=zeros(1,7);
T1=0;
%Turbine Specifications
mmax=25000; %Max flowrate
T1max=450; %Max inlet temperature
%Boiler specifications
etab=0.4; %effciency
af=2.8; %cost of fuel, $/MMBtu;
P1max=280; %Max pressure
for P1=P2+1:P1max;
    Ts1=117.42*P1^0.2289;
    for T1=Ts1:T1max;
        hf1=0.0005*Ts1^2+0.7159*Ts1+8.8982;
        As=-0.5549*log(Ts1)+3.7876;
        Bs=0.1001*exp(0.0017*Ts1);
        s1=As*T1^Bs;
        h1=0.2029*Ts1*(s1^3.647)+817.35;
        h2s=0.2029*Ts2*(s1^3.647)+817.35;
        eta1=1;
        eta2=999;
        while abs(eta1-eta2)>0.0001
            h2a=h1-eta1*(h1-h2s);
            m=Q/(h2a-hf2); %lb/hr
            Pt=(h1-h2a)*m;%btu/hr
            A=-0.1518+0.00065*Ts1;
            B=0.961977778+0.000844444*Ts1;
            DHs=h1-h2s;
            etamax=(1/B)*(1-(3.41214e6*A/(DHs*(mmax))));
            eta2=eta1;
            eta1=(6/5)*etamax*(1-(mmax/(6*m)));
        end
        s2=((h2a-817.35)/(0.2029*Ts2))^0.2742;
        As=-0.7918*s2^3+3.4575*s2^2-4.5513*s2+2.1267;
        Bs=710.22*s2^3-3910.6*s2^2+7117.3*s2-3253.5;
        T2=(h2a-Bs)/As;
        if T2<Ts2
            continue
        end
        %calculation parameters
        ae=0.05; %price of electricity in $*KW.hr
        etag=0.4; %the effciency of the electrical
         %generator
         %cost calculations and converstions
         Qb=(h1-hf1)*m; %btu/hr
         Cf=af*Qb*1e-6/etab; %fuel cost ($/hr)
         Fp=1;
         Pt=(h1-h2a)*m; %turbine shaft work (btu/hr)
         Pe=ae*Pt*etag/3413; %power price ($/hr)
         P1g=P1-14.667; %boiler gague pressure (psig)
         nP=7E-4*P1g+0.6;  %The  factor  accounting  for  the
         %pressure in the boiler cost,
         Tsh=T1-Ts1; %the superheat temperature in (F);
         nT=-1.54E-6*Tsh^2+1.3E-3*Tsh+1;     %The     factor
         %accounting  for  the  superheat  temperature  in  the
         %boiler cost,
         Cboiler=3*Qb^0.77*nP*nT; %boiler cost ($)
         Cturbine=475*Pt^0.45;%Turbine cost($)

annualcost=(470/400)*(Cboiler+Cturbine)/10+1.3*Fp*Cf*8000-
Pe*8000;


        if  annualcost<bestcost  &&  T2-Ts2<30  &&  T2-Ts2>5  &&
m<mmax
            bestcost=annualcost;
            best_P=P1;
            results(i,:)=[P1  T1-Ts1  m  T2-Ts2  etamax  eta1
bestcost];
            i=i+1;
            wk1write('newresult.xls',results,1,2);

        end
    end
end
%To output a table with the following columns:
%[inlet_pressure inlet_superheat mass_flow outlet_superheat
%max_effciency best_cost]
results
%To output the grand best cost
Bestcost
```

Code 2

```
%the number of expansion zones
n=input('How many headers do you have?  n=');
```

```
s=input('What is the entropy at the boilers exit?  s=');
Tsat=input('Insert    the    Tsat    vector    Tsat=[Tsat1
Tsat2...Tsatn] Tsat=');
P=input('Insert the pressure vector P=[P1 P2...Pn] P=');
%insert the streams coming into the system with their mass
%flowrates
%(lb/hr)and
Nc=input('How many process streams are input to the utility
Nc=');
if Nc>0
    mc=input('Insert the mass flow rate vector of the input
streams mc=[mc1 mc2..]   mc=');
    Tc=input('Insert  the  Temperature  vector  of  the  input
streams Tc=[Tc1 Tc2..]   Tc=');
    Pc=input('Insert  the  Pressure  vector  of  the  input
streams Pc=[Pc1 Pc2..]   Pc=');
    hin=input('Insert  the  enthalpy  vector  of  the  input
streams hin=[hin1 hin2.. ]  hin=');
end
%The steam required for non-heating purposes
Np=input('How many non-heating process stream requirements
Np=');
if Np>0
    mc=input('Insert the mass flow required by the process
mp=[mp1 mp2..]   mp=');
    Pp=input('Insert the Pressure vector of the non-heating
steam requirements Pp=[Pp1 Pp2..]   Pp=');
end
%Insert the heats (Btu/hr)required by the process
Nh=input('How many hot streams are needed for the process
Nh=');
if Nh>0
    Q=input('Insert  the  heat  Btu/hr  required  for  each
process  Q=');
    Th=input('Insert the minimum temperature (F) for each
stream  Th=');
end
format short g
% The saturation properties at each header
for i=1:n
    h(i)=0.2029*Tsat(i)*s^3.647+817.35;
    hf(i)=0.0005*(Tsat(i))^2+0.7159*Tsat(i)+8.8982;
    hfg(i)=-0.0022*(Tsat(i))^2+0.7319*Tsat(i)+901.69;
    hg(i)=hf(i)+hfg(i);
    sf(i)=0.0013*Tsat(i)+0.052;
    sfg(i)=-0.0024*Tsat(i)+1.9071;
    sg(i)=sf(i)+sfg(i);
end
%Establish the mass in each header
M=zeros(1,n);
m_in=zeros(1,n);
```

```
m_out=zeros(1,n);
%determine the streams to be introduced to each header
Mh_in=zeros(1,n);
for j=1:Nc
    for i=1:n
        if Pc(j)>=P(i)
            m_in(i)=m_in(i)+mc(j);
            Mh_in(i)=Mh_in(i)+mc(j)*hin(j);
            break;
        end
    end
end
mp_out=zeros(1,n);
for j=1:Np
    for i=1:n-1
        if Pp(j)<=P(i)&& Pp(j)>=P(i+1)
            mp_out(i)=mp_out(i)+mp(j);
            break;
        end
    end
end
for i=1:n
    if m_in(i)==0
        h_in(i)=0;
    else
        h_in(i)=Mh_in(i)/m_in(i);
    end
    %Initial mass out
end
h_in
for j=1:Nh
    for i=n:-1:1
        if Th(j)<=Tsat(i)
            mh(j)=Q(j)/(h(j)-hf(j));
            m_out(i)=m_out(i)+mh(j);
            break;
        end
    end
end
mh
m_out
for i=1:n
    for j=i:n
        M(i)=M(i)+(m_out(j)+mp_out(j)-m_in(j));
    end
end
M
Xz=zeros(1,n-1);
% Find the initial mass flow rates
for i=1:n
Ha_in(i)=h(i);
```

```
H_out(i)=(M(i)*Ha_in(i)+h_in(i)*m_in(i))/(M(i)+m_in(i));
end
Ha_in
H_out
m_out
M
eta(1)=1;
Mb=M;
for k=1:20
    for i=1:n
        if i==1
            Ha_in(i)=0.2029*Tsat(i)*s^3.647+817.35;
        else
            S_out(i-1)=((H_out(i-1)-817.35)/(0.2029*Tsat(i-
1)))^0.2724;
            hzsin(i)=0.2029*(Tsat(i-1))*(S_out(i-
1))^3.647+817.35;
            if S_out(i-1)>=sg(i)
                hzsout(i)=0.2029*(Tsat(i))*(S_out(i-
1))^3.647+817.35;
            else
                x=(S_out(i-1)-sf(i))/sfg(i);
                hzsout(i)=hf(i)+x*hfg(i);
            end
            Dh(i)=0.2029*(Tsat(i-1)-Tsat(i))*(S_out(i-
1))^3.647;
            A(i)=-0.1518+0.00065*Tsat(i-1);
            B(i)=0.961977778+0.000844444*Tsat(i-1);
            eta(i)=(1-(3414430*A(i)/(Dh(i)*M(i))))/B(i);
            Ha_in(i)=H_out(i-1)-eta(i)*Dh(i);
            Xz(i-1)=(Ha_in(i)-hf(i))/hfg(i);%the quality at
%the exit of each turbine
        end

H_out(i)=(M(i)*Ha_in(i)+h_in(i)*m_in(i))/(M(i)+m_in(i));
    end
    m_out=zeros(1,n);
    for j=1:Nh
        for i=n:-1:1
            if Th(j)<=Tsat(i)
                mh(j)=Q(j)/(H_out(i)-hf(i));
                m_out(i)=m_out(i)+mh(j);
                break;
            end
        end
    end
    M=zeros(1,n);
    for i=1:n
        for j=i:n
            M(i)=M(i)+(m_out(j)+mp_out(j)-m_in(j));
        end
    end
end
% Output effciencies
eta
%output Mass flowrates
M
%Output the power produced at each zone
for i=2:n %the first zone is excluded (between the boiler
and the first header where power is zero)
    E(i)=M(i)*eta(i)*Dh(i);
end
E
```

Code 3

```
%This code is used to perform targeting using the GCC.
s=input('Enter the entropy at the exit of the highest
pressure main?  s=');
Tsat=input('Insert the saturation temperature at each main
starting from the highest in the format, Tsat=[Tsat1
Tsat2...Tsatn] Tsat=');
Qout=input('Insert the heat to be provided to the process,
Qout=[Qout,1 Qout,2...Qout,n] Qout=');
Qin=input('Insert the heat to be fed from the process to
the mains, Qin=[Qin,1 Qin,2...Qin,n] Qin=');

n=size(Tsat,2);
for i=1:n
    h(i)=0.2029*Tsat(i)*s^3.647+817.35;
    hf(i)=0.0005*(Tsat(i))^2+0.7159*Tsat(i)+8.8982;
    hfg(i)=-0.0022*(Tsat(i))^2+0.7319*Tsat(i)+901.69;
    hg(i)=hf(i)+hfg(i);
    sf(i)=0.0013*Tsat(i)+0.0493;
    sfg(i)=-0.0024*Tsat(i)+1.916;
    sg(i)=sf(i)+sfg(i);
end

%Establish the mass in each header
M=zeros(1,n);
m_in=zeros(1,n);
m_out=zeros(1,n);
%determine the streams to be introduced to each header
Mh_in=zeros(1,n);
for i=1:n
    if Qin(i)>0
        m_in(i)=Qin(i)/(hfg(i));
        h_in(i)=hg(i);
    else
        h_in(i)=0;
    end
```

```
end

%initial isentropic guess for the mass flow to the process
for i=1:n
    m_out(i)=Qout(i)/(h(i)-hf(i));
end

for i=1:n
    for j=i:n
        M(i)=M(i)+(m_out(j)-m_in(j));
    end
end

Xz=zeros(1,n-1);
% Find the initial mass flow rates
for i=1:n
    Ha_in(i)=h(i);

H_out(i)=(M(i)*Ha_in(i)+h_in(i)*m_in(i))/(M(i)+m_in(i));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
eta(1)=1;
Mb=M;
for k=1:20

    for i=1:n
        if i==1
            Ha_in(i)=0.2029*Tsat(i)*s^3.647+817.35;
        else

            S_out(i-1)=((H_out(i-1)-817.35)/(0.2029*Tsat(i-
1)))^0.2724;
            hzsin(i)=0.2029*(Tsat(i-1))*(S_out(i-
1))^3.647+817.35;
            if S_out(i-1)>=sg(i)
                hzsout(i)=0.2029*(Tsat(i))*(S_out(i-
1))^3.647+817.35;
            else
                x=(S_out(i-1)-sf(i))/sfg(i);
                hzsout(i)=hf(i)+x*hfg(i);
            end
            Dh(i)=0.2029*(Tsat(i-1)-Tsat(i))*(S_out(i-
1))^3.647;
            A(i)=-0.1518+0.00065*Tsat(i-1);
            B(i)=0.961977778+0.000844444*Tsat(i-1);
            eta(i)=(1-(3414430*A(i)/(Dh(i)*M(i))))/B(i);
            Ha_in(i)=H_out(i-1)-eta(i)*Dh(i);
            Xz(i-1)=(Ha_in(i)-hf(i))/hfg(i);%the quality at
            %the exit of each turbine
        end
```

```
H_out(i)=(M(i)*Ha_in(i)+h_in(i)*m_in(i))/(M(i)+m_in(i));
    end

    m_out=zeros(1,n);
    for i=1:n

        m_out(i)=Qout(i)/(H_out(i)-hf(i));

    end
    M=zeros(1,n);
    for i=1:n
        for j=i:n
            M(i)=M(i)+(m_out(j)-m_in(j));
        end
    end

end

% Out put efficiencies
eta
%output Mass flowrates
M
%Output the power produced at each zone
for i=2:n %the first zone is excluded (between the boiler
and the first header where power is zero)
    E(i)=M(i)*eta(i)*Dh(i);
end
E
```

Code 4

```
function[]=blevel()
%The input is the matrix from the grand composite curve.
%The first column is the temperature The second column is
%the heat to be transferred per interval The third column
%is the accumulation of heat i.e. the point of Q
%corresponding with the value of T on the GCCC
g=[  260  0    0    ;
     280  20   20   ;
     290  5    25   ;
     300  13   38   ;
     320  4    42   ;
     322  15   57   ;
     330  10   67   ;
     335  15   82   ;
     340  20   102  ];
A=size(g,2);
T=transpose(g(:,1));
Q=1e6*transpose(g(2:A,2));
```

```
format short g;
Tsat=zeros(1,2);
bestTsat=zeros(1,2);
Tp=g(1,1);
Tsat(1)=g(1,A);
Etotalmax=0;
qq=1;
Emax=0;
Mf=zeros(30,6);
N=1;
%The following nested loops are to perform an exhaustive
%search for a four level utility.
for i=Tsat(1):-1:Tp
    Tsat(2)=i;
    for j=Tsat(2):-1:Tp
        Tsat(3)=j;

        for k=Tsat(3):-1:Tp

            Tsat(4)=k;

            QQ=duty(Tsat,T,Q);
            [E,Etotal,ETA,MM]=Epower(QQ,Tsat);
            Mf(N,:)=[Tsat(1,1)     Tsat(1,2)     (Tsat(1,1)-
Tsat(1,2))*QQ(1,2) Etotal MM(1,2) ETA(1,2)];
            if Etotal>Emax
                Emax=Etotal;
                Tmax=Tsat;
            end
            N=N+1;

        end
    end
end
Emax
Tmax
Mf
function [QQ]=duty(Tss,T,Q)
N=size(Tss);n=N(1,2);
Ts=zeros(1,n);
for k=1:n
    Ts(k)=Tss(n-k+1);
end
M=size(T);m=M(1,2);
Qs=zeros(1,n);
for i=1:n
    Qs(i)=0;
    Ts(i);
    for j=1:m

        T(j);
        if Ts(i)>T(j) &&Ts(i)>T(j+1)

            Qs(i)=Qs(i)+ Q(j);
        elseif Ts(i)>T(j) &&Ts(i)<=T(j+1)
            Qs(i)=Qs(i)+ (Q(j)/(T(j+1)-T(j)))*(Ts(i)-T(j));
            break
        end
    end
end
QQs=zeros(1,n);
for k=1:n
    if k==1
        QQs(k)=Qs(k);
    else
        QQs(k)=Qs(k)-Qs(k-1);
    end
end

QQ=zeros(1,n);
for k=1:n
    QQ(k)=QQs(n-k+1);
end

function[E,Etotal,ETA,MM]=Epower(Q,Tsat)
%This function is used to determine the mass flowrates
%required at the boiler and then distributed to the
%different headers according to the heat requirement

n=size(Tsat,2); %the number of headers

%Setup the steam condition at the boiler
T1=1000;
s=(-
0.5549*log(Tsat(1))+3.7876)*T1^(0.1001*exp(0.0017*Tsat(1)))
;%entropy at boiler's exit

%initially, since the exit enthalpy, mass flow and
%efficiency are functions of each other, an iterative
%procedure is followed, the %enthalpy values are initiated
%by assuming isentropic expansion throughout the %utility
%system
for i=1:n
    h(i)=0.2029*Tsat(i)*s^3.647+817.35;
    hf(i)=0.0005*(Tsat(i))^2+0.7159*Tsat(i)+8.8982;
    hfg(i)=-0.0022*(Tsat(i))^2+0.7319*Tsat(i)+901.69;
end

%Establish the mass in each header
M=zeros(1,n);
m_out=zeros(1,n);
```

```
%The initial guess of the mass flowrates
for j=1:n
    m_out(j)=Q(j)/(h(j)-hf(j));
end

%The initial guess of the mass flow rates through each
%expansion zone
for i=1:n
    for j=i:n
        M(i)=M(i)+m_out(j);
    end
end

% The initial enthalpies
for i=1:n
    Ha_in(i)=h(i);
    H_out(i)=Ha_in(i);
end

%Now, the iterative procedure starts with isentropic
%expansion (effciency=1), the mass is corrected and then
%the %effciency is updated and so on
eta(1)=1;
Mb=M;
for k=1:20
    for i=1:n
        if i==1
            Ha_in(i)=0.2029*Tsat(i)*s^3.647+817.35;
        else
            S_out(i-1)=((H_out(i-1)-817.35)/(0.2029*Tsat(i-
1)))^0.2724;
            hzsin(i)=0.2029*(Tsat(i-1))*(S_out(i-
1))^3.647+817.35;
            hzsout(i)=0.2029*(Tsat(i))*(S_out(i-
1))^3.647+817.35;
            Dh(i)=0.2029*(Tsat(i-1)-Tsat(i))*(S_out(i-
1))^3.647;
            A(i)=-0.1518+0.00065*Tsat(i-1);
            B(i)=0.961977778+0.000844444*Tsat(i-1);
            eta(i)=(1-(3414430*A(i)/(Dh(i)*M(i))))/B(i);
            Ha_in(i)=H_out(i-1)-eta(i)*Dh(i);
        end
        H_out(i)=Ha_in(i);
    end

    m_out=zeros(1,n);
  %Correct the mass flow to the headers.
    for j=1:n
        m_out(j)=Q(j)/(H_out(j)-hf(j));
    end
```

```
%Update the mass flow through each expansion zone
    M=zeros(1,n);
    for i=1:n
        for j=i:n
            M(i)=M(i)+m_out(j);
        end
    end
end

%Calculate the resulting power values.
   for i=1:n-1
       E(i)=eta(i+1)*Dh(i+1)*M(i+1);
   end

   Etotal=0;
   for i=1:n-1
       if abs(E(i))==E(i)
           Etotal=Etotal+E(i);
       end
   end
   Etotal;
   ETA=eta;
   MM=M;
```

Code 5

```
%This code works the same as Code 4 except that it uses
%the area to locate the optimal power. The input is the
%matrix from the grand composite curve. The first column is
%the temperature. The second column is the heat to be
%transferred per interval The third column is the
%accumulation of heat i.e. the point of Q corresponding
%with the value of T on the GCCC
g=[  260  0    0    ;
     280  20   20   ;
     290  5    25   ;
     300  13   38   ;
     320  4    42   ;
     322  15   57   ;
     330  10   67   ;
     335  15   82   ;
     340  20   102  ];
A=size(g,2);
T=transpose(g(:,1));
Q=1e6*transpose(g(2:A,2));

format short g;
Tsat=zeros(1,2);
bestTsat=zeros(1,2);
Tp=g(1,1);
Tsat(1)=g(1,A);
```

```
Etotalmax=0;
qq=1;
Emax=0;
Mf=zeros(30,6);
N=1;
%The following nested loops are to perform an exhaustive
%search for a four
%level utility.
for i=Tsat(1):-1:Tp
    Tsat(2)=i;
    for j=Tsat(2):-1:Tp
        Tsat(3)=j;

        for k=Tsat(3):-1:Tp

             [Qsat]=points(g(:,1),g(:,3),Tsat);
                    Area=0;
                    for i=1:size(Tsat,2)-1
                        Area                 =Area+(Tsat(i)-
Tsat(i+1))*Qsat(i+1);
                    end
                    if Area>maxArea
                        maxArea=Area;
                        maxTsat=Tsat;
                    end
        end
    end
end
Emax
Tmax
Mf
function [QQ]=duty(Tss,T,Q)
N=size(Tss);n=N(1,2);
Ts=zeros(1,n);
for k=1:n
    Ts(k)=Tss(n-k+1);
end
M=size(T);m=M(1,2);
Qs=zeros(1,n);
for i=1:n
    Qs(i)=0;
    Ts(i);
    for j=1:m

        T(j);

        if Ts(i)>T(j) &&Ts(i)>T(j+1)

            Qs(i)=Qs(i)+ Q(j);
        elseif Ts(i)>T(j) &&Ts(i)<=T(j+1)
            Qs(i)=Qs(i)+ (Q(j)/(T(j+1)-T(j)))*(Ts(i)-T(j));
            break
        end
    end
end
QQs=zeros(1,n);
for k=1:n
    if k==1
        QQs(k)=Qs(k);
    else
        QQs(k)=Qs(k)-Qs(k-1);
    end
end

QQ=zeros(1,n);
for k=1:n
    QQ(k)=QQs(n-k+1);
end
```

Code 6

```
%This function is used to establish the Mass exchange pinch
%diagram, the input arguments are two matrices, the Rich
%stream Matrix (Rd) consists of n rows that represents the
%number of streams and three columns representing flowrate,
%supply composition and Target composition respectively.
%The other matrix is the Lean stream matrix that is
%composed of m rows representing the number of lean streams
%and 6 columns representing flowrate, supply composition,
%target composition, m, epsilon and b, respectively. The
%last three are the equilibrium equation %parameters that
%are used to fit the lean stream compositions on the rich
%stream scale.
function[Nr,Ns,Yp,ExCapacity,ExternalMSA]=exmass(Rd,Sd)The
first step is define the lean stream loads and transform
the composition to fit on the rich stream scale.
Ls=zeros(size(Sd,1),1);
ys=zeros(size(Sd,1),2);
Ms=zeros(size(Sd,1),3);
for i=1:size(Sd,1)
    Ls(i)=Sd(i,1)*(Sd(i,3)-Sd(i,2));

ys(i,:)=[(Sd(i,2)+Sd(i,5))*Sd(i,4)+Sd(i,6),(Sd(i,3)+Sd(i,5)
)*Sd(i,4)+Sd(i,6)];
    Ms(i,:)=[Ls(i)/(ys(i,2)-ys(i,1))  ys(i,:)];
end

%Arrange the streams by segments to be more easily
%manipulated
```

```
[gr,gs]=curve(Rd,Ms);
%The same range of compositions is used for the
%convenience of plotting the pinch diagram and determining
%the External MSA needed %and the excess capacity of the
%lean streams.
m=size(gr,1);
n=size(gs,1);
Ymin=min(gr(1,1),gs(1,1));
Ymax=max(gr(m,1),gs(n,1));
if Ymin<gr(1,1)
    gr=[Ymin 0;gr];
end
if Ymin<gs(1,1)
    gs=[Ymin 0;gs];
end
m=size(gr,1);
n=size(gs,1);
if Ymax>gr(m,1)
    gr=[gr; Ymax 0];
end
if Ymax>gs(n,1)
    gs=[gs; Ymax 0];
end

% “Drawing the lines”
%This subroutine "Stream" is used to connect the points in
%the different segments and keep them in one continuous
%line
qs=stream(gs);
qr=stream(gr);

%For convenience, the composition vector is defined
%separately from the input matrices
Yr=gr(:,1);Ys=gs(:,1);
%This subroutine "pinch" is used to determine the pinch
%point and match the rich and lean streams on the pinch
%diagram
[Ns,Nr,Yp,ExCapacity,ExternalMSA]=pinch(Yr,Ys,qr,qs);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to connect the points of the given
%segments
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[q]=stream(g)
%Given is a matrix of two columns, the left column is the
%composition and the right one is the Load required for
%each two consecutive compositions in the first column, of
%course the first load required will be zero
n=size(g,1);
q=zeros(n,1);
q(1)=0;
for i=2:n
```

```
    q(i)=q(i-1)+g(i,2);
end
%%%%%%%%%%%%%%%%%%%%
%This function is used to establish the load requirement
%per composition segment in order to establish the heat
%pinch diagram

function[gr,gs]=curve(Rd,Sd)
format short g
%For the Rich stream
%Establish the composition column that has all possible
%values
Concenr=[Rd(:,2);Rd(:,3)];
%Eliminate the repeated Composition values
Concenr=sortrows(Concenr,1);
j=2;
while (j<=size(Concenr,1))
    if Concenr(j,1)==Concenr(j-1,1)
        Concenr(j,:)=[];
    else
        j=j+1;
    end
end
%Establish the column needed to plot the line on the pinch
%diagram
gr=zeros(size(Concenr,1),2);
gr(1,:)=[Concenr(1) 0];
for i=2:size(Concenr,1)
    for j=1:size(Rd,1)
        if Rd(j,2)>=Concenr(i)&& Rd(j,3)<=Concenr(i-1)
            gr(i,2)=gr(i,2)+  Rd(j,1)*(Concenr(i)-Concenr(i-
1));
        end
    end
    gr(i,:)=[Concenr(i) gr(i,2)];
end

%%%%%%%%%%%%%%%%%%
%The same is done for the lean stream
%Establish the composition column that has all possible
%values
Concenc=[Sd(:,2);Sd(:,3)];
%Eliminate the repeated composition values
Concenc=sortrows(Concenc,1);
j=2;
while (j<=size(Concenc,1))
    if Concenc(j,1)==Concenc(j-1,1)
        Concenc(j,:)=[];
    else
        j=j+1;
    end
```

```
end
%Establish the column needed to plot the line on the pinch
%diagram
gs=zeros(size(Concenc,1),2);
gs(1,:)=[Concenc(1) 0];
for i=2:size(Concenc,1)
    for j=1:size(Sd,1)
        if Sd(j,2)<=Concenc(i-1)&& Sd(j,3)>=Concenc(i)
            gs(i,2)=gs(i,2)+Sd(j,1)*(Concenc(i)-Concenc(i-
1));
        end
    end
    gs(i,:)=[Concenc(i) gs(i,2)];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to determine the pinch point and put
%the rich and lean streams in a plottable format on the
%pinch diagram
%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[Ns,Nr,Yp,ExCapacity,ExternalMSA]=pinch(Yr,Ys,qr,qs
,DT)
h=size(qr,1);
c=size(qs,1);
%The basic idea of determining the pinch point is to plot
%both streams starting from L=0 on the vertical axis, then
%we shift the lean stream to a position that touches the
%rich stream at one single point. Firstly, both streams are
%plotted starting from L=0 (on the horizontal axis), then
%each composition value in Yr and Ys vectors will have the
%corresponding q value on both streams (because the pinch
%point must correspond to at least one value of Y in Yr
%and/or Ys), Then at each Y, we find the corresponding q in
%the Lean stream minus the corresponding q in rich stream,
%the shift will be the maximum difference.

%Find qs in the cold stream that corresponds with the Y in
%the Yr vector

for i=1:h
    for j=2:c
        if Yr(i)>=Ys(j-1)&& Yr(i)<=Ys(j)
            qs2h(i,:)=[Yr(i)   ((qs(j)-qs(j-1))/(Ys(j)-Ys(j-
1)+eps))*Yr(i)+(qs(j)-Ys(j)*((qs(j)-qs(j-1))/(Ys(j)-Ys(j-
1)+eps)))];
       break
        end
    end
end
%Find qr in the rich stream that corresponds with the Y in
%the Ys vector
for i=1:c
```

```
    for j=2:h
        if Ys(i)>=Yr(j-1)&& Ys(i)<=Yr(j)
            qr2c(i,:)=[Ys(i)   ((qr(j)-qr(j-1))/(Yr(j)-Yr(j-
1)+eps))*Ys(i)+(qr(j)-Yr(j)*((qr(j)-qr(j-1))/(Yr(j)-Yr(j-
1)+eps)))];
            break
        end
    end
end

Nh=[qs2h(:,1) qr qs2h(:,2) ];
Nc=[qr2c(:,1) qr2c(:,2) qs];


%Establish a matrix that will include the difference in q
%of both streams at each composition value in Ys and Yr
%vectors
Diff=zeros(h+c,2);
for i=1:h
    Diff(i,:)=[Yr(i), qs2h(i,2)-qr(i)];
end

for j=1:c
    Diff(j+h,:)=[Ys(j), qs(j)-qr2c(j,2)];
end

%sort the rows of the difference matrix to determine the
%pinch point,
mDiff= sortrows(Diff,2);
inc=abs(mDiff(1,2)); %The increment required to shift the
%cold stream to the right and hence to establish the pinch
%point
%"The pinch point"
Yp=mDiff(1,1);
%Shifting the cold stream
for i=1:c
    qs(i)=qs(i)+inc;
end
%These two matrices redefine the hot and cold stream on the
%pinch diagram
Nr=[Yr qr];
Ns=[Ys qs];
if Nr(1,2)==Nr(2,2)
    Nr(1,:)=[];
end
if Ns(1,2)==Ns(2,2)
    Ns(1,:)=[];
end
if Nr(size(Nr,1),2)==Nr(size(Nr,1)-1,2)
    Nr(size(Nr,1),:)=[];
end
```

```
if Ns(size(Ns,1),2)==Ns(size(Ns,1)-1,2)
    Ns(size(Ns,1),:)=[];
end
%display of the basic information, the pinch point, the
%excess capacity of the process lean stream and the
%external MSA required The excess capacity and the external
%MSA, respectively...
ExCapacity=Ns(size(Ns,1),2)-Nr(size(Nr,1),2);
ExternalMSA=Ns(1,2)-Nr(1,2);
plot(Nr(:,1),Nr(:,2),'-r',Ns(:,1),Ns(:,2),'-
b','LineWidth',2)
title('Mass Exchange Pinch Diagram','FontSize',15)
xlabel('Composition (Y)','FontSize',13)
ylabel('Mass Exchanged (M)','FontSize',13)
legend('Rich Stream','Lean Stream',2);
grid on
xx=xlim;
yy=ylim;
text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+3*(yy(2)-
yy(1))/20,['Yp=', num2str(Yp)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+2*(yy(2)      yy(1))/20,
['Lean Excess Cap.=', num2str(ExCapacity)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+(yy(2)-yy(1))/20,
['External MSA req.=', num2str(ExternalMSA)],'FontSize',9)
```

Code 7

```
%this function is used to establish the Mass exchange pinch
%diagram, the input arguments are two matrices, the Rich
%stream Matrix (Rd) consists of n rows that represents the
%number of streams and three columns representing flowrate,
%supply composition and Target composition respectively.
%The other matrix is the Lean stream matrix that is
%composed of m rows representing the number of lean streams
%and 6 columns representing flowrate, supply composition,
%target composition, m, epsilon and b, respectively. The
%last three are the equilibrium equation parameters that
%are used to fit the lean stream compositions on the rich
%stream scale.

function[Nr,Ns,Yp,ExCapacity,ExternalMSA]=external(Rd,Sd,Ex
s)
%The first step is define the lean stream loads and
%transform the composition to fit on the rich stream scale.
Ls=zeros(size(Sd,1),1);
ys=zeros(size(Sd,1),2);
Ms=zeros(size(Sd,1),3);
for i=1:size(Sd,1)
    Ls(i)=Sd(i,1)*(Sd(i,3)-Sd(i,2));
```

```
ys(i,:)=[(Sd(i,2)+Sd(i,5))*Sd(i,4)+Sd(i,6),(Sd(i,3)+Sd(i,5)
)*Sd(i,4)+Sd(i,6)];
    Ms(i,:)=[(Ls(i)+eps)/(ys(i,2)-ys(i,1)+eps)  ys(i,:)];
end
%Arrange the streams by segments to be more easily
%manipulated
[gr,gs]=curve(Rd,Ms);
%The same range of compositions is used for the
%convenience of plotting the pinch diagram and determining
%the External MSA needed and the excess capacity of the
%lean streams.
m=size(gr,1);
n=size(gs,1);
Ymin=min(gr(1,1),gs(1,1));
Ymax=max(gr(m,1),gs(n,1));
if Ymin<gr(1,1)
    gr=[Ymin 0;gr];
end
if Ymin<gs(1,1)
    gs=[Ymin 0;gs];
end
m=size(gr,1);
n=size(gs,1);
if Ymax>gr(m,1)
    gr=[gr; Ymax 0];
end
if Ymax>gs(n,1)
    gs=[gs; Ymax 0];
end

% Drawing the lines
%This subroutine "Stream" is used to connect the points in
%the different segments and keep them in one continuous
%line
qs=stream(gs);
qr=stream(gr);

%For convenience, the composition vector is defined
%separately from the input matrices
Yr=gr(:,1);Ys=gs(:,1);
%This subroutine "pinch" is used to determine the pinch
point and match
%the rich and lean streams on the pinch diagram
[Ns,Nr,Yp,ExCapacity,ExternalMSA]=pinch(Yr,Ys,qr,qs);
if Sd==[0 0 0 0 0 0]
    Ns=[];
    FY=Nr(size(Nr,1),1);
    FL=Nr(size(Nr,1),2);
    Next=[];
else
```

```
    FY=Ns(1,1);
    FL=Ns(1,2);
    Next=Ns(1,:);
end
%Using the External MSA's

Ex=zeros(size(Exs,1),3);
%Establishing the Y values using the equilibrium data
for i=1:size(Exs,1)
    Ex(i,:)=[(Exs(i,1)+Exs(i,4))*Exs(i,3)+Exs(i,5),
(Exs(i,2)+Exs(i,4))*Exs(i,3)+Exs(i,5),
(Exs(i,6)+eps)/(Exs(i,2)-Exs(i,1)+eps)];
end
Exs=[Exs Ex];
j=1;
%Delete the unfeasibile external stream
while (j<=size(Exs,1))
    if Exs(j,7)>=FY;
         Exs(j,:)=[];
    else
        j=j+1;
    end
end

%initiate the cost
cost=0;
%Using external streams will continue as long as we have
%feasible streams and a portion of the rich streams that
%need to be covered

while(FL~=0 && size(Exs,1)~=0)
 Exs=sortrows(Exs,9);
 %selecting the suitable target value of the external
stream
        if (FY-Exs(1,7))<(Exs(1,8)-Exs(1,7))
            nys=Exs(1,7);
            nyt=FY;
        else
            nys=Exs(1,7);
            nyt=Exs(1,8);
        end
     %selecting the appropriate Load value for the external
stream composition
    if nys<=Nr(1,1)

        nLs=0;
    else
        for i=2:size(Nr,1)
            if nys>Nr(i-1,1)&&nys<=Nr(i,1)
```

```
                nLs=((Nr(i,2)-Nr(i-1,2)+eps)/(Nr(i,1)-Nr(i-
1,1)+eps))*nys+(Nr(i,2)-((Nr(i,2)-Nr(i-1,2)+eps)/(Nr(i,1)-
Nr(i-1,1)+eps))*Nr(i,1));
                break;
            end
        end
    end
     % calculate the cost
    addedcost=(FL-nLs)*Exs(1,9);
    cost=cost+addedcost;

    %Redefine the lowest point on the lean stream
    Next=[Next;nys nLs;nyt FL];
    Next=sortrows(Next,1);
    FY=Next(1,1);
    FL=Next(1,2);

    %Remove the no longer feasible streams

    j=1;
    while (j<=size(Exs,1))
        if Exs(j,7)>=FY;
            Exs(j,:)=[];
        else
            j=j+1;
        end
    end

end
%Delete the repeated rows, the repetition results from
%adding the last point again to the matrix, if we removed
%it then it will be hard to connect the lines when no
%process stream exists.
j=2;
while (j<=size(Next,1))
    if Next(j,1)==Next(j-1,1)
        Next=[];
    else
        j=j+1;
    end
end
figure(1)
if size (Next,1)==0
    disp('*There is no external lean streams provided or
none of the given is feasible!*');
elseif size(Ns,1)==0
    Externalused=Next(size(Next,1),2)-Next(1,2);
    plot(Nr(:,1),Nr(:,2),'-r',Next(:,1),Next(:,2),'-
g','LineWidth',2)
    title('Mass Exchange Pinch Diagram','FontSize',15)
    xlabel('Composition (Y)','FontSize',13)
    ylabel('Mass Exchanged (M)','FontSize',13)
    legend('Rich Stream','External Lean',2);
    grid on
    xx=xlim;
    yy=ylim;
    text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+2*(yy(2)-
yy(1))/20,['cost/unit time=', num2str(cost)],'FontSize',9)
    text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+(yy(2)-
yy(1))/20,['External MSA used=',
num2str(Externalused)],'FontSize',9)

else

    Externalused=Next(size(Next,1),2)-Next(1,2);
    plot(Nr(:,1),Nr(:,2),'-r',Ns(:,1),Ns(:,2),'-
b',Next(:,1),Next(:,2),'-g','LineWidth',2)
    title('Mass Exchange Pinch Diagram','FontSize',15)
    xlabel('Composition (Y)','FontSize',13)
    ylabel('Mass Exchanged (M)','FontSize',13)
    legend('Rich Stream','Process Lean','External Lean',2);
    grid on
    xx=xlim;
    yy=ylim;

    text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+2*(yy(2)-
yy(1))/20,['cost/unit time=', num2str(cost)],'FontSize',9)
    text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+(yy(2)-
yy(1))/20,['External MSA used=',
num2str(Externalused)],'FontSize',9)
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to connect the points of the given
%segments
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[q]=stream(g)
%Given is a matrix of two columns, the left column is the
%composition and the right one is the Load required for
%each two consecutive compositions in the first column, of
%course the first load required will be zero
n=size(g,1);
q=zeros(n,1);
q(1)=0;
for i=2:n
    q(i)=q(i-1)+g(i,2);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to establish the load requirement
%per composition segment in order to establish the heat
%pinch diagram
```

```
function[gr,gs]=curve(Rd,Sd)
format short g
%For the Rich stream
%Establish the composition column that has all possible
%values
Concenr=[Rd(:,2);Rd(:,3)];
%Eliminate the repeated Composition values
Concenr=sortrows(Concenr,1);
j=2;
while (j<=size(Concenr,1))
    if Concenr(j,1)==Concenr(j-1,1)
        Concenr(j,:)=[];
    else
        j=j+1;
    end
end
%Establish the column needed to plot the line on the pinch
diagram
gr=zeros(size(Concenr,1),2);
gr(1,:)=[Concenr(1) 0];
for i=2:size(Concenr,1)
    for j=1:size(Rd,1)
        if Rd(j,2)>=Concenr(i)&& Rd(j,3)<=Concenr(i-1)
            gr(i,2)=gr(i,2)+ Rd(j,1)*(Concenr(i)-Concenr(i-
1));
        end
    end
    gr(i,:)=[Concenr(i) gr(i,2)];
end

%%%%%%%%%%%%%%%%
%The same is done for the lean stream
%Establish the composition column that has all possible
%values
Concenc=[Sd(:,2);Sd(:,3)];
%Eliminate the repeated composition values
Concenc=sortrows(Concenc,1);
j=2;
while (j<=size(Concenc,1))
    if Concenc(j,1)==Concenc(j-1,1)
        Concenc(j,:)=[];
    else
        j=j+1;
    end
end
%Establish the column needed to plot the line on the pinch
%diagram
gs=zeros(size(Concenc,1),2);
gs(1,:)=[Concenc(1) 0];
for i=2:size(Concenc,1)
```

```
    for j=1:size(Sd,1)
        if Sd(j,2)<=Concenc(i-1)&& Sd(j,3)>=Concenc(i)
            gs(i,2)=gs(i,2)+Sd(j,1)*(Concenc(i)-Concenc(i-
1));
        end
    end
    gs(i,:)=[Concenc(i) gs(i,2)];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to determine the pinch point and put
%the rich and lean streams in a plottable format on the
%pinch diagram
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[Ns,Nr,Yp,ExCapacity,ExternalMSA]=pinch(Yr,Ys,qr,qs
,DT)
r=size(qr,1);
s=size(qs,1);
%The basic idea of determining the pinch point is to plot
%both streams starting from L=0 on the vertical axis, then
%we shift the lean stream to a position that touches the
%rich stream at one single point. Firstly, both streams are
%plotted starting from L=0 (on the horizontal axis), then
%each composition value in Yr and Ys vectors will have the
%corresponding q value on both streams (because the pinch
%point must correspond to at least one value of Y in Yr
%and/or Ys), Then at each Y, we find the corresponding q in
%the Lean stream minus the corresponding q in rich stream,
%the shift will be the maximum difference.

%Find qs in the cold stream that corresponds with the Y in
%the Yr vector

for i=1:r
    for j=2:s
        if Yr(i)>=Ys(j-1)&& Yr(i)<=Ys(j)
            qs2r(i,:)=[Yr(i)   ((qs(j)-qs(j-1))/(Ys(j)-Ys(j-
1)+eps))*Yr(i)+(qs(j)-Ys(j)*((qs(j)-qs(j-1))/(Ys(j)-Ys(j-
1)+eps)))];
       break
        end
    end
end
%Find qr in the rich stream that corresponds with the Y in
the Ys vector
for i=1:s
    for j=2:r
        if Ys(i)>=Yr(j-1)&& Ys(i)<=Yr(j)
            qr2s(i,:)=[Ys(i)   ((qr(j)-qr(j-1))/(Yr(j)-Yr(j-
1)+eps))*Ys(i)+(qr(j)-Yr(j)*((qr(j)-qr(j-1))/(Yr(j)-Yr(j-
1)+eps)))];
            break
        end
    end
end

Nh=[qs2r(:,1) qr qs2r(:,2) ];
Nc=[qr2s(:,1) qr2s(:,2) qs];


%Establish a matrix that will include the difference in q
%of both streams at each composition value in Ys and Yr
%vectors
Diff=zeros(r+s,2);
for i=1:r
    Diff(i,:)=[Yr(i), qs2r(i,2)-qr(i)];
end

for j=1:s
    Diff(j+r,:)=[Ys(j), qs(j)-qr2s(j,2)];
end

%sort the rows of the difference matrix to determine the
%pinch point,
mDiff= sortrows(Diff,2);
inc=abs(mDiff(1,2)); %The increment required to shift the
%cold stream to the right and hence to establish the pinch
%point
%The pinch point Y value
Yp=mDiff(1,1);
%Shifting the cold stream
for i=1:s
    qs(i)=qs(i)+inc;
end
%These two matrices redefine the hot and cold stream on the
%pinch diagram
Nr=[Yr qr];
Ns=[Ys qs];
%unfortunately, even if we have no lean stream, we will
%still have one
%point in the lean stream matrix. This point results from
%the epsilon we added in the equation matching formula and
%so this point will always be close to the zero load where
%the Rich stream starts, and to avoid Ns-based calculation
%errors, we redefine
%Ns as an empty matrix.
if size(Ns,1)<2
    Ns=[];
else
    %to find the pinch point
    MM=[Nr;Ns];
    for i=1:size(MM,1)
        if MM(i,1)==Yp
```

```
            Lp=MM(i,2);
        end
    end
end

if Nr(1,2)==Nr(2,2)
    Nr(1,:)=[];
end
if Nr(size(Nr,1),2)==Nr(size(Nr,1)-1,2)
    Nr(size(Nr,1),:)=[];
end
if size(Ns,1)>1
    if size(Ns,1)>1 && (Ns(1,2)==Ns(2,2))
        Ns(1,:)=[];
    end

    if size(Ns,1)>1 && (Ns(size(Ns,1),2)==Ns(size(Ns,1)-
1,2))
        Ns(size(Ns,1),:)=[];
    end
end


%display of the basic information, the pinch point, the
%excess capacity of the process lean stream and the
%external MSA required  The excess capacity and the
%external MSA, respectively...
ExCapacity=Ns(size(Ns,1),2)-Nr(size(Nr,1),2);
ExternalMSA=Ns(1,2)-Nr(1,2);
figure(2)
plot(Nr(:,1),Nr(:,2),'-r',Ns(:,1),Ns(:,2),'-
b','LineWidth',2)
title('Mass Exchange Pinch Diagram','FontSize',15)
xlabel('Composition (Y)','FontSize',13)
ylabel('Mass Exchanged (M)','FontSize',13)
legend('Rich Stream','Lean Stream',2);
grid on
xx=xlim;
yy=ylim;

if size(Ns,1)<2;
    text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+3*(yy(2)-
yy(1))/20,['No Pinch Point Available'],'FontSize',9)

else
    text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+3*(yy(2)-
yy(1))/20,['Pinch point=(',
num2str(Yp),',',num2str(Lp),')'],'FontSize',9)
end
```

```
text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+2*(yy(2)-
yy(1))/20,['Lean Excess Cap.=',
num2str(ExCapacity)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.7,yy(1)+(yy(2)-
yy(1))/20,['External MSA req.=',
num2str(ExternalMSA)],'FontSize',9)
```

Code 8

```
function[Mr,Mk,Lp,minFresh,minWaste]=matrecm(sink,source)
format short g
%Arrange the sink and source input arguments based
%ascendingly based on the maximum inlet mass fraction
sink=sortrows(sink,2);
source=sortrows(source,2);
%Prepare the two matrices that will hold the sink and
%source points on the graph.

gk=zeros(size(sink,1)+1,2);
gr=zeros(size(source,1)+1,2);
m=size(gk,1);
n=size(gr,1);
for i=2:m;

    gk(i,:)=[gk(i-1,1)+sink(i-1,1)*sink(i-1,2), sink(i-
1,1)];
end

for i=2:n;
    gr(i,:)=[gr(i-1,1)+source(i-1,1)*source(i-1,2),
source(i-1,1)];
end
%The same range of Loads is used for the convenience of
%plotting the pinch diagram and determining the minimum
%required utilities

Lmin=min(gk(1,1),gr(1,1));
Lmax=max(gk(m,1),gr(n,1));
if Lmin<gk(1,1)
    gk=[Lmin 0;gk];
end
if Lmin<gr(1,1)
    gr=[Lmin 0;gr];
end
m=size(gk,1);
n=size(gr,1);
if Lmax>gk(m,1)
    gk=[gk; Lmax 0];
end
if Lmax>gr(n,1)
    gr=[gr; Lmax 0];
```

```
end
%This subroutine "Stream" is used to connect the points in
the different
%segments and keep them in one continuous line
qr=stream(gr);
qk=stream(gk);

%For conveince, the Load vector is defined separately from
%the input matrices
Lk=gk(:,1);Lr=gr(:,1);
%This subroutine "pinch" is used to determine the pinch
%point and redefines the sink and source streams point on
%the pinch diagram

[Mr,Mk,Lp,minFresh,minWaste]=pinch(Lk,Lr,qk,qr);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to connect the points of the given
%segments
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[q]=stream(g)
%Given is a matrix of two columns, the left column is the
%Load and the right one is the flowrate associated with the
%difference of each two consecutive loads in the first
%column, of course the first flowrate will be zero
n=size(g,1);
q=zeros(n,1);
q(1)=0;
for i=2:n
    q(i)=q(i-1)+g(i,2);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to determine the pinch point and put
%the hot and cold streams line in a plottable format on the
%pinch diagram
%%%%%%%%%%%%%%%%%%%%%%%
function[Mr,Mk,Lp,minFresh,minWaste]=pinch(Lk,Lr,qk,qr)
k=size(qk,1);
r=size(qr,1);
%The basic idea of determining the pinch point is to plot
%both streams starting from Flowrate=0, then we shift the
%source stream to the left to a position that touches the
%hot stream at one single point. Firstly, both streams are
%plotted starting from flowrate (q)=0 (on the horizontal
%axes), then each Load value in Lk and Lr vectors will have
%the corresponding q value on both streams (because the
%pinch point must correspond to at least one value of L in
%Lk and/or Lr), Then at each L, we find the corresponding q
%in the source stream minus the corresponding q in sink
%stream, the shift will be the maximum difference. Find qr
```

```
%in the source stream that corresponds with the L in the Lk
%vector


for i=1:k
    for j=2:r
        if Lk(i)>=Lr(j-1)&& Lk(i)<=Lr(j)
            qr2k(i,:)=[Lk(i)   ((qr(j)-qr(j-1))/(Lr(j)-Lr(j-
1)+eps))*Lk(i)+(qr(j)-Lr(j)*((qr(j)-qr(j-1))/(Lr(j)-Lr(j-
1)+eps)))];
       break
        end
    end
end
%Find qk in the hot stream that corresponds with the T in
the Lr vector
for i=1:r
    for j=2:k
        if Lr(i)>=Lk(j-1)&& Lr(i)<=Lk(j)
            qk2r(i,:)=[Lr(i)   ((qk(j)-qk(j-1))/(Lk(j)-Lk(j-
1)+eps))*Lr(i)+(qk(j)-Lk(j)*((qk(j)-qk(j-1))/(Lk(j)-Lk(j-
1)+eps)))];
            break
        end
    end
end

Nh=[qr2k(:,1) qk qr2k(:,2) ];
Nc=[qk2r(:,1) qk2r(:,2) qr];


%Establish a matrix that will include the difference in q
%of both streams at each Temperature value in Lr and Lk
%vectors
Diff=zeros(k+r,2);
for i=1:k
    Diff(i,:)=[Lk(i), qr2k(i,2)-qk(i)];
end

for j=1:r
    Diff(j+k,:)=[Lr(j), qr(j)-qk2r(j,2)];
end
%sort the rows of the difference matrix to determine the
pinch point,
mDiff= sortrows(Diff,2);
inc=abs(mDiff(1,2)); %The increment required to shift the
%source stream to the right and hence to establish the
%pinch point
%The pinch point
Lp=mDiff(1,1);
%Shifting the source stream
```

```
for i=1:r
    qr(i)=qr(i)+inc;
end
%These two matrices redefine the sink and source stream on
%the pinch diagram
Mk=[Lk qk];
Mr=[Lr qr];
if Mk(1,2)==Mk(2,2)
    Mk(1,:)=[];
end
if Mr(1,2)==Mr(2,2)
    Mr(1,:)=[];
end
if Mk(size(Mk,1),2)==Mk(size(Mk,1)-1,2)
    Mk(size(Mk,1),:)=[];
end
if Mr(size(Mr,1),2)==Mr(size(Mr,1)-1,2)
    Mr(size(Mr,1),:)=[];
end
%display of the basic information, the pinch point, the
%minimum Fresh and the minimum waste.  The minimum utility
%requirements
minFresh=Mr(1,2)-Mk(1,2);
minWaste=Mr(size(Mr,1),2)-Mk(size(Mk,1),2);
plot(Mk(:,2),Mk(:,1),'-g',Mr(:,2),Mr(:,1),'-
m','LineWidth',2)
title('Material Recycle Pinch Diagram','FontSize',14)
xlabel('Flowrate (Q)','FontSize',13)
ylabel('Load (L)','FontSize',13)
legend('Sink','Source',2);
grid on
xx=xlim;
yy=ylim;
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+3*(yy(2)-
yy(1))/20,['Load at Pinch=', num2str(Lp)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+2*(yy(2)-yy(1))/20,['min
Fresh=', num2str(minFresh)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+(yy(2)-yy(1))/20,['min
Waste=', num2str(minWaste)],'FontSize',9)
```

Code 9

```
function[Mh,Mc,cascade,Tp,minHeat,minCool]=heatinteg(Hh,Hc,
DT)

%Arrange the streams by segments to be more easily
%manipulated
[gh,gc]=curve(Hh,Hc);
% The same range of Temperatures is used for the
%convenience of plotting the pinch diagram and determining
%the minimum required utilities
```

```
m=size(gh,1);
n=size(gc,1);
%First of all, we need to adjust the Cold stream
temperature by adding DT
%so aas to have all the calculation on the hot temperature
scale.
for i=1:size(gc,1)
    gc(i,1)=gc(i,1)+DT;
end
Tmin=min(gh(1,1),gc(1,1));
Tmax=max(gh(m,1),gc(n,1));
if Tmin<gh(1,1)
    gh=[Tmin 0;gh];
end
if Tmin<gc(1,1)
    gc=[Tmin 0;gc];
end
m=size(gh,1);
n=size(gc,1);
if Tmax>gh(m,1)
    gh=[gh; Tmax 0];
end
if Tmax>gc(n,1)
    gc=[gc; Tmax 0];
end

% Drawing the lines This subroutine "Stream" is used to
%connect the points in the different segments and keep them
%in one continuous line
qc=stream(gc);
qh=stream(gh);

%For conveince, the temperature vector is defined seprately
%from the input matrices
Th=gh(:,1);Tc=gc(:,1);
%This subroutine "pinch" is used to determine the pinch
%point and redefines the hot and cold streams point on the
%pinch diagram
[Mc,Mh,cascade,Tp,minHeat,minCool]=pinch(Th,Tc,qh,qc,DT);
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to connect the points of the given
%segments
%%%%%%%%%%%%%%%%%%%%%%%%%%
function[q]=stream(g)
%Given is a matrix of two columns, the left column is the
%temperature and the right one is the heat required for
%each two consecutive temperatures in the first column, of
%course the first heat required will be zero
n=size(g,1);
q=zeros(n,1);
q(1)=0;
```

```
for i=2:n
    q(i)=q(i-1)+g(i,2);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to establish the heat requirement
%per temperature segment in order to establish the heat
%pinch diagram
function[gh,gc]=curve(Hh,Hc)
format short g
%For the hot stream
%Establish the temperature column that has all possible
%values
Temph=[Hh(:,2);Hh(:,3)];
%Eliminate the repeated Temperature values
Temph=sortrows(Temph,1);
j=2;
while (j<=size(Temph,1))
    if Temph(j,1)==Temph(j-1,1)
        Temph(j,:)=[];
    else
        j=j+1;
    end
end
%Establish the column needed to plot the line on the pinch
%diagram
gh=zeros(size(Temph,1),2);
gh(1,:)=[Temph(1) 0];
for i=2:size(Temph,1)
    for j=1:size(Hh,1)
        if Hh(j,2)>=Temph(i)&& Hh(j,3)<=Temph(i-1)
            gh(i,2)=gh(i,2)+ Hh(j,1)*(Temph(i)-Temph(i-1));
        end
    end
    gh(i,:)=[Temph(i) gh(i,2)];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
%The same is done for the cold stream Establish the
%temperature column that has all possible values
Tempc=[Hc(:,2);Hc(:,3)];
%Eliminate the repeated Temperature values
Tempc=sortrows(Tempc,1);
j=2;
while (j<=size(Tempc,1))
    if Tempc(j,1)==Tempc(j-1,1)
        Tempc(j,:)=[];
    else
        j=j+1;
    end
end
```

```
%Establish the column needed to plot the line on the pinch
diagram
gc=zeros(size(Tempc,1),2);
gc(1,:)=[Tempc(1) 0];
for i=2:size(Tempc,1)
    for j=1:size(Hc,1)
        if Hc(j,2)<=Tempc(i-1)&& Hc(j,3)>=Tempc(i)
            gc(i,2)=gc(i,2)+Hc(j,1)*(Tempc(i)-Tempc(i-1));
        end
    end
    gc(i,:)=[Tempc(i) gc(i,2)];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function is used to determine the pinch point and put
%the hot and cold streams line in a plottable format on the
%pinch diagram
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[Mc,Mh,cascade,Tp,minHeat,minCool]=pinch(Th,Tc,qh,q
c,DT)
h=size(qh,1);
c=size(qc,1);
%The basic idea of determining the pinch point is to plot
%both streams starting from Q=0, then we shift the cold
%stream to the right to a position that touches the hot
%stream at one single point. Firstly, both streams are
%plotted starting from q=0 (on the horizontal axes), then
%each temperature value in Th and Tc vectors will have the
%corresponding q value on both streams (because the pinch
%point must correspond to at least one value of T in Th
%and/or Tc), Then at each T, we find the corresponding q in
%the cold stream minus the corresponding q in hot stream,
%the shift will be the maximum difference. Find qc in the
%cold stream that corresponds with the T in the Th vector

%To plot on the same graph, we need to adjust the cold
%temperatures by adding DT to be plotted on the hot
%temperature scale
z=size(Tc,1);


for i=1:h
    for j=2:c
        if Th(i)>=Tc(j-1)&& Th(i)<=Tc(j)
            qc2h(i,:)=[Th(i)   ((qc(j)-qc(j-1))/(Tc(j)-Tc(j-
1)+eps))*Th(i)+(qc(j)-Tc(j)*((qc(j)-qc(j-1))/(Tc(j)-Tc(j-
1)+eps)))];
       break
        end
    end
end
```

```
%Find qh in the hot stream that corresponds with the T in
the Tc vector
for i=1:c
    for j=2:h
        if Tc(i)>=Th(j-1)&& Tc(i)<=Th(j)
            qh2c(i,:)=[Tc(i)   ((qh(j)-qh(j-1))/(Th(j)-Th(j-
1)+eps))*Tc(i)+(qh(j)-Th(j)*((qh(j)-qh(j-1))/(Th(j)-Th(j-
1)+eps)))];
            break
        end
    end
end

Nh=[qc2h(:,1) qh qc2h(:,2) ];
Nc=[qh2c(:,1) qh2c(:,2) qc];


%Establish a matrix that will include the difference in q
%of both streams at each Temperature value in Tc and Th
%vectors
Diff=zeros(h+c,2);
for i=1:h
    Diff(i,:)=[Th(i), qc2h(i,2)-qh(i)];
end

for j=1:c
    Diff(j+h,:)=[Tc(j), qc(j)-qh2c(j,2)];
end

%sort the rows of the difference matrix to determine the
%pinch point,
mDiff= sortrows(Diff,2);
inc=abs(mDiff(1,2)); %The increment required to shift the
%cold stream to the right and hence to establish the pinch
%point The pinch point
Tp=mDiff(1,1);
%Shifting the cold stream
for i=1:c
    qc(i)=qc(i)+inc;
end
%These two matrices redefine the hot and cold stream on the
%pinch diagram
Mh=[Th qh];
Mc=[Tc qc];
if Mh(1,2)==Mh(2,2)
    Mh(1,:)=[];
end
if Mc(1,2)==Mc(2,2)
    Mc(1,:)=[];
end
if Mh(size(Mh,1),2)==Mh(size(Mh,1)-1,2)
```

```
    Mh(size(Mh,1),:)=[];
end
if Mc(size(Mc,1),2)==Mc(size(Mc,1)-1,2)
    Mc(size(Mc,1),:)=[];
end
%display of the basic information, the pinch point, the
%minimum heating
%utilities and the minimum cooling utilities requirements.
%The minimum utilty requirements

minHeat=Mc(size(Mc,1),2)-Mh(size(Mh,1),2);
minCool=Mc(1,2)-Mh(1,2);
figure(1)
plot(Mh(:,1),Mh(:,2),'-r',Mc(:,1),Mc(:,2),'-
b','LineWidth',2)
title('Thermal Pinch Diagram','FontSize',15)
xlabel('Hot Temperature (T)','FontSize',13)
ylabel('Heat Exchanged (Q)','FontSize',13)
legend('Hot Stream','Cold Stream',2);
grid on
xx=xlim;
yy=ylim;
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+3*(yy(2)-
yy(1))/20,['Tp=', num2str(Tp)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+2*(yy(2)-
yy(1))/20,['QH,min=', num2str(minHeat)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+(yy(2)-
yy(1))/20,['QC,min=', num2str(minCool)],'FontSize',9)
set(gcf,'position',[200 200 650 500])
%%%%%%%%%%%%%%%%%%%%%
%Building the Grand Composite Cuve (GCC)
%Establish the Matrix that will have alll Q values at each
Temperature
Temp_=zeros(size(Nh,1)+size(Mc,1),3);
Temp_=[Nh;Nc];
Temp_=sortrows(Temp_,1);

j=2;
while (j<=size(Temp_,1))
    if Temp_(j,1)==Temp_(j-1,1)
        Temp_(j,:)=[];
    else
        j=j+1;
    end
end
%Rearrange the Matrix so as to put the temperature column
%in a descending order
s=size(Temp_,1);
interval=zeros(size(Temp_,1),3);
for k=1:s
    interval(k,:)=Temp_(s-k+1,:);
```

```
end

%Establish the Cascade matrix that does the temperature
%balance at each stage
cascade=zeros(size(interval,1),2);
cascade(:,1)=interval(:,1);
for i=2:size(cascade,1)
    cascade(i,2)=cascade(i-1,2)+(interval(i-1,2)-
interval(i,2))-(interval(i-1,3)-interval(i,3));
end
%The cascade is now balanced by introducing the heat
%utility that will
%offset the deficiency
if min(cascade(:,2))>=0
    disp('No External Heating Duties is Required')
    addheat=0;
else
addheat=abs(min(cascade(:,2)));
end

for i=1:size(cascade,1)
    cascade(i,2)=cascade(i,2)+addheat;
    cascade(i,1)=cascade(i,1)-DT/2;
end

figure(2)
plot(cascade(:,2),cascade(:,1),'-k','LineWidth',2)
title('Grand Composite Curve','FontSize',15)
xlabel('Enthalpy (Q)','FontSize',13)
ylabel('(T+t)/2','FontSize',13)
grid on
xx=xlim;
yy=ylim;
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+3*(yy(2)-
yy(1))/20,['Tp=', num2str(Tp)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+2*(yy(2)-
yy(1))/20,['QH,min=', num2str(minHeat)],'FontSize',9)
text(xx(1)+(xx(2)-xx(1))/1.5,yy(1)+(yy(2)-
yy(1))/20,['QC,min=', num2str(minCool)],'FontSize',9)
```

Code 10

```
function[FinalOB]=example1()
time1=clock;
upbd=3.2;%initially any feasible point serves as an upper
%bound.
bestp=[1.6,0];%initial best point
dictionary1=[0 1.6 0 1 0 4.2 4.2 2.6];%initial box, for n
variables, it will have 2n+4 columns:
%(2 bounds x intervals)+ (2 bounds for inclusion) +
%inclusion range +total sides' width = 2n+4
```

```
%Initially the search is done in the whole given box
maxdiff1=4.2;%The maximum initial inclusion range on the
%objective function
while maxdiff1>0.0001;%The maximum allowed range in the
%final list of function inclusion
    dictionary=dictionary1; %dictionary one is used to
%temporally save the sub-bthe lower of the function
%inclusion.
    dictionary1(:,:)=[];%Empty dictionary1
    A=size(dictionary);
    w=A(1,1);
    for k=1:w; %run the algorithm for the w sub-boxes in
%the list.
        newlist=[]; %newlist is used to hold new subboxes
%at each iteration, emptied after sub-boxes added to the
%list.
        box=dictionary(k,1:4); %A box is selected for
%simulated annealing,
        if dictionary(k,7)<5 ||dictionary(k,8)<5 %Apply
%central point evaluation if inclusion range or max-width
%are less than a minimum
            for i=2:2:4
                p(i/2)=(box(1,i-1)+box(1,i))/2;        %The
%candidate point is the central point of a small box
            end
            obf=2*p(1)+p(2); %directly evaluate the
%objective function in this case
        else
            [p,obf]=annealing(box); %if the sub-box is big
%enough then SA is better than central evaluation
        end
        %Check the feasibility
        maxlength=-inf;
        for i=2:2:4 % to determine if the widest side is an
%integer
            length=box(1,i)-box(1,i-1);
            if length>maxlength
                maxlength=length;
                maxi=i/2;
            end
        end

    if maxi==2; %the second variable in the box is the
%integer
        [p,obf]=bestint(p); % A function that evaluates the
%best objective functiob value on both sides of the integer
    end
        [maxcons,c]=feasibility_check(p);%the constraint
%with the maximum value is determined, c indicates which
%constrain that has the maximum value
```

```
        if  (maxcons)<1e-8;  %which  means  the  point  is
%feasible given the maximum constraint is negative


            if  obf<upbd  %update  the  upper  bound  if  the
%objective function value is feasible.
                bestp=p;
                upbd=obf;
            end

            [bi_box]=halfcut(box);   %If   the   point   is
%feasible, the box is siplit at the feasible point across
%the widest side

[newlist,maxdiff1,upbd,bestp]=inclusion(bi_box,upbd,bestp);
%Test the resulting two boxes

        end
        if  maxcons>=1e-8 %if  the  SA  output  is  infeasible,
            %the distrust region is applied

            [infBox]=distrust(box,p,c); %the distrust
            %region starts at point p.
            %If the infeasible box turned out to be an
            %extremely small box,
            %then bisecting the box into two parts is
            %computationally better than
            %siplitting it into 2n.
            BB=0;
            for j=2:2:4
                avgint=(infBox(1,j)+infBox(1,j-1))/2;
                if  abs(avgint-box(1,j))<0.1  ||  abs(avgint-
box(1,j-1))<0.1
                    BB=1; %BB  is  an  index  used  to  mark  an
%infeasable box to bisected rather than siplitting into 2n
%sub-boxes

                end
            end
            if  BB==0  %the  default  index  for  an  infeasible
                       %box   that   will   undergo   siplitting
                       %into 2n sub-boxes
                       new_dictionary= newdict(infBox,box);
                       %This  will  produce  2n  sub-boxes  by
                       %removing  an  infeasible  box  from  the
                       %original mother box

[newlist,maxdiff1,upbd,bestp]=inclusion(new_dictionary,upbd
,bestp);
```

```
            else %i.e when B=1
                [cutbox]=halfcut(box); %To bisect the box
                      %at the central point across the
                      %widest side

[newlist,maxdiff1,upbd,bestp]=inclusion(cutbox,upbd,bestp);

            end

        end
        dictionary1=[dictionary1;newlist]; %dictionary1 is
%a temporary matrix used to store sub-boxes resulting from
%sipliting and bisecting of each iteration before they are
%added to the main list


    end

[dictionary1,maxdiff1,upbd,bestp]=inclusion(dictionary1(:,1
:4),upbd,bestp); %Before adding the sub-boxes, they are
%filtered on the hope of removing sub-boxes in case  a new
%update of the upper bound has taken place recently
    maxdiff1=max(dictionary1(:,7)); %checking the maximum
%range of the function inclusions in the list

end

%Final output
Best_value=upbd
Best_point=bestp
time2=clock;
RunTime=etime(time2,time1)
%%%%%%%%%%%%%%%%%%%%%%%%%%
function[minimum,fval]=annealing(box)
%%%%%%%%%%%%%%%%%%%%%%%%%%
A=size(box);
n=0.5*A(1,2);%the dimension of the box
volume=0; %To determine the total number of sides lenth
for i=2:2:2*n
    volume=volume+(box(1,i)-box(1,i-1));
end
si=(volume)/n; %the average side width of the box, this
%will be used in the neighborhood dynamic scheduling.
p=zeros(1,2);
%The initial point will be the center of the cube.
for i=2:2:2*n
        p(i/2)=(box(1,i)+box(1,i-1))/2;
end
results=zeros(1,2); % This is needed to tabulate the
results to observe the performance of the algorithm.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
Tinit = 1;        % initial temp
minT = 1e-8;         % stopping temp
cool = 0.85;       % annealing schedule
minF = -Inf;
max_consec_rejections    =    1000;    %maximum    consective
rejections allowed
max_try = 300; %maximum iterations per temperature
max_success = 20; %maximum consective acceptance allowed
k = 1;
tempchange=(log(minT/Tinit)/log(cool))+1;  %The  number  of
%temperature decrements
EPS=0.001; %The range of the final displacement needed to
%determine a nighbouring point
beta=001; %the fraction of the average box side for the
%first leap
factor=(beta*EPS/si)^(1/tempchange);  %This  is  a  fraction
%multiplied by itself each time the temperature changes,

%this will cause a change in the leap from beta*S in the
%begining to Epslon at the end
% The foloowings are used to initiate the counters
itry = 0;
success = 0;
finished = 0;
consec = 0;
T = Tinit; %Initial temperature set to Tinit
lpp=0; %The power of the fraction used to decrease the
%leap, initially zero anad increases by unity at each
%temperature change.
initenergy =2*p(1)+p(2); %initial funcrtion evaluation for
%the initial point
oldenergy = initenergy;

total = 0;
index=1;
while ~finished;
    itry = itry+1; % just an iteration counter
    current = p;
    % % Stop / decrement T criteria
    if itry >= max_try || success >= max_success;
        if T < minT || consec >= max_consec_rejections;
            finished = 1;
            total = total + itry;
            break;
        else
            T = cool*T;  % decrease T according to cooling
            %schedule
            lpp=lpp+1;
            total = total + itry;
            itry = 1;
            success = 1;
        end
    end

    newp(1)=-1;
    while
(newp(1)<box(1,1)||newp(1)>box(1,2)||newp(2)<box(1,3)||newp
(2)>box(1,4)) %to confine the neighbourhood selection


newp=p+(randperm(length(p))==length(p))*randn*si/500*factor
^lpp;
%the dynamic neighbourhood function

    end


    newenergy=2*newp(1)+newp(2);
    if (oldenergy-newenergy > 1e-6)
        p = newp;
        oldenergy = newenergy;
        success = success+1;
        consec = 0;
    else
        if (rand < exp( (oldenergy-newenergy)/(k*T) ));
            p= newp;
            oldenergy = newenergy;
            success = success+1;

        else
            consec = consec+1;
        end
    end
end
minimum = p;
fval = oldenergy;

%%%%%%%%%%%%%%%%%%%%
%This function is to determine the best function value if
%the longest side is integer
function[p,obf]=bestint(p);
        pi=[p(1) floor(p(2)); p(1) ceil(p(2))];
        obf=inf;
       for i=1:2
           f=2*pi(1,i)+pi(2,i);
           if f<obf
               obf=f;
               p=pi(i,:);
           end
       end
%%%%%%%%%%%%%%%%%%%%%%%%%%
function[wc,c]=feasibility_check(p);
```

```
%A point (y1,y2,x1,x2,x3,x4,x5) is tested if feasible
x=p(1);y=p(2);

P(1)=1.25-x^2-y;
P(2)=x+y-1.6;
wc=max(P);
for i=1:2 %To determine which constraint has the maximum
%value
    if wc==P(i);
        c=i;
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [xp]=bisect(point,box) %used to bisect (siplit is
%more correct) the box at a feasible point
A=size(box);
n=A(1,2)/2;
maxlength=0;
for i=2:2:2*n
    length=box(1,i)-box(1,i-1);
    if length>maxlength
        maxlength=length;
        maxi=i;
    end
end
xp=zeros(2,2*n);
for i=2:2:2*n
    if i==maxi && i==4 %if the variable of the longest side
%is supposed to be an integer then its value is rounded to
%the neighbouring integers
        xp(1,i-1:i)=[box(1,i-1) floor(point(i/2))];
        xp(2,i-1:i)=[ceil(point(i/2))   box(1,i)];
    elseif i==maxi
        xp(1,i-1:i)=[box(1,i-1) point(i/2)];
        xp(2,i-1:i)=[point(i/2)   box(1,i)];
    else
        xp(1,i-1:i)=box(1,i-1:i);
        xp(2,i-1:i)=box(1,i-1:i);
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%
function[infbox]=distrust(sbox,p,c);
%This function is used to extend an infeasible point in all
%directions to create an infeasible box within the search
%domain

infbox=[p(1) p(1) p(2) p(2)]; %the initial infeasible box
is basically the infeasible point
Eps=0.05; %The extention in a direction
for i=1:4
    exbox=infbox;
    Pc=1;
    while max(Pc)>0.0001 && ((exbox(1,i)>sbox(1,i) &&
mod(i,2)==1) || (exbox(1,i)<sbox(1,i) && mod(i,2)==0))
% this while statement is only performed if the so %far
%infeasible box is still strictly infeasible and within
%the original box

        if mod(i,2)==1
            exbox(i)=exbox(i)-Eps;
        elseif mod(i,2)==0
            exbox(i)=exbox(i)+Eps;
        end
        xL=exbox(1,1);xR=exbox(1,2);yL=exbox(1,3);
yR=exbox(1,4); %the new bigger box

        Pc(1)=1.25-xR^2-yR; %testing the lower bounds of
%the constraints inclusion which (one or more) should be
%positive for a strictly infeasible box
        Pc(2)=xL+yL-1.6;


        if max(Pc)>0.0001 && ((exbox(1,i)>=sbox(1,i) &&
mod(i,2)==1) || (exbox(1,i)<=sbox(1,i) && mod(i,2)==0))
            infbox=exbox; %the new bigger infeasible box is
%set as such only if it is strictly infeasible
        end
    end
end
infbox=[infbox(1,1) infbox(1,2) floor(infbox(1,3))
ceil(infbox(1,4))]; %The final infeasible box with
%additional rounding to the near integers for the integer
%intervals


%%%%%%%%%%%%%%%%%%%%%%%%

function[newlist,maxdiff,upbd,bestp]=inclusion(list,upbd,be
stp);
%This is the main function used to filter unquialified
%suboptimal sub-boxes
A=size(list);
n=A(1,1);
M=n;
A=size(list);
n=A(1,1);
F=zeros(n,3);
diff=zeros(n,2);
for i=1:n
    xL=list(i,1);xR=list(i,2);
    yL=list(i,3);yR=list(i,4);
    %The inclusion of the objective function.



    F(i,2)=2*xL+yL;
    F(i,3)=2*xR+yR;
    Fmin=F(i,2);Fmax=F(i,3);

    F(i,1)=i;
     diff(i,1)=i;
    diff(i,2)=F(i,3)-F(i,2);
    %The inclusion of the Constraints;
    P1(i,2)=1.25-xR^2-yR;
    P1(i,3)=1.25-xL^2-yL;
    P1(i,1)=i;

    P2(i,2)=xL+yL-1.6;
    P2(i,3)=xR+yR-1.6;
    P2(i,1)=i;

end
maxdiff=max(diff(:,2));
newlist=[list  F(:,2:3) diff(:,2)];
i=1;
M=n; %This M is needed to direct the search to the right
%box, a for loop would skip a box each time a box is
%deleted because once a book is deleted, the lsit will move
%one row upwards and that would cause missing the next box
%in order
while i<=M
 %for each of the boxes in the list, the four tests are
conducted
    if (P1(i,2)>1e-4) ||(P2(i,2)>1e-4)
        newlist(i,:)=[];  %The box is deleted if it is
%strictly infeasible
        F(i,:)=[];
        P1(i,:)=[]; P2(i,:)=[];
        M=M-1;
    elseif F(i,2)>upbd
         newlist(i,:)=[];  %If the lower bound of the
%function inclusion is more than the upper bound then the
%box is also deleted
        F(i,:)=[];
        P1(i,:)=[];P2(i,:)=[];

        M=M-1;

    else
        i=i+1;
    end
end
```

```
%Add the volume of each of the sub-boxes
A=size(newlist);
n=A(1,1);
vol=zeros(n,1);
for i=1:n
    vol(i,1)=0;
    for j=2:2:4
        vol(i,1)=vol(i,1)+(newlist(i,j)-newlist(i,j-1));
%The sum of the sides' length is added as a column in the
%list
    end
end
newlist=[newlist vol];
%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [D] = newdict(removed,whole) %This function is
%used to remove an infeasible box from within a larger box
format short g
A=size(whole);
n=A(1,2)/2;
for i=2:2:2*n
    X(i/2,:)=[whole(1,i-1) whole(1,i)];
    c(i/2,:)=[removed(1,i-1) removed(1,i)];
end
%In each direction we will have center, right and left
%interval, the center is the one that is taken out and the
%surrounding will be added to the list
for i=1:n
    L(i,:)=[X(i,1) c(i,1)];
    R(i,:)=[c(i,2) X(i,2)];
end
% Arranging the dictionary

D=zeros(2*n,2*n);
for i=1:n
    for j=1:n
        if j>i
            D(i,2*j-1:2*j)=X(j,:);
        elseif j==i
            D(i,2*j-1:2*j)=L(j,:);
        else
            D(i,2*j-1:2*j)=c(j,:);
        end
    end
end
for i=n+1:2*n
    for j=1:n
        if j>i-n
            D(i,2*j-1:2*j)=X(j,:);
        elseif j==i-n
            D(i,2*j-1:2*j)=R(j,:);
        else
            D(i,2*j-1:2*j)=c(j,:);
        end
    end
end
D;
%%%%%%%%%%%%%%%%%%%%%%%%%%
function  [cutbox]=halfcut(box)  %This function is used to
%bisect a box at the central point across the widest side
 format short g
A=size(box);
n=A(1,2)/2;
maxL=-inf;
for i=2:2:2*n %To determine the widest side
    length=box(1,i)-box(1,i-1);
    if length>maxL
        maxL=length;
        ind=i;
    end
end
for j=2:2:2*n
    if j==ind && j==4 %if the widest is the integer then
%the bounds will be rounded to the near integers
        midpoint=(box(1,j)+box(1,j-1))/2;
        cutbox(1,j-1:j)=[box(1,j-1)  floor(midpoint)];
        cutbox(2,j-1:j)=[ceil(midpoint)  box(1,j)];
    elseif j==ind
        midpoint=(box(1,j)+box(1,j-1))/2;
        cutbox(1,j-1:j)=[box(1,j-1)  midpoint];
        cutbox(2,j-1:j)=[midpoint  box(1,j)];
    else

        cutbox(1,j-1:j)=box(1,j-1:j);
        cutbox(2,j-1:j)=box(1,j-1:j);
    end
end
cutbox;
```

VITA

Name:	Nasser Ahmed Zahran Al-Azri
Address:	Sultan Qaboos University P.O.Box 33 Al-Khod 123 Muscat-Oman
Email Address:	nalazri@gmail.com
Education:	B.Eng., Mechanical Engineering, Sultan Qaboos University, 1999 M.S., Chemical Engineering, Texas A&M University, 2002 M.Eng., Industrial Engineering, Texas A&M University, 2007 Ph.D., Chemical Engineering, Texas A&M University, 2008