Process Simulation & Optimization
Sequential modular approach or global approach? And why not both?
Philippe BAUDET(*), Pierre CASTELAIN, Olivier BAUDOUIN - ProSim SA

Keywords
Power plants Simulation and Optimization - Ariane

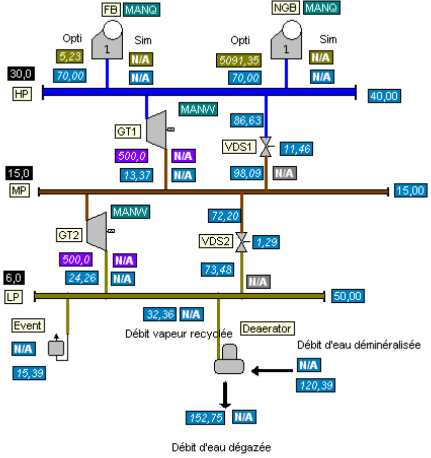
Summary
Be they steady state or transient state, "Process simulators" can be split in two categories: simulators with a sequential modular approach and simulators with a global resolution approach. Both approaches are often opposed and present specific advantages and disadvantages. The work reported here consists in reconciling those two approaches. It was carried out in Ariane^{TM}, a software dedicated to power plants simulation and optimization.

Text

Scope of Ariane^{TM} software and didactic example

The scope of Ariane^{TM} software is the modelling, simulation and optimization of utilities power plants, i.e. power plants producing simultaneously different energies like steam, hot water or electricity from fossil fuels. This type of production is carried out under increasingly severe technical and environmental constraints and when optimization studies are carried out, the chosen criterion is most often an economical one.

Let us consider the utilities power plant presented hereafter (modelled in Ariane^{TM}):



The power plant includes 3 networks (HP, MP, LP), 2 monofuel boilers (FB, NGB), 2 simple turboalternators (GT1, GT2), 2 superheating valves and a deaerator (degasser). Efficiency of the boilers and turbines are assumed constant. The aim of such plant is to provide the amount of steam required by HP, MP and LP networks, while producing a maximum of electricity or while minimizing fuel consumption with de-superheating operations. The most efficient fuel(s) and boiler(s) have to be identified too. The optimum operating point depends on the required productions and on current tariffs (fuel, water, electricity). In fact, the system is in a closed loop because the deaerator provides water to boilers and de-superheating valves. In fact, a network is modelled like the succession of a mixer and a divider and, from a modelling point of view, it is a “mixing and dividing point”.

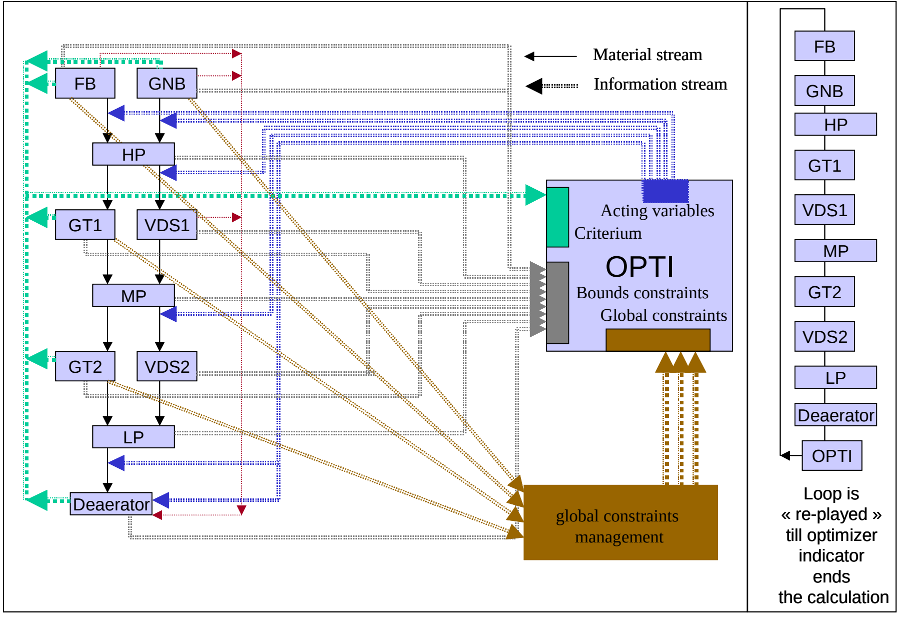
Sequential modular approach: historical approach

On the one hand, the sequential modular approach presents at least two advantages that are intrinsically related to its philosophy of resolution. The first advantage relates to the robustness: the resolution is divided-up into several subsets that are treated sequentially. This ensures rigorous convergence, even in presence of extremely complex modules that are treated in an autonomous way. The second advantage relates to initialization which is greatly facilitated by the sequential modular approach. Considering the importance of initialization in the optimization process, it is a significant advantage.

In a sequential modular approach, the unit operation calculation sequence could be as follows:

FB / NGB / HP (Mel/Div) / GT1 / VDS1 / MP (Mel/Div) /GT2 / VDS2 / LP (Mel/Div) / Deaerator

The optimization structure is presented in the figure hereafter.



The optimizer is the convergence module of the calculation loop. As long as the OPTI module does not indicate the end of calculation, the sequence is re-started, either to evaluate the criterion or the constraints (with new values for the acting variables), or to evaluate the gradients (criterion and constraints). At the end, the organization of calculations is rather complicated, with (too) many information streams. The major problem of this approach is the computation time. Based on numerical evaluations of gradients, it requires numerous runs of the entire modular sequence. In addition, the architecture becomes very difficult to manage when the size of the power plant increases as the number of information streams to be managed multiplies.

This sequential modular architecture is ill-suited for some power plants in the petrochemical field: computation time becomes too important and the management of information streams is sometimes impossible.

Global approach: advantages and disadvantages

A global solution treats simultaneously all the equations of the system. It generally allows getting the analytical expressions of the derivative. From the computing time point of view, the execution time is much shorter than for the sequential modular systems. However, it is at the cost of robustness, as all equations of the system are treated globally and only once and as initialization of such systems is generally a rather complex and tedious task to carry out.

Methodology:

The present project aims at combining both approaches. The underlying objectives are:

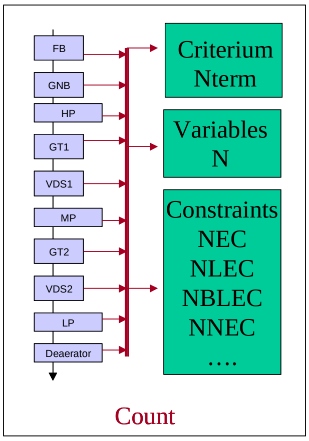
- ✓ to preserve the maximum of existing code
- ✓ to continue to benefit from variables initialization of the sequential modular architecture
- ✓ to have an automatic analysis of the resulting mathematical problems
- ✓ to avoid problems linked to the number of information streams
- ✓ to provide an externalized resolution of the problem
- ✓ to provide a library of solvers able to solve all kind of problems
- ✓ to be able to make the library of solvers evolve

The work has consisted in dividing the procedure in four major stages:

- ✓ 1: “Count” of all variables and all constraints
- ✓ 2: “Encoding” of the mathematical system to be solved and initialization of all variables
- ✓ 3: Analysis of the mathematical problem, choice of the most appropriate solver and problem solving
- ✓ 4: Final simulation with the results provided by the solver and edition of all the results.

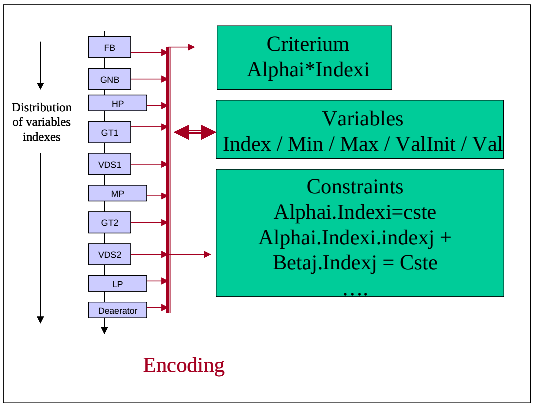
In this architecture, the sequential modular sequence is only executed three times (count, encoding and final simulation). This treatment makes it possible to keep the maximum of the existing code (a large part of the keywords file is preserved, one just have to remove the convergence module and its associated information streams), and the initialization procedure of the sequential modular mode leads to a much faster computation time (with global resolution and analytics derivatives). In other words, we preserved the best of both approaches.

The precise implementation of this architecture rests on 3 major modules: VARIABLES, CONSTRAINTS and CRITERIUM. These three modules are "loaded" by the counting procedure and the encoding procedure, then used by the solver during the calculation. Finally (after the end of externalized computation) the VARIABLES module provides the optimal values for the final simulation.



Counting procedure diagram

During the counting procedure, the execution of the sequential modular sequence makes it possible for each module to declare its own variables, its own constraints and their characteristics (linear, bilinear, or non-linear… equalities or inequalities), its own terms acting on the criterion… At the end of this procedure, the system knows exactly the size and the type of the problem to be solved (number of terms of the criterion, number of variables, number of constraints and "type" of each constraint). The allocation of memory can then be carried out.



Encoding procedure diagram

During the encoding procedure, the sequential modular execution allows process modules to send to the CRITERIUM, CONSTRAINTS and VARIABLES modules all the required data for the later global calculation. It is a rather complex procedure that we will try to illustrate through a simple example in Ariane^{TM}.

Let us consider the boiler FB. In a very simplified approach, if this module is in "automatic" mode, it has two variables declared during the counting procedure: the steam flowrate variable Qs and the fuel flowrate variable Qf (steam output temperature of boilers are fixed in Ariane^{TM}, if not, there would be a third variable: output enthalpy). These two variables have to be defined more precisely in the VARIABLES module: variable index, initial value, bounds. The index of the steam flowrate variable is provided by the VARIABLES module (it is a single identifier, an ID) and the boiler module sends to the VARIABLES module information about initial value and bounds of the variable. The variable index is inserted as additional information in the output material stream (and it can then be used in the following modules of the calculation list, HP network in this specific case). The index of the fuel variable is sent to the CRITERIUM module (with some other information) because this variable impacts the economical criterion.

If the boilers were described with a constant efficiency, the equation binding the two preceding variables is linear: Qf=Beta*Qs.
This equation was declared during the counting procedure, it must now be described completely. A linear equation is defined in the form

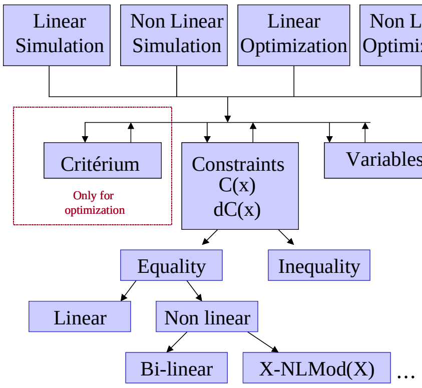
$$\sum_{i=1}^{Nterm} Alpha_i * Index_i = Const \,.$$

The information sent to the CONSTRAINTS module is then Nterm=2, Const=0, Index1= Qf index variable, Alpha1=1, Index2= Qs index and Alpha2=-Beta.
All kind of constraints met in Ariane^{TM} are "encodable" in this fashion, i.e they can be defined in a preset format. All in all, the variables indexes which are transferred by material streams are flow rate variables of enthalpy variables. For this, 2 positions are reserved in the material streams [flow (P1), enthalpy (P2)]. For a given material stream entering a module, if P1 is null, the flow is a parameter, if not, the flow is a variable and variable index is P1. It is the same for enthalpy with P2 as value analysis. These two kinds of variables are the only ones transferred from a module to another by material streams. Of course, other variables (output, NCV, electric output…) exist but are not forwarded by material streams.
This is, in a very simplified way, how the mechanism of encoding functions. Additionally it has to be noted that each type of "encodable" equation has its own derivative equation.

At the end of the encoding procedure, the resulting system is analyzed and 5 possibilities exist, by considering NEQ (number of equality constraints), Nvar (number of variables) and NLEQ, (number of linear equality constraints):

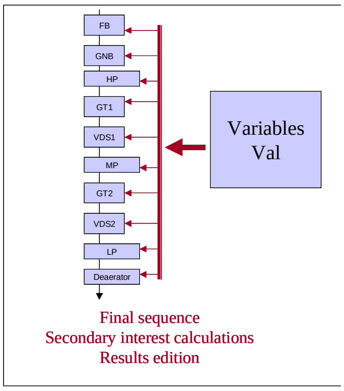
- ✓ the generated system cannot be solved (example: more equations than variables)
- ✓ the generated system is a linear simulation (NEQ (=NLEQ) = Nvar)
- ✓ the generated system is a nonlinear simulation (NEQ (>NLEQ) = Nvar)
- ✓ the generated system is a linear optimization (Nvar > NEQ and NEQ=NLEQ)
- ✓ the generated system is a nonlinear optimization (Nvar > NEQ ; NEQ > NLEQ)

For each of the last 4 cases, a dedicated method can be called, that will use the modules previously described. A library of solvers has been developed. Each solver can be called automatically or can be selected by the user. Any solver can be added to the library, if it implements the main interface (interface of the CRITERIUM, CONSTRAINTS, and VARIABLE modules), and this, in a totally disconnected way from Ariane^{TM} architecture.



Solving Architecture

At the end of the calculations, the VARIABLES module contains optimal values which are sent to final simulation, as presented in the figure hereafter.



Final simulation and edition of all results

The final simulation uses the values contained in the VARIABLES module. The sequential modular approach is used a third times to carry out the simulation with optimal values but also to calculate some other values which have not been included in the optimization problem (for example, if the user has not defined constraints on pollutant emissions, those are calculated only during final simulation) and then to edit all the results.

Validation on the didactic example

The aim is to show how, starting from a same topology, it is possible to obtain very different systems to be solved.

Case number 1: the boilers and turbines are in ‘Manual’ mode; steam networks temperatures are calculated. The generated problem is then:

- ✓ Number of variables: **17**
- ✓ Number of equality constraints: **17**

With 11 linear models, 4 bilinear models and 2 nonlinear models

The problem to be solved is then a non-linear algebraic system.

Case number 2: starting from case 1, the steam networks temperatures are fixed (to the values obtained in case 1). The problem then becomes:

- ✓ Number of variables: **12**
- ✓ Number of equality constraints: **12**

With 12 linear models

The problem to be solved is then a linear algebraic system.

Case number 3: starting again from case 1 (with calculated temperatures for networks), the boilers are switched to “automatic” mode (theirs flowrates become a freedom degree). This leads to:

- ✓ Number of variables: **26**
- ✓ Number of inequality constraints: **3**
- ✓ Number of equality constraints: **24**

With: 14 linear models, 6 bilinear models and 4 nonlinear models

The generated problem is here a true optimization problem (the number of variables is higher than the number of equality constraints). The resulting problem is then a Non Linear optimization Problem (NLP).

Case number 4: starting from case 3, the steam networks temperatures are fixed (to the values obtained in case 1). The problem then becomes:

- ✓ Number of variables: **17**
- ✓ Number of inequality constraints: **3**
- ✓ Number of equality constraints: **15**

With 15 linear models

The problem to be solved is then a Linear optimization Problem (LP).

These 4 cases show how the mathematical system to be solved can evolve with the modelling choices of the user, even if the “topology” is unchanged. After the Counting and Encoding procedures, the software can adapt to the complexity. Resort to nonlinear methods is made only when the generated mathematical systems are nonlinear. The optimizers are used only if degrees of freedoms really exist. The choice of the solving method is done after an analysis that identifies the most adapted one.

Validation on an industrial case: Industrial power plant in France

For one of the industrial plant that have been used for the project validation, the model comprises:

- ✓ 4 steam networks
- ✓ 1 fuel network (with 3 fuels entering on the network)
- ✓ 3 bi-fuel boilers
- ✓ 2 turbo-alternators
- ✓ 32 switchable turbines.

The size of the optimization problem to be solved in the non linear case is described hereafter:

- ✓ 89 variables
- ✓ 79 equality constraints [70 linear, 6 bilinear, 3 nonlinear models (X–NLMod (X) = 0)]
- ✓ 5 inequality linear constraints

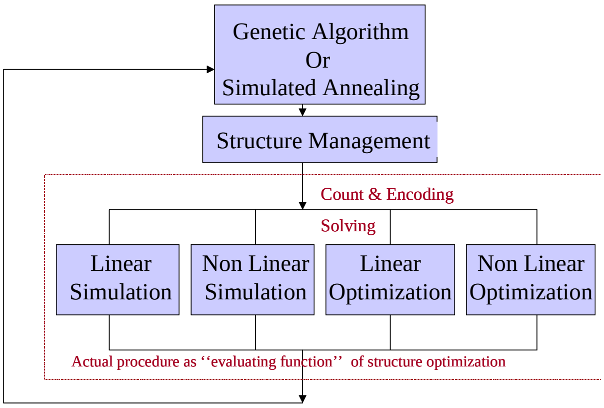
With the modular sequential based architecture, this utilities power plant could not be optimized because of the keywords file which became too big and largely exceeded the processing capacities (number of information streams to be managed). When removing some pieces of equipment, it was possible to calculate this power plant with computation time of around 20 minutes. Real time optimization was consequently not an option to consider.

With the combined structure, this utilities power plant is easily calculated. For a linear optimization (constant efficiencies, steam networks temperature are fixed…) the computing time is lower than a second. For a nonlinear optimization, the computing time is lower than 10 seconds (computing times were estimated with a todays standard performance level laptop).

Tests were carried out on other models of other utilities power plants and results are of the same order of magnitude in terms of time saving (from few minutes to few seconds on the same machine) on the most complex cases.

Outlines

The work presented in this paper relates to the optimization of operating conditions, i.e. a problem in continuous variables. *This work made it possible to increase significantly the capabilities of Ariane*^{TM} *software (size of problems, robustness and computation time)*. Nevertheless, some problems cannot be solved with this approach, such as for example equipment start/stop decision. The work to manage a “continuous– Boolean” combined approach is pending and should lead to the solving architecture presented hereafter.



This architecture will allow dealing with real time structure optimization (starting/stop of equipment according to instantaneous constraints and data). Among the Boolean methods in course of implementation, one can note a genetic algorithm (GA) method and a simulated annealing (SA) method developed by ProSim.