Heat Exchanger Network Retrofit via Constraint Logic Programming

H A Abbas*, G A Wiggins†, R Lakshmanan‡and W Morton§
University of Edinburgh, Edinburgh, United Kingdom

Abstract The retrofit design of heat exchanger networks (HENs) presents a more difficult challenge to mathematical programming methods even than grassroots design. Pinch technology methods based on thermodynamics take no account in their pure form of the benefit to be gained by using existing area in a HEN retrofit or of repiping costs which may be significant. A novel approach to the retrofit problem using Constraint Logic Programming (CLP) is described here. It employs a set of heuristics derived from an interactive retrofit method published earlier (Lakshmanan and Bañares-Alcántara, 1996), and uses CLP to efficiently prune out unattractive solutions. In most of the case studies on which the program was tested, the solutions were superior to others reported in the literature, although there is no guarantee of optimality. The approach also generates a *set* of solutions rather than just one, which may make the automatically generated retrofit proposals more acceptable in industry.

Keywords: synthesis, multicomponent, optimization

1 Introduction and Motivation

Retrofit design of heat exchanger networks (HENs) is an important practical aspect of process systems engineering. However, it has received relatively little attention in the process integration literature in contrast to the grassroots design problem. The methods used for grassroots design have also been applied to retrofit design:

- thermodynamic approaches based on pinch technology (Shokoya, 1992; Tjoe, 1986);
- formal optimization methods (Ciric and Floudas, 1989; Yee and Grossmann, 1987);
- heuristic approaches, e.g. (Lakshmanan and Bañares-Alcántara, 1996; 1998).

Pinch targeting methods which are rigorous for grassroots can be only heuristically applied in the retrofit problem, and experience has shown that they can err by as much as 86% (Lakshmanan and Bañares-Alcántara, 1996).

Some early mathematical programming approaches (Ciric and Floudas, 1989) also produced suboptimal solutions due to incomplete inclusion of the costs in the formulation of the problem. Other research (Yee and Grossmann, 1987) has yielded more robust formulations, but the computational overheads for solving even modest sized problems are significant. Decomposition schemes which approximate the problem cannot guarantee optimality and may exclude certain categories of solution (Briones and Kokossis, 1996; Asante and Zhu, 1996).

A heuristic, designer-driven graphical method for 'retrofit by inspection' has recently been developed, which uses a graphical visualisation aid known as the Retrofit Thermodynamic Diagram (RTD) (Lakshmanan and Bañares-Alcántara, 1996; 1998). The RTD highlights regions of area inefficiency in a network, thereby helping the designer to see where reconfiguring the network can most effectively improve its performance. The main limitation of this method is that it relies on the ingenuity and innovation of the designer and cannot advise the user.

This paper describes a method which uses heuristics similar to those in the RTD approach but automates their use through the technique of *Constraint Logic Programming*. This permits an efficient search of the space of potential solutions to find the most profitable ones. Although the method uses heuristics, it employs a more complete cost model and has in some cases produced solutions superior to those generated by formal optimization approaches.

2 Constraint Logic Programming (CLP)

Constraint Logic Programming (CLP) is one of the latest development of the logic programming (LP) paradigm. The idea of LP is that one can state a problem as a logical theory, and use a standard theorem proving engine to explore its consequences. In this way, we can separate the notions of *logic* (what a program does) and *control* (how it is done). In ideal circumstances, logic would be the domain of the programmer, while control would be left entirely to the computer. LP is well suited for knowledge-based reasoning (such as that employed here) because it is a symbolic language and so facilitates representa-

*Machine Learning Research Centre, QUT, Australia
†Department of Artificial Intelligence
‡Department of Chemical Engineering
§Department of Chemical Engineering

tion of problems, and because it has logical inference built in. LP's computation mechanism is logical inference: we run a program by stating a goal, which the LP system then attempts to prove true. So we find solutions by *searching* through a *search space* of possibilities. A good program is then one which directs search towards solutions effectively. Sterling and Shapiro (1994) give more details.

As an example of a logic program, consider the simple family tree shown in Figure 1, which can be represented in Prolog, the LP language we use, as shown in the definition of `parent` in Figure 2.

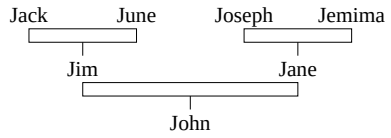

Figure 1: Family tree example

Also shown in Figure 2 is a definition of the predicate `ancestor` which is true if its first argument is an ancestor of its second. (To understand this, the reader needs to know that the `:-` operator means "if".) Given this we can derive answers to questions such as "Who are the ancestors of John?", "Does Jane have an ancestor?", "Who is descended from June?" and "Is Jim John's parent?". This is a very simple example of how a Prolog program can represent knowledge and inference clearly, explicitly, and effectively.

```
parent( jack, jim ).
parent( june, jim ).
parent( joseph, jane ).
parent( jemima, jane ).
parent( jim, john ).
parent( jane, john ).

ancestor( Older, Younger ) :-
       parent( Older, Younger ).
ancestor( Older, Younger ) :-
       parent( Middle, Younger ),
       ancestor( Older, Middle ).
```

Figure 2: A Prolog Program expressing the familty tree in Figure 1, and the `ancestor` relation defined over it.

CLP is an extension of LP where the ability to prove statements is enhanced by the ability to reason in general terms about mathematical *constraints* placed on the domains and values of variables. This allows reasoning to proceed much more efficiently, because strong interdependencies can be asserted between different parts of the search space, either preventing the search from entering infeasible areas, or encouraging it to enter potentially fruitful ones (van Hentenryck et al, 1991). In the present work, we view the HEN retrofit problem as a constrained search through a space of possibilities.

3 Retrofit Solution Heuristics

Most retrofit studies, apart from some debottlenecking problems, start with a feasible, operating heat exchanger network. The economics of operation are improved by reducing energy consumption, possibly by purchasing extra area or by relocating existing area which involves repiping heat exchangers. Stream splitting may also be used but is often deprecated because it complicates and forces reevaluation of controllability and operability issues.

In developing heuristics we consider the following steps:

Load shifting transfers heat load from the utilities to process exchangers, by adding shells to an existing unit. For load shifting to be feasible, there must be one or more exchangers linking a pair of hot and cold utilities through a *heat-load path*. The driving force in the linking exchanger(s) should be significantly greater than the minimum needed by operability considerations.

Large utilities which are initially not linked may be connected in a load path by adding one or more new exchangers to the network. When performing load shifting it may be worth considering the complete elimination of a heater or cooler whose area could be reused elsewhere in the network for process-process heat exchange.

'Criss-cross' arises when some matches have larger temperature differences than others in the same temperature range and generally indicates suboptimal use of area. Criss-cross may have been economically justifiable when the network was designed, or may arise from changes in operating conditions, or from a poorly designed network. To reduce criss-cross we can consider repiping exchangers and possibly stream splitting. Repiping is more common and enables the existing area to be used more effectively, so that the overall heat exchange is closer to "vertical" (on a RTD or pinch diagram).

Adding a new exchanger is considered where a new match would create a load-path between a pair of hot and cold utilities.

Despite the difficulty of the retrofit problem, there are relatively few modifications that are both thermodynamically feasible and economically attractive. This encourages us to think that a general-purpose approach such as CLP can be successful, provided the rules for selecting or rejecting solutions are sufficiently strong to prune out large sections of the search space.

4 HEN Retrofit via CLP

The CLP implementation uses a formal, mathematical definition of the HEN retrofit problem (Abbass,

1997). It is more general than the majority of mathematical programming formulations and includes the ability to declare individual stream match and repiping costs.

Capacity flow, source and target temperature are given for each process stream, and source and target temperature for each utility stream. A set of existing exchangers is given, each described by a type code (e.g. for process-process or process-hot utility), area and a shell efficiency.

Costs are supplied: for moving a heat exchanger, for a single repipe, for a new heat exchanger, for additional area in existing exchangers, and for hot and cold utilities. We define

$$Payback = \frac{R_M + R_P + A}{U_I - U_F}$$

where U_I is the initial cost of all utilities, U_F is the cost of utilities after retrofit, R_M, R_P are the retrofit costs for moving equipment and piping, and A is the cost of new area.

In the rules supplied to the CLP package (Sicstus Prolog) load shifting is performed both before and after each modification to the network is proposed. The linear programming solver in Sicstus Prolog adjusts the load shifts to maximise the possible energy saving at each stage.

Criss-cross is identified by comparing inlet temperatures on the hot and cold sides for pairs of exchangers. A pair of exchangers has criss-cross if their hot side inlet temperatures are ordered differently from their cold side inlet temperatures. In this case, exchanger relocation, or repiping, is attempted by switching the cold ends of the two exchangers and reshifting the loads.

Further on in the search, after changes such as stream splitting and the addition of new exchangers have been considered, there may be scope for new repipes that feasibly reduce criss-cross. The method therefore cycles through all of the steps repeatedly until the entire space of solutions has been either pruned or searched.

Despite its undesirability, stream splitting may be worthwhile if the payback in energy savings is sufficient. We therefore allow for it. In the current implementation, stream splitting is done when the temperature approaches of selected exchangers reach a specified minimum limit dictated by operability considerations, causing a *bottleneck*. Potential bottlenecks are stored in a list, and a stream split is attempted in turn on the hot and cold side in order to eliminate the bottleneck. If the split does not make the exchanger feasible, or if there is no cost improvement, the step is rejected and the system backtracks. Otherwise the split is accepted and the algorithm proceeds with the next phase.

To determine the optimal ratio for a stream split, a nonlinear program should be solved. This cannot be handled by Sicstus Prolog which can handle linear constraints only. A heuristic was used, that the streams are split in proportion to the capacity flow rates on the opposite side of the exchangers through which the split stream passes. This has worked well in practice though it does not guarantee optimality.

In the current system, we consider addition of new exchangers one at a time, in order to create a load path between utilities. This was done to reduce the extent of the search but excludes solutions which require the addition of two or more exchangers to form a path. The limitation has been found to impair performance on one example.

The current algorithm contains a number of limitations:

1. Heat exchangers are single stream.
2. HEN source and target temperatures are fixed, as commonly assumed in the literature.
3. Mixing after stream splits is assumed to be isothermal. This is thermodynamically attractive but may not lead to the lowest cost solutions overall if area can be better used by non-isothermal mixing.
4. Stream split ratios are fixed. Both the preceding assumptions were made to satisfy the linearity requirements of Sicstus Prolog.
5. A single exchanger is assumed to be sufficient to generate a heat load path. As noted, this may exclude good solutions where two exchangers are needed to create a path.

5 Case study

We report a case study to illustrate typical performance. More will be presented in a paper being prepared. This example was reported in (Ciric and Floudas, 1989). The original network is shown in Figure 3. The RTD for the same network is shown in Figure 4. The large utility exchangers indicate a potential for retrofit. The criss-cross highlights a situation in which splitting of stream C1 might be profitable.

A sample solution generated by CLP is shown in Figure 5 . The total added area is 5457 m^2 with one new exchanger added to the original network. The modification cost is \$939,623 and the annual savings is \$639,243, giving a 1.47 year payback. The originally published solutions reported a 2 year payback period so we find an improvement better than 25% . More importantly, all of them required a stream split, as intuition suggests. Due to the practical undesirability of splits in industry, the CLP solution is likely to be more attractive.

6 Conclusions

An algorithm for retrofitting a heat exchanger network using Constraint Logic Programming has been developed. In practice it seems efficient and works well. The method includes a formal description of

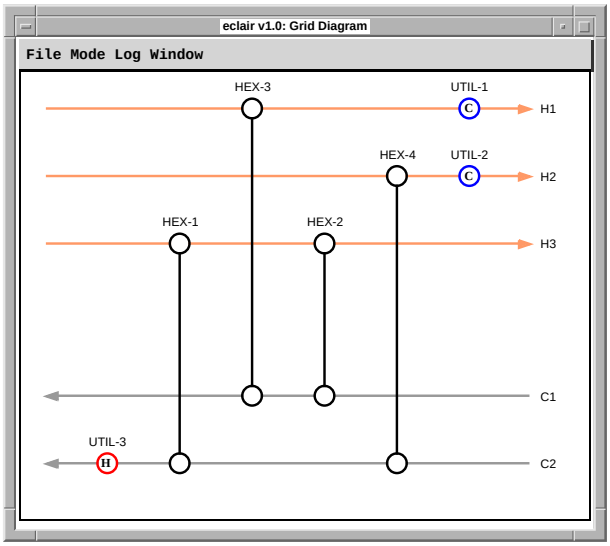

Figure 3: The initial HEN for Example 1

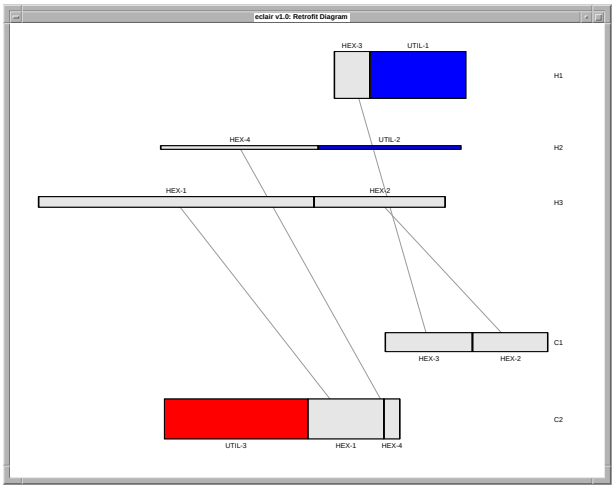

Figure 4: The RTD for the original HEN of Example 1

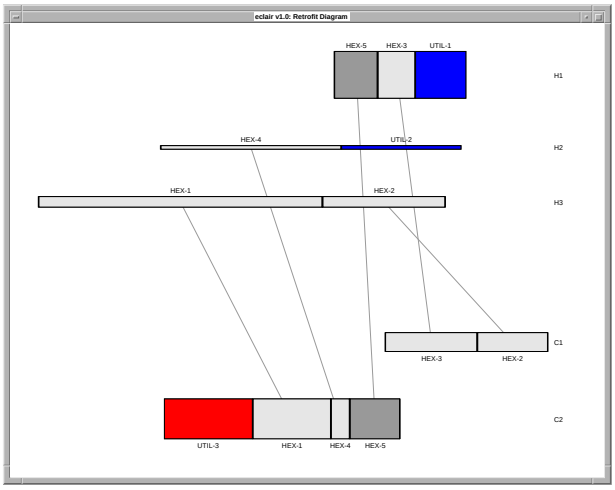

Figure 5: The RTD for the final HEN of Example 1

the retrofit problem. The key to the work is the integration of CLP with heuristics from earlier work on HEN retrofit. We believe this to be novel. Current limitations are primarily the need for linearity in the model (due to the capabilities of Sicstus Prolog), and the restriction that only one exchanger may be added to create a load path between utilities.

The resulting method has shown the ability to outperform rigorously modelled optimization methods on a number of problems, one of which is reported here.

References

Abbas, H.A., 1997, MSc thesis, Department of Artificial Intelligence, University of Edinburgh, Scotland.

Asante, N.D.K. and X.X. Zhu, 1996, An Automated Approach for Heat Exchanger Network Retrofit Featuring Minimal Topology Modifications, Computers and Chemical Engineering 20, S7-S12.

Briones, V. and A. Kokossis, 1996, A New Approach For The Optimal Retrofit Of Heat-Exchanger Networks, Computers and Chemical Engineering 20, S34-S48.

Ciric, A.R. and C.A. Floudas, 1989, A Retrofit Approach for Heat Exchanger Networks, Computers and Chemical Engineering 13(6), 703-715.

Lakshmanan, R. and R. Bañares-Alcántara, 1996, A Novel Visualisation Tool For Heat Exchanger Network Retrofit, Industrial and Engineering Chemistry Research 35(12), 4507-4522.

Lakshmanan, R. and R. Bañares-Alcántara, 1998, Retrofit by Inspection Using a Thermodynamic Process Visualisation, Computers and Chemical Engineering 22, S809-S812.

Shokoya, C.G., 1992, Retrofit of Heat Exchanger Networks for Debottlenecking and Energy Savings, Ph D thesis, UMIST, England.

Sterling, L. and E. Shapiro, 1994, The Art of Prolog (2nd ed.), MIT Press, Cambridge MA.

Tjoe, T.N., 1986, Retrofit of Heat Exchanger Networks, Ph D thesis, UMIST, England.

van Hentenryck, P., H. Simonis and M. Dincbas, 1991, Constraint satisfaction using constraint logic programming, Technical report CS-91-62, Dept. of Computer Science, Brown University, Providence RI.

Yee, T.F. and I.E. Grossmann, 1987, Optimization model for structural modifications in the retrofit of heat exchanger networks, Engineering Design Research Center report, Carnegie-Mellon University, Pittsburgh PA.