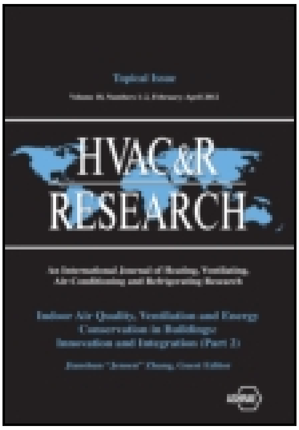

HVAC&R Research

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/uhvc20

A dual-mode evolutionary algorithm for designing optimized refrigerant circuitries for finned-tube heat exchangers

David A. Yashar [a] , Janusz Wojtusiak [b] , Kenneth Kaufman [c] & Piotr A. Domanski [a]

[a] HVAC&R Equipment Performance Group, National Institute of Standards and Technology, BFRL, 100 Bureau Drive, Mailstop 8631, Gaithersburg, MD, 20899, USA

[b] Department of Health Administration and Policy, and Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, USA

[c] Office of Research, Internal Revenue Service, Washington, DC, USA

Accepted author version posted online: 15 Dec 2011.Version of record first published: 27 Sep 2012.

Taylor & Francis
Taylor & Francis Group

A dual-mode evolutionary algorithm for designing optimized refrigerant circuitries for finned-tube heat exchangers

David A. Yashar,[1,*] Janusz Wojtusiak,[2] Kenneth Kaufman,[3] and Piotr A. Domanski[1]

[1]*HVAC&R Equipment Performance Group, National Institute of Standards and Technology, BFRL, 100 Bureau Drive, Mailstop 8631, Gaithersburg, MD 20899, USA*

[2]*Department of Health Administration and Policy, and Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, USA*

[3]*Office of Research, Internal Revenue Service, Washington, DC, USA*

**Corresponding author e-mail: dyashar@nist.gov*

Heat exchanger performance is strongly influenced by the refrigerant circuitry, i.e., the connection sequence of the tubes. This article describes an evolutionary computation-based approach for designing an optimized refrigerant circuitry used in an intelligent system for heat exchanger design. The technique used in this design employs two methods to generate designs implemented separately in two modules: the knowledge-based evolutionary computation module and the symbolic-learning-based evolutionary computation module. The optimization example presented in this article employed each module independently and used the combined approach to demonstrate the performance of each module and the power of the combined module approach. The best circuitry designs determined through these optimization runs yielded substantial capacity improvements over the original design; the symbolic-learning- and knowledge-based modules returned circuitry designs that improved the heat exchanger capacity by 2.6% and 4.8%, respectively, while the combined module approach resulted in a circuitry design that improved the capacity by 6.5%.

Introduction

The performance of finned-tube heat exchangers is greatly affected by a wide variety of design parameters (tube and fin geometry, spacing, circuitry, etc.), and the optimization objectives (capacity, material cost, refrigerant charge, etc.) are specific to each scheme. Over the past decade, researchers have examined different approaches to optimization problems associated with these heat exchangers, and the complexity of these design problems has directed much of the research toward evolutionary computation methods.

Evolutionary computation is based on the principles of evolution observed in nature. It uses an iterative process where potential solutions to a

Received February 3, 2011; accepted June 27, 2011

David A. Yashar, PhD, Associate Member ASHRAE, is Mechanical Engineer. **Janusz Wojtusiak, PhD**, is Assistant Professor and Director. **Kenneth Kaufman, PhD**, is Computer Scientist. **Piotr A. Domanski, PhD**, Fellow ASHRAE, is Leader, HVAC&R Equipment.

HVAC&R Research, 18(5):834–844, 2012.
ISSN: 1078-9669 print / 1938-5587 online
DOI: 10.1080/10789669.2011.604699

problem are grouped into a population; then each member of the population is evaluated individually, and members of the current population are selected and modified to produce members of a new improved population. The selection of "parents" of new members is guided in general by their "fitness" with respect to the optimization goal. Evaluation of the new population follows, and this process continues in a loop until the fitness criterion has been satisfied or a predetermined number of iterations executed. Evolutionary computation encompasses different optimization techniques, including evolution strategies, evolutionary programming, scatter search, genetic algorithms (GAs), and genetic programming (Michalewicz 1999). Among them, GAs have been used in various fields and have proven to provide a robust search in complex spaces (Goldberg 1989). As research in this field continues, multiple derivatives of the above methods have been developed. GAs stand out as an excellent method, since they have been proven to converge to a globally optimal solution if given sufficient run time (Fogel 1994).

Qiao et al. (2010) used a GA approach to examine the feasibility of optimizing a finned-tube heat exchanger design based on the distribution of the heat exchanger's fins. They used the heat exchanger model by Jiang et al. (2006) to evaluate the fitness of the candidate solutions. Through the course of this study, they examined optimization objectives of heat exchanger capacity and material cost. Their results showed that this approach was able to design fin spacing patterns that could improve the heat transfer by more than 2% in many cases or could reduce the fin material cost by more than 10%.

A sophisticated multi-objective GA (MOGA) approach was implemented by Aute et al. (2004), which also employed the model by Jiang et al. (2006). In their study, they examined a wide variety of design considerations, including airflow, heat exchanger size and shape, number of fins, etc., and simultaneously optimized their design for maximum heat transfer and minimal material cost. Through this approach, they generated a set of optimal solutions that could be used by an engineer to comprehensively evaluate the tradeoffs between the optimization objectives of cost and performance.

A more targeted effort has been implemented to optimize the connection sequence of the tubes in the coil, i.e., the refrigerant circuitry. The influence of circuitry design on the performance of a heat exchanger is well documented (e.g., Wang et al. 1999; Liang et al. 2001; Casson et al. 2002). After a design engineer has determined the general size, shape, fin type, tube pattern, etc., the engineer must specify a refrigerant circuitry that operates well under the conditions of the application. To date, design engineers typically attempt to design refrigerant circuitries using their experience, some basic fundamental knowledge, and repeated simulations and experimentation. This rather time-intensive process can be automated effectively by using a design optimization tool employing evolutionary computation methods. The refrigerant circuitry is an important design consideration because it dictates the path of refrigerant through the heat exchanger, which, in turn, influences refrigerant mass flux in individual tubes, refrigerant-side heat transfer coefficient, saturation temperature drop, and how temperatures of individual tubes are distributed with respect to temperature of air passing through the heat exchanger. A coil can attain the highest capacity if the circuitry is designed so that the refrigerant-side heat transfer potential and the air-side heat transfer potential are optimally paired at every point within the heat exchanger. Solving this problem can be extraordinarily difficult, particularly if the airflow is not uniformly distributed through the coil.

Several researchers have used GAs to optimize the refrigerant circuitry of finned-tube heat exchangers. Wu et al. (2008a) applied a GA approach to optimize fin-tube heat exchangers. They used the model by Liu et al. (2004) to evaluate each model's fitness. Their objective was to minimize the length of the return bends while maintaining a required capacity. Since their objective was to minimize the total length of the tube connections, they developed functions that would continuously reduce the search space to designs with shorter tube connections; this greatly improved the amount of CPU time necessary for their optimization runs. Their results showed that they were able to reduce the amount of material expended on return bends by up to 40.3% (best case) below that used by a manufactured design, which represented a 1.3% decrease in tube material.

In Wu et al. (2008b), the same researchers applied a similar technique to maximize the heat transfer capacity. Their approach implements a somewhat conventional GA method with crossover, mutation, and correction algorithms that are based largely on domain knowledge. Their method reduces the searchable domain to designs that are deemed feasible, based on manufacturing rules. Their study demonstrated that it was possible to improve the capacity of a finned-tube heat exchanger using a GA optimization scheme to redesign the refrigerant circuitry, and that this method proved to be effective for recuperating lost capacity due to non-uniformly distributed airflow.

The application of GAs to optimize the refrigerant circuitry for maximizing the heat transfer capacity of finned-tube heat exchangers was proposed by Domanski et al. (2004, 2005) and Domanski and Yashar (2007). They used the model by Domanski (2008) to evaluate each model's fitness. These researchers employed a dual-mode evolutionary algorithm incorporated into an intelligent system for heat exchanger design (ISHED). Alternating between the two modes of operation provides a unique method for preventing the optimization process from converging to a local, as opposed to global, maximum, or failing to improve the designs. Among the evolutionary computation methods, the ISHED scheme is most related to GAs; however, it incorporates several unique features that put ISHED outside the GA class. This article presents a detailed discussion of the specific techniques incorporated into ISHED.

Evolutionary process and computation

Evolution alters characteristics of a species so that it can thrive in a given environment. Charles Darwin deduced from his observances that certain members of a species would have characteristics more favorable for survival and would therefore be more likely to reproduce and pass those characteristics onto their offspring, a process he called "natural selection." Evolution became better understood when theories of heredity were discovered by Gregor Mendel and others, which included the discrete nature of heredity factors.

One can envision the evolutionary process by considering a population of rabbits living among a population of foxes. The slower and less intelligent rabbits are more likely to be caught by foxes than the faster and smarter rabbits. Consequently, the smarter and faster rabbits have a higher likelihood of surviving to maturity and producing offspring than the slower and less intelligent rabbits. The subsequent generation of rabbits is therefore more likely to be their offspring, and their characteristics of speed and intelligence are preferentially passed on through natural genetics. The generation, as a whole, has therefore undergone a slight adaption that will help the species survive (Michalewicz, 1999).

In computing, evolutionary algorithms attempt to mimic the natural processes of evolution. In an evolutionary algorithm, each tested solution to an optimization problem is akin to a rabbit in the aforementioned description. The ultimate goal of the technique described here is to find an optimal solution for the refrigerant circuitry of a heat exchanger under specified operating conditions. The process works by examining a population of different refrigerant circuitry designs. Each design in the population is evaluated based on the goal of the optimization problem and is assigned a value of how well it performed; in this case, each design is measured by its simulated capacity. These "fitness values" are then used to determine which designs in the population should continue to live and which should perish. The surviving designs are then used to populate the next generation of candidate solutions by subjecting them to reproduction, mutation, and recombination. The entire process is iterated, evolving the population toward a better performing group of refrigerant circuitry designs each time. The evolutionary process is limited to problems where candidate solutions to an optimization problem are expressible as binary representations and the candidates can be evaluated by a fitness function.

ISHED

ISHED is a program that takes an evolutionary approach to optimizing the performance of finned-tube evaporators and condensers. In ISHED, the basic geometrical considerations of the heat exchanger (size, number of tubes, fin type, etc.) and the operating parameters (air velocity profile, inlet air temperature and humidity, refrigerant inlet quality, and saturation temperature) are fixed at the onset. ISHED searches through possible tube connections in order to maximize the heat exchanger's capacity at those conditions. Candidate refrigerant circuitries are expressed as linear binary representations, i.e., a string of numbers representing the connection sequence. The refrigerant circuitry and the fixed parameters constitute a complete input dataset for EVAP-COND (Domanski 2008), a simulation program for the performance of finned-tube evaporators and condensers.

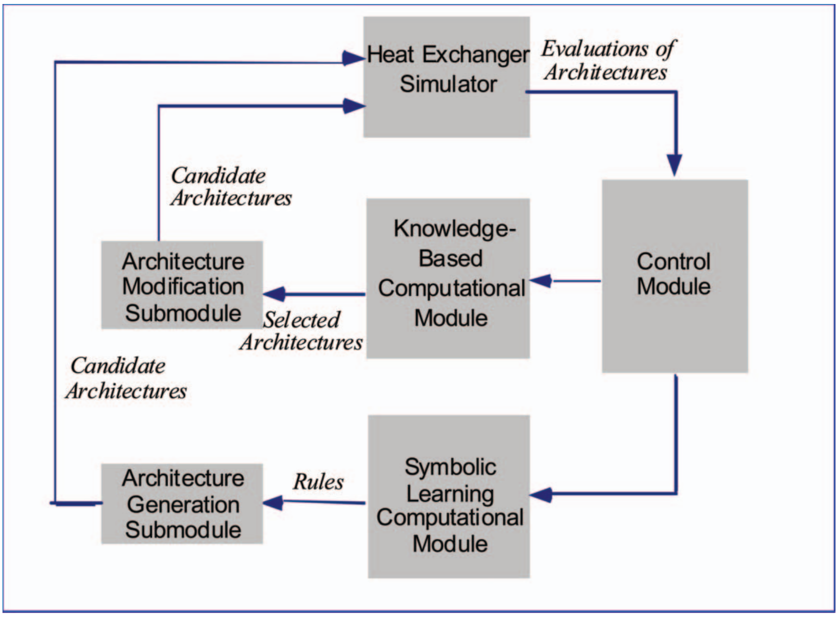
Heat Exchanger Simulator
Evaluations of Architectures
Candidate Architectures
Knowledge-Based Computational Module
Control Module
Architecture Modification Submodule
Selected Architectures
Candidate Architectures
Architecture Generation Submodule
Rules
Symbolic Learning Computational Module

Figure 1. Functional structure of ISHED (color figure available online).

The technique used in ISHED has several features that are common for all GA programs, but it also implements a few unique concepts. Consistent with a conventional GA program, ISHED operates on one population of refrigerant circuitries at a time. A population consists of a given number of circuitry designs. Each member of the population is evaluated by the fitness evaluator EVAP-COND, which simulates its performance and provides its capacity as a single numerical fitness value. The designs and their fitness values become an input for deriving the next generation of circuitry designs. The process is iterative, and it is repeated for the number of generations specified by the user.

The major difference between a basic GA program and ISHED is that the latter uses two independent modules—a knowledge-based evolutionary computation module and a symbolic-learning evolutionary module—to generate refrigerant circuitry designs for subsequent generations in the iterative scheme. ISHED uses a control module to determine which of the two operational modules is used at a given time. The control module monitors the progress of the optimization process from one generation to the next and switches between the two modules when the population no longer improves, both in terms of the best individual and the population overall. By using this method, the process is periodically “shaken up” by switching optimization approaches when the scheme fails to improve the performance of the refrigerant circuitry set over a series of generations. The functional structure of ISHED is shown in Figure 1.

Knowledge-based computational module

In the knowledge-based module, weighting factors are assigned to each candidate solution within a population based on its fitness value, in this case, the capacity. Solutions with higher capacity are assigned higher weighting factors. Members are then randomly selected from the weighted pool of solution candidates to provide a basis for generating new members in the next generation. These weighting factors therefore provide a means for improving the likeliness of selecting a well-performing design over a poorly performing design. This approach makes it more likely that the best-performing designs will survive the selection process and will influence the future generations; however, it does not systematically dismiss weaker designs. Designs that perform far better than others in its generation will likely be selected multiple times and will exert stronger influence on the next generation.

The knowledge-based module generates a new population of solutions one member at a time. The

Table 1. Design modifying operators for knowledge-based computational module.

Operator	Function
Split	Creates a split circuit from a single circuit by selecting a split point and connecting a downstream section of the original circuit to that point
Break	Creates two full circuits from a single circuit by breaking the circuit at a selected break point
Combine	Creates a single split circuit from two full circuits by combining them at a split point
Insert	Creates a single circuit from two full circuits by inserting one entire path into the other at a selected break point
Move-split	Moves the split point of a split path upstream or downstream from the original split point
Swap	Swaps the order of two adjacent tubes in a path
Intercross	Swaps the sources of two tubes in independent circuits
New-source	Assigns a new source to a randomly selected tube

new offspring members are formed by applying one of eight refrigerant circuit-specific operators to modify each selected architecture selected from the previous generation. These operators are not random, as in conventional GAs, but domain knowledge based; i.e., they will only perform changes that are deemed suitable according to the domain knowledge. The functions of the eight evaporator design-modifying operators used in ISHED are summarized in Table 1; there are eight analogous design-modifying operators for condensers.

Symbolic-learning computational module

The symbolic-learning-based module generates new designs in an entirely different way: by hypothesis formation and instantiation (Michalski 2000). In the symbolic-learning mode, ISHED tries to find an answer to the questions: *Why do some heat exchanger designs have better a capacity than others?* and *What are their characteristics?* Then, it uses hypothetical answers to these questions in order to create new designs. Specifically, the method works in three steps. Earlier work using the symbolic learning module on function optimization showed that it outperforms traditional GAs (Michalski 2000; Wojtusiak and Michalski 2006).

First, the symbolic-learning module identifies the high- and low-capacity designs from the current population of designs. The module divides the population members into three classes based on their fitness values: “good,” “bad,” and “indifferent.” The good and bad classes contain members of the population whose fitness are in the top and bottom 25% of the current generation’s fitness range, respectively.

Next, the module applies symbolic machine learning to generate general hypotheses to describe features of high-capacity designs in contrast to low-capacity ones. Specifically, the method uses an AQ-type rule learning method (e.g., Michalski et al. 1983) to induce a set of rules that describe high-capacity designs. For the purpose of symbolic learning, each design is described by listing a type of connection for each of its tubes (inlet, outlet, regular, split, and multi-split), and two derived attributes that describe the total numbers of inlets and outlets, respectively. Rules usually include only a small subset of these attributes that are sufficient to capture general characteristics of high-capacity designs. An example of such a rule is:

$$\begin{aligned}&[\text{design} = \text{high-capacity}] \Leftarrow \\ &\quad [\text{no\_inlets.no\_outlets} = 1] \;\&\; [\text{x3} = \text{outlet}] \\ &\qquad \&\; [\text{x11.x12} = \text{inlet or regular}] \;\&\; [\text{x1.x2.x4.} \\ &\qquad \text{x5.x6.x7.x 8.x9.x10} = \text{regular}].\end{aligned}$$

This rule can be paraphrased as *the design is high capacity if the number of inlets is one, the number of outlets is one, tube x3 is an outlet, and tubes x11 and x12 are either inlets or regular*, and so on. To induce rules, the symbolic-learning module starts with one high-capacity design. It tries to find all of its characteristics that distinguish it from low-performing designs and are shared among the most of other high-capacity designs. The best of these characteristics are then selected to form rules. From among high-capacity designs not described by existing rules, another design is selected, and the process is repeated until all high-performing designs are described by rules.

Finally, the module creates new designs by instantiating the rules learned in the previous step. This is done by generating circuitries consistent with the rules describing paths between inlets and outlets. The symbolic-learning module uses a set of heuristics based on special cases and plausible best connections to generate the design for the next generation. It needs to be noted that the problem of path generation is NP-hard; thus, no general efficient algorithm exists. In order to prevent the module from getting stuck, a rule is abandoned if the program fails to feasibly instantiate it after a user-specified maximum number of attempts.

There is no general limitation on the inductive-learning method guiding the evolutionary process in ISHED. In practice, however, an important factor for determining applicability is if learning methods are based on how accurately they can capture characteristics describing high-performing designs, as well as how easy it is to instantiate hypotheses in order to create new designs.

Optimization of a finned-tube evaporator

This section illustrates the potential performance improvements that can be attained using ISHED. This study used the simple single-slab evaporator shown in Figure 2, which was designed for use with R-22. This coil consisted of 54 rifled copper tubes arranged into three depth rows with lanced aluminum fins. The air velocity profile approaching this heat exchanger was determined both experimentally and computationally in a prior study (Yashar et al. 2011), which illustrates the non-uniformities inherent to even the most simple installation configurations. The heat exchanger was installed in a horizontal duct with its faces perpendicular to the duct walls. The detailed measurements of the airflow using particle image velocimetry showed that substantial non-uniformities were present. Specifically, the mounting brackets that held the heat exchanger in position within the duct caused disturbances to the airflow near the top and bottom of the coil. Note that prior work by Yashar et al. (2008) and Yashar and Domanski (2010) showed that the in situ airflow distribution is dictated by the geometry of the coil and the obstructions attached to and in proximity of the coil. Since these coils are typically packaged with the objects that influence airflow distribution (fan, condensation drip tray, etc.), the airflow distribution will be specific to the product line, and knowledge of the product's airflow distribution can provide a valid basis for the optimization problem.

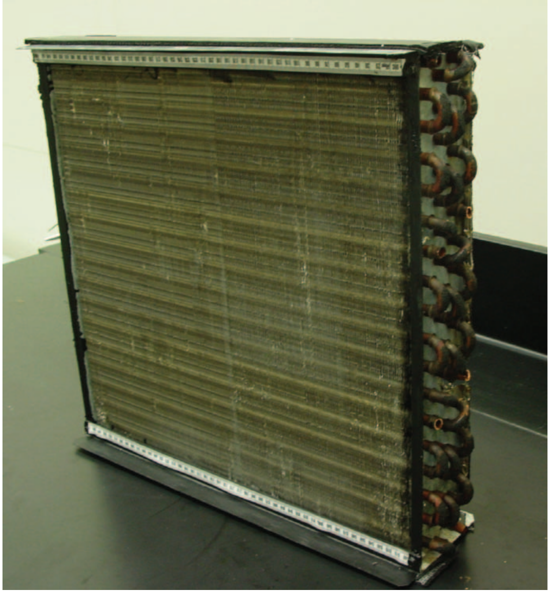

Figure 2. Original R-22 evaporator (color figure available online).

Figure 3 shows the EVAP-COND representation of the heat exchanger pictured in Figure 2 with the original circuitry and the true in situ airflow distribution. The circles numbered 42, 48, and 54 in the figure denote inlet tubes and the circles numbered 1, 7, and 13 denote outlet tubes. The airflow direction is bottom to top. The circuitry pattern for this evaporator is quite simple. The heat exchanger is comprised of three identical, parallel circuits, each consisting of 18 tubes in a zigzag pattern. This arrangement is typical of many production designs because it is easy to manufacture due to the short return bends and simple patterning. Furthermore, this design performs relatively well, since it attempts to arrange the tubes to maintain somewhat of a cross-counter flow heat exchanger configuration. However, characterization of the airflow distribution for this heat exchanger showed that the mounting brackets caused some air-side non-uniformities in the upper and lower regions of the coil. Because of this, each of the three circuits in the heat exchanger will operate with a different airflow distribution. The middle circuit (starting at tube 48

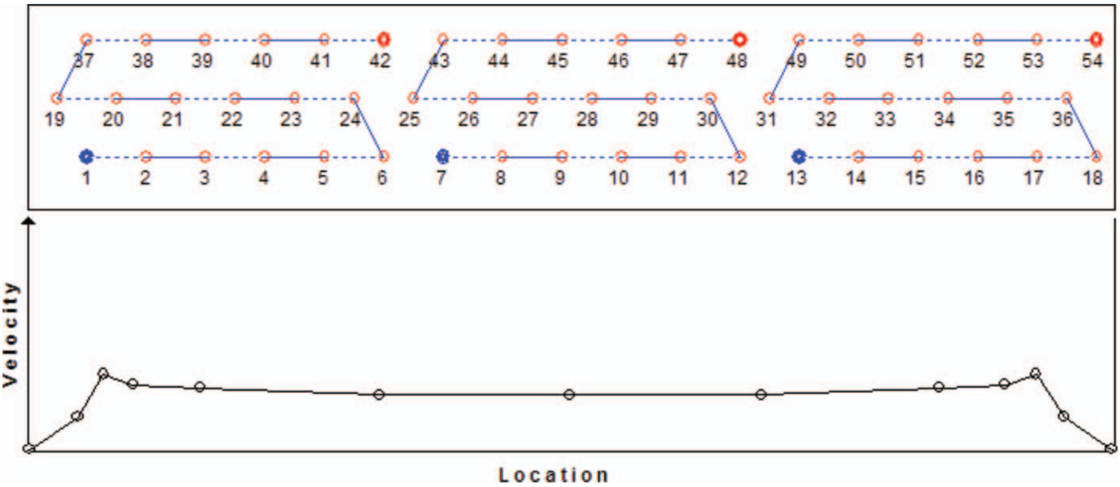

Figure 3. Original evaporator circuitry and in situ airflow distribution (color figure available online).

and ending at tube 7) will essentially operate with a uniform airflow distribution, while the other two circuits will operate with a lopsided distribution that includes a low- and high-flow region. Furthermore, the uneven air velocity profile incident on the third circuit is a mirror image of that incident on the first circuit; therefore the maldistribution influences different tubes in each sequence.

Since each circuit within the heat exchanger is designed with an identical pattern but subject to a different airflow distribution, it is obvious that each circuit will have a slightly different performance than the others. The performance of the heat exchanger is affected by the performance of each circuit, and since certain circuits perform better than others, the heat exchanger as a whole will not perform as well as it could if each circuit were operating optimally with its portion of the existing airflow distribution.

EVAP-COND was used to simulate the performance of this evaporator with the in situ airflow distribution at 26.7°C (80.0°F) with 50% RH, 101.325 kPa (1 atm), and flow rate of 0.3 m^3s^{-1} (640 CFM), resulting in an average velocity of 1.4 ms^{-1} (4.6 fts^{-1}). The refrigerant outlet saturation temperature and superheat were set to 7.0°C (45°F) and 5.0 K (9°F), respectively. The simulated capacity of the original design heat exchanger was 8.10 kW (27,600 Btu hr^{-1}). It is interesting to note that the simulation results showed how much the refrigerant exit condition varied between the three exit tubes; the refrigerant leaving tube #1 had 9.3 K (16.7°F) of superheat, that leaving tube #7 had 3.7 K (6.7°F)superheat, and that leaving tube #13 was two phase with a quality of 99%. This mismatch of refrigerant exit conditions is an indicator that it is possible to improve the performance of this heat exchanger by altering the refrigerant circuitry.

It should be noted that even if the designer of this coil had detailed knowledge of the airflow distribution, specifying an optimal circuitry design would have been a very difficult task considering the extremely large set of all possible circuitry. There are $54! = 2 \times 10^{71}$ possible configurations if only considering all of the single path connection sequences; the actual number is much greater due to the inclusion of architectures with multiple inlets and outlets.

Next, an exercise was performed to see how much improvement could be obtained by redesigning the refrigerant circuitry for operation with the specific in situ airflow distribution. The refrigerant circuitry was redesigned using each individual module within ISHED and then with the combination of both modules. For these optimization runs, a population size of 20 members was selected and each run carried out for 500 generations; therefore, each optimization run included the evaluation of 10,000 designs. The amount of computation time associated with each optimization run is directly proportional to the total number of designs evaluated; each of these optimization runs took less than 2 h using a 3-GHz processor. At the end of the process, each optimization run resulted in designs that were superior to the original design. Operating solely with the knowledge-based computational module resulted in a capacity of 8.51 kW (29,000 Btu hr^{-1}), the

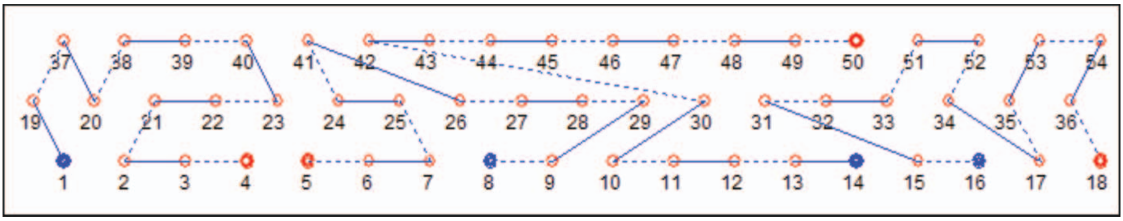

Figure 4. Design produced by knowledge based computational module, $Q = 8.51$ kW (color figure available online).

symbolic learning computational module resulted in the best design having a capacity of 8.40 kW (28,700 Btu hr^{-1}), and the optimization run using combined efforts of both modules resulted in a capacity of 8.62 kW (29,400 Btu hr^{-1}). The resulting circuitry architectures are shown in Figures 4, 5, and 6.

The designs produced by each single module consisted of four circuits, while the best design produced using both modules consisted of five. It is interesting to note that each of these optimized designs had more circuits than the original design. If the circuitry is designed so that the refrigerant is routed through locations in a manner that optimally matches the refrigerant's heat transfer potential to the air's heat transfer potential, balances the load between each circuit, and is capable of boiling all of the refrigerant, it will perform well. All of the optimized designs did this; however, the best performing design found using the dual-mode optimization scheme has one more circuit than the other designs, which proved to be beneficial for the studied evaporator. It appears that the dual-mode optimization was able to successfully explore five circuit designs, while the single-mode optimization runs converged on good, but not the best, four-circuit designs. The best five-circuit design accommodates a greater total mass flow rate of refrigerant with acceptable pressure drop, since the length of each circuit is smaller.

Figure 7 shows the progression of the highest capacity attained by the best design in each population throughout each optimization run. In two of the three cases, one of the randomly generated designs within the initial population outperformed the original design. This is not surprising, since this initial population is generated by a fairly comprehensive design algorithm, and, as previously noted, the original circuitry was not designed to account for the in situ airflow distribution.

The optimization runs using a single module produced results that exceeded the capacity of the original design with an improvement of 2.6% for the symbolic-learning module and 4.8% for the knowledge-based module. The combined-mode optimization approach, however, resulted in a design that performed better than either of the single-mode approaches, with an improvement of 6.5% over the original circuitry design.

The progression of the best-performing designs presented in Figure 7 shows that the incremental improvements slow down after a certain point in each optimization run. This is because, as with any search technique, repetitive iterations often tend to hone in on a local maximum. The approach of using a combined mode of operation provides a means for changing direction when one of the methods fails to continue improving. This switching back and forth provides a "kick-out" method to avoid local maxima within the search space and allows the optimization run to continue to improve the designs.

It can be noted that a change in operating parameters (e.g., air temperature, flow rate, distribution over the coil face, or refrigerant outlet saturation pressure) could result in different optimized refrigerant circuitries. For example, optimization runs were performed for this heat exchanger with a similar set of operating parameters except for variations in the

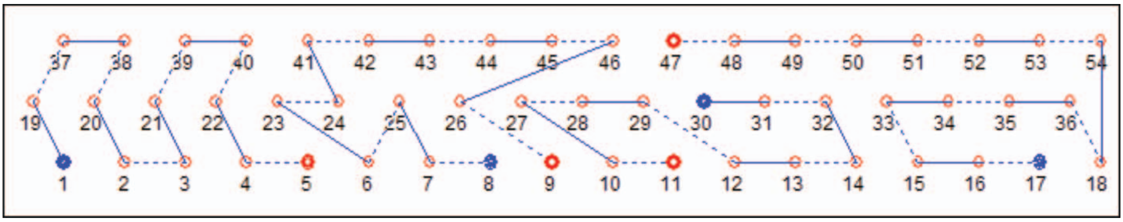

Figure 5. Design produced by symbolic learning computational module, $Q = 8.40$ kW (color figure available online).

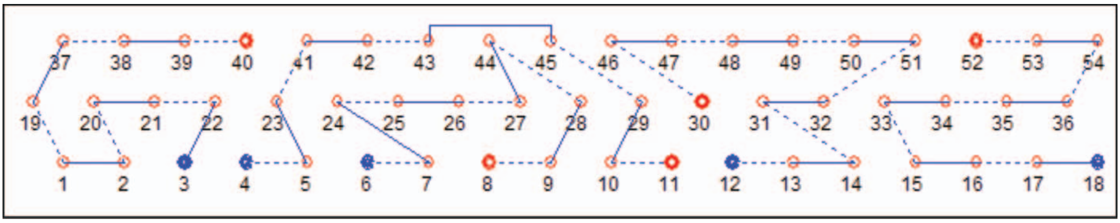

Figure 6. Design produced by combined approach with both computational modules, $Q = 8.62$ kW (color figure available online).

refrigerant outlet saturation temperature. this evaporator's refrigerant circuitry was optimized using 4.0°C and 10.0°C (39°F and 50°F) as the refrigerant outlet saturation temperature, as opposed to the 7.0°C (45°F) used in the first set of tests. The performance of the heat exchangers with the refrigerant circuitries returned from these runs were evaluated at all three refrigerant saturation temperatures, and the capacities differed by as much as 2.6%. This indicates the importance of optimizing the circuitry using the saturation temperature at which the heat exchanger is expected to operate most of the time. Previous studies (Wu et al. 2008b; Domanski et al. 2004; Domanski and Yashar, 2007) noted that much larger capacity differences would result for different airflow rates and airflow distributions.

Summary

This article has presented an evolutionary computation approach to optimize the refrigerant circuitry of a finned-tube heat exchanger. The approach presented uses two independent modules to optimize circuitry designs as the process marches forward.

The first method employs the knowledge-based evolutionary computation module. This module selects designs from the previous population using a

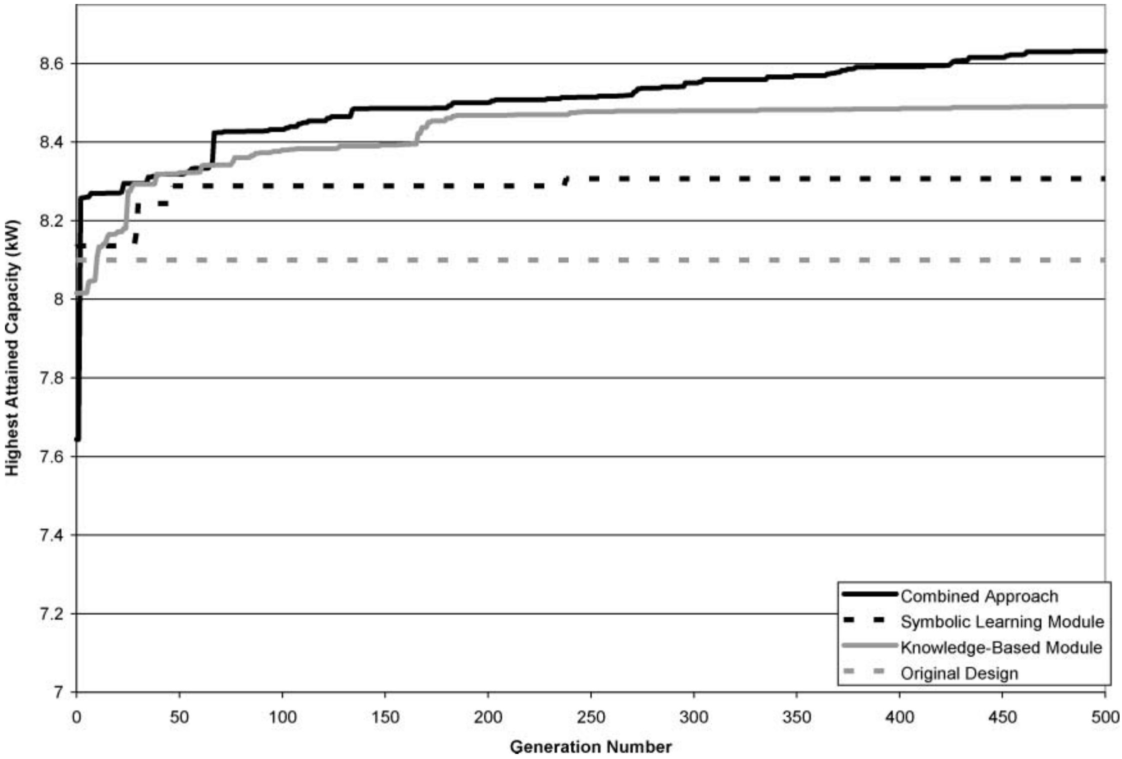

Figure 7. Progression of highest capacity attained throughout three optimization runs.

performance-weighted selection approach. Then the module generates members for the next population by altering certain characteristics of each selected design to produce an "offspring" design that is similar to its parent design.

The second method employs the symbolic-learning-based evolutionary computation module. This module works in an entirely different manner, by hypothesis formation and instantiation. The symbolic learning module examines the entire population of designs from one generation to identify attributes that are common among the well-performing designs and attributes that are common to the poorly performing designs. The module then uses these attributes to generate rules that it uses to design the members of the next population.

The combined approach employed by ISHED results in designs that are better than either of the two modes can produce individually. This is because after time, the individual approaches cease to produce radically different designs that are necessary to move the optimization process forward. The single-module approach, therefore, often becomes stale, which is a problem with any systematic optimization process. The combined-module approach provides a method to change course and switch modules when the one module fails to demonstrate improvements in the designs.

The capabilities of each individual optimization module and the combined-module approach were demonstrated by redesigning a single-slab R-22 evaporator for which detailed airflow distribution information was available. The test case presented in this article showed capacity improvements of 2.6% and 4.8% using each of the individual modules, while the combined approach resulted in a design that performed 6.5% better than the original design. This performance improvement can be achieved if an intelligent refrigerant circuitry optimization scheme is used in conjunction with detailed airflow distribution knowledge.

References

Aute, V., R. Radermacher, and M.V. Naduvath. 2004. Constrained multi-objective optimization of a condenser coil using evolutionary algorithms. International Refrigeration and Air Conditioning Conference at Purdue, July 12–15. West Lafayette, IN.

Casson, V., A. Cavallini, L. Cecchinato, D. Del Col, L. Doretti, E. Fornasieri, L. Rossetto, and C. Zilio. 2002. Performance of finned coil condensers optimized for new HFC refrigerants. *ASHRAE Transactions* 108(2):517–27.

Domanski, P.A. 2008. EVAP-COND—simulation models for finned-tube heat exchangers, Version 2.3. National Institute of Standards and Technology; Gaithersburg, MD, USA. http://www2.bfrl.nist.gov/software/evap-cond./

Domanski, P.A., and D. Yashar. 2007. Optimization of finned-tube condensers using an intelligent system. *International Journal of Refrigeration* 30(3):482–8.

Domanski, P.A., D. Yashar, K.A. Kaufman, and R.S. Michalski. 2004. Optimized design of finned-tube evaporators using learnable evolution methods. *HVAC&R Research* 10(2):201–12.

Domanski, P.A., D. Yashar, and M. Kim. 2005. Performance of a finned-tube evaporator optimized for different refrigerants and its effect on system efficiency. *International Journal of Refrigeration* 28(6):820–7.

Fogel, D.B. 1994. Asymptotic convergence properties of genetic algorithms and evolutionary programming, *Cybernetics and Systems* 25(3), 389–407.

Goldberg, D.E. 1989. A Gentle Introduction to Genetic Algorithms. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison Wesley Longman, Inc.

Jiang, H., V. Aute, and R. Radermacher. 2006. CoilDesigner: general-purpose simulation and design tool for air-to-refrigerant heat exchangers. *International Journal of Refrigeration* 29(2006):601–10.

Liang, S.Y., T.N. Wong, and G.K. Nathan. 2001. Numerical and experimental studies of refrigerant circuitry of evaporator coils. *International Journal of Refrigeration* 24(8):823–33.

Liu, J., W.J. Wei, G.L. Ding, C.L. Zhang, M. Fukuya, K.J. Wang, and T. Inagaki. 2004. A general steady state mathematical model for fin-and-tube heat exchanger based on graph theory. *International Journal of Refrigeration* 27(8):965–73.

Michalewicz, Z. 1999. *Genetic algorithms + Data Structures = Evolution Programs, 2nd extended ed*. Berlin, Germany: Springer-Verlag.

Michalski, R.S. 2000. Learnable evolution model: Evolutionary process guided by machine learning. *Machine Learning* 38:9–40.

Michalski, R.S., T.J. Carbonell, and T.M. Mitchell. 1983. *A Theory and Methodology of Inductive Learning. Machine Learning: An Artificial Intelligence Approach*, 83–134, Palo Alto, CA: TIOGA Publishing Co.

Qiao, H., V. Singh, V. Aute, and R. Radermacher. 2010. Optimization of fin density for air cooled heat exchanger. International Refrigeration and Air Conditioning Conference at Purdue, July 10–15. West Lafayette, IN.

Wang, C.C., J.Y. Jang, C.C. Lai, and Y.J. Chang. 1999. Effect of circuit arrangement on the performance of air-cooled condenser. *International Journal of Refrigeration* 22(4): 275–82.

Wojtusiak, J., and R.S. Michalski. 2006. The LEM3 implementation of learnable evolution model and its testing on complex function optimization problems. *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2006), Seattle, WA, July 8–12*.

Wu, Z., G. Ding, K. Wang, and M. Fukaya. 2008a. Application of a genetic algorithm to optimize the refrigerant circuit of fin-and-tube heat exchangers for maximum heat transfer or shortest tube. *International Journal of Thermal Sciences* 47(8):985–97.

Wu, Z., G. Ding, K. Wang, and M. Fukaya. 2008b. Knowledge-based evolution method for optimizing refrigerant circuitry of fin-and-tube heat exchangers. *HVAC&R Research* 14(3):435–51.

Yashar, D.A., and P.A. Domanski. 2010. Air flow distribution through an A-shaped evaporator under dry and wet coil conditions. International Refrigeration and Air Conditioning Conference at Purdue, July 10–15, West Lafayette, IN.

Yashar, D.A., P.A. Domanski, and H.H. Cho. 2008. Measurement of air-velocity profiles for finned-tube heat exchangers using particle image velocimetry. International Refrigeration and Air Conditioning Conference at Purdue, July 12–17. West Lafayette, IN.

Yashar, D.A., P.A. Domanski, and H.H. Cho. 2011. An experimental and computational study of approach air distribution for a finned-tube heat exchanger. *HVAC&R Research* 17(1):76–85.