Optimization of three-dimensional branching networks of microchannels for thermal management of microelectronics

Ricardo Ardila
Florida International University

DOI: 10.25148/etd.FI14032335
Follow this and additional works at: https://digitalcommons.fiu.edu/etd
Part of the Mechanical Engineering Commons

Recommended Citation
https://digitalcommons.fiu.edu/etd/1295

This work is brought to you for free and open access by the University Graduate School at FIU Digital Commons. It has been accepted for inclusion in FIU Electronic Theses and Dissertations by an authorized administrator of FIU Digital Commons. For more information, please contact dcc@fiu.edu.
OPTIMIZATION OF THREE-DIMENSIONAL BRANCHING NETWORKS OF
MICROCHANNELS FOR THERMAL MANAGEMENT OF MICROELECTRONICS

A thesis submitted in partial fulfillment of the
requirements for the degree of
MASTER OF SCIENCE
in
MECHANICAL ENGINEERING
by
Ricardo Ardila

2009
To: Dean Amir Mirmiran  
College of Engineering and Computing

This thesis, written by Ricardo Ardila, and entitled Optimization of Three-Dimensional Branching Networks of Microchannels for Thermal Management of Microelectronics, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this thesis and recommend that it be approved.

______________________________
W. Kinzy Jones

______________________________
Igor Tsukanov

______________________________
George S. Dulikravich, Major Professor

Date of Defense: November 12, 2009

The thesis of Ricardo Ardila is approved.

______________________________
Dean Amir Mirmiran  
College of Engineering and Computing

______________________________
Dean George Walker  
University Graduate School

Florida International University, 2009
DEDICATION

I dedicate this work to my parents, brother and sister. All of what I am and will ever accomplish, I owe to them.
AKNOWLEDGMENTS

I would like to thank my entire thesis committee for their mentoring and guidance. Especially, I would like to express the profound gratitude I feel for my major Professor George S. Dulikravich for his continuous support. I feel also obliged to acknowledge all my colleague researchers and friends at MAIDROC.
ABSTRACT OF THE THESIS

OPTIMIZATION OF THREE-DIMENSIONAL BRANCHING NETWORKS OF MICROCHANNELS FOR THERMAL MANAGEMENT OF MICROELECTRONICS

by

Ricardo Ardila

Florida International University, 2009

Miami, Florida

Professor George S. Dulikravich, Major Professor

The aim of this work is to present a methodology to develop cost-effective thermal management solutions for microelectronic devices, capable of removing maximum amount of heat and delivering maximally uniform temperature distributions. The topological and geometrical characteristics of multiple-story three-dimensional branching networks of microchannels were developed using multi-objective optimization.

A conjugate heat transfer analysis software package and an automatic 3D microchannel network generator were developed and coupled with a modified version of a particle-swarm optimization algorithm with a goal of creating a design tool for 3D networks of optimized coolant flow passages. Numerical algorithms in the conjugate heat transfer solution package include a quasi-1D thermo-fluid solver and a steady heat diffusion solver, which were validated against results from high-fidelity Navier-Stokes equations solver and analytical solutions for basic fluid dynamics test cases.

Pareto-optimal solutions demonstrate that thermal loads of up to 500 W/cm² can be managed with 3D microchannel networks, with pumping power requirements up to 50% lower with respect to currently used high-performance cooling technologies.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Problem Statement</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Research Objective</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Methodology</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Literature Review</td>
<td>4</td>
</tr>
<tr>
<td>PARTICLE SWARM MULTI-OBJECTIVE OPTIMIZATION</td>
<td>8</td>
</tr>
<tr>
<td>2.1 Multi-Objective Optimization Overview</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Multi-Objective Optimization Definitions</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Multi-Objective Particle-Swarm Optimization Algorithm</td>
<td>12</td>
</tr>
<tr>
<td>2.4 MOPSO Performance Parameters Analysis</td>
<td>16</td>
</tr>
<tr>
<td>2.5 Numerical Experiments</td>
<td>23</td>
</tr>
<tr>
<td>2.6 MOPSO Adaptation for Parallel Computing</td>
<td>28</td>
</tr>
<tr>
<td>GENERATION OF 3D MICROCHANNEL NETWORKS</td>
<td>31</td>
</tr>
<tr>
<td>3.1 3D Microchannel Network Definition</td>
<td>31</td>
</tr>
<tr>
<td>3.2 Automatic Network Generation</td>
<td>35</td>
</tr>
<tr>
<td>3.3 Fluid Domain Preparation</td>
<td>37</td>
</tr>
<tr>
<td>3.4 Solid Domain Meshing and Preparation</td>
<td>38</td>
</tr>
<tr>
<td>CONJUGATE HEAT TRANSFER SOLUTION PACKAGE</td>
<td>45</td>
</tr>
<tr>
<td>4.1 Overview</td>
<td>45</td>
</tr>
<tr>
<td>4.2 Conjugate Heat Transfer Formulation</td>
<td>45</td>
</tr>
<tr>
<td>4.3 Flow-Field Analysis</td>
<td>47</td>
</tr>
<tr>
<td>4.3.1 Flow-Field Solver Theory</td>
<td>47</td>
</tr>
<tr>
<td>4.3.2 Quasi-1D Thermo-Fluid Solver Description</td>
<td>52</td>
</tr>
<tr>
<td>4.3.3 Quasi-1D Thermo-Fluid Solver Validation</td>
<td>55</td>
</tr>
<tr>
<td>4.4 Heat Conduction Analysis</td>
<td>58</td>
</tr>
<tr>
<td>4.4.1 Steady Heat Diffusion Mathematical Model</td>
<td>59</td>
</tr>
<tr>
<td>4.4.2 Heat Diffusion Solver Description</td>
<td>61</td>
</tr>
<tr>
<td>4.4.3 Heat Diffusion Solver Validation</td>
<td>63</td>
</tr>
<tr>
<td>4.5 Thermal Boundary Condition Transfer Module</td>
<td>66</td>
</tr>
<tr>
<td>4.6 Additional Features</td>
<td>68</td>
</tr>
<tr>
<td>4.7 Objectives Calculation</td>
<td>69</td>
</tr>
<tr>
<td>OPTIMIZED THERMAL MANAGEMENT</td>
<td>71</td>
</tr>
<tr>
<td>5.1 Definition of the Optimization Problem</td>
<td>71</td>
</tr>
<tr>
<td>5.2 Optimization Study Results</td>
<td>72</td>
</tr>
<tr>
<td>CONCLUSIONS</td>
<td>114</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1. The Concept of Dominance.</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2. The PSO Algorithm</td>
<td>14</td>
</tr>
<tr>
<td>Figure 3. Topology of Michalewicz's Function</td>
<td>17</td>
</tr>
<tr>
<td>Figure 4. Test Function's Minima</td>
<td>18</td>
</tr>
<tr>
<td>Figure 5. Influence of Individuality Parameter on Objective Function Convergence with Low Inertial Factors of 0.0 and 0.2 (a,b)</td>
<td>18</td>
</tr>
<tr>
<td>Figure 6. Influence of Individuality Parameter on Objective Function Convergence with Medium Inertial Factors of 0.4 and 0.6 (a,b)</td>
<td>19</td>
</tr>
<tr>
<td>Figure 7. Influence of Individuality Parameter on Objective Function Convergence with high Inertial Factors of 0.8 and 1.0 (a,b)</td>
<td>20</td>
</tr>
<tr>
<td>Figure 8. Influence of Inertial Parameter on Objective Function Convergence with Low Individuality Factors of 1.0 and 1.2 (a,b)</td>
<td>21</td>
</tr>
<tr>
<td>Figure 9. Influence of Inertial Parameter on Objective Function Convergence with Medium Individuality Factors of 1.4 and 1.6 (a,b)</td>
<td>21</td>
</tr>
<tr>
<td>Figure 10. Influence of Inertial Parameter on Objective Function Convergence with High Individuality Factors of 1.8 and 2.0 (a,b)</td>
<td>22</td>
</tr>
<tr>
<td>Figure 11. MOPSO Validation Against ZDT-1</td>
<td>26</td>
</tr>
<tr>
<td>Figure 12. MOPSO Validation Against ZDT-2</td>
<td>26</td>
</tr>
<tr>
<td>Figure 13. MOPSO Validation Against ZDT-3</td>
<td>27</td>
</tr>
<tr>
<td>Figure 14. MOPSO Validation Against ZDT-4</td>
<td>27</td>
</tr>
<tr>
<td>Figure 15. MOPSO Validation Against ZDT-6</td>
<td>28</td>
</tr>
<tr>
<td>Figure 16. MOPSO Adaptation for Parallel Computing</td>
<td>29</td>
</tr>
<tr>
<td>Figure 17. Sample 3D Microchannel Network</td>
<td>31</td>
</tr>
<tr>
<td>Figure 18. Sample Planar Network</td>
<td>32</td>
</tr>
</tbody>
</table>
Figure 19. Sample 3D Multi-Story Network with Mirrored Planar Networks.............. 33
Figure 20. Distribution System of Sample Network..................................................... 34
Figure 21. Example Planar Network with 4-way Mirroring......................................... 34
Figure 22. Void Volume. ............................................................................................... 39
Figure 23. Extraction of Internal Domain from Target Domain.................................... 39
Figure 24. Resulting Hybrid Mesh............................................................................. 40
Figure 25. Sample Planar Network............................................................................ 42
Figure 26. Surface Mesh for Sample Network ............................................................. 42
Figure 27. Transformation of Last Cut Plane to Conform to a T-junction.................... 43
Figure 28. Transformation of Last Cut Plane to Conform to an Elbow Junction......... 43
Figure 29. Validation of COOLNET for Fanno Flow (Pressure Field)....................... 56
Figure 30. Validation of COOLNET for Fanno Flow (Velocity Field)....................... 56
Figure 31. Validation of COOLNET for Fanno Flow (Mass Flow Rate).................... 57
Figure 32. Validation of COOLNET for Poiseuille Flow.......................................... 58
Figure 33. Validation of OpenFOAM for 1D heat Diffusion................................. 64
Figure 34. Calculated Temperature Field with OpenFOAM for 2D Heat Diffusion........ 65
Figure 35. Analytic Temperature Field Solution for 2D Heat Diffusion Test Case........ 65
Figure 36. Normalized Error Distribution in OpenFOAM Calculation..................... 66
Figure 37. CHETSOLP/MOPSO Integration.............................................................. 70
Figure 38. Optimization Problem Definition........................................................... 72
Figure 39. Pareto Front .............................................................................................. 74
Figure 40. Pareto-Optimal Solutions After 75 MOPSO Generations......................... 74
Figure 85. Fourth Floor Branching Topology for Pareto-solution 23. ......................... 103

Figure 86. Hydraulic Diameters Along Branching Levels in all Floors for Solution 23. ...................................................................................................................... 104

Figure 87. Top Surface Temperature Distribution for Pareto Solution 19. ................. 104

Figure 88. First Floor Branching Topology for Pareto-solution 19. ............................ 105

Figure 89. Second Floor Branching Topology for Pareto-solution 19. ....................... 106

Figure 90. Third Floor Branching Topology for Pareto-solution 19. .......................... 106

Figure 91. Fourth Floor Branching Topology for Pareto-solution 19. ........................... 107

Figure 92. Hydraulic Diameters Along Branching Levels in all Floors for Solution 19. ...................................................................................................................... 108

Figure 93. Top Surface Temperature Distribution for Pareto Solution 76. .................... 108

Figure 94. First Floor Branching Topology for Pareto-solution 76. ............................ 109

Figure 95. Second Floor Branching Topology for Pareto-solution 76. ....................... 109

Figure 96. Third Floor Branching Topology for Pareto-solution 76. ............................ 110

Figure 97. Fourth Floor Branching Topology for Pareto-solution 76. ........................... 110

Figure 98. Hydraulic Diameters Along Branching Levels in all Floors for Solution 76. ...................................................................................................................... 111

Figure 99. Temperature Distribution at Top Face with Straight-Through Passages Arrangement. ........................................................................................................... 112
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3BNGEN</td>
<td>Three-Dimensional Branching Network Generator</td>
</tr>
<tr>
<td>BCTM</td>
<td>Boundary Condition Transfer Module</td>
</tr>
<tr>
<td>CFD</td>
<td>Computational Fluid Dynamics</td>
</tr>
<tr>
<td>CHETSOLP</td>
<td>Conjugate Heat Transfer Solution Platform</td>
</tr>
<tr>
<td>COOLNET</td>
<td>Cooling Networks Solver</td>
</tr>
<tr>
<td>FVM</td>
<td>Finite Volume Method</td>
</tr>
<tr>
<td>MOHO</td>
<td>Multi-Objective Hybrid Optimizer</td>
</tr>
<tr>
<td>MOPSO</td>
<td>Multi-Objective Particle-Swarm Optimizer</td>
</tr>
<tr>
<td>MPP</td>
<td>Modified Predator Prey</td>
</tr>
<tr>
<td>NSDE</td>
<td>Non-Dominated Sorting Differential Evolution</td>
</tr>
<tr>
<td>NSE</td>
<td>Navier-Stokes Equations</td>
</tr>
<tr>
<td>PSO</td>
<td>Particle-Swarm Optimization</td>
</tr>
<tr>
<td>SPEA</td>
<td>Strength Pareto Evolutionary Algorithm</td>
</tr>
<tr>
<td>SVD</td>
<td>Singular Value Decomposition</td>
</tr>
<tr>
<td>ZDT</td>
<td>Zitzler-Deb-Thiele [28]</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

1.1 Problem Statement

The development of high-performance technologies in the computer and electronics industries is currently constrained by the ability of cooling systems to deliver adequate thermal management solutions. Commercial electronics have delineated a trend of exponential increase of heat dissipation over the past decade and up to 10 times greater heat fluxes, with respect to current devices, are expected to result in next generation microelectronics.

The thermal load produced by a currently used conventional computer chip is of the order of 100 W/cm² [1], causing large temperature gradients and non-uniform temperature distributions. There are critical locations where excessive thermal loading may result in a performance decrease and can even evolve to device failure. Typically, such amount of heat dissipation has been handled with traditional cooling mechanisms implementing a combination of heat sinks and fans. However, as the power density of microchips increases, the need for more sophisticated cooling schemes capable of managing high heat fluxes, and producing uniformly cooled surfaces becomes indisputable.

Effective cooling techniques employing coolant flow have been developed to address the above described issue. One of the most compelling coolant flow techniques nowadays, in addition to jet impingement, is microchannel heat sinks. The heat transfer coefficients derived from fluid flow through passages of very small hydraulic diameter are extremely large. Nevertheless, the microchannel cooling scheme yields undesired
pressure drop and high temperature gradients along flow direction thus violating one of
the most crucial requirements of high-performance microelectronics which is uniform
cooling.

Little efforts have been invested to optimize the topology of branching networks
of microchannels in order to maximize their performance and assess their ability to solve
the problem of thermal management in microelectronics. Furthermore, existing cooling
passage-based technologies have been constrained to 2D (planar) networks. The impact
of a cooling scheme involving three-dimensional networks of microchannels has not been
studied yet, and optimization of such 3D networks has not been carried out. The
motivation of this research is to optimize the topology of 3D networks of branching
microchannels as a potential solution for the thermal management problem of next
generation high-performance microelectronics.

1.2 Research Objective

This research constitutes an endeavor to optimize 3D microchannel networks in
order to provide a reliable, cost-effective and powerful thermal management
methodology that meets the cooling requirements of future generation microelectronics,
thereby promoting further development in this industry.

It is the aim of this work to execute a multi-objective optimization study on the
topological and geometrical characteristics of 3D microchannel networks in order to
deliver uniform surface temperature distribution (at thermal load’s surface), at low mass
flow rates and with minimized pressure drop of the cooling fluid.
1.3 Methodology

A multi-objective particle-swarm optimization algorithm (MOPSO) was developed to carry out this work. Particle-swarm optimization is a stochastic technique inspired by population based behavior; it utilizes a set of multiple candidate designs to follow an iterative procedure which leads to a final set of designs, referred to as the Pareto front, which comprised of the best designs. During the iterative process, the characteristics of the set members are used to assess their strength through the process of function evaluation; the result of the function evaluation is the metric to determine which members are stronger than others.

The process of function evaluation, for the problem at hand, is comprised of the calculation of the pressure drop of the pumped fluid, the mass flow rate in the microchannel network and the temperature uniformity at the heated surface; these metrics are the optimization objectives. In order to compute these, computational fluid dynamics (CFD) is used to model the conjugate heat transfer phenomena in a solid substrate piece, internally cooled by a 3D network of microchannels. Hence, the flow-field in the microchannels network and the temperature field in the solid substrate are obtained and then analyzed to determine the three objectives mentioned above.

A conjugate heat transfer analysis software package was developed. It is composed of quasi-1D thermo-fluid solver, a steady heat diffusion solver and a thermal boundary condition transferring module. The numerical algorithms involved in this solution package were validated against results from high-fidelity solvers and analytical solutions for basic CFD test cases.
For the purposes of optimization, automatic generation of the 3D branching networks is imperative. An automatic microchannel network generator was developed and coupled with the conjugate heat transfer solution package, thus completing a fully self-sufficient platform for geometry generation and thermo-fluid fields’ computation and processing. Such platform was invoked during execution of the MOPSO optimizer, as the function evaluation process.

1.4 Literature Review

The work herein presented involves optimization techniques and electronics thermal management via microchannel flow; past research in both these fields is reviewed in this section to place this work in context. Three major areas are reviewed: 1) multi-objective optimization, 2) microchannel heat sinks for thermal management, and 3) optimization of passage-based networks for enhanced internal-cooling.

Compact liquid-cooled heat exchangers integrated to silicon substrates to address large power densities in microelectronics were proposed by Tuckerman and Pease [2] in 1981, triggering the experimental and numerical investigation efforts [3-10] on microchannel heat sinks. Experimental work related to microchannel heat sinks progressed considerably in 1994, when Bowers and Mudawar [11] achieved up to 3,000 W/cm² heat dissipation with single-phase flow of water in micro-tubes, due to the advantage that water’s thermo-physical properties represent. However, different approaches needed to be sought in the field of electronics cooling, involving dielectric coolants.

Numerical investigation of microchannel heat sinks applied as an electronic cooling solution was introduced by Kim et al. [12] in 1999, who reported analytical
solutions of velocity and temperature profiles in the microchannel heat sink, resulting from the solid fin and the fluid comprising the conjugate heat transfer problem. A year later, Fedorov and Viskanta [13] investigated the flow and conjugate heat transfer problems in microchannel-based heat sinks for electronic packaging applications, providing insight to the complex heat/flow pattern yielded by the combined convection-conduction effect in 3D geometries.

In 2006, Hong et al. [14] presented a great effort to enhance the cooling uniformity of microchannel heat sinks for microelectronic devices. Their work concentrated on the design of fractal tree-like networks to reduce flow rate and pressure drop, while delivering a uniformly cooled profile. Two years later, in 2008, Sung and Mudawar [15] presented a hybrid cooling scheme featuring jet impingement and microchannel flow, with a 1000 W/cm² heat capacity. Finally, computational and experimental investigation of pressure losses and heat transfer in microchannel networks containing T-type junctions were conducted by Haller et al. [16].

Let us now review the evolution of optimization algorithms.

Single-objective optimization through classical gradient-based methods is widely used and is the oldest in the optimization field; classical approaches are covered in the book by Vanderplaats [17]. However, further development was necessary to address complex problems involving multiple objectives and function evaluation that cannot be handled with gradient-based search logic; multi-objective evolutionary optimization algorithms emerged as alternative optimization techniques. Some of the most popular algorithms being the genetic algorithms, differential evolution and particle swarm. In his book, Deb [18] thoroughly covers the concepts of multi-objective optimization with
evolutionary algorithms. He stresses that in complex multi-variable multi-objective problems the optimum should be sought in a Pareto-optimal sense, leading to a group of non-dominated designs.

Particle-swarm optimization (PSO) was first introduced in 1995 by Eberhardt and Kennedy and it was built based on the concept of swarm intelligence [19]. It makes use of the social-psychological principles to model a population’s behavior and their convergence towards an objective of interest. Although it was originally proposed as a single-objective optimization technique, multiple efforts can be found in literature attempting to translate it to a multi-objective technique, e.g. Parsopoulos and Vrahitis [20] used a weighed sum approach to compute a global single objective. However, weight factor selection is strictly problem-dependent and in many cases this approach can become counterproductive since it may constrain population convergence over a limited region of the Pareto front.

One of the latest trends in the optimization field is that of developing hybrid optimizers capable of invoking multiple algorithms simultaneously and being able to monitor which is performing best; such tactic yields increased optimization robustness. Moral and Dulikravich [21] developed a multi-objective evolutionary hybrid optimizer (MOHO) offering automatic switching among its 3 constituent algorithms: 1) non-dominated sorting differential evolution (NSDE), 2) strength Pareto evolutionary algorithm (SPEA), and 3) multi-objective particle swarm (MOPSO). Moreover, Chowdhury and Dulikravich [22] developed a modified predator-prey (MPP) optimizer depicting similar performance capabilities to that of Moral’s hybrid optimizer [21].
Let us now focus on reviewing research related to applications of optimization on passage-based internal cooling systems.

One of the first vestiges of the application of optimization methods to improve passage topologies to augment thermo-fluid systems’ efficiency was in the design of gas turbine blades. Intensive work was performed to maximize the cooling capacity of the passage-based networks by means of optimizing the passages arrangement and the distribution of heat transfer enhancing devices. Martin and Dulikravich [23] presented a fully automated program for inverse design and optimization of internal convectively cooled gas turbine blades, which was successfully validated against experimental results from Pratt & Whitney. Jelisavcic et al. [24] applied hybrid evolutionary optimization to the same concept of passage optimization for turbo-machinery blades in 2005. Subsequently, Gonzales et al. [25] executed relevant work comprising 2D channel network optimization in 2007.
CHAPTER II
PARTICLE SWARM MULTI-OBJECTIVE OPTIMIZATION

2.1 Multi-Objective Optimization Overview

Multi-objective optimization problems can be defined by the general formulation in Eq. (2.1). A number of objective functions are to be minimized or maximized, subject to certain number of constraints, for variable values within a certain range.

\[
\begin{align*}
\text{Min} / \text{Max} & \quad f_i(\mathbf{x}), \quad i=1,2,\ldots,\text{Nobj} \\
\text{Subject to:} & \\
g_i(\mathbf{x}) & \leq 0, \quad i=1,2,\ldots,P \\
h_i(\mathbf{x}) & = 0, \quad i=1,2,\ldots,Q \\
\mathbf{x} & = \{x_1, x_2, \ldots, x_n\} \\
\mathbf{x}^{(L)} & = \{x_1^{(L)}, x_2^{(L)}, \ldots, x_n^{(L)}\} \\
\mathbf{x}^{(U)} & = \{x_1^{(U)}, x_2^{(U)}, \ldots, x_n^{(U)}\} \\
\mathbf{x}^{(L)} & = \{x_1^{(L)}, x_2^{(L)}, \ldots, x_n^{(L)}\} \\
\mathbf{x}^{(U)} & = \{x_1^{(U)}, x_2^{(U)}, \ldots, x_n^{(U)}\}
\end{align*}
\]

Consider a given system defined by a number of variables \(m\); vector \(\mathbf{x}\) is composed of specific variable values that define a unique instance of such system. Different combinations of different variable values will yield different system performance, i.e., variations in the elements of vector \(\mathbf{x}\) will yield different result for the objective functions \(f_i(\mathbf{x})\). It is the purpose of the multi-objective optimization study to determine the set of optimal system instances or designs, comprised of optimum variable values; such set is called the Pareto front. Moreover, there may be limitations or constraints that restrict the selection of variable values; these are defined by functions.
$g_i(x)$ and $h_j(x)$, the inequality and equality constraint functions, respectively. Also, variable values should be confined within a bounded space for feasibility of the system; the lower and upper limits of this bounded space are defined by vectors $x^L$ and $x^U$, respectively and such region is termed the Feasible space.

Population based optimization algorithms offer a stochastic approach in optimization; they imitate the social behavior of populations driven towards an objective. Particle-swarm, for instance, is inspired by social behaviors such as bird flocking. Although PSO shares diverse similarities with evolutionary algorithms, its nature is different due to the lack of evolution operations such as cross-over and mutation. Instead, population members are driven to an optimum state by a combination of behaviors, namely sociability and individuality.

The particle-swarm optimization concept consists of iteratively balancing the rates at which a population member simultaneously approaches both its best individual state, and the overall global best state of the population. Such iterative process is initiated with a random population and it leads to an optimum population, composed of the best performing members.

2.2 Multi-Objective Optimization Definitions

The following terms are crucial to understanding the subsequent discussions in this chapter.

*Population*. In population-based stochastic optimization, such as PSO, a population is a set of members, each one representing a unique instance of a system.
Population Member. It is a unique instance of the system subject to optimization. It is defined by unique combination of variable values for the variables that define the system.

Population Generation. Recall that multi-objective optimization algorithms are iterative; during every iteration the population members are updated. The population set during a given iteration is termed a population generation, or simply a generation.

Variable Space. It is the dimensional space in which population members are defined; it is composed of as many axes as number of variables the optimization problem has, and it contains equal number of points as the population size of the optimization problem.

Objective Space. It is the mapped space resulting from plotting the objective function values yielded by all population members; thus, it contains as many points there are in the variables space (optimization population size), but it is composed of only as many axes as number of objectives the optimization problem has.

Feasible Space. It is a bounded sub-region of the objective space defining system feasibility; any system instance (member) yielding objective results outside the feasible space are discarded since they imply system infeasibility.

Function Evaluation. The process of assessing a unique system instance; it consists of computing the objective function for a given set of variable values.

Dominance. In single-objective optimization, it is easy to determine when a population member is better than another by simply comparing the values that they yield for the unique objective function. However, in multi-objective problems, it is not a straightforward decision. When comparing two members, it may be the case that one
performs better than the other in one objective, but is inferior with respect to a different objective; in such cases, it is hard to determine which one of the compared members is stronger than the other. As a matter of fact, most of the time, this situation occurs and members are said to be non-dominated with respect to each other. The concept of dominance is vital in multi-objective optimization since the search for optimal solutions and the intermediate selection of them are driven by it. The concept is explained below and the explanation is illustrated on Fig 1. In general, it can be stated that one solution $A$ weakly dominates another solution $B$ if the two conditions below are fulfilled:

1. Solution $A$ is no worse than $B$ in any of the objective functions evaluated.
2. Solution $A$ is clearly better than solution $B$ in at least one of the assessed objective functions.

If either one of the solutions in question cannot be said to dominate the other on the basis of the conditions above, the solutions are said to be non-dominated with respect to each other.

Figure 1. The Concept of Dominance.
Consider four solutions in a two-objective optimization problem where it is intended to maximize both functions. It is observed that solution 1 is non-dominated with respect to the other 3 solutions, as it performs worse than all in objective function 1, but it is strictly better than the rest with respect to objective function 2. Solution 2 is dominated by solutions 3 and 4, since they are both better in both the objective functions. Solution 4 dominates solution 3 since it is no worse than it in objective function 2, but clearly better than it in objective function 1. This example depicts the comparability of trade-off solutions in multi-objective optimization scenarios.

**Non-dominated Set.** Based on the discussion of the dominance concept, the non-dominated set during a given generation corresponds to the set of population members that are determined to be non-dominated with respect to all the rest of the members.

**Pareto Front.** The Pareto front is the representation of the non-dominated set; it holds the optimum trade-off population set for a multi-objective problem.

### 2.3 Multi-Objective Particle-Swarm Optimization Algorithm

For the sake of clarity, it is convenient to first discuss the single-objective particle-swarm algorithm and then explain the logic implemented to translate the algorithm into a multi-objective fashion. The particle-swarm optimization algorithm is an iterative process where a population of solutions is driven towards a set of optimal solutions; iterative improvement of the population members is dictated by Eq. (2.2).

$$
\bar{x}^{k+1} = \bar{x}^k + \vec{v}^k
$$

(2.2)

The subscript $k$ indicates the iteration/generation number; the variables vector $\bar{x}$ for each member is updated with respect to the previous iteration, by adding the velocity vector $\vec{v}$, which is calculated from Eq. (2.3).
\[ \vec{v}_i^k = \alpha \vec{v}_i^{k-1} + \beta_1 \vec{r}_i \left( \vec{P}_i - \vec{x}_i^k \right) + \beta_2 \vec{r}_2 \left( \vec{P}_G - \vec{x}_i^k \right) \]  \tag{2.3}

where

- \( \alpha \) is the normalized inertia factor and should take a value between 0 and 1 [18].
- \( \beta_1 \) is the individuality factor and should take a value between 1 and 2 [18].
- \( \beta_2 \) is the sociability factor and should take a value between 1 and 2 [18].
- \( \vec{r}_i \) and \( \vec{r}_2 \) are vectors storing a different random number for every population member.
- \( \vec{P}_i \) is a vector that contains the best variable values achieved during optimization history for an individual member in the population.
- \( \vec{P}_G \) is a vector holding the global best variable values of the entire population; in single-objective optimization scenario, this is a single value.

The velocity vector is composed of three terms, each one representing an individual driving force. The first term in Eq. (2.3) governs the inertial behavior inherent to the member’s belongingness to the group; the second term represents the force that drives an individual member towards its own individual optimum state; finally, the third term in the equation symbolizes the member’s bias to approach the global optimum state. Figure 2 summarizes the PSO and algorithm.
1. The initial population is created using Sobol’s quasi random sequence generator [26].

2. The function evaluation process is problem-dependent but its only purpose is to compute objective function value based on an individual combination of variable values.
3. The random number vectors are created invoking C++’s random number generator.

4. Global and individual optimum points in the optimization history are updated. In single-objective PSO they are effortlessly determined; it is merely a comparison task.

5. Equations (2.2) and (2.3) are employed to update the next generation of candidate solutions.

6. Function evaluation process is invoked to compute objective function values for all newly updated members.

7. If the convergence criteria are not met, the iterative process advances to the next generation and re-executes the steps 2 through 6 until convergence is achieved.

   The transition from single-objective to multi-objective fashion of the particle-swarm optimization logic is accomplished by simply modifying the criteria used to determine the global and individual best variable values in step 4; the algorithm itself is identical, with the exception that evidently, the function evaluation process yields two or more objective function values.

   In MOPSO, it is irrelevant to refer to a global optimum since multiple-objective need to be favored during the optimization process. Instead, the global optimum for a given member is determined by locating the closest point in the current Pareto front approximation, to the member in question. Hence, the distance of a member’s representation in the objective space with respect to all other members is computed, and the closest point in the Pareto is determined; then, such point is mapped to the variable
space for acquisition of the current optimum variable values for the population member in question.

Moreover, individual optimum points for a given member are iteratively updated based on the dominance criterion; the individual local optimum for the member in question is initialized as the randomly generated member value. After every iteration elapses, the newly computed objectives are checked for dominance with respect to the current individual optimum for the corresponding member. If it is determined that the new variable values for this member dominate the current optimum for it, the current optimum is then updated to the newly computed variable values. If it is determined that the current optimum for the member dominates the newly computed solution, the current optimum is kept as the individual optimum for the member. Finally, if non-domination condition applies upon dominance check, current optimum is updated to the newly computed variable values in order to avoid convergence to a local optimum.

Pareto-sense calculation of the global and individual optimum points in the optimization history for all population members is where the multi-objective nature of the MOPSO algorithm relies.

2.4 MOPSO Performance Parameters Analysis

MOPSO’s performance is dictated by the three individual behaviors constituting the velocity vector in Eq. (2.3). Likewise, these behaviors are regulated by the factors $\alpha$, $\beta_1$, and $\beta_g$. It is recommended that the inertial factor takes a value between 0.0 and 1.0 and that the individuality and sociability factors take a value between 1.0 and 2.0 [18]. This section covers a study carried out to evaluate the impact of parameter value changes within the suggested range, on particle-swarm optimization performance. Conclusions are
drawn from the study, leading to appropriate MOPSO performance parameter selection for the actual multi-objective optimization study of interest in this thesis.

Consider a single-objective unconstrained optimization problem with an objective function of 2 variables known as the Michalewickz's function. The objective function is taken from a handbook of test problems in local and global optimization [27]. Michalewickz's function topology is displayed in Fig. 3 and the theoretical location of its minima in Fig. 4. Besides the optimum's location, the expected objective function value at it is known to be zero. This information is used to judge PSO performance upon parameter modifications.

![Figure 3. Topology of Michalewickz's Function.](image)

Single-objective PSO was run on different combinations of inertial and population optimum factors; the individual and social factors are assumed to be equal. A random initial population is generated and stored to be used in all PSO runs to filter performance perturbations due to initial population closeness to the final global optimum. A population size of 20 members is suitable for a 2-variable problem [18].
Let us first examine the effect of $\beta$ variations for constant $\alpha$ values, on the objective function convergence.

Figure 5. Influence of Individuality Parameter on Objective Function Convergence with Low Inertial Factors of 0.0 and 0.2 (a,b).

Figure 5a depicts Particle Swarm behavior without inertial effects; thus, entire population is driven by individuality and sociability forces. Low $\beta$ values (1.0 and 1.2) show stable convergence but to insufficient extent; function is minimized after around
600 function evaluations but the objective only drops to about -1 order of magnitude. Values of 1.4 and 1.6 for \( \beta \) improve the level of convergence (-3 and -5 order of magnitude, respectively) requiring about 850 and 1100 evaluations, respectively.

The case shown in Figure 5b represents low inertial effect. Introduction of inertial effect imposes significant improvement in general. Except for \( \beta \) values of 1.2 and 1.4, a trend indicating that convergence rate is inversely proportional to \( \beta \) parameter is apparent, i.e. the fastest converging case is for 1.0 and slowest for 2.0. Moreover, except for values of 1.2 and 1.4, excellent accuracy and convergence level is accomplished with \( \alpha \) parameter set to 0.2.

![Figure 6. Influence of Individuality Parameter on Objective Function Convergence with Medium Inertial Factors of 0.4 and 0.6 (a,b).](image)

Increasing the inertial effect up to \( \alpha \)-levels of 0.4 and 0.6 yields oscillation of the convergence history of the objective function; these oscillations have strong impact in the amount of function evaluations invoked, the computational execution time and the overall algorithm’s stability. The majority of the cases with \( \alpha = 0.4 \), shown in Fig. 6a, converge accurately down to -9 order of magnitude; in addition, all of them terminate before
exceeding the maximum number of allowable function evaluations. It is clearly evident that the amount of function executions demanded to converge is proportional to the $\beta$ parameter in this case.

In the case of $\alpha = 0.6$, see Fig. 6b, convergence power is still high for low $\beta$’s (1.0-1.4); however, larger number of function calls are implemented with respect to lower $\alpha$ cases. $\beta$ values of 1.6 and 1.8 are still acceptable in terms of minimization level accomplished for the objective function; yet, they execute all allowable function calls; finally, a value of 2.0 for $\beta$ induces divergence pattern on the PSO convergence.

Figure 7. Influence of Individuality Parameter on Objective Function Convergence with high Inertial Factors of 0.8 and 1.0 (a,b).

Inspection of Figs. 7a and 7b demonstrates that high inertial factor selection causes instability; it is observed that for all tested values of $\beta$, oscillations and insufficient minimization of the objective function become distinct.

Let us now focus on the influence of the inertial factor $\alpha$ for constant $\beta$ values.

Figures 8a and 8b prove that for low $\beta$ values (1.0 and 1.2), most $\alpha$ selections perform well, except for 0.8 and 1.0, which depict high oscillations (higher for 1.0). Also,
it is observed that $\alpha$ value of 0.0 yields low minimization in both $\beta$ cases and that same
pattern is given by 0.2 in the 1.2 $\beta$-case. Overall, it is concluded that for the discussed $\beta$
values, $\alpha = 0.4$ and $\alpha = 0.6$ provide highest ranked performance in terms of robustness
and minimization capability.

![Figure 8. Influence of Inertial Parameter on Objective Function Convergence with Low
Individuality Factors of 1.0 and 1.2 (a,b).](image)

![Figure 9. Influence of Inertial Parameter on Objective Function Convergence with
Medium Individuality Factors of 1.4 and 1.6 (a,b).](image)
Raising $\beta$ parameter up to 1.4 and 1.6, results in remarkable divergence of high inertial factor cases; the best performing $\alpha$ selection in this range of $\beta$'s is 0.4. Low inertial factors, e.g. 0.0 and 0.2, increase the algorithm’s stability; it can be noticed that oscillations of the objective function vanish, although the extent of minimization accomplished is undesired; except on the $\alpha = 0.2, \beta = 1.6$ combination (Figs. 9a and 9b).

Assessment of large $\beta$ values (1.8 and 2.0) yields that inertial factors over 0.6 are prone to causing instability and therefore, should not be implemented in order to ensure optimizer’s robustness. For high $\beta$, only low inertial factors are successful in providing consistent function minimization within an acceptable range of function evaluations; nevertheless, oscillations are still evident in such cases. See Figs. 10a and 10b.

![Particle Swarm](image)

**Figure 10.** Influence of Inertial Parameter on Objective Function Convergence with High Individuality Factors of 1.8 and 2.0 (a,b).

The following conclusions are drawn from the above study of the PSO performance parameters, and they are put into use when executing MOPSO for the optimization problem covered in his thesis.
1. Absence of inertial displacement of the population members enhances the optimizer’s stability but it causes insufficient minimization of the objective function.

2. Low inertial factor value, e.g. 0.2, strengthens minimization capability and yields stable convergence for all tested $\beta$ values.

3. In general, it is inconvenient to set the inertial factor over 0.2 since it induces oscillations in the convergence history. Such oscillations become sufficiently large to cause divergence of the optimizer, when high-range inertial factors are selected.

4. For divergence-prone cases in the high inertial factor range, the divergence level is proportional to the magnitude of $\beta$; this is an indicator that both parameters should not be simultaneously increased.

5. Combinations of inertial factor between 0.25 and 0.5 along with individual/global optimum factors between 1.2 and 1.4 are suggested to deliver a good trade-off between the optimizer’s robustness and minimization power.

2.5 Numerical Experiments

In order to evaluate the capability of the MOPSO version developed to accomplish this work, it was tested by running it on well-known test problems published in previous research work. The examined work is a comparison of multi-objective evolutionary algorithms performed by Zitzler, Deb and Thiele [28], referred to as the ZDT optimization test cases. The properties of these test cases are summarized in Table 1; they all involve minimization of 2 objective functions defined as $f1$ and $f2$, within a given feasible space, for a multiple number of variables given by $m$. 
Table 1. ZDT optimization test cases summary.

<table>
<thead>
<tr>
<th>Problem</th>
<th>$m$</th>
<th>Variable Limits</th>
<th>Objective Functions</th>
<th>Analytical Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZDT-1</td>
<td>30</td>
<td>$x_i \in [0,1]$</td>
<td>$f_1 = x_i$</td>
<td>Set $g = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$g = 1 + 9 \cdot \sum_{i=2}^{m} \frac{x_i}{m-1}; h = l - \sqrt{\frac{f_1}{g}}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$f_2 = h \cdot g$</td>
<td></td>
</tr>
<tr>
<td>ZDT-2</td>
<td>30</td>
<td>$x_i \in [0,1]$</td>
<td>$f_1 = x_i$</td>
<td>Set $g = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$g = 1 + 9 \cdot \sum_{i=2}^{m} \frac{x_i}{m-1}; h = l - \left( \frac{f_1}{g} \right)^2$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$f_2 = h \cdot g$</td>
<td></td>
</tr>
<tr>
<td>ZDT-3</td>
<td>30</td>
<td>$x_i \in [0,1]$</td>
<td>$f_1 = x_i$</td>
<td>Set $g = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$g = 1 + 9 \cdot \sum_{i=2}^{m} \frac{x_i}{m-1}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$h = l - \frac{f_1}{g} - \left( \frac{f_1}{g} \right) \sin(10 \pi f_1)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$f_2 = h \cdot g$</td>
<td></td>
</tr>
<tr>
<td>ZDT-4</td>
<td>10</td>
<td>$x_i \in [0,1]$</td>
<td>$f_1 = x_i$</td>
<td>Set $g = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$g = 1 + 10(m-1)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x_i \in [-5,5]$</td>
<td>$h = l - \sqrt{\frac{f_1}{g}} + \sum_{i=2}^{m} \left( x_i^2 - 10 \cos(4 \pi x_i) \right)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x_i \in [-5,5]$</td>
<td>$f_2 = h \cdot g$</td>
<td></td>
</tr>
</tbody>
</table>
To counteract the performance fluctuations resulting from different closeness to the optimal Pareto front in the random population initialization process, and other relevant operators in the MOPSO algorithm, the optimizer was run 10 times for each test case and consolidated non-dominated sets were assembled for each case after the 10 runs; each run invokes 20,000 function evaluations. Calculated Pareto fronts are plotted and compared to the analytical Pareto solutions in Figs. 11-15. The set of parameters defining the MOPSO program execution are summarized in Table 2.

**Table 2. MOPSO Definition Parameters.**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size (Npop)</td>
<td>100</td>
</tr>
<tr>
<td>Inertial factor (α)</td>
<td>0.25</td>
</tr>
<tr>
<td>Individuality factor (β₁)</td>
<td>1.4</td>
</tr>
<tr>
<td>Sociability factor (β₆)</td>
<td>1.4</td>
</tr>
</tbody>
</table>
Figure 11. MOPSO Validation Against ZDT-1.

Figure 12. MOPSO Validation Against ZDT-2.
Figure 13. MOPSO Validation Against ZDT-3.

Figure 14. MOPSO Validation Against ZDT-4.
2.6 MOPSO Adaptation for Parallel Computing

Simple applications, such as the ZDT test cases, involve function evaluation processes that do not imply considerable CPU time consumption. However, for high-level optimization problems like the one being dealt in this work, the function evaluation process may be comprised of a complex mathematical model which is numerically solved taking numerous minutes to assess the objectives of a single population member; now imagine the time consumption of an entire generation of members, and recall that multiple generations must be carried out in order to calculate an accurate Pareto front.

Undoubtedly, a problem of this size cannot be addressed using a single CPU. Computing power of multiple CPU's must be simultaneously used in order to solve this type of application and that implies an adaptation of the developed MOPSO algorithm to be run in a parallel computing mode. The message passing interface (MPI) standard is implemented to alter MOPSO's logic in order to support parallel execution of function
evaluation processes. MOPSO is modified to iteratively update its driving parameters and population members to then invoke a function called “geneval”, which simultaneously executes the function evaluation process for all population members by sending information of the individual members to different processors and collecting results from them. MPI commands are invoked to send variable values corresponding to a given population member within the “geneval” function in the MOPSO algorithm. See Fig. 16 for clear understanding of the parallel computing adapted MOPSO algorithm.

![Figure 16. MOPSO Adaptation for Parallel Computing.](image-url)
where

⇒ represents the call to function “geneval”, which executes MPI commands to solve each member of the population on a different processor.

`var` is an array of $N_{pop}$ rows and $N_{var}$ columns; it stores the variable attributes of the entire current generation.

$N_{var}$ is the number of variables in the optimization problem.

$N_{pop}$ is the number of population members.

`obj` is an array of $N_{pop}$ rows and $N_{obj}$ columns; it stores the computed objective values for the entire population.

$N_{obj}$ is the number of objectives in the optimization problem.

`numsent` is a counter that helps the program determine when the entire generation has been computed and ceases to send work to processors.
CHAPTER III

GENERATION OF 3D MICROCHANNEL NETWORKS

3.1 3D Microchannel Network Definition

In the scope of this work, a 3D microchannel network is defined as a multi-level arrangement of planar networks dispensed by a distribution system made of vertical feeder passages. Each of the planar horizontal networks at different vertical levels is fed with a different mass flow rate. Each of the horizontal 2D networks also has different topology and geometrical characteristics. Let us discuss in this section the parameters that define each of the planar networks and the distribution system, in order to get an insight to the method implemented for automatically generating 3D microchannel heat sinks. Figure 17 illustrates a sample 3D network and in Fig. 18, some of the basic definitions used hereafter are shown.

![Sample 3D Microchannel Network Diagram](image)

**Figure 17.** Sample 3D Microchannel Network.
Let us review the definition of some concepts shown in Figs. 17 and 18.

- **Planar networks** are the building blocks of the 3D multi-level microchannel network. They represent the topology of the microchannel arrangement at a given vertical level.

- The **solid substrate** is the actual body in which the microchannels are embedded.

- The **distribution system** is the arrangement of vertical passages feeding each of the planar networks in the multi-level/story system.

- A **planar inlet** is the location in a planar network that is directly connected to the distribution system.

- **Planar flow outlets** are the openings through which heated fluid exits the domain.

- Each planar network has multiple **connectivity levels**, which are the locations in the direction of flow where branching junctions occur.
Based on the number of connectivity levels, the number of branching levels for a given planar network can be determined.

Experimental work with fractal tree-like microchannel networks [14] has demonstrated that such organization of the passages enhances temperature uniformity at thermal load’s location; additionally, it augments heat removal. Consequently, for the purposes of this work, all planar networks may be mirrored with respect to their origin, in three different directions to yield combined planar networks. Sample 3D networks using mirrored planar networks are depicted in Fig. 19. The resulting variant of the distribution system, upon activation of planar network mirroring, is observed in detail in Fig. 20 and a stand-alone mirrored planar network can be seen in Fig. 21.

Figure 19. Sample 3D Multi-Story Network with Mirrored Planar Networks.
The following topological and geometrical characteristics fully define a given planar network.

1. Number of branching levels.
2. The level-to-level branching pattern.
3. Planar mirroring details.
4. Length of each of the microchannels.
5. Diameter size of each of the microchannels.
6. Wall surface roughness of each of the microchannels.

7. Vertical location of each horizontal 2D network of microchannels.

Parameterization details of the above characteristics and methodology for automatic generation of network geometries is discussed in the following section.

3.2 Automatic Network Generation

In order to allow automatic generation of microchannel networks, a 3D branching network generator was developed (3DBNGEN). The parameters listed below must be supplied as input for the program to generate the corresponding network. It applies a sweeping logic that constructs the planar networks starting from the planar inlets and sweeps forward determining the appropriate locations of the passage junctions and their connectivity.

General parameters:

- Number of floors \((nfloors)\). This parameter controls the number of horizontal 2D networks that the 3D microchannel network will have.

- Vertical clustering factor \((vclust)\). This factor controls the vertical locations of the floors; it performs floor clustering towards either one of the faces (top or bottom) of the solid substrate, using the single-exponential approach for point clustering (Appendix C).

Per floor parameters:

- Number of branching levels \((bln)\). This number controls the branching extent of the planar network.

- Branching level clustering \((bleclust)\). This clustering factor controls the distances between junctions in adjacent connectivity levels, thus being responsible for
passage length specification. Based on the same exponential approach as in the vertical clustering case, distances between branching levels are calculated from this factor and hence, passage lengths are determined.

- **Mirroring factor** (*mirrf*). The mirroring factor controls activation of the mirroring feature. It takes an integer value between 1 and 4 and it dictates how many simple planar networks (See Fig. 18) are used to build the final combined planar network (See Fig. 21).

- **Number of outlet paths** (*nouts*). The number of outlet paths indicates the number of outlets that the planar network presents.

- **Planar outlets clustering coefficient** (*plclust*). This coefficient is in charge of clustering the outlet paths to deliver various channel conglomeration patterns.

**Per branching level parameters:**

- **Junctions’ connectivity** (*jn*). The junction connectivity parameter is defined as the number of branches emanating from a given junction. It must be specified for all junctions in the planar network and the parameter may take an integer value of 1, 2 or 3.

- **Diameter of microchannels** (*dia*). Microchannel hydraulic diameters for the entire branching level are stored in this variable.

- **Surface roughness of microchannels** (*rgh*). Surface roughness of the microchannel walls for the entire branching level are managed by this variable.

Consequent to successful completion of the network generator, three lists are generated that fully define the created geometry:

1. A list of junction points distribution in the Cartesian coordinate system.
2. A junction points connectivity list defining microchannels’ branching topology.

3. A geometrical specification list defining diameters and wall roughness.

The above lists are used to complete the following operations:

1. Preparation of the input files for the thermo-fluid solver. This solver does not require a meshed domain. It only needs a connectivity set with the information about the physical problem and the geometrical characteristics of the network, in a very specific format.

2. Creation of an input file for the module responsible for automatic meshing the solid domain.

3.3 Fluid Domain Preparation

Preparation of the fluid domain upon completion of the network creation consists in assembling an input file that contains the following information:

- The settings for the numerical scheme involved in the flow-field calculation.
- The pressure boundary condition settings for all inlets and outlets.
- The thermal boundary conditions for the fluid at the entrance points.
- Wall temperatures for every single passage in the heat sink.
- Total number of junction nodes, passages, inlets and outlets.
- Connectivity of all junction nodes defining entire branching topology of the passages.
- Cross-sectional area for all passages.
- Hydraulic diameter for all passages.
- Length of all passages.
- Wall surface roughness for all passages.
The generated input file constitutes the only necessary item to carry out entire flow-field calculation; its specific format can be reviewed in (Appendix A).

3.4 Solid Domain Meshing and Preparation

Solid domain preparation is significantly more complex than fluid domain preparation. It involves 2 stages: 1) microchannels surface meshing, and 2) substrate domain meshing.

The objective of the solid domain meshing and preparation is to deliver the hybrid mesh and the definition files that are required for the solution of the discretized equations governing steady heat diffusion in a solid. The process of meshing the solid basically consists of determining the volume occupied by the fluid structures and subtracting it from the solid block subject to analysis, i.e. creating a void in a solid block, corresponding to the volume which the fluid passages occupy in the internal structure of the solid substrate. Figures 22 – 25 illustrate the methodology to create the solid domain mesh.

Let us consider a similar example to convey the used methodology. The starting point is the definition of the object that creates the void. In the scope of this work, this corresponds to the actual fluid region bounded by the walls of the microchannel heat sink. However, in the example below, the opposite situation applies, i.e. fluid domain is being meshed by creating a void taking the morphology of the solid domain embedded within the fluid domain. Figure 22 shows the object that will cause the void.
After the volumetric shape of the void has been discovered, such volume must be removed from a perfectly solid piece, creating a void. See Fig. 23.

The resulting volume region represents the spatial domain of interest. An unstructured or hybrid mesher of preference may be invoked to mesh such space and therefore deliver the required mesh for simulation of the physical phenomena. The resulting mesh for the example considered is shown in Fig. 24.
Let us review the functions of these computer programs.

The microchannel network geometry surface mesher sweeps through the entire network and meshes the surface of all passages individually, considering the geometrical characteristics, existing junction type and branching properties at the ends of the passage. For all passages the program executes the following commands.

- Read connectivity lists and determine Cartesian coordinates of the passages’ end points.

In order to implement the method explained above, the following resources were adapted/developed and integrated with the automatic branching network generation program to assemble a self-sufficient automatic substrate domain mesher.

1. A microchannel network surface mesher was developed.

2. A hybrid hexahedra/prism volume mesher developed by OpenCFD Ltd. [29] was adapted for use with the surface mesher and geometry generator.

Figure 24. Resulting Hybrid Mesh.
- Compute passage length.
- Create a set of perpendicular cut planes at number of locations along the passage.
- On each cut plane, a circular cross-section is drawn, and a set of points are placed in the circumference of such cross-section. The coordinates for all placed points in x,y,z space are stored.
- For odd-numbered cut planes, the coordinates are shifted to purposely induce misalignment between adjacent cut planes. Local numbering is kept to allow matching of circumferential points along all cut planes.
- Existing junction type and branching patterns at ends of the passage are read.
- The sets of points on the first and last cut planes are transformed according to the junction type, branching direction, etc.
- A sweeping algorithm is used that moves from the first to the last cut plane of the passage, recording sets of vertices that form triangular patches upon connection of the corresponding circumferential points.
- Triangular patches corresponding to the surface elements of the passage (single microchannel) are stored in a STL mesh format.
- A normal vector calculator is called to compute a unit normal vector for the created patch.

Iterative repetition of the above commands for all passages results in full mesh generation for a planar network. Then, the process is repeated for all planar networks to complete full 3D geometry surface meshing. Figures 25 – 28 demonstrate the capability of the automatic surface mesher.
Figure 25. Sample Planar Network.

Figure 26. Surface Mesh for Sample Network.
Let us describe the working principles of the hybrid hexahedra/prism volume mesher included in the OpenFOAM program, which was implemented in this work. The volume mesher is called snappyHexMesh. It meshes a rectangular block considering the dimensions of the solid substrate being analyzed. It then uses a surface mesh to calculate an enclosed volume which will cause a void in the solid piece. In the scope of this research, this utility was adapted to use the surface mesher previously described to obtain the required surface mesh for void volume calculation. It then proceeds to segment the mesh in the solid object in the vicinities of the declared surface mesh so that volumes
inside the void region are flagged for removal. Consequent to removal of these volumes, the software places the nodes onto the declared surface patch in order to yield a patch-conforming volume mesh. Successful attainment of the boundary-conforming mesh evolves to the last meshing stage, which is the construction of prismatic elements on selected boundaries for mesh improvement.

Integration of the branching network generator, the network surface mesher and the snappyHexMesh utility provides the resources necessary for automatic execution of geometry creation, computational grid generation and preparation for solution. Further integration of this subsystem with the automatic conjugate heat transfer solution package constitutes the necessary platform for automatic function evaluation that is used by the optimization algorithm.
CHAPTER IV
CONJUGATE HEAT TRANSFER SOLUTION PACKAGE

4.1 Overview

An automatic 3D conjugate heat transfer solution package (CHETSOLP) was developed to model conjugate heat transfer phenomena, calculate flow/temperature fields and assess any microchannel heat sink topology. The governing equations of conjugate heat transfer, the numerical algorithms developed and additional modules of the CHETSOLP package are described in this chapter.

The working logic of the package is to solve the flow field, transfer flow-field data to the heat conduction solver, solve heat diffusion, transfer temperature data to the flow solver, and iteratively repeat this procedure until the wall temperatures (initially guessed) converge.

4.2 Conjugate Heat Transfer Formulation

The following assumptions must apply in order for the proposed model to hold true, as proposed by Fedorov and Viskanta in 2000 [13].

1. The Knudsen number for the physical problem should not exceed 0.001 in order for the conservation equations derived from the continuum model (Navier-Stokes equations) to be valid [30].
2. Transport phenomena is assumed to be steady-state and three-dimensional.
3. Flow is incompressible.
4. Flow is one-dimensional.
5. Thermal radiation is negligible with respect to advection.
6. Flow entrance region is accurately resolved in order to capture high-gradients.
7. Thermo-physical properties of the coolant fluid are temperature-dependent.

Under the assumptions stated above, the conservation of mass, momentum and energy for the solid and fluid domains can be formulated as follows:

Mass conservation (continuity equation) for the fluid domain:

\[
\nabla \cdot (\rho \vec{V}) = 0 \tag{4.1}
\]

Linear momentum conservation in x-direction (direction of flow):

\[
\nabla \cdot (\rho |\vec{V}| \vec{V}) = -\frac{\partial P}{\partial x} + \frac{\partial}{\partial x} ([\tau]) + \rho f_x \tag{4.2}
\]

Energy conservation for the fluid domain:

\[
\nabla \left[ \rho \left( e + \frac{|\vec{V}|^2}{2} \right) \vec{V} \right] = \rho \dot{\theta} + \frac{\partial}{\partial x} \left( k \frac{\partial T}{\partial x} \right) - \frac{\partial (|\vec{V}|^2 \rho)}{\partial x} + \rho \vec{F} \cdot \vec{V} \tag{4.3}
\]

Energy conservation for the solid domain:

\[
\nabla \cdot (k_{\text{solid}} \nabla T) = 0 \tag{4.4}
\]

Thermal boundary conditions at the fluid/solid interface:

\[
T = T_{\text{wall}} \tag{4.5}
\]

\[
k_{\text{solid}} \frac{\partial T}{\partial n} \bigg|_{\text{wall}} = -h(T_w - T_{\infty}) \tag{4.6}
\]

Equations 4.1 – 4.6 are numerically solved with a conjugate heat transfer solution package developed in this research. A thermo-fluid flow/convective heat transfer analysis code and a steady heat diffusion solver are sequenced into an iterative process that couples them by means of a boundary condition transfer module. The numerical
algorithms in this solution platform and the boundary condition transfer module are described in this chapter.

4.3 Flow-Field Analysis

This section of the chapter covers the mathematical formulation of the governing equations for the fluid flow problem, gives a description of the functioning principles of the numerical scheme and demonstrates the validity of the program by exposing the results from numerically experimental validations.

4.3.1 Flow-Field Solver Theory

The proposed numerical model for approximation of the mass and momentum balance equations that follows, assumes flow conditions to be steady, incompressible and one-dimensional.

Under the proposed flow conditions, and assuming constant density and velocity across a constant-area fluid element, the momentum conservation equation reduces to Eq. (4.7).

\[
\sum [\vec{F}] = \sum [\dot{m}\vec{V}] \tag{4.7}
\]

where

\(\vec{F}\) represents the forces acting on the fluid inside the branching network microchannel.

\(\dot{m}\) is the mass flow rate through the microchannel.

\(\vec{V}\) is the velocity vector.

For all passages in a network, the momentum balance in Eq. (4.7) can be expressed in terms of their static pressure values at the element’s inlet and outlet points, the wall shear stress and the minor losses, as shown in Eq. (4.8).
\[ P_{IN} A - P_{OUT} A - [\tau]S = f_{\text{minor}} \]  \hspace{1cm} (4.8)

where

- \( P \) is the fluid’s static pressure.
- \( A \) is the cross-section area of the passage.
- \( \tau \) is the wall shear stress.
- \([\tau]\) is a scalar representing the magnitude of the average wall shear stress.
- \( S \) is the surface area of the passage.
- \( f_{\text{minor}} \) represents the forces due to minor losses.

The average wall shear stress acting on the fluid inside a passage is approximated in Eq. (4.9) as a function of the Darcy friction factor and the dynamic pressure.

\[ [\tau] = \frac{f \rho [\bar{V}]^2}{4} \]  \hspace{1cm} (4.9)

where

- \( f \) is the Darcy friction factor.
- \( \rho \) is the fluid’s density.
- \([\bar{V}]\) is a scalar that represents the average fluid velocity through the passage.

The Darcy friction factor is calculated from Eq. (4.10) for laminar flow conditions. For turbulent flow conditions, it is approximated from Eq. (4.11) as a function of Reynolds number and the relative wall roughness. Equation (4.11) was proposed by S.E. Haaland [31].
\[ f = \frac{64}{Re} \]  

(4.10)

\[
\frac{1}{\sqrt{f}} \approx -1.8 \log \left( \frac{6.9}{Re} + \left( \frac{\varepsilon/D_h}{3.7} \right)^{1.11} \right) \]  

(4.11)

The \( \beta \) parameter is defined by Eq. (4.12) and equivalent total pressures are defined in Eq. (4.13). Accounting for minor losses due to junction of passages, the momentum balance (Eq. 4.8) for the passage is reduced down to Eq. (4.14).

\[
\beta = \frac{2A}{f \left[ \bar{V} \right]} \]  

(4.12)

\[
P_{EQ} = P + \rho \left[ \bar{V} \right]^2 \]  

(4.13)

\[
\beta \left( P_{EQIN} - P_{EQOUT} \right) = \frac{1}{4} \frac{S}{A} \bar{m} - \beta \Delta P_{\text{minor}} \]  

(4.14)

where

\( \Delta P_{\text{minor}} \) represents the pressure loss due to branching of passages and is calculated from Eq. (4.15). In (Eq. 4.15), the minor loss factor is calculated based on junction type and upstream Reynolds number (see Appendix D).

\[
\Delta P_{\text{minor}} = K_f \frac{\rho \left[ \bar{V} \right]^2}{2} \]  

(4.15)

\( A \) and \( S \) represent the passage’s cross-sectional and surface areas, respectively. They are defined in Eq. (4.16) and Eq. (4.17) as functions of the hydraulic diameter, \( D_h \), and the perimeter, \( \bar{p} \):
Equation (4.14) is the simplified momentum balance law that must be solved for each and every fluid passage of the network of microchannels.

Mass continuity is enforced throughout by means of solving Eq. (4.19) at every junction node, i.e. at every junction of microchannels the incoming flows and outgoing flows must balance out since no mass sources exist.

\[ \sum \dot{m} = 0 \]  

(4.19)

Let us now discuss the energy balance implications in the fluid domain; the heating of the fluid in a passage of constant cross-section is considered. If constant wall temperature is assumed, the rate of heat transferred into the flowing fluid is defined in Eq. (4.18), based on Newton’s law of cooling.

\[ \dot{Q} = hS\Delta T_{\text{avg}} \]  

(4.20)

where

\( \dot{Q} \) is the heat transferred into the fluid.

\( \Delta T_{\text{avg}} \) is the average temperature difference for the assumed wall conditions and it may be approximated from Eq. (4.21) [31].

\( h \) is the convection heat transfer coefficient.
\[ \Delta T_{avg} = T_w - \frac{T_{IN} + T_{OUT}}{2} \]  

(4.21)

In Eq. (4.21), the rightmost term is referred to as the bulk fluid temperature and it is represented by \( T \).

Considering a heating scenario where the wall temperature yields thermodynamic energy increase in the fluid particles, it is expected that the bulk temperature will rise in the direction of flow; if a constant cross-section element, namely a single microchannel in a network, is considered, the balance on such differential control volume is given by:

\[ \dot{m} c_p dT = h(T_w - T) dS \]  

(4.22)

Expressing the infinitesimal bulk temperature change in terms of the wall temperature and the infinitesimal surface area change in terms of the perimeter and axial passage length facilitates integration of Eq. (4.22), which yields Eq. (4.23):

\[ \ln \frac{T_w - T_{OUT}}{T_w - T_{IN}} = -\frac{hS}{\dot{m} c_p} \]  

(4.23)

where

\( T_w \) is the passage’s wall temperature.

\( T_{IN} \) is the fluid temperature at passage’s inlet.

\( T_{OUT} \) is the fluid temperature at exit of the passage.

\( c_p \) is the fluid’s specific heat.

Therefore, for a given microchannel, the exit fluid temperature may be computed from Eq. (4.24), when inlet temperature, wall temperature, heat transfer coefficient, mass flow rate and fluid’s specific heat are known.
\[ T_{OUT} = T_W - \left( T_W - T_{IN} \right) e^{-\frac{hS}{m c_p}} \] (4.24)

The convective heat transfer coefficient, \( h \), can be calculated from Eq. (4.25) if Nusselt number is known.

\[ Nu = \frac{hD_h}{k_{fluid}} \] (4.25)

In order to calculate Nusselt number and hence the convective heat transfer coefficient, the second Petukhov equation [31] is used. Petukhov’s equation is shown in Eq. (4.26) and it represents a relationship between the Nusselt, Prandtl and Reynolds numbers for steady incompressible flow in straight circular cross section tube.

\[ Nu = \frac{(f/8)(Re Pr)}{1.07 + 12.7(f/8)^{0.5} (Pr^{2/3} - 1)} \] (4.26)

where

\( Nu \) is the Nusselt number.

\( Re \) is the Reynolds number.

\( Pr \) is the Prandtl number.

Successful solution of temperature values from Eq. (4.24) implies energy balance in the fluid domain. Thus, solution of this equation in the entire domain along with Eq. (4.14) and Eq. (4.19) completes numerical approximation of a flow-filed that conforms to the three fundamental balance principles in the incompressible Navier-Stokes equations.

4.3.2 Quasi-1D Thermo-Fluid Solver Description

The quasi-1D thermo-fluid solver is an iterative scheme that decouples continuity and momentum from energy balance. It was formulated, developed and tested by T.J.
Martin [33]. Primitive variables are guessed in the initial stage according to reference values and then a matrix system is assembled to compute the entire balanced pressure and velocity fields and a sweeping logic is tackled to compute the temperature field. Based on the computed flow filed the primitive variables are updated, leading to updated matrix system and new balanced flow-field solution. The updating and solving process is repeated until convergence is achieved.

Let us first look at the method for continuity and momentum solutions.

Momentum conservation principle for a single microchannel is simplified in the model proposed in the previous section and it is cast into a matrix system of the following general form:

\[
\begin{bmatrix}
\beta & -\beta & -\frac{1}{4} S \\
-\beta & \beta & A^n \\
\frac{1}{4} S & -A^n & 0
\end{bmatrix}
\begin{bmatrix}
P_{EQ1} \\
P_{EQ2} \\
\dot{m}
\end{bmatrix}
= -\beta \Delta P_{\text{minor}}
\] (4.27)

The definition of the momentum conservation matrix system is straightforward for every passage. The mass conservation matrix balance is expressed on a nodal basis rather than on an passage by passage basis and therefore, the formulation of such matrix system is entirely dependent on the branching pattern that a given node exhibits. Equation (4.28) shows an example of a node where two passages merge and only one passage carries the outflow:

\[
\begin{bmatrix}
1 & 1 & -1
\end{bmatrix}
\begin{bmatrix}
\dot{m}_{\text{in1}} \\
\dot{m}_{\text{in2}} \\
\dot{m}_{\text{out}}
\end{bmatrix}
= 0
\] (4.28)
It can be observed that in the continuity matrix system, the coefficients matrix is composed of positive and negative unity factors strategically located in order to multiply the corresponding in/outflow, depending on the connectivity exhibited by the node to which the balance applies.

Equations (4.27) and (4.28) are merged into a composite matrix system for simultaneous solution of the mass and momentum balance equations; the quasi 1D thermo-fluid solver is capable of automatically assembling such system based on the microchannels connectivity; a sample network and its corresponding coupled momentum/continuity matrix system are shown in Appendix A for understanding of the ramifications underlying in the automatic assembly of such coupled system.

For the purposes of this discussion, it can be mentioned that such system is composed of a coefficient matrix, the vector of unknowns and the boundary conditions vector; the coefficients matrix is composed of the \( \beta \) factors for all passages, defined in Eq. (4.12), and a set of unity factors arranged in a specific form depending on passages’ connectivity. This matrix multiplies the unknown vectors which is assembled by placing 1) the equivalent total pressures for all internal branching network nodes, i.e. all passage junctions that are not located at an inlet or outlet port, and 2) average passage mass flow rates. The boundary conditions vector stores all known quantities derived from prescribed values at the domain boundaries. A matrix inversion subroutine solves for equivalent total pressures and mass flow rates simultaneously. Complex network topologies yield slight problem ill-conditioning; hence the Singular Value Decomposition technique [32] is used for matrix inversion at all iterations. The static variables are subsequently computed from calculated nodal equivalent pressures and average mass flow rates.
For the temperature field solution, no matrix system is required. A simple advancing-front program sweeps the microchannel network solving Eq. (4.24) for all nodes (except for inlets); wall temperatures, calculated mass flow rate, passage length and cross-sectional area are considered for solution of this equation. The resulting nodal thermal state implies energy balance of the entire microchannel network.

4.3.3 Quasi-1D Thermo-Fluid Solver Validation

The thermo-fluid solver described in the former section is named COOLNET. This program was originally designed for analysis of passage networks in gas turbine blades [33]; it has been modified to solve fluid flow inside branching networks of microchannels. The modified COOLNET version implemented in this work has been validated against analytical solutions and high fidelity 3D Navier-Stokes solvers, for simple fluid flow problems. Compressible flow test cases include Fanno and Rayleigh flow; incompressible flow test cases include Poiseuille flow. Let us discuss the validation results from Fanno and Poiseuille test cases only.

Consider the following test case: Fanno flow conditions apply in a circular pipe with a diameter of 0.15 m and length of 21.3 m. Uniform inlet flow conditions for air are given as pressure of 137.9 kPa, temperature of 294.6 K, and speed of 123.75 m/s.

Analytical solution, Navier-Stokes solver approximation and COOLNET solution for the above test case are compared below. The NSE solver implemented in this test case is the commercially available package Ansys CFX and the solution herein used for reference was accomplished with a computational grid of 250,600 cells. Figure 29 shows static pressure axial variation comparison, Fig. 30 depicts average fluid speed axial
variation comparison and Fig. 31 demonstrates solver’s ability to accurately estimate mass flow rates.

**Figure 29.** Validation of COOLNET for Fanno Flow (Pressure Field).

**Figure 30.** Validation of COOLNET for Fanno Flow (Velocity Field).
Figure 31. Validation of COOLNET for Fanno Flow (Mass Flow Rate).

The average normalized error in COOLNET’s computed pressure values, with respect to exact solution, is 0.23%. In the case of computed average speed values and mass flow rate, average normalized errors are 0.19% and 0.27%, respectively. The magnitude of the errors in the calculated quantities demonstrates the accuracy of this solver.

Consider Poiseuille flow of air (steady, incompressible, viscous, and isothermal flow) in a 0.5 m long straight circular cross section pipe with inner diameter $D = 1$ mm. The COOLNET program will be used to determine average fluid speed upon changes in the upstream pressure conditions. Validation against analytical values yielded by the Hagen-Poiseuille equation is shown in Fig. 32.
Normalized average errors increase for higher Reynolds number flows due to the introduction of an error in the computation of the Darcy fraction factor. However, error remains below 8% with respect to exact solution given by the Hagen-Poiseuille equation. High-fidelity 3D Navier-Stokes solver solutions yield more accurate results, but their computational expense is quite high. COOLNET converges 6 orders of magnitude within 25 seconds, while an unstructured 3D mesh of roughly 100,000 grid cells consumes over 1,200 seconds to reduce the residual by 6 orders of magnitude.

4.4 Heat Conduction Analysis

Heat conduction analysis is carried out with the OpenFOAM software; this section shows the mathematical formulation of the governing equations for steady heat diffusion in three-dimensions, describes the fundamentals of the numerical scheme used for problem solution and proves the validity of the implemented OpenFOAM utility by
comparing its performance on simple heat conduction cases, for which the analytical solution can be derived, to such exact solutions.

4.4.1 Steady Heat Diffusion Mathematical Model

The heat conduction problem inside the isotropic solid object is governed by the following mathematical model:

\[
\frac{1}{\alpha} \frac{\partial T}{\partial t} - \left( \nabla^2 T \right) = 0 \quad \text{for } (t > 0) \text{ and all } x,y,z \text{ values in the solid domain.} \tag{4.29}
\]

\[
\frac{\partial T}{\partial n} = -\frac{h(T_w - T_{\infty})}{k_{\text{solid}}} \quad \text{for } (t > 0) \text{ and all } x,y,z \text{ values at solid/fluid boundaries.} \tag{4.30}
\]

\[
\frac{\partial T}{\partial n} = \frac{\dot{q}_{\text{LOAD}}}{k_{\text{solid}}} \quad \text{for } (t > 0) \text{ and all } x,y,z \text{ values at heated surface.} \tag{4.31}
\]

\[
T = \bar{T} \quad \text{for } (t > 0) \text{ and all } x,y,z \text{ values located at bottom surface.} \tag{4.32}
\]

\[
T = T_0 (x, y, z) \quad \text{for } (t = 0) \text{ and all } x,y,z \text{ values in the solid domain.} \tag{4.33}
\]

where

\( T \) is the body’s temperature.

\( \alpha \) is the solid’s thermal diffusivity.

\( t \) is the time variable.

\( \nabla^2 \) is the Laplace operator.

\( n \) represents a unit vector in a direction normal to the surface.

\( T_{\infty} \) represents the fluid’s bulk temperature.

\( k \) is the solid’s thermal conductivity.

\( \dot{q}_{\text{LOAD}} \) is the thermal load generated by the distributed heat source on the top surface.
$\bar{T}$ is a reference temperature at which the bottom surface is kept constant.

$T_0$ is the initial temperature distribution in the solid.

Equation (4.32) is added to the governing mathematical model because of the need of at least one Dirichlet boundary condition in order to make the problem well-posed and thus, solve for a unique temperature field. Neumann boundary condition defined in Eq. (4.30) corresponds to the heat flux due to convection to the fluid in the microchannels; in the case of Eq. (4.31), the heat flux represents the thermal load from the heat source.

In the context of this work, steady-state modeling is desired and therefore the mathematical model reduces to:

$$\left( \nabla^2 T \right) = 0$$ for all x,y,z values in the solid domain. \hspace{1cm} (4.34)

$$\frac{\partial T}{\partial n} = -\frac{h(T_w - T_\infty)}{k}$$ for all x,y,z values located at solid/fluid boundaries. \hspace{1cm} (4.35)

$$\frac{\partial T}{\partial n} = \frac{\dot{q}_{LOAD}}{k}$$ for all x,y,z values located at heat source surface. \hspace{1cm} (4.36)

$$T = \bar{T}$$ for all x,y,z values located at bottom surface of substrate. \hspace{1cm} (4.37)

The OpenFOAM Computational Fluid Dynamics package [29] was used to solve the above formulation; the utility involved in this solution is a steady-state version of a Laplace equation solver. Gaussian finite volume integration is used as the numerical method for derivative computation and a linear interpolation scheme is implemented.
4.4.2 Heat Diffusion Solver Description

The OpenFOAM utility invoked to solve the heat conduction problem is called laplacianFoam; this application is designed to numerically solve the general Laplace equation for unsteady, isotropic diffusion:

\[ \frac{\partial u}{\partial t} + \nabla^2 (D_r \cdot u) = 0 \]  \hspace{1cm} (4.38)

where

- \( u \) is the diffusive scalar variable.
- \( D_r \) is a diffusivity constant.

OpenFOAM achieves a numerical solution using the control volume method and the ramifications of spatial and equation discretization methods are of interest in this section; since this work assumes steady conditions, time discretization in general is neglected.

Spatial discretization of the problem is accomplished by forming control volumes (cells) bounded by internal and boundary faces. The meshing algorithm stores properties of the cells, such as: bounding patches, control volume size, centroid location and number of faces, in mesh definition files permitting appropriate discretization in space to solve the governing equations at the corresponding locations. Variables are normally solved at the centroid and interpolated to necessary locations.

Equation discretization in the OpenFOAM program is dependent on the differential operator involved in the governing equation. The steady version of the governing equation at hand only presents a Laplacian differential operator; the mathematical definition of this operator is given in Eq. (4.39).
\[ \nabla^2 \equiv \partial^2 \equiv \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \frac{\partial^2}{\partial x_3^2} \]  

(4.39)

The integration over a control volume of a Laplacian term, in general, is discretized by the OpenFOAM solver according to Eq. (4.40).

\[
\int_V \nabla \cdot (\Gamma \nabla u) dV = \int_S \Gamma \nabla \cdot (\nabla u) = \sum_f \left( \Gamma_f S_f \cdot (\nabla u)_f \right) 
\]

(4.40)

where

\( f \) is a subscript representative of the set of faces surrounding the control volume.

\( u \) is represents the scalar acted upon by the Laplace operator.

\( V \) refers to the control volume.

\( S \) refers to the surface boundary of the control volume.

As seen in Eq. (4.40), discretization of the gradient term is required to finalize Laplacian term discretization. Face gradient discretization depends on the mesh properties, but most of the time, for orthogonal meshes, it is defined by Eq. (4.41).

\[
S_f \cdot (\nabla u)_f = \left| S_f \right| \frac{u_N - u_P}{|\vec{d}|} 
\]

(4.41)

where

\( N \) subscript is representative of the centre of an adjacent cell.

\( P \) subscript is representative of the centre of the control volume in question.

\(|\vec{d}|\) is the magnitude of the vector formed between the neighboring cells’ centers.

Equations (4.40) and (4.41) are the basis for equation discretization in a spatially discretized domain, which set the foundation for the numerical method involved in the solution of the heat conduction problem.
4.4.3 Heat Diffusion Solver Validation

The OpenFOAM Laplace equation solver was validated against exact solutions for 1D and 2D steady-state heat diffusion test cases.

Let us first consider a 1 m long wire with infinitesimal cross-sectional area, such that transverse temperature gradients are negligible; one end is heated by a source that imposes a 300 K/m uniform temperature gradient and the other end is held at a constant temperature of 500 K. Equation (4.42) represents the exact solution for the general 1D steady heat diffusion problem described above. The computed steady state temperature values from the OpenFOAM solver are compared to the theoretical solution in Fig. 33.

\[
T(x) = T| + \frac{\partial T}{\partial x} (1-x) \quad (4.42)
\]

\(T|\) is the prescribed constant temperature value; for the selected test case it is 500 K.

\(\frac{\partial T}{\partial x}\) is the prescribed uniform temperature gradient value; for the selected test case it is 300 K/m.

For the 2D test case, consider steady-state diffusion of heat in a 5 m by 10 m rectangular block, held at constant temperature of 0°C on all sides, except for the bottom face which is held at 100°C. The exact solution for this steady-state problem can be derived by implementing the integral-transform technique [34] and the temperature distribution function reduces to Eq. (4.43).

\[
T(x, y) = \frac{4}{AB} \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \frac{v_n}{\beta_m^2 + v_n^2} (\sin \beta_m x)(\sin v_n y) \int_{x=0}^{A} (\phi(x) \sin \beta_m x) dx \quad (4.43)
\]
where

A is the block’s width (10 m in this test case).

B is the block’s height (5 m in this test case).

$\beta_m$ and $\nu_n$ represent the eigenvalues of the Kernel functions in the integral-transform technique (Özişik, 1989) for heat diffusion problems.

$x$ and $y$ represent the Cartesian coordinates.

$\varphi(x)$ is a function that describes the prescribed temperature along the bottom boundary.

The temperature field computed by OpenFOAM for the 2D steady-state test case is compared to a solution of Eq. (4.43) that is obtained implementing 40,000 summation terms at every $x$-$y$ location. The resulting temperature fields for both solutions are observed in Fig. 34 and Fig. 35 and a plot of the spatial variation of error of the computed solution, with respect to the analytic solution, is shown in Fig. 36.

![Figure 33. Validation of OpenFOAM for 1D heat Diffusion.](image-url)
Figure 34. Calculated Temperature Field with OpenFOAM for 2D Heat Diffusion.

Figure 35. Analytic Temperature Field Solution for 2D Heat Diffusion Test Case.
4.5 Thermal Boundary Condition Transfer Module

The thermal boundary condition transfer module (BCTM) is an essential component of the CHETSOLP package. It constitutes a two-way fluid/solid interaction package that iteratively reads flow filed data to prescribe heat flux boundary conditions in the solid domain, and reads temperature filed data to enforce temperature conditions acting upon the fluid domain. The fundamentals of the two-way interaction sequence are explained following.

Thermal boundary conditions may be transferred in any of the two domain directions: fluid to solid and solid to fluid. Let us cover fluid to solid transfer first. The discussion below applies for all patches of either domain, i.e. for boundary surfaces of all passages composing the branching microchannel heat sink.

Recall that the solid domain implies heat diffusion solution in the substrate in which the microchannels are embedded. From Eq. (4.30), it is deduced that wall
temperatures, bulk fluid temperature and thermal conductivity of the solid must be known in order to compute the corresponding prescribed heat flux, on the solid domain wall.

The fluid flow solver is launched with a set of constant wall temperatures for all microchannels which is valid for the current iteration (those values are constantly updating during the iterative process of the CHETSOLP package) and these values are used, along with the computed passage fluid temperature; thermal conductivity is a thermo physical property of the material and is known. Equation (4.30) yields the corresponding heat flux which is prescribed for a given boundary (patch).

Upon definition of heat fluxes in all patches of the solid domain (using the current flow filed data), the temperature field is solved; one of the targets of solution is to determine new wall temperatures at all patches. These boundary fields are averaged on a per passage basis to prescribe constant passage wall temperature back in the flow-field problem. Arithmetic averaging of temperatures of all nodes of a given passage’s surface is carried out and the computed value is stored for linking with the corresponding patch in the fluid domain.

A summary of the step by step procedure of commands executed by the thermal boundary condition transfer module is included following.

1. Transfer latest wall temperatures to the fluid solver input file.
2. Process computed (by fluid solver) bulk fluid temperatures and heat transfer coefficients to determine temperature gradient at the boundary for the solid domain.
3. Sweep through all patches (each patch is the surface enclosing an individual microchannel passage of the network) and average the local temperatures to obtain averaged constant temperature per passage.

4. Trigger under-relaxation and convergence-dependent mesh switches (discussed in Section 4.5).

5. Update latest wall temperature field with the relaxed wall temperature values that the dynamic relaxation feature yields.

4.6 Additional Features

Two additional features of the CHETSOLP program are discussed in this section: Dynamic wall temperature relaxation and convergence dependent switching meshing scheme.

The dynamic relaxation of wall temperatures is added for software robustness. It is the wall temperatures which drive the convergence of the solver platform; when a new temperature field is calculated, current and new wall temperature sets are used to compute a relaxed wall temperature set that is the one actually used in the subsequent iteration, rather than exactly the recently computed wall temperature values. Relaxation impacts stability and convergence rate; a dynamically increasing relaxation factor is used to favor both these conditions.

The convergence dependent meshing scheme monitors level of convergence of the wall temperatures and controls the refinement level in the solid domain meshing stage. Three levels of mesh refinement are defined by default and the switching module activates either one as a function of the residual value in the convergence history. More in depth, a coarse mesh is used for normalized residuals above 4 negative orders of
magnitude; a medium refinement level is implemented to drive the solution down to 5 negative orders of magnitude, and finally a highly refined mesh is used in the last iterations until an additional order of magnitude drop occurs (in the wall temperatures residual). These features and their interaction with the numerical solvers and transfer modules are explained in Figure 37; next section is a summary showing in detail the operations followed by the CHETSOLP package in order to compute the objectives: pressured drop, mass flow rate and temperature uniformity. An example of the results (on a step by step basis) yielded by CHETSOLP is covered in Appendix B.

4.7 Objectives Calculation

1. COOLNET runs and computes flow-field based on the latest updated values of wall temperature for all passages (in iteration 1, these values are guessed).
2. BCTM intervenes to obtain bulk temperatures and heat flux rates in order to setup heat conduction problem.
3. LaplacianFoam runs temperature field calculation based on the heat source and the heat fluxes at boundaries with microchannels.
4. BCTM processes computed temperature field and determines the new averaged per passage wall temperatures.
5. Relaxation of wall temperatures occurs.
6. Wall temperature residual is calculated.
7. If residual has converged, CHETSOLP proceeds to step 10.
8. Mesh switching is effective depending on the computed residual.
9. BCTM prescribes constant passage wall temperatures in COOLNET input files for next iterative step; Go back to step 1. with the updated wall thermal field.
10. Mass flow rates in all level networks (given by COOLNET) are added to deliver the effective mass flow rate of the multi-story network.

11. Overall pressure drop is computed similarly by comparing the total pressure values in the inlets and outlets of the converged COOLNET flow-field.

12. Statistical analysis is performed on the heat source face to assess variation of the converged temperature field at this boundary; it leads to calculation of a coefficient of variance, which acts as the metric for temperature uniformity.

13. Objective values calculated above are stored and returned to the optimizer which invoked the CHETSOLP execution.

Figure 37. CHETSOLP/MOPSO Integration.
CHAPTER V
OPTIMIZED THERMAL MANAGEMENT

5.1 Definition of the Optimization Problem

3D networks of microchannels inside a silicon solid substrate are considered. Figure 38 shows a sample 3D multiple-story microchannel network.

A thermal load of up to 1000 W/cm² exists at the top surface of the substrate. A dielectric liquid coolant at 293 K is pumped through the channels network to absorb excess heat; fluid properties are temperature-dependent and are imported from OpenFOAM’s fluid database [29]. The silicon substrate has a footprint of 1.5 cm by 1.5 cm and a thickness of 2 mm. Thermo-physical properties of silicon used are 130 W/(mK) thermal conductivity and 0.0001 m²/s thermal diffusivity. Manufacturing limitations constrain microchannel diameters to be at least 100 microns and microchannel wall roughness may vary from perfectly smooth to 50 microns [35].

Optimization of the topology and geometrical characteristics of the microchannel heat sink is performed by means of the Multi-Objective Particle-Swarm Optimization algorithm developed and adapted for this work; the objectives of the optimization study are pressure drop minimization, mass flow rate minimization and maximization of the temperature distribution uniformity at the heated surface, where the heat source is located. Recall from Chapter 2 that the MOPSO optimizer has been successfully validated against renowned Multi-objective optimization test cases proposed by Zitzler [28].

An automatic package (CHETSOLP) has been developed which executes geometry generation, conjugate heat transfer solution and flow/temperature field analysis
for optimization objectives computation. The conjugate heat transfer solution is achieved by simultaneous prediction of the internal flow-field in the microchannel network and internal temperature field in the solid substrate [36]; the MOPSO optimizer has been adapted to invoke the CHETSOLP platform for objective evaluation purposes.

![Multiple-floor network diagram]

**Figure 38.** Optimization Problem Definition.

### 5.2 Optimization Study Results

This section summarizes the results obtained from a constrained optimization study that intends to demonstrate the capability of the optimization framework developed for the optimization of microchannel heat sinks. Results of this optimization study are summarized in Table 3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum clustering factor along flow direction</td>
<td>2.5</td>
</tr>
<tr>
<td>Minimum clustering factor along flow direction</td>
<td>-2.5</td>
</tr>
<tr>
<td>Maximum clustering factor along transversal direction</td>
<td>2.5</td>
</tr>
<tr>
<td>Minimum clustering factor along transversal direction</td>
<td>-2.5</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Minimum branching quantity at junctions</td>
<td>0</td>
</tr>
<tr>
<td>Maximum branching quantity at junctions</td>
<td>2</td>
</tr>
<tr>
<td>Minimum microchannel hydraulic diameter</td>
<td>100μm</td>
</tr>
<tr>
<td>Maximum microchannel hydraulic diameter</td>
<td>250μm</td>
</tr>
<tr>
<td>Minimum relative roughness at microchannel walls</td>
<td>0% (smooth)</td>
</tr>
<tr>
<td>Maximum relative roughness at microchannel walls</td>
<td>20%</td>
</tr>
<tr>
<td>Minimum vertical clustering factor of planar networks</td>
<td>-1.5</td>
</tr>
<tr>
<td>Maximum vertical clustering factor of planar networks</td>
<td>1.5</td>
</tr>
<tr>
<td>Number of floors</td>
<td>3</td>
</tr>
<tr>
<td>Number of branching levels</td>
<td>6</td>
</tr>
<tr>
<td>Mirroring factor</td>
<td>1</td>
</tr>
<tr>
<td>Thermal load</td>
<td>400 W/cm²</td>
</tr>
<tr>
<td>Population size (Npop)</td>
<td>100</td>
</tr>
<tr>
<td>Inertial factor (α)</td>
<td>0.25</td>
</tr>
<tr>
<td>Individuality factor (β_i)</td>
<td>1.4</td>
</tr>
<tr>
<td>Sociability factor (β_C)</td>
<td>1.4</td>
</tr>
<tr>
<td>Number of variables</td>
<td>79</td>
</tr>
<tr>
<td>Number of objectives</td>
<td>4</td>
</tr>
<tr>
<td>Maximum number of population generations</td>
<td>75</td>
</tr>
<tr>
<td>Maximum number of function calls</td>
<td>11,250</td>
</tr>
</tbody>
</table>
The optimizations study described above yields the following Pareto-front after 75 optimization generations. Pareto-front is 4-dimensional and therefore, one objective is omitted in both Figures 39 and 40 in order to visualize the relationship between the objective functions clearly.

Figure 39. Pareto Front.

Figure 40. Pareto-Optimal Solutions After 75 MOPSO Generations.
A radial basis function program was used to approximate the final Pareto-front from the scattered data that the optimizations study yields. The fitted surface representing the Pareto-front is shown in Fig. 41.

**Figure 41.** Pareto Front Approximation.

Figures 42 through 47 clearly establish the relationship between the objective functions in the Pareto-front solutions.
Figure 42. Pareto Front (Total Heat Removed vs. Mass Flow Rate).

Figure 43. Pareto Front (Total Heat Removed vs. Pressure Drop).
Figure 44. Pareto Front (Mass Flow Rate vs. Pressure Drop).

Figure 45. Pareto Front (Heated Surface Temperature CV vs. Total Heat Removed).
The above figures demonstrate that mass flow rate and total heat removal trade-off are responsible for driving the Pareto-front to its final state. Moreover, it is observed that for the thermal load used in this study, 3D branching network of microchannels successfully remove excess heat and deliver uniform cooling at the heated surface.
Coefficients of variance of temperature at this location are as low as 0.01% and in no case larger than 0.06%.

Figures 48 through 50 show the values of the planar outlets clustering and branching levels clustering (in all 3 floors); clear description of clustering situations is attached in Appendix C.

Figure 48. First Floor Outlet Passages Clustering Factors.

Figure 48 shows slight predominance of negative clustering of outlet passages in floor 1, which implies clustering towards outer walls (see Appendix C).
It is drawn from Figure 49 that negative branching level clustering (backward clustering) favors the multi-objective criteria of this optimization study. 94% of the solutions in the Pareto-front depict this clustering pattern for the first floor. See Appendix C for clear explanation on backward clustering behavior.

Figure 50. Second Floor Outlet Passages Clustering Factors.
Figure 50 shows 95% of the Pareto-solutions clustering their outlet passages towards the side walls on the second floor.

![Figure 50](image)

**Figure 50.** Second Floor Branching Levels Clustering Factors.

89% of the Pareto-solutions depict backward clustering of branching levels on second floor, as observed in Figure 51.

![Figure 51](image)

**Figure 51.** Second Floor Branching Levels Clustering Factors.

89% of the Pareto-solutions depict backward clustering of branching levels on second floor, as observed in Figure 51.

![Figure 52](image)

**Figure 52.** Top Floor Outlet Passages Clustering Factors.
Figure 52 shows that in the top floor there is a tendency to cluster outlet passages towards core of the planar network.

![Diagram](image)

**Figure 53.** Top Floor Branching Levels Clustering Factors.

Backward clustering of branching levels for the third (top) floor is observed in Figure 53. 86 solutions present this branching levels clustering patterns.

Figures 54, 55 and 56 summarize hydraulic diameter variations of the Pareto-optimal solutions in floors 1, 2 and 3 respectively. It will be observed below that smaller hydraulic diameter at the top planar network favor the multi-objective criteria of this problem. However, for the rest of the floors no clear pattern is observed.
Figure 54. First Floor Hydraulic Diameters for Optimal Solutions.

Figure 55. Second Floor Hydraulic Diameters for Optimal Solutions.
Figure 56. Top Floor Hydraulic Diameters for Optimal Solutions.

Figure 56 shows that in roughly 70% of the Pareto-solutions, the passages in the top floor have the smallest allowable hydraulic diameter of 100 μm.

Figures 57 through 59 summarize relative wall surface roughness variations of the Pareto-optimal solutions in floors 1, 2 and 3 respectively. It will be observed that variable values are well spread all over the Pareto space. This is indicative of a low transcendence of this design variable in the multi-objective criteria governing this optimization problem. It is concluded that the impact of this variable is strongly dominated by the rest.
**Figure 57.** First Floor Relative Wall Roughness for Optimal Solutions.

**Figure 58.** Second Floor Relative Wall Roughness for Optimal Solutions.
Figure 59. Top Floor Relative Wall Roughness for Optimal Solutions.

Figure 60 shows that the vertical clustering factor for all solution in the Pareto-front is identical; it equals the upper limit setting for this variable. A positive 1.5 clustering factor implies smooth clustering towards the heated surface.

Figure 60. Vertical Clustering Factor of Planar Networks for Pareto-Optimal Solutions.
Analysis of the branching pattern of all members in the final population shows that the vast majority of junction nodes are set to perform dichotomous branching in all Pareto-solutions. Figures 61 through 63 are examples proving this conclusion, where the branching pattern at specific branching levels and for specific junction locations are shown for all Pareto-optimal solutions. For a dichotomously constrained optimization study, this behavior is expected. In Figures 61 through 63, the y-axis represents the number of branches emanating out of a given junction of the network; each data point represents a Pareto-optimal solution.

**Figure 61.** Branching Pattern at First Floor’s First Branching Level.
Figure 62. Branching Pattern at Second Floor’s First Branching Level.

Figure 63. Branching Pattern at Third Floor’s Second Branching Level.

Following is the analysis of 3 different solutions in the Pareto-front and visualization of their topology and the temperature profile they yield at the top surface. Since the trade-off between total heat removed and mass flow rate governs the optimization study, the selected solutions are: 1) a solution biased towards favoring total
heat removal, 2) a solution biased towards favoring minimization of mass flow rate and 3) a median solution favoring the 2 objectives simultaneously.

Figure 64. Selected Pareto-solutions.

Branching topology in all floors for Pareto solution 38 is shown in Figs. 65-67.

Figure 65. First Floor Topology for Pareto-solution 38.
Figure 66. Second Floor Topology for Pareto-solution 38.

Figure 67. Third Floor Topology for Pareto-solution 38.
Figure 68. Top Surface Temperature Distribution of Pareto-solution 38.

It can be observed in Fig. 68 that an average temperature of 362.56 K is maintained on the top surface with this optimized microchannel heat sink. The above temperature distribution is calculated to have a $\text{CV}=0.472\%$, demonstrating uniform-cooling capability.

Branching topology in all floors for Pareto solution 49 is shown in figures 69 through 71.
Figure 69. First Floor Topology for Pareto-solution 49.

Figure 70. Second Floor Topology for Pareto-solution 49.
The temperature distribution shown in Fig. 72 is calculated to have a CV = 0.997%. Pareto-solution 49 maintains acceptable temperature range (90°C +/- 3°C) at the heated surface.
Branching topology in all floors for Pareto solution 85 is shown in figures 73 through 75.

**Figure 73.** First Floor Topology for Pareto-solution 85.

**Figure 74.** Second Floor Topology for Pareto-solution 85.
Figure 75. Third Floor Topology for Pareto-solution 85.

Figure 76. Top Surface Temperature Distribution of Pareto-solution 85.

Figure 76 shows acceptable temperature ranges and profile uniformity at the heated surface for Pareto-solution 85. The above temperature distribution is calculated to have a CV=1.189%. 
Figure 77 shows the relationship between the computed pumping power requirement and total heat removed for all Pareto-solutions. A break-even line is drawn which delineates the power efficiency limit, i.e. all solutions below the break-even line remove more total heat than they consume power in order to pump the coolant.

**Figure 77.** Pumping Power vs. Total Heat Removed.

**Figure 78.** Pumping Power vs. Total Heat Removed Detail for all Pareto-Solutions.
It is observed that 100% of the Pareto solutions are power efficient, i.e. the total amount of heat removed is larger than the power required to pump the coolant through the microchannels network. In conclusion, the low flow rates in all Pareto-optimal solutions indicate pumping power efficiency and pressure drop levels are far within the acceptable ranges proposed in literature for fluid delivery devices.

In an effort to increase total heat removal for appropriate thermal management of high heat fluxes, a revised optimization study was ran. The revised optimization effort involves an additional floor in the microchannel network and fluid dispensing from different boundaries of the microchannel network.

Fluid/Thermal conditions and MOPSO performance settings remain unchanged from original optimization study described in Table 3. However, there is a conceptual difference in the objective function used during this optimization. Temperature uniformity was not used as an optimization objective for this study. Instead, it is implicitly enforced by discarding any solution that does not deliver a temperature profile with average temperature below 100°C and variation below +/- 3°C.

In the original 3-floor optimization run all planar networks were fed from the same boundary, i.e. flow direction was identical in all cases. In the revised 4-floor run, each planar network is fed from a different boundary: floor 1 is fed from south, 2 from west, 3 from north and 4 from east. Implementation of such fluid distribution system favors uniform-cooling.

The objectives in the revised 4-floor optimization study are maximization of total heat removal and minimization of pumping power (uniform cooling is implicitly
satisfied). After 75 optimization generations, the Pareto front is summarized in Table 4 and it is shown in Fig. 79.

Table 4. Objective Function Values from 4-floor Optimization Study.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>131.33129040</td>
<td>4.40764524</td>
<td>29.79624797</td>
</tr>
<tr>
<td>37</td>
<td>135.67554832</td>
<td>4.60735300</td>
<td>29.44761305</td>
</tr>
<tr>
<td>44</td>
<td>140.00923729</td>
<td>4.79883030</td>
<td>29.17570086</td>
</tr>
<tr>
<td>2</td>
<td>145.26392002</td>
<td>5.46735360</td>
<td>26.56932963</td>
</tr>
<tr>
<td>78</td>
<td>150.03677882</td>
<td>6.29838398</td>
<td>23.82147219</td>
</tr>
<tr>
<td>76</td>
<td>154.95052780</td>
<td>7.89845136</td>
<td>19.61783655</td>
</tr>
<tr>
<td>5</td>
<td>158.09637282</td>
<td>9.09277222</td>
<td>17.38703764</td>
</tr>
<tr>
<td>18</td>
<td>160.22436320</td>
<td>9.89272789</td>
<td>16.19617612</td>
</tr>
<tr>
<td>22</td>
<td>165.88272772</td>
<td>12.76728920</td>
<td>12.99279159</td>
</tr>
<tr>
<td>40</td>
<td>169.12882892</td>
<td>16.02892902</td>
<td>10.55147407</td>
</tr>
<tr>
<td>19</td>
<td>169.88560090</td>
<td>16.70786684</td>
<td>10.16800065</td>
</tr>
</tbody>
</table>

Efficiency ratios in the above table depict the power of the optimized 4-floor microchannel heat sinks. Efficiency limit is shown in Fig. 80 where it is observed that all optimal solutions are located below the break-even limit. Moreover, three solutions are selected for analysis, as shown in Fig. 80. Solution 23 is biased towards favoring low pumping power requirement. Solution 76 is a median solution in the Pareto front. Finally, solution 19 is biased towards favoring total heat removal.
Let us analyze Pareto-solution 23. The calculated temperature distribution at heated surface for this 4-floor microchannel heat sink is plotted in Fig. 81.
Average temperature is within the acceptable 373 K limit and variation is observed to be less than +/- 3 degrees.

The branching topology on 1st floor (lowest floor) is shown in Fig. 82.
Figure 82 shows clustering of planar outlets towards core of the planar network for the first floor. Forward clustering of branching levels (clustering of levels in direction of flow) is also observed. The average hydraulic diameter of microchannels in the first floor is 152.18μm.

The branching topology on 2nd floor is shown in Fig. 83.

![Figure 83. Second Floor Branching Topology for Pareto-solution 23.](image)

Second floor depicts clustering of planar outlets towards outer walls. It also depicts uniform spacing of branching levels along flow direction. The average hydraulic diameter in the second floor is 157.44μm.
The branching topology on 3rd floor is shown in Fig. 84.

**Figure 84.** Third Floor Branching Topology for Pareto-solution 23.

Third floor depicts slight clustering of planar outlets towards outer walls. Uniform spacing of branching levels along flow direction is also present in floor number 3. The average hydraulic diameter in the third floor is 199.19μm.
The branching topology on 4th floor is shown in Fig. 85.

![Branching Topology](image)

**Figure 85.** Fourth Floor Branching Topology for Pareto-solution 23.

Fourth floor depicts heavy clustering of planar outlets towards core of planar network. Fourth floor depicts clear clustering of branching levels forward. Average hydraulic diameter in fourth floor is 197.11μ.m.

Figure 86 shows hydraulic diameter variation along branching levels in all floors, for Pareto-optimal solution 23:
Let us analyze Pareto-solution 19. The calculated temperature distribution at heated surface for this 4-floor microchannel heat sink is plotted in Fig. 87.
Average temperature is lower, with respect to Pareto-solution 23, since this solution removes a larger amount of heat. The temperature variation is observed to be within a recommendable +/- 3 degrees range.

The branching topology on 1st floor (lowest floor) is shown in Fig. 88.

![First Floor Branching Topology for Pareto-solution 19.](image)

**Figure 88.** First Floor Branching Topology for Pareto-solution 19.

It is observed in Fig. 88 that the first floor depicts maximum clustering of planar outlets towards core of the planar network. Moreover, high forward clustering (clustering of levels in direction of flow) of branching levels is evident. The average hydraulic diameter of a microchannel in the first floor is 209.2μm.
Figure 89. Second Floor Branching Topology for Pareto-solution 19.

Figure 89 shows that in the second floor, planar outlets tend to cluster towards the outer walls. Backward clustering of branching levels is observed in this Figure as well. Average hydraulic diameter in the second floor is calculated as 238.28μm.

Figure 90. Third Floor Branching Topology for Pareto-solution 19.
Third floor depicts very slight clustering of planar outlets towards outer walls. Uniform spacing of branching levels along flow direction is observed. Average hydraulic diameter in third floor is $236.89\mu\text{m}$.

**Figure 91.** Fourth Floor Branching Topology for Pareto-solution 19.

Fourth floor depicts heavy clustering of planar outlets towards core of planar network. Figure 91 shows clear clustering of branching levels forward. Average hydraulic diameter in the fourth floor is $144.26\mu\text{m}$.

A summary of hydraulic diameter variation along branching levels in all floors, (for Pareto-optimal solution 19) is plotted in Fig. 92.
Finally, let us analyze the median solution in the Pareto front: Pareto-solution 76.

Temperature distribution at heated surface for solution 76 is shown in Fig. 93.
Branching topology on all floors can be observed in Figs. 94 through 97.

**Figure 94.** First Floor Branching Topology for Pareto-solution 76.

**Figure 95.** Second Floor Branching Topology for Pareto-solution 76.
Figure 96. Third Floor Branching Topology for Pareto-solution 76.

Figure 97. Fourth Floor Branching Topology for Pareto-solution 76.

Figure 98 shows hydraulic diameter variation along branching levels in all floors, for Pareto-optimal solution 76:
This optimization study on 4-floor microchannel heat sinks leads to conclude that 400 W/cm² thermal loads can be successfully managed with optimized multiple-story networks of microchannels, delivering uniformly cooled surfaces. Their efficiency is comparable to that of arrangements of multiple layers of uniformly distributed straight-through passages. For reference, a 4-floor arrangement where each floor has 8 uniformly distributed straight-through passages was evaluated under identical thermal load and available total pressure head to drive the flow. Each of the four floors in this test case was located at the elevation where each of the four floors were found to be in the four-floor optimized 3D network test case. Also, diameters of straight-through passages on each of the four floors corresponded to average optimized diameter of each of the 2D optimized networks found on each of the four floors in previous example.
Total heat removal for the 4-floor straight-through passages arrangement was calculated as 160.7 W with a pumping power requirement of 14.2 W; this implies an efficiency ratio of 11.26. Notice that Pareto-optimized solution number 18 (refer to Table 4) presents similar total heat removal (160.22 W) with a lower pumping power requirement of 9.9 W; corresponding efficiency ratio is 16.2. Hence, the optimized microchannel heat sink implies a 50% improvement in power efficiency with respect to basic straight-channel arrangements, for uniform thermal load conditions. Figure 99 shows the temperature distribution yielded by the straight-through passages arrangement.

![Figure 99](image)

**Figure 99.** Temperature Distribution at Top Face with 4-floor Straight-Through Passages Arrangement.

Up until this point, uniformly distributed thermal loads have been considered. A more realistic model of microelectronics’ power dissipation involves non-uniform thermal loads at the heated surface of the substrate. Under such conditions, the benefits of
optimized branching networks over basic straight-through passages arrangements become highly evident.
CONCLUSIONS

A multi-objective design optimization methodology for the development of cost-effective and reliable thermal management solutions for microelectronics has been formulated, developed and demonstrated. A self-sufficient software package has been assembled for the analysis and performance evaluation of three-dimensional microchannel heat sinks. The package includes an automatic generator of three-dimensional branching networks of microchannels, a quasi-1D thermo-fluid analysis code, a 3D steady heat diffusion analysis code and a thermal boundary conditions transferring module. A multi-objective particle-swarm optimization algorithm has been developed, tested and coupled with the analysis software package, creating the optimization framework that made this work possible.

In the optimization framework’s demonstration phase, optimization studies on 3D multi-floor microchannel networks were carried out targeting optimization of their geometrical and topological characteristics, based on multi-objective criteria that included: 1) maximization of the total heat removal, 2) minimization of the pumping power requirement, and 3) maximization of uniform-cooling capability. The results of such studies yielded optimized microchannel heat sink topologies with optimal geometric characteristics delivering maximally uniform cooling patterns and exhibiting power efficiency improvements with respect to basic microchannel heat sink topologies. Thermal loads of up to 4 times those of current conventional computer chips were managed with the optimized 3D networks of microchannels.
This research work covered three main areas: 1) multi-objective optimization, 2) conjugate heat transfer analysis and 3) optimized thermal management solutions development.

In the aspect of this work that pertains to multi-objective optimization, the following conclusions can be drawn:

1. The Multi-Objective Particle-Swarm Optimization algorithm (MOPSO) that has been developed in this work has been shown to successfully search and find optimal solutions based on multi-objective criteria, producing highly reliable results.

2. The MOPSO algorithm has been successfully adapted for parallel-computing environments in order to tackle optimization problems involving CPU-exhaustive processes for evaluation of the objective functions.

3. The validation results of the developed MOPSO algorithm on pertinent multi-objective test cases demonstrate its impressive performance with respect to other well-known algorithms, such as NSGA-II, MPP, SPEA, NSDE and MOHO.

The second area that this work encompasses deals with automatic generation of multiple-story microchannel heat sinks and conjugate heat transfer analysis of such geometries for heat sinks’ performance evaluation. The following conclusions are relevant to this area of the investigation:

1. The automatic microchannel network generation program developed in this work is a unique instrument to create a visibly infinite number of unique 3D microchannel heat sinks with various branching topologies and geometrical properties.
2. The modified quasi-1D thermo-fluid solver version developed in this research represents an extraordinarily efficient tool to solve fluid flow problems involving incompressible, steady, viscous conditions with heat transfer. The conservation laws from the Navier-Stokes equations are successfully enforced in the modified COOLNET version, achieving over 90% accurate flow-field solutions in as much as 100 times less time than commercially available well-known NSE solvers.

3. The assembled conjugate heat transfer analysis software package (CHETSOLP) is a high-fidelity tool to model conjugate heat transfer. Physical phenomena in electronics’ thermal management devices can be assessed with this analysis software. The built-in convergence-dependent switching logic developed for this software package makes it very robust and efficient.

Demonstration of the proposed methodology for multi-objective design optimization of microelectronics’ thermal management solutions was accomplished by running a set of optimization studies. Some of the most important conclusions from the studies are summarized below:

1. The developed optimization framework was implemented to optimize branching topology, hydraulic diameter, wall surface roughness and passage length of all microchannels in a multi-story 3D heat sink.

2. Optimized 3D networks of microchannels address the high power density and low operating temperature issues in microelectronics up to thermal loads of 500 W/cm\(^2\). Remarkable heated surface temperature uniformity is achieved. In some cases the coefficient of variance is as low as 0.01%. Heated surface’s temperature
variation is maintained within a recommended +/- 3°C range, and below critical operating conditions of 110°C.

3. The calculated temperature profiles at the heated surface indicate that thermal gradients along flow direction are not high enough to pose a non-uniform cooling condition when implementing 3D networks. Moreover, thermal stresses are not foreseen to pose a structural problem for the circuitry.

4. Pumping power requirements for the optimized microchannel heat sinks yielded by optimization studies are as much as 50% lower than those of other technologies such as jet impingement.

5. The metric for power efficiency was total heat removal vs. pumping power. Based on this metric, optimized 3D microchannel heat sinks imply up to a 48% increase in performance efficiency with respect to basic microchannel topologies, such as straight-through passages.
FUTURE WORK

The proposed methodology may be expanded in several aspects and this section suggests areas of potential for future investigation efforts. The optimization framework developed in this thesis is modular and any of its components may be replaced. Hence, expansion of this work should not entail fundamental changes in the basic background proposed for multi-objective design optimization of 3D microchannel heatsinks.

In the scope of this work, all analysis codes solved for steady-state solutions. Engineering systems always involve transient events and therefore, more accurate modeling can be accomplished by executing unsteady fluid flow and heat diffusion solvers.

The current program for automatic generation of 3D topologies does not consider coolant recollection. Design of thermo-fluid systems should always take into account such feature. Coolant recollection, routing and recirculation may be added in order to obtain more realistic results.

The conjugate heat transfer analysis software package is currently linked to a single multi-objective optimizer. Robustness and optimization power of hybrid optimization algorithms should be benefited from, by coupling the analysis software package to such optimization program.

One of the major challenges in this work was to handle high heat fluxes while simultaneously maintaining uniform temperature profiles in order to avoid high thermal stresses on the microelectronic components. A structural analysis solver may be integrated for calculation of critical stresses on electronic components located at the heated surface.
The advances in nanotechnology may lead to the development of pressure and temperature sensors accurate enough to build an experimental testing platform to verify the results obtained with this multi-objective design optimization platform.
REFERENCES


APPENDICES
Appendix A: Fluid Domain Definition and Matrix System Example of the Quasi-1D Analysis Code (COOLNET).

The Input file for COOLNET defines the fluid domain and stores all physical information necessary for quasi-1D approximation of the flow-field in the network. It is composed of 4 major blocks: convergence, boundary, connectivity and geometry.

The convergence block (see sample below) stores the relaxation scheme factors, the minimum residual and the maximum number of iterations allocated for convergence; additionally, the number of microchannels, nodes and inlet/outlet ports are defined.

```
relax  relmax  resmin  itermax
0.005000  0.100000  0.000010  150
kk nn mm
51 52 14
```

The boundary block has the port (inlet/outlet) number, the pressure boundary condition for all ports and the thermal boundary condition for the inlets only.

```
m  inlet(m)  bc(m)  tbc(m)
 1   1    180.000000  293.0000
 2  -2    101.300000
 3  -3    101.300000
 4  -4    101.300000
 5  -5    101.300000
 6  -6    101.300000
 7  -7    101.300000
 8  -8    101.300000
 9  -9    101.300000
10 -10    101.300000
11 -11    101.300000
12 -12    101.300000
13 -13    101.300000
14 -14    101.300000
```

The connectivity block defines the node numbers of the end points for all the microchannel elements in the network.
The geometry block defines cross-sectional area, diameter, length, relative roughness and wall temperature for every single microchannel element in the network.

The geometry of the COOLNET program is defined by the node numbering, element numbering and their connectivity. Figure 100 shows a simple COOLNET-type
domain, where the labels correspond to the node numbering; elements are numbered advancing from lowest to largest connectivity level (branching level), and within a connectivity level, numbering advance is done from left to right until the entire branching level is swept and counted.

![Figure 100. Simple COOLNET-type Geometry.](image)

From the previous explanation about numbering and based on Fig. 100, the coupled momentum/continuity balance is solved from the matrix system in Fig. 101. The subscripts in the terms of the matrix of coefficients are representative of the element number to which they apply. The subscripts in the equivalent pressure terms in the vector of unknowns are linked to nodal locations. Finally, the subscripts for the mass flow rates in the vector of unknowns correspond to the elements since this quantity is averaged for a given element.
Figure 101. Example of the Continuity/Momentum Matrix System.
Appendix B: Sample Execution Process of the Coupled 3DBNGEN/CHETSOLP Package.

Let us assume that the network definition parameters have been set; whether they are of interest or were randomly generated is not of importance for this discussion.

The first step in the process of automatic conjugate heat transfer modeling is that 3DBNGEN generates the planar networks. See Figs. 102-104.

![Figure 102. Planar Network on Level 1.](image)

![Figure 103. Planar Network on Level 2.](image)
Figure 104. Planar Network on Level 3.

After planar networks are formed; the multi-level network is assembled and the computational domains module runs to prepare solvers for numerical simulation. Surface mesher then runs to create network surface mesh. See Fig. 105 to observe assembled multi-story network and Fig. 106 for example of meshed fluid/solid boundary surface.

Figure 105. Assembled 3D Microchannel Network.
Solid Mesher automatically executes void calculation and extraction to create solid domain mesh; for the example network topology herein, the substrate domain mesh is illustrated in Fig. 107.

On such discretized spatial arrangement the heat diffusion solver is implemented to yield temperature field calculation; an example may be seen in Figs. 108 and 109.
Figure 108. Sample Temperature Field Solution from Laplacian Solver.

Figure 109. Temperature Field Solution for Example Network.
Appendix C: Exponential Point Clustering Method.

Among the most crucial parameters in the generation of a given network are the clustering factors; the reason for this is that they impact both positioning and length of microchannels. The junctions defining the microchannel ends are distributed in the planar space according to their total number and are subsequently clustered following the single exponential function method proposed by Thompson et al [37]. This point clustering approach maps a given location to its clustered location as illustrated in Figure 110.

![Figure 110. Single-Exponential Point Clustering.](image)

The mapping process is governed by the exponential expression in Eq. (AC.1).

\[ x_{\text{clustered}} = \frac{e^{\beta p} - 1}{e^{\beta} - 1} \quad \text{(AC.1)} \]

Recall from section 3.2 that 3 types of clustering factors are necessary to fully define a 3D branching network: vertical clustering factors, branching levels clustering factors and planar outlets clustering factors. The figures below show different clustering examples produced by different clustering factors selection.
Let us discuss the branching levels clustering and planar outlets clustering patterns first. Consider first a planar network of passages where branching levels are uniformly distributed along the z-direction and planar outlets also present uniform distribution along the x-direction, as shown in Fig. 111; such uniform distribution is accomplished by assigning no clustering, i.e. a clustering factor of 0.

**Figure 111.** Uniformly Distributed Planar Network in X and Z Directions.

Positive values for the branching levels clustering factor drive the branching levels' locations forward (towards the flow direction or the +z-direction). An example of forward clustering is observed in Fig. 112, where a branching levels clustering factor of 3.5 was used to generate such planar network.
Figure 112. Example of Forward Clustering of Branching Levels.

Negative values for the branching levels clustering factor drive the branching levels' locations backward (against the flow direction or the \(-z\) direction). An example of backward clustering is observed in Fig. 113, where a branching levels clustering factor of negative 4.25 was used to generate such planar network.
Figure 113. Example of Backward Clustering of Branching Levels.

Positive values for the planar outlets clustering factor drive the outlet passages towards the core of the network. An example of core clustering is observed in Fig. 114, where a planar outlets clustering factor of 3.8 was used to generate such planar network.

Figure 114. Example of Core Clustering of Microchannels.
Negative values for the planar outlets clustering factor drive the outlet passages towards the sides of the planar network. An example of side or outer clustering is observed in Fig. 115, where a planar outlets clustering factor of negative 2.9 was used to generate such planar network.

Figure 115. Example of Outer Clustering of Microchannels.
Appendix D: Calculation of Minor Losses.

In the scope of this work, minor losses due to four types of minor losses must be addressed: expansions, contractions, elbow bends and T-junctions. The energy dissipation in such junctions is accounted for by implementing an accurate method, based on experimental data, proposed by King [38]. It is proposed that losses in fittings are assessed by calculating a loss coefficient, dependent on the upstream velocity.

In the case of expansions and contractions, the loss coefficient is obtained from AD.1 and AD.2, respectively:

\[ K_f = (1 + 3.2f) \left( I - \frac{D_{upstream}^2}{D_{downstream}^2} \right)^2 \]  \hspace{1cm} (AD.1)

\[ K_f = (0.6 + 1.92f) \left( \frac{D_{upstream}^2}{D_{downstream}^2} \right) \left( \frac{D_{upstream}^2}{D_{downstream}^2} - 1 \right)^2 \]  \hspace{1cm} (AD.2)

In equation AD.1 and AD.2, \( f \) represents the friction factor calculated from upstream flow conditions.

In the case of elbow and T-junctions, the loss coefficient is also a function of the upstream flow conditions and the passage’s hydraulic diameter, and it is obtained from AD.3:

\[ K_f = \frac{K_1}{Re} + K_2 \left( 1 + \frac{0.0254}{D_h [in]} \right) \]  \hspace{1cm} (AD.3)

The table below summarizes the values suggested for \( K_1 \) and \( K_2 \) factors, in King’s “Practical Fluid Flow”. These values were used in the modified COOLNET version that was run during conjugate heat transfer analysis.
<table>
<thead>
<tr>
<th>Junction Type</th>
<th>$K_1$</th>
<th>$K_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elbow</td>
<td>800</td>
<td>0.4</td>
</tr>
<tr>
<td>T-junction</td>
<td>500</td>
<td>0.7</td>
</tr>
</tbody>
</table>

The loss coefficients in equations AD.1 through AD.3 are used to calculate the number of velocity heads lost by the fluid upon flowing through the junction. The overall validity of the minor losses method was assessed by testing it against a well-known NSE solver (Ansys CFX).

*Elbow Minor Losses Validation*

Consider the flow of air through a 90° bend formed by two passages; the following conditions apply:

1. Length of passages is 0.5m.
2. Hydraulic diameter of both passages is 0.01m.
3. Reynolds number at entrance is $8 \times 10^4$ (fully-developed flow assumed).
4. Inlet total pressure of 140 [kPa].
5. Air enters at a temperature of 300 [K].
6. Adiabatic, smooth walls.
7. Atmospheric pressure at the outlet.

For the above test case, a computational grid of 165,000 elements was generated to solve 3D NSE with Ansys CFX software. Flow-field data was analyzed to obtain average pressure values along the passages, and these values were compared to the pressure-field solution yielded by the modified version of the quasi-1D thermo-fluid analysis code.
Consider the flow of air through a passage which merges onto a T-junction; the following conditions apply:

1. Length of passages is 0.5m.
2. Hydraulic diameter of both passages is 0.01m.
3. Reynolds number at entrance is $8 \times 10^4$ (fully-developed flow assumed).
4. Inlet total pressure of 140 [kPa].
5. Air enters at a temperature of 300 [K].
6. Adiabatic, smooth walls.
7. Atmospheric pressure at the outlet.

For the above test case, a computational grid of 243,000 elements was generated to solve 3D NSE with Ansys CFX software. Flow-filed data was analyzed to obtain
average pressure values along the entrance passage and one of the outlet passages, and these values were compared to the pressure-field solution yielded by the modified version of the quasi-1D thermo-fluid analysis code for the same flow conditions.

Figure 117. Minor Losses through T-Junctions.