 CMES, vol.4, no.5, pp.587-601, 2003

A Real-Coded Hybrid Genetic Algorithm to Determine Optimal Resin Injection Locations in the Resin Transfer Molding Process

R. Mathur[1], S. G. Advani[2] and B. K. Fink[3]

Abstract: Real number-coded hybrid genetic algorithms for optimal design of resin injection locations for the resin transfer molding process are evaluated in this paper. Resin transfer molding (RTM) is widely used to manufacture composite parts with material and geometric complexities, especially in automotive and aerospace sectors. The sub-optimal location of the resin injection locations (gates) can leads to the formation of resin starved regions and require long mold fill times, thus affecting the part quality and increasing manufacturing costs. There is a need for automated design algorithms and software that can determine the best gate and vent locations for a composite part by using the current simulation capabilities. In the work presented here, the gates are encoded into real number strings for the GA. A sensitivity gradient-based fill time optimization algorithm was developed using the process physics, which can be used as a local optimization algorithm. The global search capabilities of the GA and the local search capabilities of the sensitivity gradient-based fill time optimization algorithm were combined in two separate hybrid optimization algorithms: a serial hybrid optimization algorithm and an interactive optimization algorithm. In addition, the sensitivity gradient-based algorithm involves the computation of the gradient of the fill time with respect to the gate location coordinates. This gradient information was included in the criteria for optimization to increase the capabilities of the hybrid GA. Several RTM molds with geometric and material complexities were selected and discretized. A number of studies were performed using the pure genetic algorithm, the gradient-based optimization algorithm and the two hybrid optimization algorithms using the mold fill time and it's gradient with respect to gate location coordinates as the cost criteria. These studies were performed for the cases of a single gate and two gates. The results were benchmarked against known best solutions in terms of quality of final solutions and the computational effort required.

keyword: resin transfer molding, automated design, gate location, model-based optimization, hybrid genetic algorithms

1 Introduction

Liquid injection molding processes, such as resin transfer molding (RTM) and vacuum assisted resin transfer molding (VARTM), are an important class of manufacturing processes which have been used for net shape manufacturing of composite parts, especially in the automotive, defense and aerospace sectors. In recent years, they have been used to manufacture composite structures with geometric and material complexities such as vehicle bodies, ship sections, bridges and turbine blades.

In RTM, the reinforcement material or 'preform' is placed inside a mold. The mold is closed and the resin is injected into it at high pressure through inlet ports or 'gates'. Outlet ports or 'vents' are used to enable the displaced air to escape out of the mold. The resin impregnates the preform and polymerizes to form the solid part, which is then demolded. (Fig 1) [Fong and Advani (1998)]

The manufacture of complex composite structures by RTM presents several design and control challenges that are critical for quality and manufacturing efficiency. The most important of these is the correct location of the gates and vents in the mold surface. The resin enters the closed mold through the gates and the air is displaced through the vents. If the resin reaches the vents before the mold is saturated with the resin, it creates a "dry spot" which is a manufacturing defect. The 'dry spots', are fibrous regions in the mold not wetted by the resin thus affecting the quality of the manufactured parts. Thus, the location

[1] ME/UD-CCM, UD, Newark, DE,USA & Fluent Inc., Lebanon, NH

[2] ME/UD-CCM, UD, Newark, DE,USA (to whom correspondence is to be addressed)

[3] US-ARL, APG, Aberdeen, MD,USA

of the gates and vents is crucial to reduce the mold filling time and also to avoid "dry spots". The optimization of the filling process can decrease the process cycle time and eliminate or reduce dry spot formation.

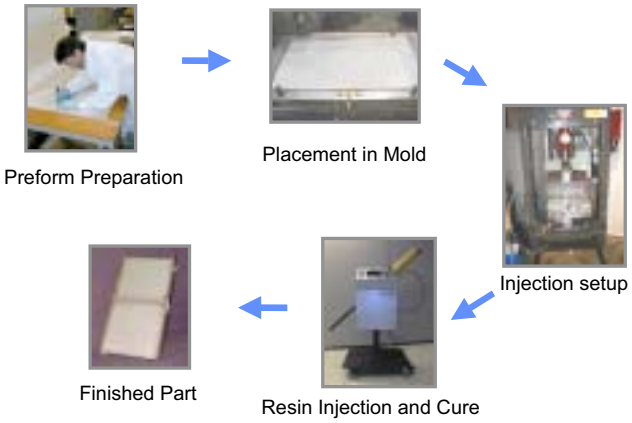


Figure 1 : The resin transfer molding process

Mold filling simulation software based on finite element modeling techniques, such as LIMS, [Simacek and Advani (2003)] can track the flow front location during the impregnation of the preform, once the user has specified the locations of inlet gates and vents. However there are as many choices for gate and vent locations as there are nodes in the finite element mesh for the mold geometry. To find the globally optimal locations, one would have to run a large number of simulations. This number could be reduced if appropriate optimization techniques are used in tandem with mold filling simulations. This paper outlines the development of design optimization algorithms and software for optimal design of the RTM process, using the available numerical simulation and optimization capabilities and benchmarks their performance.

Conventionally, optimization techniques are gradient-based, involving the computation of the derivative, or gradient, of the objective function with respect to the design parameters and then moving the design parameters along the direction indicated by this gradient. Several gradient- based algorithms have been developed for polymer processing and composite manufacturing applications using design sensitivity analysis of process models. [Mathur, Advani and Fink (2000), Smith, Tortorelli and Tucker (1998, 1998)]

Gradient-descent based techniques tend to get trapped in local minima and strongly depend on an initial guess and on the existence of derivatives. Genetic algorithms (GAs) are search algorithms that mimic natural selection and genetics to evolve 'good' solutions from a large number of alternative solutions. Genetic Algorithms have found extensive use for design optimization in the field of composite materials to address material design and design of process parameters for manufacturing. [Mahesh, Kishore and Deb (1996); Young and Yu (1997); Spoerre, Zhang, Wang and Parnas (1998); Sadagopan and Pitchumani (1998); Mathur, Advani and Fink (1999); Savic, Evans and Silberhorn (1999); Kim, Kim and Hong (1999); Shimojima (1999)]

In this work, genetic algorithms based on real number coded representations of the gate locations were coupled with sensitivity gradient-based algorithms for fill time optimization of the RTM process to form hybrid genetic algorithm optimizers, thus taking advantage of the complementary strengths of both techniques. In addition, the gradients computed by the sensitivity gradient-based algorithm were incorporated into the objective functions for the genetic algorithm providing extra information for the optimization. A number of case studies were performed testing the various combinations of hybrid algorithms and objective functions.

In the following sections, the physics of the resin flow in the mold is outlined and the previous research work in modeling and simulation for the RTM process is reviewed. The real-coded genetic algorithm and the sensitivity-based algorithms are described. The implementation of the hybrid optimization algorithms combining both optimization techniques is delineated and is followed by a description of the case studies conducted with these algorithms. Finally, the performance of these optimization techniques is evaluated by comparing them with "known" best solutions.

2 Resin Flow in RTM: Modeling and Simulation

The flow of resin in porous media is governed by Darcy's Law, which states that the velocity of a fluid flowing through a porous medium is directly proportional to the driving pressure drop:

$$\vec{u} = -\frac{\underline{\underline{K}}}{\mu}\nabla P \tag{1}$$

where $\vec{u}$ is the average velocity, ∇P is the pressure gradient in the fluid, $\underline{\underline{K}}$ is the permeability tensor and μ is the

viscosity of the resin. This can be coupled with the continuity equation for incompressible flow to give a Laplace Equation for the pressure field inside a fibrous porous media permeated by the fluid.

$$\nabla . \vec{u} = 0 \tag{2}$$

$$\nabla . (\frac{\underline{\underline{K}}}{\mu} \nabla P) = 0 \tag{3}$$

$$\begin{aligned} &B.C.s: \quad P(\vec{x}_{gate}) = P_0 \quad or \quad Q(\vec{x}_{gate}) = Q_0 \\ &P(\vec{x}_{vent}) = 0 \\ &P(\vec{x}_{flowfront}) = 0 \\ &\vec{n} \bullet (\tfrac{K}{\mu} \bullet \nabla P) = 0 \quad \textit{at the mold wall} \end{aligned} \tag{4}$$

The flow of resin in RTM has been modeled by the discretization of the governing partial differential equation, i.e. equation (3), using the finite element method. [Bruschke and Advani (1990, 1994); Antonelli and Farina (1999); Mohan, Ngo and Tamma (1999); Lin, Hahn and Huh (1998); Gauvin and Trochu (1998)] The solution involves tracking a moving boundary, using either control volume techniques (FE/CV) [Bruschke and Advani (1990, 1994); Antonelli and Farina (1999)] or the movement of a saturation field. [Mohan, Ngo and Tamma (1999); Lin, Hahn and Huh (1998)] Simulation software, using either the FE/CV technique or the saturation technique has been written and used for filling simulations of resin flow in thin shell mold geometries, i.e. Hele-Shaw flows and for full three dimensional flows in thick mold geometries. [Antonelli and Farina (1999); Mohan, Ngo and Tamma (1999); Lin, Hahn and Huh (1998); Bruschke and Advani (1991); Gallez and Advani (1997); Tan and Springer (1999)] The simulation software used for this research work is the Liquid Injection Molding Simulation (LIMS) software. [Simacek and Advani (2003), Bruschke and Advani (1990, 1991)] This is based on the FE/CV technique and is capable of simulating the flow and cure of resin in thin shell molds or in fully three-dimensional molds. [Ngo, Mohan and Tamma (1998)]

Usually, (i) the geometry, (ii) the material parameters, (iii) the gate and vent positions and (iv) the pressure or the flow rate or a combination of the two at the gate and vent, are specified before the filling simulation is carried out. The simulation code is used to track the location of the flow fronts and estimate fill times. LIMS is capable of showing dry spot formation and tracking dry spots as filling progresses. The user interface uses a built-in scripting language, LBASIC, to assign gates and vents and also change them during flow with conditional statements, which allows for flexibility and enables automation. LIMS can also be used as a 'slave' using specialized Digital Link Libraries (DLLs), which provide an additional interfacing functionality. Such flow simulations have also been used to study the effects of different configurations of gates and vents on mold filling. The simulation, interfacing and automation capability of LIMS have been used to simulate resin flow in different mold geometries, investigate phenomena such as racetracking effects, to test control and sensing schemes for mold filling and for mold design optimization. [Mathur, Advani and Fink (2000, 1998); Maier, Rohaly, Advani and Fickie (1996); Simacek, Sozer and Advani (1998)]

3 Design Optimization Algorithms

3.1 Genetic Algorithms

Genetic Algorithms emerged from the research work on adaptive systems by computer scientists who sought to study the processes by which biological systems and organisms adapt to complex changes in their environment, and to use the knowledge gained to develop algorithms that could be useful for problem solving and for artificial intelligence. Genetic Algorithms were first analyzed and presented in detail by John Holland in his path breaking work, *"Adaptation in Natural and Artificial Systems"*. [Holland (1992)] Since then they have found wide applications in solving problems in fields as diverse as engineering, biology, mathematics, economics and financial markets. [Mitchell (1996); Goldberg (1997)]

The standard genetic algorithm involves partial swapping and copying of strings, which are representations of the optimization variables. The variables themselves can be continuous or discrete, since they are mapped to strings. Each string has a 'fitness' value, f, associated with it.

This algorithm employs three operators: Reproduction, Crossover and Mutation. The reproduction operator operates on the strings of each generation to produce the strings of the next generation. Pairs of strings are selected on the basis of fitness using proportional stochastic selection, also known as roulette wheel selection. These

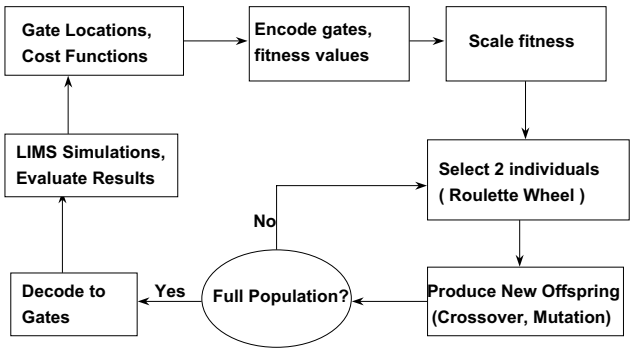


Figure 2 : Schematic of the use of the Simple Genetic Algorithm (SGA) for RTM process optimization

two strings are operated on, by the crossover and mutation operators, to produce two new strings, which belong to the next generation. New strings are produced until the population size, i.e. the number of strings in each generation, which is a fixed number, is attained. This new generation is evaluated and fitness values assigned to each member of the generation. [Figs 2 and 3]

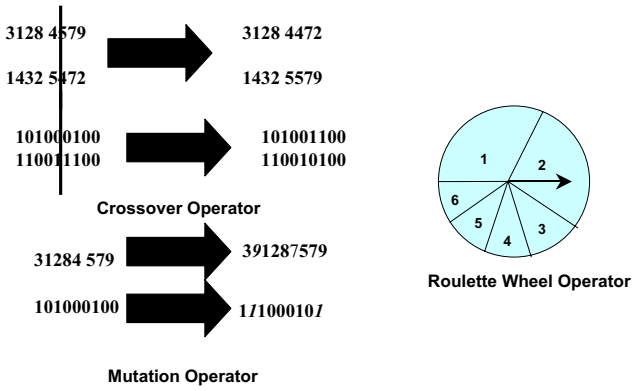


Figure 3 : Operators used by the genetic algorithm : reproduction (roulette wheel), crossover and mutation.

The reproduction and crossover operators work in tandem to use high performance strings having high fitness values and generate better strings having higher fitness values, thus emulating natural selection, which favors 'survival of the fittest'. The mutation operator works by taking the new strings produced and randomly changing the values of a few members of the strings. This ensures genetic diversity by producing strings, which contain new material and are not totally derived from the previous generation.

3.2 Design Sensitivity Gradient-based Algorithm for Fill Time Optimization

A design sensitivity based optimization algorithm was developed for the resin transfer molding process with the injection of resin at constant pressure. The governing equations for the design sensitivity fields were derived from equations (3-4), which describes the resin flow into the mold. The derived equations are directly parallel to equations (3-4) and can be solved in the same manner. The gradients of the cost function, i.e. fill time were derived in terms of the design sensitivity fields.

Consider a single gate location for a given mold, which has the coordinates (ε, η, ξ). The pressure fields at each time step and the fill time are going to be dependent on the location of this gate and the pressure imposed at it. The effect of the gate location on the pressure field is encapsulated in the design sensitivity fields, $\frac{\partial P}{\partial \varepsilon}$, $\frac{\partial P}{\partial \eta}$ and $\frac{\partial P}{\partial \xi}$. The governing equations governing these fields can be derived directly from the primary PDE equation and boundary conditions (Eqs.3-4) describing the resin flow in the mold by direct differentiation as follows:

$$\frac{\partial}{\partial \varepsilon}\left(\nabla\cdot(\frac{K}{\mu}.\nabla P)=0\right) \quad ;\frac{\partial}{\partial \eta}\left(\nabla\cdot(\frac{K}{\mu}.\nabla P)=0\right)$$
$$\textit{and} \quad \frac{\partial}{\partial \xi}\left(\nabla\cdot(\frac{K}{\mu}.\nabla P)=0\right)$$
$$\Rightarrow \nabla\cdot(\frac{K}{\mu}.\nabla\frac{\partial P}{\partial \varepsilon})=0 \quad ;\nabla\cdot(\frac{K}{\mu}.\nabla\frac{\partial P}{\partial \eta})=0$$
$$\textit{and} \quad \nabla\cdot(\frac{K}{\mu}.\nabla\frac{\partial P}{\partial \xi})=0 \qquad (5)$$

Thus we obtain auxiliary PDEs for the design sensitivity fields. The boundary conditions for these PDEs are similarly derived from the original boundary conditions and are stated as follows:

$$\frac{\partial P}{\partial \varepsilon}(\vec{x}_{gate}) = -\frac{\partial P}{\partial x}$$
$$\frac{\partial P}{\partial \varepsilon}(\vec{x}_{flowfront}) = 0 \qquad \frac{\partial P}{\partial \varepsilon}(\vec{x}_{vent}) = 0$$

$$\vec{n} \bullet (\frac{K}{\mu} \bullet \nabla \frac{\partial P}{\partial \varepsilon}) = 0 \quad \text{at the mold wall}$$
$$\frac{\partial P}{\partial \eta}(\vec{x}_{gate}) = -\frac{\partial P}{\partial y}$$
$$\frac{\partial P}{\partial \eta}(\vec{x}_{flowfront}) = 0 \qquad \frac{\partial P}{\partial \eta}(\vec{x}_{vent}) = 0$$
$$\vec{n} \bullet (\frac{K}{\mu} \bullet \nabla \frac{\partial P}{\partial \eta}) = 0 \quad \text{at the mold wall}$$

$$\frac{\partial P}{\partial \xi}(\vec{x}_{gate}) = -\frac{\partial P}{\partial z}$$
$$\frac{\partial P}{\partial \xi}(\vec{x}_{flowfront}) = 0 \qquad \frac{\partial P}{\partial \xi}(\vec{x}_{vent}) = 0$$
$$\vec{n} \bullet (\frac{K}{\mu} \bullet \nabla \frac{\partial P}{\partial \xi}) = 0 \quad \text{at the mold wall} \tag{6}$$

These auxiliary equations are completely parallel to the primary governing equations (eq.3-4) for the pressure field. In the calculation of the pressure field at any time step, the domain is discretized, the stiffness matrix **[K]** has been constituted and inverted/decomposed and the pressure field is calculated using the matrix equation **[K]**{**P**}={$\boldsymbol{f}$}. The design sensitivity fields can then be easily computed by determining the pressure gradients at the gate and using them to change the appropriate terms in the forcing vector {$\boldsymbol{f}$} and then multiplying {$\boldsymbol{f}$} with the inverted or decomposed stiffness matrix.

In gate location optimization, one measure of the performance of the gate(s) is the time it takes to fill the mold. The design sensitivity fields, once determined at each time step can be used to find the gradient of fill time with respect to the gate locations. Supposing the volume of the mold is V, then for a single gate, it can be stated that:

$$V = \int_0^{t_f} Q_{gate}\, dt \tag{7}$$

Since the volume of the mold is a constant, it follows that:

$$\frac{\partial V}{\partial \varepsilon} = \frac{\partial}{\partial \varepsilon} \int_0^{t_f} Q_{gate}\, dt = 0$$
$$\frac{\partial V}{\partial \eta} = \frac{\partial}{\partial \eta} \int_0^{t_f} Q_{gate}\, dt = 0$$
$$\frac{\partial V}{\partial \xi} = \frac{\partial}{\partial \xi} \int_0^{t_f} Q_{gate}\, dt = 0 \tag{8}$$

Differentiating the integrals in equation (8) and separating out the terms we get:

$$\int_0^{t_f} \frac{\partial Q_{gate}}{\partial \varepsilon}\, dt + \frac{\partial t_f}{\partial \varepsilon} \quad Q_{gate}(t_f) = 0$$
$$\int_0^{t_f} \frac{\partial Q_{gate}}{\partial \eta}\, dt + \frac{\partial t_f}{\partial \eta} \quad Q_{gate}(t_f) = 0$$
$$\int_0^{t_f} \frac{\partial Q_{gate}}{\partial \xi}\, dt + \frac{\partial t_f}{\partial \xi} \quad Q_{gate}(t_f) = 0 \tag{9}$$

$$\Rightarrow \quad \frac{\partial t_f}{\partial \varepsilon} = -\frac{\int_0^{t_f} \frac{\partial Q_{gate}}{\partial \varepsilon}\, dt}{Q_{gate}(t_f)}, \quad \frac{\partial t_f}{\partial \eta} = -\frac{\int_0^{t_f} \frac{\partial Q_{gate}}{\partial \eta}\, dt}{Q_{gate}(t_f)}$$
$$\frac{\partial t_f}{\partial \xi} = -\frac{\int_0^{t_f} \frac{\partial Q_{gate}}{\partial \xi}\, dt}{Q_{gate}(t_f)} \tag{10}$$

Now the flow rate at any gate is usually calculated by drawing a small control volume around the gate and then the outflow of resin through that control volume is evaluated. This is given by the following expression:

$$Q_{gate} = \int_{C.V.} \vec{V} \bullet \hat{n}\, dS = \; Q_{gate} = \int_{C.V.} -\frac{\overline{\overline{K}}}{\mu} \bullet \nabla P \bullet \hat{n}\, dS \tag{11}$$

Differentiating this equation with respect to gate coordinates and noting that the control volume is arbitrary we get:

$$\frac{\partial Q_{gate}}{\partial \varepsilon} = \frac{\partial}{\partial \varepsilon} \int_{C.V.} -\frac{\overline{\overline{K}}}{\mu} \bullet \nabla P \bullet \hat{n}\, dS$$
$$= \int_{C.V.} -\frac{\overline{\overline{K}}}{\mu} \bullet \nabla (\frac{\partial P}{\partial \varepsilon}) \bullet \hat{n}\, dS = Q^{\varepsilon}_{gate}$$
$$\frac{\partial Q_{gate}}{\partial \eta} = \int_{C.V.} -\frac{\overline{\overline{K}}}{\mu} \bullet \nabla (\frac{\partial P}{\partial \eta}) \bullet \hat{n}\, dS = Q^{\eta}_{gate}$$
$$\frac{\partial Q_{gate}}{\partial \xi} = \int_{C.V.} -\frac{\overline{\overline{K}}}{\mu} \bullet \nabla (\frac{\partial P}{\partial \xi}) \bullet \hat{n}\, dS = Q^{\xi}_{gate} \quad (12)$$

Thus the gradients of the flow rates can be determined by computing the pseudo-flow rates in the auxiliary flow fields, Q^{ε}_{gate}, $Q^{\eta}_{gate}$and Q^{ξ}_{gate}. Upon substituting into equation (10), we obtain:

$$\Rightarrow \quad \frac{\partial t_f}{\partial \varepsilon} = -\frac{\int_0^{t_f} Q^{\varepsilon}_{gate}\, dt}{Q_{gate}(t_f)}$$
$$\frac{\partial t_f}{\partial \eta} = -\frac{\int_0^{t_f} Q^{\eta}_{gate}\, dt}{Q_{gate}(t_f)}$$
$$\frac{\partial t_f}{\partial \xi} = -\frac{\int_0^{t_f} Q^{\xi}_{gate}\, dt}{Q_{gate}(t_f)} \quad (13)$$

Thus the gradient of the fill time with respect to the gate coordinates are found by calculating the flow rates at the gate in the solution for the pressure field and the pseudo-flow rates at the gate in the parallel solution for the design sensitivity-fields and then integrating these flow rates through the history of the mold filling process.

4 Implementation of Automated Design Optimization

4.1 Real-Coded GA

The gate locations are coded as real number strings using the following formula:

$$Gate\ String = int\left(10^4 \frac{Gate\ Node\ Number}{Total\ \ Number\ of\ Nodes}\right) \quad (14)$$

An initial population of strings is randomly generated, decoded into gates and then evaluated using LIMS simulations. These results are then fed into the GA as cost functions, which encapsulate the desired results, such as minimum fill time. The cost function values are then used to determine the fitness values for each member of the population. A ranking fitness scheme was used to assign fitness to each member of the population, where the best member has a fitness of 1 and the worst member has a fitness of N, where N is the size of the population. Then the three reproduction operators are used to generate the next generation of designs, which are evaluated then in turn. This optimization loop is repeated until a specified number of generations or there are no further improvements in the population. The GA implementation for RTM Optimization is shown in Fig 4.

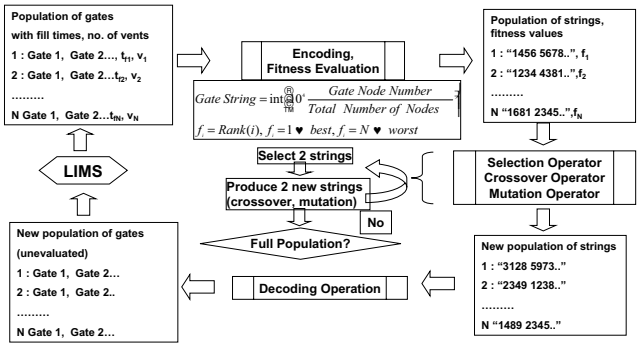


Figure 4 : LIMS-based Implementation of Real-coded Genetic Algorithms for RTM Optimization

5 Gradient-based Optimization Algorithm

The design sensitivity algorithm was implemented in LIMS using the built-in scripting language, LBASIC. At each step in the flow simulation the pressure field is determined and the flow front advanced until one node is filled, which is then included in the flow domain for the next step. This is encapsulated in the ***solve*** function in LIMS. Following the application of the ***solve*** function, the pressure field became available and was used to calculate the pressure gradients at every gate. These pressure gradients were then used in the boundary conditions for the design sensitivity fields. The stiffness matrix **[K],** decomposed during the pressure calculation, is accessible only to the simulation kernel. In order to access this

matrix and use it to perform the design sensitivity calculations, a special function, called ***updatepq*** was written into LIMS. This function was used to calculate the design sensitivity fields (eq. 6) and then the pseudo-gate flow rates. (eq. 12) These steps were repeated until the mold filling simulation was completed. The pseudo-gate flow rates thus calculated at each step of the filling simulation were integrated over the filling history using the trapezoidal integration. After the completion of the filling simulation, equation 13 was used to calculate the gradients of the mold fill time with respect to mold coordinates.

These gradients were then fed into a standard binary line search algorithm that searches in the direction indicated by the gradient. [Vanderplaats (1984)] When the line search algorithm yielded a set of gate locations, the gradient calculation and search was repeated. This iterative calculation was performed until convergence, i.e. mold fill time could not be decreased further. A schematic of the design sensitivity gradient-based algorithm is shown in Fig 5.

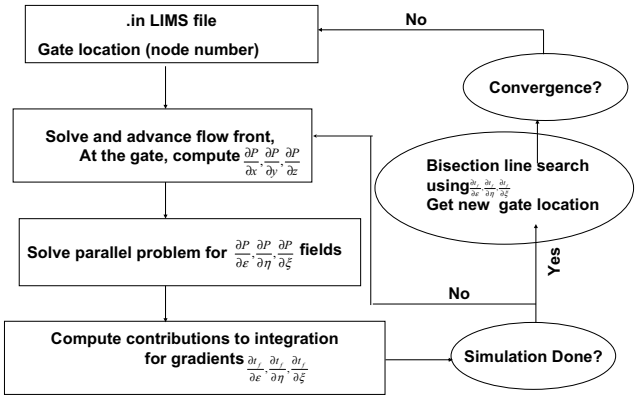


Figure 5 : Schematic of Gradient-based Optimization Algorithm for Minimum Mold Fill Time in the RTM Process

6 Hybrid Genetic Algorithm

These were motivated by the fact that GAs do not directly use the process physics but do not get trapped in a local minima and gradient based methods can get trapped in a local minima but do employ the process physics directly. Hence we introduce two methods that combine the strength of both these methods. We constructed two hybrid optimization algorithms that use the real coded GA and the sensitivity-based gradient algorithm. The difference is the manner they are coupled. In the first algorithm, the GA is used as a global optimizer to search out the "survival of the fittest" parts of the search space. Then the final generation of designs is taken and the gradient algorithm is used for further improvement as a local optimizer. This optimization algorithm can be called a serial hybrid optimization algorithm, since the GA and the gradient are used in series.

Alternatively, the gradient-based algorithm can be used in an interactive fashion to improve each generation of designs generated by the GA thus enriching the pool of good designs for the GA to consider at the next step. This kind of interactive optimization algorithm is called a Memetic Algorithm and its analog in the natural world has been known to arrive at optimal solutions due to the Baldwin Effect. [Dawkins (1995)] The Baldwin Effect occurs due to the result of learning in a population of organisms within each generation, which are simultaneously evolving to adapt to their environment. In such a population, the organisms that are better at learning are fitter and leave more offspring in the next generation. This influences the next generation and speeds up the evolution of well-adapted (and highly intelligent) organisms. Similarly, if the gradient optimizer acts to improve each generation of designs, then the GA should be able to consistently arrive at good designs. (Fig 6) In this paper, both types of hybrid optimization algorithms have been evaluated and their results compared for the selected case studies.

7 Implementation and Selection of Parameters

The optimization algorithms are the pure-GA, sensitivity-based gradient algorithm, the serial hybrid GA and the memetic GA. The cost functions are (i) the mold fill time, (ii) the magnitude of the gradients of the fill time, (iii) a linear combination of the gradient magnitude and the fill time and (iv)the number of vents. Optimization software was written which is capable of taking any mold configuration and using it to run the different configurations of optimization algorithms and cost functions. This software consists of a portion written in C++ with a set of complementary LBASIC scripts; LIMS DLLs that allow it to seamlessly interface with LIMS; and the LIMS executable itself in slave form. The implementation of the optimization algorithms using

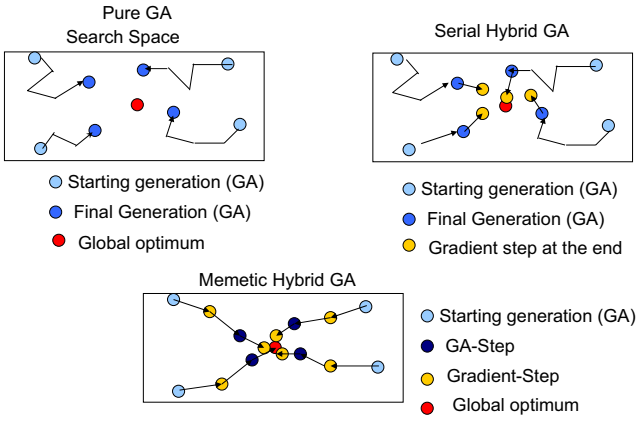


Figure 6 : Search capability of the pure genetic algorithm, the serial hybrid algorithm and the memetic hybrid algorithm

LIMS DLL is shown in Fig 7.

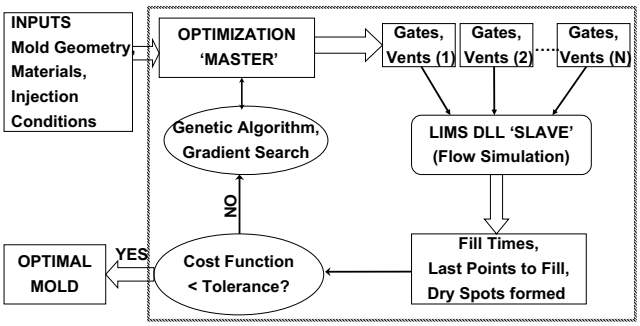


Figure 7 : Implementation of the RTM optimization using LIMS and Digital Linked Libraries

The C++ part of the program consists of an information database, an optimization master, functions implementing the real-GA, the gradient algorithm and connection functions for the LIMS slave. The set of LBASIC scripts contains a list of global variables and functions for initializing these variables, evaluating populations and conducting gradient searches.

When the program is invoked, the optimization master reads the parameters and the other information for the optimization from a file and starts the LIMS slave and establishes contact using the LIMS DLLs and connection functions. This information includes the optimization algorithm, cost function, number of gates, gate pressure, parameters for the GA, the type and name of the file containing the mold, connectivity information as well as the output file names. The database and the global variables needed for LIMS are initialized using this information. Then the optimization is performed using commands from the optimization master which are transmitted to LIMS slave to run the LBASIC scripts through the data pipeline between them. Data files are used to transmit the results back to the master, which reads them, calculates statistical information and writes to the output files. Three output files are written: the first contains the raw data; the second contains statistical information for the optimization while the third is a graphical output file for TECPLOT, a scientific graphing package for FEM analysis.

This software was used in six case studies to evaluate the hybrid optimization algorithms for three mold geometries to either find one optimal gate location or two optimal gates. In each case study, an exhaustive search was conducted for the one gate case and the global optimum was determined. The performance of the different algorithms for the one gate case was benchmarked against the global optimum from the exhaustive search. The case studies are described in the next section.

8 Case Studies

Three composite parts were chosen for optimization case studies with the gradient-based optimization algorithm. The three parts are – a two-dimensional rectangular geometry with thick sections, a flat plate with H-shaped ribs and the five-sided box with a central partition (RISC reduction box). (Fig 8). These geometries had sufficient complexities and have been experimentally investigated at the Center for Composite Materials, University of Delaware..

The hybrid genetic algorithms were investigated in all the case studies and their performance benchmarked against the best solution or the globally optimal solution. The studies were conducted for the case of single gate resin injection and double gate resin injection at a constant pressure of 10^5 Pa each. The genetic algorithm, the serial genetic algorithm and the Memetic algorithm were used to determine optimal gate location(s) with the objective of minimizing the mold fill time. An exhaustive study was conducted for the single gate case using LIMS simulations and LBASIC scripts. The mold fill time and number of last points to fill (potential vent locations) were recorded for constant pressure single injection at every

possible node location in the mesh. The results from this global search space were used to benchmark the performance of the hybrid genetic algorithms for the single gate case.

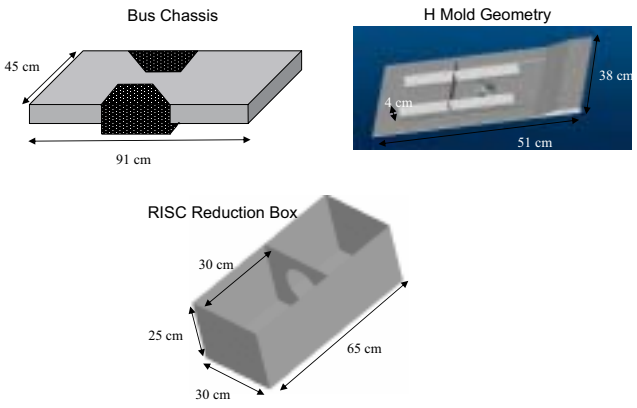


Figure 8 : Three composite parts chosen for the case study with the gradient-based optimization algorithm. The bulk permeability is of the order of 10^{-9} m^2 in all cases, and the permeability of the thick section in the bus chassis is 10^{-12} m^2.

The parameters for the genetic algorithm, such as population size and the crossover and mutation rates, were fixed for each geometry investigated. An initial population was generated at random and used as a common starting point for each of the algorithms tested here: the pure GA, the serial hybrid GA and the Memetic algorithm. In addition, three cost functions were used for the simplest GA optimization: (i) the fill time, (ii) the gradient of the fill time and (iii) a linear combination of the fill time and the gradient of the fill time. The gradient of the fill time was chosen because when a function is at its extremis (maximum or minimum), the gradient of that function is zero at that point. Hence, searching for the minimum fill time is equivalent to searching for the zero gradients. The gradient and the combination of fill time and gradient were thus used as tests of the capability of the GA to arrive at good designs using different cost functions. Two statistical parameters used for the benchmarking, which measured the differences of the final population of designs from the global optimum, are given below:

$$\frac{\sum_{1}^{N} \boldsymbol{t}_f - \boldsymbol{t}_{f,global}}{N} = \boldsymbol{t}_{f,average} - \boldsymbol{t}_{f,global}$$

$$\sqrt{\frac{\sum_{1}^{N} (\boldsymbol{t}_f - \boldsymbol{t}_{f,global})^2}{N}} = \sqrt{\boldsymbol{\sigma}^2 + (\boldsymbol{t}_{f,average} - \boldsymbol{t}_{f,global})^2}$$

$$= \boldsymbol{\sigma}^* \tag{15}$$

These statistical measures were used to determine the effectiveness of the hybrid genetic algorithms and the practicality of incorporating different cost functions into genetic algorithms, for the single gate case.

An exhaustive study could not be performed for the two gates case, as the permutation of gates required were prohibitively large. For example, if one would have a mesh of 5000 nodes with 5000 possible gate locations, 5000 x 4999 simulations will be required to find the best possible two-gate combination in an exhaustive search. In this case, the best solution from the optimization runs themselves was chosen, instead of the global solution, and incorporated into the statistical measures. Thus the effectiveness of the hybrid genetic algorithms is measured in terms of the quality of the final generation from the best (known) solution for each case. The exhaustive search and the benchmarking studies for both the one gate and the 2 gates were carried out for each of the three geometries and are described in the next section.

9 Results

9.1 Exhaustive search for the one gate case

The mold fill time and the locations of the last points to fill for constant pressure injection at every node in the finite element meshes for the three selected geometries were recorded after executing a LIMS simulation. The mold fill time could now be plotted as contours of a field on the surface of the mesh using TECPLOT thus giving a graphical representation of the global search space for one gate injection. The locations of the last points to fill, which are potential vent locations can be plotted in TECPLOT. In addition, a Pareto chart can be plotted, which has mold fill time for all nodes on the Y axis and the associated number of last points to fill on the X axis. This can yield insight into the potential trade-offs between the mold fill time and the number of last points to fill. The

contours of mold fill time, the last points to fill and the Pareto chart are plotted for each of the three selected geometries in Figs 9-11.

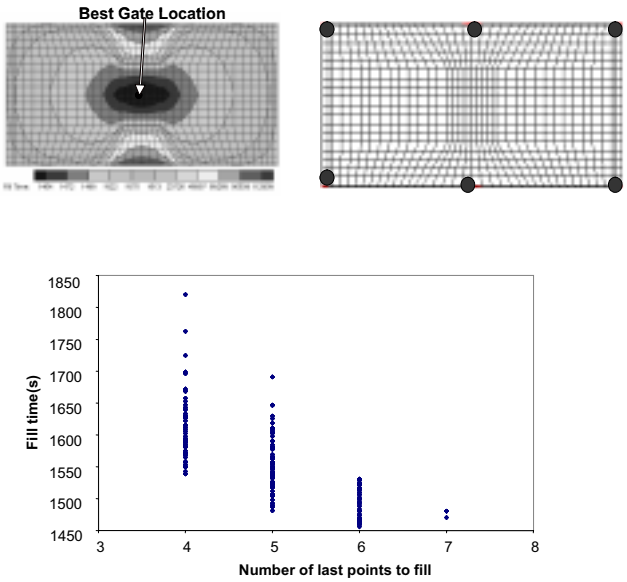


Figure 9 : Map of the global search space for the bus mold geometry and one gate case from exhaustive simulations: Contour plot of the mold fill time and the last points to fill, and Pareto chart

9.1.1 Discussion of exhaustive search

It can be observed from Fig 9 (rectangular flat plate with thick sections) that for the search space of mold fill times, the global minimum lies at the center of the mold where the planes of symmetry of the part intersect. This is because the center of the mold is the shortest distance from the resin flow fronts during mold injection at all times during the mold filling process, thus minimizing the resistance to flow. This resistance is generated when the resin being injected at the gate has to push through the saturated preform and then advance the flow front. Since the injection is performed under a constant finite pressure, the resistance to flow will increase with the progression of the mold filling process, the flow rate of resin through the gate will decrease over time, taking more and more time to fill the extra volume of the mold. When the gate is located far away from the flow fronts for most of the filling history, the fill time increases nonlinearly. But when the gate is closer to the flow fronts, such as in this case, then the mold fill time will be at a minimum. In this case, the worst gate locations lie at the center of the dense thick sections, where the permeability is low and the inherent resistance to flow is high.

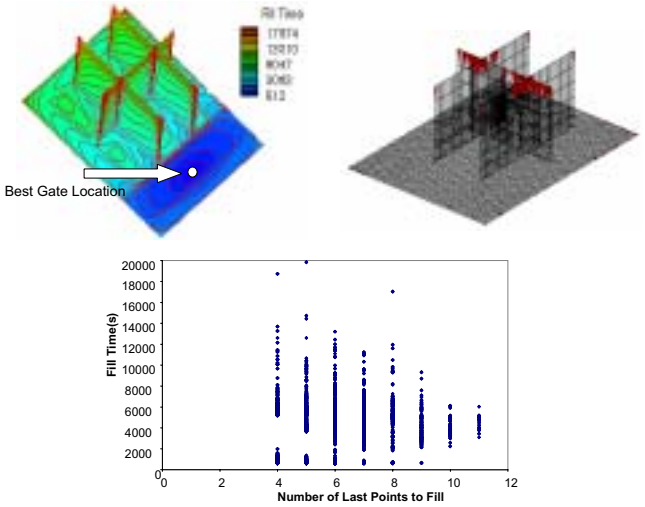


Figure 10 : Map of the global search space for the H mold geometry and one gate case from exhaustive simulations: Contour plot of the mold fill time, the last points to fill, and Pareto chart

The Pareto chart shows that the corresponding number of last fill areas (potential vent locations) goes from a minimum of four to a maximum of seven, with the global optimum requiring six vent locations, which are shown. This is because this part has four sharp corners at the four edges of the rectangular boundary as well as two thick sections in which air could get entrapped. By injecting resin from one of the edges, it is possible to eliminate the last regions to fill on that edge, but the other two corners and the thick sections will contain region that will fill last and create “dry spots”. When the gate lies in the central regions, all six areas will be potential vent locations.

In the case of the H mold geometry, the gate location with lower fill time is not at the center of the mold but lies at one end. (Fig 10) This is because there is an area of higher thickness and same density/permeability as the rest of the part with a “step” in the part. Since the flow through the thickness is proportional to the cube of the thickness (Hele-Shaw approximation), the flow rates at the gates located in this area will be higher. In this case, air will be trapped not only at the edges of the base of the part, but also in the “H” shaped ribs. Hence the number of last areas to fill is quite high, ranging from 4 to 11.

In the case of the RISC box mold geometry, the only edges where air can be trapped are the top (open end of the box) five edges of the part. (Fig 11) Since sections of the box are interconnected, air will not get entrapped on all edges for injection at given gate location. Hence the number of last regions to fill is a maximum of three only for single gate injection. The best gate location lies at the center of the part, where the flow resistance for mold filling is the lowest. (Fig 11)

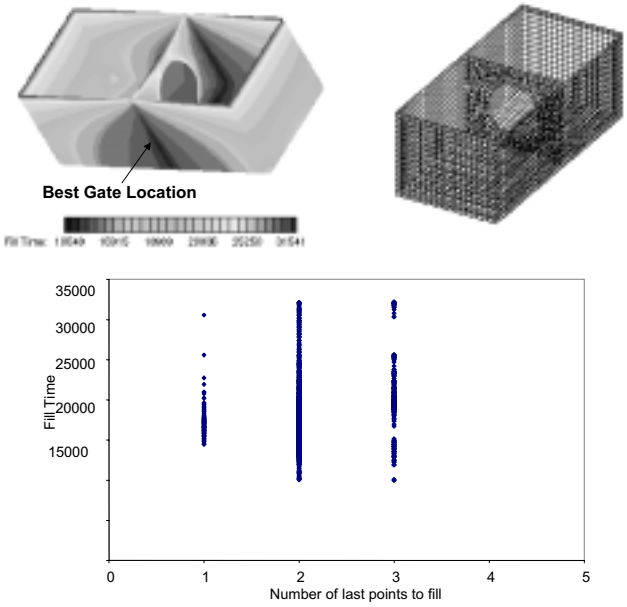


Figure 11 : Map of the global search space for the RISC box mold geometry and one gate case from exhaustive simulations: Contour plot of the mold fill time and the last points to fill, and Pareto chart from the simulations

9.2 Benchmarking study with optimization algorithms

The benchmarking study for each mold geometry, for both one and two gate cases, was conducted in two parts. In the first part of the study, the real coded GA was used for optimal gate location for mold fill time minimization for three cost functions. The pure genetic algorithm was run with cost functions – fill time, fill time gradient and combination of fill time and fill time gradient. In the second part, the serial hybrid GA and the Memetic algorithm were used for determining the optimal gate location. The cost function for the Memetic and the serial hybrid GA is the mold fill time. An initial starting population was generated randomly and then used as the first set of gates for each optimization algorithm. The population size for the one gate case was 20 gate locations and that for the two-gate case was 40 sets of gate locations. Each optimization algorithm was run until no further improvements could be obtained on the average fill time per generation.

The statistical measures introduced earlier were used to measure distance of the final generation from the global optima. The number of simulations required by each algorithm to come to the final generation was measured. Bar graphs were used to compare the performance of each algorithm. Mold filling simulations were performed for the best gate locations generated by the algorithms and the resin flow contours and last regions to fill were plotted.

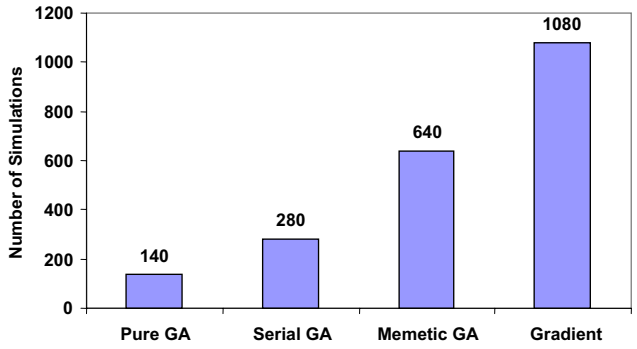


Figure 12 : Computational cost of optimization in terms of the number of simulations (one gate case)

For the one gate case, Fig 12 shows the computational cost in terms of numbers of LIMS mold-filling simulations required to reach optimal solutions. Bar graphs of statistical performance measures of the three hybrid optimization algorithms, with reference to the global optimum from exhaustive search are shown in Fig 13. The results from mold filling simulations for the best performing gate locations located by the optimization algorithms for the three complex mold geometries are shown in Fig 14, and the best algorithm for each geometry indicated.

For the two gate case, the global optima for the three mold geometries could not be located due to the prohibitive cost of executing the exhaustive search. Hence the best possible gate locations located from the optimization search were used in lieu of global optima. In this case, the computational cost is given in Fig 15, statis-

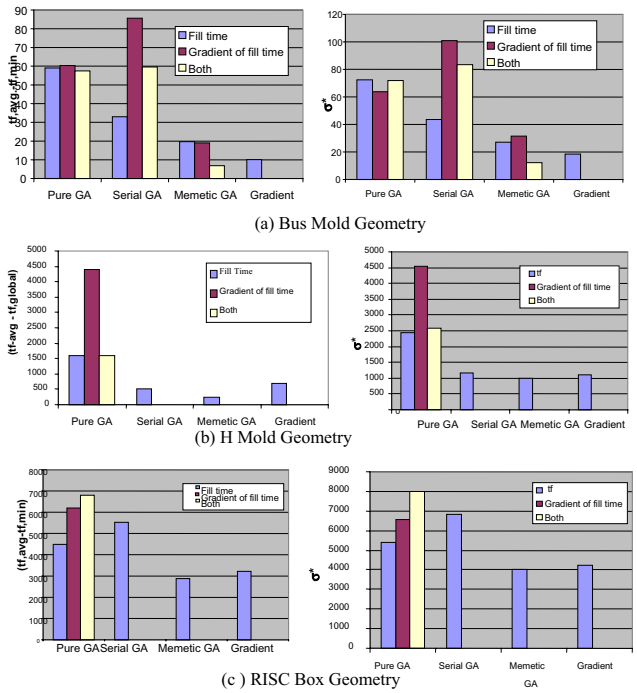
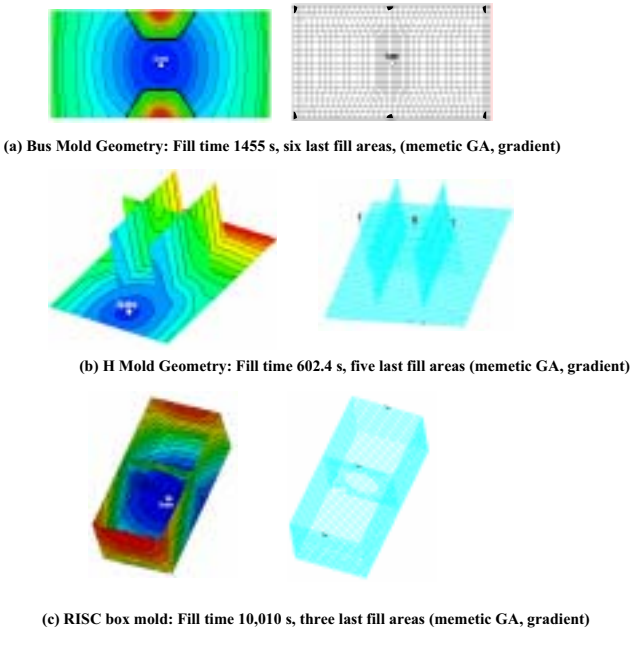


Figure 13 : Benchmarking results for the one gate case for three complex mold geometries: Charts of the statistical measures of performance of the hybrid genetic algorithms against the best-known solution



(a) Bus Mold Geometry: Fill time 1455 s, six last fill areas, (memetic GA, gradient)

(b) H Mold Geometry: Fill time 602.4 s, five last fill areas (memetic GA, gradient)

(c) RISC box mold: Fill time 10,010 s, three last fill areas (memetic GA, gradient)

Figure 14 : Flow contours and last areas to fill for optimal gate locations for the single gate case.

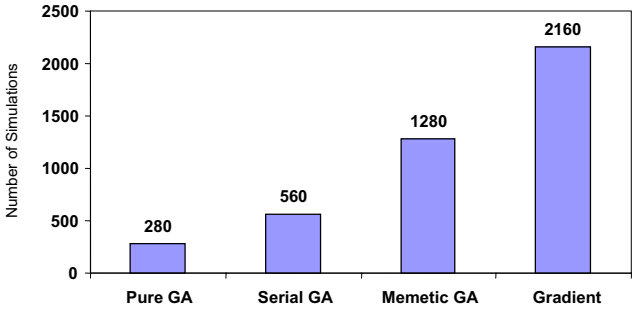


Figure 15 : Computational cost of optimization in terms of the number of simulations (two gate case)

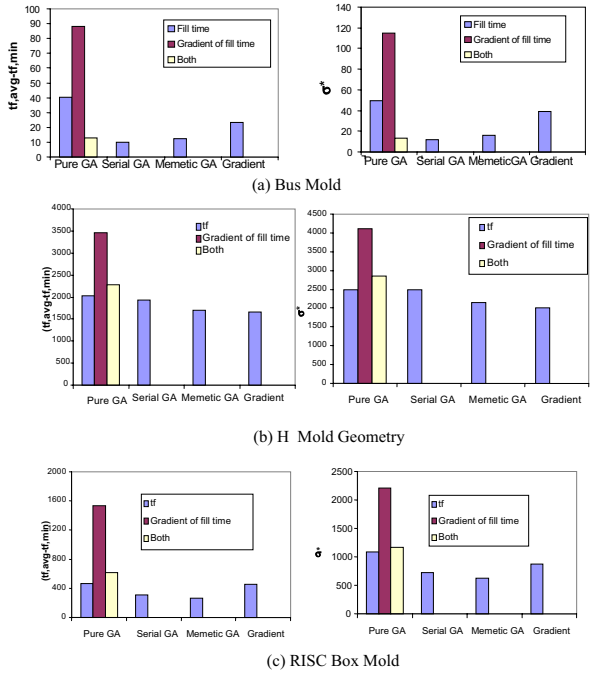
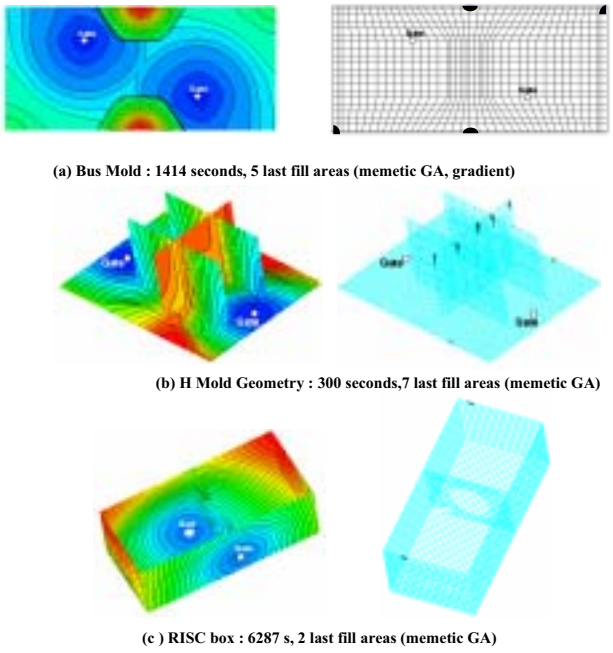


Figure 16 : Benchmarking results for the two gate case : Charts of the statistical measures of performance of the hybrid genetic algorithms against the best-known solution

tical measures are plotted in Fig 16 and the flow contours for best gate locations are plotted in Fig 17. Again, the best performing optimization algorithms for each one of the mold geometries are indicated.

10 Discussion and Conclusions

It can be observed that the Memetic genetic algorithm was able to outperform the pure GA and the serial hy-



(a) Bus Mold : 1414 seconds, 5 last fill areas (memetic GA, gradient)

(b) H Mold Geometry : 300 seconds,7 last fill areas (memetic GA)

(c) RISC box : 6287 s, 2 last fill areas (memetic GA)

Figure 17 : Flow contours and last areas to fill for optimal gate locations for the two gate case for the three complex mold geometries

brid GA for all cases. The serial hybrid GA is able to improve on the performance of the pure GA but cannot match the Memetic GA, which was closely matched only by the gradient-based algorithm, in terms of the statistical metrics and gate locations. However, the computational cost of the gradient-based optimization algorithm is much higher than that of the GA-based optimization algorithm. This is due to the fact that the GA utilizes information from an entire population, while the gradient-based optimizer is a single point optimizer that cannot exploit the diverse information inherent in the population.

The gradient of the fill time could serve as a cost function for the bus mold geometry but it performed poorly as an optimization cost function for the other two-mold geometries. The gate locations obtained at the end of the optimization were farther from the global optimum for most cases, and the algorithms performed consistently for only the bus mold geometry. This may be due to the non-symmetric nature of the H mold and RISC box mold geometries. This is in contrast to the bus mold geometry, has a smooth and symmetric search space with a single global optimum at the center of the part. The search space of gradients is much more complex than that of the primary cost function, i.e., the fill time, whereas in the case of the bus mold geometry, both the space of the fill time and the gradients can be visualized geometrically as a bowl with a steep slope and single minimum. This space would be much more complicated for the other two geometries. In addition to geometric asymmetry, other factors that may affect the utilization of cost functions such as gradient of fill time and mold fill time are material complexity, i.e. variations in thickness and material properties, and the presence of multiple corners, ribs and double curvatures.

For the one gate case, the Memetic GA was able to find the globally optimal solutions in fewer simulations than the gradient-based optimizer. It can be seen that the Memetic GA and the gradient algorithm were able to find the globally optimal solutions, perhaps due to a simplicity of the search space. For the two gate case studies, it was able to find better solutions than the gradient-based optimization algorithm for all three geometries. In all cases, the Memetic GA was able to outperform the simple genetic algorithm in terms of the quality of solutions and the statistical measures of performance. The next best performance was by the gradient-based algorithm for the one-gate case, and the serial GA for the two gate case. This is due to the fact that the search space is relatively simple for the one-gate case, which makes it tractable for the gradient-based algorithm. In the two-gate case, the search space is considerably more complex which makes it difficult for the gradient-based algorithm to perform without good starting points. The serial GA, in which the results from a pure GA are improved upon by the gradient-based algorithm, provides high quality initial starting points to the gradient-based algorithm to perform local search.

Though the Memetic GA required a larger number of simulations, the number of simulations was much less than that required by the exhaustive search and the gradient-based optimization algorithm. The inherent trade-off is definitely worthwhile in consistently good performance and high quality of solutions obtained. Hence, it can be concluded that the Memetic GA is a better optimization algorithm for gate location with the objective of mold fill time minimization for complex mold geometries.

This research work has demonstrated that though general-purpose global optimization algorithms such as

GAs can be used to solve design optimization problems, there is no guarantee of good performance. Problem specific algorithms developed using the process physics do guarantee improvement but they perform best as local optimizers. The development of problem specific global optimization algorithms that can outperform GAs at a lower computational cost may be a possibility. Such a global optimization algorithm would take advantage of the features of the problem domain to determine the global optimum in a smaller number of computations than the GA. This is especially important as the complexity of the composite parts increases, since the search space for optimization will show a corresponding growth. The addition of design for control and sensor locations to the optimization problem will also increase the complexity of the search space. Hence, robust and reliable optimization algorithms are necessary to address needs such as use of simulations of manufacturing flows for design and control of the process.

Acknowledgement: The US Army Research Laboratory under the Collaborative Materials Research (CMR) Program provided funding for this research.

References

Antonelli, D.; Farina, A. (1999): Resin Transfer Moulding: Mathematical Modelling And Numerical Simulations, *Composites - Part A: Applied Science and Manufacturing* v30 n12.

Bruschke, M. V.; Advani, S. G. (1990): A Finite Element/Control Volume Approach to Mold Filling in Anisotropic Porous Media, *Polymer. Composites*, 11, pp. 398-405

Bruschke, M. V.; Advani, S. G. (1991): RTM: Filling Simulation of Complex Three-Dimensional Shell-Like Structures, *SAMPE Quarterly*, 23(1) pp. 2-1.

Bruschke, M. V.; Advani, S. G. (1994): A Numerical Approach to Model Non-isothermal, Viscous flow with Free Surfaces through Fibrous Media, *International Journal of Numerical Methods in Fluids*, vol. 19, pp.575-603.

Dawkins, R. (1995): *River Out of Eden: A Darwinian View of Life*, Harper Collins.

Fong L.; Advani S.G. (1998): in S.T.Peters, (editor) *Lubin's Handbook of Composites II*.

Gallez, X. E.; Advani, S. G. Resin Infusion Process Simulation (RIPS) (1997): A Fast Method For Three Dimensional Geometries Materials *Proceedings of the 55th Annual Technical Conference, ANTEC 1997,* Part 2 (of 3) pp 2454-2458.

Gauvin, R.; Trochu, F. (1998): Key Issues In Numerical Simulation For Liquid Composite Molding Processes, *Polymer Composites*, v 19 no. 3.

Goldberg; D.E. (1997): *Genetic Algorithms in Search, Optimization and Machine Learning*,Addison-Wesley.

Holland, J. (1992): *Adaptation in Natural and Artificial Systems : An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*, MIT Press.

Kim, J-S; Kim, C-G; Hong, C-S (1999): Optimum Design Of Composite Structures With Ply Drop Using Genetic Algorithm And Expert System Shell, *Composite Structures*, v. 46 no. 2.

Lin, M.; Hahn, H. T.; Huh, H. (1998): Finite Element Simulation Of Resin Transfer Molding Based On Partial Nodal Saturation And Implicit Time Integration, *Composites - Part A: Applied Science and Manufacturing* v 29, no. 5-6.

Mahesh, K.; Kishore, N.N.; Deb, K. (1996): Optimal Design Of Composite Turbine Blade Using Genetic Algorithms, *Advanced Composite Materials: The Official Journal of the Japan Society of Composite Materials*, v5 n2.

Maier, R. S.; Rohaly, T. F.; Advani, S. G.; Fickie, K. D. (1996): A Fast Numerical Method for Isothermal Resin Transfer Mold Filling, *International Journal of Numerical Methods in Engineering*, 39 p. 1405-1422.

Mathur, R.; Advani, S. G.; Fink, B. K. (1999):Use of Genetic Algorithms to Optimize Gate and Vent Locations for the Resin Transfer Molding Process, *Polymer Composites*, 20, pp. 167-178.

Mathur, R.; Advani, S. G.; Fink, B. K. (2000): A Sensitivity-based Gate Location Algorithm for Optimal Mold Filling During the Resin Transfer Molding Process", *Advances in Computational Engineering and Sciences*, Tech Science Press, Volume 1, pp 138-144.

Mitchell, M.E. (1996): *An Introduction to Genetic Algorithms*, MIT Press.

Mohan, R.V.; Ngo, N.D.; Tamma, K.K. (1999): On A Pure Finite-Element-Based Methodology For Resin Transfer Mold Filling Simulations", *Polymer Engineering and Science* v 39 no. 1.

Nedanov, P.; Advani, S. G. (2000): Mold Filling Simulation of Sandwich Composite Structures manufactured by Liquid Molding: A Parametric Study, *J. of Sandwich Structures and Materials*, 2, pp. 117-130

Ngo, N.D.; Mohan, R.V.; Chung, P.W.; Tamma, K.K. (1998): Recent Developments Encompassing Non-Isothermal/Isothermal Liquid Composite Molding Process Modeling/Analysis: Physically Accurate, Computationally Effective, And Affordable Simulations And Validations, *Journal of Thermoplastic Composite Materials* v 11 n 6.

Ngo, N. D.; Tamma, K.K. (2000): Non-Isothermal Three-Dimensional Developments in Process Modeling of Composites: Flow/Thermal/Cure Formulations and Experimental Validations", *CMES: Computer Modeling in Engineering and Sciences*, Vol. 1, No. 3, pp 57-72.

Papalambros, P.Y.;Wilde, D.E. (1988): *Principles of Optimal Design: Modeling and Computation*, Cambridge University Press

Sadagopan, D.; Pitchumani, R. (1998):Application Of Genetic Algorithms To Optimal Tailoring Of Composite Materials, *Composites Science and Technology* v 58 n 3-4.

Savic, D. A.; Evans, K. E.; Silberhorn, T. (1999): Genetic Algorithm-Based System For The Optimal Design Of Laminates, *Computer-Aided Civil and Infrastructure Engineering*, v 14 n 3.

Shimojima, K. (1999): Optimization Method Of FGM Compositional Distribution Profile Design By Genetic Algorithm, *Proceedings of the 1998 5th International Symposium on Functionally Graded Materials*, v 308-311.

Simacek, P.; Sozer, E.M.; Advani, S.G. (1998): Numerical Simulations Of Mold Filling For Design And Control Of RTM Process, *Proceedings of the 1998 56th Annual Technical Conference, ANTEC98,* Part 2 (of 3) Apr 26-30 1998 v 2.

Simacek, P.; Advani S.G. (2003): Desirable Features in Mold Filling for Liquid Composite Molding Processes, *Polymer Composites*, (to appear).

Smith, D.E.; Tortorelli, D.E.; Tucker III, C.L. (1998): Analysis and Sensitivity Analysis for Polymer Injection and Compression Molding, *Computer Methods Appl. Mech. Engrg.*, pp. 325-344, Vol. 167.

Smith, D.E.; Tortorelli, D.E.; Tucker III, C.L. (1998): Optimal Design for Polymer Extrusion : Parts I and II", *Computer Methods Appl. Mech. Engrg.*, Vol. 167, pp. 283-324.

Spoerre, Julie; Zhang, Chuck; Wang, Ben; Parnas, R.S. (1998): Integrated Product And Process Design For Resin Transfer Molded Parts, Journal of Composite Materials v 32 n 13.

Tan, C. P.; Springer, G. S. (1999): Composite manufacturing: Simulation of 3-D Resin Transfer Molding, *Journal of Composite Materials* v33 n18.

Vanderplaats, G.N. (1984): *Numerical Optimization Techniques for Engineering Design*, McGraw Hill Press Company.

Young, W-B; Yu, H-W (1997): Optimal Design of Process Parameters for Resin Transfer Molding, Journal of Composite Materials, v. 31, num. 11.