GRAPH DESIGN USING KNOWLEDGE-DRIVEN, SELF-ADAPTIVE MULTI-OBJECTIVE EVOLUTIONARY GRAPH ALGORITHMS

Christodoulos A. Nicolaou

University of Cyprus, 2010

Optimal Graph Design (OGD) is a problem frequently occurring in several common applications ranging from designing communication and transportation networks to discovering new drugs. More often than not the graphs to be designed need to satisfy multiple, conflicting, objectives e.g. total length, complexity or other shape and property limitations. In addition to problem specific criteria, the methods proposed to solve the problem need to consider several issues related to the representation of the solutions and the manipulation of graphs. These graph-structure specific issues coupled with the multi-objective nature of OGD form a challenging problem of increased complexity with wide applications in several research fields.

Our research proposes, MEGA, an algorithmic framework for solving the problem of multi-objective optimal graph design for labeled, undirected graphs. The method uses the multi-objective evolutionary graph, a graph-specific optimization meta-heuristic that combines evolutionary algorithms with graph theory and local search techniques exploiting domain-specific knowledge, to efficiently explore the feasible search space and obtain multiple equivalent compromising solutions. The algorithm introduces a novel niching mechanism that takes into account both parameter and objective space solution diversity. Morevoer, the method implements a self-adaptive approach to control and ensure appropriate local search use. In the experimental section we present results for the problem of designing molecules satisfying multiple pharmaceutically relevant objectives. The results suggest that the method can provide a variety of valid, interesting graph solutions. In comparisons with commonly used algorithms, MEGA is found to produce statistically significant better results.

GRAPH DESIGN USING KNOWLEDGE-DRIVEN, SELF-ADAPTIVE MULTI-OBJECTIVE EVOLUTIONARY GRAPH ALGORITHMS

Christodoulos A. Nicolaou

A Dissertation
Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy
at the
University of Cyprus

Recommended for Acceptance
By the Department of Computer Science
June, 2010

APPROVAL PAGE

Doctor of Philosophy Dissertation

GRAPH DESIGN USING KNOWLEDGE-DRIVEN, SELF-ADAPTIVE MULTI-OBJECTIVE EVOLUTIONARY GRAPH ALGORITHMS

Presented by

Christodoulos A. Nicolaou

Research Supervisor

____

Constantinos S. Pattichis

Committee Member

____

Christos N. Schizas

Committee Member

____

Christos Christodoulou

Committee Member

____

Dimitrios Fotiadis

Committee Member

____

Georgios Kontaxakis

University of Cyprus

June, 2010

Acknowledgements

This dissertation has been a task I have been working on for over 10 years. My initial plans, back in 1998, for earlier completion were put on hold due to my desire to explore the world of applied innovation, commercial software development and entrepreneurship. That desire, although still firmly in place, never extinguished my drive to complete my studies and invest time and effort to thoroughly investigate the potential benefits of knowledge acquisition, use and transfer to everyday problems. To me, it was only a matter of time to formally enter the appropriate Ph.D. program when circumstances allowed, something which happened in 2004.

Still, the completion of what started out as a side-task required the support of numerous family members, friends and colleagues to whom I am greatly indebted and grateful. Above all I would like to thank my family starting from my daughters Sonia, Polymnia and Thea and son Antonis for putting up with me during this time, especially the final two years when I was more often absent than not; my parents who have instilled the urge for learning and improving myself; and, last, but not least, my wife Artemis, without whose continued support, encouragement, understanding and love this work would have most probably remained a dream.

Credits

My efforts would never have been successful had it been not for the mentorship, support and friendship of Prof. Costas Pattichis with whom I have worked for the past 6 years. It truly would have been a much more difficult task without his advice and guidance and, at times, patience.

This work also owns a lot to Christos Kannas, whom I had the privilege of working with and mentoring for his M.Sc. degree. Christos in fact implemented many of the techniques used for the validation of the work described and offered substantial support in running experiments and compiling results. He has been an excellent student with an enquiring mind, the type of whom ultimately “forces” the mentor to learn as much if not more than he teaches.

Thanks are also due to Dr. Joannis Apostolakis, for insightful discussions and access to the software he and his team at Ludwig Maximillian Universitat have developed, Prof. Emmanuel Mikros of the University of Athens for providing expert guidance, Prof. Anna Tsantili also of the University of Athens for reviewing sections of this dissertation and providing valuable feedback, Prof. Christos Christodoulou for numerous discussions and helpful pointers, Dr. Terence Brunck for long, invigorating discussions ultimately leading to my decision to select the specific research topic and, Dr. Susan Bassett for giving me the opportunity to enter the research informatics field and experience my first successes as a young student and, later, professional.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

2D – 2-dimensional

3D – 3-dimensional

ADME – Absorption-Distribution-Metabolism-Excretion

ANN – Artificial Neural Network

API – Advanced Programmer Interface

BFS – Breadth First Search

CG – Compatibility Graph

CPU – Central Processing Unit

DFS – Depth First Search

DND – De Novo Drug Design

EA – Evolutionary Algorithm

EMO – Evolutionary Multi-objective Optimization

ER – Estrogen Receptor

FSP – Flow-Shop Scheduling Problem

GA – Genetic Algorithms

GNP – Genetic Network Programming

GP – Genetic Programming

HC – Hill Climbing

ICCS – International Conference on Chemical Structures

ID – Identity

ITAB – International Conference on Information Technology and Applications in Biomedicine

MA – Memetic Algorithms

MC – Monte Carlo

MCIS – Maximum Common Induced Subgraph

MCES – Maximum Common Edge Subgraph

MCS – Maximum Common Substructure

MEGA – Multi-objective Evolutionary Graph Algorithm

MOCO – Multiobjecitve Combinatorial Optimization

MOEA – Multi-objective Evolutionary Algorithm

MOGA – Multi-objective Genetic Algorithm

MOP - Multi-objective Optimization Problem

MOOP – Multi-objective Optimization

MOS – Maximum Overlapping Set

NFL – No Free Lunch theorem

NPGA – Niched Pareto Genetic Algorithm

NSGA – Nondominated Sorting Genetic Algorithm

OGD – Optimal Graph Design

PAS – Pareto Approximation Set

PDB – Protein Data Bank

PEA – Parallel Evolutionary Algorithm

pMEGA – Parallel MEGA

PIC – Population Intelligence Coefficient

QSPR – Quantitative Structure-Property Relationship

RECAP – Retrosynthetic Combinatorial Analysis Procedure

RL – Reinforcement Learning

SA – Simulated Annealing

sMEGA – simple-MEGA

SOP – Single-objective Optimization Problem

SPEA – Strength Pareto Evolutionary Algorithm

STIR - Stagnation Identification and Resolution

Tox - Toxicity

TSP – Traveling Salesman Problem

Chapter 1

Introduction

1.1 Overview

Optimal Graph Design (OGD) is a problem of significant interest to a wide range of problem domains including engineering (e.g. transport network design), telecommunications (e.g. wireless antenna positioning), electronics (e.g. digital circuit design), computer science (e.g. neural network architecture) and life sciences (e.g. molecular design) [G89], [BBS97], [CS04]. The objective in all of these applications is to design from scratch, or refine from a given initial graph, the optimal graph(s) satisfying any constraints or criteria imposed on the problem. Applied OGD, as most other real life problems, is often subject to multiple criteria or objectives. Depending on the specific problem these objectives may include requirements related to the sets of vertices and edges to use in the graph [EMS01], limitations on the size of the graph, constraints on the number of edge crossings and the type of angles used [UBSE98], or matching specific 2D shapes or 3D volumes [NAP09]. In addition to the problem specific criteria, methodologies proposed to solve the OGD problem need to take into account several issues related to the graph representation of the solutions. Such issues include the encoding of the graphs using appropriate data structures, the generation of valid -according to the specific problem- graphs, the construction and preservation of certain geometries and the accommodation of special topological features related to the problem. These graph-structure specific issues coupled with the multi-objective nature of applied OGD form a challenging problem of increased complexity with wide applications across several research fields.

The purpose of the research described in this thesis is to propose an algorithmic framework for the problem of multi-objective optimal graph design for labelled, undirected graphs. Solutions to this problem are graphs consisting of components from two sets, the set of vertices and the set of edges. Multiple types/labels of vertices and edges are allowed therefore, the problem suffers from the combinatorial explosion of the number of potential graph solutions. Similar to most multi-objective problems, the OGD problem usually has a complex, multi-modal solution space due to the multiple potentially conflicting objectives that need to be satisfied by the solution graphs and, the combinatorial nature of the problem. Consequently, from a computational optimization perspective the problem corresponds to searching the huge space of valid graphs to discover and select few designs satisfying, or compromising in the case of conflicts, the objectives imposed. In this context, validity of the resulting graphs is problem specific and, as such, the inclusion of problem domain knowledge to the process can facilitate the process [CLV07]. To solve the problem, a search strategy capable of efficient exploration of the space needs to be implemented.

1.2 Strategy

The strategy required to address the OGD problem needs to combine efficiency, to enable sufficient exploration of the space, and effectiveness since the presence of multiple objectives and the complexity of the space point to the presence of multiple solutions at different regions of the space. Our research aims to develop an algorithmic framework that can be generally applicable to problems requiring multi-objective optimization of solutions that may be better and more naturally represented as graphs. The approach we have chosen to follow is twofold, blending global search methods with local search techniques to create a new, hybrid algorithm, and also expanding on current research on Evolutionary and Memetic Algorithms by introducing novel features, such as self-adaptation capabilities. Global search is achieved via the use of multi-objective evolutionary algorithm principles while local search is driven by knowledge, either specific to the problem available through previous efforts or, acquired during the search process. In this sense, the proposed methodology we term Multi-objective Evolutionary Graph Algorithm (MEGA) [NAP09] is both knowledge-driven and self-adaptive, related to research performed in the fields of Memetic Algorithms and Genetic Local Search [M89], [K02].

An implementation of the proposed framework is applied to the problem of computer-aided small molecule design [SF05] complemented by domain specific techniques to design chemical structures satisfying multiple pharmaceutically related objectives. In this problem the primary objective is that the chemical design exhibits appropriate pharmacodynamic properties (i.e., binding affinity) to the target receptor while satisfying several additional objectives/constraints that are essential for a molecule to be an effective and safe drug. Predicted performance of chemical designs has been provided by computational modeling methods developed for the purpose of this work that enable scoring of molecular designs to each objective considered to guide chemical space search [NAP09].

The evaluation of our research is performed on test cases with potential practical use to the drug design community via both, quantitative measures implemented for this purpose, and qualitatively through expert validation of the results. Quantitatively, computational performance measures widely used for the validation of multi-objective methods are used for the evaluation of the results produced by the experimental runs. Numerous runs with multiple starting populations and attribute settings are rigorously applied to assess the ability of the algorithms to consistently converge. Qualitatively, the algorithm implementation is evaluated through its performance in producing chemical structures satisfying multiple objectives in real problem cases selected in collaboration with expert academic partners. The qualitative evaluation of our results is facilitated by the choice of test cases with known 3D structures and solutions (i.e., ligands). The available knowledge is used for comparison purposes with our results and provides indications for the success of the method. Expert validation is performed by collaborating partners in medicinal chemistry.

It is worth noting that the general algorithmic framework may be used in numerous other problems, for example the graph-drawing problem [BBS97] where the aim is to design graphs with an optimal layout according to some aesthetic criteria, the traveling salesman problem [G89] where a graph satisfying the requirement for the shortest route visiting all the cities is sought, or the design of a telecommunication network [CS04] where the requirements may be related to the determination of the vertices and the links of the network and the selection of the best possible router patterns to use to ensure the best possible coverage at the minimum cost.

1.3 Original Aspects

In the field of optimization research, when confronted with complex, unknown search landscapes, it is common to utilize algorithms inspired by nature. Such algorithms, known as Evolutionary Algorithms (EA), are known to have excellent capabilities for global search even when little or no information is available about the underlying functions defining the feasible search space [G89]. EAs are stochastic, heuristic-based approaches to objective optimization that have been shown to be especially suitable for exploring large search spaces [MZ96]. EAs have also been used extensively for multi-objective problems with several Multi-objective Optimization Evolutionary Algorithms (MOEA), sometimes referred to as Evolutionary Multi-objective Optimization (EMO), cited in the literature [CLV07]. However, EA-based approaches suffer from comparatively slow performance which may limit applicability in several problem cases and cannot guarantee the discovery of the optimal solution [KCL08]. A related class of techniques, Memetic Algorithms (MA), has been proposed to alleviate these problems. MAs combine the excellent global search capabilities of EAs with local search methods to improve performance and at the same time reduce the time to convergence [M89], [K02]. In effect MAs incorporate appropriate methods known to excel in local search problem settings to EAs to guide search and expedite convergence.

Our research is expanding on current state of the art Multi-objective Optimization (MOOP) research to introduce MEGA, a new class of self-adaptive multi-objective memetic algorithms that use graphs for solution representation and exploit available knowledge to control and enhance search space exploration and efficient solution discovery. The method combines the best characteristics of EAs and local search with graph theory elements, data mining techniques and knowledge-driven approaches to meet its goals.

Specifically, the work presented in the following chapters combines and advances the latest research from the fields of multi-objective optimization, memetic and evolutionary algorithms and optimal graph design. Although each of the three fields has periodically seen intense research, work at bridging them and utilizing ideas across research domains has been extremely limited. Most existing implementations in the optimal graph design field use evolutionary algorithms but few use graph chromosome representations and even fewer attempt to accommodate the conflicting objectives imposed by the problem through MOEA algorithms. Alternatively, a number of graph drawing applications resort to the use of evolutionary algorithms and several use graph-based chromosomes but very few have attempted to take into account multiple objectives. Our strategy has been to select the best approaches, as reported in the literature, integrate them and build on them to form new, hybrid optimal graph design methods able to search complex, vast solution spaces both effectively and efficiently. On several occasions algorithmic novelties had to be introduced to ensure improved performance of the proposed systems. The list of original aspects introduced includes:

- **Graph-based chromosome representation:** Typical Evolutionary Algorithms use bit strings to represent the individual chromosome members of the population. While in some problems this representation is convenient, when solutions to a problem can be naturally represented as graphs, the usage of bit strings requires creative techniques to encode solutions appropriately. This encoding and subsequent decoding step has three main drawbacks: (a) increased algorithmic complexity (b) information loss and, (c) performance costs. Our approach to design and use graphs as chromosomes deals with all of these problems effectively. This approach has seen only limited use previously and our results support its usefulness and demonstrate its advantages to the scientific community.
- **Information-rich subgraph genes:** Our methodology provides for subgraph genes containing information about their weight and likely attachment points. Weights are set so as to encode available knowledge about the privileged status, or suitability, of the specific subgraph for the problem investigated. Information on attachment points enables appropriate graph extension/modification according to the problem constraints. Taken together, information-rich subgraph genes have the ability to encode problem specific knowledge to favor the design of more promising graphs and increase chances to success.
- **Niching mechanism:** The proposed framework uses a unique niching mechanism designed to preserve population diversity in both, parameter (genotype) and objective (phenotype) space. The mechanism makes use of graph clustering to ensure that a variety of different promising graph designs survive long enough in the evolutionary cycle to contribute to the solution search. It is worth noting that the niching mechanism does take into account

the performance of the solutions in objective space in order to ensure that the solution set produced combines chromosome diversity with performance.

- **Elitism:** The algorithm proposed is -to the best of our knowledge- the first genetic graph algorithm reported to incorporate a secondary population mechanism designed specifically to preserve good nondominated solutions from getting lost. Elitism is also incorporated by allowing solutions from previous generations to compete on equal terms with the solutions created by evolutionary steps, i.e., mutation and crossover.
- **Exploitation of Knowledge (Memetic) Component:** The algorithm incorporates novelties that exploit knowledge of two types: problem-specific knowledge available during the set-up of the search process and knowledge acquired during optimization. To accommodate available problem specific knowledge the method incorporates simple heuristics to improve local search efficiency. These heuristics involve the usage of subgraph genes weighted according to information about the performance of pre-existing graphs containing them, rules to restrict the formation of new graphs using appropriate edge types and attachment points and custom graph-specific mutation and crossover operators. Knowledge obtained during the optimization search requires first, a mechanism to discover that knowledge and second an operator to exploit it. The discovery mechanism relies on the use of quantitative performance measures monitoring the progress of the search and, thus, providing information on conditions indicating premature convergence or stall/stagnation of the process. Exploitation takes place through a memetic mechanism which uses a novel, self-adaptive, anti-stagnation mechanism that attempts to escape local minima by selectively applying local search techniques to intensively explore local neighborhoods for promising solutions and enrich the population with new genetic material. This memetic, self-adaptive component of the algorithm complements the

global search capabilities inherent to EA-based approaches, provides a means for focused exploration of select space regions and promotes steady search progress.

At the application level, our algorithm has made contributions through its application to a unique type of small molecule structure design -also known as de novo drug design- problem, that of selectivity between closely resembling target receptors. In this context, molecules are interpreted as graphs whose vertices and edges correspond to atoms and bonds respectively [LG03]. This is in line with current structure-activity relationship theory according to which, drug action is a function of its chemical graph and specifically its ability to complement, or bind to, specific sites of biological targets (the receptors) and, thereby, cause certain biological effects [MG04]. We have designed tests cases where for example, the solutions are required to bind effectively to a “positive” pharmaceutical receptor and show little or no affinity to a “negative” pharmaceutical receptor. The compound selectivity problem is of immense importance to the drug discovery community, since a drug needs to bind selectively to the target it was designed for and avoid off-target interactions which cause undesired side-effects and toxicity. Exploiting the unique features of the proposed method, especially multi-objectivity, population-based search, graph representation of chromosomes, chromosome structure-based niching and self-adaptive local search, we have shown that our method can successfully design compounds binding to ER-β and not to the highly similar ER-α. The above results have been presented at the [NPA08], and, published at the Journal of Chemical Information Modeling of the American Chemical Society [NAP09]. Additional computational experiments, designed to compare MEGA against widely used MOEA methods produced results indicating that, for the problem domain under investigation, the proposed algorithm outperforms competing methods at a statistically significant level. The latter results are described in later chapters of the present dissertation. A full list of the publications resulting from this dissertation is given in Appendix 1.

In summary, the implementation of our graph design and optimization algorithmic framework is able to generate solutions satisfying multiple, even conflicting objectives; the solutions are represented as graphs to avoid information loss and spurious results; the feasible search space is explored both in a global sense using the inherent global search properties of evolutionary algorithms to identify potential solutions of diverse nature, and, in a local sense through the intensive application of local search methods to detect the best possible solution of a certain region of the space. To expedite solution discovery, available knowledge, supplied a´ priori or generated during the optimization process is exploited through custom algorithmic evolutionary operations and self-adaptive means, among others. Special emphasis is also placed on system performance and scalability issues to ensure the practical usefulness of our research. To this end a pilot parallel implementation of our algorithm has been prepared and is also presented.

1.4 Thesis Organization

The remainder of this dissertation is organized in three sections: the first, consisting of three chapters, introduces the problem and provides the necessary background information and related research. An overview of the targeted research area of optimal graph design, multi-objective optimization and evolution-based search methods is also provided. The second section of the thesis, consisting of chapters 5 and 6 describes in detail the research performed both at an algorithmic and at an application level. Chapter 5 describes the method proposed and the implementation for the purposes of this work. Chapter 6 focuses on the experiments performed for the evaluation of the proposed method and discuses the experimental design followed and the results obtained from the validation tests performed. Section 3 consists of chapter 7 that discusses in detail the findings from the research performed and relates them to the general research area of multi-objective problems and algorithms, chapter 8, that draws conclusions and summarizes the lessons learned, and chapter 9, that presents open research questions related and/or inspired by this thesis and outlines directions for future research.

Chapter 2

Multi-objective Optimization Problem Considerations

Optimization problems involve the identification of solutions to one or more objective functions considering all the decision parameters involved in the specific problem. Optimizing an objective -also known as cost- function amounts to finding valid solutions minimizing or maximizing the function as defined by the problem at hand. Potential solutions are only valid if they comply with the constraints of the specific function(s). Optimal solutions are those optimizing, i.e., minimizing or maximizing, the cost function of the investigated problem. Without loss of generality, for the remainder of this thesis only minimization objective functions are considered. Definitions and solution processes are similar for maximization problems.

Famous optimization problems include the Traveling Salesman Problem (TSP) where the objective is to minimize the cumulative distance traveled by a salesman to visit a number of destinations D and return to their starting point, and, the Flow-Shop scheduling Problem (FSP) where the aim is typically to minimize the time for completing a number of jobs N on M machines. A solution to the TSP problem is valid if the proposed route begins and ends at the same point and visits all the destinations D. Moreover, a solution S_{opt} is optimal for a specific TSP problem if there is no other valid solution defining a shorter route for the journey of the salesman, i.e., it is a global minimum.

Optimization problems can be divided into two broad categories: single-objective and multi-objective, depending on the number of objective functions to be optimized.

The solution to a single-objective optimization problem (SOP) involves finding the optimum to one-objective function, whereas a multi-objective problem (MOP) problem requires the more difficult task of finding solutions that satisfy a whole spectrum of objectives [NBP07]. Depending on the type of decision variables optimization problems can be categorized into continuous, where the variables are continuous real numbers, discrete, where the variables are integer numbers, and combinatorial, where the variables consist of a finite set, e.g. the permutations on a set of numbers such as pairs of destinations in the TSP problem [CS04]. This dissertation focuses on optimal graph design, a multi-objective combinatorial optimization problem.

The next sections of this chapter describe in detail the mathematical definition of multi-objective problems and related notions such as domination, optimality and diversity, and, introduce the class of multi-objective optimization algorithms. The final section of the chapter presents the methods used in this research for assessing solution quality in MOPs and, thereby, measuring the performance of the available algorithms.

2.1 Multi-objective Optimization Problems

Solutions to multi-objective problems can be described by decision vectors $\vec{x}$ of the form (x_1, x_2, …, x_n) in the decision space X. Objective functions measure the quality of a decision vector and collectively assign it an objective vector $\vec{y}$ = (y_1, y_2, … , y_c) in the objective space Y [BST03]. In essence, multi-objective optimization focuses on optimizing (i.e., minimizing) a function of the form $\vec{f}(\vec{x})$ consisting of many objective functions. In this context, optimization is the process of identifying solutions in the decision space X minimizing the C objectives under consideration.

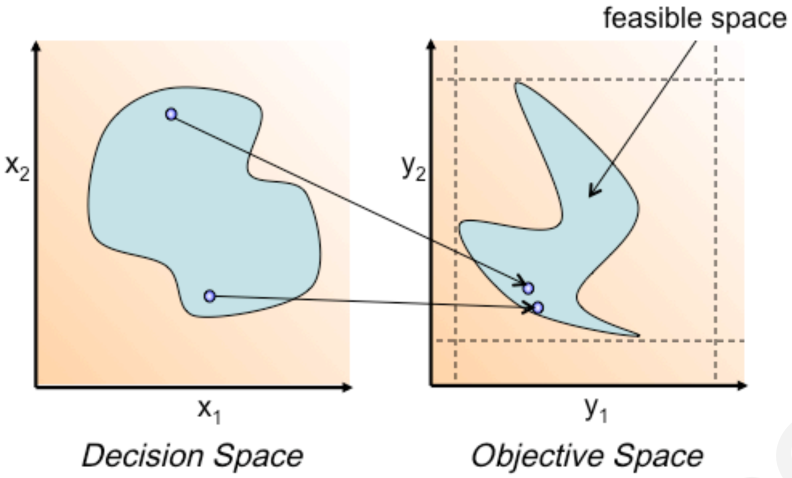


Fig. 2.1 Solutions (represented by small circles) of a multi-objective problem are mapped from decision space to objective space through objective functions. Proximity of solutions in objective space is not related to solution similarity in the decision space and vice versa. The objective space, as defined by the objective functions y_1 and y_2, is enclosed by the dashed lines. In practice, the feasible search space of a MOP may be further constrained by limitations related to solution validity in the decision or objective space.

In formal terms optimization is the process of [CS04]:

minimizing function $\vec{f}(\vec{x})$

with $\vec{g}(\vec{x}) \le 0$ inequality constraints Eq. (1)

and $\vec{h}(\vec{x}) = 0$ equality constraints

where $\vec{x} \in R^n$, n is the number of decision variables, $\vec{g}(\vec{x}) \in R^m$, m is the number of inequality constraints and $\vec{h}(\vec{x}) \in R^p$, p is the number of equality constraints. The constraints delineate the feasible search space for a solution $\vec{x}$ to be valid.

In the special case where the objective vector $\vec{f}(\vec{x})$ consists of a single objective function the problem is single-objective, i.e., $\vec{y}$ has a single element. Identifying optimal solutions in SOPs is relatively simple since comparing solutions is straightforward. For example, comparing two solutions to the TSP problem amounts to comparing a pair of numbers each corresponding to the total cost (length of journey) described by a solution. Searching for the optimal solution, the global minimum, is then performed via modifications of the decision vector variables that result in new candidate solutions.

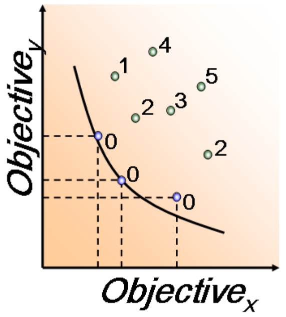


Fig. 2.2. A graph showing solutions to a bi-objective problem. Each point represents a potential solution to the problem. The curved line represents the Pareto-front, identified nondominated solutions are labeled with a Pareto rank of zero, and the Pareto rank of the other solutions refers to the number of solutions dominating it. Note that the problem requires minimization of both objectives and so the ideal solution is located at point (0,0).

Comparing solutions when the objective vector $\vec{y}$ consists of two or more elements is more complex. When the objectives are in competition, the task of finding a solution is further complicated because there is no single best solution that outperforms all the other solutions in all criteria [BM04]. Instead, several equally good solutions, the so-called nondominated solutions, exist representing various possible compromises among the objectives. For a particular solution to be nondominated it is necessary for it to have no superordinate solutions among all the objectives being considered. A specific solution s is a Pareto global optimum if and only if there is no s' $\in$ Y such that s' is better in one or more objectives and not worse in any objective, where Y is the set of feasible solutions [P06]. This is best illustrated in Fig. 2.2, where the optimal compromise surface between two objectives (depicted by the curved line) is shown together with a set of candidate solutions. Each solution is designated a Pareto rank – that is, is ordered according to the number of other solutions that dominate it over all objectives. It is important to note that for this specific example the ideal solution, usually only theoretically feasible, is assumed to be located at the bottom, left-most corner of the figure where both objectives have a zero value. Nondominated solutions have a Pareto rank of zero. The set of nondominated solutions is also known as Pareto-front or optimal set or the trade-off surface. We say that a set of nondominated solutions is the Pareto global optimum set if and only if it contains only and all Pareto global optimum solutions.

The following section considers component-wise vector orders in the Euclidean space since this binary ordering relation is the basis of Pareto-based multi-objective optimization techniques that are of interest to this research. Alternative ordering relations can be used which give rise to different methodological approaches to multi-objective optimization algorithms, such as the lexicographic relation [CS04].

Let $\vec{u}$ and $\vec{v}$ be vectors in the Euclidean space representing solutions of a multi-objective problem. Vector $\vec{u}$ dominates $\vec{v}$, denoted by $\vec{u} \prec \vec{v}$, if $u_i < v_i$, i = 1, ..., n where n is the dimension of vectors u and v, and $u_i \neq v_i$. Vector $\vec{u}$ weakly dominates $\vec{v}$, denoted by $\vec{u} \preceq \vec{v}$, if $u_i \leq v_i$, i = 1, . . . , n where n is the dimension of vectors u and v. Two vectors are nondominated if neither of them dominates the other. Similarly, two vectors are weakly nondominated if neither of them weakly dominates the other. Defining domination relations between pairs of vectors thus requires the component-wise ordering of the vectors. A solution that is not dominated by any other feasible solution is globally nondominated, also known as Pareto global optimum. Mathematically the domination relation is defined as:

Vector $\vec{u}$ dominates vector $\vec{v}$ if:

$\vec{u}_i$ is at least as good as $\vec{v}_i$ for all the objectives i Eq. (2)

$\vec{u}_j$ is better than $\vec{v}_j$ for at least one objective j

Typically, and especially in the case of conflicting objectives, there is a multitude of Pareto global optima. Note that the Pareto global optimum set is formed by the collection of the Pareto solutions in the decision space. The image of this set in the objective space is called the efficient set or the Pareto-front [PSS07], [CLV07].

Multi-objective problems are often characterized by vast, complex search spaces with various local optima that are difficult to explore thoroughly. The challenge facing Pareto-based optimization methods is to ensure the convergence of well-dispersed solutions to guarantee the effective coverage of the true optimal front (Fig. 2.3A) [ZLB04]. The Pareto ranking algorithm is of the order O(MN^2), where M is the number of objectives, and N is the number of data points [NBP07]. Such an order can therefore be computationally expensive for large numbers of objectives and data points, and lead to non-convergence of the solutions (Fig. 2.3B). A further potential issue with Pareto ranking is that the nondominated frontier of solutions may be vast, particularly in circumstances with large numbers of objectives. The distribution of solutions on the Pareto-front may also lead solutions to drift to more densely distributed regions of the surface and, in more extreme circumstances, lead to dictatorship conditions where a single objective dominates (Fig. 2.3C). Therefore, it is often prudent to employ techniques, known as niching, to ensure the appropriate distribution of the solutions produced and the coverage of the Pareto-front. Typically, niching uses techniques for maintaining diversity in the population through solution similarity calculations, clustering analysis, etc [CS04].

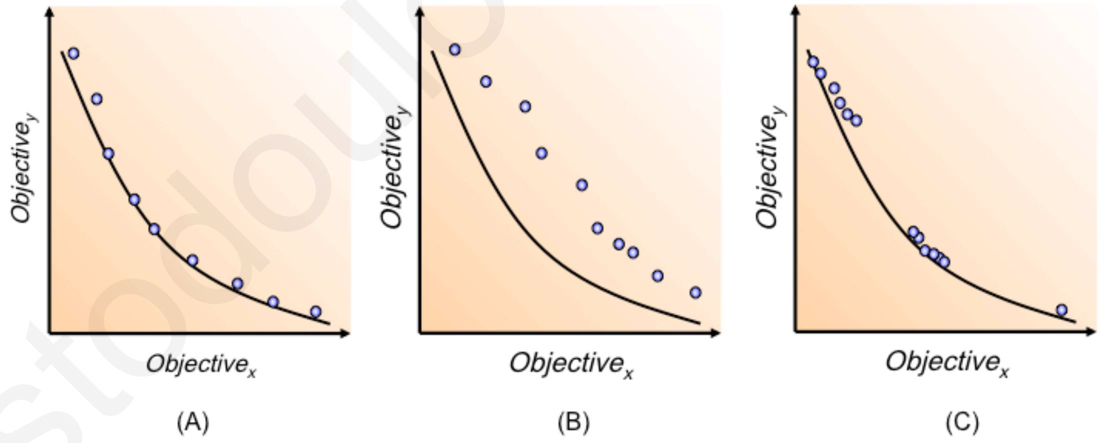


Fig. 2.3: The solution set produced by a Pareto-based multi-objective optimization method must converge to the true Pareto-front and cover it effectively. (A) Effective coverage of the true Pareto-front. (B) Non-convergence. (C) Lack of effective coverage.

Solution proximity in the objective space is not necessarily indicative of similarity between solutions in the decision space (see Fig. 2.1). Consequently, it is possible for multiple distinct solutions in the decision space to share the same objective function vector and, therefore, one single point on the Pareto front may correspond to several solutions in parameter space. Typical MOOP algorithms are commonly interested in one solution for each element of the efficient set and do not attempt to identify multiple solutions giving rise to a single Pareto solution. However, there are specific cases where solution diversity in parameter space is equally -if not more- important as in the case of the optimal graph design problem. Therefore, it is a specific goal of the work presented in this dissertation to address solution diversity in both decision and objective space.

2.2 Multi-objective Optimization Algorithms

Multi-objective optimization algorithms can be classified based on the quantity of the solutions produced; for example, those producing single 'best' solutions and those that attempt to map the entire Pareto-front. A second classification scheme divides MOOP methods into the following three groups: (a) a´ priori methods that take into account the preferences of the user before the optimization process is conducted; (b) progressive methods that enable the user to interact with the optimization process to guide the search; and (c) a´ posteriori methods that produce a Pareto-front and allow users to choose the most appropriate solution subset [VL00].

The straightforward approach to finding compromise solutions when numerous objectives are present is to transform the problem to a single-objective one by combining the multiple objectives. An example of this approach is the weighted-sum-of-objective-functions method [CS04]. According to this method, a weight is associated a´ priori with each objective function and the weighted sum of the functions is taken as the new composite fitness function, as defined by the following equation:

$$f(n) = w_1(\text{Objective}_1) + w_2(\text{Objective}_2) + \ldots + w_n(\text{Objective}_n) \qquad \text{Eq. (3)}$$

where f(n) is the fitness function, and w_i are the user defined weights. A great advantage of using such a scalarized objective function is that the same algorithms used for solving single-objective problems can be used for multi-objective problems. A major drawback of the method is the selection of the most appropriate weighting, because it is often not clear how the different objectives should be ranked. A further drawback is that the method ignores the presence of the Pareto-front of the objectives, leading to unpredictability about where the identified solutions will lie on this surface. Finally, the method is limited in its ability to find solutions to problems involving competing objectives [CS04] since, when finding optimal solutions with respect to the scalarized objective function, only solutions on the convex hull of the efficient set, can be obtained [CLV07].

Pareto-based MOOP methods introduce a different approach to optimization that is founded on compromises and trade-offs among the various objectives. The aim of MOOP methods is to discover a set of satisfactory compromises and, through them, the global optimal solutions by optimizing numerous dependent properties simultaneously [CS04]. In terms of Pareto optimality, the goal of these methods is to find solutions that are not worse than any other solution and strictly better in at least one of the objectives. The major benefit of MOOP methods is that local optima corresponding to one objective can be avoided by consideration of all the objectives simultaneously, thereby escaping single objective dead-ends and leading to a more efficient overall process. Moreover, the ability of Pareto-based methods to optimize numerous properties simultaneously helps to avoid the pitfalls associated with methods combining multiple objectives into a single one described previously. The methods return a set of nondominated solutions, the Pareto-approximation set (PAS), and allow the users to choose the solutions that are most suitable for the task a´ posteriori. Ideally, the PAS should consist of a well-distributed set of solutions representative of the true Pareto-front and, therefore, there is a need for both, identifying true Pareto global optima and preserving solution diversity. To this end, population diversity analysis is applied either in objective or decision space with the former being the area of most activity. It is worth noting that few efforts to combine population diversity in both decision and objective space concurrently have been recorded and therefore this specific research direction remains largely unexplored. The algorithm proposed in Chapter 5 aspires to contribute in filling this gap. Section 3.3.2 elaborates further on a popular class of Pareto-based MOOP methods, the multi-objective evolutionary algorithms.

2.3 Quantitative Quality Assessment Measures

Quantitative measuring of the quality of the solution set produced by a Pareto-based MOOP method has attracted considerable interest in recent years [VL00], [ZTLF02], [CS04], [TKL05]. The presence of multiple equivalent solutions combined with the frequent lack of information of the best possible set of solutions complicates the process further. A number of performance assessment measures have been proposed in the literature aiming to quantify the goodness of a produced Pareto-approximation set. Some of the methods focus on calculating the proximity of the approximation to the true Pareto-front, others on capturing characteristics such as solution diversity and dispersion while others are concerned solely with the comparison of two proposed approximation sets, the so-called relative performance metrics. Despite the multitude of methods proposed, no single measure has been proven to be able to provide definitive information on the quality of the Pareto-approximation set produced and therefore combinations of measures are often used.

The performance measures encoded for the purposes of this research include the calculation of the Pareto-approximation set hypervolume [ZT99], the spacing measure [CS04] and a measure based on the calculation of the average diversity [TTW97]:

a. **Hypervolume:** The hypervolume measure HV, also known in the literature as the S metric, calculates the volume of the objective space dominated by an approximation set and produces an arithmetic value indicative of the quality of the set [Z99]. Note that in the case of two objectives the measure actually describes the area dominated by the Pareto-front assessed. Calculations take into account a reference, nadir point, i.e., a sort of a “worst” solution, to define the boundaries of the dominated space as seen in Fig. 2.4 for a bi-objective problem case. The reference point may be defined by taking the worst value for each objective from the solutions in a population or via other, typically heuristic, means. Care must be exercised in the selection of this point in order to enable valid performance comparison between Pareto-approximations sets produced by different experiments [ZBT07].

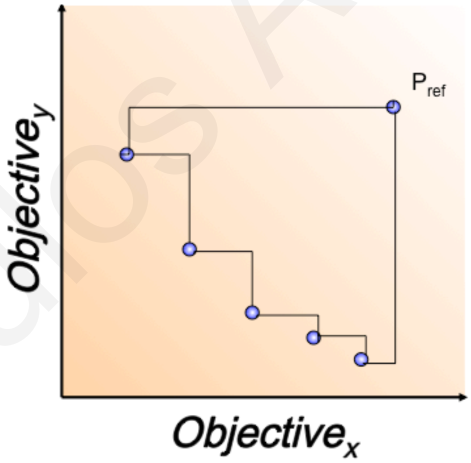


Fig. 2.4: A Pareto approximation set and the hypervolume it defines with respect to the reference point P_{ref} (enclosed by the solid line). The hypervolume measures the size of the dominated space covered by the Pareto-set. In the case of two objectives the measure describes the area dominated by the front assessed.

Hypervolume is defined by the combination of hyper-rectangles formed by individual solution vectors and the reference point. Formally, the definition of the measure is:

$$HV(PAS, P_{ref}) = \bigcup_{i \in 1..|S|} HV(PAS_i, P_{ref}) \qquad \text{Eq. (4)}$$

where PAS is the Pareto approximation set of solutions $i \in 1..|S|$.

Given a solution A = (y_{a1}, y_{a2} , …, y_{an}) in PAS and the reference point P_{ref} = (p_1, p_2, …, p_n) dominated by A, the hyper-rectangle enclosed in the space bounded by A and P_{ref} is defined by intersections of the following hyper-planes: for each axis in the objective space, there exists a hyperplane perpendicular to the axis and passing through P_{ref} and, for each axis in the objective space, there exists a hyperplane perpendicular to the axis and passing through y. Figure 2.5 illustrates the hypervolume defined by two solutions A and B individually and collectively.

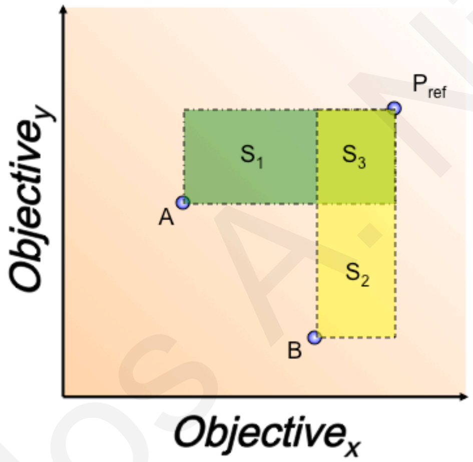


Fig. 2.5: The hypervolume measure is defined as the size of the dominated space by a Pareto-set and a reference point. The size of the space is calculated for each point of the Pareto-set individually and then combined. In a two dimensional problem the space defined by a solution A and the reference point P_{ref} is depicted by the union of the rectangles S_1 and S_3; the space defined by solution B and P_{ref} is depicted by the union of the rectangles S_2 and S_3; the hypervolume of the set {A, B} with respect to P_{ref} is the union of the space defined by points A and B individually, i.e., rectangles S_1, S_2, and S_3.

b. **Spacing:** The spacing measure S is based on the distribution of the nondominated solutions on the Pareto-surface in objective space [CS04]. Ideal surfaces have uniformly spread solutions along the Pareto-surface. The metric is defined as:

$$S = \left[\frac{1}{n-1} . \sum_{i-1}^{n} \left(\bar{d} - d_i \right)^2 \right]^{\frac{1}{2}} \qquad \text{Eq. (5)}$$

where

$$d_i = \min_i \left(\left| f_1^i(\vec{x}) - f_1^j(\vec{x}) \right| + \left| f_2^i(\vec{x}) - f_2^j(\vec{x}) \right| \right) \qquad \text{Eq. (6)}$$

$i \neq j$, n is the number of solutions in the approximation set and $\overline{d}$ is the mean value of all d_i.

c. **Diversity:** The diversity measures used aim to capture the average solution diversity either in the parameter or the objective space. Two measures, one for each space have been encoded. The measures are essentially calculated using the same method, i.e., by averaging the Euclidean distances of each solution to all other solutions in the proposed set [TTW97]. The definition of the measure is:

$$D = \frac{\left[\sum_{i=1}^{n}\sum_{j=1}^{n}(d_{i,j})^2\right]^{\frac{1}{2}}}{n^2} \qquad \text{Eq. (7)}$$

where $d_{i,j}$ is the Euclidean distance between the vectors representing solutions *i* and *j* and n is the number of solutions available. The Euclidean distance between vectors A, B is defined as:

$$E(A,B) = \sqrt{\sum_{j=1}^{j=n}(x_{jA} - x_{jB})^2} \qquad \text{Eq. (8)}$$

where j is the length of vector elements and x_{jA} is the jth element of the vector of object A. Calculating solution diversity in objective space uses the objective function vectors $\vec{y}$ of each solution. Solution diversity in parameter space is problem specific since it uses the vector representation of the solution description i.e., in the case of the TSP problem the actual journey description.

In the remainder of this dissertation the performance measures described above are used to calculate the quality of each solution set produced during our experimental runs using popular multi-objective algorithms described in the literature or the algorithm proposed by this research. The combination of these measures allows the comparisons of the Pareto-surfaces produced by each algorithm and facilitates the extraction of conclusions with respect to their performance.

Chapter 3

Optimization Methods and Evolutionary Algorithms

An optimization algorithm is a method for searching for solutions to a problem given a representation of the problem and a way to measure the quality of a proposed solution. The methods are typically used to find the global optimum(s) in non-trivial problems with large solution spaces. Several classifications of optimization algorithms are possible based on various characteristics such as the encoding of the solutions, the methodology used to generate candidate solutions, the specific strategy used for exploring the feasible solution space, the method for evaluating the quality of the candidate solutions, etc [CS04].

The simplest example of such an algorithm would be the total enumeration of all possible solutions, the evaluation of each one of them and the selection of the one producing the best results. This naive, brute-force approach suffices when the search space is sufficiently small and the evaluation of each candidate solution is trivial. However, with increasing problem dimensionality and complexity this approach becomes impossible. Other approaches utilize systematic search strategies or special properties of the solution space to find an optimal solution in an efficient manner. Local search algorithms use local properties of the solution space to guide the search and tend to work well in some problem cases especially where the target function is unimodal. Global search algorithms try to explore the whole of the search space when trying to detect the optimal solution. These techniques are appropriate when the target function is multimodal, e.g. contains multiple local and global optima, or completely unknown. Heuristic-based approaches employ problem specific information to limit the size of the search space and increase the efficiency of the optimization process. Heuristics are rules of thumb used to help solve problems by incorporating problem domain knowledge. Artificial Neural Networks (ANN), a technique inspired by the structure of biological neural networks and their way of encoding and solving problems, have also shown great promise in identifying approximation solutions in difficult optimization problems [PS03]. ANNs explore the search space through iteratively modifying their structure in order to reach good solutions in a systematic manner. Evolutionary Algorithms (EAs) are a special class of techniques often applied to search for the optimum when the optimization problem is overly complex due to high dimensionality, noise, non-linearity or other unusual properties of the search space [CLV07].

Following is a brief overview of some traditional search-based optimization methods. Although these methods may not be the most sophisticated optimization techniques they can be very useful in the case of non-complicated solution spaces, e.g. single optimum point, or as first pass approaches to get an estimate of the magnitude of the solution space and the complexity of the problem. Heuristic-based approaches are discussed further in a later section of the chapter. Throughout this chapter emphasis is placed on the review of approaches used later in this dissertation.

Blind-Search Methods

Methods following the Blind-Search paradigm typically make no use of information related to the problem domain. A typical example of these methods is Monte-Carlo (MC) Search, also known as Random Search [W08]. MC is an iterative blind search strategy that makes no use of knowledge gained from previously generated solutions. In each iteration, MC performs a random sampling of solutions of the complete search space followed by their evaluation. The “best” solution is memorized and is only replaced if some solution in a subsequent iteration is superior. MC terminates when the “best” solution meets some predefined quality criteria or a certain number of iterations has been completed. A special case of blind search strategies are certain implementations based on search trees [RN95]. These strategies define a hierarchical tree structure where each node represents a solution or a step to a solution. In each following step the algorithm searches the neighborhood of the node following some strategy that does not take into account any problem related information. Examples of such tree-based blind-search strategies are depth-first search (DFS) and breadth-first search (BFS).

Heuristic-based Optimization Methods

Heuristic-based methods employ problem specific information to focus the search and expedite the solution finding process. In order to achieve this they incorporate domain knowledge often in the form of rules to guide the process to segments of the search space more likely to contain the solution sought. In the case of tree-based search problems mentioned previously a heuristic-based strategy would be the best-first search [RN95] where the best solution -according to some evaluation criterion- is explored first.

Several heuristic methods are inspired by the Hill-Climbing (HC) approach [W08]. HC algorithms start with a random initial solution. In each generation the current solution is mutated, i.e., it is subjected to a random modification. If the mutant is better, the mutant replaces the current solution; if not the current solution is kept for the next generation. In effect HC methods perform a greedy local search. Neighborhood-based iterative improvement algorithms are variation of HC that start from an arbitrary feasible solution and search neighbors for better solutions to replace the current one. The methods require the definition of a neighborhood function that associates a set of feasible solutions *N(s)*, the neighbors, to every feasible solution *s*. The solution *s* is a local optimum with respect to *N* if, and only if, there is no *s' ∈ N(s)* such that *f(s')* is better than *f(s)*. The neighborhood search is repeated until no improvement is found anymore and the algorithm stops in a local optimum [PSS07]. The HC strategy can be rather efficient in simple unimodal search spaces, but is prone to premature convergence in local optima in case of multimodal search spaces. To reduce the chance of premature convergence the Hill-Climbing strategies can be extended to a Multi-Start also known as Random-Restart Hill-Climber [RN95], where multiple local HCs start from randomly chosen initial solutions.

Simulated Annealing (SA) [KGV83] is an evolution of the HC methods where the replacing scheme is adjusted to keep more than just the best solution. This strategy is inspired by the mechanism of metals cooling into a minimum energy structure and the search for an optimum in a given solution space. This mechanism is simulated using a control parameter, the temperature, which is decreased during the optimization process. The degradation function for the temperature is called the Annealing Schedule and causes the optimization process to become more and more restrictive accepting better solutions towards the end of the optimization process. The speed of convergence and also the vulnerability to premature convergence of the SA strategy depends on this Annealing Schedule. The ability to allow temporary degradation in solution quality enables the SA to escape local optima. Similar to the Hill-Climber, a population in Simulated Annealing may be exploited using a multi-start strategy, which can further decrease the chance of premature convergence in a local optimum [W08].

Evolutionary Algorithms

Evolutionary Algorithms (EA) are a special class of stochastic, population-based optimization approaches that have been inspired by principles of natural evolution.

EAs employ heuristic search techniques and are generally categorized in the broadly defined domain of artificial intelligence [W08]. Any EA designed to solve a particular problem must have the following five components [MZ96]:

- A genetic representation for potential solutions to the problem
- A way to create an initial population of potential solutions
- Evaluation function(s) for rating solutions in term of their fitness
- Genetic operators that alter the composition of children
- Values for various parameters that the EA uses (population size, probabilities of applying genetic operators, etc.).

An EA starts with a population of usually randomly initialized individuals and iteratively applies genetic operators (e.g. recombination and mutation) and environmental pressure (e.g. evaluation function and selection process) to produce subsequent generations of candidate solutions. The process terminates when a termination criterion is met, often when the current set of individuals meets some quality threshold. The consecutive rounds of reproduction and selection explore various sections of the search space and gradually produce one or more optimized solutions that ideally converge close to optimal solutions. Figure 3.1 describes the main steps in the execution of EA algorithms.

```
Generate initial population P
Evaluate solutions in P against objective O
While Not Stop Condition:
        Select parents $P_{parents}$ in proportion to fitness scores
        Generate population $P_{offspring}$ by variation of $P_{parents}$
        Evaluate solutions in $P_{offspring}$ against objective O
        Select population $P_{new}$ from P and/or $P_{offspring}$
```

Fig. 3.1: General scheme for Evolutionary Algorithms

There are several variations of EA algorithms that differ mainly in the representation of the solutions and the genetic operators used to generate new candidate solutions by varying representatives from the current population. A widely accepted categorization distinguishes four main categories of EAs [MZ96]:

a) Genetic Algorithms which use bit-string solution representations and, typically, two parent crossover
b) Evolutionary Programming which uses a finite-state machine and mutation operators
c) Evolutionary Strategies which use real-valued vectors and Gaussian mutation, and,
d) Genetic Programming which employs an executable structure representation and two parent crossover.

It is worth noting that many EA implementations use strategies from a combination of categories and therefore cannot be classified to a single category.

EAs have been applied successfully to a variety of optimization problems even when the target function has been noisy, non-linear, non-differentiable, or multi-modal and high dimensional [W08]. The robustness of the method coupled with its limited requirements, i.e., a suitable objective function for a given solution to guide the selection process and an appropriate solution representation, have increased its popularity among researchers. Unfortunately, EAs often do not find the global optimal solutions. Rather, they are able to find sufficiently good solutions within a limited amount of time [G89]. In the following section we describe in more detail some of the main EA categories as defined by the encoding mechanism used to represent members of the population. Section 3.2 overviews the field of Multi-objective EA and 3.3 discusses Memetic Algorithms, a hybrid method that combines natural evolution principles with other search methods.

3.1 EA Categories

The representation of individuals in an EA is a task of crucial importance and several encoding mechanisms have been suggested for this purpose. These mechanisms influence the ability to represent accurately the individuals and measure their fitness. In addition, the mechanisms are instrumental to the processes of evolution since both mutation and crossover operations act on the chromosome structure. The main EA categories according to the representation mechanism used are described below.

3.1.1 Genetic Algorithms

Genetic Algorithms (GA), originally proposed by Holland [HJ75], are the most widely known representatives of EAs. GAs are stochastic, population-based search heuristics, which require nothing but the target function of the optimization problem to guide their search [W08]. Individuals, i.e., potential solutions to the optimization problem, are represented using a string of characters, the genes, often consisting of only two, “0” and “1”. The availability of a population enables GAs to simultaneously search various regions of the search space instead of focusing on the neighborhood of a single, best, current solution. GAs utilize the evolutionary operators of selection, random variation and mating, through mutation and recombination/crossover on the population of individuals.

```
Generate initial population P
Evaluate solutions in P against objective O
While Not Stop Condition:
        Select parents $P_{parents}$ in proportion to fitness scores
        Generate population $P_{offspring}$ by reproduction of $P_{parents}$
                Mutation on individual parents
                Crossover on pairs of parents
        Evaluate solutions in $P_{offspring}$ against objective O
        Select population $P_{new}$ from P and/or $P_{offspring}$
```

Fig. 3.2: General scheme for Genetic Algorithms

The initial population P of the GA is usually initialized randomly to obtain a diverse sampling of the search space. The increased diversity in the initial population of solutions enables the GA to search a sufficiently large section of the search space.

P is then evaluated and a fitness score is computed for each solution. The evaluation of the solutions is performed by assessing the fitness of the individuals to the optimization problem, i.e., by testing each individual on the target function. Following fitness calculation the algorithm enters its core, iterative section that continues until a termination criterion is met. The first step of the iterative cycle is to select possible parents from the current population based on their fitness score. The selected parents are then used to generate a new population $P_{offspring}$ mostly through recombination of parents. Recombination combines the characteristics/traits from multiple parents to generate novel solutions. The aim is to obtain suitable combinations of positive traits and, thus, individuals adapted to the optimization problem at hand. The process of recombination is achieved through the crossover mechanism where one point -or more- in the genotype of the parents is selected, and, the genotypes are split at that point and exchanged between the parents. A round of random mutations may also be applied on the $P_{parents}$ or the offspring from the recombination process often using a simple mutation operator to invert a randomly chosen bit of an individual. Recombination combines two or more parent solutions to form one or more offspring whereas a mutation results in a small random change to a single solution. A crossover and a mutation probability set the frequency of recombination and mutation respectively. In GAs recombination is considered to be the primary operator whereas mutation is intended for recovering lost traits in the population [F06]. It is important to note that the evolutionary operators of crossover and mutation usually treat the binary string as a whole and ignore word boundaries in the chromosome. The reproduction stage with recombination and mutation is then followed by the evaluation of the resulting solutions. The new population P_{new} is generated from the population of parents $P_{parents}$ and $P_{offspring}$ using some generation strategy, often the complete replacement of the $P_{parents}$ by $P_{offspring}$ [G89]. The process aims to increase the fitness of newer generation solutions and gradually find optimal solutions. Several extensions to GAs that use alternative solution representation, e.g. non-binary or real-valued genes, have also been proposed.

3.1.2 Genetic Programming

Genetic Programming (GP) is an extension of the GA that uses a procedural or functional representation for solutions [PLP08]. This representation enables GP to encode and optimize, for example, computer programs using EA principles. Examples of GP applications include the evolution of program code to solve a given programming problem [PLP08], the discovery of a mathematical function for a symbolic regression problem [W08] and the design of neural network architecture [RWPHM03]. Traditionally GP uses a tree-based representation for programs/solutions. This representation uses a directed acyclic graph with functions as nodes and terminals as leaves. The execution order is given by evaluating the left child node before the right node. An example is shown in Fig. 3.3. Note that except for the solution representation GP is identical to standard GA.

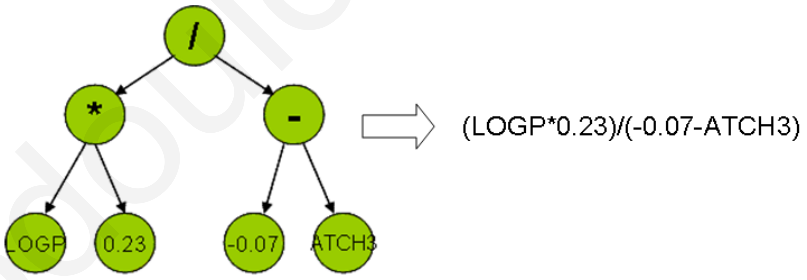


Fig. 3.3: The tree based GP geno- and phenotype

The initial population of programs for a GP application consists of a number of randomly initialized individual syntax trees. Tree initialization can take place using one of a number methods specifically designed for this purpose. Typically the process starts with a random root node and recursively new random nodes are added until a certain depth has been reached. Each program is then evaluated and a fitness score is computed for each solution [GSM04]. Similarly to a GA the algorithm selects a set of parents and applies genetic operators to generate offspring. In the case of GP, recombination is implemented as subtree crossover between two parents. The process simply selects two nodes of the parents and exchanges their subtrees. Mutation selects a random node in a GP program tree and alters the node accordingly depending on its type. Multiple alternatives have been suggested depending on e.g. whether the successive nodes of the mutation node remain unchanged. Subtree crossover tends to be the dominant operator in GP while mutation operators are often used at lower rates [PLP08]. The rest of the GP process, including the evaluation of the resulting solutions, the generation of the new population and looping, is similar to that of GAs. A common GP problem worth mentioning is bloat, which refers to the over excessive growth of trees with no analogous improvement in the fitness of the solution [W08]. Bloat can lead to a significant increase of computation time as well as over-fitting of an individual. Several strategies are usually employed to limit the overall tree size and avoid bloat.

3.1.3 Genetic Graphs

Graph representations of individuals have also been used in combination with EA methodology. Initial efforts appeared in the graph drawing field where efforts focus on the design of graphs with an optimal layout according to some measurable aesthetics such as the number of edge crossings, the variation in the length of edges, the variation of the angles etc. [BBS97]. Typically, graphs are represented by two distinct sets, a vertex/node set and an edge set. For example, to represent a graph with n nodes and m edges a *2*n* matrix may be used to indicate the positions of the nodes and a *2*m* matrix to indicate the edges by storing pairs of the nodes. The corresponding end points for an edge are then found from the node matrix [EM01]. Graph chromosome encoding may consist of the graph and edge tables only [EM01], a mechanism for serializing the information in the tables [MJV00], or graph data structures that enable manipulation of node and edge objects [G99]. In all cases, appropriate mutation and crossover operations need to be used to operate on the graph chromosome representation. Both node and edge mutation may be used. In node mutation, node positions, or labels if available, may be altered or swapped, while in edge mutation the pair of nodes connected by the edge may be modified. Additionally, mutation operations may include the removal of a node or an edge. Crossover operations involve the exchange of subgraphs. Often, genetic graph algorithms contain a large number of mutation and crossover operations that are meaningful for the specific problem under consideration. Fitness evaluation of the individual graphs involves a method for quantifying their performance, e.g. in the problem of graph drawing the method evaluates the aesthetics of a graph based on the number of edge crossings and the uniformity of the graph by taking into account the variation in edge length, distance between pairs of nodes and angle sizes. Genetic graphs have also been used in other problem areas including the design of digital circuits [MJV00], and small molecule design [G99]. An overview of graph theory fundamentals and a detailed description of genetic graphs literature with an emphasis on molecular design is presented in chapter 4.

3.2 Multi-objective Evolutionary Algorithms (MOEA)

Among the most popular algorithms used in optimization, including Pareto-based MOOP approaches, are evolutionary algorithms [CS04]. Intensive research efforts for almost two decades have focused on the application of EA methodology to MOOP with considerable advances being reported in various fields [ZLB04], [CS04], [NBP07]. The popularity of MOEAs is probably due to some inherent algorithmic characteristics. Namely, the population-based approach enables the simultaneous search of multiple search space regions and thus the identification of numerous Pareto solutions in a single run. Additionally, EAs impose no constrains on the morphology of the search space and are therefore suitable for complex, multi-modal surfaces such as the ones typically produced by MOOP problems. Algorithmically, MOEAs are an extension of traditional EAs that can address multiple objectives simultaneously by the addition of appropriate components such as Pareto-based selection that incorporates fitness assessment on multiple objectives, calculation of domination relations and Pareto-rank and definition of a scalar efficiency value for each solution, and the techniques of niching and elitism aiming to maintain population diversity and avoid good solution loss [CLV07]. Figure 3.4 outlines the main steps of a simple MOEA algorithm. Below we discuss the key features of MOEAs distinguishing them from normal, single-objective EAs.

```
Generate initial population P
Evaluate solutions in P against objectives $O_{1-n}$
Assign Pareto-rank to solutions
Assign efficiency value to solutions based on Pareto-rank
While Not Stop Condition:
        Select parents $P_{parents}$ in proportion to efficiency values
        Generate population $P_{offspring}$ by reproduction of $P_{parents}$
                Mutation on individual parents
                Crossover on pairs of parents
        Evaluate solutions in $P_{offspring}$ against objectives $O_{1-n}$
        Merge P, $P_{offspring}$ to create $P_{new}$
        Assign Pareto-rank to solutions
        Assign efficiency value to solutions based on Pareto-rank
```

Fig. 3.4: A typical MOEA algorithm.

3.2.1 Pareto-based selection

Selection in MOEAs involves the generation of the objective vector of each individual and the use of that set of vectors for the selection of solutions for use in subsequent steps of the algorithm. The objective vectors are obtained by assessing the fitness of each individual to all objective functions available for a specific problem. Selection requires the establishment of a comparison mechanism among objective vectors and the identification of the subset to be used in next steps. According to Zitzler (1999), there are three main approaches to selection based in MOEAs: (a) selection by aggregation of objective values, where fitness scores are combined into a single composite one, (b) selection by switching objectives, where the choice of each individual is based on a single objective function and, (c) Pareto-based selection where objective vectors are used to establish domination relations between all pairs of individuals, followed by the application of a technique for Pareto-ranking the individuals based on their domination relations and selection based on the rank [Z99]. A number of Pareto-based selection variations have been proposed that differ mainly in the technique used to determine the Pareto-rank from the domination relations and, in the definition of a scalar solution efficiency score from the Pareto-rank. Some of the most prominent of these techniques are described below.

3.2.2 Niching

Niching aims to preserve solution diversity during the optimization process to enable the concurrent search of multiple regions of the search space and the identification of solutions representative of the Pareto-front in a single run. The need for niching is related to the problem of genetic drift [DJ75] where the population converges towards a less fit region from multiple, similarly fit sampled regions through stochastic sampling effects in the sampling procedure [KJ02]. Although existing, the problem is not as obvious in single-objective settings where the goal is to identify the single best solution. In MOPs the lack of diversity in the population leads to a limited range of solutions and, in the worst of cases, dictatorship conditions where solutions from a single local optimum dominate the Pareto approximation set. Among the many techniques proposed for niching fitness sharing, crowding and clustering deserve special mention:

a. **Fitness sharing**, the most commonly used niching technique in the MOEA field, is based on identifying neighborhoods of solutions and reducing the efficiency of the solutions from more dense neighborhoods. Neighborhoods are defined using a distance measure $d(\vec{u},\vec{v})$ and a neighborhood radius σ_{share} provided by the user (Fig. 3.5).

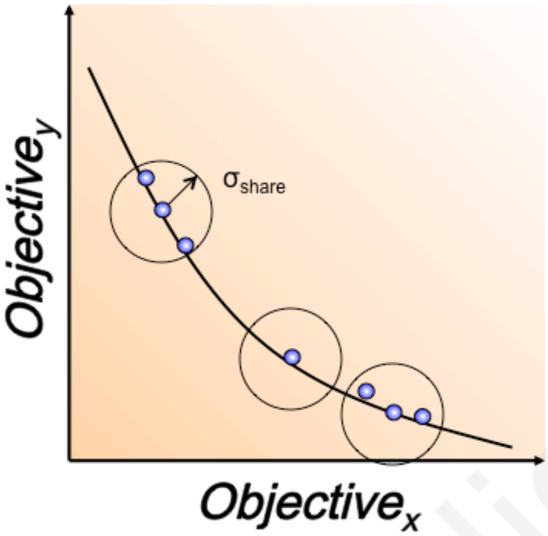


Fig. 3.5: Fitness sharing adjusts solution efficiency so that solutions from a scarcely populated neighborhood have increased chance of being selected. The method uses the variable σ_{share} to define the solution neighborhoods. Fitness sharing can take place in either objective or decision space with the former being overwhelmingly preferred.

The underlying assumption behind fitness sharing is that solutions within the same neighborhood are more similar and therefore the inclusion of multiple representatives from a neighborhood does not contribute to the diversity of the working population. It follows that isolated solutions, i.e., solutions from less dense regions of the Pareto-front should be favored during the selection process. In practice, the fitness of a solution *s* is adjusted by dividing with its niche count *NC(s)* defined as the sum of the values of the function *share* between *s* and all members of the population P. Mathematically:

$$NC(s) = \sum_{i \in P} share(d(s,i)) \qquad \text{Eq. (9)}$$

where the function *share* typically has values ranging between zero and one with solutions having a distance to *s* larger than the σ_{share} radius producing a zero value and solutions closer to *s* producing values approaching to one. Fitness sharing may be applied in genotypic (decision) space to ensure diversity of the genotypes or, phenotypic (objective) space to maintain diversity among the nondominated solutions with respect to their objective vectors.

b. **Crowding**, in which fit offspring added to the working population replace similar existing solutions from the current population. The technique has similar end results as fitness sharing since new-coming offspring members of a neighborhood of solutions remove existing members and therefore the neighborhoods are prevented from growing at the expense of diversity. However, and despite being one of the oldest techniques introduced for niching, crowding has found limited applications in the MOEA field [Z99].

c. **Clustering** is also founded on the assumption that dense neighborhoods of solutions, are not beneficial to the optimization process. The method uses a clustering method to define natural groups of solutions, the clusters. In a following step the density of clusters is taken into account when selecting parents for reproduction by favoring candidates representing the entire collection of clusters.

3.2.3 Elitism

In the EA and MOEA field, elitism refers to the set of techniques used to ensure the presence and continued influence of good solutions found in earlier generations throughout the search process. The technique has been proposed as early as 1975 [DJ75] and has been applied in a variety of forms. A simple implementation of elitism provides for the selection of the best individuals from the parent population and their inclusion into the current population. Alternatively, the parent population may be merged with the offspring and the new population may be selected from this enlarged set thereby allowing good solutions to survive in later generations.

Implementation of elitism in modern MOEAs also takes place through a dedicated data structure, the Pareto-archive, which contains a second set of solutions in addition to the working population. This archive is used to store the best nondominated solutions found during the algorithm's run. During execution time, upon discovery of a new Pareto approximation set, the algorithm updates the archive by (a) adding new nondominated solutions to it, (b) Pareto-ranking the extended set of solutions, and, (c) removing the dominated ones. Depending on the specific problem and algorithm implementation the archive may grow continuously and become difficult, and time consuming, to maintain. To cope with this problem the technique of archive bounding has been proposed where an upper limit to the size of the archive is imposed and a solution pruning procedure, often based on diversity analysis, is used to limit the number of nondominated solutions in the archive.

The introduction of elitism in the form of an external archive of solutions has been the most important innovation in the MOEA field in recent years [KJ02]. In addition to preserving good solutions from getting lost due to sampling effects the Pareto-archive can also accommodate sets of nondominated solutions exceeding the working population size, a feature not possible using ordinary EA data structures.

3.2.4 MOEA literature review

The Multiple-Objective Genetic Algorithm (MOGA) algorithm [FF98] was one of the first MOEAs to be proposed. In MOGA, the efficiency calculation of a solution is inversely proportional to its Pareto-rank, i.e., the lower the rank, the higher the efficiency score and therefore the higher the probability to be selected for reproduction. In this manner, nondominated solutions which have the lowest possible rank (set to rank 1) have the highest chance to become parents and generate offspring. Niching, in the form of a fitness sharing technique, was also introduced to maintain population diversity by adjusting efficiency values appropriately.

The downside of MOGA is that it can introduce a bias towards certain solutions in the search space due to the nature of its rank-based fitness assignment method and thereby allow solutions with substantially better performance at an iteration to dominate the population of later generations. Furthermore, setting the value of σ_{share}, a task typically left to the user, can have dire effects on the efficiency scores produced with larger values of σ_{share} resulting in extended neighborhoods that may fail to map the Pareto-front accurately, and smaller values running the risk of allowing solutions with a worse rank to have better fitness than solutions with a better rank [DK01].

The Nondominated Sorting Genetic Algorithm (NSGA) [SD94], [DAPM00] modifies the Pareto-ranking and the efficiency calculation step of the algorithm using the nondominated sorting concept. In NSGA the population is classified into layers, or waves, of nondominated sets. The process successively defines the nondominated set of the population, removes its members from the current population and iterates until all solutions have been taken into account. Fitness sharing and solution sampling are performed at the nondominated layer level starting from the globally nondominated solution level. Fitness values of solutions in successive layers are reduced to be less than the worst fitness value of the previous layer. In the original version of NSGA, selection is performed using a stochastic-remainder wheel-like operator while in an updated elitist version, named NSGA-II, selection is performed by choosing the best solutions from a population combining both parents and offspring. The NSGA-II conducts niching through the use of a crowding distance calculated for each solution, used to maintain population diversity during selection by ensuring that selected solutions are sufficiently apart. This keeps the population diverse and helps the algorithm to explore the fitness landscape [CLV07]. The NSGA-II algorithm successfully addresses some of the shortcomings of MOGA and succeeds in preserving the diversity of the population although the method is still sensitive to the choice of σ_{share}. The Niched Pareto Genetic Algorithm (NPGA) method [HNG94] is a further extension of the NSGA where the selection step is based on a modified tournament-based method that uses a larger subset of the population and shared efficiency values of the individuals.

The Strength Pareto Evolutionary Algorithm (SPEA) presented by Zitzler [ZLB04] introduced elitism in the form of an external population archive to ensure that nondominated solutions are not lost because of population size limitations or sampling effects. A clustering process is used to reduce the number of solutions stored in the archive by removing some of the solutions from the same cluster without loosing the characteristics of the trade-off front [Z99]. Solutions from the external archive also participate in the selection process since, for example, the pairs of individuals for the crossover operation are formed by one solution from each of the two populations. In SPEA2 archive bounding is introduced to limit the size of the archive within manageable numbers. SPEA also introduced an alternative fitness assignment technique that combines dominance rank with dominance count, i.e., the number of individuals dominated by a solution [CLV07]. According to this method the fitness of a population member is determined only from the dominance relation of individuals stored in the external set to individuals in the working population; domination relations between members of the working population is not taken into account.

MOEA is an active field of research and new developments are reported frequently. Several developments in selection and niching techniques as well as elitism have contributed to the development of more efficient algorithms able to address multi-objective problems. Ongoing research aims to build on the latest findings to create improved versions of the algorithms that converge quicker and map the entire Pareto front in a single run. Zitzler et.al. [ZLB04] have prepared an excellent overview of the field. The interested reader is also referred to Collette and Siarry [CS04] as well as Coello [C99] for a concise, although dated, introduction to the field.

3.3 Memetic Algorithms

Memetic Algorithms (MA) is a population-based approach for heuristic search in optimization problems closely related to EAs [M89]. In general, MAs may be described as EAs which include a stage of individual optimisation or learning usually in the form of local search [K02]. Due to their relation to EAs they are also known as Genetic Local Search, Hybrid Genetic Algorithms and Parallel Genetic Algorithms. The name has been inspired by the Dawkins "Meme" theory [D76]. A "meme" stands for "unit of imitation" in cultural transmission and the methodology is named memetic due to its analogy to cultural instead of biological evolution. The first use of the term Memetic Algorithms in the computing literature has appeared in 1989 in [M89].

The local search technique employed can be a simple hill-climbing method, a more sophisticated method such as simulated annealing or tabu-search [K02] or even one or more problem-specific heuristics [KN04], [S07]. The overall strategy reflects the fact that although good performance is commonly achieved by using the most appropriate general purpose optimization algorithm, much greater gains can often be made by combining it with heuristics or operators that incorporate 'domain knowledge' [KC04].

In several cases MAs have been proven to be orders of magnitude faster than traditional EAs [M04]. Due to its improved performance the method is gaining wide acceptance, in particular in well-known combinatorial optimization problems where large instances have been solved to optimality and where other metaheuristics have failed. A typical outline of MAs is presented in Fig. 3.6.

```
Generate initial population P
Evaluate solutions in P against objective O
While Not Stop Condition:
        Select parents $P_{parents}$ in proportion to fitness scores
        Generate population $P_{offspring}$ by reproduction of $P_{parents}$
                Mutation on individual parents
                Crossover on pairs of parents
                Select $P_{improve}$ subset:
                        Improve via local search
        Evaluate solutions in $P_{offspring}$ against objective O
        Select population $P_{new}$ from P and/or $P_{offspring}$
```

Fig. 3.6: The general framework of Memetic Algorithms.

More recently newer generations of memetic algorithms have been reported in the literature. Multi-meme algorithms provide a range of memes (i.e., local searchers), which compete in each generation for selection and application [CHKB02]. Other approaches seek to produce a metaheuristic that creates from scratch the appropriate local searcher to use under different circumstances [KN04]. Coevolving memetic algorithms [S07] are a family of metaheuristic search algorithms in which a rule-based representation of local search is co-adapted alongside candidate solutions within a hybrid evolutionary system. Simple versions of these systems have been shown to outperform other non-adaptive memetic and evolutionary algorithms on a range of problems [M04].

The MA field is still new, dynamic and rapidly evolving, and several open research issues remain. A list of typical issues to address when designing an MA include [K02]:

- when to use local search; may be applied before or after mutation or crossover
- which individuals to improve using local search; may be applied on the parents, the offspring individuals or a combination of the two.
- at what intensity to apply local search

- how to integrate genetic operators with local search
- how to escape local minima
- how to avoid premature convergence.

The algorithm presented in the next chapters belongs to the MA family. The method strives to incorporate problem-domain knowledge in numerous ways and combines global search with knowledge-driven local search methods. Moreover, the genes used contain problem-specific information potentially useful in the course of optimization and thereby exhibiting more similarity to the "meme", as defined by Dawkins, than commonly used MA methods.

Chapter 4

Research Question and Related Work

The main motivation for the work presented in this dissertation has been to design and implement a new algorithm for the Optimal Graph Design problem. OGD is a multi-objective combinatorial (MOCO) problem with several instantiations in the real world, ranging from transport and water distribution network design to wireless antenna positioning in telecommunications and the design of molecules with desired biological properties in the pharmaceutical industry. We have chosen to use a hybrid approach, combining evolutionary principles with a strong memetic component so as to support the optimization process with available problem specific knowledge. Due to the overall approach followed, but also to the interdisciplinary nature of the OGD problem, our work required research in multiple directions including chromosome representation in EAs, niching and elitism in MOEAs, and knowledge encoding and reuse in a memetic approach. Following a brief section providing the necessary background information in graph theory this chapter reviews previous work in graph design in general (section 4.2) and small molecular graph design (section 4.2.1) which will be the area of application of the proposed algorithm. The specific research questions that the present thesis attempts to answer are outlined in section 4.3.

4.1 Graph Theory Fundamentals

A graph G = (V, E) consists of a set of vertices V(G) and a set of edges E(G) [G88]. In the case of labeled graphs both vertices and edges have identifiers, i.e., each vertex and edge has a label drawn from a predefined set of vertex labels L_V and edge labels L_E. Note that vertices and edges need not have unique labels, as is the case in molecular graphs where, for example, multiple vertices in any drug-like molecule have the C (carbon) label. Graphs can be directed or undirected. In directed graphs edges are ordered pairs of the vertices they connect where, in undirected, edges simply list the pair of vertices they connect. A vertex V_I is said to be incident with an edge if one of the two endpoints of the edge is V_I while an edge E_a is incident with an edge E_b if they have a vertex in common. Two vertices V_I, V_J of graph G are connected, or adjacent, if there is an edge $E_{IJ} = (V_I, V_J) \in E(G)$. If there is a path P = (E_1, E_2,…, E_n) between every pair of vertices in a graph G, then G is a connected graph [G88].

A graph S = (Vs, Es) is a subgraph of G = (V, E) if and only if $V_S \in V$ and $E_S \in E$. If E_S contains all edges in E connecting the vertices in V_S then S is an induced subgraph of G. An additional property of induced subgraphs is that it can be shown that there is a one-to-one mapping between the edges in E_S and all edges in E incident on vertices in V_S when V_S is mapped on V. A clique is a special case of an induced subgraph where all its vertices are incident on each other. A maximum clique of a graph G is its largest clique.

The problem of determining whether two graphs are identical is known as graph isomorphism [LG03]. In graph theoretic terms two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there is a mapping from V_1 to V_2 such that there exists a mapping for each edge in E_1 to an edge in E_2. A common induced subgraph between G_1 and G_2 is a graph CS that is an induced subgraph of both G_1 and G_2. The largest induced subgraph between G_1 and G_2 is known as the Maximum Common Induced

Subgraph (MCIS). A related concept is that of Maximum Common Edge Subgraph (MCES) also known as Maximum Overlapping Set (MOS). An MCES is a subgraph consisting of the largest number of edges common to both G_1 and G_2 [RGW02]. It is worth pointing out that the MCIS and MCES between two graphs may consist of several disconnected subgraphs as seen in Fig. 4.1. The largest contiguous common substructure is known as the Maximum Common Substructure (MCS). Informally, the MCS of two graphs G_1 and G_2 is the largest possible graph that is isomorphic to subgraphs of G_1 and G_2.

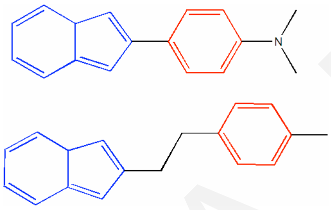


Fig. 4.1: A pair of molecular graphs and their corresponding MCS (in blue) and MOS (in blue and red).

Several algorithms have been proposed in the literature dealing with the problem of graph isomorphism. Among them the category of clique finding algorithms is one of the most popular [RGW02], [LG03]. These methods rely on the calculation of a *new graph*, the compatibility graph CG, via a modular product operation on graphs G_1 and G_2. Following is the determination of the maximum clique in the CG by applying one of the many maximum clique algorithms available. The maximum clique of the CG has been shown to be equivalent to the MCIS of the two input graphs [RGW02b]. The modular product of the graphs G_1 and G_2, denoted as $G_1 \lozenge G_2$, is defined as:

$$V(G_1 \lozenge G_2) = V(G_1) x V(G_2) \qquad \text{Eq. (10)}$$

where two vertices (u_i, v_i) and (u_j, v_j) are adjacent if:

$$(u_i, v_i) \in E(G_1) \text{ and } (u_j, v_j) \in E(G_2) \qquad \text{Eq. (11)}$$

or

$$(u_i, v_i) \,!\!\in E(G_1) \text{ and } (u_j, v_j) \,!\!\in E(G_2) \qquad \text{Eq. (12)}$$

Note that the vertices of CG consist of pairs of vertices, one vertex from each input graph. For a more detailed explanation please look at [RGW02], [RGW02b].

Graph data structures may be used in problems that can be represented naturally as sets of vertices related through edges. For example, graphs may be used to represent solutions to the TSP problem with cities as vertices and paths connecting the cities as edges. Similarly, water distribution network designs may be represented as graphs using edges for water pipes and channels and vertices for reservoirs, desalination plants, pumping stations etc. Chemical structures can also be represented as labeled, undirected graphs where atoms correspond to vertices and chemical bonds are represented by edges. In this context, molecular fragments, or substructures, are induced subgraphs of molecular graphs. Interestingly, the latter have been used as units for knowledge encoding and transfer through the development of the privileged substructure and chemical scaffold concepts which relate chemical structure with certain biological characteristics, often increased potency towards a specific pharmaceutical target [NP06].

4.2 Optimal Graph Design Review

There exist a large number of references in the literature describing OGD related algorithms and applications. The applications come from a wide variety of problem domains that can best be conceptualized as graphs and, therefore, the use of a graph representation facilitates meaningful algorithmic operations. In effect, the use of the graph structure imposes a geography on the solutions and thereby constrains the solution space, something that can lead to the improvement of the overall algorithm performance [BACW06]. However, due to the combinatorial nature of a MOCO problem the solution space to be searched can grow to sizes impossible to exhaustively enumerate. Optimization approaches have been the methods of choice since they are quite appropriate to this type of problems, especially those capable of handling large, complex, multimodal search spaces with minimal information on the underlying problem specifics.

The general objective in OGD applications is to design from scratch, or refine from a given initial graph, the optimal graph(s) satisfying the constraints imposed on the problem. The methodologies used to solve the OGD problem need to also take into account issues related to the graph representation of the solutions. Such issues include the encoding of the graphs using appropriate data structures, the generation of valid -according to the specific problem- graphs, the construction and preservation of certain geometries and the accommodation of special topological features related to the problem. These graph-structure specific issues coupled with the multi-objective nature of applied OGD form a challenging problem for any optimization method. In the following sections we initially present briefly some representative OGD algorithms from several problem domains and then review in detail the field of de novo design on which our proposed algorithm is tested.

Emmerich *et al.* (2001) introduced a graph-based evolutionary algorithm for the optimization of chemical processes [EMS01]. The algorithm, based on the Evolution Strategies methodology, represents chemical plants as parameterized networks (i.e., graphs) and uses custom genetic operators working on both, the structure and the parameters of this graph. The method uses problem specific structure and parameter mutation operators and, a simple recombination operator that selects similar subgraphs from two parents to exchange. The algorithm has been applied successfully to the optimization of chemical plants and the optimization of feed water strings of thermal power plants [HHMS00].

Several graph based evolutionary algorithms have been described in the literature for the general graph drawing problem that aims to identify the “best” way to draw a given graph. Typically, in the graph drawing problem, the number of vertices and edges are fixed but the placement of the vertices in space is not. Solutions to the problem need to satisfy aesthetic criteria to facilitate readability i. e. the capability to convey the meaning of the diagram quickly and clearly to the user [KT98]. The algorithm proposed by Utech *et al.* [UBSE98] primarily aims at minimizing edge crossings. Chromosomes contain information about the graph vertices which is modified through simple crossover operations and two types of mutation, a standard random-based mutation referred to as “shake” and, a more elaborate mutation step that incorporates the knowledge of a well-known problem-specific heuristic. The algorithm was compared extensively with popular graph-drawing heuristics on a number of problem cases and was found to perform significantly better. Kobler and Tettamanzi [KT98] describe a similar algorithm for the general graph drawing problem that manages several constraints. Their implementation combines the several aesthetic objectives considered into a single one. Individuals are represented using vectors of the form:

$$((x_1, y_1), (x_2, y_2), \ldots, (x_n,y_n)) \qquad \text{Eq. (13)}$$

where n is the number of nodes and each tuple x_i, y_i corresponds to the coordinates of the vertex *i* encoded using integer numbers. The algorithm uses mutation and a rich set of crossover operations to evolve solutions. The population is distributed in several subpopulations evolved independently with good solutions migrating between them on a regular basis. Elitism is also implemented to ensure that the best individual of the current population survives unchanged. Rosete and Ochoa [RO98] presented their version of a multi-objective genetic graph algorithm for graph drawing where coordinates are encoded using bit vectors and mutation and crossover use standard bit-based operations. The proposed method was tested on several planar and non-planar graphs of varying complexity that showed its wide range of potential applications in the automatic graph drawing field. Vrajitoru (2007) describes an application designed to build consistent graph layouts using a hybrid multi-objective optimization genetic algorithm approach [V07]. The method employs a linear aggregate objective function with equal coefficients for each of the measures pursued. Solutions were encoded by real-valued vectors where each vertex is represented by its coordinates in 3D space and evolved using crossover and a hybrid, problem-specific mutation operator. The application was tested on the design of Platonic solids and was found to outperform standard methods used for the same problem [V09].

In a series of publications, Bryden, Ashlock *et al*. present a different approach using a graph to represent the entire population [BACW06] with the goal to constrain evolutionary operations as required by the problem and ensure the diversity of the population. In this representation, evolution can be restricted to neighboring solutions in an effort to mimic an analog of the biological refuge found in nature. A local mating rule that operates similar to roulette selection but only within the bounds of a well-defined neighborhood is used for selection. The experimental results provided on a large variety of problems showed that the graph-based algorithm performs significantly better than normal genetic algorithms on identical problems [ABC05], [BACW06].

Mabu *et al.* describe Genetic Network Programming (GNP) for evolving software agents active in dynamic environments [MHH07]. The method was intended as an extension of GP exploiting the higher expression ability of graph structures as opposed to that of trees. GNP individuals consist of a directed graph containing a fixed number of nodes of different types (start, judgement, processing) that can perform different actions, connected by a number of directed edges. Chromosomes represent graphs with a two-segment data structure; the first section contains information about the nodes of the chromosome and the second the connectivity information. Evolution occurs through mutation and crossover while selection follows an elitist approach where the best individuals are preserved in next generations. A hybrid version of GNP, complemented with Reinforcement Learning (RL), aimed at combining the advantages of the evolutionary approach, i.e., the ability to perform a global search, with the intensified, local search provided by RL through the immediate rewards obtained after agent actions. Experimental results provided by the authors show that GNP compares favourably with GP, with the RL enhanced version further improving the quality of the solutions [MHH07].

In the engineering domain, Farmani *et al*. [FSW05] presented a multi-objective optimization evolutionary application for the design of a water distribution network. Their work involved the implementation of popular MOOP algorithms, i.e., SPEA and NSGA, and the optimization of several test cases including the New York tunnel system and the Hanoi distribution network. The objective used was a non-linear aggregate function combining water demand, construction cost and overall length goals. The design task was simplified since vertices were known and fixed and therefore only graph edges could be modified by adjusting for example, the number of pipes, the diameter of each pipe, etc. Individual designs were encoded as bit strings with standard single-point crossover and bit-flip mutation used for evolution. In their conclusions, the authors suggested that the application of such methods produced several potential solutions although the solution sets were incomplete [FSW05].

A similar algorithm was described in [CS04] for the design of the extension of a telecommunication network. In this application the nodes of the graph were also known and so the problem consisted of determining the optimal combination of edge capacities so as to minimize the cost and maximize the reliability of the network. Solutions were encoded using vectors of 15 elements each of which could take one of four different values. The method used a Pareto archive updated in each iteration. Solution breeding took place using high level processes designed for local optimization and diversification of the solution population, combining multiple steps of the mutation and crossover operations. Selection was performed using an efficiency score calculated according to the MOGA process described previously. The overall method was tested on a real scenario and successfully produced a Pareto approximation set consisting of a diverse set compromise solutions.

In the following section a more detailed review of the area of small molecule design is given. This specific field has long attracted significant interest from the optimization community and has provided the motivation and test-bed for several innovations in the field. A brief introductory description of the problem is also given to facilitate user understanding.

4.2.1 De Novo Design

Drug discovery focuses on identifying biological keys, i.e., molecules that interact with specific biological receptors and cause a certain desired behaviour. The ability to interact is controlled by the molecular graph of the drug and namely, by its complementarity to the targeted, receptor site. More specifically, to achieve appropriate binding a molecule must have the right pharmacodynamic properties, i.e., complementary shape, size and electrostatic properties to the receptor site [MG04]. However, not all potent binding molecules are suitable as drugs. In order to be truly effective within a living organism a molecule must satisfy several additional properties, e.g. Absorption-Distribution-Metabolism-Excretion (ADME), collectively known as pharmacokinetics and, toxicity (Tox). These properties enable drugs to move appropriately “in vivo” (i.e., in the living organism to be treated), reach the region of the target without causing side effects and bind selectively to the pharmaceutical target [BM04]. The presence of the pharmacokinetics and toxicity requirements turn drug discovery into a true multi-objective problem where a solution can only be found if multiple objectives are concurrently fulfilled. From an optimization perspective drug discovery can be thought of as an optimal graph design problem subject to sufficient binding and appropriate pharmacokinetics and toxicity properties.

De-novo drug (or ligand) design (DND) is an attempt to generate ligands from scratch based only on information about the pharmaceutical target site or known ligands [NAP09]. Effectively, DND methods face the task of exploring a chemical search space estimated to be in the order of 10^{60} [BMG96]. Such space cannot possibly be fully enumerated and so powerful search methods need to be applied to detect the best possible solutions in a limited amount of time.

DND algorithms proposed in the literature typically use an evolutionary algorithm related technique for searching and a set of molecular fragments as genes. The use of a predefined collection of molecular fragments is sometimes combined with the identification of reaction points and synthetic rules that, when used, increase the synthetic feasibility potential of the designed chemical structures. Most methods are designed to accommodate a single objective, either predicted binding affinity to a known protein target or, similarity to a known ligand. MOOP-based methodologies are limited to [NAP09] and [BMGG04] although the need for their greater adoption is gaining support within the drug discovery community [ES02], [BM04], [NBP07].

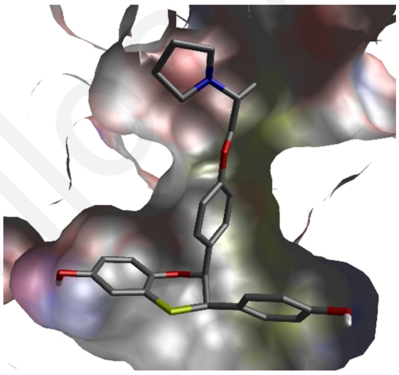


Fig. 4.2: A crystallographic image of the receptor of the ER-alpha protein, with a bound ligand (1ert). In the lock and key paradigm the protein is the lock, the receptor is the keyhole and the key is the drug/ligand.

Excellent reviews on the topic can be found in [SHRTPS08], [SF05] and [CW96] for the older methodologies. A review of the use of EAs in drug design in general can be found in [LBKI05]. In the following section we focus on EA related approaches that seem to be the overwhelmingly preferred method in recent years. Table 1 presents a comprehensive list of published DND methods using EA related approaches. The subsequent discussion section elaborates on these methods and, highlights commonalities, shortcomings and potential improvement research directions.

TABLE 4.1: A SUMMARIZATION OF EA-BASED DND METHODS

Name/Year	Genotype Encoding					Fitness Function	Description
	Genes		Chromosomes				
	Atoms	Fragments	String/linear	Tree	Graph		
“Chemical Genesis” Glen & Payne (1995)		✓ (3D fragments)			✓	Receptor (fit to predefined grid) and ligand-based (molecular properties)	Uses mutation (extensively) and constrained crossover; crossover limited not to break rings; multi-objective (composite)
"PRO_Ligand" Westhead (1995)		✓ (3D fragments)			✓	Ligand-based; based on superposition of functional groups	Uses mutation and crossover; crossover limited to single-point, does not brake rings; mutation limited to single-point; single-objective
Weininger, Daylight (1995)	✓			✓ (SMILES)	✓	Ligand-based; Descriptor-based molecular similarity; mentions possibility of other	Uses mutation and crossover; single-objective; based on SMILES; could produce disconnected fragments
Venkatasubramanian (1995)		✓		✓		Ligand-based; Similarity to target molecule	Uses mutation and crossover; reduces cycles to single nodes; multi-objective (composite)
Nachbar (1998)		✓		✓		Ligand-based; QSAR model prediction	Uses mutation and a constrained crossover version; molecules as rooted trees; problems with cycles, special ring nodes; single-objective
Patel (1998)		✓	✓			Ligand-based; QSAR/NN model prediction	Uses mutation and crossover; used simpler structures/peptides; single-objective
Globus (1999)	✓				✓	Ligand-based; Descriptor-based molecular similarity	Introduced genetic graphs and crossover rings; evolution only through crossover
Nachbar (2000)	✓			✓		Ligand-based; Descriptor-based molecular similarity	Uses mutation and constrained crossover; molecules as rooted trees; special ring nodes sophisticated mutation to alter rings; single-objective;
“LEA” Douguet (2000)		✓		✓ (SMILES)		Ligand-based; QSAR model prediction	Uses mutation and crossover; multi-objective (composite)
“LigBuilder” Wang & Gao (2000)		✓			✓	Receptor (empirical scoring) and ligand based (molecular properties)	Given receptor site and "seed" structure constructs ligands from fragments; sturcture-based, "grow", "link" and "mutate" processes; multi-objective(composite)

“TOPAS” Schneider (2000)		✓	✓			Ligand-based; Descriptor-based molecular similarity	Evolution strategy; Taking template structure, uses mutation guided by chemical reaction; single best parent, numerous offspring; single objective
“ADAPT” Pegg & Haresco, (2001)		✓		✓ (SMILES)		Receptor-based; docking and scoring	Uses mutation and crossover; single-objective
Kamphausen (2002)		✓	✓ (varying size)			Ligand-based (molecular structure); actual wet-lab screening	Uses mutation and an elaborate version of crossover; used simpler structures/peptides; single-objective
“Synopsis” Vinkers (2003)		✓ (whole molecules)			✓ (molecules)	Two examples: calculated property value (ligand-based) and docking score (receptor-based)	Operates on numerous input molecules; randomly selects one and uses mutation guided by chemical reaction; single mutant produced; iterates; single objective
“CoG” Brown (2004), (2006)	✓	✓			✓	Ligand-based; Descriptor-based molecular similarity; QSAR model prediction	Uses mutation and crossover; multi-objective (pareto)
“LEA3D”, Douguet (2005)		✓ (3D fragments)	✓			Ligand and Receptor based	Operates on linear combination of fragments; uses mutation and crossover; multi-objective (composite)
“Flux” Fechner (2006)		✓	✓ (varying size)			Ligand-based; Descriptor-based molecular similarity;	Evolutionary strategy; uses mutation and single best parent; single-objective
“Molecular Evoluator” Lameijer (2006)	✓	✓ (not used)		✓ (SMILES)		Ligand-based; Expert user; interactive	Uses mutation (mostly) and crossover; single-objective
“GANDI” Dey (2008)		✓ (3D fragments)		✓		Receptor-based; docking and scoring	Restricted search by receptor site; combines evolutionary algorithm with tabu search; multi-objective (composite)
“MOBIUS” Ecemis (2008)		✓		✓		Flexible; accommodates both Ligand and Receptor-based; Expert user; interactive	Restricted search by a chemical graph template (blueprint); multi-objective (composite)
"MEGA" Nicolaou (2009)	✓	✓			✓	Flexible; accommodates both Ligand and Receptor-based	Evolutionary graphs; multi-objective (Pareto); uses mutation and crossover; niching (genotype and phenotype) and elitism; accommodates/exploits problem knowledge

Pro_Ligand reported by Clark *et al*. [CF95] uses a fragment-based approach and a DFS method to incrementally design de novo ligands fitting a model derived from a well-defined target receptor site or a collection of highly similar actives. The method constructs the ligands by matching fragments with the model components and may "link" or "grow" virtual molecules using standard chemical rules. The DFS strategy, chosen primarily for performance reasons resulted inevitably in the generation of structures with varying degrees of quality. In later publications Pro_Ligand was complemented by a post-processing GA-driven module that further evolved the designed compounds using a limited set of crossover and mutation operations and roulette selection type [WC95].

Glen and Payne proposed their Chemical Genesis system in 1995 [GP95]. Their program used 3D molecular fragments to design molecules using a single-objective evolutionary algorithm. The method clearly recognized the need to accommodate multiple objectives and therefore used a composite objective function, combining both receptor and ligand-based objectives. Chromosomes were represented via a molecular graph structure with additional bits to indicate various topological and geometrical features. Structure evolution took place by directly operating on the chemical graph via mutation and crossover with a preference on the former. Crossover involved exchanging fragments between two molecules taking care not to break rings, while mutation, with 12 different variants, allows modifications including changing atom and bond types, inserting and removing fragments and breaking and forming rings.

A sizeable category of EA-driven applications proposed has been using linear or simple tree representations of molecules. Most of these approaches opt to use molecular fragments as genes, a choice that in effect reduces complexity by sidestepping the issue of ring handling. Under these conditions perturbation of the chemical structure is achieved by actions such addition, deletion, substitution or exchange of whole fragments. Venkatasubramanian *et al*. [VCC94] were among the first ones to use such an approach and test it successfully to design polymer structures. Similar approaches have been reported in [KH02] and [DML05]. The latter used a pool of 3D fragments, which were combined in a linear fashion in a chromosome string. The method used a composite fitness function taking into account both receptor and ligand-based constraints. The application TOPAS [SLS00] used 2D subgraph genes derived from known drug molecules and an EA method to design molecules similar to a target chemical structure. TOPAS, and its ancestor Flux [FS05], [FS07], used retrosynthetic analysis [LBWH98] to generate the fragment genes and kept information about the type of bonds at each attachment point. Candidate compounds were then evolved via operations that take into account chemical synthetic rules. TOPAS used exclusively mutation in the form of fragment substitution. Flux [FS05] uses a richer collection of genetic operators that enables recombination of parent molecules via crossover, and changes in the number of fragments a molecule contains. In both algorithms the fitness function was based on the chemical similarity of the candidate compounds to a known active molecule calculated by first transforming the molecules into descriptor vectors and applying a vector distance/similarity measure. Certain drug-likeness rules were taken into account for compound selection. GANDI [DC08] joins a collection of predocked 3D fragments to a receptor site with a set of linkers to generate candidate molecules. Individuals are represented as simple trees whose general shape and structure is restricted by the receptor site present. The method divides the working population into subpopulations and uses a parallel genetic algorithm (see section 5.2) with binary tournament selection of parents for selecting the predocked fragments and tabu-search [W08] to select the linkers. The scoring function in GANDI is a linear combination of terms measuring both 2D and 3D properties of an individual.

Nachbar [N98], [N00] proposed a ligand-based DND algorithm that uses a tree-like structure for molecule representation. Rings were represented using special pointer nodes to link appropriate tree branches. The method uses a single objective evolutionary approach employing both mutation and a constrained crossover version paying special attention not to make or break rings. The objective used to drive the process was a predictive regression model generated using a set of known actives.

A different category of DND methods exploit the simplicity of the SMILES chemical language to represent molecules [W89]. Weininger (1995), the inventor of SMILES, proposed an evolutionary method that used single atoms and bonds as genes to design molecules satisfying any single given objective function [W95]. The method used both mutation and crossover operators and was shown to be successful in test cases where the objective function was similarity to a known ligand. Douguet et. al [DTG00] also used SMILES for molecular representation and an EA-based search strategy but chose fragments for genes. Their method used an extensive set of mutation and crossover operators, and a set of repair mechanisms to ensure the validity of the produced SMILES strings, especially with regard to branching and ring correctness.

Globus [G99] introduced an evolutionary algorithm designed to evolve molecular graphs. The individuals used are labeled, cyclic graphs able to accommodate vertices and edges of various types. In this application evolution is taking place using crossover through a process of initially “ripping” individuals into two parts and then combining the parts from the different parents. Ripping selects and removes randomly an edge and adds on the two resulting fragments “cut bonds”. If this operation fails to break the graph in two, i.e., when the edge is part of one or more cycles, the algorithm iterates until there is no path connecting the two vertices of the original edge. During mating one of the cut bonds is picked randomly and merged with a cut bond edge belonging to a fragment from a different parent. The merger results in connecting the two graph fragments from different parents into a new offspring graph. This approach is capable of crossing over rings a feature not available in most other algorithms reviewed for the purposes of this report.

Tournament selection is used for choosing parents and replacing poor individuals. The algorithm implements a steady-state genetic system where new individuals replace existing poor individuals in the population rather than creating a new generation [G99]. Globus applied the technique to evolve individuals similar to a specific target molecule. The fitness function used was simply graph similarity to the target molecule. In their tests the authors were able to reproduce several target molecules of varying complexity.

Multi-objective optimization technology has been introduced to de novo design through the system proposed by Brown *et al.* that optimizes candidate drugs using as objective functions similarity to existing molecules of interest [BMGG04]. The system, named COG, explores the chemistry space of interest via perturbations of a genetic graph molecular representation. COG, is flexible enough to accommodate both molecular fragments and atoms/bonds as genes and, imposes no constraint on the size or complexity other than those required for the graph to represent a valid molecule. The set of genetic operators available includes an extended set of mutations on both nodes and edges, and crossover that enables crossing over rings. The multi-objective fitness score mechanism implemented in COG is based on the Pareto ranking procedure of MOGA described previously. COG has been applied successfully to fill gaps in property space by using quantitative structure-property relationship (QSPR) models to calculate individual fitness [BMG04]. However, a recognized limitation of optimizing in similarity space is that the resulting designs are highly similar to existing molecules. To overcome this problem, Brown *et al*. have more recently published a similar method using the inverse QSPR methodology that optimizes molecules directly in property space, allowing multiple molecular properties to be optimized simultaneously [BMG06].

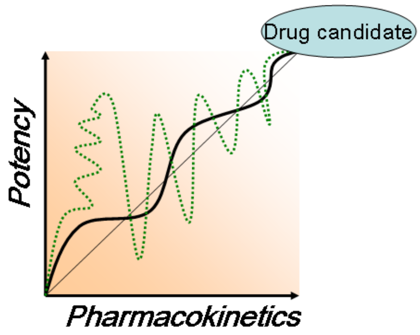


Fig. 4.3. A graph showing different approaches to objective optimization in drug discovery. The dashed line represents the sequential single-objective optimization of conflicting objectives. The continuous straight line represents the ideal optimization solution (not achievable in practice). The continuous wavy line represents the multi-objective optimization of conflicting objectives, whereby the solution space for satisfactory compromises to all objectives is searched simultaneously, resulting in a more direct route to the drug candidate compared to the SOOP method. (Figure adapted with permission from Wiley-VCH Verlag GmbH & Co KGaA and Baringhaus K-H, Matter H: Efficient strategies for lead optimization by simultaneously addressing affinity, selectivity and pharmacokinetic parameters. In: Chemoinformatics in Drug Discovery. Oprea T (Ed), Wiley-VCH, Weinheim, Germany (2004):333-379. © 2004 Wiley-VCH Verlag GmbH & Co KGaA.)

More recently, two independent groups have published their research on the design and implementation of the software programs MoleculeEvoluator and Mobius that provide a progressive, user-directed de novo design approach [LKBI06], [CO08], [EWBB08]. In both algorithms the generated candidate compounds are profiled and a range of properties of pharmaceutical interest, such as molecular weight, number of hydrogen-bond acceptors/donors, and polar surface area are calculated. Selection is left to the user who assigns a score for each of the candidate molecules based on their expert knowledge, taking into account the molecular structure and the associated property values calculated previously. The MoleculeEvoluator represents molecules using the SMILES chemical language and emphasizes mutation while Mobius uses a simple tree representation with molecular fragments as genes and limits the evolutionary operations allowed so that any new design produced will conform to a predefined blueprint, i.e., a recipe for combining fragments to create a molecule. Mobius also offers the ability to measure the fitness of the individuals on additional objective functions that can be added to the process. In effect, in this approach, the user is the multi-object optimizer and can therefore, focus the area of exploration to those regions deemed to be of most interest for the particular application [NBP07].

Overall, a wide variety of representation schemata, molecule synthesis engines, compound evaluation criteria and search methods have been applied to the de novo design problem [NBP07]. While no standard methodology has evolved most of the recent approaches use some form of an EA to search the chemical space due to its flexibility, robustness and the limited requirements it imposes on the objective functions. Furthermore, the majority of the methods uses either a single-objective optimization approach or transforms the multi-objective problem into a single one using a composite, weighted-average function. This has profound effects on the procedure since such methodology can only provide solutions from a single region of the search space in a single run and has known issues in finding solutions in complex spaces with non-convex surfaces [CS04]. An additional issue arises from the various limitations imposed by most of the methods on the compound design component mainly due to their choice of representation schema (linear or tree structure) and the perturbation operators encoded. Limitations such as the exclusive use of fragment-based genes and constrained versions of mutation and crossover reduce the feasible search space and facilitate the search process. Although such reduction in the search space risks missing potentially interesting solutions, it may also be advantageous, provided that the search space is reduced in a meaningful way since it enables a more thorough exploration of the remaining space. A final point that needs to be raised is the lack of use of domain-specific knowledge in the process other than what is needed for representing and generating solutions (only valid molecules are allowed) and fitness scoring (problem specific objective functions are used). Such knowledge, already available in various forms in the drug discovery process, has already proven of great assistance in other fields by guiding search to more promising regions of the space and enabling convergence to optimality with less effort and higher certainty [M89], [M04].

4.3 Research Perspective

The purpose of the research described in this thesis is to propose an algorithmic framework for the problem of multi-objective optimal graph design for labeled, undirected graphs. Solutions to this problem are graphs consisting of genes from two sets, the set of vertices and the set of edges. Multiple types/labels of vertices and edges are allowed and therefore the problem suffers from the combinatorial explosion of the number of potential graph solutions. Additionally, the OGD problem usually has a complex, multi-modal solution space due to the multiple potentially conflicting objectives that need to be satisfied by the solution graphs and, the combinatorial nature of the problem. Consequently, from a computational optimization perspective, the problem corresponds to searching the huge space of valid graphs to discover and select the few designs satisfying, or compromising in the case of conflicts, the objectives imposed. In this context validity of the resulting graphs is problem specific and, as such, the inclusion of problem domain knowledge to the process can facilitate the process. The role of diversity in the population of solutions also assumes increased importance; since multiple solutions, and not only the single best one, are being sought, the process needs to ensure that the population is -to the degree feasible- representative of the range of solutions existing in the various regions of the search space. To solve the problem a search strategy capable of global exploration while paying special attention to the diversity of the population and the ability to converge to individuals in promising localities of the space needs to be implemented. Below we outline the directions investigated for the purposes of this research:

- **Graph-based Representation of Solutions:** Problems naturally represented as more complex data structures, e.g. graphs, are often encoded as numerical vectors using abstract descriptors for the sake of simplicity. This transformation enables the usage of a variety of software tools designed to manipulate vectors at the cost of information loss occurring due to the inability of simpler abstractions to represent complex structures. Specifically for EAs, the choice of chromosome representation is a crucial design step with direct impact on performance. However, there exist only a few systematic approaches for the design of EAs on nonstandard representations with most algorithms using string or matrix based data structures [EMS01]. In order to avoid information loss we are using graph representations of solutions and appropriate computational data structures and methods. Graph representation is especially useful in assessing the diversity of the population since actual calculations can be performed on the natural representations of the individuals using graph theory techniques.

- **Solution Quality – Niching and Elitism:** In a multi-objective optimization setting multiple equivalent solutions representing different compromises among the objectives are possible. Although in many applications the presence of multiple solutions may be considered a problem, and therefore methods for selecting a´ priori the single 'best' solution are employed, in graph design, including drug discovery, it is widely accepted that multiple instances may provide a different alternative solution equally viable as for example in the case of different chemical structures with the potential to interact with the same target in a different binding mode. Based on this requirement the proposed method produces multiple, diverse solutions and enable users to choose a´ posteriori from a variety of candidates. Of prime importance is the issue of solution diversity. Most current MOEA approaches focus exclusively on diversity of solutions in objective space and ignore diversity in parameter space. This can lead to niching conditions in the parameter space, i.e., the potential domination of the population by a comparatively better family of solutions sharing structural similarity surfacing at a specific generation. This condition, widely recognized in the field of evolutionary algorithms, results in insufficient exploration of the search space, loss of population diversity in objective space and production of nearly identical solutions in genotype space. The problem may remain unnoticed when searching for a single solution e.g. in single-objective optimization problems where the population of solutions used serves only as a breeding mechanism for the single best solution. However, in multi-objective settings, where a set of solutions representative of the true Pareto-front is the goal, niching constitutes a serious problem. A novel niching mechanism securing solution diversity in both parameter and objective space is introduced in the algorithm proposed in chapter 5. The proposed algorithm also uses elitism to ensure that valuable solutions discovered during the optimization search are preserved in a secondary population.

- **Exploitation of Knowledge (Memetic) Component:** It is often the case that there exists substantial knowledge on a given optimization problem that can facilitate the search process if used appropriately. For example, currently, there exists a large volume of pharmacologically relevant knowledge accumulated during past efforts to discover drugs. In order to focus the search effort and speed-up search our supports the inclusion -and exploitation- of available problem-specific knowledge to achieve fast exploration of the search space. A mechanism has been implemented to encode and import the knowledge to the algorithmic process and, to effectively use it during the optimization process. The modular representation of our graph-based algorithm is exploited for the integration of knowledge into the operators and algorithm. Additionally, knowledge related to the search progress, in the form of the advancement of the Pareto-front towards the optimal point, is mined and exploited to self-adapt the system and achieve improved performance. Such knowledge is used to activate the memetic component of the algorithm that divides the population into natural groups according to solution genotype and applys a local search technique on each group to exploit that region.

- **Performance and Scalability:** The inclusion of multiple objectives to an optimization problem defines a complex search space with multiple local minima, and potential non-uniform landscape features. Such search space needs to be effectively explored and, therefore, it has been a requirement for the proposed platform to be able to search both globally and locally to detect solutions within a reasonable amount of time. In order to achieve this goal a system must be able to maintain performance standards for reasonably bigger and more complex problems. Our research efforts have focused on exploring the knowledge available to the system from previous runs to avoid non-promising regions of the search space and on maintaining archives of already visited solutions. Further, our work has investigated the use of modern hardware features, i.e., multi-core processors, that if exploited can provide additional computational resources and increase method scalability. The approach followed focused on preparing a parallel implementation of the method through the evolution of numerous subpopulations, and, on distributing the optimization process during run time on multiple cores of a single processor so as to increase scalability with the minimum algorithmic modifications.

Chapter 5

The MEGA Algorithm: Design, Implementation and Application Domain

The global-search capabilities of evolutionary algorithms, the successful record of multi-objective optimization methods utilizing evolutionary principles in various applications and the ability to combine these principles with knowledge-driven local search techniques has driven us to design a memetic algorithm for the optimal graph design problem. The research proposed in this dissertation follows a hybrid approach, combining methods from the fields of multi-objective optimization, evolutionary/memetic algorithms and graph theory. In addition to smoothly integrating these fields, the resulting method expands appropriately research in each area by introducing innovative steps as described initially in section 1.3 and in more detail in 4.3. In this chapter we describe in detail the proposed method.

5.1 The MEGA Algorithm

The Multi-objective Evolutionary Graph Algorithm (MEGA) framework proposed in this dissertation combines evolutionary techniques with graph data structures to directly manipulate graphs and perform a global search for structurally diverse, promising solutions compromising the objectives. MEGA can incorporate problem-specific knowledge and local search heuristics and techniques, to improve performance and scalability. MEGA was initially proposed in [NPA08], [NAP09] where it was applied successfully to the design of novel small molecules exhibiting selectivity to one of two closely related pharmaceutical targets. Applications of MEGA have also been published in [NKP09] while pMEGA has been introduced in [KNP09]. A general diagram of the algorithm framework showing its major components is shown in Fig. 5.1. The pseudocode of the algorithm is shown in Fig. 5.2.

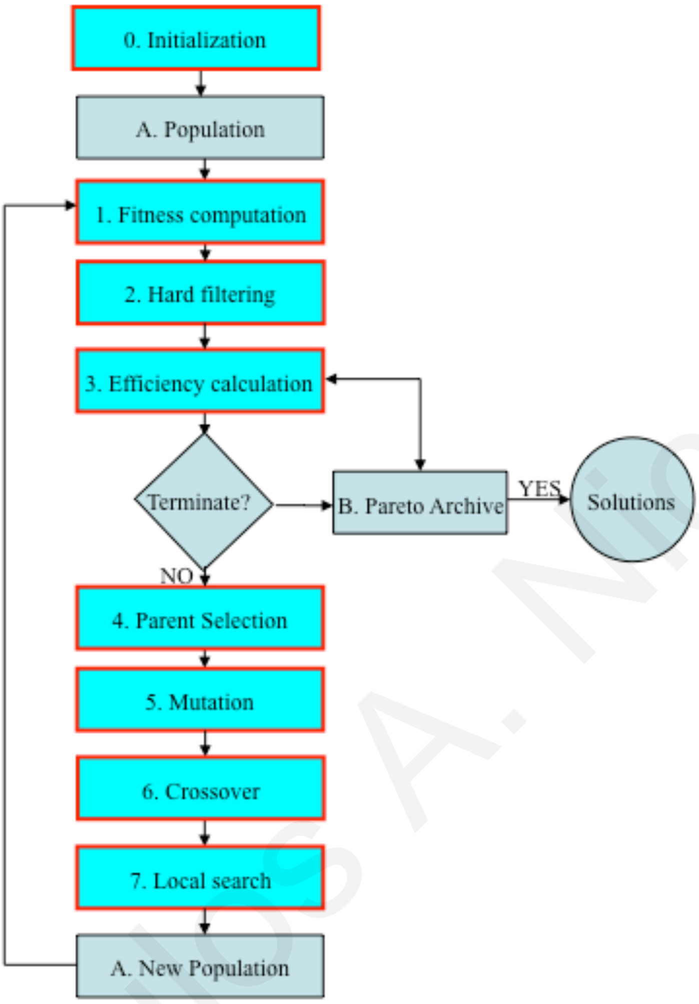


Fig. 5.1: The MEGA algorithmic framework. Efficiency calculation involves the calculation of Pareto-ranking, chromosome clustering and assignment of the efficiency value to each solution. The local search step involves the usage of local search methods on subsets of the population following the approach of Memetic Algorithms.

```
0. Initialization
    Generate initial population P of size S
    Initiate Pareto-archive
1. Fitness computation
    Evaluate solutions in P against primary objectives $O_{1-n}$
2. Hard filtering
    Perform hard-filtering using secondary objectives
3. Efficiency calculation
    Update Pareto-archive
    Assign Pareto-rank to solutions
    Perform graph-based clustering of solutions
    Assign efficiency value to solutions based on Pareto-rank and clustering
While Not Stop Condition:
    4. Select parents $P_{parents}$ in proportion to efficiency values
    Generate population $P_{offspring}$ by reproduction of $P_{parents}$
        5. Domain-specific mutation on individual parents
        6. Domain-specific crossover on pairs of parents
        7. Local search improvement
    Merge P, $P_{offspring}$ to create $P_{new}$
    1. Fitness computation
    2. Hard-filtering
    3. Efficiency calculation
```

Fig. 5.2: The pseudocode of the MEGA algorithmic framework.

MEGA operates on two population sets, the normal, working population (Fig. 5.1.A), and the secondary population or Pareto-archive (Fig. 5.1.B) [NKP09]. The former population consists of the individuals subjected to objective performance calculation and obtained through evolution in a single iteration. The latter supports a form of elitism aimed at preserving promising solutions found throughout evolution and ensuring that the final Pareto-approximation will contain the best solutions found. The algorithm requires the supply of a set of genes, the implemented objectives to be used for scoring the graphs and a set of attributes controlling mutation and crossover methods and probabilities, parent selection, size of working and secondary population, hard filters for solution elimination, etc. Optionally, a set of graphs to be used as the initial working population may be supplied as well. The supplied data is used for initialization purposes, i.e., to create graph-based chromosomes, to construct a list of subgraph gene objects and to initiate additional internal data structures required for the execution of the algorithm (Fig. 5.1.0). At this stage the external archive of solutions intended to store the secondary population of elite solutions is also created.

The first phase of the algorithm (Fig. 5.1.1) applies the objectives on the working population to obtain a list of scores for each individual. The list of scores may be used for the elimination of solutions with values outside a predefined range allowed by the corresponding active hard filters provided by the user (Fig. 5.1.2). Objectives used in this manner are typically referred to as secondary, while objectives used to guide optimization are considered primary. Secondary objectives are used to filter out solutions not conforming to some well-established problem specific constraints and thus limit the search space. In the next step (Fig. 5.1.3), the two populations, working and secondary, are merged and the individuals' list of scores is used in a Pareto-ranking procedure to set the rank of each individual. The combined population forms the new working population. The algorithm then proceeds to calculate an efficiency score for each individual using a novel methodology that operates both in parameter and objective space. The methodology employs an elaborate niching mechanism that performs diversity analysis of the population based on the genotype, i.e., the chromosome graph structure, and subsequently assigns an efficiency score that takes into account both the Pareto-rank and the diversity analysis [NAP09]. The efficiency score of each individual is then used to update the Pareto-archive. The current Pareto-archive is replaced with a subset of the working population that favors individuals with high efficiency score, i.e., low domination rank and high chromosome graph diversity. Note that the size of the subset selected to populate the Pareto-archive is limited by a user-supplied parameter.

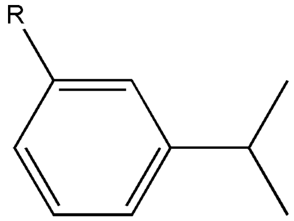
Following the update of the Pareto-archive MEGA checks for the termination conditions, typically if the number of preset maximum allowed iterations has been reached; if satisfied the process terminates. However, if this is not the case the process moves to select the parent subset population (Fig. 5.1.4) from the combined population set using a variation of the “roulette” method [CS04] on the efficiency scores of the candidate solutions. The “roulette” selection method chooses solutions via a probabilistic mechanism that assigns higher selection probability to solutions with higher efficiency score. The parents are then subjected to evolution, i.e., mutation (Fig. 5.1.5) and crossover (Fig. 5.1.6) as well as local search (Fig. 5.1.7) according to the probabilities indicated by the user. The new working population is formed by merging the original working population with the newly produced offspring of the parent evolution step and reducing it to the user defined population size also using a “roulette”-like method. The method is essentially identical to the “roulette” parent selection method described previously except that it assigns a higher selection probability to the worst performing solutions. Best performing solutions, i.e., nondominated solutions, essentially have a selection probability of zero. The process then iterates as shown in Fig. 5.1 and 5.2.

MEGA incorporates heuristics to enable the exploitation of existing problem specific knowledge. The heuristics involve the usage of the weights associated with the subgraph genes provided and result in favoring those with an increased weight. Additionally, the progress of the Pareto approximation set is monitored to self-adapt certain attributes controlling the optimization process and specifically the initiation of local search on specific subsets of solutions from certain regions of the space.

In order to avoid duplicate work and the resulting performance degrade, MEGA incorporates two additional mechanisms worth special mention. The first is a caching mechanism that contains each and every chromosome evaluated during the execution of the algorithm. This includes all members of the initial population as well as the complete set of offspring generated in all iterations. The size of the cache is limited since it only includes the identity (ID) and fitness values of the chromosome measured in some previous iteration. An associative memory hash data structure is used to store the cache to ensure negligible cost to the execution run time. When new chromosomes need to be evaluated against the set of objectives the cache is used to identify whether a specific chromosome has been previously scored and, if so, omit the potentially costly fitness evaluation process and return the values calculated previously. The choice of a chromosome ID is crucial to the success of this scheme since it needs to guarantee that different chromosomes have different IDs and identical chromosomes have the same ID. The operation of the mechanism is especially useful in optimization processes with costly objective functions, such as docking in the de novo design problem case. The second mechanism, active during the evolutionary steps, simply checks and removes those offspring that are identical to some parent chromosomes. The key features of the algorithm are described in detail below:

Graph-based Chromosome Repesentation. MEGA uses graph-based chromosomes to avoid the information loss associated with the encoding of more complex structures into simpler ones, i.e., graphs into bit vectors, and the need for encoding and decoding graphs into other data structures. In addition to the graph data structure, MEGA chromosomes contain information that can be used during evolutionary design including details related to ways that they can be connected to other fragments. Additionally, chromosomes contain information about their parent structure(s) and the operation that produced them including the fragments used (if any), the type of operation (mutation or crossover), etc.

Information-Rich Subgraph Genes. MEGA uses collections of vertices and edges as well as subgraphs for genes. These collections are also referred to as alphabets. Similar to chromosomes, a subgraph gene may contain information about its attachment points and type, i.e., what kind of edges can be attached to it, and about its weight so as to favor the selection of those that are known to produce individuals with higher chances of exhibiting increased fitness to the problem. The weights are exploited in later steps of the algorithm in two ways: first, to increase the chances of the highly weighted genes to take part in the formation of the initial population and, second, to favor the selection of privileged genes for use during evolutionary steps, and especially mutation, described in detail in a later section. Effectively, this type of gene information can be used to apply mating restrictions during evolution, a technique also used in some other MOEAs [CLV07]. Mating restrictions in MEGA are based on available knowledge, for example rules determining what kind of edges can be used to connect two subgraphs. Fig. 5.3 below presents a sample chemical subgraph gene and its associated information.



Weight: 68%

Fig. 5.3: A sample, relatively simple chemical subgraph gene with information on the subgraph structure, its single attachment point marked with R and its weight.

Graph-specific Evolutionary Operations. MEGA uses a rich set of operations inspired by nature to evolve solutions. The modifications effected by the operations cause changes at the level of single edges and vertices as well as on subgraphs that may be selected from a predefined list or adaptively calculated. The available processes can be divided into two main categories, those inspired by mutation and those inspired by crossover.

A. Mutation-inspired processes:

- Flip-Vertex: Modifies the type/label of a vertex by choosing from the available vertex alphabet collection; for example in the case of chemical graphs the vertex alphabet may consist of the collection of valid atoms.
- Flip-Edge: Modifies the type/label of an edge by choosing from the available edge alphabet collection; for example in the case of chemical graphs the alphabet may consist of the collection of valid bond types.
- Remove-Vertex: Removes a vertex
- Remove-Edge: Removes an edge
- Remove-Ring: Identifies and removes a ring/cycle
- Remove-Fragment: Identifies and removes a subgraph

- Exchange-Fragment: Identifies a subgraph of the existing chromosome and exchanges it with a subgraph from the available subgraph alphabet collection.
- Insert-Fragment: Identifies an attachment point and inserts a subgraph from the subgraph alphabet collection.

B. Crossover-inspired processes:

- Combine-1-point: Given two parent chromosomes the process identifies and cleaves an edge from each of them and recombines the resulting fragments to generate two offspring.

The frequency of occurrence of the evolutionary operations is controlled by user-defined probability parameters supplied during the initialization of the process. Mutation probability ranges from 0 to 1 with probability 0 resulting in no mutation operations taking place and probability 1 resulting in the generation of one mutant offspring for each of the mutation-inspired processed described above. Note that the mutation operator can result in several offspring per parent contributing in this way to a more intensive local search of the parent neighborhood. Similarly, crossover probability ranges from 0 to 1 with 0 resulting in no crossover taking place between the supplied parents and 1 resulting in one crossover operation producing two offspring.

Problem-domain specific knowledge is incorporated via the usage of weights on the collection of subgraph genes that in effect reflect privileged status and affect the probability of usage of each gene, and, through the mating restrictions enabled by the information-rich genes used. The weight value of each gene may be calculated using problem domain specific data; for example, the weight for chemical subgraph genes used in a de novo drug design problem can be determined by the drug-like characteristics of each subgraph.

Objective Encoding-Scorers. Several methods for assessing the fitness of a graph solution have been prepared. These so-called objective scorers are typically problem specific measuring the quality of the graph as relating to the optimization problem investigated. Moreover, a methodology for incorporating additional objective scorers as needed has been defined. The methodology defines an advanced programmer interface (API) that enables the encoding of graph fitness calculation tools and their use to obtain quality measures in real time.

Niching Mechanism. MEGA avoids the niching problem by performing a diversity analysis at the genotype level, i.e., of the graph chromosomes, using a novel mechanism. As previously mentioned, proximity of solutions in objective space does not necessarily correlate to proximity in parameter space (see Fig. 2.1). In order to preserve population diversity, both at the genotype and phenotype level, MEGA clusters the chromosome graphs, ranks the clusters based on their size and the number of Pareto solutions they contain and samples parents based on solution cluster assignment and rank. Specifically, the selection method is applied on the clusters rather than the entire population. The process picks one solution from each cluster starting from the most populous ones and proceeding to clusters containing fewer compounds. The process traverses the set of clusters until the required number of parents is selected. This approach enables sampling of diverse graphs and the preservation of the overall population diversity.

The specific Pareto ranking method used is the one proposed by MOGA (see section 3.2.4). According to this procedure the rank of an individual is set to the number of individuals that dominate it incremented by one, thus, nondominated individuals are assigned rank order one (see Fig. 2.2). A linear transformation function that assigns a higher score to solutions with low Pareto-rank is used to calculate the first element of the MEGA efficiency score. This function operates exclusively on phenotypes, i.e., in objective space. A second method performs diversity analysis of the population via clustering of the genotypes, i.e., the chemical structures, and assigns the cluster placement of each individual as the second element of the efficiency score. The efficiency score calculation technique can be fine-tuned by user supplied parameters to favor the parameter space, the objective space or to balance between the two. Note that in the current implementation, traditional methods for efficiency calculation operating exclusively on the phenotype, i.e., the objective space, have also been implemented for performance assessment comparisons.

Elitism. Pareto-archiving, an elitist mechanism designed specifically to preserve good nondominated solutions from getting lost [ZT99], is implemented in MEGA. The mechanism uses a secondary population where nondominated solutions found during previous iterations are stored. In each iteration MEGA merges the Pareto archive with the current population before the efficiency score calculation step and uses this larger set as the current, working population. This extended population is used during the parent selection step. The Pareto-archive is then reset based on the efficiency scores of the extended working population. Note that the size of the Pareto-archive is typically set to a large number so as to allow the storage and preservation of a number of solutions exceeding the user-defined population size. When the number of nondominated solutions exceeds the size of the archive clustering of the solutions based on the chromosome graphs is used to appropriately reduce the number of the elit solutions. Specifically, solutions are eliminated from the most populous clusters while care is exercised to preserve solutions from under-represented clusters.

The mechanism is a result of observations made during runs of initial versions of MEGA where some promising solutions were lost due to the large number of Pareto solutions found. This paradox, partly caused by the success of Pareto-based MOOP methods in generating large, dense populations with multiple nondominated solutions, resulted in the obligatory elimination of good solutions since the number of nondominated individuals exceeds the size of the population. Zitzler [ZT99] already identified the problem and proposed techniques based on Pareto-front archiving and creation of an elite population of solutions.

Exploitation of Knowledge (Memetic) Component. In an effort to improve search efficiency, MEGA incorporates simple heuristics that can be used to exploit existing pharmacologically relevant problem specific knowledge. The heuristics involve the usage of the weights associated with the genes provided and result in favoring those with a privileged status, i.e., increased weight. Specifically, the flip_edge, flip_vertex, insert_fragment and exchange_fragment chromosome evolution operations select the gene to use using a mechanism based on a "rulette" principle that favors blocks with higher weight. Provided that the schema used to assign weights to genes reflects the likelihood of increased performance when a specific gene is used, the mechanism results in the production of individuals with genes associated with better performance.

Additionally, MEGA monitors the optimization process execution and acquires knowledge about the progress performed through the calculation of quantitative performance measures of the Pareto approximation set obtained in successive iterations. This knowledge may optionally be exploited to self-adapt and adjust the search performed by initiating local search on selected search space regions. Specifically, MEGA identifies lack of progress during the search and selectively applies local search to improve certain solution subsets. This characteristic is similar to the memetic algorithms reported in [KN04], [S07]. Our efforts have initially focused on utilizing the self-adaptive capabilities of MEGA to escape local minima. In order to achieve this, a STagnation Identification and Resolution mechanism (STIR) has been implemented. STIR relies on measuring progress over successive generations and identifying the lack of new, nondominated solutions a condition referred to as stagnation. The method calculates and stores the hypervolume measure of the current Pareto-approximation set at the end of each iteration. Stagnation, or lack of progress, is identified when the hypervolume measure value remains unchanged over a user-defined number of iterations. When stagnation conditions are detected STIR invokes a subpopulation-based mechanism to focus search space exploration on local regions. The careful selection of a subset of closely related promising solutions is seen as a method to explore thoroughly the solution space defined by them and converge to the local optimum more efficiently. The normal processes of fitness calculation, mutation, crossover and selection take place within each subpopulation to emphasize local, rather than global, solution search. The use of subpopulations requires a more intelligent strategy for splitting the original population since evolution continues within the limited subset of solutions and diversity issues may arise. MEGA employs a clustering method on the solutions in the Pareto-archive identical to the one used to eliminate the niching problem. Upon clustering of the available population the algorithm simply selects subpopulations based on the resulting clusters. Following the application of STIR, the algorithm merges the various subpopulations and proceeds as usual. The self-adaptive version of MEGA may invoke STIR several times in a given run to re-split and perform local search (see Fig 5.4).

```
Initiate STIR mechanism
        Assess progress achieved with current P
        if progress:
                Continue
        else:
                Identify clusters in population P
                Form subpopulations $SP_{1-n}$ based on clusters
                Improve subpopulations independently via local search
                Merge subpopulations to form $P_{intermediate}$
End STIR mechanism [set $P_{new} = P_{intermediate}$]
```

Fig. 5.4: The STIR (STagnation Identification and Resolution) mechanism is designed to detect lack of progress in successive generations, a sign of getting stuck in local minima, and use local search techniques to perform local region exploration of subpopulations and boost search efforts.

5.2 pMEGA: Parallelizing the MEGA Algorithm

EAs are easily parallelizable partly due to their population-based search strategy. Efforts for Parallel EAs (PEA) often follow the fine-grained parallelization paradigm [CLV07] that focuses on distributing calculations based on individuals, i.e., processes that only use information from single chromosomes, such as fitness calculation. For example, a population may be divided randomly in subsets and each subset may be forwarded to a different processor for fitness scoring. The scores obtained in such distributed manner are re-integrated and supplied to the later steps of the algorithm that proceeds to evolve and select parents as usual [W08]. Similarly, individuals (or subsets) of the working population may be evolved independently in a distributed manner and re-integrated into a combined set for subsequent use by the algorithm.

A second approach to adopt parallelization requires the division of the working population into subpopulations, the distribution of each subpopulation to a different processor and its independent evolution [MZ96]. This approach, known as coarse-grained parallelization, requires a master process controlling general evolution and the ability to generate slave-processes that can operate independently on different processing units for a number of iterations [AT02]. In this paradigm, subpopulations evolve independently of each other for a certain number of generations the so-called isolation time. Upon completion of the isolation time the master process collects and merges the results produced and reinitiates evolution of the total population or in a distributed fashion. During merger a number of the resulting individuals is re-distributed between the subpopulations, a process referred to as migration. The number of exchanged individuals (migration rate), the selection method of the individuals for migration and the scheme of migration determines how much genetic diversity can occur in the subpopulation as well as the exchange of information between subpopulations. The selection of the individuals for migration typically takes place in a random manner or using fitness scores, e.g. select the best individuals for migration.

Several possibilities exist for the migration scheme of individuals among subpopulations. Common migration schemes include the complete, unrestricted net topology, which exchanges individuals among all subpopulations, the ring topology, where exchange of individuals is allowed only to a specific subpopulation, and the neighborhood topology which exchanges individuals across a “neighborhood” [AT02].

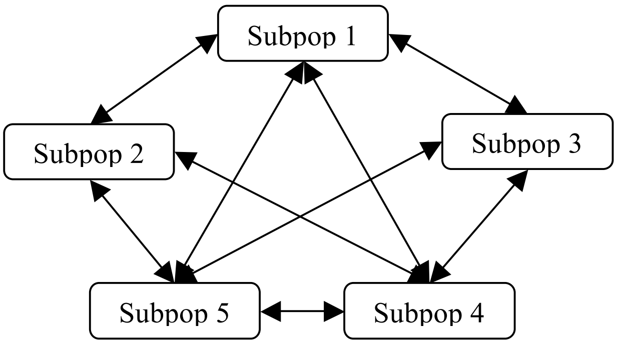


Fig. 5.5: Diagram of a coarse-grained parallel EA (PEA) using the subpopulations model and unrestricted migration topology.

pMEGA, [KNP09] the parallel implementation of MEGA, uses population level parallelism, i.e., follows the coarse-grained paradigm, and distributes the population into several smaller subpopulations to be evolved concurrently. The algorithm randomly splits the working population to several subpopulations and uses a predefined pool of processes generated by the master process, to which it assigns tasks for execution. An example of a task is the independent evolution of a subpopulation set. In order to take full advantage of the processes that will handle the tasks, care is taken so that the number of subpopulations is greater or equal to the number of the processes created. A diagram of pMEGA is shown in Fig. 5.6. The pseudo-code of pMEGA is given in Fig. 5.7.

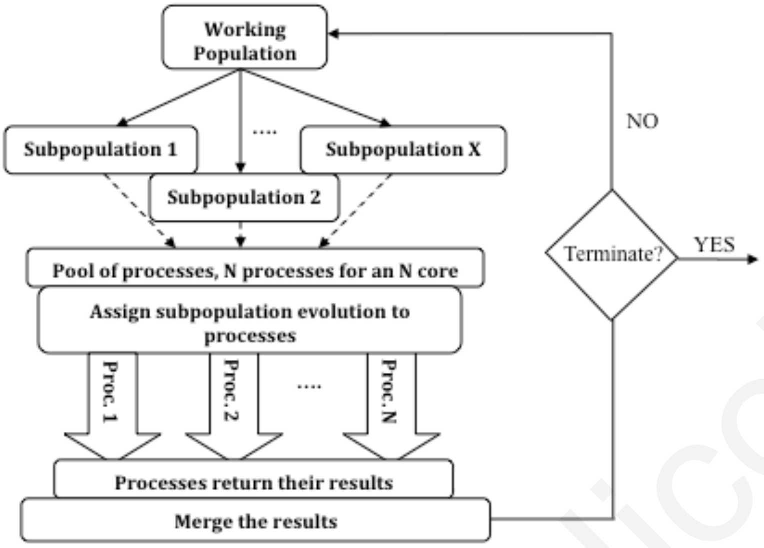


Fig. 5.6: A diagram of the pMEGA algorithm.

```
# Master process
Create pool of processes
Initiate working population
While Not Stop Condition:
        Split working population into several subpopulations
        Create tasks list (Each task evolves a given subpopulation using MEGA)
        Assign tasks to processes in pool
        Wait for results
        Collect results
        Merge and post process results
        Create new working population

# Process
Receive task
Evolve subpopulation for epoch_counter iterations;
Return results
```

Fig. 5.7: The pMEGA algorithm pseudo-code.

Subpopulations are evolved independently for a specific number of iterations defined by a user-supplied epoch_counter, which is set to a percentage of the total iterations the algorithm has to run. The default setting for pMEGA is set to 10% of total iterations. The independent evolution of each subpopulation is a scaled-down execution of the MEGA algorithm as shown in Fig. 5.6. Specifically, during execution time a pre-constructed process from the pool of processes is assigned a task i.e., to execute a scaled-down MEGA. The working population of the process/task is set to a subpopulation set and the number of iterations is set to the epoch_counter. During the evolution of subpopulations, migration is not permitted between the subpopulations. Upon completion of the task, the process returns the results produced and gets assigned a new task, if one is pending [K10].

When all subpopulations complete their evolution, their results are gathered and merged. The new working population is created from the merger of the resulting populations provided by the set of task executions, thus performing complete, unrestricted migration among the subpopulations. The new working population is subjected to Pareto-ranking and dominated individuals are removed so that the working population consists of the Pareto approximation set of the solutions. Following, pMEGA checks for the termination conditions; if satisfied the process terminates. However, if this is not the case the process moves to repeat the previous steps.

5.3 De Novo Design Specific Implementation and Materials

The validation of MEGA was performed through application on the DND problem (see section 4.2.1) using a variety of objective types. Pareto-based MOOP methods have recently been introduced to the drug discovery field [NBP07], a domain which has traditionally been in the forefront of computational science research [J04], with some successes reported [SF05], [SHRTPS08], [NAP09]. DND involves the design of molecular graphs that satisfy multiple objectives concurrently. The problem represents a challenging test case for the optimal graph design problem with intense ongoing research from a variety of research groups due to its direct applications in drug discovery, agrochemicals and fine chemicals research, among others. Following, some implementation specific details are provided as well as the materials used for the experiments performed in chapter 6.

Subgraph Gene Generation. As previously mentioned MEGA uses collections of vertices and edges as well as subgraphs for genes. In the DND specific implementation vertices correspond to atoms, edges to chemical bonds and subgraphs to molecular fragments or substructures. The sets of atoms and bonds correspond to those found typically in chemical structures and especially drugs. For the purposes of this implementation we use a substructure mining tool [NOES08] able to extract fragments from graphs in a variety of ways including frequent subgraph mining [NP06] and the RECAP (Retrosynthetic Combinatorial Analysis Procedure) chemical bond type identification and cleaving technique [LBWH98]. This tool facilitates the preparation of a large pool of subgraph genes that contain information about their attachment points and the type of bond cleaved at each attachment point prior to evolutionary design. The resulting fragments are profiled using available knowledge on the molecules that contain them, and weights, reflecting their privileged or not status, are recorded. In this context, knowledge is the result of past efforts to discover drugs, for example experiments performed to experimentally measure the potency of a set of compounds to a specific target. Essentially, the weight of a specific fragment is incremented by one for each molecule containing it that has a favorable biological profile, e.g. is a drug, or is "active" against the target of interest, and, decremented by one for each molecule containing it that is considered unfavorable. The RECAP utility of the tool is also used during evolutionary operations as explained in a following section.

Graph-specific Evolutionary Operations. For the fragment removal and exchange operations RECAP is used to break the molecule in two disconnected parts and either remove or replace one of them with a fragment from the fragment collection. Note that fragment weights influence the probability of selection of a fragment for the insertion and exchange operations. Also note that users can optionally restrict the exchange fragment operation to subgraph genes with attachment points of compatible RECAP bond types in order to increase synthetic feasibility chances.

Crossover takes place using two operations. The first identifies and cleaves a RECAP-type bond in each of two parents and recombines the resulting fragments to generate offspring. In a manner similar to the exchange fragment operation described above, this type of crossover can also be restricted to breaking specific bond types and combining fragments with compatible bond types in order to produce chemical designs with higher chances of being synthesizable. The second uses a methodology initially proposed in [G99] and later used in [BMGG04] as well. A bond is selected randomly in each of two parents and removed. If the bond is part of a ring system, then additional appropriate bonds are also cleaved to obtain two fragments. The resulting fragments are then recombined to generate offspring. Note that a check and repair or discard mechanism is applied to ensure that the resulting offspring are valid molecules with respect to valences. Briefly, in its current implementation the mechanism identifies atoms with valence problems and attempts to repair them by either removing hydrogens attached to the atom or by downgrading atom bonds to a lower order, i.e., converts a double bond to single or a triple to double. If such action is not possible or sufficient to fix the problem, then the offspring is discarded.

Caching. The caching mechanism implemented in the current implementation is specific to chemical structures. The required unique IDs for the molecular chromosome graphs are calculated using the SMILES chemical language [W89] which represents molecules using unique alphanumeric strings. The SMILES implementation provided in the NSisToolkit [NOES08] chemoinformatics was used.

Niching. In the current implementation we have used the Wards agglomerative clustering technique [WB00] and a variation of atom-type descriptors [KSF96] for the encoding of molecular structure in vector representation. The resulting Wards cluster tree is processed with the Kelley cluster level selection method [WB00] to identify the main solution clusters. The clusters obtained are ranked based on their size, and solutions are labeled with the characteristics of the cluster they belong to. Care is exercised to accommodate the likely presence of singleton and under-represented clusters often found when the population size is small or particularly diverse. Such clusters may cause problems during selection therefore MEGA implements appropriate rules, such as allowing only simple selection from singleton clusters.

Objectives. Several methods for assessing the fitness of a molecule have been prepared. The methods fall in three main categories:

(a) Binding Affinity Scorers. We have chosen to use the docking program Glamdock [TA07] recently developed by Tietze and Apostolakis. We have developed pyChill, a python wrapper for Glamdock, to enable tight integration into our de novo design system and facilitate the encoding of docking related objectives, i.e., objectives based on the predicted binding affinity of a designed molecule to a target protein. The designed molecules are docked into the binding site of the corresponding protein, and the interaction score of the best solution is used as an objective function. Settings for docking correspond to the slow settings described in ref [TA07]. The ChillScore is used to score interactions. This integration allows us to easily prepare a binding affinity scorer for a chosen target protein and produce fitness scores of our designed molecules via an interactive process in real time.

(b) Molecular Similarity Scorers. Our system can use molecular similarity as a distinct objective when one of our goals is to produce molecules that resemble (or are quite different from) a known ligand. Similarity was calculated using the tool Fuzzee [MOD09]. The specific method used operates on abstractions of molecular graphs that replace atoms with molecular features to produce the so-called feature graphs. The actual similarity is calculated in a pair-wise manner by first aligning the feature graphs of two molecules, identifying common features and then applying the Tanimoto similarity measure [WBD98]. An alternative method relies on the calculation of a variation of the atom-type descriptor vectors proposed in [KSF96] that includes ring membership information of the calculated atom types. Again similarity calculations are performed using the Tanimoto measure [WBD98].

(c) Chemical Structure Scorers. Often, selected classes of drug molecules tend to obey some simple rules easily calculable from the chemical graph of a molecule. An example of this type of rules is the widely known oral bioavailability Rule-of-Five described by Lipinski *et al*. [LC97]. In order to exploit such rules and generate fitness scores for the molecular structure we have encoded scorers measuring simple molecular structure properties such as the number of rotatable bonds, the number of hydrogen bond donors and acceptors, and the molecular weight.

Self-Adaptive Mechanism. Stagnation, the condition used to trigger the activation of the STIR mechanism, depends on measuring lack of PAS progress over consecutive iterations of the optimization process. The current implementation of MEGA enables the user to set the number of iterations required to invoke STIR. For the purposes of this dissertation, STIR is invoked when the hypervolume measure values of a number of iterations equal to 5% of the total iterations of the optimization progress remain unchanged. As an effect, the STIR mechanism can be activated multiple times, every 5% of the total number of iterations in the worst case.

Evaluation. The application of MEGA, as well as its parallel version pMEGA, includes runs with a variety of input parameter settings. Multiple runs were performed for each combination of parameter settings to obtain a reliable profile of the implementation performance and its robustness. The results from the application of MEGA were evaluated both in a quantitative and a qualitative manner. Quantitative evaluation was performed via the application of the performance measures implemented (e.g. hypervolume, spacing, diversity) on the Pareto-approximation set produced in each run. Visual inspection, including expert partner review, provided qualitative evaluation. The results of MEGA have been presented successfully at the 8th International Conference on Chemical Structures (ICCS) Conference in the Netherlands (June 1-5, 2008) [NPA08], the Leiden Workshop on Drug Design Optimization also in the Netherlands (July 19-23, 2009) [NPK09] and the 13th International Conference on Information Technology and Applications in Biomedicine

(ITAB) Conference in Larnaca, Cyprus (November 4-7 2009) [NKP09]. Furthermore, the algorithm and some of the results obtained were published in the Journal of Chemical Information and Modeling of the American Chemical Society.

Performance Boosting. Following the validation of the algorithm implementation through the initial successful runs, efforts aiming at improving the computational performance of MEGA have been made. These efforts have produced pMEGA, a parallel version of the algorithm. Moreover, several mechanisms focusing on caching and reusing of fitness scores have been successfully implemented. Additional efforts for parallelizing the execution of the algorithm are planned for the future.

Materials. Two datasets were used during the MEGA and pMEGA tests performed. Dataset 1, a set of well-known Estrogen Receptor (ER) ligands, contains five compounds, three with increased selectivity to ER-β and two with selectivity to ER-α. Dataset 2 is an ER inhibitor dataset obtained from Pubchem [WEA06]. The dataset consists of 86098 compounds tested on both ER-α (Bioassay 629) and ER-β (Bioassay 633). Two collections of subgraph genes were used for all the tests performed, the first with 51123 blocks and the second with 2363. Both subgraph gene collections were obtained via fragmentation of molecular datasets with the substructure mining tool provided by [NOES08]. The 51123-collection was the result of fragmentation of Dataset 2 described above. The weights for these subgraph genes were assigned according to the activity labels of the molecules containing it. The 2363-collection was the result of fragmentation of 53 known ER ligands from [SR01]. Additionally, the known structures of the ER-alpha (Protein Data Bank - PDB code: 1xpc) and -beta (PDB code: 2fsz1) have been used for the encoding of optimization objectives. Finally, the structure of Tamoxifen, a known ER-alpha ligand marketed as a drug, has also been used.

The application of the algorithm presented in the experimental section relied on the encoding of receptor-based objectives to guide the design of molecules with sufficient binding affinity and/or ligand-based objectives that measure the average similarity of a query molecule to known ligands. The receptor-based objectives encoded the ER-α and ER-β receptors and measured the potential binding affinity of a query molecule. The ligand-based objectives measured similarity of a given query molecule to a set of molecules, e.g. known ER-α selective and ER-β selective ligands. The use of these objectives enabled the design of molecules possessing a variety of properties, e.g. selectivity to one of the receptors by maximizing average similarity to the ER-β ligand set and minimizing the average similarity to the ER-α ligand set. A set of hard filters based on chemical structure objectives was also applied in order to remove potentially problematic designs from further consideration. Similarity to Tamoxifen was also used as a hard filter in order to constrain the chemical structures designed to those exhibiting substantial Tanimoto similarity and thereby facilitate qualitative evaluation of the resulting designs.

Tanimoto similarity is defined as [WBD98]:

$$T(A,B) = \frac{c}{a+b-c} \qquad \text{Eq. (14)}$$

where a is the number of enabled features in object/vector A, b is the number of enabled features in object/vector B, and c is the number of enabled features in both A and B. Soergel distance is defined as the complement of the Tanimoto similarity coefficient for binary vectors [WBD98]:

$$S(A,B) = 1 - T(A,B) \qquad \text{Eq. (15)}$$

Euclidean distance, also used in the experiments performed has been defined previously (see Eq. (8)).

Performance has been assessed using the hypervolume, spacing and diversity (genotype and phenotype) quantitative measures as described in section 2.3. In the case of hypervolume, lower measure values indicate better performance while in spacing and diversity better performance is shown by higher measure values. Note that the implementation of the hypervolume measure relies on the PISA platform and programming language interface for search algorithms [BLTE03], [PISA09] which, by convention, uses lower values for better performance. The performance measure results and analysis are depicted using primarily box-plots as implemented in the package MatPlotLib/Pylab [PYLAB09]. In each plot we display the median, lower, and upper quartiles and confidence interval around the median. The box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers. The performance measure values are typically processed using the Mann-Whitney non-parametric test for statistical significance. We have used the implementation of this test found in the [SCIPY09] scientific computing software package.

All runs were performed on two computers: Machine 1 is equipped with an Intel Core 2 Duo E8400 @ 3.0 GHz (2 Physical Cores) Central Processing Unit (CPU) and 4 GB of memory; Machine 2 is equipped with an Intel Core 2 Duo Quad Q6600 @ 2.4 GHz (4 Physical Cores) CPU and 4 GB of memory as well.

Chapter 6

Results

The next sections describe a series of validations tests performed and the results obtained. The experiments include runs with MEGA using a variety of input parameters configurations testing, among others, the effect of the population size and iterations number as well as the impact of elitism and the STIR mechanism, and runs with our implementation of the popular MOEA algorithms MOGA and SPEA for comparison purposes. In order to fully assess the influence of elitism and STIR, simpler versions of MEGA were prepared where the execution of those components has been disabled. These versions of the algorithm are termed simple-MEGA (sMEGA) and elitist-MEGA (eMEGA) in the remainder of this document. Parallel MEGA (pMEGA) runs were executed to evaluate the performance of this variation of the algorithm to MEGA both with respect to the quality of the solution set produced and the speed-up achieved. Runs of the MEGA algorithm on optimization problems with a single-objective were also performed to validate the correct functionality of the algorithmic components and investigate the ability of our method to discover solutions from the entire range of the Pareto-front and specifically from the extremities of the search space obtained when focusing the optimization process on a single objective.

Analytically, the experimental design for the evaluation of MEGA includes investigating the impact of the elitism and memetic components on performance (section 6.2), applying the algorithm on a significant de novo design case study in close cooperation with medicinal chemistry experts who validated the results obtained qualitatively (section 6.3), and, comparing MEGA with commonly used MOEA algorithms, namely MOGA and SPEA (section 6.4). Section 6.5 aims at assessing the performance of the parallel implementation of MEGA, the pMEGA, against MEGA. A brief section, preceding the main experimental part of the chapter, was designed to test the various components of MEGA as well as its overall search capability (section 6.1). This section addresses single-objective de novo design problems that are simpler to evaluate. All experiments were performed using the materials described in section 5.4. Any additional materials used by each set of experiments are described at the beginning of each subsection.

6.1 Validation of MEGA Evolutionary Operations

MEGA has been initially applied to a series of single-objective de novo design problems in order to evaluate its search capability on problems with known, easy to assess objectives. Through these tests the correctness of individual MEGA components such as the graph-based evolutionary operations and the implementations of the objectives have also been tested. The tests also served to test various algorithmic settings including population size, mutation and crossover probabilities and number of maximum iterations. The results produced by the single-objective tests using different combinations of input parameters were subsequently used in the experimental design of the multi-objective tests presented in later sections of this chapter. Both ligand-driven and receptor-driven tests were performed to test the ability of the algorithm to design molecules meeting single criteria. The tests involved designing molecules with high similarity to a known ligand or predicted binding affinity to a given receptor.

6.1.1 Molecules Similar to a Query Molecule

In the similarity-driven tests the single objective function used was similarity to Ibuproxam, a molecule with a relatively simple chemical structure (see Fig. 6.1.1). This test case mimics the capabilities of several modern, ligand-driven DND algorithms including those reported in [G99], [SLS00] and [FS07]. The objective function used was the Euclidean distance of the atom-type [KSF96] descriptor vectors of the query molecule to the individuals in the population. The optimization process aimed at minimizing the Euclidean distance (see Eq. 8) value to zero. The initial population was selected randomly from Dataset 1, and the 2363 subgraph gene collection was used (see section 5.4). Since the subgraph genes were not related in any way with the specific similarity objective, the weights were disabled, i.e., all fragments were set to weight one. The experiments used population sizes 10, 20, 50, and 100. Runs were performed with MEGA and its simpler version sMEGA that does not use the elitism and STIR mechanisms. Three types of evolutionary operation combinations were tested with mutation only, crossover only, and both mutation and crossover applied. When mutation was enabled its probability was set at 0.25 while when crossover was used its probability was set at 1.0. Multiple runs, a total of five,

TABLE 6.1.1: EXPERIMENTAL DESIGN FOR THE TESTS VALIDATING THE CORRECTNESS OF MEGA

Objectives	Population	Iterations	Evolutionary Operations	Configurations
Descriptor distance to Ibuproxam	10, 20, 50, 100	20, 50, 100, 500, 1000	Mutation: 0, 0.25 Crossover: 0, 1.0	Niching, Elitism, STIR: on (MEGA); Niching: on, Elitism, STIR: off (sMEGA)

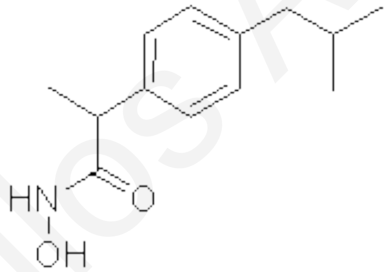
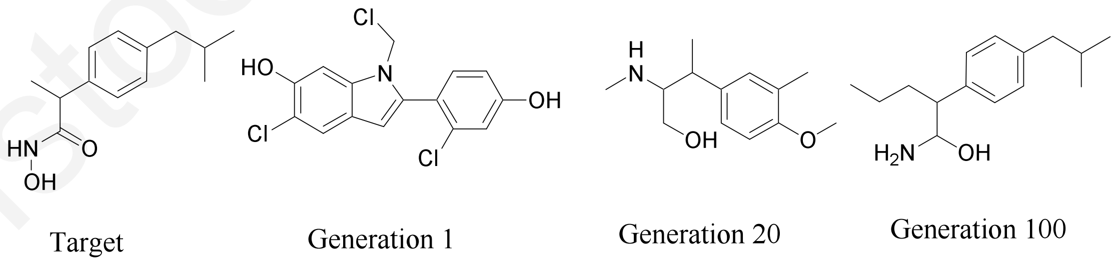


Fig. 6.1.1: The 2D chemical structure of Ibuproxam

were performed for each parameter settings combination with different initial populations to avoid drawing conclusions from chance results produced by single runs. Results were assessed after 20, 50, 100, 500, and 1000 iterations.

In all runs performed with 500 and 1000 iterations the end structures proposed by the algorithm were close to identical to the query molecule even for the smallest of populations (see Fig. 6.1.2). Smaller numbers of iterations revealed varying performance highlighting the effect of additional input parameters. The results demonstrated that the effect of population size can be significant with larger population sizes consistently producing better results both in number of iterations required to converge and the similarity of the final products to the query molecule. In general, tests using both mutation and crossover provided better results than using only mutation or only crossover. Note that the use of the STIR mechanism overall produced slightly better results although it is difficult to conclude whether that mechanism contributes to a consistently better performance. All runs were performed on Machine 1 (see section 5.4). Time requirements for the execution of the runs were limited to less than two minutes for the lengthiest of the runs, i.e., a run with population 100, 1000 iterations, mutation, crossover, and diversity analysis enabled. Figure 6.1.2 presents the progress of the designed molecules until convergence using MEGA and similarity to Ibuproxam.



Target

Generation 1

Generation 20

Generation 100

Fig. 6.1.2: Successive evolution of a chemical structure towards a known molecule, in this case Ibuproxam. The molecules shown were the best performers (most similar) in generations 1, 20, 100 in a MEGA run with population 100 and both, mutation and crossover enabled.

6.1.2 Molecules with Binding Affinity to a Target Receptor

The second single-objective test case aims to design molecules based on predicted binding affinity to a known target receptor site using the ChillScore interaction score family [TA07]. The receptor chosen for this purpose is the ER-α receptor, a relatively constrained receptor where the search space is limited by its well-defined shape. In addition to testing the correct functionality of MEGA components, this test case aims to establish a benchmark against which receptor-based multi-objective results may be compared and validate the evolutionary search capability of the algorithm on a problem where known solutions exist. Note that this test case corresponds to most of the original DND approaches [SF05]. The initial population was sampled in a quasi-random approach from Dataset 2.

TABLE 6.1.2: EXPERIMENTAL DESIGN FOR THE TESTS VALIDATING THE CORRECTNESS OF MEGA

Objectives	Population	Iterations	Evolutionary Operations	Configurations
Docking score to ER-a	10, 20, 50, 100	20, 50, 100	Mutation: 0, 0.25 Crossover: 0, 1.0	Niching, Elitism, STIR: on (MEGA); Niching: on, Elitism, STIR: off (sMEGA)

The method used atom-type descriptors [KSF96] and the Tanimoto measure [WBD98] to calculate the similarity of all compounds in Dataset 2 to Tamoxifen and then selected randomly from the subset of compounds having similarity values less than 0.4 to Tamoxifen. This guaranteed that no member of the initial population would be similar to a molecule known to bind strongly to the target receptor. The 2363 fragment collection was used with the weights enabled since the fragments (and weights) were calculated from a dataset screened on ER-alpha. The population sizes used were 10, 20, 50, and 100 and runs were performed with MEGA and simple-MEGA. When used, mutation probability was set at 0.25 and crossover at 1.0. Multiple runs, a total of five, were performed for each set of parameter settings in combination with different initial populations to avoid drawing conclusions from chance results produced by single runs. Results were assessed after 20, 50 and 100 iterations. All runs were performed on Machine 1 (see section 5.3). Time requirements for the execution of the experiments were substantially higher than those of the similarity based objective tests. A typical run with population 50, 100 iterations, mutation, crossover, and niching enabled took approximately 20 hours.

Results indicate that MEGA can generate solutions comparing favourably with the known solutions, i.e., the drugs designed for the specific receptors, in this case Tamoxifen. Consistently, in all runs performed, the fitness/docking score of the single best solution quickly exceeded the docking score of Tamoxifen. This conclusion refers only to the docking scores obtained for the new compounds in comparison to Tamoxifen and not their drug-like potential that was not taken into account in any way. The role of a larger population heavily influenced the quality of the final solution with larger populations generating better solutions. The effect of the STIR and elitism mechanisms as far as the fitness of the final best solution was negligible. A simple diversity analysis of the final solution sets indicated, as expected, that the final populations consisted of diverse chemical structures due to the use of the MEGA niching mechanism.

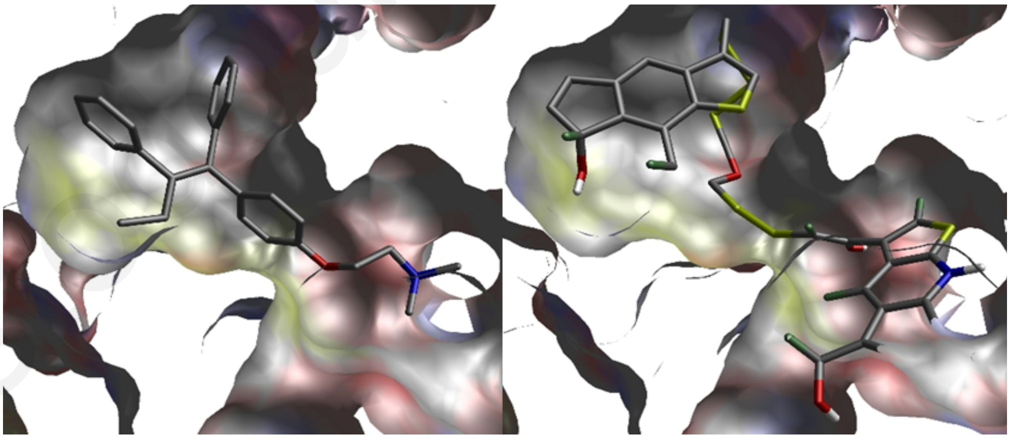


Fig. 6.1.3: Predicted docking poses of two molecules to ER-alpha receptor. Tamoxifen, the known drug modulating ER-alpha is shown to the left. A compound designed using MEGA in a single-objective mode is shown to the right. Note that the compound designed by MEGA is predicted to have higher binding affinity than Tamoxifen despite its non drug-likeness.

Figure 6.1.3 presents the known drug and one of the designed molecules docked into the ER-alpha receptor. Tamoxifen, at the left is predicted to bind with interaction score -22.8 while the designed molecule at the right has a score of -40.4 (see section 5.3. Objectives, for more information on the definition of interaction score). Note the obvious non-druglike characteristics (i.e., large size and increased flexibility) and potential structural issues of the design caused by focusing the exploration of the space exclusively to interaction score. This outcome exemplifies the risks associated with excessive training of the optimization process in combination with single-objective optimization. Figure 6.1.4 presents the 2D structure of another of the designed molecules with a better docking score than Tamoxifen and its docking pose in ER-alpha.

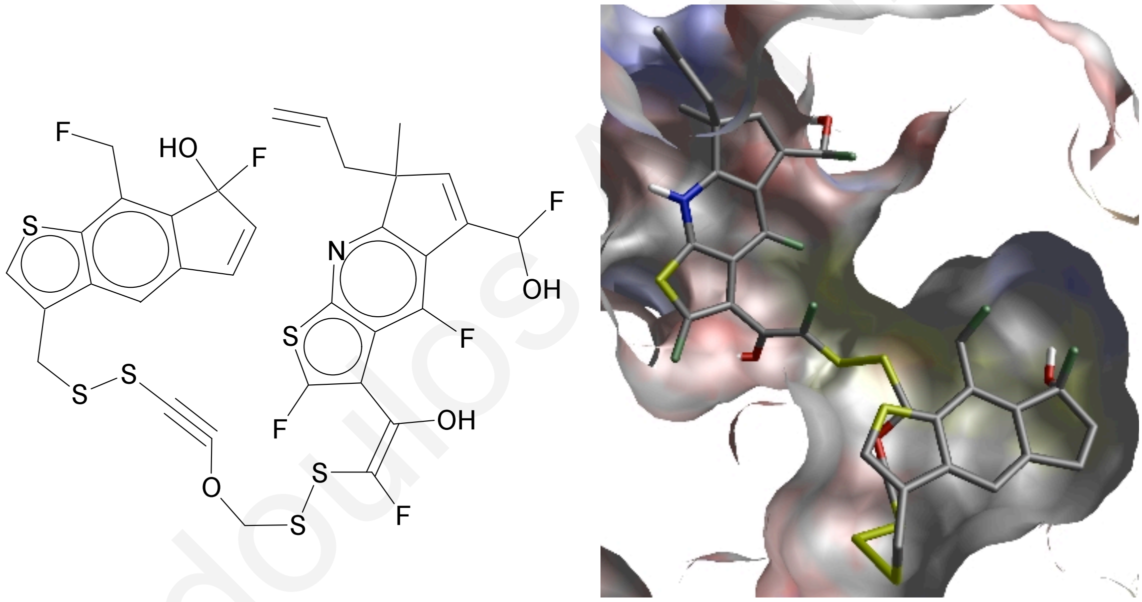


Fig. 6.1.4: 2D structure of a *de novo* designed molecule (left figure) predicted to bind to ER-a with a higher affinity than Tamoxifen. The molecule, posessing several non-druglike characteristics such as large size, flexibility and structural issues, has been designed with a single-objective optimization process focusing only on maximizing the predicted binding affinity. The docking pose of the molecule in ER-alpha is shown to the right.

6.2 Investigation of MEGA Components

The objective of this set of tests is to investigate the influence of the main, custom components of MEGA and better understand the inner workings of the algorithm. For the purposes of these tests the MEGA algorithm implementation was executed with each of the elitism and the STIR/memetic components active or not.

The impact of the population size, already proven important in past experiments, in conjunction with the usage of the above components was also examined.

Table 6.2.1: Experimental design for the tests assesing the impact of MEGA custom components

Test	Objectives	Population	Iterations	Evolutionary Operations	Configurations
Test 1	Descriptor dissimilarity to Ibuproxam, molecular complexity	50, 100, 150	100, 250, 500	Mutation: 0.25 Crossover: 1.0	Niching: on Elitism, STIR: off (sMEGA) Niching, Elitism: on, STIR: off (eMEGA) Niching, Elitism, STIR: on (MEGA)
Test 2	Fuzzee dissimilarity to ER ligands, molecular complexity				
Test 3	Fuzzee similarity to ER-a ligands, Fuzzee dissimilarity to ER-b ligands				
Test 4	Fuzzee similarity to ER-a ligands, Fuzzee dissimilarity to ER-b ligands, molecular complexity				

The experimental design, summarized in Table 6.2.1 above, included experiments on two and three objectives and population size 50, 100 and 150. Five runs were performed for each input parameter combination to eliminate the chance of drawing conclusions on extreme sets of results produced due to serendipity. The number of iterations was set to 500 but, results were also assessed at 100 and 250 iterations. The choice of parameters for population size and maximum number of iterations took into account observations on the results of the single-objective experiments (see section 6.1 above) which indicated the adequacy of the chosen settings in enabling the identification of good quality solutions. The evolutionary operations of mutation (probability = 0.25) and crossover (probability = 1.0) were used throughout the runs. The maximum size allowed for a Pareto-archive set, controlling the number of solutions stored in the secondary population when elitism was used (see Fig. 5.1.B: Pareto Archive), was set to 1000. The niching mechanism was set to balance diversity between parameter and objective spaces. The STIR mechanism was activated when successive iterations, equal to 5% of the experiment iterations, produced identical hypervolume measure values since such behaviour may be interpreted as an indication of stagnation. The objectives used included similarity to Ibuproxam, similarity to ER-alpha ligands, similarity to ER-beta ligands and molecular complexity. Two methods described in the Materials section (5.3) were used for (dis)similarity calculations: the atom-pairs [KSF96] descriptor method and the Fuzzee shape-based technique [MOD09]. Performance assessment of the results from each run took place in a post-processing step that included the calculation of the Pareto-approximation set hypervolume, spacing and solution diversity (see section 2.3). Note that in the case of the bi-objective problems the hypervolume measure actually calculates the area dominated by the Pareto-front under assessment.

The simplest multi-objective tests, **Test 1** and **Test 2**, involved the design of molecules compromising similarity to query molecules and, molecular complexity. The two different test cases used Ibuproxam and a set of ER ligands respectively for the similarity objective implementations. The tests aimed at investigating the performance of various MEGA versions, or configurations, in a setting where both simple (complexity-based) and more complex (similarity-based) objectives need to be compromised. Note that in **Test 1** the objectives are not necessarily conflicting since Ibuproxam is a molecule with a relatively simple chemical structure. The remaining tests are substantially more difficult with **Test 3** compromising ligand-based similarity objectives to ER-beta and ER-alpha and **Test 4** adding a third objective, that of complexity, to the two similarity-based objectives of **Test 3**. We have purposefully chosen to allow the algorithm to search for the global Pareto-front, i.e., no hard filters were imposed to restrict the search space.

In all tests the initial population was chosen from the Dataset 2 described in the Materials section. In **Test 1** the 51123 subgraph gene collection was used with no weights applied since the weights available were not relevant to similarity to Ibuproxam. In **Tests 2, 3** and **4** the weighted 2363 collection was used. An account of the tests performed and the results obtained follows.

Figure 6.2.1 presents a typical view of the progress of a Pareto-approximation set during the execution of a MEGA run. Initial populations contain few nondominated solutions further from the ideal point. Later generations gradually approach the ideal point and include more solutions spanning a wider range of values.

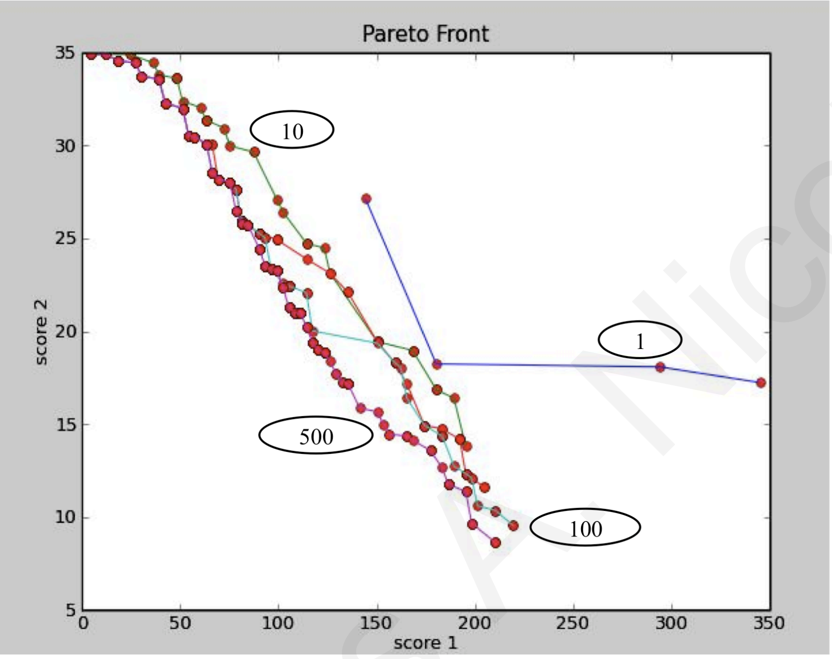


Fig. 6.2.1: Pareto-front approximations obtained with MEGA at iterations 1, 10, 50, 100, 500 for Test 1. Fronts 1,10,100 and 500 are labeled with the corresponding number. Later generations have a more advanced (closer to the ideal point – leftmost, bottom point) and dense front approximation. The x-axis corresponds to chemical graph complexity; the y-axis to descriptor dissimilarity/distance to Ibuproxam.

Solutions at the extremities of the Pareto-front produced represent individuals satisfying exclusively one of the objectives and therefore, in principle, should be comparable to the individuals obtained using a single-objective optimization process on each of the objectives. Tests comparing such extreme solutions with products of single-objective optimization runs have been performed and verified the ability of MEGA to satisfy each of the objectives on an individual basis. Following is a detailed presentation of the results produced by the four tests. The results produced by the first two tests, the simplest of the four attempted, bear significant similarities and are therefore presented together.

Figure. 6.2.2 compares the performance of the different algorithmic versions on all runs performed for **Test 2**, with population 50, using the hypervolume measure while 6.4.3 uses the spacing, and diversity measures to compare the algorithms. Results for different population sizes were almost identical.

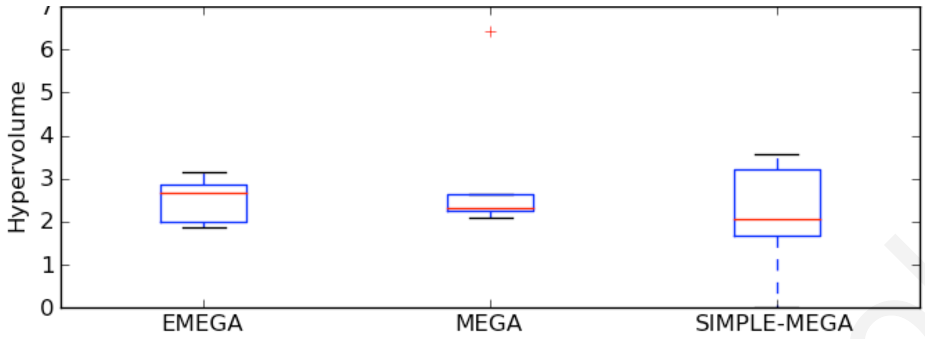


Fig. 6.2.2: Hypervolume performance for eMEGA, MEGA and simple-MEGA for population 50 on Test 2. Each boxplot depicts the range of hypervolume values obtained by the runs of the corresponding algorithm. Lower values correspond to better performance. The box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

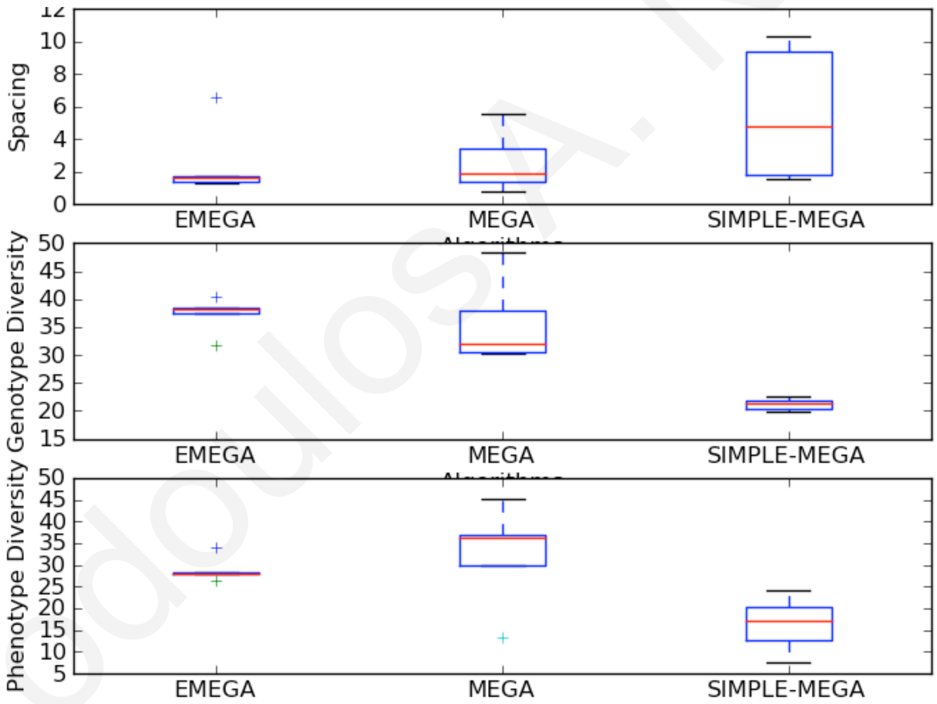


Fig. 6.2.3: Comparison of the Pareto-approximations produced by eMEGA, MEGA and simple-MEGA using spacing (top) and diversity in parameter/genotype space (middle) and in objective/phenotype (bottom) space for Test 2, population 50. Higher values correspond to better measure performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

As it can be clearly seen in Fig. 6.2.2 the hypervolume measure results for the three versions of the algorithm are of similar quality. Even simple-MEGA, the version of the algorithm not using elitism and the STIR mechanism, produces comparable results with the two other implementations. MEGA and eMEGA results are even more similar. Similar observations with respect to the similarity of the results produced by the three algorithms have been made for both **Test 1** and **Test 2** tests and varying population sizes. Table 6.2.2 confirms the lack of statistically significant differences among the results produced by the three MEGA versions as measured using the hypervolume measure for all population sizes attempted.

Table 6.2.2: Statistical significance analysis for the Hypervolume performance measure for runs of sMEGA, eMEGA and MEGA with population 50, 100, 150 on Tests 1, 2.

	eMEGA	**MEGA**	**sMEGA**
Test 1 – Population 50, 100, 150			
eMEGA		NS	NS
MEGA			NS
Test 2 – Population 50, 100, 150			
eMEGA		NS	NS
MEGA			NS

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant

Table 6.2.3: Statistical significance analysis for the spacing and diversity performance measures for runs of sMEGA, eMEGA and MEGA with population 50 on Test 2.

	eMEGA	**MEGA**	**sMEGA**
Spacing			
eMEGA		NS	NS
MEGA			NS
Genotype Diversity			
eMEGA		NS	S
MEGA			S
Phenotype Diversity			
eMEGA		NS	NS
MEGA			NS

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant

With respect to the spacing measure (Fig. 6.2.3 top) MEGA and eMEGA produce comparable results, while simple-MEGA seems to produce results widely varying in performance but, on average, better than the other two algorithm versions. Statistically, the results for the spacing measure are not significant. Results for population 50 runs are shown in Table 6.2.3. The results are representative of runs with all populations for Test 2.

In the diversity measures (Fig. 6.2.3 middle and bottom) sMEGA is the worst performer. For the phenotype diversity (Fig. 6.2.3 bottom), functioning in objective/phenotype space as is the spacing measure, MEGA and eMEGA seem to produce better results than sMEGA but, this conclusion is not supported by the statistical analysis. In genotype diversity (Fig. 6.2.3 middle), functioning in parameter space, sMEGA performs considerably worse than MEGA and eMEGA, a conclusion also supported by the statistical significance tests shown in Table 6.2.3. The different measure values for these three measures, observed despite the similar hypervolume performance of the three algorithms, may be related to the smaller count of solutions in the final Pareto approximation set produced by simple-MEGA. The reduced count facilitates larger spacing measure values along the Pareto-front surface. However, the significantly lower performance in phenotype diversity indicates that the distribution along the front may not be optimal while the lower performance in genotype diversity shows that the lack of a Pareto archive mechanism affects the structural diversity of the solutions found.

Figure 6.2.4 graphically presents Pareto-fronts produced by the three versions of the algorithm examined, MEGA, eMEGA and simple-MEGA, for the **Test 2** and provides additional insides. Specifically, one of the multiple runs performed for **Test 2** with population 50, for each of the three versions of the algorithm has been chosen randomly and visualized. As clearly seen the three runs produced near-identical results with only minimal differences between the algorithms tested. This pattern, i.e., the generation of near identical results for **Test 2** (and **Test 1**) by the three algorithms has been observed throughout the experiments performed and can be attributed to the relatively simpler nature of the problem to be solved. Tests attempted with a larger population, even though producing a more dense Pareto front, failed to advance the front closer to the optimal point and/or identify a set with wider extremities.

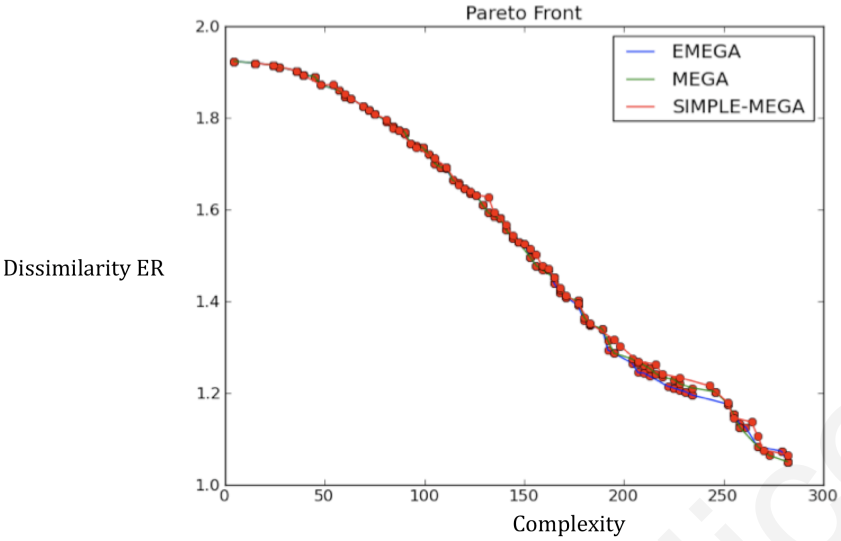


Fig. 6.2.4: The Pareto-fronts obtained with simple-MEGA (red line), eMEGA (blue line) and the fully enabled MEGA (green line) for population 100 on Test 2. Note the resemblance between the three fronts indicating near identical performance. The x-axis corresponds to chemical graph complexity; the y-axis to fuzzee dissimilarity to known ER ligands.

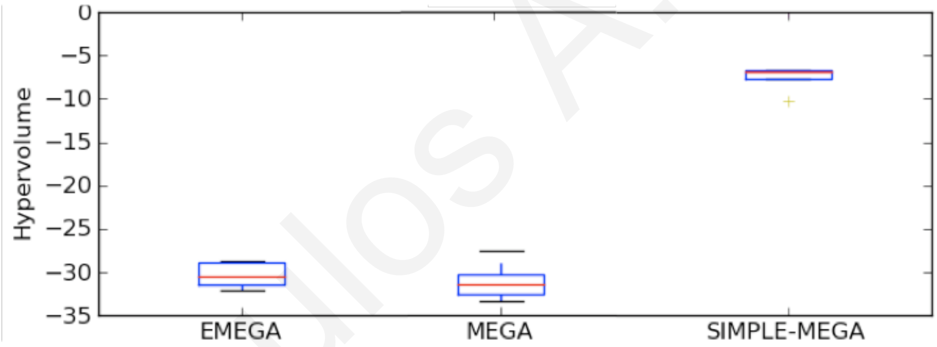


Fig. 6.2.5: Hypervolume performance for eMEGA, MEGA and simple-MEGA for population 50 on Test 3. Each box depicts the entire range of hypervolume values obtained by the runs of the corresponding algorithm. Lower values correspond to better performance. The box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

In the case of **Test 3**, a bi-objective problem optimizing shape similarity-based objectives on distinct sets of ligands, results indicate varying performance between the three algorithms. In this test, as in the following **Test 4**, sMEGA significantly lags when compared with MEGA and eMEGA. Note that **Test 3** and **4** are substantially harder than **Test 1** and **2** described previously requiring the search of a more complex space compromising more demanding objectives. The difference is immediately apparent in Fig. 6.2.5 (and Table 6.2.4). The performance of MEGA and eMEGA, although comparable also differs (note the difference in scale necessary to accommodate the huge deficiency of the sMEGA results).

Table 6.2.4: Statistical significance analysis for the Hypervolume performance measure for runs of sMEGA, eMEGA and MEGA with population 50, 100, 150 on Tests 3, 4.

	eMEGA	**MEGA**	**sMEGA**
Test 3 – Population 50, 100, 150			
eMEGA		NS	S
MEGA			S
Test 4 – Population 50, 100, 150			
eMEGA		NS	S
MEGA			S

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant

The results for **Test 4**, the hardest test for which results are presented in this dissertation, are shown in Fig. 6.2.6-6.2.7 below for population size 50 and 500 iterations. With respect to hypervolume simple-MEGA again shows significantly worse performance than either eMEGA or MEGA as is also verified in the statistical analysis presented in Table 6.2.4. This is attributed to the generation of Pareto approximation sets with fewer solutions further from the ideal point confined in a smaller range of values. eMEGA and MEGA produce comparable values for the hypervolume measure given the same input parameters indicating high similarities in the overall nature of the Pareto-fronts produced.

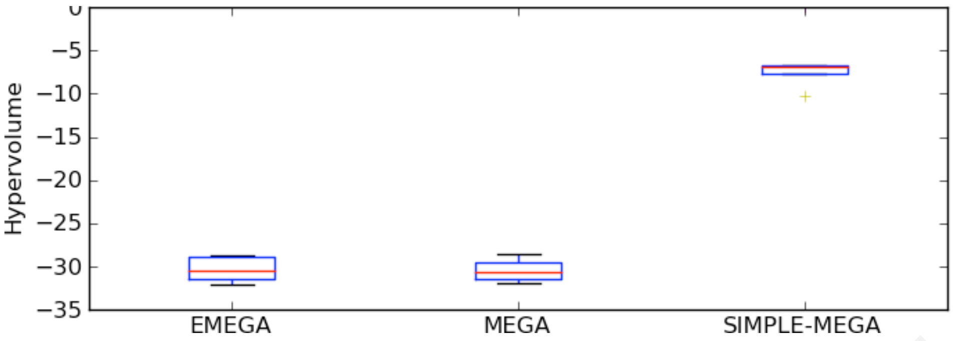


Fig. 6.2.6: Hypervolume performance for elitist-MEGA (eMEGA), MEGA, and simple-MEGA for population 50 on Test 4. Lower values correspond to better performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

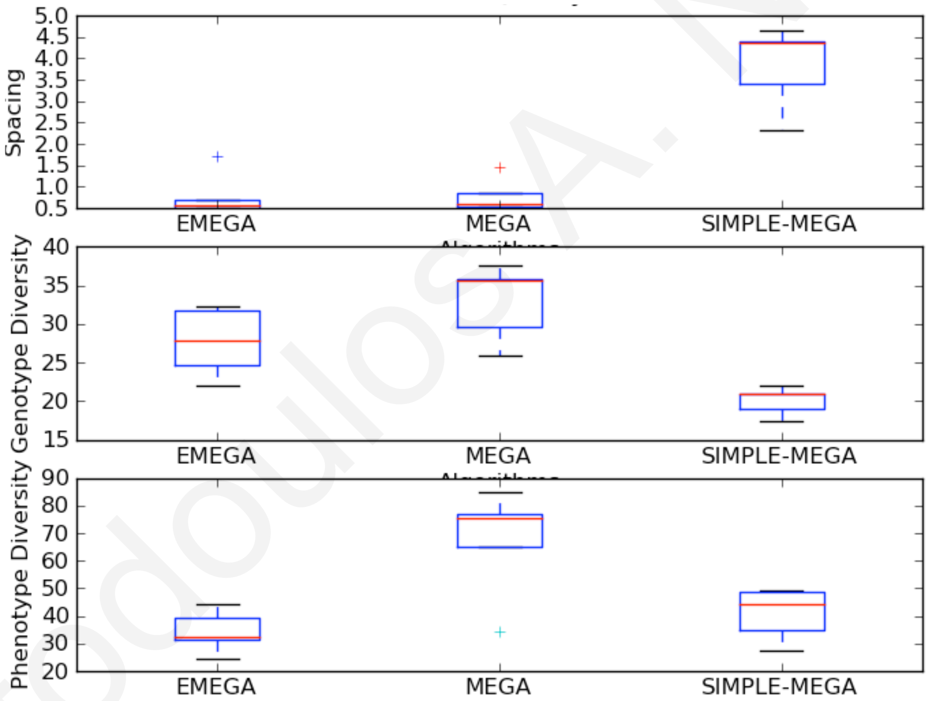


Fig. 6.2.7: Comparison of the Pareto-approximations produced by eMEGA, MEGA and simple-MEGA using spacing (top), diversity in parameter/genotype space (middle) and in objective/phenotype (bottom) space for Test 4 for population 50. Higher values correspond to better performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

MEGA and eMEGA produce comparable results also with respect to the spacing measure (Fig. 6.2.7 top). However, MEGA produces superior results in both diversity measures (Fig. 6.2.7 middle, bottom), indicating that the Pareto approximation sets it produces have a better distribution both in parameter and objective space. Once again, sMEGA results produce better spacing measures than the other two algorithms attributed to the significantly smaller number of solutions it identifies. Note that we have found no significant performance difference during the analysis of the results produced for different number of maximum iterations, i.e., when results are compared at 100, 250 or 500 iterations. Results for the spacing and the diversity measures were similar -almost identical- for **Test 3**.

The influence of ***population size*** on the performance of MEGA was also investigated through comparing the results produced by runs with different populations on the same problem. Figure 6.2.8 compares the quality of the results produced by runs of MEGA on **Test 2, 3** and **4** for population sizes 50, 100, 150 using the hypervolume measure.

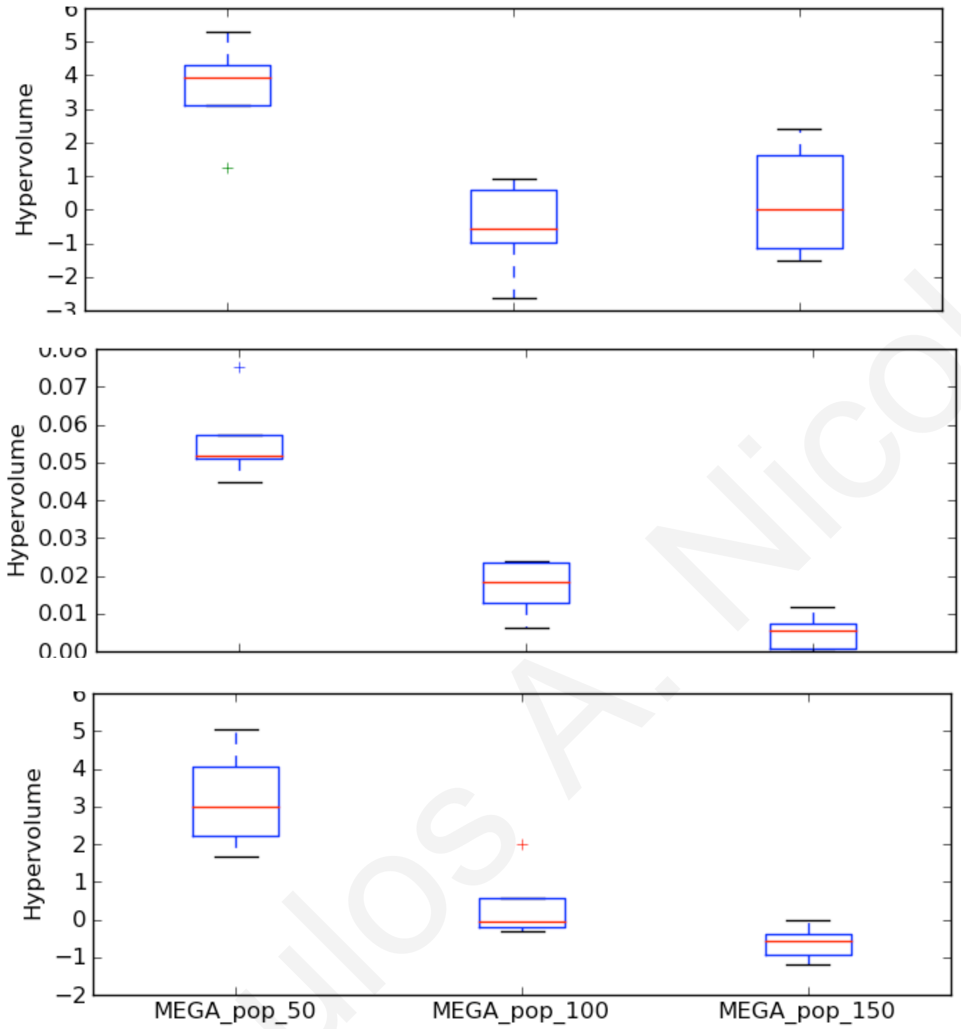


Fig. 6.2.8: Hypervolume performance for runs of MEGA for population 50, 100, 150 on Test 2 (top section), Test 3 (middle) and Test 4 (bottom). Lower values correspond to better performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

The results show that the population size can have a significant impact on the quality of the solutions produced given certain circumstances. For example, runs with smaller populations, e.g. 50 in the specific problem investigated, have worse hypervolume with respect to runs with larger populations sizes. However, this conclusion cannot be generalized since the hypervolume volume measures of the runs with 100 and 150 populations are sometimes comparable with the former having a slight edge (Fig. 6.2.8 top).

Specifically, for **Test 2** (and **Test 1**) results obtained with population 50 are worse than those obtained for population 100 and 150. However, no noticeable difference in performance is observed for the results of the runs with populations 100 and 150. On the contrary, in the case of the more difficult problems, **Test 3** and **4**, the increase of population size is associated with a steady trend of hypervolume measure improvement confirming the significant positive impact of increased populations. It is thus apparent that the benefit of using larger population size in MEGA is clearer for more difficult problems with a more complex solution space, while in problems with simpler solution spaces population size does not seem to influence the end result given a sufficient number of iterations for the runs to converge.

TABLE 6.2.5: STATISTICAL SIGNIFICANCE ANALYSIS FOR THE HYPERVOLUME PERFORMANCE FOR RUNS OF MEGA WITH POPULATION 50, 100, 150 ON TEST 2, 3, 4.

	Population 50	**Population 100**	**Population 150**
Test 2			
Population 50		NS	NS
Population 100			NS
Test 3			
Population 50		S	S
Population 100			NS
Test 4			
Population 50		S	S
Population 100			NS

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant

Table 6.2.5 presents the outcomes of the statistical analysis with the Mann-Whitney non-parametric test performed on **Tests 2,3** and **4** with varying population sizes. The results confirm that population size, in relation to the complexity of the problem investigated, can result in statistically significant differences in performance. Specifically, the results produced with populations 50, 100 and 150 for **Test 2** (Fig 6.2.8, top) do not differ significantly from a statistical perspective (at level 0.05) indicating that the smallest population tested may be sufficient for this simpler problem and that the use of larger populations is not justified by the qualitative improvement of the results. On the contrary, in the case of **Tests 3** and **4**, which are substantially more complex than **Test 2,** results obtained when using population size 100 and 150 are statistically significantly better than those obtained with population 50. However, differences between the results produced with population 100 and 150 are insignificant.

Figure 6.2.9 compares the final Pareto-approximation sets produced by simple-MEGA and MEGA on **Test 1** at 100 iterations. We have already shown that in the case of this simple bi-objective problem both algorithms reach approximately the same Pareto surface given 500 iterations, sufficient for the algorithms to converge (see Fig. 6.2.4). Overall, and despite the lack of niching, a Pareto-archive or the STIR mechanism in simple-MEGA, the two versions of the algorithm generate Pareto-fronts with several common solutions. Specifically, the solution set produced by simple-MEGA is almost a proper subset of the solution set produced by MEGA. However, the front produced by MEGA has a much wider spread, especially at fewer generations (i.e., 100 versus 500) and contains many more solutions. Additional results obtained for **Test 1** runs confirmed the ability of all versions of the algorithm used to converge to (or near) the real Pareto-front. The differences between the Pareto approximation sets generated by the various algorithms was found in the spread and density of the sets with more elaborate versions of the algorithm having an edge in performance. Similar observations have been made for the results of **Test 2** despite the higher difficulty associated with identifying solutions for that problem.

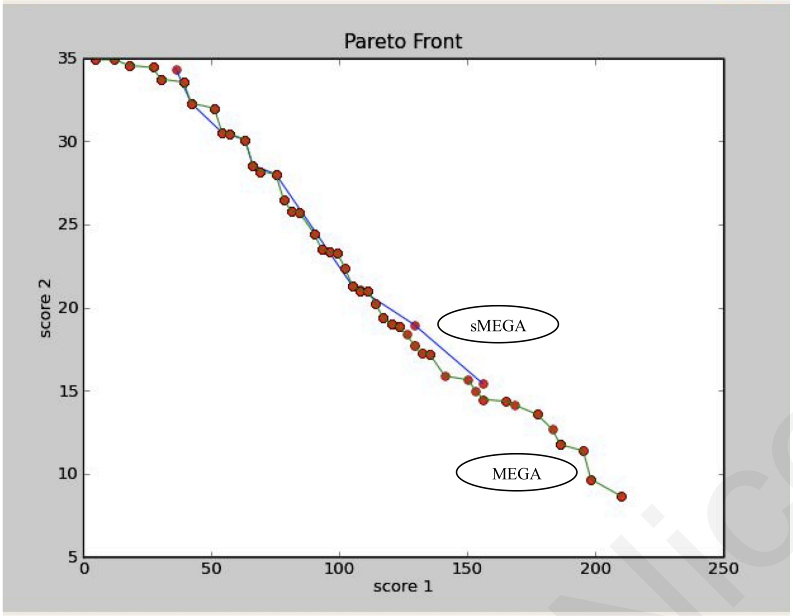


Fig. 6.2.9: A comparison between the Pareto-front obtained with MEGA (dots connected with line marked with MEGA (green)) and simple-MEGA (dots connected with line marked with sMEGA (blue)) for Test 1 at 100 iterations. The MEGA front is denser with larger spread. The x-axis corresponds to chemical graph complexity; the y-axis to descriptor dissimilarity/distance to Ibuproxam.

Overall, MEGA runs resulted in solutions with more diverse genotypes, i.e., chemical structures, than runs with simple-MEGA in all tests included in Table 6.2.1. The density of the Pareto-front, i.e., the number of Pareto-solutions in between the two extremes, was heavily influenced by the use of elitism in the form of the Pareto-archive. In simple-MEGA the size of the final Pareto-front was limited by the user-defined population size. Note that despite this limitation, the final Pareto-front produced may slightly exceed the population size depending on the number of non-dominated solutions produced in the final population since the method retains all best performing solutions until after the termination conditions are checked. In runs with MEGA proper where no such limitations exist due to the storage of the entire set of non-dominated solutions found over all iterations in the Pareto-archive, a larger number of non-dominated solutions, exceeding the user-defined population size, was produced. Since no hard-filters have been used we obtained a wealth of solutions ranging, for example in the case of **Test 1**, from the minimum possible complexity and maximum possible dissimilarity to the other extreme of maximum complexity and minimum possible dissimilarity. Typically, when the user-defined population size is too small, for example 10 or 20 (attempted in tests described in section 6.3), the number of non-dominated solutions in the Pareto-archive reaches the user-defined population size more quickly than when the population size is larger.

A general observation is that the application of simple-MEGA occasionally shows problems with niching and lack of ability to produce diverse solutions despite the enabled niching mechanism. The use of the niching mechanism clearly works in identifying a diverse set of nondominated solutions from a given population, but in later iterations of the optimization search, as more and more points on the Pareto-front are produced the algorithm necessarily drops important solutions representing whole regions of the solution space. This effect results in genetic-drift, a behaviour closely related to niching, and loss of diversity. The use of the Pareto-archive manages to preserve diversity and achieve convergence in each execution run. The additional use of the STIR mechanism provides an added performance benefit, especially in obtaining Pareto approximation sets with a better distribution in objective space and a higher diversity in parameter space.

6.3 MEGA Application in Estrogen Receptor Selectivity

In a test specifically designed in close consultation with computational and medicinal chemistry experts, MEGA was used to design molecules exhibiting selectivity between two target receptors. The collaborating experts (Chemistry Department, National and Kapodestrian University of Athens) provided assistance in the experimental design and set-up, and in the qualitative evaluation of the results produced. Note that the results presented in this section and the related discussion section in the next chapter (section 7.3), have been published in [NAP09].

The goal in the ER selectivity experiments is to discover molecules with high binding affinity towards a known target receptor site and low binding activity to another receptor. For this specific test case we used ER-beta (PDB code: 2fsz1) as the "positive" target and ER-alpha (PDB code: 1xpc), a closely related target as a "negative" target. The description files for both receptors were pre-processed and prepared for computational modeling by the collaborating experts. In addition to the two primary optimization objectives several others were used as hard filters. Specifically, similarity to a known ER-alpha ligand, Tamoxifen, has been encoded via the similarity scorer mechanism described in the Materials section and applied as a hard filter to constrain the chemical structures designed to those exhibiting Tanimoto similarity to Tamoxifen greater than 0.4. The similarity objective has been applied to favor the design of more familiar chemical structures that would facilitate validation of the results by human experts. A collection of chemical structure scorers, i.e., molecular weight, hydrogen-bond donors and acceptors, and number of rotatable bonds, were also used as hard filters. These scorers, set to values in line with the Rule-of-Five [LC97] for oral bioavailability, were applied as hard filters in each generation to remove potentially problematic designs from further consideration. In the case where the application of the hard filters resulted in fewer solutions than required by the algorithm for parent selection the filters were ignored. This only proved necessary in very few instances during the initial iterations of the algorithm.

TABLE 6.3.1: EXPERIMENTAL DESIGN FOR THE QUALITATIVE ASSESSMENT TESTS OF MEGA

Test	Objectives	Population	Iterations	Evolutionary Operations
Test 1	Predicted high binding affinity to ER-b, predicted low binding affinity to ER-a, similarity to Tamoxifen (hard filter)	10, 20, 50, 100	20, 50, 100	Mutation: 0.25 Crossover: 1.0

The experimental settings for the MOOP tests provided for population sizes 10, 20, 50 and 100 and, 100 iterations. Three types of evolutionary operation combinations were tested with mutation only, crossover only, and both mutation and crossover applied. Mutation probability was set at 0.25 and crossover at 1.0. Runs were performed for each settings combination. Results were assessed after 20, 50 and 100 iterations. For each test case the initial population was selected from a user-defined data set. Specifically, the initial population was sampled in a quasi-random fashion from Dataset 2. The method used atom-type descriptors [KSF96] and the Tanimoto measure [WBD98] to calculate the similarity of all compounds to Tamoxifen and then selected randomly from the subset of compounds having similarity values less than 0.4 to Tamoxifen. This guaranteed that no member of the initial population would be similar to a molecule known to bind strongly to the target receptors. The 2363-collection of subgraph genes (see section 5.3) was used. Runs were performed with MEGA and the simpler, simple-MEGA version of the algorithm with the diversity mechanism for niching on and elitism and STIR disabled. The receptor-based MOOP selectivity tests carry a heavy computational burden due to the multiple, repeated docking experiments performed at each generation. Indicatively, a MEGA run with the settings described above and population 20, 100 iterations took approximately 40 hours on Machine 1 described section 5.3.

Apart from the solutions removed during the hard filtering step all other designs were kept. This allowed the algorithm to search for the entire Pareto-front compromising the targeted objectives. As a result we obtained solutions ranging from one extreme, i.e., high binding affinity to both receptors, to the other extreme, i.e., reduced binding affinity to both receptors. In between the two extremes several compromising solutions were obtained approximating the ideal point (Fig. 6.3.1).

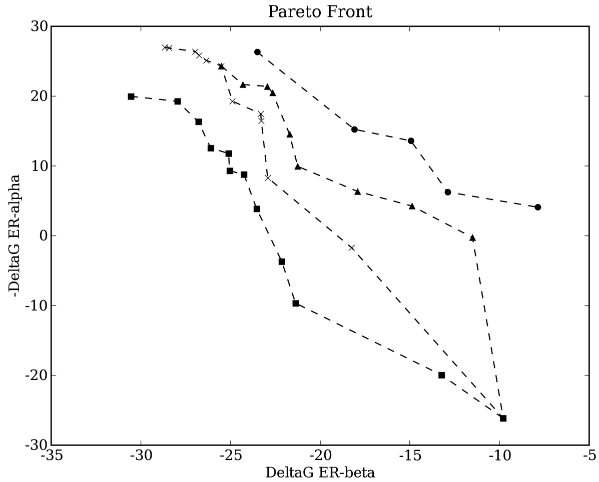


Fig. 6.3.1. Pareto-front approximations with population 10 at generations 1, 20, 50, 100 (lines with circles, triangles, x's, squares respectively). The objective function results were transformed for clarity to depict two minimization objectives. The x-axis corresponds to the predicted interaction score to ER-beta; the y-axis to the inverted predicted interaction score to ER-alpha. Later generations tend to "move" the approximated Pareto-front towards the ideal-point (bottom left). The run shown uses the simple-MEGA implementation and thus the number of Pareto solutions is limited by the user defined population size.

As expected, MEGA runs resulted in sets of solutions with diverse genotypes, i.e., chemical structures. The density of the Pareto-front, i.e., the number of Pareto-solutions in between the two extremes, was heavily influenced by the population size and the number of iterations performed. Typically, the final Pareto approximation set produced by a MEGA run contained a large number of non-dominated solutions, far exceeding the user-defined population size. Indicatively, a typical run of MEGA with a population of 20, 50 iterations using both mutation and crossover generated a Pareto-front consisting of 23 non-dominated solutions. In runs with simple-MEGA, where no archive exists, a smaller Pareto approximation set is produced, with the population size being the upper limit to the number of solutions found. Note that despite this limitation, the final Pareto-front produced may slightly exceed the population size depending on the number of non-dominated solutions produced in the final population since simple-MEGA retains all best performing solutions until after the termination conditions are checked. An example of a final Pareto-front produced by a simple-MEGA run with population size 10 consisting of 12 non-dominated solutions is shown in Fig. 6.3.1. In general, in runs with too small user-defined population size, for example 10 or 20, the number of non-dominated solutions in the Pareto-archive reaches the user-defined population size more quickly than when the population size is larger.

The solutions in the final Pareto-front provide a global view of the possible solution space to the user and enable him/her to make their selections a´ posteriori based on their preferences and goals. Although all the solutions of the generated Pareto-front satisfy the multiple hard filters applied and represent different compromises of the two objectives used to guide the optimization, i.e., docking scores towards ER-beta and ER-alpha, the interesting region of the Pareto-front for the problem under consideration is found where solutions exhibit high docking affinity to ER-beta and substantially lower affinity to ER-alpha. For the test run described above, with elitism enabled and a resulting Pareto-front of 23 solutions, the interesting region contained 14 structures. Several of these structures have reasonable, interesting chemical designs and may serve as idea generators for further development following expert validation, while others are less promising due to reasons such as relatively poorer selectivity performance or potential structural issues. Further analysis via visual inspection and Tanimoto similarity calculations revealed that the 14 structures belong to four structural groups. Figure 6.3.2 presents Tamoxifen and a group of four chemical structures, one from each of the identified structural groups. Each of the structures shown is characterized by the highest predicted selectivity potential within its group. Note the diversity of the structures, a result of the niching mechanism used by MEGA.

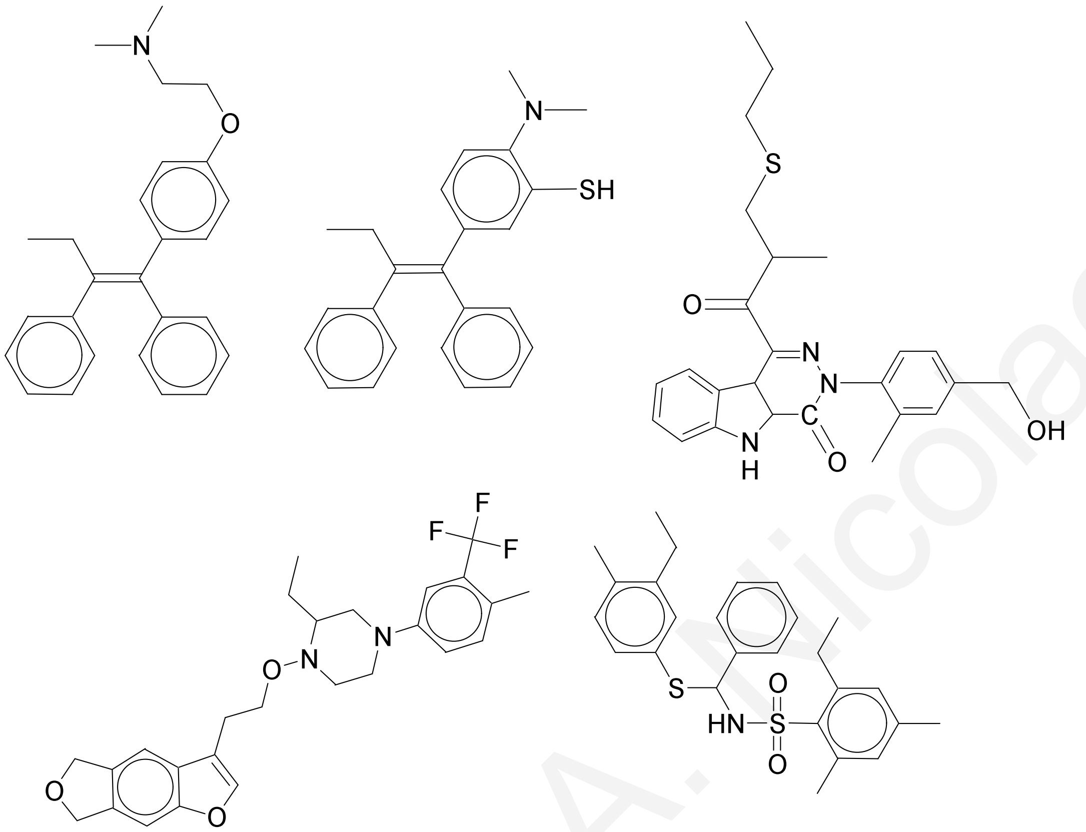


Fig. 6.3.2. Tamoxifen (upper left) and a group of four chemical structures designed by MEGA during a run performed for the MOOP validation tests. The four structures are representative of the section of the Pareto-front where solutions exhibit increased docking affinity to ER-beta than ER-alpha and are sorted according to predicted selectivity potential.

Figure 6.3.3 presents one of the structures, the most similar to Tamoxifen, docked in the ER-beta and ER-alpha pockets. The predicted selectivity of the structure, as explained from the docking pose in the two receptors may be due to the presence of the sulphur atom, labelled with ‘S’, which causes collisions in the “neck” of the ER-alpha and therefore forces the molecule to be docked in a non-optimal manner. In contrast, the proposed molecule docks nicely in the ER-beta receptor due to the different morphology of the pocket and especially the shorter, more accommodating “neck”.

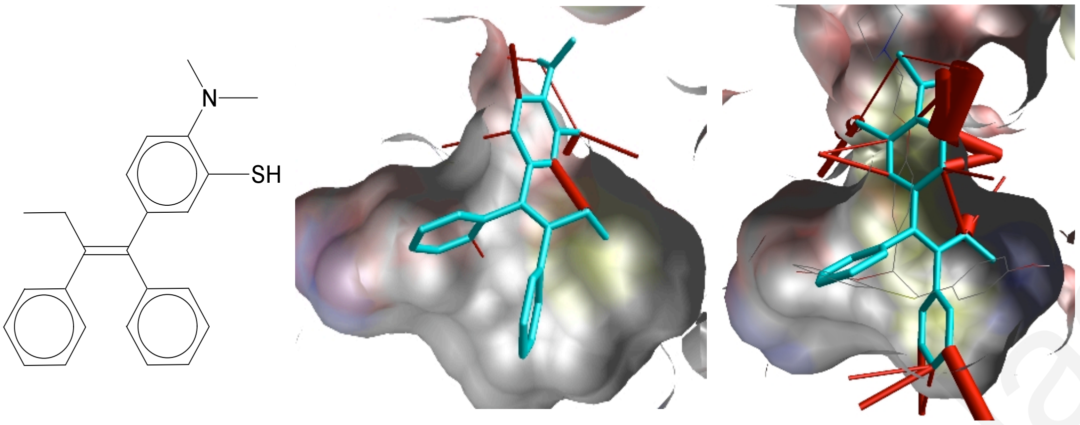


Fig. 6.3.3. Sample chemical structure proposed by the algorithm. Left: The sample design in 2D format. Center: The sample docked in ER-beta. Right: The sample design docked in ER-alpha. Note that the proposed design causes several collisions (red barrels) in ER-alpha.

6.4 MEGA Versus Established MOEAs

A major experimental part for the testing of MEGA involved the comparison of the performance of the algorithm against two established, commonly used MOEA algorithms, MOGA and SPEA. MOGA [FF98] was selected since it is one of the earliest, most commonly cited algorithms in the MOOP field. SPEA [ZLB04] is a more recent method that popularized the use of the Pareto-archive in MOEA algorithms. The comparative tests of MEGA, MOGA and SPEA included multi-objective problems with varying degree of difficulty. The experimental design, aiming to quantitatively define the performance of MEGA is described below.

TABLE 6.4.1: EXPERIMENTAL DESIGN FOR THE TESTS COMPARING MEGA TO ESTABLISHED MOEAS

Test	Objectives	Population	Iterations	Evolutionary Operations
Test 1	Fuzzee similarity to ER-a ligands, Fuzzee dissimilarity to ER-b ligands	50, 100, 150	100, 250, 500	Mutation: 0.25 Crossover: 1.0
Test 2	Fuzzee dissimilarity to ER-a ligands, molecular complexity			

The experiments performed applied MOGA, SPEA and MEGA on two multi-objective problems, the design of selective ERs, i.e., ligands that bind to ER-β and not ER-α and the design of ER-alpha inhibitors with reduced complexity. The experimental settings used population sizes 50, 100 and 150, and 500 iterations. Results were assessed at 100, 250 and 500 iterations. For each combination of input parameter settings five runs were performed, using different initial population sets. In each test case the initial population was selected from a user-defined data set. Runs were performed using both mutation and crossover. Mutation probability was set at 0.25 and crossover at 1.0. In the case of MEGA and SPEA the maximum size of the Pareto-archive was set to 1000. The objectives used included similarity to ER-alpha ligands, similarity to ER-beta ligands and molecular complexity [BC01]. The Fuzzee [MOD09] tool, described in the Materials section (section 5.3), was used for similarity calculation. Performance assessment of the results from each run took place through a post-processing step that included the calculation of the Pareto-approximation set hypervolume [ZT99], spacing [CS04] and the solution set diversity [TTW97] as described in section 2.3. The former in fact corresponds to the area dominated by the PAS assessed since the test cases are bi-objective while the latter has been calculated by averaging the Euclidean distances of all pairs of solutions in the proposed set, using the atom-pair descriptors [KSF96] of the molecules involved. MEGA niching was set to balance between the diversity in parameter and objective space. The experimental design is summarized in Table 6.4.1. An account of the tests performed and the results obtained follows.

The two tests described in Table 6.4.1 have different degrees of difficulty. **Test 2** seeks chemical structures that are similar in shape and properties to a set of ER ligands and exhibit low graph complexity. The latter objective is quite easy to achieve, e.g. by decreasing the size of the graph or by creating linear graphs, while the former requires the more difficult task of designing structures similar to known ligands. Consequently, **Test 1** that optimizes two similarity-based objectives is a more difficult problem to solve.

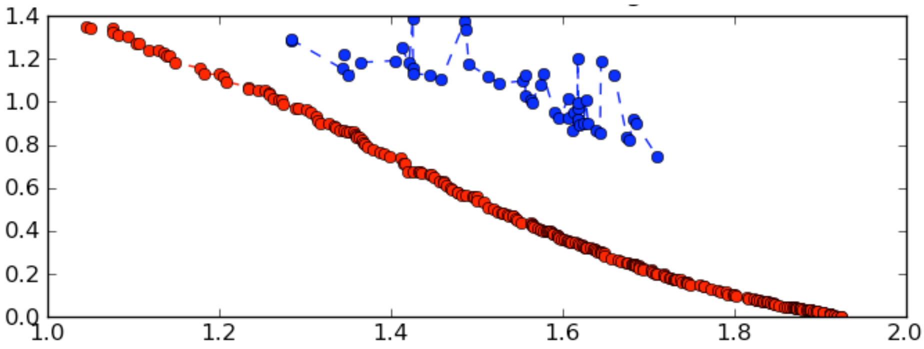


Fig. 6.4.1: Sample Pareto approximation set produced by MEGA for Test 1. The X and Y axes represent the two objectives, dissimilarity to ER-β ligands and similarity to ER-α ligands respectively. Note that both objectives need to be minimized. The upper (blue) dots connected with a (blue) dashed line represent the starting population. The lower (red) dots connected with a (red) dashed line represent the individuals of the final Pareto Front.

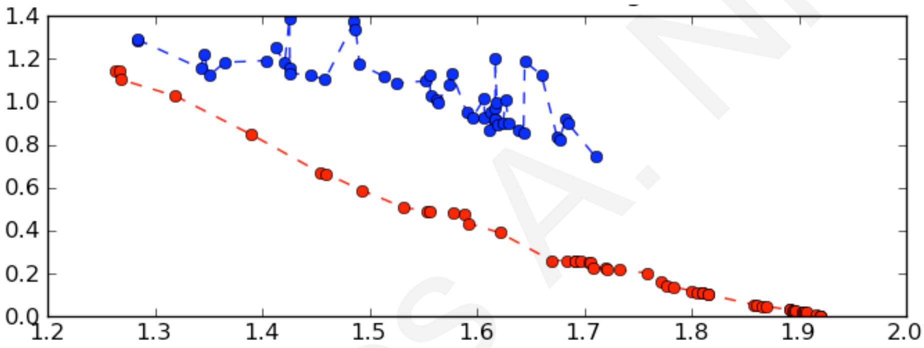


Fig. 6.4.2: Sample Pareto approximation set produced by MOGA for Test 1. The X and Y axes represent the two objectives, dissimilarity to ER-β ligands and similarity to ER-α ligands respectively. Note that both objectives need to be minimized. The upper (blue) dots connected with a (blue) dashed line represent the starting population. The lower (red) dots connected with a (red) dashed line represent the individuals of the final Pareto Front.

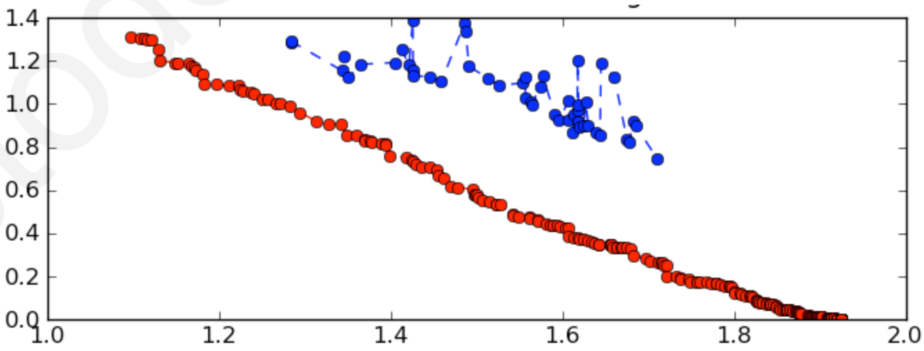


Fig. 6.4.3: Sample Pareto approximation set produced by SPEA for Test 1. The X and Y axes represent the two objectives, dissimilarity to ER-β ligands and similarity to ER-α ligands respectively. Note that both objectives need to be minimized. The upper (blue) dots connected with a (blue) dashed line represent the starting population. The lower (red) dots connected with a (red) dashed line represent the individuals of the final Pareto Front.

In Figures 6.4.1-6.4.3 Pareto-approximation sets produced by MEGA, MOGA and SPEA for **Test 1**, with population 50 and 500 iterations, are presented. The sets are indicative of the type of results produced by the three algorithms for one of the numerous runs performed for **Test 1**, the more complex test, with the parameter combinations described in the experimental design. Note that the upper (blue) dots and line represent the initial population, which, for the examples presented is identical. As it can be clearly seen, the MOGA run produces a less populated Pareto front than both MEGA and SPEA that can be attributed to the lack of a Pareto archive and, thus, the limitation on the size of non-dominated solution set imposed by the user defined population size. The Pareto fronts produced by the latter algorithms are comparable although MEGA seems to have a slightly more dense and extended front.

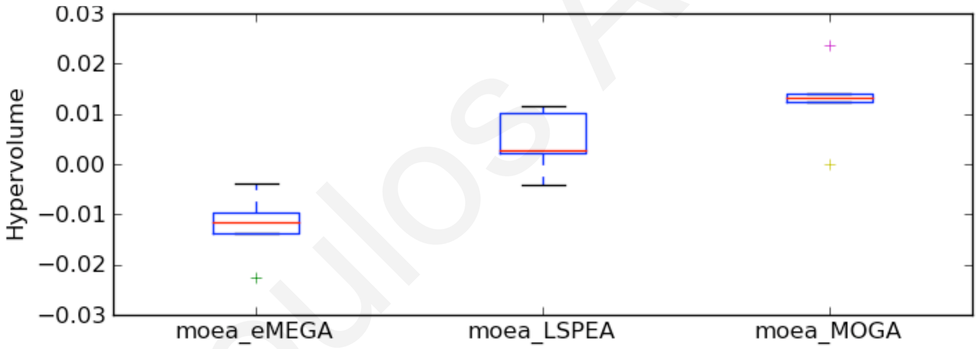


Fig. 6.4.4: Hypervolume performance for MEGA (eMEGA), SPEA (LSPEA), and MOGA for population 150 on Test 1. Each boxplot depicts the range of hypervolume values obtained by the runs of the corresponding algorithm. Lower values correspond to better performance. The box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

Figures 6.4.4 and 6.4.5 present a cumulative view of the results produced that facilitate performance comparison across multiple runs. Figure 6.4.4 presents the hypervolume measure box plots for MEGA, MOGA and SPEA for all five runs performed for **Test 1** with population 150 and 500 iterations.

Table 6.4.2 presents the outcomes of the Mann-Whitney non-parametric test performed on **Test 1** with varying population sizes. The results confirm that there are statistically significant differences in the performance of the three methods for this problem. Specifically, the results produced with populations 50, 100 and 150 for **Test 1** differ significantly from a statistical perspective (at level 0.05) indicating that MEGA outperforms both SPEA and MOGA, and that SPEA has an overall edge in performance over MOGA especially when smaller populations are used.

Table 6.4.2: Statistical significance analysis for the Hypervolume performance for runs of MEGA, SPEA and MOGA with population 50, 100, 150 on Test 1.

	MEGA	**SPEA**	**MOGA**
Population 50			
MEGA		S	S
SPEA			S
Population 100			
MEGA		S	S
SPEA			NS
Population 150			
MEGA		S	S
SPEA			NS

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant

The hypervolume indicating the area (since the problem investigated is bi-objective) dominated by the Pareto-fronts of the MEGA algorithm is substantially larger than those produced by SPEA and MOGA, with the latter exhibiting the worst performance (lower values correspond to larger hypervolume and thus better performance). Note that the nadir point, crucial to the calculation of the hypervolume, has been carefully chosen and is identical across the various runs of the three algorithms to enable comparisons. This conclusion is of special importance since better performance in hypervolume indicates Pareto-fronts closer to the optimal point that, in essence, is the primary indication of the success of an MOEA and, in general, any Pareto-based optimization run. Of interest is the consistency in hypervolume values obtained in different runs for both MEGA and MOGA indicating that the algorithms are capable of reaching results of similar quality despite different starting populations, all other input parameters being equal.

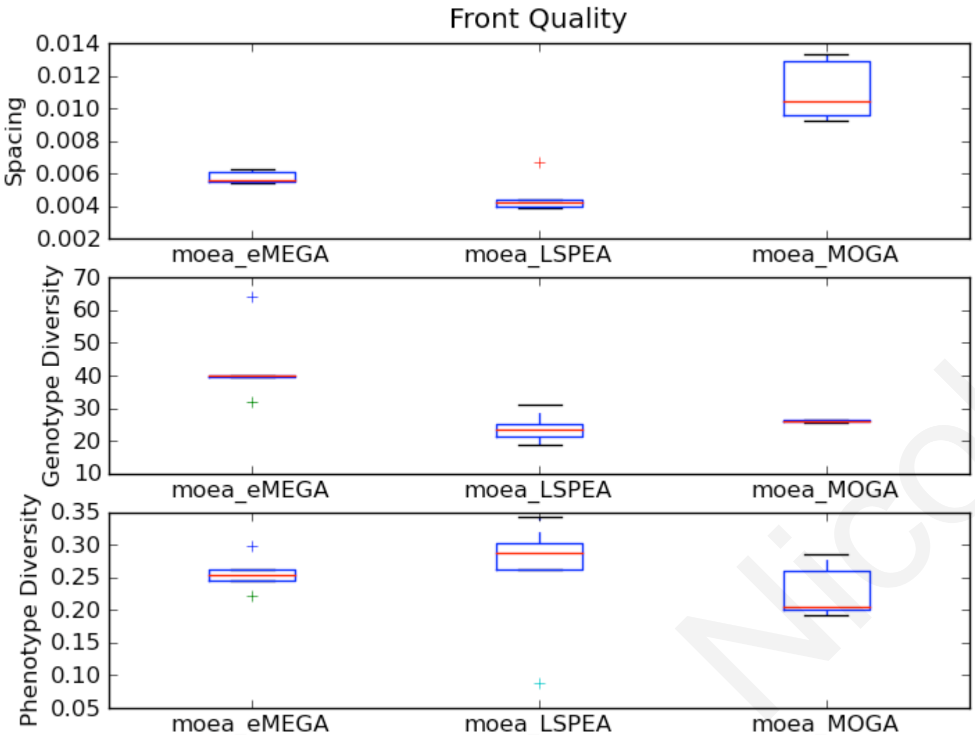


Fig. 6.4.5: Comparison of the Pareto-approximations produced by MEGA, SPEA and MOGA using spacing (top) and diversity in parameter/genotype space (middle) and in objective/phenotype (bottom) space for Test 1 for population 150. Higher values correspond to better measure performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

Figure 6.4.5 presents the results of additional performance measures for the three algorithms, namely the spacing and the diversity in parameter and objective space, for all five runs performed for **Test 1** with population 150 and 500 iterations. The plot clearly shows that MEGA, with its unique niching mechanism specifically designed to promote structural diversity, produces solution sets with increased diversity in parameter space, i.e., individuals with diverse graph structures (see Fig. 6.4.5, middle). The sets proposed by MOGA and SPEA, show comparable decision space diversity performance but at a much lower level than MEGA. The performance of the three algorithms in the objective space diversity measure is roughly comparable with SPEA producing slightly better performance followed by MEGA, however, at a non-statistically significant level (see Table 6.4.3). MOGA generates Pareto-sets more evenly spread across the front identified as measured by the spacing measure. These conclusions are also supported by the statistical analysis results shown in Table 6.4.3. Similar observations, indicating the consistently superior results produced by MEGA were made for all population sizes attempted for **Test 1**.

TABLE 6.4.3: STATISTICAL SIGNIFICANCE ANALYSIS FOR THE SPACING AND DIVERSITY PERFORMANCE MEASURES FOR RUNS OF MEGA, SPEA AND MOGA WITH POPULATION 150 ON TEST 1.

	MEGA	**SPEA**	**MOGA**
Spacing			
MEGA		NS	S
SPEA			S
Genotype Diversity			
MEGA		S	S
SPEA			NS
Phenotype Diversity			
MEGA		NS	NS
SPEA			NS

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant

Results differed for the simpler **Test 2** that searches the solution space compromising a ligand-based similarity objective and the easier complexity objective. Figure 6.4.6 presents the hypervolume measure for the three algorithms for the specific run with population 50 and 500 iterations on **Test 2**. Note the resemblance with Fig. 6.4.4 showing the hypervolume measure for the three algorithms for **Test 2**. Similar results were obtained for the spacing and diversity measures as well.

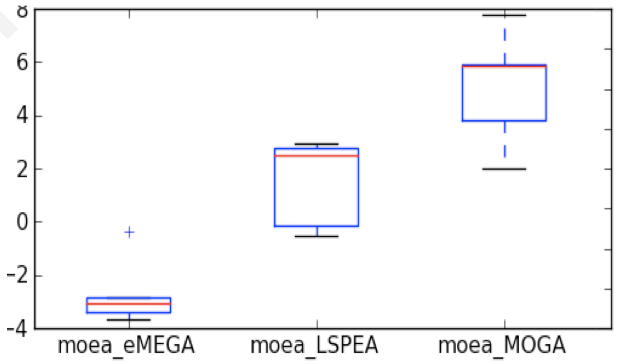


Fig. 6.4.6: Hypervolume performance for MEGA (eMEGA), SPEA (LSPEA), and MOGA for population 50 on Test 2. Low values correspond to better performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

Figure 6.4.7 presents the plot for population 100 for the same experiment. In contrast to the results for the case study with population 50, the hypervolume measure performance for the three algorithms is showing great similarities. Results for population size 150 where almost identical, with negligible performance difference, indicating that **Test 2** can be solved by all three algorithms tested for the purposes of this research for population 100 and 150 and 500 iterations. This conclusion is supported by the results of the Mann-Whitney test shown in table 6.4.4. The analysis can only detect statistically significant differences in the performance of MEGA versus the other two methods for population 50 only. For larger populations, i.e., 100 and 150, performance is comparable.

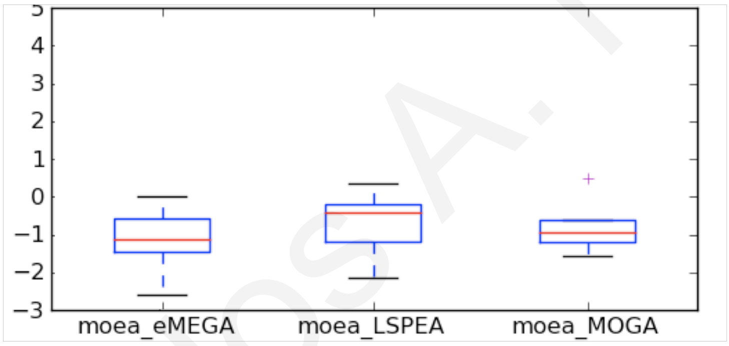


Fig. 6.4.7: Hypervolume performance for MEGA (eMEGA), SPEA (LSPEA), and MOGA for population 100 on Test 2. Low values correspond to better performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

TABLE 6.4.4: STATISTICAL SIGNIFICANCE ANALYSIS FOR THE HYPERVOLUME PERFORMANCE FOR RUNS OF MEGA, SPEA AND MOGA WITH POPULATION 50, 100, 150 ON TEST 2.

	MEGA	**SPEA**	**MOGA**
Population 50			
MEGA		S	S
SPEA			NS
Population 100			
MEGA		NS	NS
SPEA			NS
Population 150			
MEGA		NS	NS
SPEA			NS

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant

It is worth noting that despite the near-identical hypervolume performance MEGA still exhibits significantly better genotype diversity as shown in Fig. 6.4.8 below. This feature indicates that although all three algorithms produce Pareto approximation sets that converge to the true Pareto-front and are highly similar in objective space (as also seen from the performance of the spacing measure in the figure below), the niching mechanism incorporated in MEGA contributes to the generation of more diverse solution sets in decision space.

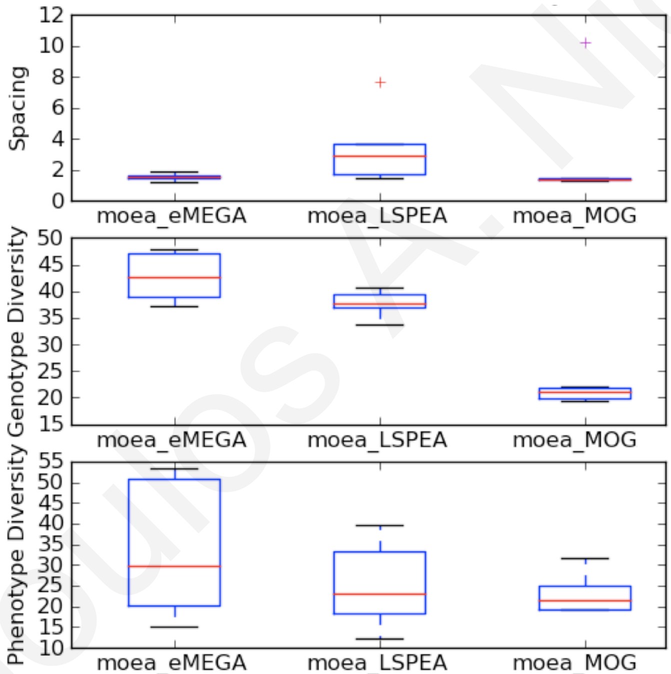


Fig. 6.4.8: Comparison of the Pareto-approximations produced by MEGA, SPEA and MOGA using spacing (top) and diversity in parameter/genotype space (middle) and in objective/phenotype (bottom) space for Test 2 for population 100. Higher values correspond to better performance. Each box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers.

Overall, time requirements for the execution of the runs were sufficiently reasonable. A typical run of MEGA with population 50 and 500 iterations took approximately 40 minutes on Machine 1 (see section 5.4). For comparison purposes, our implementations of MOGA and SPEA took roughly 70% and 80% respectively of the time required by MEGA. The increase in time requirements for the experiments attempted was linear with respect to the population size and the number of iterations.

6.5 Parallel MEGA Tests

This section briefly summarizes experimental results from the execution of MEGA versus its parallel, process-enabled version of pMEGA. The design and implementation of pMEGA has been the subject of [K10]. Additional experiments and results can be found in [KNP09], [K10]. The experimental results presented investigate a major feature of pMEGA that affects its behaviour, specifically the number of subpopulations used. A second feature affecting pMEGA's performance, the isolation time, i.e., the number of iterations for which the subpopulations evolve independently, was set to a default value of 10% of the maximum number of iterations.

Table 6.5.1: Experimental design for the evaluation of parallel MEGA Versus MEGA

Test	Objectives	MEGA type	Population	Number of Subpopulations	System CPU	Iterations	Evol. Operations
1	Descriptor similarity to ER-a ligands, descriptor dissimilarity to ER-b ligands	MEGA	100, 150, 200	1	2-core	200	Mutation: 0.25 Crossover: 1.0
2		pMEGA		4, 8			
3		MEGA		1	4-core		
4		pMEGA		4, 8			

The initial population was taken from the compounds in Dataset 2 and the set of 2663 subgraph genes was used, obtained as described in section 5.3, subsection Materials. The experimental setup provided for runs with MEGA and pMEGA with two different subpopulation sizes (4, 8) and three different population sizes (100, 150, 200). Each of the above experimental settings was run five times, each time using a different initial population. The same set of experiments was performed on dual and quad core computers (see section 5.3, Machine 1 and 2 description) and the results obtained were compared to identify the effect of the usage of a different subpopulation set on the quality of the Pareto approximation set produced and, the speed-up achieved. The two objectives used were based on the calculation of the atom-pair descriptor [KSF96] similarity to three ligands that are known to be selective to ER-β and, dissimilarity on two ligands known to be selective to ER-α. As in previous experiments described above, in order to represent the problem as a bi-objective minimization problem, similarity is calculated using the Tanimoto coefficient and dissimilarity using the Soergel distance measure as described in section 5.3. Smaller Tanimoto coefficient values indicate less similarity between the patterns compared with zero being the least similar/completely different. Conversely, a Soergel value of zero indicates identical input patterns while higher values point to lower similarity.

Figure 6.5.1 is a graphical demonstration of the solutions in the Pareto-approximation set of a run of the pMEGA algorithm with four subpopulations on a quad-core processor system and population 200. The X and Y-axes represent the two objectives. The upper (blue) dots on the (blue) dashed line represent the individuals of the starting working population. The lower (red) dots represent the individuals of the working populations over a period of iterations.

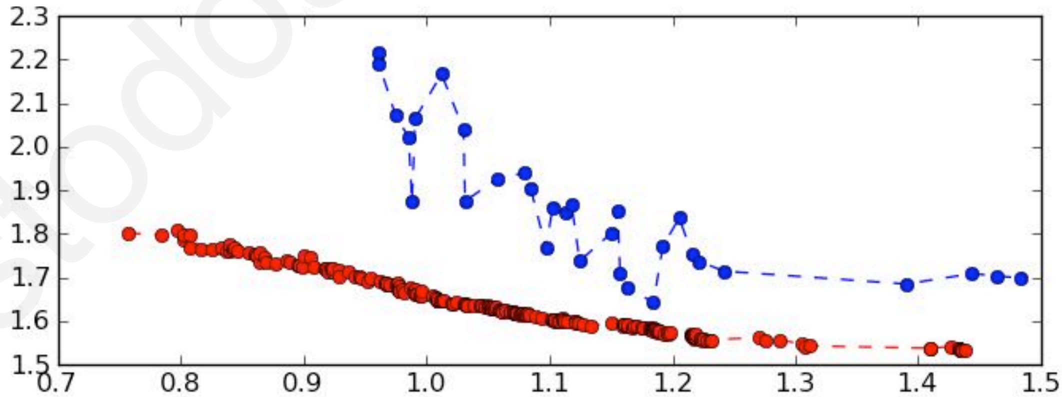


Fig. 6.5.1: A graph of Pareto fronts taken from one of the runs for pMEGA with four subpopulations. The X and Y axes represent the two objectives, dissimilarity to ER-β ligands and similarity to ER-α ligands respectively. Note that both objectives need to be minimized. The upper (blue) dots connected with a (blue) dashed line represent the starting population. The lower (red) dots connected with a (red) dashed line represent the individuals of the final Pareto Front.

Figure 6.5.2 and 6.5.3 compare the Pareto approximation sets generated during the runs of MEGA and pMEGA with four and eight subpopulations on a quad-core computer with population size 200. The results are compared using the hypervolume (Fig. 6.5.2) and the spacing, genotype diversity and phenotype diversity (Fig. 6.5.3) quantitative performance measures described in section 2.3. Figure 6.5.2 presents the hypervolume (or hyper-area since the test problem is bi-objective) measure values for the five runs of each of the three algorithm implementations tested using box-plots. The values indicate that pMEGA with four subpopulations consistently produces Pareto approximation sets with seemingly a slightly better hypervolume measure, i.e., compromise surfaces that are closer to the ideal point (or conversely, further from the nadir point) than MEGA. The performance of MEGA (as indicated by the range of vales and the red, median line) is comparable with the performance of the pMEGA with eight subpopulations. However, from a statistical analysis perspective, the runs of MEGA and pMEGA with four and eight subpopulations produced results with non-significant differences (see Table 6.5.2). Results and conclusions were identical for the runs on the dual-core system as well.

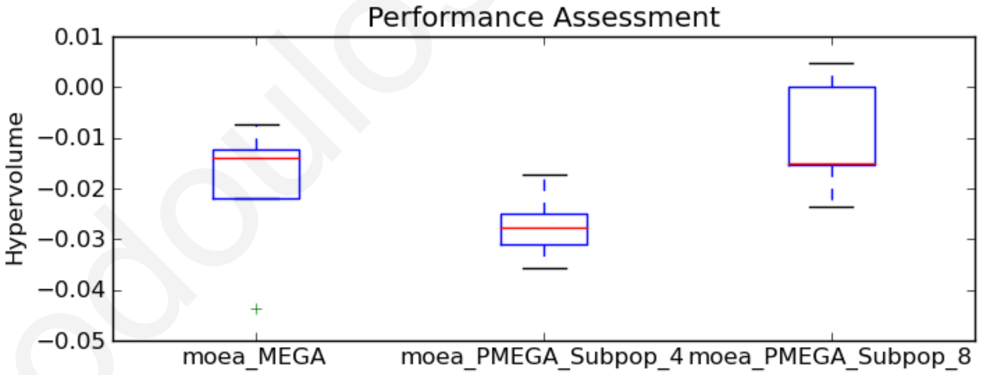


Fig. 6.5.2: A comparison of the Pareto fronts produced by MEGA, pMEGA with four subpopulations and pMEGA with eight subpopulations for population 200 on a quad-core system using the hypervolume performance measure. Each boxplot corresponds to the measure values of the runs executed by the corresponding algorithm with the same input parameters but different initial population. The box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers with values beyond the whiskers. For the hypervolume measure lower values correspond to better performance.

Table 6.5.2: Statistical significance analysis for the Hypervolume performance for runs of MEGA and pMEGA with population 100, 150, 200 on Tests 3, 4.

	MEGA	**pMEGA-4**	**pMEGA-8**
Population 100			
MEGA		NS	NS
pMEGA-4			NS
Population 150			
MEGA		NS	NS
pMEGA-4			NS
Population 200			
MEGA		NS	NS
pMEGA-4			NS

Statistical test used: Mann-Whitney test; S: Statistically significant at 0.05 level; NS: Non-statistically significant; pMEGA-4: parallel MEGA with four subpopulations; pMEGA-8: parallel MEGA with eight subpopulations;

The results presented in the plots in Fig. 6.5.3 indicate the similarity of the Pareto sets produced, on average, in terms of the spacing and diversity performance measures used. Specifically, the Pareto solutions have comparable distribution on the compromise surface for the three algorithm implementations used. Similarly, the results from the three implementations exhibit similar solution diversity both in parameter (genotype) and objective (phenotype) space. It is worth noting that this conclusion, as well as the one related to the hypervolume measure, has been generally observed for all population sizes used (100, 150, 200) and for both subpopulations numbers (4,8).

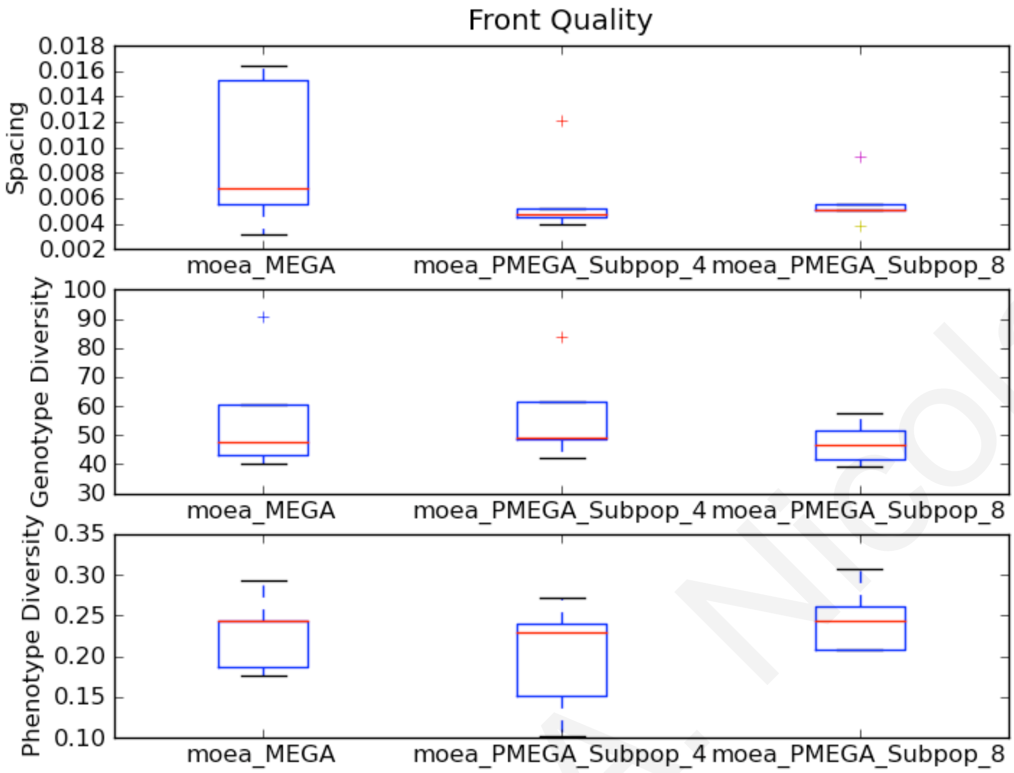
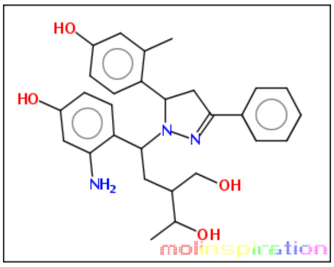
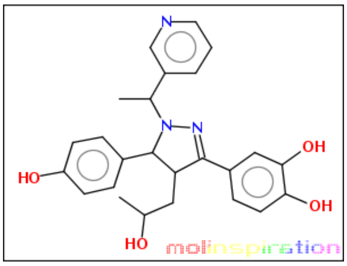
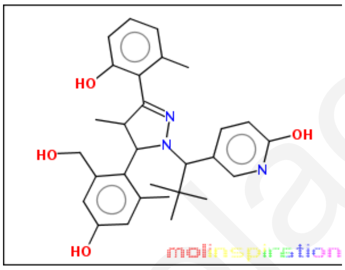
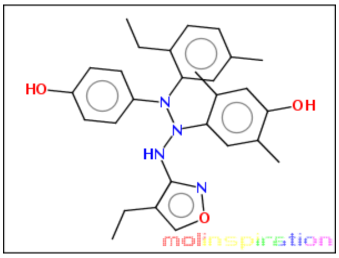
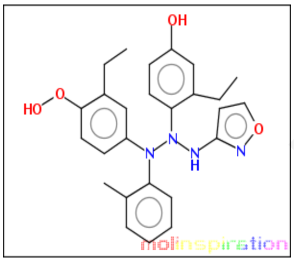
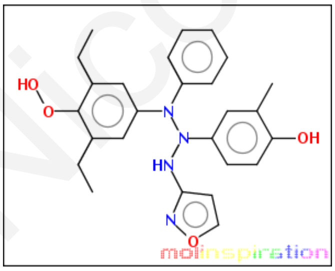


Fig. 6.5.3: A comparison of the Pareto fronts produced by MEGA, pMEGA with four subpopulations and pMEGA with eight subpopulations for population 200. Each boxplot corresponds to the measure values of the five runs executed by the corresponding algorithm with the same input parameters but different initial population. The box extends from the lower to the upper quartile values of the data, with a line at the median. The whiskers extend from the box to show the range of the data. Crosses indicate possible outliers. For all three measures higher values indicate better performance.

The graphs presented in Fig. 6.5.3 show a general trend of the solutions over all experiments performed with the aim to compare pMEGA to MEGA and evaluate the performance of the former. An alternative view of the results, placing emphasis on the similarity of individual solutions produced by each algorithm, has also been investigated. Table 6.5.3 below presents a small subset of the solutions generated from the three sets of experiments, each column corresponding to one of the implementations. The solutions have been designed using the online depiction tool provided by MolInspiration [MI09]. The samples have been chosen by visual inspection of the sets produced by the author.

TABLE 6.5.3: SAMPLE OF PROPOSED SOLUTIONS BY MEGA AND PMEGA

MEGA	pMEGA (4 Subpopulations)	pMEGA (8 subpopulations)
		
		

As far as performance improvement on time requirements, Table 6.5.4 summarizes the execution times of the experiment performed with population size 200 on a quad-core computer system (see section 5.3, Machine 2). The three columns correspond to the time obtained by the MEGA, pMEGA using four subpopulations each with 50 individuals and, pMEGA with eight subpopulations each with 25 individuals.

TABLE 6.5.4: A SUMMARY OF THE TIME REQUIREMENTS FOR MEGA AND PMEGA ON A QUAD-CORE SYSTEM

	Time Measured (HH:MM:SS)		
	MEGA	pMEGA 4 Subpop	pMEGA 8 Subpop
Max Time	02:00:38	00:48:17	00:42:43
Min Time	01:50:20	00:35:38	00:40:22
Average	01:56:45	00:42:39	00:41:45
Speedup		2.74	2.80
Efficiency		0.68	0.70

Time measured is wall clock time (total execution time). Average is calculated as (Total – Max – Min) / (5 - 2). Speedup is calculated as Time of serial / Time of parallel. Efficiency is calculated as Speedup / Number of processes.

Table 6.5.5 shows a summary of all the experiments performed on the system with the quad-core CPU. Using the measured speedup observed in all the experiments the average speedup is calculated to be about 2.7 and the efficiency 0.7 which is undoubtedly substantial given the lack of any fine-grained parallelization of the implementation and the limited modifications required to the algorithm itself.

TABLE 6.5.5: SUMMARY TABLE FOR THE EXPERIMENTS HELD ON THE QUAD-CORE SYSTEM

		MEGA	pMEGA 4 Subpopulations	pMEGA 8 Subpopulations
Population 100, Iterations 200	Average Execution Time	0:55:57	0:20:53	0:21:27
	Speedup		2.68	2.61
	Efficiency		0.67	0.65
Population 150, Iterations 200	Average Execution Time	1:26:18	0:32:02	0:32:12
	Speedup		2.69	2.68
	Efficiency		0.67	0.67
Population 200, Iterations 200	Average Execution Time	1:56:45	0:42:39	0:41:45
	Speedup		2.74	2.80
	Efficiency		0.68	0.70

Chapter 7

Discussion

The specific test case used for assessing the performance of MEGA is that of de novo design, i.e., the design of chemical structures with favorable biological properties. The objective functions guiding the search have been implemented by developing models for the identification of the target specific chemical space (binding site-based and ligand-based) and simpler rules encoding the established knowledge about the chemical properties of drugs and the notion of privileged subgraph genes, i.e., chemical fragments that are positively correlated with a favorable biological profile of the compounds containing them. Several experiments have been performed aiming, among others, to investigate the influence of the various components of MEGA on its performance and to measure and compare the performance of the MEGA, MOGA and SPEA algorithms on the instances of the graph design problem described previously. Tests using MOGA and SPEA served to assess the performance of MEGA in comparison to commonly used algorithms from the MOOP field. The results showed that MEGA compares favourably with these algorithms and fulfils the initial objectives set for the de novo design and general optimal graph design problem. These conclusions were quantitatively confirmed using formal Pareto-front performance measures. Qualitative validation of the results produced by MEGA has been provided by collaborating experts for one of the selectivity experiments performed.

7.1 Effectiveness of MEGA Evolutionary Operators

The first set of experiments designed for MEGA aimed to verify the intended functionality of the implementation and assess the effectiveness of the evolutionary operators of the algorithm. For this reason the experiments performed concerned single objective optimization problems with relatively easy to assess objectives such as similarity to a target molecule and increased predicted binding affinity to a target receptor. Results indicated that the MEGA algorithm easily meets these objectives since throughout the multiple runs performed the method was able to consistently identify solutions meeting the single objective. For example, in the experiments using the compound similarity objective, the method could design compounds similar to the target molecule starting from entirely different graph designs every single time. These results, combined with the overall behaviour of MEGA on the single-objective tests demonstrate the ability of the algorithm to explore the chemical space given a clear objective to use for solution scoring. The tests verified the correct functionality of the graph-based evolutionary operations and the niching mechanism. It was also observed that the latter, i.e., the diversity-based niching mechanism, did not really affect the quality of the single best solution produced. The implemented objectives, both the ligand, similarity-based, and the receptor, docking-based proved to be functioning according to expectations and be suitable to guide the search process.

Furthermore, our tests demonstrated the importance of the population size used which had a substantial effect on the quality of the solutions produced. This is probably true because the size of the population provides a greater, more diverse sample of the global search space to the algorithm to start with, and, a breeding environment more representative of the global search space. The number of iterations was also of substantial importance, however, it could not compensate for the effect of an overly small population size. Tests with varying number of maximum iterations indicated that the method could converge to solutions satisfying the criteria imposed at 250-500 iterations for the similarity objective and approximately 100 for the docking objective provided that the population size matched the complexity of the search space. These indications were used in setting the maximum number of iterations in subsequent, multi-objective experiments. Finally, our findings highlighted the problem of over fitting associated with excessive training and the focusing of the search process on a single objective. Results such as those produced for the docking based experiments, for example Fig. 6.1.4 where the solution proposed satisfies the objective but is unworthy of further investigation due to its size and complexity, confirm that failure to take into account any other objectives to guide the optimization, or as hard filters, may lead to over fit individuals that suffer from overspecialization to the objective under consideration.

7.2 Impact of MEGA Components

In the evolutionary mechanism nature seems to have found a robust optimization method capable of adapting species to their environment and thus ensuring their survival. The effectiveness of natural selection has living proofs in the innumerable variations of life in our world. The method works simply by exploiting large populations that strive to satisfy their needs and instincts in an environment that has limited resources and, often, lacks the capacity to accommodate all existing needs. The resulting competition is applied at the level of individuals, where fitter individuals have first pick on the available resources and therefore a better survival chance, as well as at the level of species, where whole population groups compete with one another in an effort to accommodate their needs and, in some cases, exert a collective effort for the benefit of their species. The fitness of an individual to its environment is key in determining its success in surviving and passing its genes to the next generation of individuals. Consequently, generation after generation is born by reproduction of fitter parents gradually leading to populations better adapted to the environmental conditions of their world. In this setting it is obvious that the ability of a species to adapt is key in the natural selection. However, and despite its proven robustness, evolution may not lead to the best individual [D76]. Rather, it converges on individuals that are good enough to survive, prosper and reproduce and so ensure the survival of the species. This lack of perfection may be necessary in the real world where the balance of species is necessary to avoid the domination of specific species and the extermination of others. However, from an optimization perspective this lack of perfection essentially means that the adaptability of species could be sped up, or even lead to better individuals, by interventions to the evolutionary process that, for example, would recognize and eliminate a dead-end evolutionary path or exploit knowledge related to the environment and apply appropriate selection pressure. The MEGA framework implements several novelties in an effort to improve the search process through exploiting available knowledge, ensuring population diversity and facilitating self-adaptation among others. The following sections discuss these components of the algorithm.

7.2.1 Exploitation of Knowledge

This research has aimed to contribute in filling the current gap in Optimal Graph Design, a hard multi-objective combinatorial problem commonly found in the real world. Nature-inspired multi-objective population-based algorithms have been increasingly popular in the last decade and have been proven to produce good results in ill-defined problems, problems with large complex search spaces, problems with dynamic environments, etc. However, the possibility of exploiting available knowledge either from pre-existing information on the nature of the problem, or knowledge gained during the optimization process has not received as much attention. The method presented attempts to do exactly this by customizing the evolutionary process through the incorporation of features such as information rich subgraph genes, monitoring of the progress of the population of solutions and applying local search to improve select subsets of individuals similar in parameter space. From an implementation perspective, the method facilitates the inclusion of additional objectives encoding problem-specific knowledge through the provision of a well-defined programmer interface.

According to the No-Free-Lunch (NFL) theorem, no search-based optimization algorithm is better than another when performance is averaged over all possible classes of problems [WM97]. However, the theorem does not eliminate the possibility of an appropriately designed custom algorithm to exhibit better performance compared to other optimization algorithms for specific problem cases. It is a commonly held belief, partly derived from the results produced by custom knowledge or data-driven algorithms, that incorporating knowledge into the computational analysis can improve performance and lead to results of better quality or equally good results in less time. Intuitively, this expectation can be justified by arguing that the incorporation of problem-specific knowledge, as is done in many algorithmic designs, will facilitate the search by intelligently limiting the space to the most promising regions and lead to improved performance. More formally, an extension of the NFL theorem has been proposed for multi-objective optimization in [K03]. This implies that on average, each multi-objective optimization algorithm has the same performance when applied to all possible problems, provided that no a priori knowledge of the problem is assumed [CLV07].

In the field of computational optimization, the success of this approach can be attested by the achievements of Memetic Algorithms that have been proven to outperform general-purpose metaheuristics including Evolutionary Algorithms. As previously reported, MAs extend EAs by incorporating into the process a local search component that attempts to search more intensely specific regions of the search space. The use of local search within a MOEA is an interesting topic that has been only scarcely studied. Although memetic MOEAs [M89] have existed for some time, most of the hybridizations between a MOEA and a local search mechanism are relatively straightforward algorithmic designs, lacking a careful analysis of the trade-offs involved [CLV07]. In MEGA, the STIR component combines self-adaptive with memetic elements to invoke local search only when stagnation conditions occur. This novel -to the best of our knowledge- mechanism was shown to have clear benefits to the quality of the Pareto-front produced at the expense of some additional computational cost. However, in contrast to the majority of the existing methods where the memetic component is an integral part of the evolutionary cycle or is regularly invoked regardless of the optimization process status [CLV07], STIR intelligently applies local search when most needed, i.e., when the EA-driven global search is facing problems in finding new solutions.

Our work follows a self-adaptive paradigm [BMK03] to develop an optimization method for graph design, a multi-objective combinatorial optimization problem with numerous real-life incarnations. The nature of the problem, i.e., the graph representation of the solutions, combined with domain specific information related to the characteristics of the problem examined, i.e., the de novo design problem, and the objectives to be met, provide several opportunities for knowledge inclusion into the method implementation. The results presented in chapter 6 verify the potential of knowledge inclusion and the superiority of appropriately designed MAs when compared to EAs. Firstly, the tests presented in section 6.2 (see Fig. 6.2.5-6.2.7) showed that the use of local search in MEGA, through the STIR mechanism, consistently improves the quality of the solution set with respect to the hypervolume, the genotype diversity and the phenotype diversity measures, all other conditions being equal. Secondly, MEGA was proven to outperform both MOGA and SPEA which are among the most popular MOEA algorithms used currently (see Fig. 6.4.4-6.4.6). Note that the above conclusions hold for the more complex tests attempted for the purposes of this dissertation whereas in the case of simpler tests all MOEA methods, as well as MEGA configurations, were able to produce Pareto approximation sets with small, often negligible, differences in performance. This observation indicates the suitability of MEGA, and appropriately designed MAs in general, to address complex multi-objective combinatorial optimization problems through incorporating domain-specific knowledge and local-search techniques.

The computational cost related to the usage of STIR depends heavily on the nature of the specific problem investigated, the implemented objectives and the input parameters provided by the user that indirectly control the frequency of activation of the mechanism. As such safe conclusions can only be drawn on specific application problems. As an indication, MEGA required approximately 25% more time than SPEA or eMEGA for the complex test cases described in section 6.2 and 6.4. This difference generally reflects the computational overhead of the STIR mechanism in complex optimization processes where it is frequently invoked.

7.2.2 Diversity-Niching in Population-based Algorithms and MEGA

The influence of a diverse population is known to be crucial to the quality of solutions obtained in population optimization algorithms [CLV07], [DS07]. In the MOEA field, where the solution consists of a set of compromise solutions, diversity is even more important. Several schemata for diversity preservation have been suggested, however the overwhelming majority of them tend to focus on diversity in objective space and ignore diversity in parameter space. This maybe acceptable in single-objective optimization search problems where the goal is to detect the single best solution possible, or in multi-objective algorithms where one objective is more important than others and so the solution sought needs to have the best possible fitness to that specific objective. However, in many real life multi-objective problems, including optimal graph design instances, the identification of numerous equivalent solutions with differing chromosomes, e.g. graph structures, is highly desirable since they represent alternative solutions compromising the objectives in different ways. In such cases, users have the ability to analyze a´ posteriori the solution set produced and select the one that better fits the objective set imposed.

The important issue of population diversity in EAs (and MAs) is given substantial attention in MEGA. In EAs the population provides the main reservoir of genetic material from which to produce new solutions and, therefore, the population is expected, to the degree possible, to simultaneously occupy different parts of the search space [GSM04]. The genetic drift phenomenon, a result of stochastic errors associated with genetic operators, causes EAs to converge to a single solution when used with a finite population [CLV07]. Lack of diversity in the population essentially means that the process is unable to explore globally the feasible search space, which in turn may lead to increased chances of getting stuck in suboptimal local minima and failure to produce sufficiently good solutions. The consequences of this phenomenon may not be apparent or even seriously problematic in normal single-objective optimization processes, however, when multiple solutions are desired as in multi-objective problems it leads to suboptimal solution sets. For example, loss of population diversity in multi-objective optimization may cause that the population remains stack in areas far from the true Pareto front or that individuals are located only in selected areas of the Pareto front. Moreover, in the case of multi-objective problems with many local Pareto frontiers the loss of population diversity may result in locating only a local Pareto frontier instead of a global one [DS07].

The concept of diversity, as well as its preservation, can be applied to the decision or objective space. In the decision space, population diversity is a function of the similarity between genotypes whereas in the objective space it is based in the relation among phenotype vectors. Promoting all kinds of diversity during the entire search process is thought to potentially be counter-productive and the type and amount of diversity required at different times remains unclear [GSM04]. Typical MOEA applications apply diversity preservation in the objective space regardless of the specific method used for this purpose. However, in principle, diversity should be applied to the space of interest to each specific application. Phenotype diversity should be used if one is attempting to obtain a uniform a Pareto approximation set while genotype diversity, facilitating the generation of multiple equivalent solutions in phenotype space with different chromosome representation, may be more appropriate in other situations including several real world problems [CLV07]. Efforts in line with the latter approach include the Genetic Diversity Evaluation Method, which considers a distance-based measure of genetic diversity as a real objective in fitness assignment [TB03].

Several diversity preservation mechanisms have been proposed, originally in the SOOP GA field, including “fitness sharing” and “crowding” and the EA’s using them to either maintain or introduce diversity seem to have better chances of succeeding [ZWA06]. The crowding operator, suggested by Holland [HJ75] aims to identify concentrations of similar individuals and remedy this situation by replacing some of them with new offspring. Similarity between individuals is measured in genotype space. Goldberg and Richardson [GR87] used a different approach in which the population is divided in different subpopulations according to the similarity of the individuals [CLV07]. The method relies on a fitness sharing function that ultimately adjusts the fitness of each individual depending on the density of its neighborhood. In the MOEA field the approaches proposed to preserve population diversity include fitness sharing and niching, clustering, the use of geographically-based schemata to distribute solutions and the use of entropy and mating restriction schemes, among others [KJ02].

The algorithm proposed in this dissertation introduces a novel niching method that ensures both, diversity in decision space, in this case structurally dissimilar graphs, and diversity in objective space so that an adequate representation of the Pareto-front is achieved. In addition to balancing between the two space domains the diversity method introduced can be tuned so as to favor either one of the two. The key to this method is the graph-based clustering step applied on the chromosomes and the assignment of each solution to a cluster. The cluster label of each solution is subsequently used during selection to balance the effect of choosing just by using efficiency measures based solely on the Pareto-rank of the solutions. Of special interest is that the niching mechanism introduced, guarantees that even in cases where one of the objectives is "easier" than others and may easily produce multiple solutions with competitive fitness values, the effect of using chromosome-based clustering ensures that the few solutions advancing the "difficult" objective(s) get equal treatment and opportunities to evolve. Note that the proposed niching methodology, which has been applied in MEGA on solutions represented using graph chromosomes, can be applicable to EAs using other representation schemata, selection operators, etc.

Overall, the MEGA niching method has been shown capable of maintaining diversity in both decision and objective space as was the initial goal set for the mechanism. The experiments presented in section 6.4 comparing MEGA with MOGA and SPEA that use niching mechanisms exclusively focused on objective space, serve to showcase the advantages of the technique proposed by MEGA. The results obtained indicate that MEGA produces solutions with higher genotype diversity than MOGA and SPEA in all runs performed, even in the case of simpler problems where other performance measures, and especially the hypervolume measure, have similar outcomes. Of interest is the observation that the implemented MEGA niching technique produces results that are comparable with those of MOGA and SPEA with respect to diversity in the objective space despite the fact that the latter two algorithms place emphasis on this type of diversity. This encouraging result suggests that genotype diversity can, and probably should, be taken into account since it does not necessarily result in solutions with low objective space diversity. Balancing between diversity in the two spaces still remains an open question and is likely to only be solvable in a problem-specific manner. For the de novo design problem examined in the present work we opted to use a scheme that ensured the selection of at least an individual from each of the genotype-based solutions clusters as explained in chapter 5. This resulted in the preservation of different chemical scaffolds in the population and their continued influence during the generation cycle.

7.2.3 Impact of Elitism

Our experimental results also confirmed the major positive influence of elitism as implemented by a Pareto-archive as reported by researchers before [Z99], [CLV07], [CS04]. Storing all solutions discovered in previous iterations of the process benefits the final Pareto approximation set produced both, with respect to the quantity of solutions and to the range covered by the extremes of the front. Population diversity also benefitted since no solutions, no-matter when they were found or how unique they are, were dropped due to algorithmic or implementation specific limitations.

In particular, the use of elitism in combination with the niching mechanism, provided a larger number of solutions covering a more extended range of compromises. In comparison, the application of simple-MEGA (niching: on; elitism: off) resulted not only in less dense Pareto-fronts as expected, but also in Pareto-fronts of comparatively limited span. In Fig. 7.2.1 the final solution sets produced by eMEGA and simple-MEGA which only differ in the use of elitism or not (respectively) are shown. The experiment involved the design of selective chemical structures with high estimated binding (i.e., interaction score) to ER-β and low estimated binding to ER-α (see section 6.3) in a run with population 20 and 50 iterations. As shown, the eMEGA solution set contains more Pareto solutions and has a larger spread than the front without elitism.

Further analysis of the results provided in section 6.2 indicated that the niching mechanism on its own sometimes fails to meet its goals, i.e., preserve solution diversity at the genotype level, despite the diversity analysis and sampling of the non-dominated solutions it performs. The mechanism clearly works initially in identifying a diverse set of non-dominated solutions, but as more and more points on the Pareto-front are produced, the algorithm necessarily drops some of the non-dominated solutions because of the fixed population size allowed. This behaviour may sometimes lead to dropping solutions important enough to cause loss of large sections of the Pareto-approximation surface and thus reduce diversity. However, when combined with elitism in the form of a Pareto-archive, the niching mechanism proved to be effective in preventing genetic drift and domination conditions.

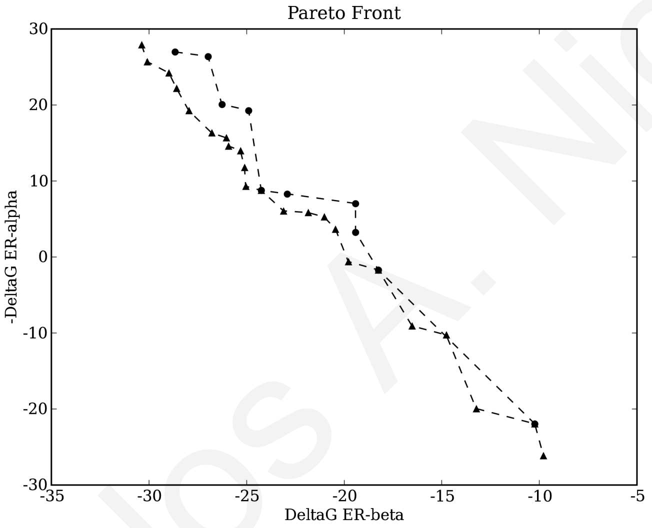


Fig. 7.2.1. A comparison between the final Pareto-front approximations obtained with and without elitism for a de novo design selectivity test. Note that the front with elitism (line with triangles) contains more Pareto solutions and has a larger spread than the front without elitism (line with circles). The x-axis corresponds to the predicted interaction score to ER-beta; the y-axis to the inverted predicted interaction score to ER-alpha.

7.2.4 General Remarks

The results obtained through the extensive set of experiments presented in chapter 6 stressed the positive effect of providing an appropriate population size to the MEGA algorithm, both in the case of larger and more complex search spaces and in simpler problems (see Fig. 6.2.8). As a general rule, runs with larger population sizes consistently outperformed runs with smaller, inadequate for the specific problem population sizes, all other parameters being equal. This observation held true as long as a certain, problem-specific population size was not provided, indicating that an increased population size can only benefit the optimization process up to a certain problem-specific number; providing larger populations above that number has no positive effect (see Fig. 6.2.8, top) and may complicate and presumably affect performance with respect to time.

An additional general observation on the results produced in the series of tests performed is that MEGA produces similar Pareto-approximations in different runs when the same parameters -except the initial population- are provided. In this context, similarity refers to the values of the performance measure obtained and in particular those of the hypervolume and genotype diversity measures which are of special interest to the present work. On the contrary, results produced using other methods and especially MOGA and simple-MEGA show a much greater variation. Overall, smaller populations tend to show greater variation than larger populations possibly explained by the inability of small populations to cover effectively a representative sample of the solution space.

Observations on the effect of the number of iterations pointed to the importance of this parameter for the quality of the solutions produced. However, it was evident in the experimental results that the number of iterations could not compensate for the effect of a small population size.

The set of subgraph genes used by the algorithm, an additional way of encoding and exploiting problem domain knowledge, is also very important for the obtained results. Experimentation with smaller and/or unweighted subgraph gene sets, all other parameters kept equal, showed that the resulting Pareto-approximations had a more limited span and were less advanced compared to the fronts produced by runs using the 2363-collection derived from compounds tested on the biological targets of interest. A similar conclusion was reached for the importance of the datasets used for the selection of the initial populations although at a lesser extend. These observations indicate that the sources of subgraph genes and the datasets used to select the initial population represent a simple, yet effective, possibility for the user to control the region of the chemical space for the search.

MEGA also proved able to cope with situations where one of the primary objectives is easier to achieve than the other. For example, in the selectivity tests performed in section 6.2.3, identifying solutions with reduced binding affinity to ER-alpha is substantially simpler than designing solutions with increased binding affinity to ER-beta. If left unattended, the uneven nature of the objectives can lead to the discovery of more solutions satisfying the easier objective, and a more intense exploration of the corresponding section of the search space, which in turn may result to genetic drift and potential domination conditions. Owing to its hard-filtering capabilities MEGA enabled the elimination of solutions overfit to the simpler objective while the usage of the niching mechanism ensured that no domination conditions from solutions found in any specific region of the search space could take place. As a result, the progress of the search process as indicated by the Pareto-sets produced in successive iterations of the specific MOOP study has been even, spanning an increasingly larger range of the search space.

7.3 Multi-objective De Novo Design of Selective Chemical Structures

Qualitatively, MEGA performance has been validated by human experts on a challenging type of the de novo design problem. Specifically, we have designed a test where the solutions are required to bind effectively to a “positive” pharmaceutical receptor and show no, or limited, affinity to a “negative” pharmaceutical receptor. This setting, known as compound selectivity in the drug discovery community, is of immense importance since a drug needs to bind selectively to the target it was designed for and avoid off-target interactions which cause the much dreaded –and outrageously costly in industrial settings– side-effects and toxicity. Exploiting the unique features of MEGA, especially graph-based chromosome representation, multi-objectivity, niching that maintains diversity in both decision and objective space, and local search exploitation, we have shown that our method can successfully design diverse compounds predicted to bind stronger to ER-β than the highly similar ER-α.

In order to obtain such range of solutions with traditional, single-objective methods, multiple runs with different settings favoring one or the other objective at a different ratio each time could be attempted. However, even an exhaustive set of experiments with different objective weights would fail to discover solutions in certain solution surfaces [NAP09]. Moreover, single-objective optimization runs require caution to avoid overtraining and generation of over fit individuals. MEGA, through the use of multiple objectives can restrict the search space and prevent the survival of overspecialized solutions.

Our results have been presented at [NPA08] and have been the subject of a publication [NAP09] to the Journal of Chemical Information Modeling of the American Chemical Society. Similar problem cases and experiments are currently in preparation in association with partners from the life-sciences domain at their request. This research direction provides for the close collaboration with pharmaceutical scientists to prepare and use the MEGA method within the scope of an ongoing drug discovery project. MEGA, equipped with expert knowledge, would design new chemical structures, which would then be synthesized and tested in the lab for biological activity. This scenario could prove the usefulness of the method and provide feedback for further improvements to be made. In general, the selectivity application is especially important to the life sciences informatics and chemoinformatics communities since it contributes to the recognition of the necessity and usefulness of multi-objective methods in the field [ES02]. Standard computational methods supporting drug discovery and specifically lead optimization, modelled after traditional experimental optimization procedures, traditionally ignore the multi-objective nature of the problem and focus on the optimization of one molecular property at a time [XH02]. While de novo design has long been recognized as being of high importance for drug discovery, it is only relatively recently that methods have been proposed in which the concept of multiple objectives is exploited [NBP07]. The lack of research in this field presents substantial opportunities for novel algorithms to be developed to researchers with the know-how in the areas of multi-objective optimization and graph/de novo design. The ability of MEGA to design chemical structures acceptable to human experts for a major, common problem such as selectivity to ER-beta over ER-alpha is one of the few, recent success cases in the field. Overall the specific test case, which constitutes one of the main motivating factors for this research, showed that the MEGA algorithmic framework can be of use to the drug discovery process. The effect of various evolutionary algorithm parameters has been investigated, and partially elucidated, and the contributions that a graph structure-based niching mechanism, a secondary archive population to preserve solution loss and, a self-adaptive local search mechanism have been demonstrated. We believe that the latter findings are particularly important.

7.4 Comparison of MEGA to Established MOEAs

The experimental results presented in section 6.4 indicate that solution sets generated by MEGA compare favourably with those obtained using MOGA and SPEA. It is worth noting that our experimental results show that in the case of simple multi-objective problems (e.g. Section 6.4: Test 2) the performance of the three algorithms is comparable whereas in more complex problems (e.g. Section 6.4: Test 1) differences become more apparent.

With respect to the hypervolume measure, indicative of the span of a Pareto-front and its proximity to the ideal point, MEGA consistently outperforms at a statistically significant level both SPEA and MOGA in the complex tests attempted. It is worth pointing out that for Test 1, optimizing two ligand-based similarity objectives, no run of SPEA and MOGA matched any of the runs of MEGA with respect to hypervolume, given that the initial parameters provided were the same. A similar observation holds for the hypervolume-based comparison between SPEA and

MOGA; SPEA runs perform better than those of MOGA throughout the experiments performed although statistical significance was only found for runs with smaller population sizes.

A similar, also statistically significant, trend has been observed for the measure of genotype diversity, i.e., diversity of solutions in parameter space. MEGA, with its unique mechanism of niching specifically designed to take into account graph structure and preserve solution diversity both in parameter and objective space, is more successful than either MOGA or SPEA in this measure for both sets of tests attempted. Even the worse performing run of MEGA produced Pareto-fronts comparing favourably to the genotype diversity of any set of solutions provided by the more traditional MOEA algorithms of SPEA and MOGA, which in this measure exhibit similar performance. This conclusion verifies that the niching mechanism proposed in MEGA fulfils its goal and manages to ensure that the set of nondominated solutions produced by the algorithm will contain diverse graph structures, well dispersed in parameter space.

Conclusions are not so clear for the measures focusing on objective space. Specifically, for the spacing measure, MOGA performs better than the other two that show similar performance. Moreover, MOGA results show the greatest variation in performance among different runs of the same experiment while MEGA and SPEA produce results with a much smaller range of spacing values although consistently worse than MOGA. This may be partly attributed to the smaller size of the Pareto-approximation sets generated by MOGA due to the lack of a Pareto archive. Results for the objective space diversity measure are inconclusive since they do not show a trend for or against a specific algorithm.

Overall, the results of MEGA demonstrate the ability of the algorithm to explore the chemical graph space given clear objectives to use for solution scoring. The comparisons with commonly used MOOP algorithms show that MEGA compares favourably with MOGA, the most commonly used technique in DND, and SPEA, probably the most popular MOEA of all, since it produces Pareto-approximation sets with better performance in the hypervolume and genotype diversity measures in our experimental tests. Differences in performance are less evident in easier problems but become clearer in problems with more complex objectives. The above conclusions are supported by observations on the results produced for both sets of problems used to compare the three approaches and for varying population sizes. The conclusions have been verified by the Mann-Whitney statistical significance test.

7.5 Effect of MEGA Parallelization

Our parallelization efforts with pMEGA focused on the use of coarse-grained parallelization through the use of subpopulations. pMEGA follows an island model EA and implements restricted mating in a geographic sense where solutions mate only with neighbors residing within some restricted topology [CLV07]. The experimental results produced led to the following conclusions and related future research directions. Firstly, as far as the quality of the solutions produced, MEGA and pMEGA behave comparably, with pMEGA having a slight edge in performance on average. This conclusion holds even though the current pMEGA implementation splits the population in a random fashion without using any knowledge related to the morphology of the Pareto-approximation set and the density of solutions at any region of the search space. A potentially better way to split the population may be the clustering-based method used by the STIR mechanism of MEGA. Secondly, pMEGA achieves a speedup of almost 1.6 on a common dual-core CPU and 2.7 on a quad-core CPU which is considerable, especially for large experimental applications [K10].

The measured results and speedup provided by pMEGA have been used below to calculate the percentage of parallelism of the algorithm and estimate the upper bound of speedup when using additional processing units via Amdahl's Law [AG67] defined as:

$$\frac{1}{(1-P)+\frac{P}{N}} \qquad \text{Eq. (12)}$$

where P is the proportion of the program that can be made parallel and N is the number of processing cores. The average speedup measured is S_{avg} = 2.7. Using:

$$P_{estimated} = \frac{\frac{1}{S}-1}{\frac{1}{N}-1} \qquad \text{Eq. (13)}$$

with S = 2.7 and N = 4, the $P_{estimated}$ = 0.84, meaning that 84% of the algorithm is executed in parallel. Table 7.5.1 below contains estimates for the speedup of the algorithm when more processing units are utilized by substituting $P_{estimated}$ in Amdahl's law using as reference the speed-up measured for the quad-core CPU system.

TABLE 7.5.1: ESTIMATED SPEEDUP AND EFFICIENCY USING ADDITIONAL CORES.

N	$S_{estimated}$	E
4	2.7	0.7
8	3.8	0.5
16	4.7	0.3
32	5.4	0.2
64	5.8	0.1
$P_{estimated}$	0.84	Estimated parallel percentage of the algorithm.
$S_{estimated}$	$\frac{1}{(1-P)+\frac{P}{N}}$	Estimated Speedup
S_{max}	$\frac{1}{(1-P)}$	6.3

P: proportion of program made parallel; N: number of cores, E: Efficiency; The quad-core CPU is used as reference.

According to these estimates the algorithm has an average maximum speed-up of around 6.3 as more and more cores are provided to the process since efficiency gradually drops to a very low level. From the above we can conclude that using more than eight cores to run the current pMEGA algorithm implementation is probably unnecessary due to the low efficiency and small speedup from that point and on. This is also supported by Amdahl's law, which states that if an algorithm can be parallelized at a degree of 90% then the maximum speedup it can achieve regardless of the number of CPUs is 10. In the case of pMEGA the estimated parallelized part of the algorithm is 84% and therefore the application of Amdahl's law in combination with the setting of the number of CPUs to infinite, results in a maximum estimated speedup of 6.3. This value indicates that there is plenty of room for additional, especially fine-grained, parallelization efforts. It can safely be expected that sizeable savings can be achieved since in de novo design, as well other optimization and optimal graph design problem instances, the cost of fitness evaluation dominates the time requirements of the process.

Regarding the quality of the solutions produced by pMEGA compared to MEGA, both the quantitative performance measures as well as the visual inspection of the proposed solutions show that the two methods produce comparable Pareto fronts. A further observation is that pMEGA subpopulations typically provide solutions that differ between them indicating that speciation occurs due to the independent evolution of each subpopulation. It is interesting that speciation occurs despite selecting subpopulations at random, without the use of any knowledge relating the members of each subpopulation in this first, simple implementation of parallel MEGA.

In a related set of experiments, the performance of pMEGA (i.e., the speedup gained) was shown to gradually worsen when using increased number of subpopulations. This can be attributed to the higher inter-process communication overhead and, naturally, the non-parallelizable part of the implementation, i.e., the main process, which collects and merges the results produced by each process independently evolving a subpopulation. The collection and merging requires additional computation to take place in the main process as additional subpopulations get added, resulting in a small, but noticeable amount of overhead to the pMEGA execution time.

Chapter 8

Conclusions

The research presented in this thesis introduces a new hybrid multi-objective evolutionary algorithm that emphasizes the use of knowledge, both problem-specific, available through previous attempts to tackle the problem under investigation and, knowledge produced during the search for the solution. The latter characteristic requires real-time monitoring of the search, evaluation of the progress achieved and self-adaptation of the algorithm execution to facilitate the discovery of solutions. The former functionality allows the inclusion of prior knowledge related to the problem and its exploitation in order to focus the search process and avoid pitfalls. MEGA uses graphs for solution representation and encodes available problem-specific knowledge through the use of weighted subgraph genes. The algorithm uses a novel, niching mechanism specifically designed to preserve both genotype and phenotype diversity in the population of solutions, and elitism, mainly in the form of a Pareto archive, to avoid loss of promising solutions [NKP09]. An additional novel mechanism has been improvised to self-adapt the optimization process through the use of a local search technique based on the progress of the search. This mechanism, referred to as STIR, monitors the progress of the Pareto approximation set through the calculation of the hypervolume measure at the end of each iteration, compares the progress achieved over successive iterations and, when deemed necessary, applies local search on subsets of the population to boost the optimization process. In the current implementation, STIR uses clustering of the solutions in parameter space to identify groups of similar individuals that it evolves independently through intensive application of local search in the form of dedicated MEGA sub-processes. As a general framework MEGA has the ability to accommodate numerous objectives as primary, to guide the optimization process, or secondary, as hard filters, to limit the search space and exclude sections known to contain solutions unfit due to inherent solution structure or other problems. Accordingly, the current MEGA implementation has been equipped with mechanisms for the inclusion of multiple objective functions and their appropriate subsequent use by the algorithm during execution time.

As a case study, we have applied our graph design method to the significant and challenging problem of de novo design and specifically in designing small molecular structures exhibiting selectivity to select pharmaceutical targets. Modern drug discovery process typically emphasizes potency and underestimates additional molecular properties in the early stages of lead identification and optimization. Indeed, one of the common causes for lead compounds to fail in the later stages of drug discovery is the lack of adequate consideration of multiple objectives (e.g. ADME and Toxicity) at the early stage of optimization of candidate compounds [BM04]. Current de novo design approaches also focus on optimizing a single property, typically similarity to a known ligand or docking affinity to a receptor. A number of objectives, measuring fitness as predicted binding affinity to a receptor, similarity to known ligands or chemical structure-based properties such as molecular weight or complexity, have been encoded and used during our optimization runs.

The experimental design aimed at obtaining a comprehensive profile of the MEGA implementation with respect to its sensitivity to population size, number of iterations, and the usage of the niching, elitism and STIR mechanisms, as well as comparing the algorithm to other, established algorithms in the MOEA domain, namely MOGA and SPEA. A series of experiments has been performed, thoroughly testing the proposed method. For each combination of input parameters multiple runs were executed each time changing only the initial population. Our conclusions have been drawn on these collections of runs to eliminate the chance of arriving to erroneous results due to results obtained from single runs affected by (un)fortunate events taking place during the stochastic process.

The results obtained indicate that the use of niching and elitism, as implemented and applied in MEGA, contribute to improved performance. This finding confirms previous findings of [Z99], [KJ02], [CS04]. It was also observed that population size played a crucial role and therefore larger populations can lead to Pareto-approximations sets of better quality compared to smaller populations in more complex problems. STIR, the self-adaptive mechanism was also proven to contribute positively to the quality of the Pareto approximation set produced. Specifically, STIR managed to detect lack of progress and apply local search to improve individual solutions, advance the trade-off surface and enrich the gene pool of the population with diverse graph structures. Consequently, this self-adaptive, memetic feature of the algorithm was crucial in generating better results or the same quality of results in fewer iterations.

Overall, the results of MEGA compare favourably with established MOOP methodologies. Three quantitative performance measures were used to evaluate the performance of MEGA and compare it to MOGA and SPEA. Hypervolume [ZT99], which measures the hyperarea defined by a Pareto approximation set and a nadir point, spacing [CS04] which measures the distribution of solutions in objective space and diversity which measures the average similarity of solutions in parameter or objective space. The analysis of the algorithmic results show that MEGA outperforms both MOGA an SPEA, consistently producing better Pareto approximation sets at a statistically significant level for the hypervolume and genotype diversity measures. This confirms previous reports comparing hybrid, memetic algorithms such as MEGA, with pure EA-based methods [M04]. The improved performance can be traced to several of the features of the method including the graph representation of chromosomes, the use of niching and elitism but also to the inclusion of knowledge-driven components enabling the exploitation of available information useful to the search and, the activation of local search to boost the process through a self-adaptive step. This was confirmed by comparisons between MEGA and simpler versions of the method without elitism or the memetic, self-adaptive component that showed the value of each of the above mechanisms.

Qualitative validation of the products of the proposed method has been performed through visual inspection by expert partners of the results on one of the test cases. This approach, conceptually aligned with the a΄ posteriori principle of Pareto-based MOOP methodologies, showed that MEGA is able to identify compromising solutions of satisfactory quality and structural diversity and meet the stated optimization objectives. The test case chosen was that of designing molecules exhibiting selective potency to one of two closely related pharmaceutical targets, ER-α and ER-β. The problem is of high importance as ERs are receptors related to breast cancer and no drug currently available has been designed with selectivity in mind. The experts identified several interesting designs predicted to have a significantly higher binding affinity to ER-β than the similar ER-α. The findings were further confirmed via computational docking-scoring experiments and the visualization of the results which provided further insight and justification to the predicted selectivity. These results were the subject of a journal publication [NAP09].

In a development phase performed later in the project we implemented pMEGA, a parallel version of MEGA aiming to reduce the time needed for the completion of the design process [KNP09]. The need for this version of the algorithm became apparent when using larger population sizes and/or resource-intensive objectives functions such as those based on docking, i.e., computational prediction of the binding affinity of a small molecule to a protein receptor. The first implementation of pMEGA focused on coarse-grained parallelization and some changes to the algorithm had to be performed. In pMEGA the working population is now divided into subpopulations evolved independently for a specific number of iterations on different processing units. The results produced by the evolution of each subpopulation are periodically merged, Pareto-ranked, subjected to evolutionary selection pressure and re-split into subpopulations thereby performing unrestricted migration of good quality individuals. The role of the master process is limited to coordinating iterative search process through the initiation of sub-processes assigned with the task of evolving subpopulations, the collection and merger of the results and the re-distribution of solutions to subpopulations. pMEGA fulfilled its initial objectives by producing results of comparable performance with MEGA utilizing multiple cores of a single CPU. The similarity of the results produced was also confirmed through statistical significance analysis. In conclusion, pMEGA showed an average speed-up of 1.6 on a dual-core and 2.7 on quad-core machine. The theoretical speed-up boundary was calculated at approximately 6.3 indicating that further parallelization efforts need to be performed for effective application on larger computational infrastructure such as the Grid or high performance computing systems. To this end fine-grained parallelization provides a potential solution that will be explored in the immediate future.

As a final conclusion, MEGA has been shown to be a powerful and flexible new algorithmic framework specifically designed to address the multi-objective graph design problem. We believe that the method as described in the previous chapters may be of great use to a number of real life problems that fall in the same category and intend to pursue such applications in the near future. Similarly, the method incorporates several novel features and ideas that may be directly applicable to other optimizations methods. Prime among them is the inclusion of knowledge to guide the search process, through the use of information-rich subgraph genes and rules, and, self-adaptive memetic mechanisms applying local search to advance subsets of solutions when premature convergence conditions are observed.

Chapter 9

Future Work

A number of different research directions have already been initiated to expand on the work presented in this dissertation. These initiatives can be grouped into three general categories; the first two involve research on algorithmic enhancements and improvements of the computational performance while the third focuses on problem-specific applications of the method. The potential directions of future work are outlined below:

9.1 Algorithmic Enhancements

Self-adaptation. Efforts will be made to further exploit knowledge acquired during evolution to guide the optimization process by enhancing self-adaptation. In order to achieve this, additional mechanisms to encode and store qualitative information related to the performance of solutions will be implemented. The information may be used in the adaptation of evolutionary operations and subgraph gene weights to influence subsequent generations. One potential direction could exploit the readily available cache of solutions used currently only for the improvement of execution time. It is possible that the method counts the number of new solutions generated in a generation by comparing to the known solutions in the cache and devising a scheme whereby the measure of progress -or lack of- is a function of the new solutions produced. An alternative measure of progress, also easy to implement, may use the number of new Pareto-solutions added to the archive in a generation. The latter may also be used in conjunction with the STIR mechanism currently in place so that certain information about the progress of individual solution groups is obtained and exploited. For example, more emphasis could be placed in exploring the graph space defined by a certain group if it has been steadily producing better solutions through the evolutionary process.

Objectives Evaluation and Selection. The availability of multiple sources of knowledge often leads to the encoding of several objective functions to guide the optimization process. In turn, this produces a complex search space that complicates the optimization process and imposes a heavy computational burden to any MOEA implementation. High-dimensional problems pose additional challenges compared to low-dimensional problems in terms of the identification of a good Pareto approximation set, the computational resources required and the choice of an appropriate solution subset from a set of alternative solutions by the end user [BSDZ07]. As part of the future work planned we plan to exploit current research findings on dimensionality reduction techniques to decrease the number of objectives [BSDZ07], [DS06] and develop methods to assess the usefulness of each objective function and, to exploit potential objective orthogonality and redundancy issues.

Subpopulation-based MEGA. The implementation of STIR and pMEGA provided an opportunity to experiment with multiple subpopulations evolving independently and exchanging information at regular intervals. The results produced show that pMEGA can provide us with equivalent solution sets in substantially less time while STIR, which combines subpopulation use with the normal evolutionary cycle, provides solutions with improved performance. While the emphasis in pMEGA has been on reducing time requirements and maintaining comparable performance with MEGA our planned implementation will aim at exploiting the presence of distinct subpopulations to facilitate the preservation of population diversity in parameter and objective space. Initial work will focus on the selection of subpopulations, as currently performed in STIR, and the exploration of different migration schemata. Population neighborhoods will be identified using knowledge discovery methods and used to ensure population diversity and promote the concurrent optimization search along multiple regions of the Pareto-front in a more deterministic manner.

Local Search Techniques. MEGA currently performs local search through the formation of subpopulations based on natural groups of solutions and the initiation of short, intensive MEGA runs on each of these subpopulations. Effectively, this process focuses the exploration on promising regions of the search space as mapped by clusters of parent solutions. Recent research in the memetic algorithm field has shown that the use of a variety of local search operators combined with adaptive selection may improve the performance of the algorithm [CHKB02]. To investigate this claim and improve MEGA performance we plan to encode additional local search techniques related to graph design and, for the DND problem, drug-likeness. The techniques implemented will include human expert–defined rules encoding problem-specific knowledge, and simpler search methods such as blind search, hill-climbing and simulated annealing. An approach similar to self-adaptation will be used to choose the local search technique to use at each step.

Defining the Evolutionary Potential of a Population. One of the main research questions that surfaced in the course of this research, especially during our efforts to implement self-adaptation, is related to our inability to explain why some initial populations lead to better final solution sets than others, or to solutions of the same quality in less time; why some subpopulations of the same species are more "successful" than others. These observations have led us to explore the likely presence of collective characteristics of a given population that increase its chances to achieve better adaptability rate than seemingly similar populations. It naturally follows that, if indeed such characteristics exist, it may be possible to design or favor populations possessing them so as to improve the quality of the search results and the optimization process in general. Furthermore, it may also be possible to predict the likely success/performance of a given population set and when likely performance jumps in the evolution process may be approaching, or, when our population improvement potential has stalled. Recent research reviewed in [SBB09], supports that critical points in a dynamic system indicate imminent major and abrupt shifts from one population balance to another with substantially different characteristics. Using the same terminology, is it possible to predict the likelihood of critical points from the characteristics of a population and, conversely, to design populations likely to continue evolving to future generations with better performance. An obvious characteristic that should be investigated is population diversity and therefore we will continue to give it particular attention to ensure it is promoted and maintained both in objective and parameter space.

Some initial investigations have been made to quantify the evolutionary potential of a population set. These have led to the vision of formulating a Population Intelligence Coefficient (PIC), a scalar value indicating the collective improvement potential of a group of solutions which could be used to self-adapt the optimization process through targeted modifications of parameters controlling mutation, crossover, selection pressure as well as the re-initialization of the population by activating STIR-like mechanisms. We are currently investigating the idea of the PIC index and its use in conjunction with the STIR mechanism. Although still incomplete and inconclusive, the available results point out that the concept of a population collective intelligence coefficient may be useful to assess the potential of an optimization run and thereby adjust the search process accordingly.

9.2 Performance Improvement

Parallelization. pMEGA has been successful in using coarse-grained parallelization to considerably reduce the computational requirements of MEGA while maintaining performance at a comparable level. However, the limit of the performance improvements possible by the current pMEGA implementation is limited (according to Amdahl’s law) to less than x7 (see section 6.5). Further improvement of the algorithm implementation using fine-grained parallelization will enable the effective deployment of the system on high performance computing systems and/or large scale distributed systems (e.g. the grid). Consequently, additional speed-up gains will be made possible which will allow handling a considerably larger number of objectives and effectively exploring more complex search spaces.

9.3 System Application

De Novo Design. The ultimate evaluation test of any DND application is the experimental validation of the molecules designed. To this end, we have been collaborating with expert partners on real-life problems requiring the de novo design of molecules for specific diseases. Such collaborations require considerable resources (knowledge, funding) and typically involve a long time commitment to produce results. It is expected that feedback provided by expert users will be crucial for the improvement of MEGA for the DND problem.

Encoding Additional DND Objectives. In order to improve the search for promising molecular designs additional objective functions will be implemented to enrich the pool of computational criteria available. Namely, additional drug-likeness and/or ADME-Tox criteria will be encoded to guide search and optimization away from problematic chemical structures. Special emphasis will be given to the latter criteria known to be one of the main obstacles in the quest for effective drugs. Ongoing research in this field aims at developing models estimating the toxicity profile of a molecule based on its chemical structure, for example, through the definition of the so-called toxicophores, i.e., chemical fragments frequently occurring in toxic molecules.

New OGD Domains: DND is a challenging test case for the OGD problem with real-life application potential. For the purposes of this dissertation it has served as both, the motivating force for the research performed and the proof of concept for the potential of the proposed MEGA method. However, further work needs to be pursued in order to prove the ability of MEGA to successfully tackle OGD problems from additional domains with different requirements and specific objectives. To this end, one of our immediate future goals is to extend the generic MEGA implementation to other problem domains such as the TSP and telecommunication network design. Work on these extensions is already ongoing. We expect that the successful conclusion of our efforts to apply MEGA to additional OGD problems will facilitate the adoption of our general knowledge-driven, self-adaptive approach, as well as its individual components, by the greater MOOP and graph design communities.

Bibliography

[ABC05] D. A. Ashlock, K. M. Bryden, and S. Corns, “Graph Based Evolutionary Algorithms Enhance the Identification of Steiner Systems,” In Proceedings of the 2005 Congress on Evolutionary Computation, vol. 2, pp. 1861-1866, 2005.

[AG67] G. Amdahl, "Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities," AFIPS Conference Proceedings (30): 483–485, 1967.

[AT02] E. Alba, and M. Tomassini, “Parallelism and Evolutionary Algorithms,” IEEE Transactions on Evolutionary Algorithms, Vol. 6, No. 5, October 2002, pp. 443-462.

[AW09] Amdahl's law article in Wikipedia. Online available at http://en.wikipedia.org/wiki/Amdahl's_law (Accessed December 1, 2009).

[BACW06] K. M. Bryden, D. A. Ashlock, S. Corns, and S. J. Willson, “Graph-Based Evolutionary Algorithms,” IEEE Transactions on Evolutionary Computation, vol. 10, no. 5, October 2006.

[BBS97] J. Branke, F. Bucher and H. Schmeck, “A genetic algorithm for drawing undirected graphs”, *The Third Nordic Workshop on Genetic Algorithms and their Applications,* 1997.

[BC01] R. Barone and M. A. Chanon, “New and Simple Approach to Chemical Complexity. Application to the Synthesis of Natural Products,” *J. Chem. Inf. Comput. Sci.*, 41(2), pp. 269-272, 2001.

[BLTE03] S. Bleuler, M. Laumanns, L. Thiele and E. Zitzler, “PISA - A Platform and Programming Language Independent Interface for Search Algorithms,” *In*

Proceedings of Evolutionary Multi-Criterion Optimization (EMO 2003), Eds. C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb and L. Thiele, Lecture Notes in Computer Science, Berlin, Springer, pp. 494-508, 2003.

[BM04] K-H. Baringhaus and H. Matter, "Efficient strategies for lead optimization by simultaneously addressing affinity, selectivity and pharmacokinetic parameters," in *Chemoinformatics in Drug Discovery*, T. Oprea (Ed), Wiley-VCH, Weinheim, Germany 2004, pp. 333-379.

[BMG96] R. S. Bohacek, C. Martin and W. C. Guida, "The Art and Practice of Structure-Based Drug Design: A Molecular Modelling Approach," *Med. Res. Rev.,* 16, 3-50, 1996.

[BMG04] N. Brown, B. McKay and J. Gasteiger, "The de novo design of median molecules within a property range of interest," *J. Comput-Aided Mol. Des.,* vol. 18 No. 12, pp. 761-771, 2004.

[BMG06] N. Brown, B. McKay and J. Gasteiger, "A novel workflow for the inverse QSPR problem using multi-objective optimization," *J Comput-Aided Mol Des,* vol. 20, No. 5, pp. 333-341, 2006.

[BMGG04] N. Brown, B. McKay, F. Gilardoni and J. Gasteiger, "A graph-based genetic algorithm and its application to the multi-objective evolution of median molecules," *J. Chem. Inf. Comput. Sci.*, 44(3), pp. 1079-1087, 2004.

[BMK03] D. Buche, S. Muller, and P. Koumoutsakos, "Self-Adaptation for Multi-objective Evolutionary Algorithms," *In proceedings of the Second International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003)*, 2003. 2632: p. 267–281.

[BSDZ07] D. Brockhoff, D. K. Saxena, K. Deb, and E. Zitzler, “On Handling a Large Number of Objectives A Posteriori and During Optimization,” In *Multi-Objective Problem Solving from Nature: From Concepts to Applications*; Knowles, J., Corne, D., Deb, K. Eds.; Springer: Berlin, 2007; pp 377-403.

[BST03] M. Basseur, F. Seynhaeve, and E.-G. Talbi, “Adaptive mechanisms for multi-objective evolutionary algorithms,” In *Proceedings of Congress on Engineering in System Application CESA’03*, Lille, France, July 2003 pp. 72–86.

[C99] C. A. C. Coello, “A comprehensive survey of evolutionary-based multi-objective optimization,” *Knowledge and Information Systems,* 1(3), pp. 269–308, 1999.

[CF95] D. E. Clark, D. Frenkel, S. A. Levy, J. Li, C. W., Murray, B. Robson, B. Waszkowycz and D.R. Westhead, “PRO LIGAND: an approach to *de novo* molecular design. 1. Application to the design of organic molecules,” *J. Comput.-Aided Mol. Design*, vol. 9, pp. 13-32, 1995.

[CHKB02] R. Carr, W. Hart, N. Krasnogor, E. Burke, J. Hirst and J. Smith, “Alignment of protein structures with a memetic evolutionary algorithm,” in *GECCO-2002: Proceedings of the Genetic and Evolutionary Computation Conference*, W. Langdon, E. Cantu-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. Potter, A. Schultz, J. Miller, E. Burke and N. Jonoska (Eds), 2002.

[CLV07] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Boston, MA: Springer Science+Business Media, LLC, 2007 Second Edition.

[CO08] Mobius Candidate Design Environment, Coalesix Inc, http://www.coalesix.com. (Accessed February 14, 2008).

[CS04] Y. Colette Y., P. Siarry (Eds): *Multi-objective Optimization: Principles and Case Studies,* Springer-Verlag, Berlin, Germany, 2004.

[CW96] D. E. Clark and D. R. Westhead, "Evolutionary algorithms in computer-aided molecular design," *J. Comput.-Aided Mol. Design*, vol. 10, pp. 337-358, 1996.

[D76] R. Dawkins, *The Selfish Gene,* 2nd ed., Oxford University Press, 1989.

[DAPM00] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. "A fast elitist nondominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Parallel Problem Solving from Nature (PPSN VI)*, M. Schoenauer *et al*., (Eds), pp. 849–858, Berlin, 2000. Springer.

[DC08] F. Dey and A. Caflisch, "Fragment-based de Novo Ligand Design by Multi-objective Evolutionary Optimization," *J. Chem. Inf. Model.,* vol. 48, pp. 679-690, 2008.

[DJ75] A. K. De Jong, *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, 1975.

[DK01] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.

[DML05] D. Douguet, H. Munier-Lehmann, G. Labesse and S. Pochet, "LEA3D: A Computer-Aided Ligand Design for Structure-Based Drug Design," *J. Med. Chem.,* vol. *48,* pp. 2457-2468, *2*005.

[DS06] K. Deb, and D. Saxena, "Searching For Pareto-Optimal Solutions Through Dimensionality Reduction for Certain Large-Dimensional Multi-Objective Optimization Problems," In *Proceedings of the World Congress on Computational Intelligence*, IEEE Press: Vancouver, Canada, 2006; pp 3352-3360.

[DS07] R. Dreżewski, and L. Siwik, “Techniques for Maintaining Population Diversity in Classical and Agent-Based Multi-objective Evolutionary Algorithms,” *Lecture Notes in Computer Science.* Springer Berlin / Heidelberg. pp 904-911, vol. 4488/2007. Y. Shi *et al*. (Eds.): ICCS 2007, Part II, LNCS 4488.

[DTG00] D. Douguet, E. Thoreau and G. Grassy, “A Genetic Algorithm for the Automated Generation of Small Organic Molecules: Drug Design using an Evolutionary Algorithm,” *J. Comput.-Aided Mol. Des.*, vol. 14, pp. 449-466, 2000.

[EM01] T. Eloranta and E. Makinen, “TimGA: A Genetic Algorithm for Drawing Undirected Graphs,” *Divulgaciones Matematicas* Vol. 9 No. 2, pp. 155- 171, 2001.

[EMS01] M. Emmerich, M. Grotzner, and M. Schutz, “Design of Graph-Based Evolutionary Algorithms: A Case Study for Chemical Process Networks,” *Evolutionary Computation*, 9(3), pp. 329-354, 2001.

[ES02] S. Ekins, B. Boulanger, P. W. Swaan and M. A. Z. Hupcey, “Towards a New Age of Virtual ADME/TOX and Multidimensional Drug Discovery,” *J. Comp. Aided Mol. Design*, 16, pp. 381-401, 2002.

[EWBB08] M. I. Ecemis, J. Wikel, C. Bingham and E. Bonabeau, “A Drug Candidate Design Environment Using Evolutionary Computation,” *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 5, pp. 591-603, 2008.

[F06] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence,* IEEE Press, Piscataway, New Jersey, USA, Third Edition, 2006.

[FF98] C. M. Fonseca and P. J. Fleming, “Multi-objective optimization and multiple constraint handling with evolutionary algorithms. I: A unified formulation,” *IEEE Trans Syst Man Cybernet,* 28(1), pp. 26-37, 1998.

[FS05] U. Fechner and G. Schneider, “Flux (1): A Virtual Synthesis Scheme for Fragment-Based De Novo Design,” *J. Chem. Inf. Model.* vol. 46, pp. 699-707, 2005.

[FS07] U. Fechner and G. Schneider, “Flux (2): Comparison of Molecular Mutation and Crossover Operators for Ligand-Based de Novo Design,” *J. Chem. Inf. Model.,* vol. 47, pp. 656-667, 2007.

[FSW05] R. Farmani, D. A. Savic, and G. A. Walters, “Evolutionary Multi-objective Optimization in Water Distribution Network Design,” *Engineering Optimization*, vol. 37, no. 2, pp. 167-183, March 2005.

[G88] R. Gould, *Graph Theory,* Menlo Park, California, USA: The Benjamin/ Cummings Publishing Company, Inc., 1988.

[G89] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[G99] A. Globus, J. Lawton and W. T. Wipke, “Automatic Molecular design using evolutionary algorithms,” *Nanotechnology,* 10, pp. 290–299, 1999.

[GP95] R. C. Glen and A. W. R. Payne, “A genetic algorithm for the automated generation of molecules within constraints,” *J. Comput. Aided. Mol. Des.* 9, pp. 181–202 1995.

[GR87] D. E. Goldberg and J. Richardson, “Genetic algorithm with sharing for multimodal function optimization,” In *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, J. J. Grefenstette (Ed), pp. 41–49, Hillsdale, New Jersey, 1987.

[GSM04] S. M. Gustafson, *An Analysis of Diversity in Genetic Programming*, Doctor of Philosophy Thesis, University of Nottingham, February 2004.

[HHMS00] C. Hillermeier, S. Huester, W. Maerker, and T. Sturm, “Optimisation of power plant design: Stochastic and adaptive solution concepts,” In *Evolutionary Design and Manufacture (Selected Papers from ACDM 2000)*, I. Parmee (Ed), pp. 1–18, Springer, Berlin, Germany.

[HJ75] J. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Systems,* The University Press of Michigan Press, Ann Arbor, USA, 1975.

[HNG94] J. Horn, N. Nafpliotis, and D. E. Goldberg. “A niched pareto genetic algorithm for multi-objective optimization,” in *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Computation*, vol. 1, pp. 82–87, Piscataway, NJ, 1994. IEEE Press.

[J04] W. L. Jorgensen, “The many roles of computation in drug discovery,” *Science* 303 (5665), pp. 1813-1818, 2004.

[K02] N. Krasnogor, “Memetic Algorithms: A Tutorial,” *7th International Conference on Parallel Problem Solving from Nature (PPSN VII)*, Granada, Spain, September 2002.

[K03] M. Koeppen, “On the Benchmarking of Multi-objective Optimization Algorithm,” In *Proceedings of the 7th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES 2003)*, V. Palade, R. J. Howlett, and L. C. Jain (Eds), Part I, pp 379–385, Oxford, UK, September 2003. Springer. Lecture Notes on Computer Science Vol. 2773.

[K10] C. Kannas, *A Parallel Implementation of a Multi-objective Evolutionary Algorithm*. M. Sc. Thesis, University of Cyprus, February 2010.

[KC04] J. Knowles and D. Corne, “Memetic algorithms for multi-objective optimization: issues, methods and prospects,” in *Recent Advances in Memetic*

Algorithms, N. Krasnogor, J.E. Smith and W. E. Hart (Eds). pp. 313-352, 2004, Springer.

[KCL08] V. Kelner, F. Capitanescu, O. Léonard, and L. Wehenkel, “A hybrid optimization technique coupling an evolutionary and a local search algorithm,” *Journal of Computational and Applied Mathematics*, 215, pp. 448 – 456, 2008.

[KGV83] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, “Optimization by Simulated Annealing,” *Science*, 220(4598):671–680, May 13, 1983. Online available at: http://fezzik.ucd.ie/msc/cscs/ga/kirkpatrick83optimization.pdf (Accessed November 14, 2009)

[KH02] S. Kamphausen, N. Höltge, F. Wirsching, C. Morys-Wortmann, D. Riester, R. Goetz, M. Thürk and A. Schwienhorst, “Genetic algorithm for the design of molecules with desired properties,” *J. Comput.-Aided Mol. Design*, vol. 16, No. 8-9, pp. 551-567(17), August 2002.

[KJ02] J. Knowles, *Local Search and Hybrid Evolutionary Algorithms for Pareto Optimization*. The University of Redding, 2002

[KN04] N. Krasnogor, S. Gustafson, “A Study on the use of Self-Generation in Memetic Algorithms,” *Natural Computing*, Vol:3:1, pp. 53-76, 2004.

[KNP09] C. Kannas, C. A. Nicolaou and C. S. Pattichis, “A Parallel Implementation of a Multi-objective Evolutionary Algorithm,” In *Proceedings of the 9th International Conference on Information Technology and Applications in Biomedicine (ITAB 2009),* Larnaca, Cyprus, November 5 - 7, 2009.

[KSF96] S. K. Kearsley, S. Sallamack, E. M. Fluder, J. D. Andose, R. T. Mosley, and R. P. Sheridan, “Chemical Similarity Using Physiochemical Property Descriptors,” *J. Chem. Inf. Comput. Sci.*, 36, 118–127, 1996.

[KT98] D. Kobler and A. G. B. Tettamanzi, “Recombination Operators for Evolutionary Graph Drawing,” In *Proc. of Parallel Problem Solving from Nature PPSN-V,* volume 1498 of Lect. Notes in Comp. Sc, Springer-Verlag, pp. 988--997, 1998.

[LBKI05] E. W. Lameijer, T. Baeck, J. N. Kok and A. P. Ijzerman, “Evolutionary algorithms in drug design,” *Natural Computing,* vol. 4(3), pp. 177-243, 2005.

[LBWH98] X. O. Lewell, D. B. Budd, S. P. Watson and M. M. Hann, “RECAP–Retrosynthetic Combinatorial Analysis Procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry,” *J. Chem. Inf. Comput. Sci.* 38, 511–522, 1998.

[LC97] C. A. Lipinski, F. Lombardo, B. W. Dominy and P. J. Feeney, “Experimental and Computational Approaches to Estimate Solubility and Permeability in Drug Discovery and Development Settings,” *Adv. Drug Discovery Rev.*, 23, pp. 3-25, 1997.

[LG03] A. R. Leach, V. J. Gillet, *An Introduction to Chemoinformatics*, Dordrecht, The Netherlands: Springer, 2003.

[LKBI06] E.-W. Lameijer, J. N. Kok, T. Baeck and A. P. Ijzerman, “The Molecule Evoluator. An Interactive Evolutionary Algorithm for the Design of Druglike Molecules,” *J. Chem. Inf. Model.*, vol. 46, pp. 545-552, 2006.

[M89] P. Moscato, "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms," *Caltech Concurrent Computation Program Report 826*, 1989.

[M04] P. Merz, “Advanced Fitness Landscape Analysis and the Performance of Memetic Algorithms,” *Evolutionary Computation*, 12(3), pp. 303-325, 2004.

[MG04] G. R. Marshall, "Introduction to Chemoinformatics in Drug Discovery – A Personal View," in *Chemoinformatics in Drug Discovery,* T. Oprea (Ed), Wiley-VCH, Weinheim, Germany, 2004, pp. 333-379.

[MHH07] S. Mabu, K. Hirasawa and J. Hu, "A Graph-Based Evolutionary Algorithm: Genetic Network Programming (GNP) and Its Extension Using Reinforcement Learning," *Evolutionary Computation*, 15(3): 369-398, 2007.

[MI09] MolInspiration cheminformatics software tools, http://molinspiration.com (Accessed December 16 2009).

[MJV00] J. F. Miller, D. Job, and V. K. Vassilev "Principles in the Evolutionary Design of Digital Circuits -- Part I," *Journal of Genetic Programming and Evolvable Machines,* Vol. 1, No. 1, pp. 8-35, 2000.

[MOD09] Chil² Modelling platform, MODEST, http://www.chil2.de (Accessed April 3, 2009).

[MZ96] Z. Michalewicz, *Genetic Algorithms* + *Data Structures* = *Evolutions Programs,* Springer-Verlag, New York, 1992.

[N98] R. B. Nachbar, ''Molecular evolution: A hierarchical representation for chemical topology and its automated manipulation,'' In *Proc. Third Ann. Gen. Prog. Conf.*, Madison, WI, pp. 246-253, 1998.

[N00] R. B. Nachbar, "Molecular evolution: automated manipulation of hierarchical chemical topology and its application to average molecular structures," *Genet Programming Evolvable Machines,* **1**, pp. 57–94 2000.

[NAP09] C. A. Nicolaou, J. Apostolakis, and C. S. Pattichis, “De Novo Drug Design Using Multi-objective Evolutionary Graphs,” *J. Chem. Inf. Model.*, vol. 49(2), pp. 295-307, 2009.

[NBP07] C. A. Nicolaou, N. Brown and C. S. Pattichis, “Molecular optimization using computational multi-objective methods,” *Curr. Opin. Drug Discovery Dev.,* 10, 316–324, 2007.

[NKP09] C. A. Nicolaou, C. Kannas and C. S. Pattichis, “Optimal graph Design Using A Knowledge-driven Multi-objective Evolutionary Graph Algorithm,” In *Proceedings of the 9th International Conference on Information Technology and Applications in Biomedicine (ITAB 2009)*, Larnaca Cyprus, November 5 - 7, 2009.

[NOES08] NSisToolkit chemoinformatics toolkit software, Noesis Chemoinformatics, Ltd. Nicosia, Cyprus. http://www.noesisinformatics.com (Accessed February 5, 2008).

[NP06] C. A. Nicolaou and C. S. Pattichis, “Molecular Substructure Mining Approaches for Computer-Aided Drug Discovery: A Review,” In In *Proceedings of the 6th International Conference on Information Technology and Applications in Biomedicine (ITAB 2006),* Ioannina, Greece, October 26-28, 2006.

[NPA08] C. A. Nicolaou, C. S. Pattichis and J. Apostolakis, “De Novo Drug Design Using Multi-Objective Evolutionary Graphs,” In *8th ICCS Conference,* Noordwijkerhout, The Netherlands, June 1-5, 2008.

[NPK09] C. A. Nicolaou, C. S. Pattichis, and C. Kannas, “Focusing Multi-Objective Optimization Search: Knowledge-driven De Novo Drug Design,” *Lorentz Center Workshop on Optimizing Drug Design*, Leiden, Netherlands, July 19-23, 2009. Online available at http://www.lorentzcenter.nl/lc/web/2009/359/program.php3?wsid=359 (Accessed November 15, 2009).

[P06] V. Pareto, *Manuale di economia politica*. Milan: Societa Editrice, 1906.

[PISA09] PISA Platform and Programming Language Independent Interface for Search Algorithms, Systems Optimization Group – ETH Zurich, http://www.tik.ee.ethz.ch/pisa (Accessed January 13, 2009).

[PLP08] R. Poli, W. B. Langdon, and N. F. McPhee. *A field guide to genetic programming.* Lulu.com, 2008. Online available at: http://www.gp-field-guide.org.uk (Accessed December 12, 2008).

[PS03] C. Peterson and B. Soderberg, "Artificial Neural Networks," in *Local Search in Combinatorial Optimization,* E. Aarts, J. K. Lenstra (Eds), Princeton University Press, Princeton, New Jersey, 2003.

[PSS07] L. Paquete, T. Schiavinotto, and T. Stutzle, "On Local Optima in Multi-objective Combinatorial Optimization Problems", *Annals of Operations Research*, 156(1):83-97, 2007.

[PYLAB09] Matplotlib 2D Plotting Library, http://matplotlib.sourceforge.net/index.html (Accessed August 10, 2009)

[RGW02] J. W. Raymond, E. J. Gardiner, and P. Willet, "RASCAL: Calculation of Graph Similarity using Maximum Common Edge Subgraphs," *The Computer Journal,* vol. 45, no. 6, pp. 631-644, 2002.

[RGW02b] J. W. Raymond, E. J. Gardiner, and P. Willet, "Heuristics for Similarity Searching of Chemical Graphs Using a Maximum Common Edge Subgraph Algorithm," *J. Chem. Inf. Comput. Sci.*, vol. 42, pp. 305-316, 2002.

[RN95] S. Russel, P. Norvig, *Artificial Intelligence, A Modern Approach*, Prentice Hall, New Jersey, USA, 1995.

[RO98] A. Rosete and A. Ochoa, “Genetic graph drawing,” In *Proceedings of the 13th International Conference of Applications of Artificial Intelligence in Engineering.* Galway, pages 37-41, 1998.

[RWPHM03] M. D. Ritchie, B. C. White, J. S. Parker, L. W. Hahn, and J. H. Moore. “Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases,” *BMC Bioinformatics,* 4 (1), 28, 2003.

[S07] J. E. Smith, “Coevolving Memetic Algorithms: A Review and Progress Report,” *Systems, Man, and Cybernetics, Part B, IEEE Transactions on,* V. 37(1), pp. 6-17 2007.

[SBB09] M. Scheffer, J. Bascompte, W. A. Brock, V. Brovkin, S. R. Carpenter, V. Dakos, H. Held, E. H. van Nes, M. Rietkerk and G. Sugihara. “Early-warning signals for critical transitions,” *Nature*, 461:53-59, 2009.

[SCIPY09] SciPy Scientific Software Tools for Python, http://www.scipy.org/ (Accessed February 24, 2009)

[SD94] N. Srinivas and K. Deb. “Multi-objective optimization using nondominated sorting in genetic algorithms,” *Evolutionary Computation*, 2(3), pp. 221–248, 1994.

[SF05] G. Schneider and U. Fechner, “Computer-based de novo design of druglike molecules,” *Nat. Rev. Drug Discov.* 4(8), pp. 649-663, 2005.

[SHRTPS08] G. Schneider, M. Hartenfeller, M. Reutlinger, Y. Tanrikulu, E. Proschak, and P. Schneider, “Voyages to the (un)known: adaptive design of bioactive compounds,” *Trends in Biotechnology,* vol. 27, no. 1, pp. 18-26, 2008.

[SLS00] G. Schneider, M.-L. Lee, M. Stahl and P. Schneider, “De novo design of molecular architectures by evolutionary assembly of drug-derived building blocks,” *J. Comput. Aided Mol. Des.*, vol. 14, pp. 487–494, 2000.

[SR01] M. Stahl, and M. Rarey, “Detailed Analysis of Scoring Functions for Virtual Screening,” *J. Med. Chem.*, 44, pp. 1035–1042, 2001.

[TA07] S. Tietze and J. Apostolakis, “GlamDock: development and validation of a new docking tool on several thousand protein-ligand complexes,” *J. Chem. Inf. Model.* 47(4), pp. 1657-1672, 2007,

[TB03] A. Toffolo, and E. Benini, “Genetic Diversity as an Objective in Multi-Objective Evolutionary Algorithms,” *Evolutionary Computation*, Vol. 11, No. 2: pp 151–167, Summer 2003.

[TKL05] K. C. Tan, E. F. Khor, and T. H. Lee, *Multi-objective Evolutionary Algorithms and Applications*. Springer-Verlag, United Kingdom, 2005.

[TTW97] D. B. Turner, S. M. Tyrrell and P. Willet, “Rapid Quantification of Molecular Diversity for Selective Database Acquisition,” *J. Chem. Inf. Comput. Sci.*, 37, pp. 18-22, 1997.

[UBSE98] J. Utech, J. Branke, H. Schmeck, P. Eades, “An Evolutionary Algorithm for Drawing Directed Graphs,” In *Proceedings of the 1998 International Conference on Imaging Science, Systems, and Technology (CISST'98)*, pp. 154—160, CSREA Press, 1998.

[V07] D. Vrajitoru, “Hybrid Multi-objective Optimization Genetic Algorithms for Graph Drawing,” in *GECCO 2007*, London, UK.

[V09] D. Vrajitoru, “Multi-objective Genetic Algorithm for a Graph Drawing Problem,” In *Proceedings of the Midwest Artificial Intelligence and Cognitive Science Conference,* April 18-19, Fort Wayne, IN, 28-43, 2009.

[VCC94] V. Venkatasubramanian, K. Chan and J. M. Caruthers, “Computer-aided molecular design using genetic algorithms,” *Computers Chem. Eng.* 18, pp. 833–844 1994.

[VL00] D. A. van Veldhuizen and G. B. Lamont, “Multi-objective evolutionary algorithms: Analyzing the state-of-the-art,” *Evol. Comput.,* 8(2), pp. 125-147, 2000.

[W89] D. Weininger, “SMILES: A Chemical Language and Information System: 1. Introduction to Methodology and Encoding Rules,” *J. Chem. Inf. Comput. Sci.*, Vol. 29, No. 2, pp. 97-101, 1989.

[W95] D. Weininger, “Method and Apparatus for Designing Molecules with Desired Properties by Evolving Successive Populations,” U.S. Patent No. 5,434,796, 1995.

[W08] T. Weise, *Global Optimization Algorithms – Theory and Application.* Online available at: http://www.it-weise.de (Accessed November 16, 2009)

[WB00] D. J. Wild, and C. J. Blankley, “Comparison of 2D fingerprint types and hierarchy level selection methods for structural grouping using wards clustering,” *J. Chem. Inf. Comput. Sci*. 40, pp. 155–162, 2000.

[WBD98] P. Willet, J. M. Barnard, and G. M. Downs, “Chemical Similarity Searching,” *J. Chem. Inf. Comput. Sci.*, 39, pp. 983–996, 1998.

[WC95] D. R. Westhead, D. E. Clark, D. Frenkel, J. Li, C. W. Murray, B. Robson, and B. Waszkowycz, “PRO_LIGAND: An approach to de novo molecular design. 3. A genetic algorithm for structure refinement," *J. Comput.-Aided Mol. Design*, vol. 9, pp. 139-148, 1995.

[WEA06] D. L. Wheeler, *et al.* Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res.*, 34, D173–D180, 2006.

[WM97] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation,* 1:pp. 67-82, 1997.

[XH02] J. Xu and A. Hagler, "Chemoinformatics and dug discovery," *Molecules* 7(8), pp. 566-600, 2002.

[Z99] E. Zitzler. *Evolutionary Algorithms for Multi-objective Optimization: Methods and Applications.* PhD Thesis, ETH Swiss Federal Institute of Technology Zurich, November 1999.

[ZBT07] E. Zitzler, D. Brockhoff, and L. Thiele, "The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicators Via Weighted Integration," In *Proc. EMO 2007*, S. Obayashi *et al*. (Eds.), LNCS 4403, pp. 862–876, Springer-Verlag Berlin Heidelberg 2007.

[ZLB04] E. Zitzler, M. Laumanns, and S. Bleuler. "A Tutorial on Evolutionary Multi-objective Optimization," in *Metaheuristics for Multi-objective Optimisation, Lecture Notes in Economics and Mathematical Systems,* X. Gandibleux *et al*., (Eds), vol. 535 Springer, 2004.

[ZT99] E. Zitzler and L. Thiele, "Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Trans. Evolutionary Computation,* 3(4), pp. 257-271, 1999.

[ZTLF02] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca and V. G. da Fonseca, “Performance Assessment of Multi-objective Optimizers: An Analysis and Review,“ *TIK-Report No. 139*, Institut fur Technische Informatik und Kommunikationsnetze, ETH Zurich, 2002.

[ZWA06] L. Zwanepol Klinkmeijer, E.D. de Jong, and M. A. Wiering, “A Serial Population Genetic Algorithm for Dynamic Optimization Problems,” In *Proceedings of the 15th Belgian-Dutch Conference on Machine Learning*, Y. Saeys, E. Tsiporkova, B..De Baets & Y..Van de Peer (Eds.), pp. 41-48, 2006.

Appendix 1: List of Publications

a. Publications resulting from and related to this research

Journal Papers

C. A. Nicolaou, N. Brown and C. S. Pattichis, “**Molecular Optimization using Computational Multi-objective Methods**, ”*Current Opinion in Drug Discovery & Development,* Vol. 10, No 3, pp. 316-324, 2007

E. Loizidou, C. A. Nicolaou, A. Nicolaides and L. G. Kostrikis, “**HIV-1 Integrase: From Biology to Chemotherapeutics,”** *Current HIV Research*, Vol 5, No 4, 2007, pp. 365-388.

C. A. Nicolaou, J. Apostolakis and C. S. Pattichis, “**De Novo Drug Design Using Multi-Objective Evolutionary Graphs,**” *J. Chem. Inf. Model.*, vol. 49(2), pp. 295-307, 2009.

C. A. Nicolaou and C. S. Pattichis, “**An Algorithmic Framework for Knowledge-driven Multi-objective Optimization of Graphs,**” (in preparation, to be submitted to *IEEE Transactions on Evolutionary Computation*).

C. A. Nicolaou, C. S. Pattichis, U. Fechner, “**A Review of Evolutionary Algorithms in De Novo Drug Design**” (in preparation, to be submitted at *IEEE Transactions on Systems, Man and Cybernetics--Part C: Applications and Reviews or Natural Computing*).

Conference Papers

C. A. Nicolaou and C. S. Pattichis, “**Molecular Substructure Mining Approaches for Computer-Aided Drug Discovery: A Review**,” ITAB 2006, Ioannina, Greece.

K. Neophytou, C. A. Nicolaou, C. S. Pattichis and C. N. Schizas, “**Deriving Quantitative Structure-Activity Relationship Models Using Genetic Programming for Drug Discovery**,” ITAB 2007, Tokyo, Japan.

C. A. Nicolaou, C. Kannas and C. S. Pattichis, “**Optimal graph Design Using A Knowledge-driven Multi-objective Evolutionary Graph Algorithm,**” ITAB 2009, Cyprus.

C. Kannas, C. A. Nicolaou and C. S. Pattichis, “**A Parallel Implementation of a Multi-objective Evolutionary Algorithm,**” ITAB 2009, Larnaca, Cyprus.

Book Chapters

C. A. Nicolaou and C. S. Pattichis, “**Molecular Substructure Mining Approaches for Computer-Aided Drug Discovery: A Review**,” In Information Technology in Biomedicine, Pattichis, C.S., Fotiadis, D.I. (eds.), IEEE-Wiley (invited, submitted).

C. A. Nicolaou and C. Kannas, “**Molecular library design using multi-objective optimization methods**,” In Combinatorial Chemical Library Design, Zhou, J. (ed.), Humana Press Inc., Totowa, NJ (invited, submitted).

Published Abstracts

C. A. Nicolaou and C. S. Pattichis, “**Molecular Optimization Using Multi-Objective Computational Methods: A Survey**,” 16th European Symposium on Quant. Structure Activity Relationships & Molecular Modelling, Sept. 10-17, 2006, Italy.

C. A. Nicolaou and C. S. Pattichis, “**Multi-objective de novo drug design using evolutionary graphs**,” from 3rd German Conference on Chemoinformatics, Goslar, Germany. 11-13 November 2007, *Chemistry Central Journal,* 2008, 2(Suppl 1):P7

C. A. Nicolaou and C. S. Pattichis, “**Utilizing Grid Technologies for Efficient Multi-Objective De Novo Design,**” 17th European Symposium on Quant. Structure Activity Relationships & Molecular Modelling, Sept. 21-26, 2008, Sweden.

C. A. Nicolaou, C. Kannas and C. S. Pattichis, “**Knowledge-Driven Multi-Objective De Novo Drug Design**,” 4th German Conference on Chemoinformatics, Goslar, Germany, November 9-11, 2008, *Chemistry Central Journal,* 2009, 3(Suppl 1):P22.

b. Presentations resulting from this research

C. A. Nicolaou and C. S. Pattichis, “**Molecular Optimization Using Multi-Objective Computational Methods: A Survey,**” 16th European Symposium on Quant. Structure Activity Relationships & Molecular Modelling, September 10-17, 2006, Italy.

C. A. Nicolaou and C. S. Pattichis, "**Multi-Objective De Novo Drug Design Using A Novel Evolutionary Algorithm**," 1st Hellenic Bioinformatics Meeting in Athens, Greece, October 4-5, 2007.

C. A. Nicolaou and C. S. Pattichis, “**Multi-Objective De Novo Drug Design Using Evolutionary Graphs**,” 3rd German Conference on Chemoinformatics, Goslar, Germany, November 11-13, 2007.

C. A. Nicolaou, C. S. Pattichis, “**De Novo Drug Design Using Multi-Objective Evolutionary Graphs**,” 13th Hellenic Symposium on Medicinal Chemistry (HSMC-13), Athens, Greece, March 14-15 , 2008.

C. A. Nicolaou and C. S. Pattichis, “**Utilizing Grid Technologies for Efficient Multi-Objective De Novo Design,**” 17th European Symposium on Quant. Structure Activity Relationships & Molecular Modelling, Sept. 21-26, 2008, Sweden.

C. A. Nicolaou, C.S. Pattichis, J. Apostolakis “**De novo drug design using multi-objective evolutionary graphs**,” The 8th International Conference on Chemical Structures, Noordwijkerhout, The Netherlands, 1-5 June, 2008.

C. A. Nicolaou, C. Kannas and C. S. Pattichis, “**Knowledge-Driven Multi-Objective De Novo Drug Design**,” 4th German Conference on Chemoinformatics, Goslar, Germany, November 9-11, 2008.

C. A. Nicolaou, C. S. Pattichis, C. Kannas, “**Focusing Multi-Objective Optimization Search: Knowledge-Driven De Novo Drug Design,**” Optimizing Drug Design Workshop, Leiden, The Netherlands, July 20-23, 2009 (invited, keynote speaker).

C. Nicolaou, C. Kannas, C. Pattichis, “**Optimal graph Design Using A Knowledge-driven Multi-objective Evolutionary Graph Algorithm,”** In Proceedings of the Ninth International Conference on Information Technology and Applications in Biomedicine, Larnaca, Cyprus, November 5 - 7, 2009, 6 pages.

C. Kannas, C. Nicolaou, C. Pattichis, “**A Parallel Implementation of a Multi-objective Evolutionary Algorithm**,” In Proceedings of the Ninth International Conference on Information Technology and Applications in Biomedicine, Larnaca, Cyprus, November 5 - 7, 2009, 6 pages.