Impact of Population Size and Selection within a Customized NSGA-II for Biochemical Optimization Assessed on the Basis of the Average Cuboid Volume Indicator

Susanne Rosenthal, Markus Borschbach
University of Applied Sciences, FHDW
Faculty of Computer Science, Chair of Optimized Systems,
Hauptstr. 2, D-51465 Bergisch Gladbach, Germany
Email: {susanne.rosenthal, markus.borschbach}@fhdw.de

***Abstract*—The key part in the area of peptide design is the prediction of the peptides' molecular features. The performance of the drug design process depends on the identification of peptides that optimize several molecular properties at the same time. The synthesis of peptides for laboratory characterization is very cost-intensive. Therefore, drug development is a wide field of activity for multi-objective Genetic Algorithms (moGAs). A customized NSGA-II has been especially evolved for biochemical optimization with the focus on producing a great number of very different high quality peptides within a very low number of generations (under 20), termed early convergence. The main task of this paper is to verify empirically the effect of early convergence for this customized NSGA-II within a limited range of population size. Furthermore, an insight into the impact of the interdependence between the population size and the selection procedure is examined with the objective of giving a configuration rule for the selection parameter and the population size exemplary determined for a three-dimensional biochemical minimization problem. Although, this optimization problem is as generic as possible. The performance is assessed on the basis of a convergence indicator especially evolved for our preference of comparing the convergence behavior of populations with different sizes. Moreover, we propose a summarization of open source Java tools that are discussed regarding the potential of an easy implementation of the customized NSGA-II for biochemical optimization.**

***Index Terms*—multi-objective biochemical optimization; population size; average cuboid volume; open source Java tools.**

I. INTRODUCTION

Small peptides are of special interest in the area of drug design as they have some favorable features like conformational restriction, membrane permeability, metabolic stability and oral bioavailability [1]. Nevertheless, for this purpose these peptides have to optimize several molecular features at the same time. As both the synthesis and the laboratory characterization of peptides is very cost-intensive [23], moGAs provide an economical and robust method for peptide identification.
For this purpose, a customized Non-dominated Sorting Genetic Algorithm (NSGA-II) has been evolved and introduced in [2] with a considerable low number of generations and population size, termed early convergence. The NSGA-II is customized w.r.t. the encoding and the components mutation, recombination and selection. Different mutation and recombination methods have been evolved for this purpose and are introduced in [3][4]. These components and their parameter are not only inter related, but are also responsible for the performance of a GA. So far, less work has been done to gain an insight in the influence of the population size on the performance and in the interdependence with the selection operator and its parameters in the case of moGAs. The population size is an important topic in influencing the performance of evolutionary algorithms [5]. Small population sizes tend to result in poor convergence and large populations extend the computational complexity of a GA in finding high quality solutions [6]. Therefore, an adequate population size that results in good performance is challenging. Diverse results have been presented w.r.t. the choice and the handling of the populations size for single-objective GA: Yu et. al [7] study the connection between selection pressure and population size and ratify the concept of interdependence of parameters and operators in GA. The concept of self-adaption is used to overcome the problem of determining the optimal population size. Two forms of self-adaption are used: First, Bäck et al. [8] uses self-adaption as a previous setup and configuration step for evolutionary strategies. The population size then remains the same over all iterations. Second, Arabas et al. [9] introduces a GA with varying population size. The self-adaption of the population size is used throughout the whole GA run and depends among others on different parameters like the reproduction ratio. Eiben et al. [10] provide empirical studies that self-adaption of selection pressure and population size is possible and further rewarding w.r.t. algorithm performance. In this case study, the global parameters tournament size and population size are regulated.

The questions that we consider in this paper are: 1. Do large populations speed up the convergence behavior of the customized NSGA-II for a three-dimensional biochemical minimization problem? 2. Is there a predictable impact between population size and selection? 3. Is there a range of population size which is able to perform well?

These questions are answered in an empirical way: The performance of the customized NSGA-II is assessed w.r.t. its early convergence and a high diversity within the solutions. Some metrics have been proposed to evaluate the convergence behavior of a moGA [24]. These metrics, generally, measure the distance of non-dominated solution sets to the true Pareto front [24]. This makes a comparison of generations with different sizes impossible. Therefore, a convergence indicator is introduced especially for the comparison of the generations with different sizes based on the hypervolume. The favorable features of this indicator are also discussed. Furthermore, we

 ISBN: 978-1-61208-335-3

will discuss available open source Java tools that allow an easy implementation of the customized NSGA-II to solve multi-objective biochemical optimization problems.

The remainder of this paper is organized as follows: Section II describes the components of the customized NSGA-II. Section III provides a comparison of open source Java frameworks focussed on a most simple implementation of the customized NSGA-II. Section IV introduces the new convergence metric and discusses the motivation for its evolution and the indicator features. Section V provides the performance results of the configurations with different population sizes. Section VI gives responses to the questions raised in this section.

II. The customized NSGA-II

In this section, the customized NSGA-II is described as used in the presented experiments. In the previous work [2][3], we have assessed the performance and interaction of different recombination and mutation operators. In these experiments, we have selected the optimal combination of recombination and mutation method that is used within the following experiments. Additionally, we have customized the encoding and selection for the purpose of peptide optimization. The procedure corresponds to the procedure of the traditional NSGA-II [3].

A. The encoding

The individuals are encoded as 20-character strings symbolizing the 20 canonical amino acids. This is the most intuitive way of peptides encoding. The individuals have a fixed length of 20 amino acids.

B. Three-dimensional biochemical minimization problem

We use three fitness functions predicting molecular features. Two fitness functions make use of the primary structure and the third works on the secondary structure. These fitness functions provide physiochemical properties that are used for drug design [11]. Moreover, this combination of fitness functions describes important peptide properties [25].The first fitness function is the calculation of the Molecular Weight (MW) that is an important peptide feature for the purpose of drug design [1]. This fitness function is selected from the open source library BioJava [12]. The second fitness function is the determination of the hydrophilicity (hydro) of a peptide. A hydrophilicity value is assigned to each peptide via the hydrophilicity scale of Hopp and Woods with a window size of the peptide length [13]. These two fitness functions work on the primary structure. The third fitness function determines the optimal global similarity score provided by the Needleman-Wunsch Algorithm (NMW) that is also part of the BioJava library. These three fitness functions act comparatively: individuals are compared to a predefined reference-solution. Therefore, these three objective functions have to be minimized.

C. The recombination operator

The $n-$point recombination operator is used, where n is determined by a linearly decreasing function:

$$x_R(t) = \frac{l}{2} - \frac{l/2}{T} \cdot t \quad (1)$$

that depends on the peptide length l, the total number of the GA generations T and the index of the actual generation t. This operator results in combination with the following mutation in the - so far - best performance.

D. The mutation operator

An adaption of the deterministic dynamic operator of Bäck and Schütz is used to determine the number of mutation points.

$$p_{aBS} = (5 + \frac{l-2}{T-1}t)^{-1}, \quad (2)$$

Again, l describes the peptide length, T the total generation number of the GA and t the index of the actual generation number.

E. The Aggregate Selection

The flow diagram in Figure 1 depicts the selection methods. The Aggregate Selection is tournament-based. From the tournament set individuals are chosen from the first front with a probability p_0 and with a probability $1-p_0$ the individuals are chosen via Stochastic Universal Sampling (SUS). The number N of pointers is the number of fronts and the segments are equal in size to the number of individuals in each front.

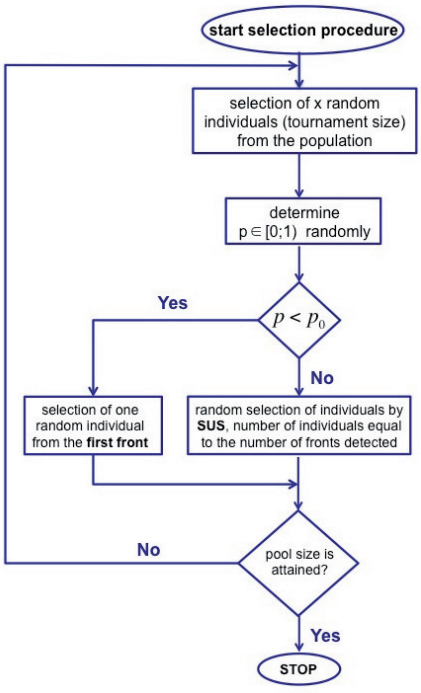

Fig. 1. Aggregate selection strategy

Therefore, the selection method has two parameters, the tournament size and the probability of choosing individuals

from the first front. The tournaments size of 10 has proven to be an optimal choice. The parameter p_0 is challenging w.r.t. the population size.

III. Open source Java frameworks

In this section, we summarize and describe different open source Java tools that provide Genetic Algorithm implementations. The summarization is focussed on Java frameworks for a most simple implementation of BioJava, which provides several physiochemical properties via APIs. The main goal of this framework analysis is the selection of a tool which allows an easy implementation of the proposed customized NSGA-II.

The framework Java API for Genetic Algorithm (JAGA) in its current version 1.0 beta is a research tool developed and supported by the Computer Science Department of the University College London [19]. This tool does not include any moGAs, but it provides a protein string sequence encoding using 20 different characters symbolizing the 20 canonical amino acids. Among others, eight different scales like hydrophobic, aliphatic, aromatic and polar are pre-defined for each canonical amino acid. In addition, it contains for each genotype a parameter-depending crossover and mutation method and a elongation for amino acid patterns. The user interested in a moGA application has to extend this tool for this purpose, but the amino acid character encoding is a clear benefit. Other useful functions are the opportunity of creating a random initial population of protein sequences and the implementation of the Needleman-Wunsch or Smith-Waterman Algorithm.

The framework Metaheuristic Algorithms in Java (jMetal) in its current version 4.3 is an extensive and complex tool especially for moGA applications [20]. It contains beneath NSGA-II the moGA variants: Pareto Envelope-based Selection Algorithm (PESA), improved Strength Pareto Evolutionary Algorithm (SPEA2), improved PESA (PESA2), S-Metric Selection Evolutionary Multiobjective Evolutionary Algorithm (SMS-EMOA), Indicator-Based Evolutionary Algorithm (IBEA) and Multiobjective Evolutionary Algorithm based on Decomposition (MOEA/D). Further, different variation operators are implemented like single-, two- point, Simulated Binary Crossover (SBX) and polynomial, uniform and swap mutation. 'Ranking&crowding selection' is included as the traditional NSGA-II selection method as well as tournament and PESA2 selection. Additionally, jMetal provides several established metrics to evaluate the performance like the hypervolume, Inverse General Distance (IGD), General Distance (GD) and a measure for diversity. A definite advantage of jMetal is the intuitive and clear program construction, which allows an easy algorithmically extension. The disadvantage is a missing character or string encoding.

The framework Java-based Evolutionary Computation Research System (ECJ) in its current version 21 is comparable with jMetal in the issues functional complexity and potential extension. ECJ is developed at George Mason University's Evolutionary Computation Laboratory [21]. It includes the moGAs NSGA-II and SPEA2. Furthermore, different vector representations with corresponding variation operators are included as well as SUS and tournament selection, among others. Moreover, it proposes the potential to read populations from files. It does not show the intuitive and clear program structure of jMetal.

Evolutionary Algorithms workbench (EvA2) is a Java framework developed by the department of computer science at the Eberhard Karls University in Tübingen [22]. It is not only intended for research, but is also deployed for industrial applications and is available under LGPL license. Its specificity is its easy-to-use graphical user interface and provides a MATLAB interface to optimize functions in MATLAB with standard algorithm implementations in EvA2. It also has a client-server structure and provides NSGA-II, PESA and SPEA2 as moGA implementations. A string or character encoding is not implemented and an implementation afterwards is challenging, because encoding affects all parts of the tool box.

The framework Java Class library for Evolutionary Computation (JCLEC) in the current version 4 includes the evolutionary features NSGA-II and SPEA2. It proposes different encodings with various variation operators except string or character encoding, but provides an expendable program structure. Further, selection strategies like tournament and SUS based selection are also included.

Figure 2 gives an overview of the reviewed Java frameworks. These frameworks are compared under the aspects of: (i) configuration of a character or string encoding as an option, (ii) an implementation of NSGA-II, (iii) potential of a simple extension, and (iv) an intuitive program structure according to the moGA components.

TABLE I
Overview of the special framework aspects

	JAGA	jMetal	ECJ	EvA2	JCLEC
(i)	x				
(ii)		x	x	x	x
(iii)		x	x		x
(iv)		x			

Table I reveals that none of the open source Java frameworks attains all required aspects in an adequate level. As a consequence, the experiments are conducted with an user-specific implementation of this customized NSGA-II. In other cases, the open source tool jMetal is a possible alternatives for a user-specific implementation.

IV. Evaluation measures for convergence and diversity

Firstly, the convergence measure is introduced, which has been especially evolved to evaluate generations with different sizes. Subsequently, the features of this indicator are discussed followed by the presentation of measurement for diversity.

A. Introduction of the average cuboid volume

In the past, several metrics have been proposed to evaluate the convergence behavior of populations produced by a moGA.

 ISBN: 978-1-61208-335-3

Usually, they act on the distance of the non-dominated solution set of a generation to the true Pareto front. The hypervolume or the S-metric measures the overlapped space of the non-dominated solution set to a predefined anti-optimal reference point [14]. The hypervolume is a very established convergence metric with its favorable mathematical properties as one reason. Another convergence metric is the D-metric [24]. The D-metric makes use the hypervolume and calculates the coverage difference of two solution sets. A reference set is needed to assess the convergence to the true Pareto front. The C-metric is an appropriate measure to compare the dominance of two Pareto optimal sets [14]. The Error Ratio (ER) is a percentage measure for the number of solutions in a set that are to be found on the true Pareto front [24]. GD is a measure of the average distance between a Pareto optimal solution set to the true Pareto front [15]. It includes the minimal component-wise distance of a solution set to the nearest one on the true Pareto front. The convergence metric of Deb also measures the distance between a solution set and a reference set of the Pareto front [16]. It calculates the average normalized distance for all solutions in the solution set. A recently published convergence metric is the Averaged Hausdorff Distance Δ_p [17]. It is based on GD and the IGD [18].

The reasons for the evolution of a new convergence metric in this paper and in the scientific community in general are multiple: The disadvantage of the metrics D-metric, ER, GD, Δ_p and the convergence metric of Deb is their dependency on the true Pareto front or at least a reference set of Pareto optimal solutions that are usually unknown in the case of real-world MOPs. Furthermore, these metrics are not useful indicators for an entire ranking between generations of different sizes. However, the populations in moGAs are generally limited in size. From a more global point of view, the evaluation and comparison of the global convergence behavior of whole populations - not only the non-dominated solution set of a generation - is required with respect to the influence of the population size or the selection pressure.

For this purpose, a new metric is presented that reflects the convergence behavior of a whole population and is a 'fair' indicator for comparison of generations of different sizes. This Average Cuboid Volume (ACV) is evolved according to the model of the hypervolume. The motivation for the exploitation of the hypervolume model is to profit from its preferable properties as mentioned above. The benefit of this new metric compared to the hypervolume is the low computational complexity as no point ordering is required.

In the following, we assume that the underlying optimization problem is to minimize. The metric calculates the average cuboid volume of the cuboids spanned by the solution points to a pre-defined reference point r:

$$ACV(X) = \frac{1}{n}\sum_{i=1}^{n}\left(\prod_{j=1}^{k}(x_{ij} - r_j)\right), \qquad (3)$$

where n is the population size, k is the number of objectives, x_i are the solutions on the population X and x_{ij} is the $j-th$ component of a solution x_i. It holds $(x_{ij} - r_j) > 0$ as the predefined reference point is chosen as the theoretical minimal limit of the true Pareto front. The lower the indicator values the more positive is the global convergence behavior as the reference point is chosen as a theoretical optimal point.

B. Discussion of the average cuboid volume

The question regarding the suitability of a metric for evaluation depends on the intention of the investigation object and the preferences. ACV is intended to evaluate the global convergence behavior of a whole population with the ultimate aim of comparing solution sets of different sizes according to the proximity to the true Pareto front.

The first expectation that is important for the use of ACV is that the convergence quality shall not change if the number of equally solutions increases. ACV does not fulfill this averaging strategy: Let $x \in \mathbb{R}^k$ be a solution of the optimization problem and $X = \{x\}$. Further, $Y = \{x, ...x\}$ is a set of n equally copies of the solution x, then $ACV(Y) = ACV(X)$.

The second expectation is described by the following observation: An intuitive indicator reflecting the quality of approximation sets of different Pareto front refinements requires 'better' indicator values for the finest approximation set. This effect is demonstrated for ACV by an example also used in [10]:

Example 1: The Pareto front is the line segment between the points $y_1 = (0, 1)$ and $y_2 = (1, 0)$ meaning

$$PF_{true} = \{i \cdot y_1 + (1 - i) \cdot y_2 | i \in (0, 1)\}. \qquad (4)$$

We consider the following three approximation sets of increasing refinement of the Pareto front

$$Y_1 = \{(i \cdot 0.2, 1 - i \cdot 0.2) | i \in \{1, 2, 3, 4\}\}, \qquad (5)$$

$$Y_2 = \{(i \cdot 0.1, 1 - i \cdot 0.1) | i \in \{1, 2, ..., 9\}\}, \qquad (6)$$

$$Y_3 = \{(i \cdot 0.01, 1 - i \cdot 0.01) | i \in \{1, 2, ..., 99\}\}. \qquad (7)$$

The indicator values of ACV for the approximation sets with the reference point $(0, 0)$ are: $ACV(Y_1) = 0.2$, $ACV(Y_2) = 0.183$ and $ACV(Y_3) = 0.167$.

The third preferable expectation of this indicator is the averaging effect. It is trivial that a dominating solution x yields better indicator values than the dominated one y, because $ACV(\{x\}) = \prod_{i=1}^{k}(x_j - r_j) < \prod_{i=1}^{k}(y_j - r_j) = ACV(\{y\})$. From this observation it can be interpreted that if one dominated solution x_1 in the solution set $X = \{x_1, x_2, ..., x_n\}$ is replaced by a dominating one $\bar{x}_1$, then $ACV(\{x_1, x_2, ..., x_n\}) > ACV(\{\bar{x}_1, x_2, ..., x_n\})$. The averaging effect of ACV is illustrated by the example which has also been used for Δ_p [17]:

Example 2: The true discrete Pareto front is described by $P = \{p_i | p_i = (0.1 \cdot (i-1); 1-(i-1) \cdot 0.1)$ with $i = 1, ..., 11\}$. Two solution sets are given by $X_1 = \{x_{1,1}, p_2, ..., p_{11}\}$ and $X_2 = \{x_{2,1}, x_{2,2}, ..., x_{2,11}\}$ with the elements $x_{1,1} = (\epsilon, 10)$ and $x_{2,i} = p_i + (\frac{\epsilon}{2}, 5)$ with $i = 1, ..., 11$. For the outlier $x_{1,1}$

the values $\epsilon = 0.001$ is used for numerical evaluations. X_1 is a better approximation of the true Pareto front, but it contains the outlier $x_{1,1}$. On the other side, X_2 is close to the true Pareto front and the difference of each element to the Pareto front is less than the one of the outlier. As we are interested in an averaging effect, the indicator values of X_1 has to be better than the one of X_2. This is true for ACV as $ACV(X_1) = 0.15$ and $ACV(X_2) = 2.65$.

C. The diversity measure

The measure for diversity calculates the average distance of all pairs of solutions (see [4]):

$$\Delta = \sum_{i,j=1,i<j,i\neq j} \frac{|d_{ij} - \bar{d}|}{N} \qquad \text{with } N = \binom{n}{2}. \qquad (8)$$

$d_{i,j}$ symbolizes the Euclidean distance of two solutions x_i and x_j, $\bar{d}$ is the mean of all measured distances and n is the population size.

V. Simulation onset and experiments

The test runs are performed for different configurations. The configurations are composed of a differing population size (30, 50, 70, 100, 130, 150) and the selection parameters $p_0 = 0\%, 30\%, 50\%$. These parameters have been emphasized by previous experiments. The selection parameter $p_0 = 0\%$ stands for SUS exclusively. Each multi-objective configuration is repeated 20 times until the 18th generation - for statistical reasons. The test runs are evaluated by the convergence indicator ACV and the diversity measure as introduced in the last section. ACV uses the theoretical minimal limit $(0/0/0)$ of the Pareto front as an optimal reference point. Therefore, a good performance is achieved if the ACV value is as low as possible and the diversity value is as high as possible. Boxplots are created for each configuration and for each objective of evaluation (Fig. 2 - Fig. 8). The values of ACV and diversity are scaled under the same criterion for a better graphical presentation. The figures are ordered according to the population size. The standard population size within the customized NSGA-II is 100 [2][3] (Fig. 5). Therefore, the results are discussed w.r.t. an increase and an decrease of this size: In general, a decrease of the population size to 70 and 50 results in an increase of the ACV values and a decrease of the diversity values (Fig. 3, Fig. 4). This means that the convergence and the spread within the solutions is reduced caused by decreasing the population size. The ACV values decrease for a population size of 30 (Fig. 2) independent of the choice of the selection parameter. Moreover, the diversity also decreases and results in the lowest diversity among all configurations. An increase of the population size to 130 results in a decrease of ACV and in an increase of the diversity, once more independent of the selection parameter (Fig. 6). A further increase of the population size to 150 and 200 results in a stagnation of the ACV and diversity values (Fig. 7, Fig. 8).

Further, the effect of the selection parameter is discussed: Varying the population size from 50 to 100 (Fig. 3- Fig. 6),

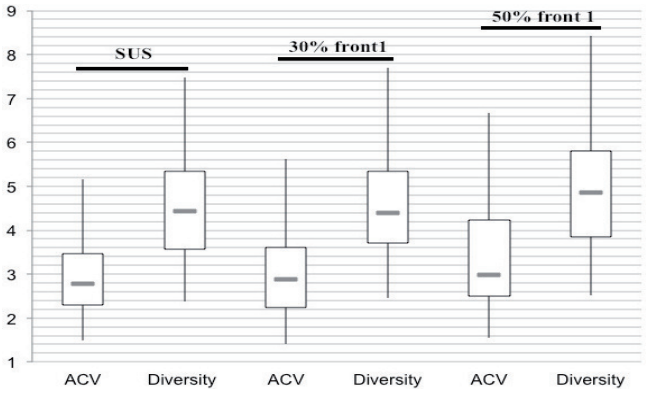

Fig. 2. Population size 30

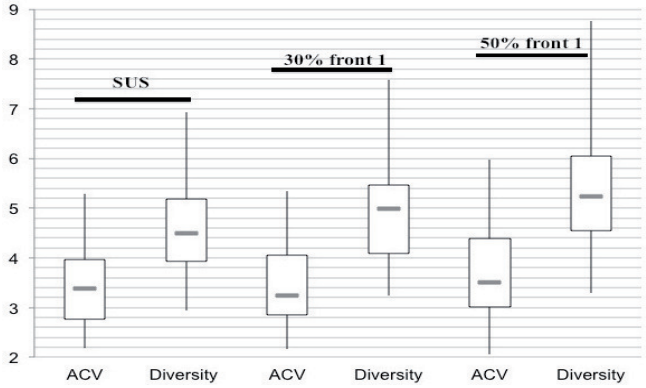

Fig. 3. Population size 50

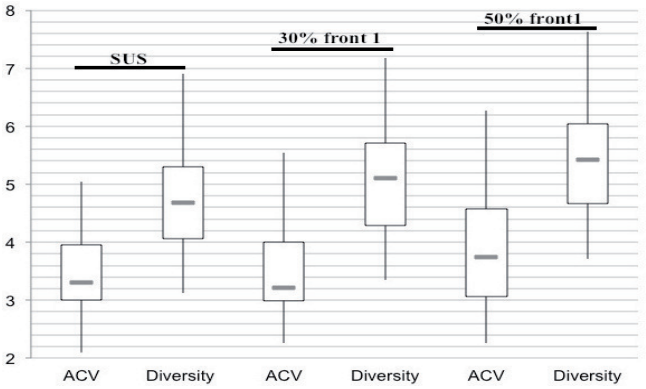

Fig. 4. Population size 70

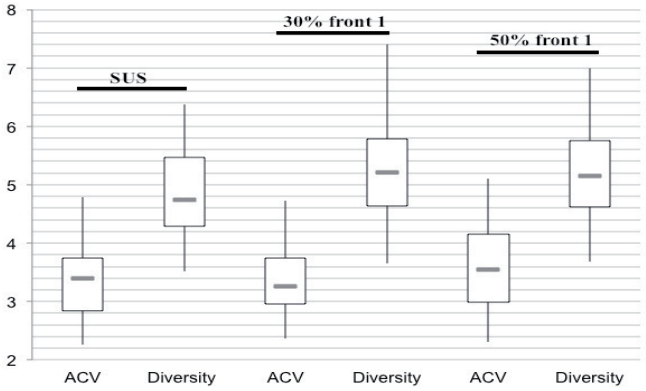

Fig. 5. Population size 100

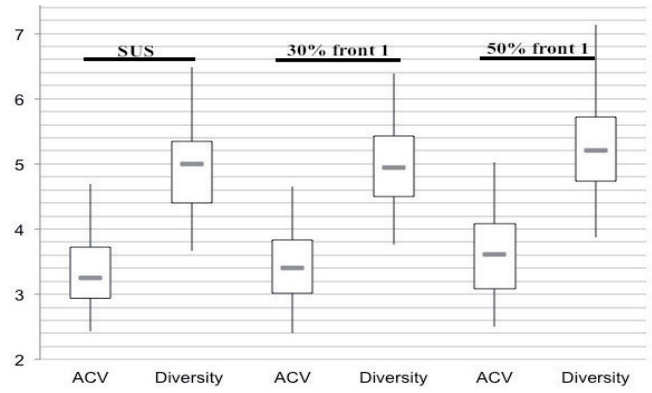

Fig. 6. Population size 130

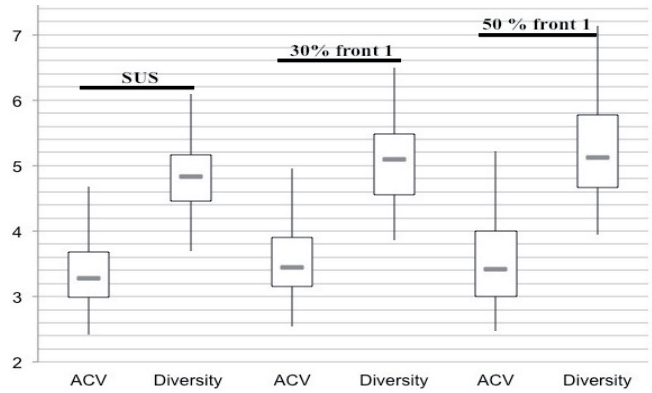

Fig. 7. Population size 150

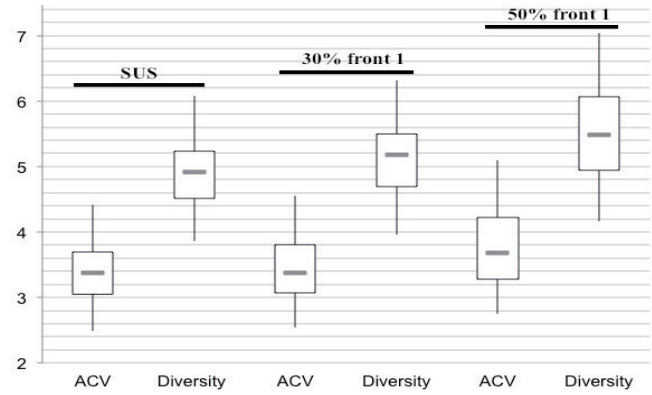

Fig. 8. Population size 200

the ACV values are comparable for $p_0 = 0\%$ (denoted as 'SUS' in the figures) and $p_0 = 30\%$ (denoted as '30% front 1' in the figures), though the diversity improves evidently for $p_0 = 30\%$ compared to SUS. Independent of the population size, $p_0 = 50\%$ (denoted as '50% front 1' in the figures) results in a remarkable increase of the ACV values and only a slight improvement of diversity compared to SUS and $p_0 = 30\%$. For the population sizes from 130 to 200, the influence of the selection parameter is reduced (Fig. 6- Fig. 8): There is only a slight improvement to report in diversity for $p_0 = 30\%$ compared to SUS. The convergence is remarkable reduced for $p_0 = 50\%$, though the diversity is improved.

The best performance of the configurations is received with a population size from 70 to 100 and a selection parameter of 30% as the values for ACV are at most low, whereas the diversity values are at most high. At least, the performance of the configurations with a population size from 50 to 100 with $p_0 = 30\%$ are comparable in convergence and diversity with the performance of the configuration population size of 130 and SUS. Concluding, the best configuration is expectable with a population size in the range from 70 to 100 and a selection parameter of $p_0 = 30\%$.

Regarding the questions presented in the introduction we conclude that an increase of the population size does not result in better performance. The customized NSGA-II provides good performance regarding convergence and diversity within a limited range of population size for the presented three-dimensional minimization problem. Empirically, there is no interdependence between population size and selection: The choice of $p_0 = 30\%$ usually results in the best performance independent of the population size. Therefore, it is not possible to speed up the convergence by increasing or decreasing of the population size and a suitable adaption of the selection parameter.

VI. Conclusion and Future Work

The interdependence of the population size and the selection parameter in this customized NSGA-II is exemplary examined on a generic three-dimensional biochemical minimization problem focused on three central questions: The first question is aimed at the influence of large populations on the convergence speed. Early convergence as a main goal of our moGA is defeated since an increase of the population size results in higher speed of convergence. The experiments show that the optimal population size w.r.t. convergence and diversity is in a limited range from 70 to 100. An increase of the population size above 100 results in a stagnation of the convergence behavior and the diversity. A population size lower than 50 does not provide a convincing diversity within the solutions. Our second question is focused on the impact of the population size and the selection parameter. A configuration rule for the selection parameter depending on the population size is necessary in the case of a large dependence of both. However, the experiments do not reveal an interdependence of the population size and the selection parameter. Though, the diversity of the configurations with a population size from 50 to 100 is remarkably improved with a selection parameter of 30% compared to $p_0 = 0\%$ (SUS). Higher values for p_0 are not advisable as the speed of convergence is reduced. The third question asks for a range of the population size providing the best performance: This range is fixed to a population size from 70 to 100 based on the evaluation of the experiments.

The convergence performance of the experiments is assessed via a newly introduced convergence indicator, which is especially evolved to compare the convergence behavior of populations with different sizes. It is based on the established

hypervolume and calculates the average cuboid volume of the cuboids spanned by the solution points to a pre-defined reference point, which is chosen as a theoretical optimal point. The benefit of ACV compared to the hypervolume is the lower computational complexity and the choice of the reference point. It is easier to determine an optimal point than an anti-optimal one for real world applications. Furthermore, a comparison of Java frameworks is submitted as a guidance for a simple implementation of the customized NSGA-II. The frameworks are compared to the aspects of a character or string encoding, the implementation of NSGA-II and the potential of a most simple extension.

For future work, we currently work on the confirmation of these results on a four-dimensional biochemical optimization problem. Further, we will evolve a selection strategy based on ACV for ongoing improvements.

REFERENCES

[1] D. J. Craik, D. P. Fairlie, S. Liras, and D. Price, "The Future of Peptide-based Drugs," Chemical Biology & Drug Design, 81(1), 2013, pp. 136-147.

[2] S. Rosenthal, N. El-Sourani, and M. Borschbach, "Introduction of a Mutation Specific Fast Non-dominated Sorting GA Evolved for Biochemical Optimization," SEAL 2012, LNCS 7673, 2012, pp. 158-167.

[3] S. Rosenthal, N. El-Sourani, and M. Borschbach, "Impact of Different Recombination Methods in a Mutation-Specific MOEA for a Biochemical Application," L. Vanneschi, W. S. Bush, and M. Giacobini (Eds.): EvoBIO 2013, LNCS 7833, 2013, pp. 188-199.

[4] S. Rosenthal and M. Borschbach, "A Benchmark on the Interaction of Basic Variation Operators in Multi-Objective Peptide Design evaluated by a Three Dimensional Diversity Metric and a Minimized Hypervolume," M. Emmerich et al. (eds.): EVOLVE - A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation IV, 2013, pp. 139-153.

[5] J. T. Alander, "On Optimal Population Size of Genetic Algorithms," in Proceedings of the IEEE Computer Systems and Software Engineering, 1992, pp. 65-69.

[6] V. K. Koumousis and C. P. Katsaras, "A Saw-Tooth Genetic Algorithm Combining the Effects of Variable Population Size and Reinitialization to Enhance Performance," IEEE Transactions on Evolutionary Computation, vol. 10, no. 1, 2006, pp. 19-28.

[7] T.-L. Yu, K. Sastry, D. E. Goldberg, and M. Pelikan, "Population sizing for entropy-based model building in genetic algorithms," Illinois Genetic Algorithms Laboratory, University of Illinois, Tech. Rep., 2006.

[8] T. Bäck, A. Eiben, and V. der. Vaart, "An empirical study on GAs without parameters," in Proceedings of the 6th International Conference on Parallel Problem Solving from Nature, 2000, pp. 315-324.

[9] Z. M. Jaroslaw Arabas and J. Mulawka, "GAVaPSa genetic algorithm with varying population size," in Proceedings of the IEEE International Conference on Evolutionary Computation, 1995, pp. 73-78.

[10] A. E. Eiben, M. C. Schut, and A. R. Wilde, "Is Self-Adaption of Selection Pressure and Population Size Possible? a Case Study," in Parallel Problem Solving from Nature - PPSN IX, vol. 4193, 2006, pp. 900-909.

[11] T. Sovany et al., "Application of Physiochemical Properties and Process Parameters in the Development of a Neural Network Model for Prediction of Tablet Characteristics," AAPS PharmSciTech, vol. 14(2), 2013, pp. 511-516.

[12] BioJava: CookBook, URL: http://www.biojava.org/wiki/BioJava/ [retrieved: December, 2013].

[13] T. P. Hopp, K. R. Woods, "A computer program for predicting protein antigenic determinants," Mol Immunol, 20(4), 1983, pp. 483-489.

[14] E. Zitzler and L. Thiele, "Multiobjective Optimization using Evolutionary Algorithms - a Comparative Case Study," in A. E. Eiben, T. Bäck, M. Schoenauer, and H. P. Schwefel (EDS.), Fifth International Conference on Parallel Problem Solving form Nature (PPSN-V), 1998, pp. 292-301.

[15] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective Evolutionary Algorithm Test," in Proceedings of the 1999 ACM Symposium on Applied Computing, San Antonio, Texas, 1999, pp. 351-357.

[16] K. Deb, S. Jain, "Running performance metrics for Evolutionary Multi-objective Optimization," Kan GAL Report No. 2002004, Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology Kanpur, 2002.

[17] O. Schütze, X. Esquivel, A. Lara, and C. A. Coello Coello, "Using the Averaged Hausdorff Distance as a performance measure in evolutionary multiobjective optimization," IEEE Transactions on Evolutionary Computation, vol. 16(4), 2012, pp. 504-522.

[18] C. A. Coello Coello and N. Cruz Cortis, "Solving Multiobjective Optimization Problems using an Aritifical Immune System. Genetic," Programming Evolvable Mach., vol. 6 (2), 2005, pp. 163-190.

[19] Java API for Genetic Algorithm (JAGA), URL: www.jaga.org/ [retrieved: January, 2014].

[20] Metaheuristic Algorithms in Java (jMetal), URL: www.jmetal.sourceforge.net/ [retrieved: January, 2014].

[21] Java-based Evolutionary Computation Research System (ECJ), URL: www.cs.gmu.edu/∼edab/projects/ecj/ [retrieved: January, 2014]

[22] Evolutionary Algorithms workbench (EvA2), URL: www.ra.cs.uni-tuebingen.de/software/EvA2/introduction.html/ [retrieved: January, 2014]

[23] N. Röckendorf, M. Borschbach, and A. Frey, "Molecular Evolution of Peptide Ligands with Custom-tailored Characteristics," PLOS Comput Biol 8(12), 2012

[24] G. Grosan, M. Oltean, and D. Dumitrescu, "Performance Metrics for Multiobjective Evolutionary Algorithms," Proceedings of Conference on Applied and Industrial Mathematics (CAIM), 2003

[25] D. Heider et al. "A Computational Approach for the Identification of Small GTPases based on Preprocessed Amino Acid Sequences," in Technol. Cancer Res. Treat 8, 2009, pp. 333-341.

 ISBN: 978-1-61208-335-3