Closed state model for understanding the dynamics of MOEAs

Hugo Monzón, Hernan Aguirre, Sébastien Verel, Arnaud Liefooghe, Bilel Derbel, Kiyoshi Tanaka

Closed State Model for Understanding the Dynamics of MOEAs

Hugo Monzón
Shinshu University
Nagano, Japan
hugo91@gmail.com

Hernán Aguirre
Shinshu University
Nagano, Japan
ahernan@shinshu-u.ac.jp

Sébastien Verel
Univ. Littoral Cote d'Opale
Calais, France
verel@lisic.univ-littoral.fr

Arnaud Liefooghe
Univ. Lille, CNRS, CRIStAL
Inria Lille – Nord Europe
Lille, France
arnaud.liefooghe@univ-lille1.fr

Bilel Derbel
Univ. Lille, CNRS, CRIStAL
Inria Lille – Nord Europe
Lille, France
bilel.derbel@univ-lille1.fr

Kiyoshi Tanaka
Shinshu University
Nagano, Japan
ktanaka@shinshu-u.ac.jp

ABSTRACT

This work proposes the use of simple closed state models to capture, analyze and compare the dynamics of multi- and many-objective evolutionary algorithms. Two- and three-state models representing the composition of the instantaneous population are described and learned for representatives of the major approaches to multi-objective optimization, i.e. dominance, extensions of dominance, decomposition, and indicator algorithms. The model parameters are trained from data obtained running the algorithms with various population sizes on enumerable MNK-landscapes with 3, 4, 5 and 6 objectives. We show ways to interpret and use the model parameter values in order to analyze the population dynamics according to selected features. For example, we are interested in knowing how parameter values change for a given population size with the increase of the number of objectives. We also show a graphical representation capturing in one graph how the parameters magnitude and sign relate to the connections between states.

CCS CONCEPTS

•**Theory of computation → Evolutionary algorithms;**

KEYWORDS

Empirical study, Working principles of evolutionary computing, Genetic algorithms, Multi-objective optimization

ACM Reference format:
Hugo Monzón, Hernán Aguirre, Sébastien Verel, Arnaud Liefooghe, Bilel Derbel, and Kiyoshi Tanaka. 2017. Closed State Model for Understanding the Dynamics of MOEAs. In *Proceedings of the Genetic and Evolutionary Computation Conference 2017, Berlin, Germany, July 15–19, 2017 (GECCO '17),* 8 pages.
DOI: http://dx.doi.org/10.1145/3071178.3071259

GECCO '17, Berlin, Germany
2017. 978-1-4503-4920-8/17/07.
DOI: http://dx.doi.org/10.1145/3071178.3071259

1 INTRODUCTION

In recent years, multi-objective evolutionary algorithms (MOEAs) have been used to solve complex optimization problems. However, as the number of objectives increases, a reduction in performance is becoming an important issue for several classes of MOEAs. This is not only due to the dimensionality of the objective space, but also to the lack of specific algorithmic components that can accurately accommodate the evolutionary search within the specific properties of a many-objective optimization problem. It is for instance well known that when the number of objectives is two or three, standard selection and replacement of individuals are usually robust enough to evolve a population of reasonable size towards an approximation set of good quality. However, this is obviously not the case when the number of objectives goes above three, since the number of mutually non-dominated solution increase drastically, which makes it more challenging for standard operators, configurations or settings to maintain a population at an accurate diversity level, while ensuring a good convergence towards the high-dimensional Pareto set [6].

New algorithms and new design components, with possibly additional parameters and tuning procedures, are being designed to overcome the loss of performance experienced when tackling many-objective problems [8]. The success of new approaches or methods to many-objective optimization is then often measured by its ability to reach a better approximation set using some quality indicators. However, there is not much work aiming at understanding the dynamics of the search process implied by particular design choices, especially how the intrinsic properties of the population maintained in the course of the search evolve, and what makes this evolution successful or not. As a result, most of such design choices could only rely on intuition or problem-specific issues. In this paper, we advocate for a more systematic approach to elicit the search behavior of many-objective optimization algorithms which we consider as a complex process that should be modeled using high-level mathematical tools. This is actually to be viewed as a first step allowing to gain a fundamental knowledge about the main issues that one might face when tackling a many-objective optimization problem with a particular evolutionary algorithm, which shall hopefully lead to more robust and more efficient design choices. Dynamic modeling using Markov chains have proved useful [7][9] to advance our understanding of the behaviour and convergence of canonical evolutionary algorithms on single objective optimization.

We consequently propose to introduce a simple high-level model to capture the evolution of the population maintained by the evolutionary multi-objective search, in an attempt to better grasp the dynamics of the corresponding algorithm and to obtain a better fundamental understanding of their internal operations. More precisely, inspired by state-space representation in control engineering and by the modeling of complex system dynamics, we propose a closed state model and study its accuracy in capturing the evolution of simple and high-level problem-independent features characterizing the population at any time of the search process. The proposed model can also be viewed as a finite state automaton that allows to obtain a compact snapshot of the population, and to estimate the next state of the search. Hence, it can be used to acquire meaningful information about the behavior and possible performance of an algorithm without fully running it. Moreover, the parameters used by the model can be used to compare different algorithms in the sense that they allow to map the complex runtime search behavior of an algorithm to a simple and relatively small set of numerical values. Our proposed model is studied using a simple setting in which two and three states are used to effectively capture the dynamics of many-objective optimizers for a set of selected features in enumerable MNK-landscapes. Our contributions and findings are the by-product of a high-level mathematical modeling effort combined with a throughout empirical analysis requiring the experimentation of different evolutionary optimization algorithms, and the use of different calibration techniques from machine learning and statistical analysis.

The rest of the paper is organized as follows. Section 2 introduces the method with an explanation on the features, the proposed models, and how parameter estimation for them is done. Section 3 describes the experimental setup, which algorithms were selected and what information was collected when running them. Section 4 covers the experimental analysis performed to find parameters on the models for the collected data on representative algorithms, verification of these parameters, and how the obtained parameters can be used to analyze the dynamics of the algorithms. Section 5 summarizes the contributions of this work and discusses further investigations.

2 METHOD

2.1 Features

In order to study the dynamics of MOEAs, we here focus on features that capture the instantaneous composition of the population. Table 1 gives generational search assessment indices that have been used before to study the behavior and performance of MOEAs [1]. Most of these indices are calculated with respect to the Pareto Optimal Set (POS) and are useful to analyze and verify whether the algorithms are capable of finding the optimal set. Thus, to use these indices a problem where the POS is known or can be computed by enumeration is required.

In this work we use some of these indices as features to keep track of the algorithms' dynamics, namely the number of dominated δ_t (DOM), non-dominated $\bar{\delta}_t$ (ND), non-dominated non Pareto Optimal γ (NDNP), and Pareto Optimal solutions τ_t (PO).

Although we consider indices requiring the POS to be known, notice that the proposed models can be used with other features associated with the population dynamics, as long as they comply with assumptions described in the Subsection 2.2

2.2 State Model with Linear Relations

To capture the population dynamics of MOEAs, we assume the following: (i) The population can be split in two or more non-overlapping groups using one or various rules, (ii) a group size variates in a linear way but the population size remains constant, (iii) the model gives the proportion of population members in each group at time t, (iv) the aim is to track the size of the groups within a constant population size, i.e. the change of proportions, and not the individuals themselves. Based on these assumptions we propose two- and three-state models.

2.2.1 *Two-State Model.*

The first simple model splits the population into two different groups, or states. At each generation t, x_t and y_t represent the proportion of the population that belongs in each state. Thus, $x_t + y_t = 1$. The differential equations that describe the two-state model in time discrete form are as follows:

$$\begin{cases} x_{t+1} = (1-\alpha)x_t + \beta y_t \\ y_{t+1} = \alpha x_t + (1-\beta) y_t \\ 1 = x_t + y_t, \end{cases} \tag{1}$$

where α and β are coefficients that describe the loss in x_t and y_t, respectively. Since the model is closed, a loss in x_t becomes a gain in y_t, and vice versa.

Figure 1 illustrates a two-state model that uses the *Pareto dominance* relation to split the population and to capture the dynamics of non-dominated (x_t) and dominated (y_t) solutions. Of course, the two-state model can be applied to other combination of features, by using an appropriate rule to split the population. For example, using *membership to the POS* as a criterion, a two-state model can capture the dynamics of PO solutions retained within the population. In such a model, x_t represents the fraction of PO solutions in the population, and y_t the fraction of solutions that are not PO solutions i.e. non-dominated solutions but not PO solutions plus the dominated solutions. Using the nomenclature described in Table 1, $x_t = PO = \tau_t$ and $y_t = NDNP + DOM = \gamma_t + \delta_t$. These two-state models are summarized in Table 2.

With the two-state model, given the values of each state at generation t and the parameters α and β for a given algorithm, we can estimate the values of the states at future generations.

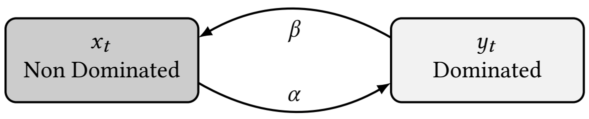

Figure 1: A two-state model.

2.2.2 *Three-State Model.*

In three-state models, we split the population into three non-overlapping groups, or states, in order to analyze the dynamics with more details. Similar to the two-state model, x_t, y_t and z_t represent the proportion of the population that belongs to each state at time t. Thus, $x_t + y_t + z_t = 1$.

Table 1: Generational search assessment indices.
$\mathcal{F}_1$: First front. POS: Pareto Optimal Set. t: Current generation. P: whole population including $\mathcal{F}_1$.

Abbr.	I_t	Formula	Comment
PO	τ_t	$\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{F}_1(t) \wedge \boldsymbol{x} \in POS\}$	PO solutions
POpnew	τ_t^+	$\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{F}_1(t) \wedge \boldsymbol{x} \notin \mathcal{F}_1(t-1) \wedge \boldsymbol{x} \in POS\}$	Possibly new PO solutions
POold	τ_t^-	$\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{F}_1(t) \wedge \boldsymbol{x} \in \mathcal{F}_1(t-1) \wedge \boldsymbol{x} \in POS\}$	Old PO solutions
POanew	τ_t^*	$\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{F}_1(t) \wedge \boldsymbol{x} \notin \cup_{k=1}^{t-1}\mathcal{F}_1(k) \wedge \boldsymbol{x} \in POS\}$	Absolutely new PO solutions
NDNP	γ_t	$\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{F}_1(t) \wedge \boldsymbol{x} \notin POS\}$	Non-dominated, not PO sol.
ND	$\bar{\delta}_t$	$\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{F}_1(t)\}$	Non-dominated sol. (incl. PO sol.)
DOM	δ_t	$\{\boldsymbol{x} : \boldsymbol{x} \in P \wedge \boldsymbol{x} \notin \mathcal{F}_1(t)\}$	Dominated solutions

The differential equations that describe the three-state model in time discrete form are as follows:

$$\begin{cases} x_{t+1} = (1-(\alpha+\beta))x_t + \bar{\alpha}y_t + \bar{\beta}z_t \\ y_{t+1} = \alpha x_t + (1-(\bar{\alpha}+\gamma))y_t + \bar{\gamma}z_t \\ z_{t+1} = \beta x_t + \gamma y_t + (1-(\bar{\beta}+\bar{\gamma}))z_t \\ 1 = x_t + y_t + z_t, \end{cases} \tag{2}$$

where α and β are coefficients that describe the loss in x_t that becomes gain for y_t and z_t, respectively. Likewise, $\bar{\alpha}$ and γ are coefficients that describe the loss in y_t that becomes gain for x_t and z_t, respectively. Similarly, $\bar{\beta}$ and $\bar{\gamma}$ are coefficients that describe the loss in z_t that becomes gain for x_t and y_t, respectively.

In the previous section, the dominance relation was used to split the population into non-dominated and dominated solutions. Adding *membership to the POS* as another rule, we can further split the non-dominated group into PO solutions and non-dominated non-PO solutions (NDNP). Figure 2 illustrates such a three-state model. Using the nomenclature described in Table 1, $x_t = PO = \tau_t$, $y_t = NDNP = \gamma_t$ and $z_t = DOM = \delta_t$.

The three-state model can be applied to other combination of features as well. For example, adding the *not present at time* $(t-1)$ rule to verify whether a PO solution was also present in the previous generation, we can have a model with the following states: PO solutions that are possibly new, i.e. not present in the previous generation, PO solutions that were present in the previous generation, and the rest of non-dominated and dominated solutions. Using the nomenclature described in Table 1, $x_t = POpnew = \tau_t^+$, $y_t = POold = \tau_t^-$, and $z_t = NDNP + DOM = \gamma_t + \delta_t$. Similar, adding the *not present from time 0 to* $(t-1)$ rule to verify whether a PO solution was also present in any previous generation, we could analyze the dynamics of PO solutions that are absolutely new at time t, i.e. we could study the discovery of PO solutions and relate dynamics to performance. Accumulating the absolutely new PO solutions, we can compute the resolution of the approximation [1], which can be used as a performance measure. These three-state models are summarized in Table 2.

Similar to the two-state model, knowing initial values for the states and the parameters of the model, we can use the above equations to estimate the values of each state at future generations.

2.3 Model Parameter Estimation

2.3.1 Model Transformation. In order to estimate the model parameters based on experimental data, we first change the representation of the systems of equations that describe the model, since their recursive nature do not allow us to proceed directly. Here, we use a simple coordinate transformation, by first representing the equations of the two-state model in a matrix form as shown below:

Table 2: Possible models from different combinations of generational search assessment indices.

Two-state models

x_t	y_t	Model No.
ND	DOM	2-1
PO	NDNP + DOM	2-2

Three-state models

x_t	y_t	z_t	Model No.
PO	NDNP	DOM	3-1
POpnew	POold	NDNP+DOM	3-2
POanew	PO - POanew	NDNP+DOM	3-3

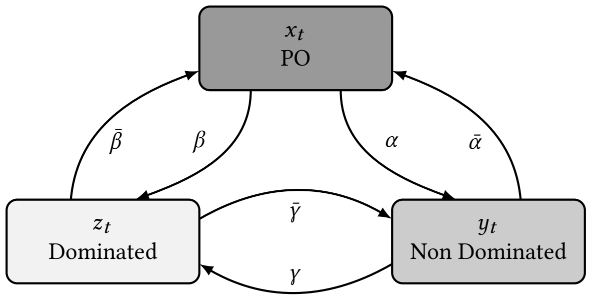

Figure 2: A three-state model.

$$A = \begin{bmatrix} (1-\alpha) & \beta \\ \alpha & (1-\beta) \end{bmatrix} \qquad B_{t-1} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \end{bmatrix}$$

$$X_t = AB_{t-1} \tag{3}$$

By finding the eigenvalue decomposition of A, we obtain the eigenvalues λ_1 and λ_2. By using the constants from the eigenvectors (k_{11}, k_{12}) and (k_{21}, k_{22}) we can express B_t as:

$$\begin{cases} x_t = k_{11}\lambda_1^t + k_{12}\lambda_2^t \\ y_t = k_{21}\lambda_1^t + k_{22}\lambda_2^t. \end{cases} \tag{4}$$

A similar transformation process can be applied to the three-state model, which gives us the following system:

$$\begin{cases} x_t = k_{11}\lambda_1^t + k_{12}\lambda_2^t + k_{13}\lambda_3^t \\ y_t = k_{21}\lambda_1^t + k_{22}\lambda_2^t + k_{23}\lambda_3^t \\ z_t = k_{31}\lambda_1^t + k_{32}\lambda_2^t + k_{33}\lambda_3^t. \end{cases} \tag{5}$$

To obtain back the original parameters, we use the obtained values for k and λ, and we express them in a matrix form as K and L. Then we need to solve Eq. (6).

Two-States

$$K = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \qquad L = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

Three-States

$$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \qquad L = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

$$KLK^{-1} = A \tag{6}$$

2.3.2 Model Fitting. The fit of data to a two-state model is performed by taking the system of Eq. (4), and by numerically solving each equation following the process illustrated in Figure 3. Here we define the problem of finding a good set of parameters that gives an optimal fit for the model as an optimization problem, where we minimize the mean square error (*mse*) or $\frac{1}{n}\sum_{i=1}^{n}(\bar{X} - X)^2$, between the data of the state corresponding to the equation we are fitting X and the values produced by the model $\bar{X}$.

To do so, we use the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) [5], a single-objective numerical optimizer, and set the *mse* between $\bar{X}$ and X as the function to be minimized. To improve the results, we run CMA-ES several times using as initial values the ones obtained previously, and using the R^2 of the fitting to stop the process, when it reaches a value of 0.95 or larger, or when no further improvement is seen.

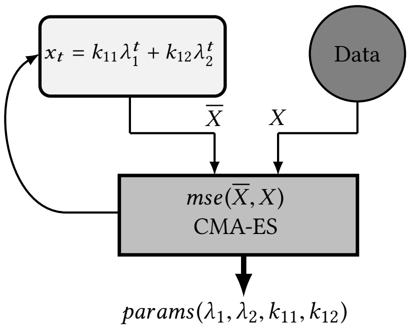
$x_t = k_{11}\lambda_1^t + k_{12}\lambda_2^t$
Data
$\overline{X}$
X
$mse(\overline{X}, X)$
CMA-ES
$params(\lambda_1, \lambda_2, k_{11}, k_{12})$

Figure 3: Fitting a two-state model.

For the three-state model, we use the system of Eq. (5) following the two step process illustrated in Figure 4. In the first step we select one equation of the system and fit all parameters λ_i and k_{ij}. In step two, we reuse the common parameters λ_i previously fitted, and fit only the parameters k_{ij}. We decided to proceed in this way because when all parameters λ_i and k_{ij} were fitted separately for each equation we observed small differences in the obtained λ_i values, which produced inconsistencies on the model parameters once changed back into the original representation.

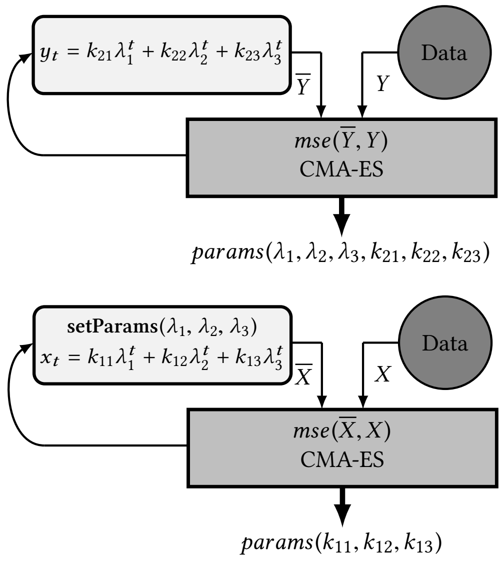
$y_t = k_{21}\lambda_1^t + k_{22}\lambda_2^t + k_{23}\lambda_3^t$
Data
$\overline{Y}$
Y
$mse(\overline{Y}, Y)$
CMA-ES
$params(\lambda_1, \lambda_2, \lambda_3, k_{21}, k_{22}, k_{23})$
setParams(λ_1, λ_2, λ_3)
$x_t = k_{11}\lambda_1^t + k_{12}\lambda_2^t + k_{13}\lambda_3^t$
Data
$\overline{X}$
X
$mse(\overline{X}, X)$
CMA-ES
$params(k_{11}, k_{12}, k_{13})$

Figure 4: Fitting a three-state model.

3 EXPERIMENTAL SETUP

The test problems used are MNK-landscapes [3], a multi-objective problem generator, where the parameter K controls the ruggedness of the landscape, M is the number of objectives, and N the number of variables. In this study, we randomly generate instances with K = 1 bit, N = 20 bits and M = 3,4,5,6 objectives.

Models are created over data collected from five different representative multi- and many-objective optimization algorithms, selected from conventional approaches such as Pareto dominance, relaxation of Pareto dominance, performance indicators, and decomposition. More concretely, we consider the following algorithms in their out-of-the-box implementation: Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [4], Adaptive ϵ-Sampling ϵ-Hood (AϵSϵH) [2], Indicator Based EA (IBEA) with the ϵ-indicator and hypervolume (HV) indicator [11], and MOEA based on Decomposition (MOEA/D) [10].

Each algorithm was run 30 times setting the number of generations to 100. The number of dominated (DOM), non dominated (ND), non dominated non PO (NDNP) and PO individuals at each generation were collected, including the initial population. During the model parameter estimation, the average value over the 30 runs were computed. For each number of objectives, we also consider various population sizes, as indicated in Table 3. We build a model, per configuration, for each algorithm.

4 EXPERIMENTAL ANALYSIS

4.1 Goodness of Fit

We performed the parameter estimation according to the method detailed in Section 2 for all models. Due to space restrictions, in this section we report results only for population size 200 and

Table 3: Population sizes for each number of objectives.

Objectives	Population Sizes (Pop)
3	50, 100, 200
4	50, 100, 200, 500, 1000
5	50, 100, 200, 2000, 4000
6	50, 100, 200, 2000, 5600, 11200

3 objectives. The coefficient of determination (R^2) is reported in Table 4 for each fitting. In the two-state model case, we omit the R^2 for the feature dominated DOM since the values are the same as for non-dominated ones ND.

The R^2 measures the "goodness" of the fit. It can be interpreted as the proportion of variability of the response variable explained by the regression model. The closer R^2 to 1.0, the better the fitting. Actually, an R^2-value of 1.0 indicates that the regression model perfectly fits the experimental data. We must note however that R^2 cannot determine whether the estimate and predictions are biased. In this case, we are comparing the values produced by the model against the mean of the data, for which we have high values in all algorithms. A visual comparison is pictured in Figures 5 and 6, which show a good fit between the model (red line) and the mean of the runs (black line).

Table 4: R^2 on average indice-values for M=3, Pop=200.

	Two States	Three States		
Alg.	R^2 **ND**	R^2 **PO**	R^2 **NDNP**	R^2 **DOM**
NSGA-II	0.99059	0.99418	0.98381	0.99182
AϵSϵH	0.99624	0.99612	0.98670	0.99292
IBEA$_{hv}$	0.98756	0.99427	0.96424	0.99465
IBEA$_{\epsilon}$	0.99091	0.99009	0.95336	0.98749
MOEA/D	0.96994	0.98858	0.95963	0.97246

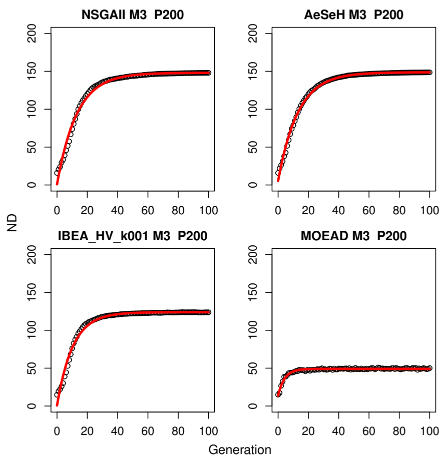

Figure 5: Fitting two-state models to mean ND data.

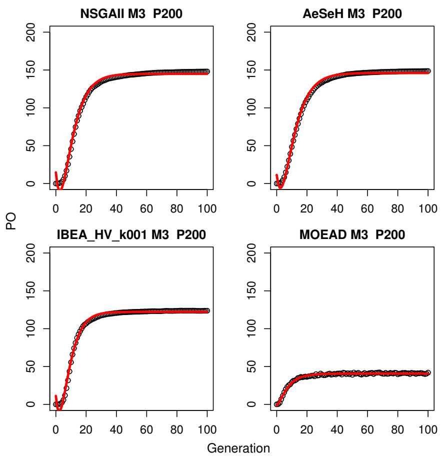

Figure 6: Fitting three-state models to mean PO data.

4.2 Model Verification

For a simple verification of the models, we set the initial states ($t = 0$) with the data from the initial population (generation 0), for each run of each configuration, and use the model to generate estimates for all generations ($t = 1, \ldots, 100$), using the values estimated at time t as input to estimate the values at $t + 1$. We plot in red the estimates of the 30 runs generated by the model and in black the data of the 30 runs obtained from the algorithms. Figure 7 shows the result for non-dominated ND feature of the two-state model, and Figure 8 shows the result for the PO feature of the three-state model.

Visually, from these figures we can see that the estimates obtained using the two- and three-state models follow the trend of the experimental data. Note that the estimates by the model go through the middle of the dataset, since the model parameters were trained based on the mean. Table 5 reports the average R^2 obtained from the estimated data for each run against the real data. The high R^2 values match the previous results, showing that the model produces a good-quality estimation, even though it was fed only with the data from the initial population for each run.

Table 5: Average R^2 of individuals runs for M=3, Pop=200.

	Two States	Three States		
Alg.	R^2 **ND**	R^2 **PO**	R^2 **NDNP**	R^2 **DOM**
NSGA-II	0.96343	0.96312	0.65107	0.96353
AϵSϵH	0.97240	0.97370	0.85654	0.96794
IBEA$_{hv}$	0.95669	0.97257	0.79520	0.97011
IBEA$_{\epsilon}$	0.96299	0.95750	0.69450	0.96241
MOEA/D	0.64085	0.79477	0.35883	0.64463

In addition, we performed cross-validation using the repeated random sub-sampling validation. The input data was split with a proportion of 80/20. Thus, from the 30 runs performed for each algorithm, data from 24 runs where used as training data for model fitting while the data of the remaining 6 runs were used for the validation step. We repeated this process 30 times, each time selecting

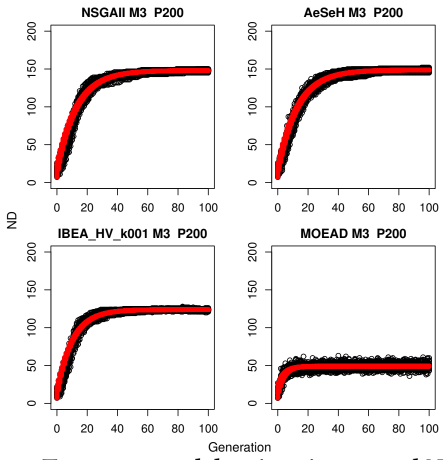

Figure 7: Two-state model estimation vs real ND data.

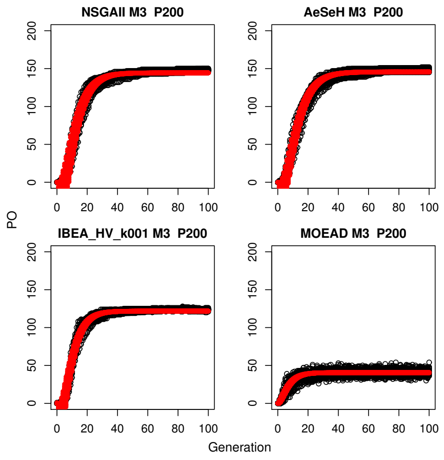

Figure 8: Three-state model estimation vs real PO data.

different runs for model fitting and validation. Similar to Table 4 and Table 5, Table 6 reports the R^2 of the model fittings using the training data and Table 7 the R^2 between the estimation produced by the models and the validation data. For validation, the models use the parameters learned with the training data and its initial states are set with the values of the features present in the initial population of the unseen data.

Table 6: R^2 on average values of training runs for M=3, Pop=200.

	Two States	Three States		
Alg.	R^2 **ND**	R^2 **PO**	R^2 **NDNP**	R^2 **DOM**
NSGA-II	0.99052	0.99415	0.98253	0.99172
AϵSϵH	0.99594	0.99612	0.98614	0.99275
IBEA$_{hv}$	0.98721	0.99415	0.96331	0.99452
IBEA$_\epsilon$	0.99077	0.98999	0.95334	0.98741
MOEA/D	0.96659	0.98709	0.95334	0.96899

Table 7: average R^2 on validation runs for M=3, Pop=200.

	Two States	Three States		
Alg.	R^2 **ND**	R^2 **PO**	R^2 **NDNP**	R^2 **DOM**
NSGA-II	0.96337	0.95966	0.59491	0.96015
AϵSϵH	0.97195	0.97358	0.84680	0.96841
IBEA$_{hv}$	0.95726	0.97218	0.78932	0.96957
IBEA$_\epsilon$	0.96340	0.95627	0.67900	0.96155
MOEA/D	0.64381	0.78746	0.35177	0.64307

For the training set, from the results of Table 6, we notice that we have similar satisfying results to the ones obtained when we used all the runs as training data. As for the validation set, from Table 7 we see equally good results for the two-state model, but somehow low R^2 for some algorithms in the three-state models, in particular MOEA/D, for which the variability of the data affects the R^2, as can be visually appreciated in Figures 7 and 8.

4.3 Parameter Analysis

The parameters, obtained from fitting the models to the data, encapsulate the dynamics of the population composition in a small set of values and provide us with a valuable tool for analysis.

A parameter relates two states within the model. To interpret them, we need to know what they encapsulate according to the equations of the model. For example, α in the three-state model relates the number of PO and NDNP solutions. Expanding the first two equations of system (2) and focusing on α we get the following

$$PO_{t+1} = PO_t - \alpha PO_t + \cdots$$

$$NDNP_{t+1} = NDNP_t + \alpha PO_t + \cdots .$$

Note that the values of PO and NDNP at time $t+1$ change in opposite directions, both proportionally to PO at time t by αPO_t. Hence, a loss in one state becomes a gain in the other. However, α can take positive or negative values. Thus, when $\alpha > 0$, the model tell us that the number of PO solutions reduces with time proportionally to its previous value and the same amount increases in NDNP. On the other hand, when $\alpha < 0$, the model says that the number of PO solutions increases proportionally to its previous value and the same amount reduces in NDNP. In the above example, $\alpha > 0$ can be interpreted as a negative feedback of PO on PO, whereas $\alpha < 0$ a positive feedback of PO on PO.

We should emphasize that changes in the number of individuals in each state do not imply transition of individuals between states. We do not track whether an individual make a transition from one state to another, we just count at each generation how many are present in each state. However, it remains true that a loss (decrease in number) in one state is correlated to an equal gain (increase in number) in another state by the corresponding parameter of the model.

4.3.1 Graph Representation. One way to employ the parameters is to construct a graph to visualize the changes in each state and their relationships. In such a graph, each state S_i is represented with a node, and each parameter is represented with a link that shows the relationship between two states according to the model equations. We use solid and dashed links with the following notation

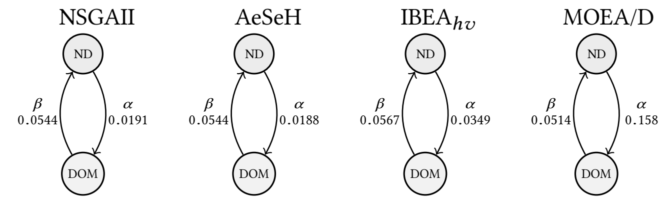

Figure 9: Two states, 3 objectives, population 200.

(parameter α is used as example)

$$S_1 \overset{\alpha>0}{\rightarrow} S_2, \text{ implies} \begin{cases} S_2 \text{ gains } |\alpha S_1| \\ S_1 \text{ looses } |\alpha S_1| \end{cases}$$

$$S_1 \overset{\alpha<0}{\dashrightarrow} S_2, \text{ implies} \begin{cases} S_1 \text{ gains } |\alpha S_1| \\ S_2 \text{ looses } |\alpha S_1| \end{cases}$$

Figures 9 and 10 show examples of a graph representation for two- and three-state models according to the above notation, corresponding to the parameters learned for the configuration with 3 objectives and a population of size 200. Figure 11 shows a similar example of three-state models learned for 5 objectives and population of size 200.

From Figure 9, we can remark that all parameters are positive. Here, the quotient β/α between the states' rates of change are 2.85, 2.89,1.62, and 0.33 for NSGA-II, AϵSϵH, IBEA$_{hv}$ and MOEA/D, respectively. This suggests that the number of ND solutions (and DOM) vary and accumulate similarly in NSGA-II and AϵSϵH, whereas in IBEA$_{hv}$ the accumulation is smaller and in MOEA/D is the smallest, as can be seen in Figure 7.

From Figure 10 note that all parameters of NSGA-II, AϵSϵH, IBEA$_{hv}$ have the same signs, while MOEA/D differs in the signs of γ and β. Focusing on the PO solutions state, we can see that in these three algorithms PO gains come from three sources, i.e βPO, αPO and $\bar{\alpha}$NDNP, and loses $\bar{\beta}$DOM. In MOEA/D the difference is that it loses βPO instead of gaining them. This suggests that MOEA/D accumulates less PO solutions compared to the other algorithms, as can be seen in Figure 8.

When we increase the number of objectives, the dynamics of the algorithms change. For example, from Figure 11 note that $\bar{\alpha}$NDNP is the only source of gain for PO in NSGA-II, AϵSϵH and MOEA/D, whereas IBEA$_{hv}$ also gains αPO. Another difference is that MOEA/D loses γNDNP in favor of DOM, whereas the other algorithms gain the same amount in favor of NDNP. Looking at the relationship between DOM and NDNP, this means that in the first three algorithms NDNP only have gains with respect to DOM, whereas in MOEA/D NDNP loses in favor of DOM and vice versa.

4.3.2 Parameters vs Number of Objectives. Another way to exploit the model parameters is to use them to analyze the scalability of algorithms. As an example, Figure 12 plots the six parameters of our three-state model for the 5 algorithms set with population of size 200, varying the number of objectives from 3 to 6. As a visual aid, on top of each plot we include the states related by a given parameter following the notation described before.

Looking at parameters α, β, γ and $\bar{\beta}$, we see that, for some algorithms, the sign of the parameter changes when the number of objectives increases. For example, NSGA-II and AϵSϵH have a negative α for 3 objectives, and MOEA/D for 3 and 4 objectives. This means that with fewer objectives these algorithms gain PO solutions in proportion to αPO and NDNP loses an equal number of solutions. In other words, the size of PO correlates to an increase of solutions in PO at the expense of NDNP. In 5 objectives α is positive for all the algorithms, except $IBEA_{hv}$. That is, the size of NDNP increases at the expense of PO.

Looking at parameter β, it is interesting to notice that, in the case of MOEA/D, β remains positive up to six objectives and therefore PO always loses in favor of DOM. However, the value of β reduces with the number of objectives, dropping approximately by 0.09 from 3 to 4, 0.05 from 4 to 5, and 0.0258 from 5 to 6. In the other three algorithms, PO increases with PO in three objectives. However, β is very close to 0 for four or more objectives. So, very few solutions are lost in PO in favor of DOM. There are other interesting observations about the parameters and algorithms. However, we do not include them due to space restriction.

5 CONCLUSION

We proposed simple closed state models that capture the population dynamics of evolutionary algorithms. As an example we showed a two-state model that uses *Pareto dominance* as a rule to split the population into non-dominated and dominated solutions, and a three-state model that extends it by adding *membership to the POS* as another rule, splitting the non-dominated group into Pareto optimal, non-dominated non-PO solutions. Using this models, we tested the methodology to find the parameters on data obtained from representative MOEAs ran on MNK-landscapes as a test problem. The results of the parameter fitting, verification of the model against the data and cross-validation suggest that our models are able to follow the trend of the data, and provide a good estimation for future states when feeding real data of a previous state.

We also proposed a graphical representation of the model and its parameters, that allows us to see how parameters determine the gain/loss relationship between states and its ratio.

Finally, we showed how parameters can be used to analyze the scalability of algorithms, varying the number of objectives from 3 to 6. We observed that the increase in the number of objectives in some algorithms affects the signs and magnitude of our model parameters, which in turn affect the relationship between the states related by the parameters. This type of analysis shows the power and usefulness of the models to encapsulate the underlying complexity of population dynamics.

In a future work, we would like to test the models on the unused features mentioned here, on new features not related to the POS, and also on features that are related to other performance measures. It would also be of interest to experiment with the models proposed here on other relevant multi- and many-objective optimization problems.

REFERENCES

[1] Hernán Aguirre, Arnaud Liefooghe, Sébastien Verel, and Kiyoshi Tanaka. 2014. An Analysis on Selection for High-Resolution Approximations in Many-Objective Optimization. In *Parallel Problem Solving from Nature – PPSN XIII: 13th International Conference, Ljubljana, Slovenia, September 13-17, 2014. Proceedings.* 487–497.

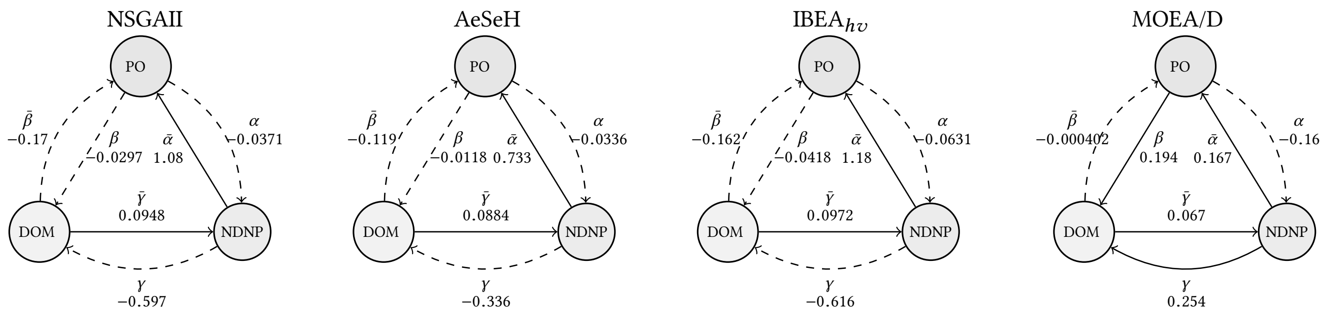

Figure 10: Three states, 3 objectives, population 200.

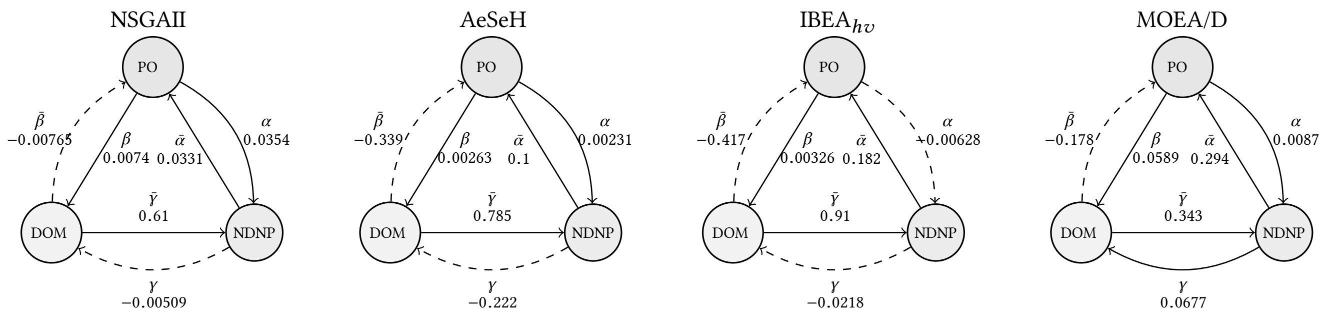

Figure 11: Three states, 5 objectives, population 200.

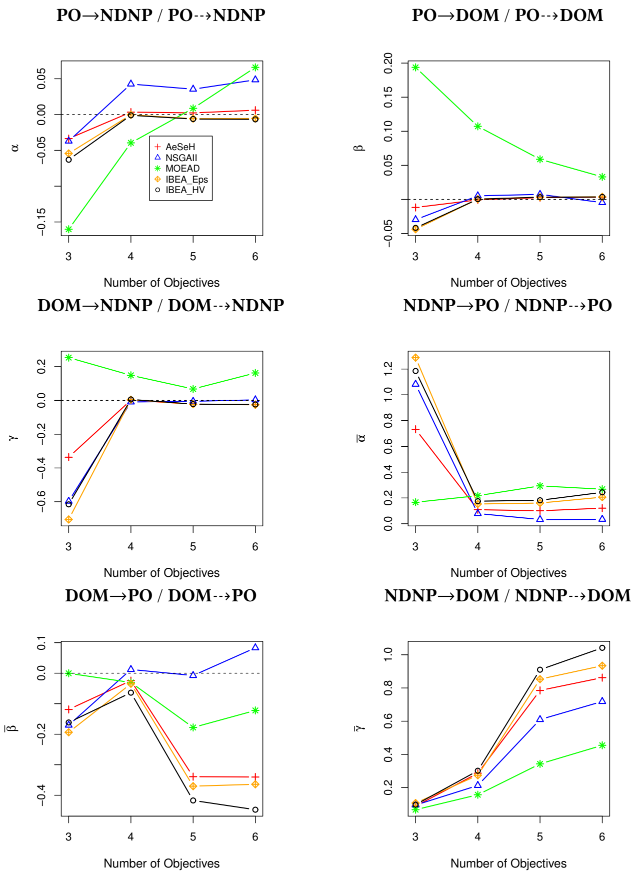

Figure 12: Model parameters variation when the number of objectives increases on a fixed population of size 200.

[2] Hernán Aguirre, Akira Oyama, and Kiyoshi Tanaka. 2013. Adaptive ϵ-Sampling and ϵ-Hood for Evolutionary Many-Objective Optimization. In *Evolutionary Multi-Criterion Optimization: 7th International Conference, EMO 2013, Sheffield, UK, March 19-22, 2013. Proceedings.* 322–336.

[3] Hernán Aguirre and Kiyoshi Tanaka. 2004. Insights on properties of multiobjective MNK-landscapes. In *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753)*, Vol. 1. 196–203 Vol.1.

[4] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. 2000. A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In *Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, September 18–20, 2000 Proceedings.* 849–858.

[5] Nikolaus Hansen and Andreas Ostermeier. 2001. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evol. Comput.* 9, 2 (June 2001), 159–195.

[6] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. 2008. Evolutionary many-objective optimization: A short review. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence).* 2419–2426.

[7] Günter Rudolph". 1994. Convergence analysis of canonical genetic algorithm. *IEEE Transactions on Neural Networks* 5, 1 (1994), 96–101.

[8] Christian von Lücken, Benjamín Barán, and Carlos Brizuela. 2014. A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications* 58, 3 (2014), 707–756.

[9] Michael D. Vose. 1995. Modeling Simple Genetic Algorithms. *Evol. Comput.* 3, 4 (Dec. 1995), 453–472.

[10] Qingfu Zhang and Hui Li. 2007. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation* 11, 6 (Dec 2007), 712–731.

[11] Eckart Zitzler and Simon Künzli. 2004. Indicator-Based Selection in Multiobjective Search. In *Parallel Problem Solving from Nature - PPSN VIII: 8th International Conference, Birmingham, UK, September 18-22, 2004. Proceedings.* 832–842.