Clonal Selection Algorithms for Optimal Product Line Design
A Comparative Study

Pantourakis, Michail; Tsafarakis, Stelios; Zervoudakis, Konstantinos; Altsitsiadis, Efthymios ; Andronikidis, Andreas; Ntamadaki, Vasiliki

Document Version
Accepted author manuscript

Published in:
European Journal of Operational Research

DOI:
10.1016/j.ejor.2021.07.006

Publication date:
2022

License
CC BY-NC-ND

Citation for published version (APA):
Pantourakis, M., Tsafarakis, S., Zervoudakis, K., Altsitsiadis, E., Andronikidis, A., & Ntamadaki, V. (2022). Clonal Selection Algorithms for Optimal Product Line Design: A Comparative Study. *European Journal of Operational Research*, *298*(2), 585-595. https://doi.org/10.1016/j.ejor.2021.07.006

Clonal Selection Algorithms for Optimal Product Line Design: A Comparative Study

Michail Pantourakis[1], MSc
e-mail: pantourakis.michail@gmail.com

Stelios Tsafarakis[1], PhD (corresponding author)
e-mail: tsafarakis@dpem.tuc.gr
tel: +302821037160
ORCID: 0000-0001-5535-4787

Konstantinos Zervoudakis[1], MSc
e-mail: kzervoudakis@isc.tuc.gr
ORCID: 0000-0002-2146-1493

Efthymios Altsitsiadis, PhD
Department of Management, Society and Communication, Copenhagen Business School, Copenhagen, Denmark
e-mail: ea.msc@cbs.dk
ORCID: 0000-0003-4208-927X

Andreas Andronikidis, PhD
Department of Business Administration, University of Macedonia, Thessaloniki, Greece
e-mail: a.andronikidis@uom.gr
ORCID: 0000-0003-4855-1111

Vasiliki Ntamadaki[1], MSc
e-mail: vasodmk@gmail.com

[1]*School of Production Engineering and Management, Technical University of Crete, Kounoupidiana, 731 00, Chania, Greece*

Clonal Selection Algorithms for Optimal Product Line Design: A Comparative Study

Product design constitutes a critical process for a firm to stay competitive. Whilst the biologically inspired Clonal Selection Algorithms (CSA) have been applied to efficiently solve several combinatorial optimization problems, they have not yet been tested for optimal product lines. By adopting a previous comparative analysis with real and simulated conjoint data, we adapt and compare in this context 23 CSA variants. Our comparison demonstrates the efficiency of specific cloning, selection and somatic hypermutation operators against other optimization algorithms, such as Simulated Annealing and Genetic Algorithm. To further investigate the robustness of each method to combinatorial size, we extend the previous paradigm to larger product lines and different optimization objectives. The consequent performance variation elucidates how each operator shifts the search focus of CSAs. Collectively, our study demonstrates the importance of a fine balance between global and local search in such combinatorial problems, and the ability of CSAs to achieve it.

Keywords: OR in marketing, clonal selection algorithm, combinatorial optimization, product line design

1. Introduction

Product design constitutes one of the most important phases of a comprehensive product development process (Hauser et al., 2006). The Product (Line) Design (PLD) problem was introduced by Zufryden (1977), and 40 years later is still a priority in management science and operations research. In PLD, a product (e.g. a laptop computer) is represented by a set of attributes (e.g. screen size, processor speed, memory size), each taking a finite number of levels (e.g. the screen size can be 13'', 15'', 17'' etc.). Consumer preferences for each attribute level (i.e., *partworths*) are usually measured through Conjoint Analysis (CA). Assuming additive utility functions, utilities for different product configurations are derived from summing up the partworths. These utilities are then transformed into hypothetical choice shares for each product with the application of a choice model, such as the first choice rule or the MNL. The goal of the optimal PLD problem is the design of a number (line) of products (selection of the attribute levels for each product in the line) that will optimize an objective of the company, usually profit, market share, or customer welfare (Green & Krieger, 1985; Tsafarakis & Matsatsinis, 2010).

The problem is usually formulated as an integer program where we seek the levels of product attributes that optimize the company's objective given consumer partworths and competitive product configurations. Kohli and Krishnamurti (1989) proved that even the design of a single product belongs to the class of NP-hard problems, and hence, different heuristic mechanisms have been proposed in an attempt to provide near optimal solutions in tractable time. Belloni et al., (2008) compared the performance of nine heuristics using data from a conjoint study previously conducted for a real product line design problem, as well as simulated data of various problem sizes. They

benchmarked the performance of the nine heuristics against the global optimum found by an implementation of Lagrangian relaxation with branch and bound. Simulated Annealing (SA), and Genetic Algorithms (GA) were the two heuristics that exhibited the best performance.

In this paper, we apply Clonal Selection Algorithms (CSA) for solving the optimal PLD problem. CSA have been applied to a wide range of combinatorial optimization problems, showing very promising results. We investigate which CSA variants perform better in the specific optimization problem, and which are the underling mechanisms that drive the algorithms' performance. In particular, we benchmark the performance of 23 CSA variants against SA and GA, using the datasets from Belloni et al., (2008), which we also extend to larger problem sizes and different optimization objectives.

The rest of the paper is organized into five sections as follows: Section 2 provides a brief description of the optimal PLD problem, while in Section 3 and Section 4 the conceptual framework of the proposed approach as well as the problem formulation are descripted. In Section 5 the effectiveness of the CSA variants is evaluated through a comparison of their performance with that of GA and SA. Finally, Section 6 provides an overview of the main conclusions of the study and future research areas are suggested.

2. Previous work in PLD

The optimal product design problem was originally formulated by Zufryden (1977), and a few years later, Green and Krieger (1985) conceptualized the optimal PLD problem. Several optimization criteria have been utilized, the most important being the share of choices (market share maximization), the seller's welfare (profit maximization), and the buyer's welfare (customer utility maximization). Kohli and Krishnamurti (1989) proved that the share of choices for the single product design problem is NP-hard. Since global optimality cannot be guaranteed in polynomial time, various heuristic approaches have been applied to solve the problem, including Integer programming (Zufryden, 1977), Interchange (or product-swapping) heuristic (Green & Krieger, 1985), Greedy Heuristics (Green & Krieger, 1985), Genetic algorithms (Alexouda & Paparrizos, 2001; Balakrishnan et al., 1995, 1996, 2004, 2006; Steiner & Hruschka, 2003), Ant Colony Optimization (Albritton & McMullen 2007), Particle Swarm Optimization (Saridakis et al., 2015), Simulated Annealing (Tsafarakis, 2016)**,** and Differential Evolution (Tsafarakis et al., 2020). Lagrangian relaxation with branch and bound has also been employed to prove global optimality for the seller's welfare in the PLD (Belloni et al., 2008), while recently Bertsimas and Mišić (2017) adopted a robust optimization approach to account for parameter and structural uncertainty in the choice model.

Although CSAs are reported to perform better compared with other heuristics (i.e., GAs, Neural Networks) in problems like function optimization and pattern recognition (Ulutas & Konak, 2011), they have never been applied to the PLD literature and the broader area of marketing science. The next section introduces the basic concepts of Clonal Selection theory in immunology from which CSAs were inspired, and explains how the algorithm is adapted to solve the PLD problem.

3. Conceptual Framework

Clonal Selection Theory (Burnet et al., 1959) with its subsequent improvements has become a widely accepted model (Cohn et al., 2007), describing the main characteristics behind a basic, antigen-specific, adaptive response. The human immune system is a complex, highly adaptive, defense system of interacting organs, cells and chemical species. The ability to protect the body from pathogens that are recognized as non-self molecules (called immunity) is determined by two components: the innate and the adaptive immune system. The latter largely consists of B and T lymphocytes, which execute the recognition of non-self species through the specific, *affinity* binding of their membrane receptors to associated foreign substances, called *antigens*. Specific *antibodies* to these antigens are then produced by B-cells to initiate the clearance response (Goldsby et al., 2002). Therefore, the adaptive system is responsible for specifically-targeted immune responses, and its functional features received most of the attention of both immunological (Germain, 2004; Timmis et al., 2008), and Artificial Immune System (AIS) research (Timmis et al., 2008, Hart & Timmis, 2008, Dasgupta et al., 2011).

In the context of combinatorial optimization, a candidate solution to the problem is represented in CSAs by a vector called *Antibody* (*Chromosome* is the analogous in GAs). Each Antibody is evaluated on the problem's objective function (the so-called *Antigen*) taking a value called *Affinity*. Regarding the operators of the algorithm, the following three physiological processes are of particular importance (De Castro & Von Zuben, 2002):

1. A **cloning** process (reproduction and proliferation) of stimulated B-cells.
2. **Cellular** diversity due to an affinity maturation process, a combination of both point and larger accelerated genetic mutations (called **somatic hypermutation**). Particularly, somatic hypermutation involves random genetic changes to the genes of the cloned cells. Cell mutation derives from the Genetic Algorithm and can be considered a form of local search. Moreover, it helps the algorithm to explore new areas of the search space that may not be visited otherwise. For instance, suppose that the solution is represented as a nine-length binary vector (Antibody) e.g. [1 **0** 1 0 **0** 1 0 0 1], with the second and the fifth value (bold values) to be mutated. The solution vector is then altered as [1 **1** 1 0 **1** 1 0 0 1].
3. **Selection** of **high**-affinity clones and elimination of low-affinity clones. Like GAs, CSA are

classified as computational systems inspired by genetics mechanisms since it can be viewed as a type of Darwinian microcosm where the fittest cells (best match with antigens) are selected for survival.

Having introduced the terms and concepts in the setting of a physiological immune system, we can build the analogy among CSA, GA, optimization problems, and the PLD (Table 1).

Table 1. Corresponding terms among CSA, GA, optimization problems and product line design.

CSA	Genetic Algorithm	Optimization Problem	Product Line Design
Antibody	Chromosome	Solution	Product line
Affinity	Fitness	Value of objective function	Profit
Antigen	Fitness function	Objective function	Function for profit
Cloning antibodies	Selection process	Copying solutions	Copying product lines
Hypermutation	Mutation	Changes in a solution	Changes in a product line
Clonal selection	Replacement (migration) strategy	Selecting the best solutions	Selecting the most profitable product lines

4. Problem Formulation and Methods

In this section the basic CSA that was adapted is examined to tackle the optimal PLD problem. In the literature, differences between CSA variants are mainly found during three steps, which correspond to the aforementioned processes of cloning, hypermutation and selection. In total, 23 CSA variants were tested, which were derived from a combination of 13 different operators. Therefore, the unique characteristics of each operator are briefly explained. For better readability, each operator is associated with specific prefixes or suffixes, which are used in combination with previously proposed names for our variants throughout our analysis. The operators that each CSA variant employs, as well as their exact steps, are outlined in the Appendix A.1.

4.1 Adaptation of Clonal Selection Algorithm

Following the work of Belloni et al., (2008), we also begin our comparison using a real-life example. In particular, we use a real data set from the paper of Toubia et al., (2003), where a field experiment was conducted to measure consumer preferences for a new laptop bag product to be launched by Timbuk2 Designs Inc. (San Francisco, CA, USA). The authors collected pairwise comparison data from 330 respondents, and used these to estimate respondent-level partworths for nine binary product attributes, as well as price, which was restricted to two levels ($70 and $100). Belloni et al., (2008) considered seven price levels (they interpolated to derive partworths for the five intermediate levels) and used these partworths along with estimates of each feature's cost (a base cost of $35 was also assumed for each bag), for designing a line of five laptop bags with the objective of profit

maximization. To derive status quo utilities for each consumer, the authors included a selection of three competing products. These competing bags were arbitrarily designed to include a product with all nine of the optional attributes priced at $100, a product with five of the attributes priced at $85, and a product with no optional attributes priced at $70. A first-choice rule is employed, where the customer deterministically purchases the product that offers her/him the highest utility. It is assumed that if the competing products provide the same utility as one of the five Timbuk2 products then the customer purchases the competing product.

Next, we provide a complete step-by-step CSA for solving the optimal PLD problem, based on the Belloni et al., (2008) formulation. A complete list of the notations used here are provided in the Appendix A.2. Assume that a manufacturer focuses on n_{att} product attributes. Then the total number of product combinations n_{comb} is known and fixed. Furthermore, we define $s_{prod}^{j} \in \{0, 1, \dots, n_{lev}^{j} - 1\}^{1 \times n_{att}}$ as the jth row of a matrix $\boldsymbol{S}_{prod} \in \{0, 1, \dots, n_{lev}^{j} - 1\}^{n_{comb} \times n_{att}}$, which includes all possible products, while n_{lev}^{k} denotes the number of possible levels for each attribute k. Part-worth matrix $\boldsymbol{W} \in \mathrm{R}^{n_{cust} \times n_{att}}$ of all n_{cust} customers and all attributes is provided by CA. Moreover, the company assumes an estimated cost per attribute $d_{att} \in \mathrm{R}^{1 \times n_{att}}$. As a result, the perceived utility u_{ij} per customer $\boldsymbol{i}$ and the total profit p_j of selling a unit of each product $\boldsymbol{j}$ at a certain price can be precomputed and stored. We further assume n_{comp} competing bags that are already in sale and calculate the status quo utility $u_{i,base}$ per customer i. Since a first choice (maximum utility) model is used, each customer chooses a product of the manufacturer's new line only if its utility is higher than the utility of the most preferred competing bag. By denoting $\boldsymbol{s}_{comp} \in \{1, \dots, n_{comb}\}^{1 \times n_{comp}}$ as a vector that contains the competing products, our algorithm consists of the following steps:

1. Precompute p_j for each possible product $j \in \{1, 2, \dots, n_{comb}\}$ based on the cost of the selected attributes d_{att} and the price level that is sold
2. Precompute $u_{ij} = \boldsymbol{s}_{prod}^{j} \boldsymbol{w}_i^{\mathrm{T}}$ for each customer $i \in \{1, 2, \dots, n_{cust}\}$ and product j.
3. Find $u_{i,base} = max_{j':j' \in \boldsymbol{s}_{comp}} U_{ij'}$
4. Initialize optimization algorithm, by generating a random population of $\boldsymbol{P}_1 = U\{[1, 2, \dots, n_{comb}]\}^{N_{P1} \times n_{prod}}$, where U denotes a Uniform distribution, $N_{\boldsymbol{P1}}$ denotes the amount of antibodies in the initial population, n_{prod} represents the number of products in the new line. As a result, each antibody (solution vector) is a n_{prod}-length vector which consists of values corresponding to one out of all possible products n_{comb}, like the solution representation that Belloni et al., (2008)

used in their research.

5. Calculate for each antibody $\boldsymbol{a}\in\boldsymbol{P}_1$ a binary variable $x_{ij}^{a}=\begin{cases}0, \text{if } u_{ij}\le u_{i,base}\\ 1, \text{otherwise}\end{cases}$, which represents whether a customer i chooses product j.
6. Find profit $\forall \boldsymbol{a}\in \boldsymbol{P}_1$, $p_a=\sum_{j\in a}\sum_{i\in I} p_j x_{ij}^{a}$.
7. **Cloning** operator $\forall \boldsymbol{a}\in \boldsymbol{P}_g$ (g: generation counter), creating set of clones $\boldsymbol{C}$.
8. **Hypermutation** operator for each clone $\boldsymbol{c}\in\boldsymbol{C}$, $\boldsymbol{c}_z \leftarrow U\{[0,1,\dots,n_{lev}^{k}-1]\backslash \boldsymbol{c}_z\}$. For example, for clones consisted of 2 products with 3 attributes and 2 levels per attribute, position z=5 means that a mutation would occur at the second attribute of the second product, and its value would be simply inverted (since we force the attribute to change its prior value).
9. Calculate $\forall \boldsymbol{c}\in\boldsymbol{C}$, $x_{ij}^{c}=\begin{cases}0, \text{if } u_{ij}\le u_{i,base}\\ 1, \text{otherwise}\end{cases}$
10. Find profit $\forall \boldsymbol{c}\in\boldsymbol{C}$, $p_c=\sum_{j\in c}\sum_{i\in I} p_j x_{ij}^{c}$.
11. (Optional step) Generate a new random population $\boldsymbol{P}_{new}$
12. Calculate for each antibody $\boldsymbol{a}\in\boldsymbol{P}_{new}$ a binary variable $x_{ij}^{a}=\begin{cases}0, \text{if } u_{ij}\le u_{i,base}\\ 1, \text{otherwise}\end{cases}$, which represents whether a customer i chooses product j.
13. Find profit $\forall \boldsymbol{a}\in \boldsymbol{P}_{new}$, $p_a=\sum_{j\in a}\sum_{i\in I} p_j x_{ij}^{a}$.
14. **Selection** operator for next generation $\boldsymbol{P}_{g+1}$, choosing a number of the most profitable clones, old, and new random antibodies.
15. Repeat steps 7-14 for g_{max} generations or for $g_{imp} < g_{max}$ generations without improvement of the most profitable solution found.

As can be seen, steps 7, 8 and 14 contain the processes of cloning, hypermutation and selection, and differences between the tested CSA variants depend on which operators are used respectively.

4.2 Cloning Operators

Static cloning

This is the default form of cloning, as the amount of clones per antibody (per generation) is a constant, with the total amount of clones given by $N_C=\sum_{i=1}^{N} round(\beta * N)$ (De Castro & Von Zuben, 2002; Cutello et al., 2004, 2005), where β is a multiplication factor, and $N = N_{\boldsymbol{P}_g}\forall g$ (namely population size remains constant with time). A pseudocode of static cloning operator is presented in Appendix A.1.

Proportional cloning (prop-)

As the name implies, the amount of clones per antibody is proportional to the *relative fitness* $f_i \in [0,1]$. A form of a step function was used by De Castro and Von Zuben (2002) to calculate N_C. However, we found it inefficient during the initial stages of our work. Instead, we employed an exponential function $N_c = \sum_{i=1}^{N} ceil(e^{\rho f_i} N)$, where ρ is an exponent factor, modifying the inverse exponential function for the corresponding hypermutation operator (see Section 3.3). Finally, we adopted and properly modified the fitness functions used by Engin and Döyen (2004): $f_i = F_i / \sum_{i=1}^{N} F_i$, where $F_i = p_i + 1 - \min_i p_i$ denotes (absolute) fitness of antibody i. A pseudocode of proportional cloning (prop-) operator is presented in Appendix A.1.

4.3 Hypermutation Operators

Static hypermutation (stat-)

The amount of positions (features) to be mutated for each clone m_c is predetermined and constant (Cutello et al., 2004).

Inversely proportional hypermutation

The default hypermutation operator, it mimics the natural affinity maturation process, as $m_c = ceil(e^{-\rho \cdot f_i} \cdot n_{prod} \cdot n_{att})$ is now determined by the current fitness value (see Section 3.2) of the parental antibody (De Castro & Von Zuben, 2002; Engin & Döyen, 2004; Cutello et al., 2005). Notice that in our variants this operator is mutually exclusive with proportional cloning to avoid large computational time per generation.

First constructive mutation (-fcm)

Although the aforementioned hypermutation operators assume that exactly m_c mutations happen, a strategy called "stop at First Constructive Mutation" (FCM) (Cutello et al., 2004, 2006) allows them to stop mutating as soon as a clone with higher affinity than the parent has been produced. Hence, FCM may prevent detrimental mutations on already improved clones and delay premature convergence (Cutello et al., 2004).

Contiguous hypermutation (BCA-)

Based on the B-Cell Algorithm (BCA) (Kelsey & Timmis, 2003), this operator performs mutations on a randomly chosen contiguous part of a clone.

Pseudocodes of hypermutation operators mentioned above are presented in Appendix A.1.

4.4 Selection Operators

As can be seen, our variants differ the most in their clonal selection operators. Nevertheless, all of them are based on two selection strategies presented for CLONALG by Cutello et al., (2004), thus naming their variants CLONALG1 and CLONALG2. Pseudocodes of selection

operators mentioned below are presented in Appendix A.1.

Individual Selection Strategy (-CLONALG1-, -OPTIA1-, BCA1)

The main feature of this strategy is that it takes into account the origin of each clone. Namely, each antibody can be substituted only by the best individual of its own clones, even if it is better than that clone. Note that in case of inserting new random antibodies in each generation, these antibodies substitute the worst performing selected clones. Therefore, population size *N* remains constant.

Lumped Selection Strategy (-CLONALG2-, -OPTIA2-, BCA2)

In contrast to Individual selection strategy, this operator selects the best individuals out of all clones, regardless of their origin. Furthermore, it can retain a subset of the best parental antibodies, while population size *N* remains constant in a similar way as before.

optAInet

A combination of CLONALG and immune networks with a dynamic population size, optAInet was originally developed to face multimodal optimization problems (De Castro & Timmis, 2002). In our adaptation, optAInet is identical to statCLONALG1 up to the selection of the best subset of clones. Then, relative profit $\widehat{p_i} = \frac{p_i}{max_i p_i}$ is calculated for each antibody. When the average relative profit of the new population is different enough, the algorithm continues as statCLONALG1. Otherwise, minimal changes indicate that optAInet has reached areas of local optima. To increase diversity, current antibodies (namely product lines) are compared pairwise, and if at least 70% of their products are the same, the worst of the two antibodies is removed. Subsequently, new random antibodies enter the population. As a consequence, population size may vary from generation to generation.

Aging (-OPTIA-)

Various aging operators consist another proposed way of escaping local optima during selection, with OPTIA algorithm being the best known example (Cutello et al., 2004, 2005). The present study adopted a static pure aging mechanism, because it has showed better results compared to stochastic aging (Cutello et al., 2005). Having defined a maximum age limit, t_{lim}, this operator removes any antibodies that have remained in the population for more than t_{lim} generations. For our study, we combine aging with both selection strategies.

Elitism

Adding this operator during the selection process results in keeping a subset of the best antibodies to the next generation, regardless of how they fair against their mutated clones. In the context of genetic algorithms, *strong* elitism (Cutello et al., 2004) seems to lead in premature convergence (Ahn & Ramakrishna, 2003). Therefore, we chose to utilize strong elitism only once (in

CLONALG1elit), whereas *weak* elitism (preserving only the best antibody of each generation) is exploited in most of our variants.

A dendrogram depicting which cloning, hypermutation and selection algorithms compose each CSA variant is presented in Figure A.1 of Appendix A.1.

4.5 Parameter Configuration

Efficient parameter values were manually determined by numerous trial-error steps. Succinctly, the Genetic Algorithm (GA) and Simulated Annealing (SA) of Belloni et al., (2008) served as benchmarks. To make GA and SA directly comparable, we first adjusted their parameters so that they run at a similar CPU time. Since all our CSAs (and SA) include a variable tracking the best achieved solution out of all generations, we also added it to GA to remove possible confounding effects.

For CSA parameters, we began trials with values reported from previous implementations (De Castro & Von Zuben, 2002; Cutello et al., 2005). Next, we fine-tuned our methods according to the execution time of SA, reducing their mean computational time to SA levels while preserving the quality of their solutions as much as possible. Complete tables of parameter values, and observations for their role in CSA performance can be found in the Appendix A.2. Finally, all algorithms and data analysis were implemented using MATLAB 2016a for Windows 10 64-bit in an Intel Core i-7 laptop with 8 GB RAM.

5. Results

In our analysis we utilize and extend the comparison paradigm of Belloni et al., (2008). Hence, we also investigated differences between the CSA methods using both real conjoint and simulated data. The performance of all CSA methods is compared to that of GA and SA. Two different implementations for GA and SA were used when performing on the real and the simulated data set, respectively, as Belloni et al., (2008) used in their research, to maximize the performance of both GA and SA.

5.1 Comparison on Real Conjoint Data - Five Products

To solve the problem of laptop bags, we likewise assumed a line of five bags ($n_{prod} = 5$), with $n_{att} = 10$, one attribute denoting the bag's price with seven possible levels \$70, \$75, \$80, \$85, \$90, \$95 or \$100, and nine binary attributes (existence or not of a particular feature), resulting in a solution space of 3,584 different products. Belloni et al., 2008 found the globally maximal earnings of \$12,226 after one-week computational time by applying lagrangian relaxation with branch and bound to the same real conjoint dataset.

Table 2 presents averaged results for all CSAs, including SA and GA for comparison (further results can be found in the Appendix A.3). To control the variability of CSAs, each method was repeated 100 times and for $g_{max} = 1000$ in each run. We also recollected results from 100 trials of SA and GA. For a direct comparison, we configured all algorithms to run (on average) for approximately the same CPU time per trial; ±2 s from that of GA.

Table 2 Comparison of all CSA methods on the real dataset assuming five bags per product line. Results are averaged over 100 trials.

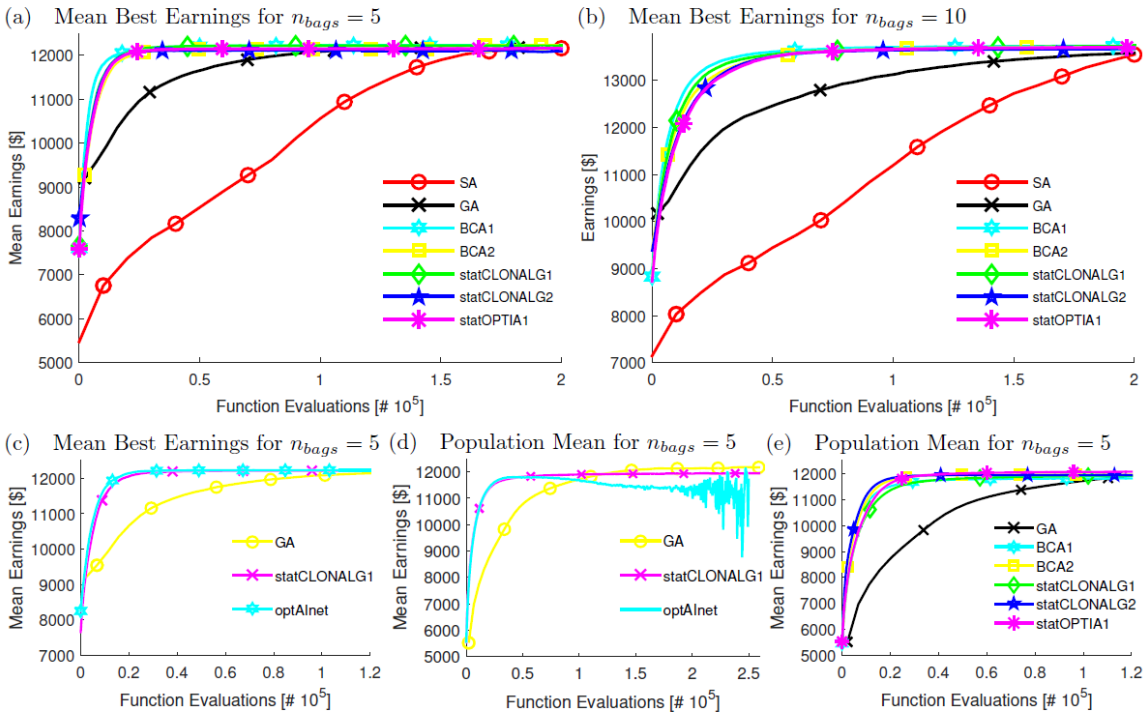
Methods	**Mean Best Earnings** ($)	**Best Solution Frequency** (%)	**Standard Deviation** ($)	**Difference from SA** (p-value %)	**Difference from GA** (p-value %)	**Mean # of Evaluations per Trial**	**Mean CPU Time per Trial** (s)
optAInet	12225.98	99	0.25	<0.01	32.57	188730	17.98
BCA1	12225.33	90	4.00	<0.01	55.71	172580	17.88
GA	12224.30	99	17.05	<0.01	100.00	202725	17.36
statCLONALG1fcm	12219.90	63	24.51	<0.01	14.22	211499	18.21
statCLONALG1	12218.94	95	33.62	<0.01	15.66	198090	17.09
propCLONALG1	12218.54	92	33.64	<0.01	12.86	212358	18.49
CLONALG1	12215.28	87	37.92	<0.01	3.13	150470	17.15
CLONALG1fcm	12207.14	72	51.56	<0.01	0.18	193857	18.07
statOPTIA1fcm	12172.44	67	78.95	33.09	<0.01	208798	17.71
statCLONALG2fcm	12170.58	53	79.68	41.93	<0.01	207079	18.26
propOPTIA1	12165.00	54	82.65	74.81	<0.01	191825	16.48
SA	12161.21	49	83.99	100.00	<0.01	145001	16.36
statOPTIA2fcm	12156.30	58	84.02	67.98	<0.01	201320	17.89
OPTIA1fcm	12152.32	54	85.72	46.00	<0.01	224920	19.22
statOPTIA1	12149.54	46	86.19	33.36	<0.01	209396	18.02
CLONALG1elit	12147.44	45	86.28	25.42	<0.01	184676	19.01
OPTIA1	12142.71	50	87.75	12.94	<0.01	167668	18.89
BCA2	12133.44	44	84.94	2.11	<0.01	148501	15.76
OPTIA2fcm	12128.52	42	84.97	0.68	<0.01	172256	17.04
CLONALG2fcm	12120.71	33	84.00	0.08	<0.01	179490	16.49
OPTIA2	12114.34	33	82.95	0.01	<0.01	166331	18.00
CLONALG2	12111.52	32	80.26	<0.01	<0.01	137480	15.57
statCLONALG2	12108.67	29	79.50	<0.01	<0.01	192604	17.42
statOPTIA2	12108.61	30	81.61	<0.01	<0.01	163550	15.61
propCLONALG2	12099.05	22	76.81	<0.01	<0.01	204024	17.87

Table 2 includes the average earnings and standard deviation (of the best reached in each trial), finding best solution frequency, average number of evaluations per trial, average execution time per trial, and p-values (two-sample t-test) for determining statistically significant differences from SA and GA. Judging by average earnings, several CSA variants exhibit significantly (p-value<0.05) better solutions than SA. On the other hand, only optAInet and BCA1 slightly surpass GA, whereas statCLONALG1fcm, statCLONALG1 and propCLONALG1 are slightly behind without statistically significant differences. Nevertheless, Figure 1a depicts a clear advantage of

CSAs: while GA needed on average approximately 100000 function evaluations to reach earnings above $12000, CSAs reached that level with just 10000 to 20000 evaluations.

CSAs based on Individual selection strategy showcased a better performance than Lumped selection strategy. For instance, the best performing method (on average earnings terms) that uses the second operator (statCLONALG2fcm) was still surpassed by eight other CSAs. This finding shows the advantage of independent evolution of each antibody, leading to more efficient solution space search. Furthermore, the static hypermutation and proportional cloning operators, in combination with CLONALG1 and OPTIA1, illustrated slightly better results. In contrast, BCA1 and inversely proportional hypermutation methods terminated using considerably less evaluations than their counterparts.

Figure 1 **Evolution of earnings averaged over the number of profit evaluations. Figures (a-c) depict the average earnings of the best solution, whereas figures (d-e) depict the trimmed average population earnings.**



Note. We chose a two-sided (5% each) trimmed population average to reduce the effect of outliers, especially of new, extremely low earning, random antibodies.

Figure 1c compares the evolution of solutions between optAInet and statCLONALG1. Barely faster improvement by optAInet highlights the fact that statCLONALG1 constitutes a large part of optAInet. The network interactions of optAInet mainly emerged as soon as the method hit solutions close to or at the global optimum, appearing in Figure 1d with a highly fluctuating population mean due to insertion of random antibodies.

Interestingly, although FCM demonstrated a positive effect on mean best earnings when accompanying CLONALG2- and OPTIA-based methods, its contribution to CLONALG1-based methods was mostly negative. Similar effects of FCM on their frequency of best solutions also establishes this finding. Despite these differences, FCM required a higher average number of evaluations in all cases.

5.2 Testing robustness to error in partworth estimation

To ensure the robustness of our methods with respect to errors in customer preference measurement, we also assessed their performance in the presence of errors in partworth utilities. Following Belloni et al., (2008), we repeated our analysis after perturbing the original part-worth estimates by adding (simulated) error to the original partworths:

$$w'_{ij} = w_{ij} + \varepsilon_{ij}$$

where w'_{ij} is the perturbed part-worth, $w_{ij}$is the original part-worth for respondent *i* on product feature *j*, and ε_{ij} is a zero-mean, independent (but not identical across customers or attribute levels) normal error term. The standard deviation of the perturbations was obtained by using the standard errors for the respective w_{ij} terms from the OLS estimations. These standard errors averaged approximately 55% of the part-worth absolute values.

Table 3 Comparison of methods on the real dataset under measurement error. The robustness test of Belloni et al., (2008) was utilized, assuming five bags per product line. Results are averaged over 100 trials.

Methods	**Mean Best Earnings** ($)	**Standard Deviation** ($)	**Products in Common** (%)	**Features in Common** (%)	**Difference from SA** (p-value %)	**Difference from GA** (p-value %)	**Mean # of Evaluations per Trial**	**Mean CPU Time per Trial** (s)
optAInet	11809.18	138.29	39.60	85.84	2.27	30.12	195720	18.42
propCLONALG1	11806.65	142.43	39.00	85.98	2.84	34.74	185418	15.95
BCA1	11806.42	137.86	39.60	85.86	2.64	34.24	181464	18.78
CLONALG1	11793.81	148.41	38.80	85.74	8.12	64.52	165122	18.59
statCLONALG1	11792.34	142.35	38.80	85.50	9.08	68.53	198090	17.81
GA	11781.34	196.36	38.00	85.12	19.74	-	204163	17.51
statCLONALG1fcm	11773.97	137.91	39.20	85.54	27.21	77.85	205290	17.62
statOPTIA1fcm	11772.70	140.16	38.00	85.20	31.76	75.61	209521	17.57
statCLONALG2fcm	11756.56	150.03	36.60	84.88	65.37	38.46	210231	17.26
SA	11742. 93	222.79	36.00	84.26	-	19.74	145001	16.21

Table 3 illustrates that CSA methods are quite robust, exhibiting lower standard deviation in best earnings than SA, and more importantly, GA. Nevertheless, the mean best earnings of all methods declined by a similar amount (3-4%). A complete table of the results is presented in Table A.9, in Appendix A.3.

5.3 Comparison On Real Conjoint Data - Ten Products

To further test our algorithms in a larger solution space, we also investigated the performance of CSAs assuming the company wanted to manufacture a line of ten bags ($n_{prod} = 10$). In this

case their average CPU time per trial was ±3 s from that of GA.

Table 4 Comparison of methods on the real dataset assuming ten bags. Representative variants of each operator are shown. Results are averaged over 100 trials.

Methods	Mean Best Earnings ($)	Best Solution Frequency (%)	Standard Deviation ($)	Difference from SA (p-value %)	Difference from GA (p-value %)	Mean # of Evaluations per Trial	Mean CPU Time per Trial (s)
propCLONALG1	13741.04	45	11.56	<0.01	<0.01	365555	35.40
statCLONALG1	13738.55	43	14.66	<0.01	<0.01	347636	35.99
statOPTIA1fcm	13736.66	32	13.42	<0.01	<0.01	365376	36.90
BCA1	13731.78	23	17.39	0.01	<0.01	293388	37.10
CLONALG1	13725.13	17	20.47	2.89	<0.01	231110	35.60
statCLONALG2fcm	13721.40	14	24.86	24.79	<0.01	448659	38.08
optAInet	13718.41	23	28.60	68.41	<0.01	299076	35.24
SA	13716.64	16	32.70	-	<0.01	290001	37.28
GA	13674.10	3	41.60	<0.01	-	347985	36.52
statCLONALG1fcm	13673.01	0	28.46	<0.01	82.98	371767	36.18

This problem was considerably more challenging, as shown in Table 4 by the much lower frequencies of finding the best attained solution of $13753 (only five methods managed to exceed 22%, namely the frequency of the worst performing algorithm for five bags). We can see that propCLONALG1, statCLONALG1, statOPTIA1fcm, BCA1 and CLONALG1 generated significantly more profitable solutions than SA, while GA was surpassed significantly by additional 14 CSAs. Among them, CLONALG1 and BCA1 found these solutions with considerably fewer function (profit) evaluations. Figure 1b illustrates how much sooner CSAs reach near best solution results than GA and SA (around 50000 against 200000 evaluations).

Similar to our previous findings, CSAs following Individual selection strategy generally produced better solutions than Lumped strategy. However, this performance discrepancy was more prominent in the five bag problem, while it was again exaggerated when adding proportional cloning into the mix. This observation probably indicates that Lumped selection strategy variants reach local optima faster and are easier trapped in their neighborhood of solutions. Seeing that the five bag problem may benefit less by local solution improvement (due to smaller solution space) than the ten bag problem, Lumped strategy appears worse in the former case.

This behavior can also explain the much better earnings of proportional cloning when combined with Individual than Lumped selection strategy. Considering that CLONALG2 essentially focuses on selecting copies of the best antibodies, proportional cloning leads to even greater probability of being trapped in local optima. On the contrary, since CLONALG1 already guarantees a wider selection range, the increased number of clones from solutions at local

optima also increases the chance of jumping to better local or global optima.

Intriguingly, GA and (to a lesser extend) optAInet ranked noticeably low for ten bags. A common feature of both methods is interactions between individuals (antibody network interactions in the former, chromosomal crossover in the latter). This trait allows wider exploration of the solution space, while it reduces local searches and consequently may prevent local optimization, which as illustrated is more important for the bigger sized ten bag problem.

5.4 Comparison on Simulated Data

With regard to simulated problems, we again expanded the approach of Belloni et al., (2008). Following Kohli and Sukumar (1990), individual part-worths and seller profits for each attribute level were simulated using iid uniform [0,1] distributions, while for each simulation three status quo products were randomly chosen to be offered by competing firms. We simulated 12 larger problem sizes in addition to the original 12, and generated 10 problem instances for each size. Tables A.10 and A.11 with the complete results and the exact problem sizes, are included in the Appendix A.3, respectively.

Table 5 Comparison of methods on 13 small-sized simulated problems, including sizes from Belloni et al., (2008). The same representative variants from Table 4 are shown. Results are averaged over 13×10 trials. Relative variables are normalized by the highest valued quantity out of all methods (best overall solution for each subproblem, and highest overall number of evaluations for each subproblem).

Method	Mean Relative Best Earnings (%)	Best Solution Frequency (%)	Difference from GA (p-value %)	Mean Relative # of Evaluations per Trial (%)	MeanCPU Time per Trial (s)
propCLONALG1	99.91	90.77	25.01	28.89	9.99
statOPTIA1fcm	99.87	90	68.92	27.07	9.28
GA	99.86	89.23	-	53.95	9.46
CLONALG1	99.84	86.15	41.94	26.57	9.55
statCLONALG1	99.83	89.23	59.19	25.69	9.15
BCA1	99.74	86.15	2.16	18.73	9.38
SA	99.73	92.31	4.95	96.75	32.24
statCLONALG1fcm	99.67	91.54	1.24	27.29	9.12
optAInet	99.61	90	1.36	26.51	9.05
statCLONALG2fcm	99.55	78.46	0.02	22.74	8.94

In Table 5 we observe that for the small problems there are no significant differences (paired t-tests) between the mean relative earnings of GA and the best Individual strategy-based CSAs.

However, Table 6 confirms that in larger problem sizes, the performance of GA and (again to a lesser extend) optAInet diminishes. Tables A.12 and A.13 with the complete results and the exact problem sizes are included in the Appendix A.3, respectively.

Table 6 Comparison of methods on 12 big-sized simulated problems. The same representative variants from Table 4 are shown. Results are averaged over 12×10 trials.

Method	**Mean Relative Best Earnings** (%)	**Best Solution Frequency** (%)	**Difference from GA** (p-value %)	**Mean Relative # of Evaluations per Trial** (%)	**MeanCPU Time per Trial** (s)
CLONALG1	99.84	72.50	<0.01	55.22	15.17
propCLONALG1	99.84	68.33	<0.01	50.77	14.20
statOPTIA1fcm	99.83	68.33	<0.01	58.03	14.55
statCLONALG1	99.71	60.00	0.99	58.19	14.80
GA	99.59	42.50	-	93.18	14.65
optAInet	99.56	60.00	84.85	57.61	14.55
BCA1	99.54	45.00	52.95	38.19	15.40
statCLONALG1fcm	99.28	42.50	<0.01	57.22	13.72
statCLONALG2fcm	99.15	32.50	<0.01	55.71	15.18

Finally, noticing that we simulated problems with smaller n_{att} than the real problem, BCA1 and BCA2 could not perform similarly well. Since the contiguous hypermutation operator usually changes a continuous part of the clone, few features per product mean that from such changes radically new product lines will emerge with more than one different products. Therefore, this operator limits the fine-tuning ability of CSA, which (as we have already seen in several instances) is important for large problem sizes.

5.5 Comparison using different objectives

To further explore and demonstrate conditions under which CSA could perform well, the performance of the comparing algorithms is now assessed using different objectives, such as market share, income and buyer's welfare (Green & Krieger, 1985; Tsafarakis et al., 2011). A summary of the performance of the comparing algorithms is presented in Tables 7-9, while Table 10 shows if there are any statistically significant differences between them, at the 5% significance level. Particularly, (0) indicates that there is no statistically significant difference and (1) indicates that there is statistically significant difference. For instance, considering

optAInet and SA, (1,1,0) indicates that the differences when using market share or income as an objective function are statistically significant, and that the difference when using buyer's welfare as an objective function, is not, respectively. Complete versions of the following tables (Tables A.15 -A.18) are included in Appendix A.3.

Table 7 Comparison of methods when using market share (%) as an objective function

	Worst	Best	Mean	Median	St.d.
optAInet	98.15	98.15	98.15	98.15	2.00E-13
GA	98.15	98.15	98.15	98.15	2.00E-13
SA	95.99	98.15	97.89	98.15	5.79E-01

Table 8 Comparison of methods when using income as an objective function

	Worst	Best	Mean	Median	St.d.
optAInet	28570	28570	28570	28570	0.00E+00
GA	28440	28570	28567.4	28570	1.83E+01
SA	28180	28570	28551	28570	8.34E+01

Table 9 Comparison of methods when using buyer's welfare as an objective function

	Worst	Best	Mean	Median	St.d.
optAInet	54807.5	54807.5	54807.5	54807.5	0.00E+00
GA	54807.5	54807.5	54807.5	54807.5	0.00E+00
SA	54706.9	54807.5	54806.49	54807.5	1.01E+01

Table 10 Statistical analyses of comparison of methods

	optAInet	GA	SA
optAInet	-	0,0,0	1,1,0
GA	0,0,0	-	1,1,0
SA	1,1,0	1,1,0	-

As we can see the performance of optAInet is better than that of SA and comparable to that of GA, across three different objectives for the optimal PLD problem.

5.6 Diversity of the final population of CSA

Since most of the times product managers seek for a set of near-optimal solutions (Balakrishnan & Jacob, 1995, 1996), the diversity of the alternative solutions retrieved from CSA, is assessed through the estimation of the number of unique solutions in CSA's final population, by calculating the percentage of unique solutions in the final population as well as the percentage of unique solutions whose fitness is at

least 99% of the best solution's fitness. 100 reapplications of each algorithm are performed, when using the real dataset used in Subsection 5.1, and the results are demonstrated in Table 11. GA was not included in this comparison for the following two reasons: First, the diversity of GA's final population has already been investigated in previous research (Balakrishnan & Jacob, 1995, 1996; Tsafarakis et al., 2011) and second, GA's comparison with CSA regarding diversity, would not be considered fair, since GA is fine-tuned by Belloni et al. (2008) when performing on the particular dataset, to run until the entire population is homogeneous (all product lines are the same). According to Tsafarakis et al. (2011), two product lines are different if they differ in at least one product, while two products are different if they differ in the level of at least one attribute.

Table 11 Diversity of solutions in the final population

Variant	*Population size*	*Unique solutions (%)*		*Unique solutions with fitness at least 99% of the best (%)*	
		Mean	St.d.	Mean	St.d.
BCA1	20	94.90	5.27E+00	33.55	1.52E+01
BCA2	100	70.25	5.16E+00	8.49	7.81E+00
CLONALG1	20	97.05	4.03E+00	7.45	4.63E+00
CLONALG1elit	100	9.16	2.44E+00	4.67	2.20E+00
CLONALG1fcm	75	98.12	1.64E+00	3.41	1.84E+00
CLONALG2	100	84.25	3.25E+00	2.97	3.41E+00
CLONALG2fcm	100	88.61	2.95E+00	1.87	2.22E+00
statCLONALG1	20	94.25	4.73E+00	9.95	4.90E+00
statCLONALG1fcm	20	99.25	1.79E+00	2.80	2.69E+00
statCLONALG2	100	67.40	5.00E+00	8.87	8.68E+00
statCLONALG2fcm	100	90.99	2.66E+00	1.73	1.28E+00
statOPTIA1	20	48.60	1.21E+01	22.95	1.95E+01
statOPTIA1fcm	20	90.95	7.13E+00	27.80	2.01E+01
statOPTIA2	100	54.10	6.73E+00	12.28	1.23E+01
statOPTIA2fcm	100	84.46	3.69E+00	11.22	7.45E+00
optAInet	100 (Initial)	7.54	3.02E+00	1.32	5.84E-01
OPTIA1	20	56.90	1.01E+01	21.10	2.17E+01
OPTIA1fcm	20	84.10	9.60E+00	27.00	2.43E+01
OPTIA2	100	47.15	8.54E+00	12.44	1.18E+01
OPTIA2fcm	100	88.15	3.37E+00	1.41	1.60E+00
propCLONALG1	20	95.40	4.80E+00	10.60	5.70E+00
propCLONALG2	100	29.90	4.41E+00	11.18	1.03E+01
propOPTIA1	20	53.55	1.25E+01	15.75	1.61E+01

Table 11 demonstrates that most CSA approaches except CLONALG1elit and optAInet, have a

satisfying percentage of unique solutions in their final population, while 11 out of the 23 CSA variants have a percentage greater than 10 % of unique solutions in the final population with fitness at least 99% of the best. As a result, most CSA variants provide the decision maker with a wide range of unique high-quality solutions, among which s/he can make his/her choice. CLONALG1elit and optAInet appear to converge to a final population of low diversity with multiple copies of a few good solutions. Such a small set of unique chromosomes can be too restrictive in many situations because the solutions may prove to be almost identical, representing product lines that differ in only a single attribute level of one product.

6. Choosing Operators: Balance Between Local and Global Search

In the previous sections we have developed and tested 23 CSA variants that contain several different operators. In spite of being the last step in the sequence of operations in CSAs, our results have shown that the choice between the clonal selection strategy of CLONALG1 and CLONALG2 by Cutello et al., (2005) has the largest impact on the efficiency of CSAs. The former, Individual selection strategy, clearly dominates in all our product line optimization examples. As Cutello et al., (2005) recognized, Individual selection strategy allows for more diversity in the population by independently evolving each initial antibody. On the contrary, Lumped strategy mainly focuses on the best antibodies of each generation, because these are the most probable ones to generate the best clones. Therefore, we conclude that CLONALG1-based CSAs are better suited for optimizing product lines (among other multimodal objective functions (Cutello et al., 2005)), and contributions from the other operators probably depend on how much they modify the searching focus of CSAs. Similarly, various operators have been used to increase the local search capabilities of GA (Fruchter et al., 2006). In an effort to further disentangle their contributions, we continue with analysis and discussion for the remaining operators.

6.1 OptAInet

We previously saw that the mean population earnings of optAInet highly fluctuate during the late generations. This is in agreement with the findings of De Castro and Timmis (2002), who considered this behavior as an effort to locate several local optima. However, we have showed that this network operation prevents local optimization of the best antibodies, which is normally present in CLONALG1 and has proven to be important in our larger-sized problems.

Parameter configuration of optAInet additionally indicated its limitations. Modifying the threshold parameter (see Appendix A.2 for used values) for increasing diversity to higher values resulted in premature population increase and insertion of new random antibodies. Furthermore, we found insertion of new random antibodies fruitless, not only for optAInet, but for almost all CSAs as well. Finally, reducing the per cent similarity under 70% resulted in large population reduction and thus insufficient

search around the neighbor of deleted solutions.

6.2 Elitism and Aging

The use of strong elitism in CLONALG1elit returned considerably less profitable solutions. This finding is in concordance with Ahn and Ramakrishna (2003) for Genetic algorithms. In particular, they showed that strong elitism limited diversity in the population and therefore led to premature convergence.

Intriguingly, the effect of weak elitism in our CSAs depended on their population size. For large populations (100 antibodies), weak elitism was beneficial. However, CSAs with just 20 antibodies per generation performed better without weak elitism. According to the same rationale as strong elitism, the impact of weak elitism on the diversity of such small populations is relatively large. BCA1 is the exception that proves the rule, since its hypermutation operator may incur dramatic changes to clones. Consequently, weak elitism in this case permits better local search around the best antibody of each generation.

As far as aging operators go, we have not found significant differences to CSA efficiency, except statOPTIA1fcm for the problem of ten bags, which agrees with previous mixed results from various objective functions (Cutello et al., 2005). Since a large age limit t_{lim} for OPTIA1 would turn CLONALG1 into the worse performing CLONALG1elit, we chose $t_{lim} = 2$, which kept its performance similar to CLONALG1 methods. For OPTIA2, t_{lim} was varied between 2 and 5, as $t_{lim} = 1$ would mean no chance of evolving the antibodies of each generation, while large t_{lim} would simply turn OPTIA2 into CLONALG2.

6.3 Somatic Hypermutation

Last but definitely not least, our results showed that the type of hypermutation plays an important role both in quality and speed of generating solutions. As previously mentioned, static hypermutation returned slightly better results than contiguous and inversely proportional hypermutation. Recall that we limited all algorithms to similar average CPU time intervals. Since the latter operators spent more time per generation due to more mutations (contiguous) or due to the extra fitness function evaluations (inversely proportional), their optimization process probably halted slightly prematurely compared to static hypermutation.

These operators do not come without perks though, as they both required considerably less objective function evaluations. Recollecting that this profit maximization problem was supported by precomputation, in cases where precomputation is not possible and the computational cost of function evaluations is significant, both operators will offer faster convergence to best solutions (previously shown for BCA solving continuous optimization

problems (Kelsey & Timmis, 2003)). More specifically for BCA, small- and large-scale mutations by the contiguous hypermutation operator probably offer the opportunity to both adequately explore and escape areas of local optima. However, our results confirmed the hypothesis that the performance of BCA is data- representation-dependent (Kelsey & Timmis, 2003), as illustrated by the reduced average earnings of BCA1 for problems with just a handful of features per product. Analogously, inversely proportional hypermutation performs local search around the best clones, and wider search for the worst ones. Our value for parameter ρ ensured that the operator would incur a single mutation on the best clones, two or three to middle-performing ones, and several mutations to the last ranked clones. On top of this advantage, its methods (such as CLONALG1, unlike BCA1) were robust to changes in problem size.

Finally, FCM had been previously adopted to prevent premature convergence and improve search when optimizing trap functions (Cutello et al., 2004, 2006). Although they indeed reported better results when using FCM, they only tested its performance with Lumped selection strategy. Our results revealed that the contribution of FCM to CSA performance depends on the type of selection strategy. While we confirmed previous findings for CLONALG2 and OPTIA2 variants, FCM instead worsened the solutions of CLONALG1-based methods. The little significance of FCM is further supported by having to set the amount of mutations per clone for static hypermutation to its meaningfully lowest value ($m_c = 2$ for FCM methods, $m_c = 1$ for the rest). Therefore, as independent antibody evolution in Individual selection strategy has already prevented premature convergence to local optima, further delay of convergence by FCM is detrimental.

7. Conclusions

In this paper we have implemented and compared several Clonal Selection Algorithms for solving the optimal PLD problem. Based on the previous comparative analysis of Belloni et al., (2008), we have illustrated the potential of several CSA methods compared to other optimization algorithms. Moreover, similarly to Balakrishnan and Jacob (1995) we also believe that the use of maximally different methods has permitted us to get a better fix on the solution and thereby increasing the product manager confidence in using CSA in the product designing process. Among our methods, CLONALG1-based variants have clearly and consistently exhibited the best results. Additionally, we have extended their paradigm to problems with larger solution space. Large problem sizes have allowed us to further investigate the robustness of each CSA operator. In this fashion, we have exemplified how searching focus modifications by optAInet, elitism, aging and other hypermutation strategies affect convergence speed, and thus solution quality. In this

context, and to help the reader understand the way the CSA variants were coded, a Matlab code of optAInet can be found in Appendix B (.m file is available online).

Although an extensive amount of CSA variants was considered here, we restricted our comparison to other optimization algorithms provided by Belloni et al., (2008). For instance, the performance of GA could be enhanced by a more robust approach to parameter selection or with the addition of Hybrid techniques that can improve GA's convergence rate and speed (e.g. Zervoudakis et al., 2020). Furthermore, we limited our investigation to canonical CSA forms. Consequently, several hybrid methods that combine CSA with other heuristics remain to be tested in the present context (reviewed in Ulutas and Kulturel-Konak 2011, Dasgupta et al., 2011). Considering the local search ability of CSA, it is expected that its cooperation with globally focused methods (such as GA, Particle Swarm and Ant Colony Optimization) could improve convergence speed in terms of objective function evaluations. However, translating fewer evaluations into significantly faster iterations should be more challenging.

As more advanced underlying market models could be adopted for product line optimization (considering for example competitive reactions, dynamic customer preferences and multiobjective optimization), we believe that CSAs would solve these problems equally well. Obviously, the shape of solution space differs depending on problem type and formulation. Therefore, one should take these factors into account when selecting an optimization method. Nevertheless, our general conclusions should remain beneficial not only when applying CSAs to the field of PLD, but to other combinatorial problems as well.

Finally, despite the more than promising algorithm's performance in the present study, its effectiveness may be questionable in some cases. For instance, even though the best values for the algorithm's parameters were used, the performance of the CSA variants is not only highly dependent on its parameter settings, but also on the dataset (Tsafarakis et al., 2020). As a result, CSA variants may not be as effective using the same parameters when performing on different datasets, and therefore it is very likely that a tuning process may be needed which may be time consuming. To overcome such difficulties, techniques as Fuzzy Logic (FL) can be applied for future research, to automatically determine the parameter settings of CSA (Noorbin & Alfi, 2018; Olivas et al., 2018). Moreover, the proposed method cannot be applied when some of the design decisions and product attributes are continuous variables (e.g. length size or weight capacity) (Michalek et al., 2011). In such cases, continuous operators must be combined with CSA in order to perform on continuous variables (Zhang et al., 2019).

Acknowledgments

We wish to thank professors John Hauser and Olivier Toubia for the provision of the datasets, and professor Matthew Selove for the provision of the GA and SA codes.

References

Ahn CW, Ramakrishna RS (2003) Elitism-based compact genetic algorithms. Evolutionary Computation, IEEE Transactions on 7(4): 367–385.

Albritton DM, McMullen PR (2007) Optimal Product Design Using a Colony of Virtual Ants. *European Journal of Operational Research* 176 (1): 498-520.

Alexouda G, Paparrizos K (2001) A Genetic Algorithm Approach to the Product Line Design Problem Using the Seller's Return Criterion: An Exhaustive Comparative Computational Study. *European Journal of Operational Research* 134 (1): 165-178.

Balakrishnan P. V., and Varghese S. Jacob (1996). Genetic algorithms for product design. *Management Science* 42(8), 1105-1117.

Balakrishnan P. V., and Varghese S. Jacob (1995). Triangulation in decision support systems: algorithms for product design. Decision Support Systems 14(4), 313-327.

Balakrishnan P. V., Gupta R, Jacob V. (2004) Development of hybrid genetic algorithms for product line designs. *IEEE Transactions on Systems, Man, and Cybernetics* 34(1): 468-483.

Balakrishnan P. V., Gupta R., & Jacob V. S. (2006). An investigation of mating and population maintenance strategies in hybrid genetic heuristics for product line designs. *Computers & Operations Research*, 33(3), 639–659

Belloni A, Freund R, Selove M, Simester D (2008) Optimizing product line designs: Efficient methods and comparisons. *Management Science* 54(9):1544–1552.

Bertsimas D, Mišić VV (2017) Robust product line design. *Operations Research* 65(1):19–37

Burnet SFM, et al. (1959) *The clonal selection theory of acquired immunity*, volume 3 (Vanderbilt University Press Nashville).

Cohn M, Mitchison NA, Paul WE, Silverstein AM, Talmage DW, Weigert M (2007) Reflections on the clonal-selection theory. *Nature Reviews Immunology* 7(10):823–830.

Cutello V, Narzisi G, Nicosia G, Pavone M (2005) Clonal selection algorithms: A comparative case study using effective mutation potentials. Jacob C, Pilat M, Bentley P, Timmis J, eds., *Artificial Immune Systems*, volume 3627 of *Lecture Notes in Computer Science*, 13–28 (Springer Berlin Heidelberg), ISBN 978-3-540-28175-7.

Cutello V, Nicosia G, Oliveto PS (2006) Analysis of an evolutionary algorithm with hypermacromutation and stop at first constructive mutation heuristic for solving trap

functions. *Proceedings of the 2006 ACM symposium on Applied computing*, 945–949 (ACM).

Cutello V, Nicosia G, Pavone M (2004) Exploring the capability of immune algorithms: A characterization of hypermutation operators. Nicosia G, Cutello V, Bentley P, Timmis J, eds., *Artificial Immune Systems*, volume 3239 of *Lecture Notes in Computer Science*, 263–276 (Springer Berlin Heidelberg), ISBN 978- 3-540-23097-7.

Dasgupta D, Yu S, Nino F (2011) Recent advances in artificial immune systems: models and applications. *Applied Soft Computing* 11(2):1574–1587.

De Castro LN, Timmis J (2002) An artificial immune network for multimodal function optimization. *Evolu- tionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, 699–704 (IEEE).

De Castro LN, Von Zuben FJ (2002) Learning and optimization using the clonal selection principle. *Evolu- tionary Computation, IEEE Transactions on* 6(3):239–251.

Engin O, Döyen A (2004) A new approach to solve hybrid flow shop scheduling problems by artificial immune system. *Future generation computer systems* 20(6):1083–1095.

Fruchter G, Fligler A, Winer R (2006) Optimal product line design: Genetic algorithm approach to mitigate cannibalization. *Journal of optimization theory and applications* 131(2):227–244.

Germain RN (2004) An innately interesting decade of research in immunology. *Nature medicine* 10(12):1307– 1320.

Goldsby RA, Kindt TJ, Kuby J, Osborne BA (2002) *Immunology, Fifth Edition* (W. H. Freeman), 5th edition, ISBN 9780716749479

Green PE, Krieger AM (1985) Models and heuristics for product line selection. *Marketing Science* 4(1): 1-19.

Hart E, Timmis J (2008) Application areas of AIS: The past, the present and the future. *Applied soft computing* 8(1):191–201.

Hauser J, Tellis GJ, Griffin A *(*2006) Research on innovation: A review and agenda for marketing science. *Marketing science* 25 (6): 687-717.

Kelsey J, Timmis J (2003) Immune inspired somatic contiguous hypermutation for function optimisation. *Genetic and Evolutionary Computation-GECCO 2003*, 207–218 (Springer).

Kohli R, Krishnamusti R (1989) Optimal product design using conjoint analysis: Computational complexity and algorithms. *European Journal of Operational Research* 40(2): 186-195.

Michalek J. J., Ebbes P., Adigüzel, F., Feinberg F. M., & Papalambros P. Y. (2011). Enhancing marketing with engineering: Optimal product line design for heterogeneous markets. *International Journal of Research in Marketing, 28*, 1–12.

Noorbin, S. F. H., & Alfi, A. (2018). Adaptive parameter control of search group algorithm using fuzzy logic applied to networked control systems. *Soft Computing*, 22(23), 7939–7960. https://doi.org/10.1007/s00500-017-2742-0

Olivas, F., Valdez, F., Castillo, O., & Melin, P. (2018). *Dynamic Parameter Adaptation for Meta-Heuristic Optimization Algorithms Through Type-2 Fuzzy Logic*. Springer International Publishing. https://doi.org/10.1007/978-3-319-70851-5

Saridakis, C., Tsafarakis, S., Delias, P., Baltas, G., & Matsatsinis, N. (2015). Optimizing differentiation and commonality levels among models in car line-ups: An empirical application of a nature-inspired heuristic mechanism. *Expert Systems with Applications*, *42*(5): 2323–2335.

Steiner, W., Hruschka, H. (2003). Genetic algorithms for product design: How well do they really work? *International Journal of Market Research* 45(2), 1-13.

Timmis J, Hone A, Stibor T, Clark E (2008) Theoretical advances in artificial immune systems. *Theoretical Computer Science* 403(1):11–32.

Toubia O, Simester DI, Hauser JR, Dahan E (2003) Fast polyhedral adaptive conjoint estimation. *Marketing Science* 22(3):273–303.

Tsafarakis, S. (2016). Redesigning product lines in a period of economic crisis: a hybrid simulated annealing algorithm with crossover. *Annals of Operations Research*, *247*(2): 617–633.

Tsafarakis, S., Marinakis, Y., & Matsatsinis, N. (2011). Particle swarm optimization for optimal product line design. *International Journal of Research in Marketing*, 28(1), 13–22. https://doi.org/10.1016/j.ijresmar.2010.05.002

Tsafarakis, S., & Matsatsinis, N. (2010). Designing Optimal Products: Algorithms and Systems. In J. Casillas, F.J. Martínez-López (Eds.), *Marketing intelligent systems using soft computing: Managerial and research applications, Studies in fuzziness and soft computing*, Vol. 258, Springer, pp. 295-336 https://doi.org/10.1007/978-3-642-15606-9_19

Tsafarakis, S., Zervoudakis, K., Andronikidis, A., & Altsitsiadis, E. (2020). Fuzzy self-tuning differential evolution for optimal product line design. *European Journal of Operational Research*, *287*(3), 1161–1169. https://doi.org/10.1016/j.ejor.2020.05.018

Ulutas BH, Kulturel-Konak S (2011) A review of clonal selection algorithm and its applications. *Artificial Intelligence Review* 36(2):117–138.

Zervoudakis, K., Tsafarakis, S., & Paraskevi-Panagiota, S. (2020). A New Hybrid Firefly – Genetic Algorithm for the Optimal Product Line Design Problem. In N. Matsatsinis, Y. Marinakis, & P. Pardalos (Eds.), *Learning and Intelligent Optimization. LION 2019. Lecture Notes in Computer Science* (pp. 284–297). Springer, Cham. https://doi.org/10.1007/978-3-030-38629-0_23

Zhang, W., Zhang, W., Yen, G. G., & Jing, H. L. (2019). A cluster-based clonal selection algorithm for

optimization in dynamic environment. *Swarm and Evolutionary Computation*, 50, 100454.

Zufryden F (1977) A conjoint measurement-based approach for optimal new product design and market segmentation. In *Analytical approaches to product and marketing planning*, Shocker A.D. (ed.), Marketing Science Institute, Cambridge, MA.