Journal of Computational and Applied Mathematics 293 (2016) 180–191

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Studying self-balancing strategies in island-based multimemetic algorithms

Rafael Nogueras, Carlos Cotta *

Dept. Lenguajes y Ciencias de la Computación, Universidad de Málaga, ETSI Informática, Campus de Teatinos, 29071 Málaga, Spain

ARTICLE INFO

Article history:
Received 4 November 2014
Received in revised form 27 March 2015

Keywords:
Memetic algorithms
Multimemetic algorithms
Load balancing
Self-adaptation
Faulty environment

ABSTRACT

Multimemetic algorithms (MMAs) are memetic algorithms that explicitly exploit the evolution of memes, i.e., non-genetic expressions of problem-solving strategies. We aim to study their deployment on an unstable environment with complex topology and volatile resources. We analyze their behavior and performance on environments with different churn rates, and how they are affected by the use of self-balancing strategies aiming to compensate the loss of existing islands and react to the apparition of new ones. We investigate two such strategies, one based on quantitative balance (in which populations are resized dynamically to cope with node failure/recoveries) and another on qualitative balance (in which genetic/memetic information is actually exchanged to achieve balance). We evaluate these on scale-free network topologies and compare them to an unbalanced strategy that keeps island sizes constant. Experimentation firstly focuses on memetic takeover, carried out on an idealized selecto-Lamarckian model of MMAs (used as a surrogate of the latter) and indicating that the two balancing strategies exhibit complementary profiles in terms of diversity preservation. The results also indicate that the qualitative version is more robust to churn than both the unbalanced and the quantitatively balanced counterpart. This is subsequently confirmed with an empirical evaluation of full-fledged MMAs on a benchmark composed of four hard pseudo-Boolean problems. The qualitative version provides the best performance in global terms, significantly outperforming the remaining variants.

1. Introduction

Memetic algorithms (MAs) [1] are optimization techniques based on the orchestrated interplay of elements from population-based global search methods and trajectory-based local search techniques [2]. A central tenet in MAs is the notion of *meme* [3]: originally defined as units of imitation, memes can be interpreted in the context of MAs as computational problem-solving procedures. While these can take different forms, they commonly represent local-search techniques, often fixed or pre-defined in advance. Hence, these MAs can be regarded as operating with static implicit memes. This is not the only possibility though. Indeed, explicitly handling (and evolving) memes is an idea that has been around for some time now – cf. [4] – and is now a core idea in the concept of memetic computing [5–8]. Such an explicit treatment of memes can be found in, for example, multimemetic algorithms (MMAs) [9–13]. In these techniques, each solution carries memes that determine the way self-improvement is conducted. Since these memes evolve alongside solutions, the whole system constitutes a self-adaptive search approach [14–17].

* Corresponding author. Tel.: +34 952 137158; fax: +34 952 131397.
E-mail address: ccottap@lcc.uma.es (C. Cotta).

http://dx.doi.org/10.1016/j.cam.2015.03.047

When analyzing the way memes propagate throughout the population in an MMA, we can observe that the propagation dynamics is more complex than that of genes, if only because memes are only indirectly evaluated according to the effect they exert on the latter. For this reason, mismatches between genes and memes may cause potentially good memes to become extinct or poor memes to proliferate [18]. These issues are particularly relevant to multi-population models of MMAs, in which, besides internal population dynamics, we have to consider the effect of the communication between populations too [19]. This is even more true in the presence of complex, dynamic computational environments such as those emerging from the use of peer-to-peer networks [20] and volunteer computing networks [21]. These are characterized by the volatility of computational resources, the term *churn* having been coined to denote the collective effect of a plethora of peers entering or leaving the system independently over time.

Focusing on the use of island-based evolutionary algorithms on these kinds of unstable computational platforms, the presence of churn can cause the best solution to be lost (if the island comprising it goes down before it has the opportunity to migrate [22]) and will, in general, have detrimental effects on the overall population diversity. This can be tackled using corrective measures – e.g., using a fault-management strategy to recover from failures [23–26] – or by preventive measures, whereby the algorithm self-adapts to failures as they happen, trying to maintain a broad genetic/memetic pool at all times. The latter may have the advantage of being inherently autonomous and decentralized, not requiring the global state of the system to be monitored or for external snapshots of it to be maintained. This approach is precisely the focus of this paper: we depart from the use of fault-recovery strategies considered in previous work [27] and investigate the effect that introducing decentralized balancing strategies has on the functioning of the algorithm. To this end, we firstly use an idealized selecto-Lamarckian model of MMAs [18] which allows studying issues such as memetic diversity and convergence. This model is extended here to an island-based context, as described in Section 2.1. Subsequently, we describe a model of the computational environment (analogous to that used in [27]) in Section 2.2 and present a self-balancing algorithm in Section 2.3. Then, we report a broad experimental evaluation in Section 3. After analyzing the behavior of the surrogate model, results are reported on actual full-fledged MMAs in order to confirm the previous findings, analyzing performance and providing a sensitivity analysis of the self-balancing strategy. We close the paper with an overview of conclusions and an outline of future work in Section 4.

2. Material and methods

2.1. Algorithmic setting

As stated in the previous section, the first part of the experimentation has been done using an idealized selecto-Lamarckian model so as to obtain a preliminary assessment of the behavior of MMAs in terms of convergence and diversity when deployed on a dynamic computational scenario. This model is an abstract characterization of MMAs, first introduced in [18]. It consists of a population $P = [\langle g_1, m_1\rangle, \ldots, \langle g_\mu, m_\mu\rangle]$ of μ individuals, which are subject to the evolutionary operations of selection, local search and replacement as shown in Algorithm 1 (selecto-Lamarckian phase). Each individual is a tuple $\langle g_i, m_i\rangle \in D^2$, for some $D \subset \mathbb{R}$. In each tuple, g_i is the genotype (also representing fitness for simplicity) and m_i is a meme (its *potential*, to be precise). The latter is an idealized concept that tries to capture how good solutions can become by using this meme (thus constituting an abstract notion of meme fitness [28]). More precisely, this potential is expressed via a function $f : D^2 \to D$ monotonically increasing in the first parameter, which represents the application of a meme to a gene: an individual $\langle g, m\rangle$ becomes $\langle f(g, m), m\rangle$ after the application of the meme. It must hold that (i) $\lim_{n\to\infty} f^n(g, m) = m$ if $g < m$ ($f^n(g, m)$ being the n-fold application of the meme m to g) and that (ii) $f(g, m) = g$ if $g \geqslant m$. This means that the meme has no effect on solutions whose quality is higher than the meme's potential, but in the case that the quality is lower it improves the latter, reaching its potential in the limit. While this is obviously a highly idealized description of the action of memes (which in general depends on the match between the genotype and the meme on a problem-specific basis) it constitutes an initial approximation that can be used to study the generalities of meme propagation as shown in [18].

Interaction between individuals in a population is restricted by a spatial structure given by a $\mu \times \mu$ Boolean matrix S, where $S_{ij} = \texttt{true}$ if, and only if, the individual in the ith location can interact with the individual in the jth location [29]. In this case we consider panmixia, i.e., $S_{ij} = \texttt{true}$ for all i, j. This basic model is here extended to a multi-population setting [30,31] as illustrated in Algorithm 1: n_ι islands are assumed to work in parallel (being interconnected according to a certain topology $\mathcal{N}$) and migration steps are added before/after the selecto-Lamarckian phase. Migration is performed asynchronously: at the beginning of each cycle the island checks whether or not migrants have been received. If this is the case, they are accepted into the population following a given migrant replacement policy. Then, at the end of each cycle, migration is stochastically performed just like any other evolutionary operator. If done, some migrants are selected using a certain migrant selection policy and sent to neighboring islands. Following the results in [19], we use random selection of migrants and deterministic replacement of the worst individuals in the receiving island.

The selecto-Lamarckian model can be readily extended to a full-fledged MMA. Following previous work – e.g., [19,32] – we have specifically considered an MMA inspired by the work of Smith [13,33] wherein each individual in the population carries a binary genotype and a single meme representing a rewriting rule $A \to C$, where both A and C are patterns of a certain length taken from {0, 1, #}; the symbol '#' is a wildcard interpreted as "don't care" in the antecedent A of the rule and as "don't change" in the consequent C. These memes are utilized to generate neighbors of the solutions they are attached to, by looking for instances of A and substituting them with C; for example, let a genotype be 11101100, and let a meme be