Efficient Search of Relevant Structures in Complex Systems

Laura Sani[1], Michele Amoretti[1], Emilio Vicari[1], Monica Mordonini[1], Riccardo Pecori[1,4], Andrea Roli[2], Marco Villani[3], Stefano Cagnoni[1], and Roberto Serra[3]
Contact: michele.amoretti@unipr.it

[1]Dip. di Ingegneria dell'Informazione, Università degli Studi di Parma, Italia
[2]Dip. di Informatica - Scienza e Ingegneria
Università di Bologna - Sede di Cesena, Italia
[3]Dip. di Scienze Fisiche, Informatiche e Matematiche
Università degli Studi di Modena e Reggio Emilia, Italia
[4]SMARTest Research Centre, Università eCAMPUS, Novedrate (CO), Italia

Abstract. In a previous work, Villani et al. introduced a method to identify candidate emergent dynamical structures in complex systems. Such a method detects subsets (clusters) of the system's elements which behave in a coherent and coordinated way while loosely interacting with the remainder of the system. Such clusters are assessed in terms of an index that can be associated to each subset, called Dynamical Cluster Index (DCI). When large systems are analyzed, the "curse of dimensionality" makes it impossible to compute the DCI for every possible cluster, even using massively parallel hardware such as GPUs.
In this paper, we propose an efficient metaheuristic for searching relevant dynamical structures, which hybridizes an evolutionary algorithm with local search and obtains results comparable to an exhaustive search in a much shorter time. The effectiveness of the method we propose has been evaluated on a set of boolean models of real-world systems.

Keywords: complex systems, hybrid metaheuristics, local search

1 Introduction

The study of complex systems is related to the analysis of collective behaviors and emerging properties of systems whose components are usually well-known. Measuring the complexity of a composite system is a challenging task; dozens of measures of complexity have been proposed, several of which are based on information theory [1]. Detecting clusters of elements which interact strongly with one another is even more challenging, especially when the only information available is the evolution of their states in time.

The method proposed by Villani et al. [2] identifies emergent dynamical structures in complex systems, also referred to as relevant sets (RSs) in the following. To do so, it assesses the relevance of each possible subset of the system's variables, computing a quantitative index, denoted as Dynamical Cluster Index

(DCI). To fully describe a dynamical system based on the DCI it would therefore be necessary to compute such an index for all possible subsets of the system variables. Unfortunately, their number increases exponentially with the number of variables, soon reaching unrealistic requirements for computation resources. Therefore, to extract relevant DCI information about a system by observing its status over time, it is absolutely necessary to design efficient strategies, which can limit the extension of the search by quickly identifying the most promising subsets.

In this paper, we propose HyReSS (Hybrid Relevant Set Search), a hybrid metaheuristic for searching relevant sets within dynamical systems, based on the hybridization of an evolutionary algorithm with local search strategies. In the tests we have made on data describing both real and synthetic systems, HyReSS has been shown to be very efficient and to produce results comparable to an exhaustive search in a much shorter time.

The paper is organized as follows. In Section 2, we discuss previous related work. In Section 3, we describe the DCI-based approach. In Section 4, we present the evolutionary metaheuristic. In Section 5, we report some experimental results. Finally, in the last section, we conclude the paper summarizing our achievements and discussing future research directions.

2 Related Work

Several measures of complexity are based on information theory [1], which is convenient since any dynamically changing phenomenon can be characterized in terms of the information it carries. Hence, these measures can be applied to the analysis of any dynamical system. A widely-known information-theoretic framework by Gershenson and Fernandez [3] allows one to characterize systems in terms of emergence, self-organization, complexity and homeostasis. Such a framework has been applied, for example, to characterize adpative peer-to-peer systems [4], communications systems [5] and agroecosystems [6].

The DCI method [7, 8] is an extension of the Functional Cluster Index (CI) introduced by Edelman and Tononi in 1994 and 1998 [9, 10] to detect functional groups of brain regions. In our previous work, we extended the CI domain to non-stationary dynamical regimes, in order to apply the method to a broad range of systems, including abstract models of gene regulatory networks and simulated social [11], chemical [2], and biological [8] systems.

Genetic Algorithms (GAs) are popular search and optimization techniques, particularly effective when little knowledge is available about the function to optimize. However, some studies [12, 13] show that they are not well-suited to fine-tune searches in complex spaces and that their hybridization with local search methods, often referred to as memetic algorithms (MAs) [12], can greatly improve their performance and efficiency. Nevertheless, basic GAs and MAs are designed to find absolute optima and therefore are not capable of maintaining the diversity of solutions during evolution, which is essential when multimodal functions are analyzed, and the goal is to find as many local optima as possible.

To compensate for this shortcoming, various techniques, commonly known as *niching methods*, have been described in the literature, that maintain population diversity during the search process and allow the search to explore many peaks in parallel. Most niching methods, however, often require that problem-specific parameters, strictly related to the features of the search space, be set *a priori* to perform well. This is documented, for example, in [14], [15], and [16], that describe applications to mechatronics, image processing, and multimodal optimization, respectively.

Among the most renowned niching algorithms we can recall fitness sharing [17], sequential niching [18], deterministic crowding [19], and restricted tournament selection [20]. In this work we have used a modified version of deterministic crowding, because that method does not require *a priori* setting of problem-related parameters, such as the similarity radius, and its complexity is low, since it scales as $O(n)$ with the number of dimensions of the search space. This is probably the main reason why the usage of deterministic crowding is still often reported in the recent literature [21–23].

3 Approach

Many complex systems, both natural and artificial, can be represented by networks of interacting nodes. Nevertheless, it is often difficult to find neat correspondences between the dynamics expressed by these systems and their network description. In addition, network descriptions may be adequate only in case of binary relationships. In case systems characterized by non-linear interactions among its parts, the dynamic relationships among variables might not be entirely described **XXXX by the relevant topology, which could, for instance, permit interactions without effect on target elements [???? ha ragione il referee, va chiarito]**. In contrast, many of these systems can be described effectively in terms of coordinated dynamical behavior of groups of elements; relevant examples are Boolean networks [24], chemical or biological reaction systems [2] and functional connectivity graphs in neuroscience [25, 26]. Furthermore, in several cases, the interactions among the system elements are not known; it is therefore necessary to deduce some hints about the system's organization by observing the behavior of its dynamically relevant parts.

The goal of the work described in this paper is to identify groups of variables that are good candidates for being relevant subsets, in order to describe the organization of a dynamical system. We suppose that *(i)* the system variables express some dynamical behavior (i.e., there exists at least a subset of the system's observed states within which they change their value), *(ii)* there exist one or more subsets where these variables are acting (at least partially) in a coordinated way, and *(iii)* the variables of each subset have weaker interactions with the other variables or RSs than among themselves. The outcome of the analysis is essentially a list of possibly overlapping subsets, ranked according to some criteria, which provide clues for understanding the system organization.

The approach we use (*Dynamical Cluster Index* or DCI, method) has been previously presented by some of the authors of this paper. Here we briefly summarize the method, pointing to the relevant literature for further details [7, 8]. The DCI method relies on information theoretical measures, related with the Shannon Entropy [27].

Given the observational nature of our data, the probabilities are estimated by the relative frequencies of their values. Let us now consider a system U composed of K variables (*e.g.*, agents, chemicals, genes, artificial entities) and suppose that S_k is a subset composed of k elements, with $k < K$. The $DCI(S_k)$ value is defined as the ratio between the *integration* I of S_k and the *mutual information* M between S_k and the rest of the system:

$$DCI(S_k) \ = \frac{I(S_k)}{M(S_k; U \setminus S_k)} \tag{1}$$

where $I(S_k)$ measures the statistical independence of the k elements in S_k (the lower $I(S_k)$, the more independent the elements) while $M(S_k; U \setminus S_k)$ measures the mutual dependence between the subset S_k and the rest of the system $U \setminus S_k$. In formulas:

$$I(S_k) \ = \sum_{s \in S_k} H(s) - H(S_k) \tag{2}$$

$$M(S_k; U \setminus S_k) \ = H(S_k) + H(U \setminus S_k) - H(S_k, U \setminus S_k) \tag{3}$$

where $H(X)$ is the entropy or the joint entropy, depending on X being a single random variable or a set of random variables.

Any subset of the system's variables (Candidate Relevant Set - CRS - in the following) having $M = 0$ does not communicate with the rest of the system: it constitutes a separate system and its variables can be excluded from the analysis. The DCI scales with the size of the CRS, as already pointed out in [9], so it needs to be normalized by dividing each member of the quotient in Eq. 1 by its average value in a reference system where no dynamical structures are present. Following our previous works [7, 28, 8] our reference is a homogeneous system composed of the same number of variables and described by the same number of observations as the system under analysis. The values of the observations for the homogeneous system are generated randomly, according to the uni-variate distributions of each variable that could be estimated from the real observations if all variables were independent. Formally:

$$C'(S) = \frac{I(S)}{\langle I_h \rangle} / \frac{M(S; U \setminus S)}{\langle M_h \rangle} \tag{4}$$

Finally, in order to assess the significance of the normalized DCI values, a statistical index T_c can be computed [9]:

$$T_c(S) = \frac{C'(S) - \langle C'_h \rangle}{\sigma(C'_h)} = \frac{C(S) - \langle C_h \rangle}{\sigma(C_h)} \tag{5}$$

XXX nell'eq. precedente. fra il secondo e il terzo termine, c' un'uguaglianza esatta o approssimata ?

where $\langle C_h \rangle$, $\sigma(C_h)$, $\langle C'_h \rangle$ and $\sigma(C'_h)$ are, respectively, the average and the standard deviation of the DCI indices and of the normalized cluster indices from a homogeneous system with the same size as S_k.

The CRSs can be ranked according to their T_c: in both cases the analysis returns a huge set of candidates, most of which are a subset (or superset) of other CRSs. In order to identify the most relevant information, in [7] a post-processing sieving algorithm has been proposed able to reduce the list of CRSs to the most representative ones. The algorithm is based on the consideration that if CRS A is a proper subset of CRS B and ranks higher than CRS B, then CRS A should be considered more relevant than CRS B. Therefore, the algorithm keeps only those CRSs that are not included in or do not include any other CRS with higher T_c. This "sieving" action stops when no more eliminations are possible: the remaining groups of variables are the proper RSs. This procedure can also be extended to the identification of hierarchical relations among RSs: this topic is the subject of ongoing work.

In this paper, we focus onto a particularly critical issue, i.e., how to efficiently detect the highest-ranked CRSs according to their T_c. Indeed, the number of CRSs increases exponentially with its size, the number of CRSs of size k in a system of size K being $\binom{K}{k}$. However, to characterize a dynamical system of interest, one does not need to know the T_c index of all possible CRSs, but only to detect the CRSs for which the T_c is highest. To do so, we developed HyReSS, a hybrid metaheuristic described in the following section, postponing to future investigations the use of its results to detect the RSs and their hierarchy.

4 HyReSS: a Hybrid Metaheuristic for RS detection

HyReSS hybridizes a basic genetic algorithm with local search strategies driven by statistics on the results specific to the problem under consideration that are being obtained.

A genetic algorithm is first run to draw the search towards the basins of attraction of the main local maxima in the search space. Then, the results are improved by performing a series of local searches to explore those regions more finely and extensively.

The method can be subdivided into five main cascaded steps:

1. Genetic algorithm;
2. CRS relevance-based local search;
3. CRS frequency-based local search;
4. Group cardinality-based local search;
5. Merging.

4.1 Genetic Algorithm

The first evolutionary phase is a genetic algorithm based on the Deterministic Crowding (DC) algorithm, one of the most commonly used niching techniques.

We chose this method because of its lower complexity ($O(p)$ where p is the population size) with respect to other niching techniques. As specified above, HyReSS does not search a single CRS, but the set $\mathcal{B}$ of the N_{best} highest-Tc CRSs.

Each individual corresponds to one CRS and is a binary string of size N, where each bit set to 1 denotes the inclusion in the CRS of the corresponding variable, out of the N that describe the system. A list (termed "best-CRS memory" in the following) is created to store the best individuals that have been found along with their fitness values. At the end of the run, it should contain all CRSs in $\mathcal{B}$.

The initial population, of size p, is obtained by generating random individuals according to a pre-set distribution of cardinality (pairs, triplets, etc.). This kind of generation aims to create a sample that is as diversified as possible (avoiding repetitions) as well as representative of the whole search space.

The fitness function to be maximized corresponds directly to the Tc and is implemented through a CUDA[1] kernel that can compute in parallel the fitness values of large blocks of individuals.

Evolution proceeds by selecting $p/2$ random pairs of individuals and creating p children by means of a single-point crossover. After crossover, each child possibly replaces the most similar parent of lower fitness. To safeguard genetic diversity, a parent is only replaced if the child is not already present in the population.

This evolutionary process is iterated until the population is no more able to evolve, *i.e.*, the new generation remainss equal to the previous one. When that happens, new random parents are generated.

Mutation (implemented as bit flips) is applied with a low probability (P_{mut}) after each mating.

The termination condition for this evolutionary phase is reached when the number of evaluations of the fitness function exceeds a threshold α_f or new parents have been generated for α_p times.

The implemented algorithm is elitist, since a child is inserted in the new population only if its fitness is better than the fitness of the parent it substitutes. Therefore, the overall fitness of the population increases monotonically with the algorithm iterations. After the end of the evolutionary algorithm, the N_{gBest} fittest individuals are selected to seed the subsequent phases.

4.2 Variable relevance-based search

While running the genetic algorithm, a relevance coefficient RC_i is computed for each variable i of the system under examination. RC_i is higher if variable i is frequently included in high-fitness CRSs.

At the end of each generation t of the GA, a fitness threshold is set, separating high-fitness CRSs from low-fitness ones, and corresponding to a certain percentile β of the whole fitness range.

[1] https://developer.nvidia.com

$$\tau(t) = minFitness + (maxFitness - minFitness) * \beta \tag{6}$$

A presence coefficient (PC_i) and an absence coefficient (AC_i) are defined, for variable i, as the sum of the fitness values of the CRSs having fitness greater than τ, in which the variable has been, respectively, present or absent, cumulated over the generations and normalized with respect to the number of generations in which the corresponding CRSs have been included.

Based on these two coefficients, the ratio $R_{ap,i} = AC_i/PC_i$ is computed. The variable is classified as relevant if PC_i is greater than a threshold (the γ_{th} percentile of the full range of PC_i values) and $R_{ap,i}$ is lower than a certain threshold δ.

The corresponding local search procedure performs a recombination of the most relevant variables with other, randomly chosen, ones. As a first step, all possible subsets (simple combinations) of the most relevant variables are computed, excluding the subsets of cardinality 0 and 1. Then, for each subset dimension, the individual with the highest fitness is selected. Such individuals are the basis for generating new CRSs, by forcing the presence or absence of relevant/irrelevant variables and by randomly adding other variables into the RCSs. Every newly generated individual is evaluated and, should its fitness be higher, replaces the lowest-fitness individual in the best-CRS memory.

At the end of this phase a local search is performed in the neighborhood of the best individual of the best-CRS memory, which is updated in case new individuals with appropriate fitness are obtained.

4.3 Variable frequency-based search

In this phase, the same procedure used to generate new individuals and to explore the neighborhood of the best one is repeated, based on a different criterion.

We consider the frequency with which each variable has been included in the CRSs evaluated in the previous phases and use this value to identify two classes of variables, which are assigned higher probability of being included in the newly generated CRSs:

- variables with frequency much lower than the average;
- variables with frequency much greater than the average.

In fact, variables of the former kind may have been previously "neglected", thus it may be worth verifying whether they are able to generate good individuals, while variables of the latter kind are likely to have been selected very frequently in the evolutionary process because they actually have a significant relevance.

4.4 Group cardinality-based search

During the previous phases, HYReSS records the frequency with which groups of each possible cardinality (2, . . . , N-1) have occurred. These indices are then normalized according to the a priori probability of occurrence of such groups, given by the corresponding binomial coefficient $\binom{N}{c}$ where N is the total number of variables and c the cardinality of the group.

New CRSs are then generated using a procedure driven by such indices, such that cardinalities having lower values have higher probability of occurring and are possibly stored into the best-CRS memory according to their fitness.

4.5 Merging

In this phase a limited pool of variables is selected by considering all variables that are included in the highest-fitness CRSs in the best-CRS memory. In practice, a size θ for the pool is set; then, the best individuals are progressively OR-ed bitwise, in decreasing order of fitness starting from the best two CRSs, until the result of the bitwise OR contains θ bits set to 1 or all the CRSs have been processed. A final exhaustive search over all the possible CRSs that comprise the selected variables, is made, and the best-CRS memory is updated accordingly.

5 Experimental Results

In this section we illustrate three examples of dynamical systems we have used as benchmarks for HyReSS. The first one is a deterministic simulation of a chemical system called *Catalytic Reaction System (CatRS)*, described by 26 variables. The second one is a stochastic artificial system reproducing a *Leaders & Followers (LF)* behavior, featuring 28 variables. These examples have been analyzed using both exhaustive search and HyReSS. The third example, denoted as *Green Community Network (GCN)*, features 137 variables, a size for which an exhaustive search is not feasible on a standard computer. Thus, it was analyzed only by HyReSS. However, we could compare its results with those provided by field experts.

In all our test, we have performed 10 independent runs of HyReSS, to take the stochastic nature of the tool properly into account. We evaluated the results of HyReSS, when possible, by comparing the list of highest-T_c subsets it produced with the results of an exhaustive search. To let results be comparable, we relied on the same homogeneous system to compute normalized DCI values in both approaches. Tests were run on a Linux server equipped with a 1.6 GHz Intel I7 CPU, 6 GB of RAM and a GeForce GTX 680 GPU by NVIDIA. The parameters regulating the behavior of HyReSS were set as reported in Table 1.

Results are summarized in Table 2 and are discussed in the following subsections.

5.1 Catalytic Reaction System

The set of observations comes from the simulation of an open well-stirred chemostat (CSTR) with a constant incoming flux of feed molecules (empty ellipses in

Table 1. HYReSS parameter settings. The parameters are defined in section 4.

System	P_{mut}	p	α_f	α_p	β	γ	δ	θ
CatRS	.1	16384	163840	3	.75	.75	.3	15
LF	.1	16384	163840	3	.75	.75	.3	15
GCN (56 vars.)	.1	25600	256000	3	.75	.75	.3	15
GCN (137 vars.)	.1	50176	501760	3	.75	.75	.3	15

Table 2. Summary of HyReSS performances and comparison with an exhaustive search (ES), when possible.

System	N. Variables	N. Samples	Time (ES) [s]	Time (HyReSS) [s]	Speed-up
CatRS	26	751	180	24	7.5
LF	28	150	300	19	15.8
GCN	56	124	n.a.	71	n.a.
GCN	137	124	n.a.	258	n.a.

Figure 1) and a continuous outgoing flux of all molecular species proportionally to their concentration. Six catalyzed reactions produce six new chemical species (pattern- filled ellipses in figure 1) and are divided in two dynamical arrangements, a linear chain and a circle. The system asymptotic behavior is a fixed point: we perturbed each single produced chemical species, in order to allow the variables to change their concentrations over time and thus highlight their dynamic relationships (for detail, see [2]). In this work, we encoded each species' trajectory as a binary variable, the 0 and 1 symbols meaning respectively "concentration is changing" and "concentration is not changing".

As the system has "only" 26 variables, it has been possible to perform an exhaustive search to be used as reference, which took about 180 s. Producing almost identical results (the resulting error rate is less than 0.02[2]), the average running time of HyReSS was 24 s.

5.2 Leaders & Followers

The model is an abstract representation of a basic leader-followers (LF) model: it consists of an array of n binary variables $X = [x_1, x_2, \ldots, x_n]$, which could represent, for example, the opinion of n people in favor or against a given proposal. The model generates independent observations on the basis of the following rules:

- the variables are divided into four groups:
  - $G1 = \{A0, A1, A2, A3\}$
  - $G2 = \{A7, A8, A9\}$

[2] In one of the 10 runs, HyReSS failed to detect 1 of the first 50 RSs detected by the exhaustive search.

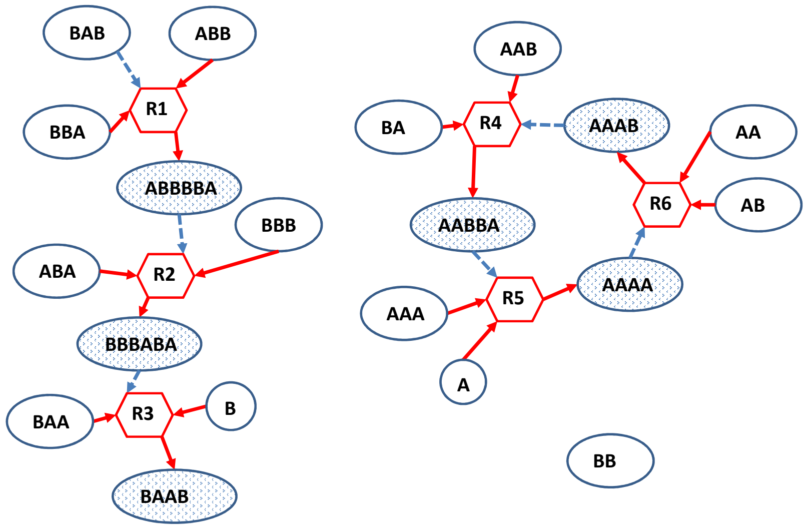

Fig. 1. The simulated CatRS. Circular nodes represent chemical species, while the white ones represent the species injected in the CSTR and the pattern filled ones those produced by the reactions. The diamond shapes represent reactions, where incoming arrows go from substrates to reactions and outgoing arrows go from reactions to products. Dashed lines indicate the catalytic activities.

  - $G3 = \{A12, A13, A14, A15, A16, A17, A18, A19, A20\}$
  - $G4 = \{A22, A23, A24, A25, A26, A27\}$

- the remaining variables $A4$, $A5$, $A6$, $A10$, $A11$ and $A21$ assume the value 0 or 1 with identical probability;
- variables $A0$, $A7$, $A12$, $A22$ and $A23$ are the leaders of their respective groups; at each step they randomly assume value 0 respectively with probability 0.4, 0.3, 0.3, 0.3 and 0.6, 1 otherwise;
- the other variables (*i.e.*, the followers) copy or negate the values of their leaders, with the exception of the followers belonging to group $G4$ computing the OR or AND function of their two leaders.

Given these rules, the system comprises only well-defined and non-interacting groups, dynamically separate with respect to the other independent random variables. However, its stochasticity could occasionally support the emergence of spurious relationships, which make the automatic detection of groups non-trivial.

The LF case we have considered features 28 variables. An exhaustive search takes about 300 s. HyReSS completes its run in 19 s, on average, always providing the same results as the exhaustive search, considering the **XXX** highest-T_c subsets as a reference.

XXX Quanto era grande (N_{best}) l'insieme usato come riferimento ?

5.3 Green Community Network

In this case, the data come from a real situation and show the participation (*i.e.*, the presence or absence) of 137 people in a series of 124 meetings, held during a project (the so-called Green Community project) which involved four mountain communities and focused on studies addressing energy efficiency and renewable energy production. The full original data set was multimodal, by far broader and more complex: some of us however (during the project "MD", within which the DCI methodology was first proposed) extracted this simplified data set to verify whether the DCI analysis would be able to evidence the formation of specific dynamics among subsets of participants, despite the apparently simplicity of the information carried by the observations.

During the MD project some metaheuristics were also applied to DCI, in order to deal with the enormous number of candidate RSs. **XXX e' proprio necessario dirlo, anche se non si riporta alcun risultato ?**

We have considered two versions of the GCN. The first one includes all 137 variables. The second one has only 56 variables, representing people who attended more than one meeting. Both cases have too many variables to perform an exhaustive search. Fortunately, HyReSS is quite effective, finding almost all the expected "dominant" communities, and efficient, with average running time of 71 s for 56 variables and 258 s for 137 variables.

6 Conclusion

In this paper we have presented HyReSS, an ad-hoc hybrid metaheuristic, tailored to the problem of finding the candidate Relevant Subsets of variables that describe a dynamical system. In developing our search algorithm, we have combined GAs' capacity of providing a good tradeoff between convergence speed and exploration, with local searches which refine and extend results, when correctly seeded. Using the deterministic crowding algorithm as a basis for the GA we guarantee that a large number of local maxima are taken into consideration in the early stages of HyReSS. The subsequent local search stages extend the results of the GA (stochastic) search more systematically to those CRSs that are most likely to have high fitness values, according to a few rules essentially derived from common sense.

In the benchmarks we took into consideration, HyReSS was very fast on an absolute scale, thanks to the GPU implementation of the fitness function. Even more importantly, at least for the smaller-size systems for which the comparison was possible, it could provide the same results as an exhaustive search based on the same parallel code, performing much fewer fitness evaluations and, consequently, in a significantly shorter time. The results obtained on the larger problems, on which the speed-up with respect to an exhaustive search is virtually incommensurable and for which a ground truth is therefore not available, were qualitatively aligned with the expectations of a domain expert who analyzed the data.

The availability of an efficient algorithm will allow us to extend our research on the detection of candidate RSs to dynamical systems of much larger sizes than previously possible. At the same time, it will allow for devising more complex analyses, by which we aim to detect also hierarchical relationships among RSs.

From an algorithmic viewpoint, we expect to be able to further optimize HyReSS by fully parallelizing the search, whose GPU implementation is currently limited to the evaluation of the fitness function, which, as usually happens, is the most computation-intensive module within the algorithm. The modular structure of HyReSS will allow us to perform a detailed analysis of the algorithm in order to highlight which stage is most responsible for the algorithm performance and possibly design some optimized variants accordingly. Finally, we will also study the dependence of the algorithm on its parameters, to further improve its effectiveness and, possibly, to devise some self-adapting mechanisms to automatically fit their values to the system under investigation.

Acknowledgments

The authors thank the UE project "MD – Emergence by Design", Pr.ref. 284625 7th FWP-FET programme for providing the data, which where in turn kindly provided by the Green Community project, sponsored by the National Association for Municipalities and Mountain Communities (UNCEM).

References

1. Prokopenko, M., Boschetti, F., Ryan, A.J.: An information-theoretic primer on complexity, self-organization, and emergence. Complexity **15**(1) (2009) 11–28
2. Villani, M., Filisetti, A., Benedettini, S., Roli, A., Lane, D., Serra, R.: The detection of intermediate-level emergent structures and patterns. In Miglino, O. et al., ed.: Advances in Artificial Life, ECAL 2013, The MIT Press (2013) 372–378
3. Gershenson, C., Fernandez, N.: Complexity and information: Measuring emergence, self-organization, and homeostasis at multiple scales. Complex. **18**(2) (November 2012) 29–44
4. Amoretti, M., Gershenson, C.: Measuring the complexity of adaptive peer-to-peer systems. Peer-to-Peer Networking and Applications (2015) 1–16
5. Febres, G., Jaff, K.: Calculating entropy at different scales among diverse communication systems. Complexity (2015)
6. Marull, J., Font, C., Padr, R., Tello, E., Panazzolo, A.: Energylandscape integrated analysis: A proposal for measuring complexity in internal agroecosystem processes (barcelona metropolitan region, 18602000). Ecological Indicators **66** (2016) 30 – 46

7. Filisetti, A., Villani, M., Roli, A., Fiorucci, M., Serra, R.: Exploring the organisation of complex systems through the dynamical interactions among their relevant subsets. In Andrews, P. et al., ed.: Proceedings of the European Conference on Artificial Life 2015, ECAL 2015, The MIT Press (2015) 286–293
8. Villani, M., Roli, A., Filisetti, A., Fiorucci, M., Poli, I., Serra, R.: The search for candidate relevant subsets of variables in complex systems. Artificial Life **21**(4) (11 2015)
9. Tononi, G., McIntosh, A., Russel, D., Edelman, G.: Functional clustering: Identifying strongly interactive brain regions in neuroimaging data. Neuroimage **7** (1998) 133–149
10. Tononi, G., Sporns, O., Edelman, G.M.: A measure for brain complexity: relating functional segregation and integration in the nervous system. Proceedings of the National Academy of Sciences **91**(11) (1994) 5033–5037
11. Filisetti, A., Villani, M., Roli, A., Fiorucci, M., Poli, I., Serra, R.: On some properties of information theoretical measures for the study of complex systems. In Pizzuti, C., Spezzano, G., eds.: Advances in Artificial Life and Evolutionary Computation: 9th Italian Workshop, WIVACE 2014, Vietri sul Mare, Italy, May 14-15, Revised Selected Papers, Cham, Springer International Publishing (2014) 140–150
12. Chen, X., Ong, Y.S., Lim, M.H., Tan, K.C.: A multi-facet survey on memetic computation. IEEE Transactions on Evolutionary Computation **15**(5) (Oct 2011) 591–607
13. Hu, X.M., Zhang, J., Yu, Y., Chung, H.S.H., Li, Y.L., Shi, Y.H., Luo, X.N.: Hybrid genetic algorithm using a forward encoding scheme for lifetime maximization of wireless sensor networks. IEEE Transactions on Evolutionary Computation **14**(5) (Oct 2010) 766–781
14. Behbahani, S., de Silva, C.W.: Niching genetic scheme with bond graphs for topology and parameter optimization of a mechatronic system. IEEE/ASME Transactions on Mechatronics **19**(1) (Feb 2014) 269–277
15. Chang, D., Zhao, Y., Zheng, C.: A real-valued quantum genetic niching clustering algorithm and its application to color image segmentation. In: International Conference on Intelligent Computation and Bio-Medical Instrumentation (ICBMI). (Dec 2011) 144–147
16. Pereira, M.W., Neto, G.S., Roisenberg, M.: A topological niching covariance matrix adaptation for multimodal optimization. In: IEEE Congress on Evolutionary Computation (CEC). (July 2014) 2562–2569
17. Goldberg, D.E., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application, Hillsdale, NJ, USA, L. Erlbaum Associates Inc. (1987) 41–49
18. Beasley, D., Bull, D.R., Martin, R.R.: A sequential niche technique for multimodal function optimization. Evol. Comput. **1**(2) (June 1993) 101–125
19. Manner, R., Mahfoud, S., Mahfoud, S.W.: Crowding and preselection revisited. In: Parallel Problem Solving From Nature, North-Holland (1992) 27–36
20. Harik, G.R.: Finding multimodal solutions using restricted tournament selection. In: Proceedings of the 6th International Conference on Genetic Algorithms, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1995) 24–31
21. Lacy, S.E., Lones, M.A., Smith, S.L.: Forming classifier ensembles with multimodal evolutionary algorithms. In: IEEE Congress on Evolutionary Computation (CEC). (May 2015) 723–729

22. Will, A., Bustos, J., Bocco, M., Gotay, J., Lamelas, C.: On the use of niching genetic algorithms for variable selection in solar radiation estimation. Renewable Energy **50** (2013) 168 – 176
23. Yannibelli, V., Amandi, A.: A deterministic crowding evolutionary algorithm to form learning teams in a collaborative learning context. Expert Systems with Applications **39**(10) (2012) 8584 – 8592
24. Villani, M., Barbieri, A., Serra, R.: A dynamical model of genetic networks for cell differentiation. PloS one **6**(3) (January 2011) e17703
25. Shalizi, C., Camperi, M., Klinkner, K.: Discovering functional communities in dynamical networks. In Airoldi, E. et al., ed.: Statistical Network Analysis: Models, Issues, and New Directions: ICML 2006 Workshop on Statistical Network Analysis, Pittsburgh, PA, USA, June 29, 2006, Revised Selected Papers, Berlin, Heidelberg, Springer Berlin Heidelberg (2007) 140–157
26. Sporns, O., Tononi, G., Edelman, G.: Theoretical neuroanatomy: Relating anatomical and functional connectivity in graphs and cortical connection matrices. Cerebral Cortex **10**(2) (2000) 127–141
27. Cover, T., Thomas, A.: Elements of information theory, 2nd Edition. Wiley-Interscience, New York (2006)
28. Villani, M., Carra, P., Roli, A., Filisetti, A., Serra, R.: On the robustness of the detection of relevant sets in complex dynamical systems. In Rossi, F. et al., ed.: Advances in Artificial Life, Evolutionary Computation and Systems Chemistry: 10th Italian Workshop, WIVACE 2015, Revised Selected Papers, Springer International Publishing (2016) 15–28