Impact of the Topology on the Performance of Distributed Differential Evolution

Ivanoe De Falco[1], Antonio Della Cioppa[2], Domenico Maisto[1](✉),
Umberto Scafuri[1], and Ernesto Tarantino[1]

[1] ICAR-CNR, Via P. Castellino 111, 80131 Naples, Italy
{ivanoe.defalco,domenico.maisto,umberto.scafuri,
ernesto.tarantino}@na.icar.cnr.it

[2] Natural Computation Lab, DIEM, University of Salerno,
Via Ponte don Melillo 1, 84084 Fisciano, SA, Italy
adellacioppa@unisa.it

Abstract. Migration topology plays a key role in designing effective distributed evolutionary algorithms. In this work we investigate the impact of several network topologies on the performance of a stepping–stone structured Differential Evolution model. Although some issues on the control parameters of the migration process and the way they affect the efficiency of the algorithm and the solution quality deserve further evaluative study, the influence of the topology on the performance both in terms of solution quality and convergence rate emerges from the empirical findings carried out on a set of test problems.

1 Introduction

Evolutionary Algorithms (EAs) [1–4] have proven to be very effective in dealing with hard optimization problems whose solution space is so large as to make an exhaustive search unviable [5,6]. Nonetheless, their main disadvantage is related to the convergence speed. A popular way for contrasting this drawback and achieving a speedup is to implement structured versions where the population is divided into multiple semi–isolated subpopulations (*demes*) connected each other according to a particular network topology. These subpopulations evolve independently and interact by means of a migration operator used to exchange individuals. The number of individuals that are sent to (received from) other demes is determined by the *migration rate*, while a *replacement function* defines how to include the immigrants into the target subpopulation. Besides, the *migration interval* establishes the exchange frequency among neighboring subpopulations [7]. Concerning the network topology, this distributed framework may be categorized as following either the *island model* (fully connected demes) or the *stepping–stone model* (interaction restricted to customized logical or physically connected demes) [8]. The connectivity degree of the topology beneath determines the number of the neighboring subpopulations and its diameter is the most important factor influencing the propagation of good individuals [9].

A.I. Esparcia-Alcázar et al. (Eds.): EvoApplications 2014, LNCS 8602, pp. 75–85, 2014.
DOI: 10.1007/978-3-662-45523-4_7

The separation of demes serves as a natural way to maintain the diversity reducing the possibility of population stagnation [9], may guide the evolution in many directions simultaneously, and may allow speedup in computation and improve solution quality with respect to a single EA evolution [10,11].

Originally developed for Genetic Algorithms (GAs) [1,3], the distributed approach has been employed also for different paradigms. Among these paradigms, distributed Differential Evolution (dDE) has been the subject of significant research [12–18]. The choice of DE [19] is due to its simple but powerful searching capability, and to its overall performance with respect to other stochastic and direct search global optimization techniques on a wide range of benchmark problems [20] and real world problems [21].

In the following we make reference to the stepping–stone dDE model. To assess the impact of the migration topology on a dDE algorithm, simulations have been performed on a range of test problems and for several network topologies by making use of a standard dDE algorithm, *i.e.*, DDE [22].

Paper structure is as follows: Section 2 illustrates the state of the art; Section 3 presents a description of the parallel framework. In Section 4 the experimental findings are shown and discussed together, and a statistical analysis is performed. The last section contains final remarks and future works.

2 State of the Art

Since the distributed models were introduced in connection with parallel GAs, it is not surprising that all the issues involved, including the migration topology, have been studied in this context. Several surveys have been published in the nineties [7,23]. Although in some case the influence of the migration topology has been neglected [7], research was conducted to analyze its impact [10,24]. Naturally the distributed approach has not been investigated exclusively in relation to GAs. There is a wide research on the dDE models which can be characterized on the basis of the neighborhood topology, the migration policy, the selection function and the replacement function.

In [12] the migration mechanism as well as the algorithmic parameters are adaptively coordinated according to a criterion based on genotypical diversity. An adaptive DE is executed on each subpopulation for a fixed number of generations. Then a migration process, based on a random connection topology, is started: each individual in each subpopulation can be probabilistically swapped with a randomly selected individual in a randomly chosen subpopulation (including the one containing the initial individual).

Tasoulis et. al [13] propose a dDE, named PDE, characterized by unidirectional ring topology, a selection function that picks up the individuals with the best performance and, with a given probability, send these individuals to the neighboring subpopulations. When the migration occurs, the migrating individuals substitute random individuals of the target subpopulations.

In Apolloni et al. [15] a distributed version, known as IBDDE, is presented: the migration policy is based on a probabilistic criterion depending on five parameters. The individuals to migrate are randomly selected and the individuals arriving from other islands replace randomly chosen local individuals only if the former ones are fitter. The topology is a unidirectional ring in which the individuals are exchanged with the nearest neighbors.

In De Falco et al. [22] a distributed version of DE, called DDE, has been proposed. It consists of a set of classical DE schemes, running in parallel, assigned to different processing elements arranged in a torus topology, in which each generic DE instance has four neighboring communicating subpopulations. The individual sent is the best one and it randomly replaces an individual in the neighboring subpopulation, except the local current best one.

In the paper by Ishimizu and Tagawa [17] a structured DE approach still based on the stepping–stone model is presented. Different network topologies, ranging from ring to torus and hypercube, are taken into account. The migration takes place every fixed number of generations and the exchange involves only the best individual which migrates towards only one of the adjacent subpopulations on the basis of the topological neighborhood and randomly replaces an individual, except the best one, in the receiving subpopulation.

An improved version of PDE algorithm which entails the employment of four different scale factor values within distributed differential evolution structures is advanced in [18]. The subpopulations are arranged according to a ring or a torus topology. Although proper choice of a scale factor scheme appears to be dependent on the distributed structure, any of the proposed simple schemes has proven to significantly improve upon the single scale factor distributed differential evolution algorithms.

In [25] a structured DE which uses a biological invasion inspired migration strategy is advanced. The subpopulations are displaced in a torus topology. During the migration the individuals with the fitness better than the average fitness in their subpopulation are sent to all the neighboring subpolulations and a replacement strategy is performed to keep unchanged the size of each subpopulation.

3 The Distributed Model

Our Distributed DE (DDE) algorithm is based on the classical coarse–grained approach to EAs [7] in which a collection of networked subpopulations cooperate in the solution of a problem by a migration operator. It consists in a locally–linked strategy, known as *stepping stone–model* [8], in which each DE instance is connected to a number of instances according to the connectivity degree of the topology beneath. Each subpopulation can communicate with the other ones only through its neighbours.

Decision must be taken for the migrant selection, i.e. the choice of the elements to be sent, and replacement, i.e the individuals to be replaced by the migrants. Different strategies can be devised: the migrants can be selected either according to fitness or randomly, and they might replace the worst individuals or substitute them only if better, or they might finally replace any individual (apart from the very best ones, of course) in the neighbouring subpopulation. Consistently with the biological events, it was noted that the number of migrants should not be high and the migration should occur after a period of stasis otherwise the subsearch in a subpopulation might be very perturbed by these continuously incoming elements [7,26].

This mechanism allows attaining both *exploitation* and *exploration*, which are basic features for a good search. Exploration means to wander through the search space so as to prevent premature convergence to local optima. Exploitation implies that one area is thoroughly examined, so that we can be confident to state whether this area is promising. In such a way, good solutions will spread within the network with successive diffusions, so more and more demes will try to sample that area (exploitation), and, on the other hand, there will exist at the same time clusters of subpopulations which will investigate different subareas of the search space (exploration). Therefore, a suitable percentage of migrants each subpopulation sends to its neighbours, called *Migration Rate* (M_R), and an appropriate exchange frequency between neighbouring subpopulations every M_I generations, named *Migration Interval*, are to be introduced to exploit at the best the potential of this cooperating stepping–stone model. A rigorous theoretical analysis that leads into new insights into the usefulness of migration, how information is propagated in island models, and how to set parameters such as the migration interval is reported in [27]. This study is corroborated by empirical results that investigate the robustness with respect to the choice of the migration interval and compare various migration topologies using statistical tests.

Within this general framework we have implemented a distributed version for DE, which consists of a set of classical DE schemes, running in parallel, assigned to different processing elements arranged in several topologies in which each generic DE instance has a different number of neighbouring communicating subpopulations.

4 Experiments

To investigate the influence of the network topologies in DDE we have compared their performance on a set of benchmark thirty–dimensional functions as defined in [28]. Namely, the unimodal functions F_1 and F_3, and among the multimodal, the basic functions F_6 and F_{10}, the expanded functions F_{13} and F_{14}, and the hybrid composition functions F_{16} and F_{22} have been taken into account. Among these, F_1, F_3, and F_6 are separable. As suggested in [29], throughout the experiments, the values for the DE parameters have been chosen as follows: *scale factor* ($F = 0.9$) for all the functions and the *crossover ratio* (CR) has been set to 0.1 for all the separable functions and 0.9 for all the other functions. The $DE/rand/1/bin$ [19] mutation mechanism has been used. As topologies a Ring, a bidirectional ring (Bring), an incomplete binary tree (IBtree), a Torus, a WK–recursive (WK), and a Hypercube, each constituted by a total of 16 nodes, have been investigated. Some of these topologies are outlined in Fig. 1.

The total population size has been chosen as 160, which results in sixteen subpopulations with 10 individuals. The number of generations has been set

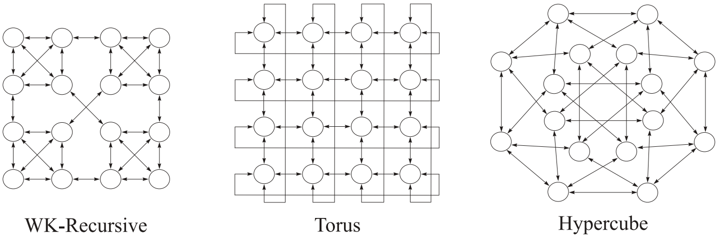

Fig. 1. The network topologies

Table 1. Best migration interval and related average final value for each problem

	Ring		Bring		IBtree		Torus		WK		Hypercube	
Problem	M_I	$\langle \phi_{b_f} \rangle$	M_I	$\langle \phi_{b_f} \rangle$	M_I	$\langle \phi_{b_f} \rangle$	M_I	$\langle \phi_{b_f} \rangle$	M_I	$\langle \phi_{b_f} \rangle$	M_I	$\langle \phi_{b_f} \rangle$
F_1	10	$5.68 \cdot 10^{-14}$	10	$3.18 \cdot 10^{-14}$	10	$4.32 \cdot 10^{-14}$	10	$4.09 \cdot 10^{-14}$	10	$4.09 \cdot 10^{-14}$	10	$5.00 \cdot 10^{-14}$
F_3	10	$4.85 \cdot 10^{+5}$	10	$3.62 \cdot 10^{+5}$	10	$3.62 \cdot 10^{+5}$	10	$3.40 \cdot 10^{+5}$	10	$2.89 \cdot 10^{+5}$	10	$6.41 \cdot 10^{+5}$
F_6	30	$4.37 \cdot 10^{+1}$	40	$3.66 \cdot 10^{+1}$	20	$4.67 \cdot 10^{+1}$	10	$3.74 \cdot 10^{+1}$	10	$3.53 \cdot 10^{+1}$	10	$7.47 \cdot 10^{+1}$
F_{10}	30	$5.07 \cdot 10^{+1}$	40	$5.10 \cdot 10^{+1}$	50	$5.40 \cdot 10^{+1}$	50	$5.34 \cdot 10^{+1}$	50	$5.31 \cdot 10^{+1}$	50	$5.84 \cdot 10^{+1}$
F_{13}	50	$2.17 \cdot 10^{0}$	50	$1.98 \cdot 10^{0}$	40	$2.15 \cdot 10^{0}$	50	$2.64 \cdot 10^{0}$	50	$2.43 \cdot 10^{0}$	50	$2.81 \cdot 10^{0}$
F_{14}	50	$1.24 \cdot 10^{+1}$	40	$1.26 \cdot 10^{+1}$	40	$1.27 \cdot 10^{+1}$	50	$1.26 \cdot 10^{+1}$	40	$1.26 \cdot 10^{+1}$	50	$1.27 \cdot 10^{+1}$
F_{16}	50	$9.21 \cdot 10^{+1}$	50	$9.36 \cdot 10^{+1}$	50	$9.93 \cdot 10^{+1}$	50	$9.81 \cdot 10^{+1}$	50	$9.86 \cdot 10^{+1}$	50	$9.39 \cdot 10^{+1}$
F_{22}	50	$8.68 \cdot 10^{+2}$	50	$8.75 \cdot 10^{+2}$	50	$8.77 \cdot 10^{+2}$	50	$8.83 \cdot 10^{+2}$	50	$8.84 \cdot 10^{+2}$	50	$8.81 \cdot 10^{-2}$

to $1,875$, so as to have a total number of fitness evaluations equal to $300,000$, following the rules widely used to face those testbeds, as for example in [29].

The parallel algorithm, which uses the Message Passing Interface is written in C language. All the experiments have been carried out on a Vega cluster constituted by 16 Pentium 4 processors with a frequency of 1.5 GHz and 512Mb of RAM, interconnected by a FastEthernet switch.

A first phase of our investigation has aimed at finding the best possible value for the migration interval M_I for each function and for each topology. We have considered a given range of possible values, i.e., 10, 20, 30, 40, and 50. For any such value 25 runs have been effected for each function and each topology, and the averages $\langle \phi_{b_f} \rangle$ of the best final fitness values over the 25 runs have been computed. Table 1 reports the best values of M_I, together with the corresponding values of $\langle \phi_{b_f} \rangle$.

Examination of the results shows that for the easiest functions F_1 and F_3 the best value for M_I is obtained at the lowest tested migration interval. For the most difficult problems the results are better and better as the migration interval increases, and this holds true until a given value for M_I is reached; after this value, the performance worsens more and more as M_I further increases.

4.1 Statistical Analysis

To compare the algorithms from a statistical point of view, a classical approach based on nonparametric statistical tests has been carried out, following [30]. To do so, the ControlTest package [31] has been used. It is a Java package developed to compute the rankings for these tests, and to carry out the related post–hoc procedures and the computation of the adjusted p–values.

Table 2. Average Rankings of the algorithms

Topology	Friedman	Aligned Friedman	Quade
Ring	3.000	24.125	3.083
Bring	2.313	17.938	2.667
IBtree	4.125	25.125	3.944
Torus	3.688	24.813	3.764
WK	2.563	18.688	2.125
Hypercube	5.313	36.313	5.417
statistic	14.286	6.822	4.148
p–value	0.014	0.234	0.005

The results for the one–to–all analysis are reported in the following. Table 2 contains the results of the Friedman, Aligned Friedman, and Quade tests in terms of average rankings obtained by all the topologies. The last two rows show the statistic and the p–value for each test, respectively. For Friedman and Aligned Friedman tests the statistic is distributed according to chi–square with 5 degrees of freedom, whereas for Quade test it is distributed according to F–distribution with 5 and 35 degrees of freedom.

In each of the three tests, the lower the value for an algorithm, the better the algorithm is. Bring turns out to be the best in two out of the three tests while WK is the best according to the Quade test. Among the other four topologies, their order is in all the tests the following: Ring is always the third best heuristic, Torus is the fourth, followed by ITree, and finally the Hypercube is the sixth.

Furthermore, with the aim to examine if some hypotheses of equivalence between the best performing algorithm and the other ones can be rejected, the complete statistical analysis based on the post–hoc procedures ideated by Holm, Hochberg, Hommel, Holland, Rom, Finner, and Li has been carried out following [30]. Moreover, the adjusted p–values have been computed by means of [31].

Table 3 reports the results of this analysis performed at a level of significance $\alpha = 0.05$. In this table the other algorithms are ranked in terms of distance from the best performing one, and each algorithm is compared against this latter to investigate whether or not the equivalence hypothesis can be rejected. For each algorithm each sub–table reports the z value, the unadjusted p–value, and the adjusted p–values according to the different post-hoc procedures. The variable z represents the test statistic for comparing the algorithms, and its definition depends on the main nonparametric test used. In [30] all the different definitions for z, corresponding to the different tests, are reported. The last row in each sub–table contains for each procedure the threshold value Th such that the procedure considered rejects those equivalence hypotheses that have an adjusted p–value lower than or equal to Th.

Summarizing the results of these tables, the equivalence hypothesis between WK and Bring cannot be rejected by any test and by any post–hoc procedure. The hypothesis of their equivalence to the Hypercube, instead, is rejected by all post–hoc procedures, and that with IBtree in many cases. Finally, their equivalence with Torus and Ring is always excluded by Li post–hoc procedure.

Table 3. Results of post–hoc procedures for Friedman(top), Aligned Friedman (center), and Quade (bottom) tests over all tools (at $\alpha = 0.05$)

i	Algorithm	$z = (R_0 - R_i)/SE$	p	Holm/Hochberg/Hommel	Holland	Rom	Finner	Li
5	Hypercube	3.207	0.001	0.010	0.010	0.010	0.010	0.011
4	IBtree	1.938	0.053	0.013	0.013	0.013	0.020	0.011
3	Torus	1.470	0.142	0.017	0.017	0.017	0.030	0.011
2	Ring	0.735	0.462	0.025	0.025	0.025	0.040	0.011
1	WK	0.267	0.789	0.050	0.050	0.050	0.050	0.050
Th				0.013/0.010/0.013	0.013	0.011	0.020	0.011

i	Algorithm	$z = (R_0 - R_i)/SE$	p	Holm/Hochberg/Hommel	Holland	Rom	Finner	Li
5	Hypercube	2.625	0.009	0.010	0.010	0.011	0.010	0.004
4	IBtree	1.027	0.305	0.013	0.013	0.013	0.020	0.004
3	Torus	0.982	0.326	0.017	0.017	0.017	0.030	0.004
2	Ring	0.884	0.377	0.025	0.025	0.025	0.040	0.004
1	WK	0.107	0.915	0.050	0.050	0.050	0.050	0.050
Th				0.013/0.010/0.013	0.013	0.011	0.020	0.004

i	Algorithm	$z = (R_0 - R_i)/SE$	p	Holm/Hochberg/Hommel	Holland	Rom	Finner	Li
5	Hypercube	1.983	0.047	0.010	0.010	0.010	0.010	0.013
4	IBtree	1.096	0.273	0.013	0.013	0.013	0.020	0.013
3	Torus	0.987	0.323	0.017	0.017	0.017	0.030	0.013
2	Ring	0.577	0.564	0.025	0.025	0.025	0.040	0.013
1	Bring	0.326	0.744	0.050	0.050	0.050	0.050	0.050
Th				0.010/—/0.010	0.010	—	0.010	0.013

4.2 Behavior of the Topologies

A very interesting remark is that the migration frequency corresponding to the best performance for any given topology has a strong relationship to the degree of difficulty of the problem: the simpler the problem the lower the value for M_I, the harder the problem the higher the value. This holds true for all the topologies and for all the problems. Just to give some examples, Fig. 2 shows four different situations. Namely, the top–left pane deals with the quite easy function F_3 for the bidirectional ring: the lower the value for M_I the better the performance. Top–right pane reports on the behavior of WK topology over F_6 function: this is a quite easy one, and same conclusions as before hold true. The bottom–left pane, instead shows the behavior over the more difficult F_{13} function: now the situation is reversed, and the higher the value for M_I the better the performance. Similarly, the bottom–right pane sketches the behavior of WK over the difficult F_{22} problem: same considerations as before hold true. This seems to imply that as the problem becomes more and more complex to solve, the demes should exchange individuals less frequently, probably because each deme needs now to more deeply perform exploitation.

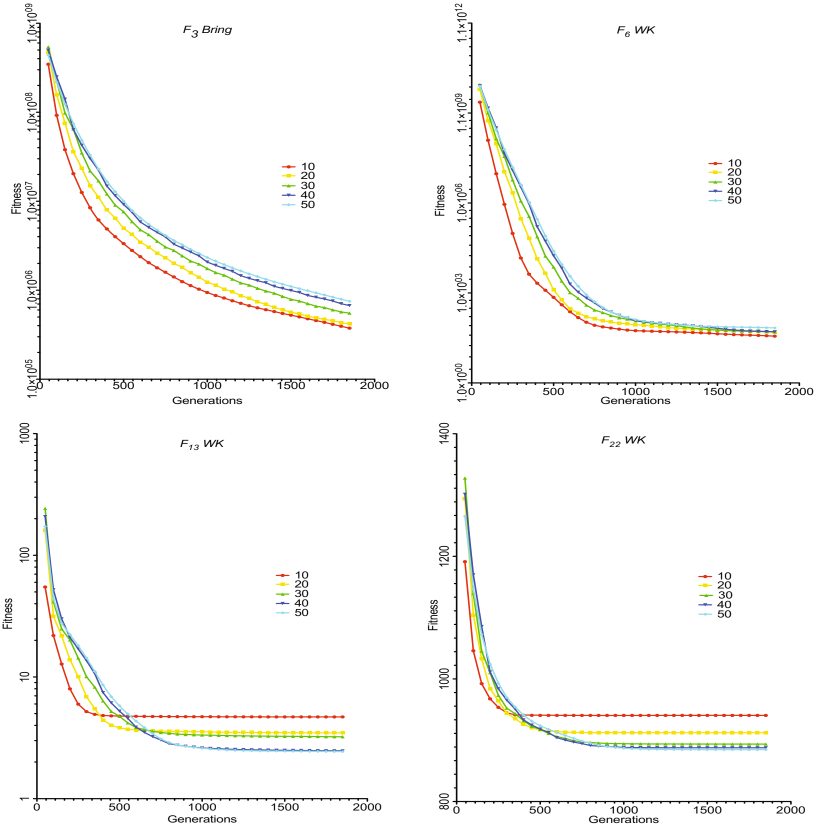

Fig. 2. A few examples of behavior of some topologies over some functions supporting the hypothesis that the harder a function, the higher the best value for M_I

A second feature worth noting is that WK and Torus topologies have faster convergence capability to suboptimal solutions than the other topologies. This takes place in general for any given function, and for any value of M_I. It is interesting to note that this holds true also in the circumstances in which these two topologies do not reach the best values at the end of the evolutions, rather they are overtaken by other topologies that start more slowly. Figure 3 shows this feature for four exemplary situations. Its top–left pane deals with F_3 function at $M_I = 30$, the top–right one reports on F_{10} at $M_I = 50$, the bottom–left one sketches the situation for F_{14} at $M_I = 50$, and finally the bottom–right pane shows F_{16} test case at $M_I = 50$. In all the cases Ring topology is the slowest. This feature could be profitably used whenever speed becomes of paramount importance in solving a problem: WK and Torus are very appealing, if a good suboptimal solution is needed in a very low amount of time.

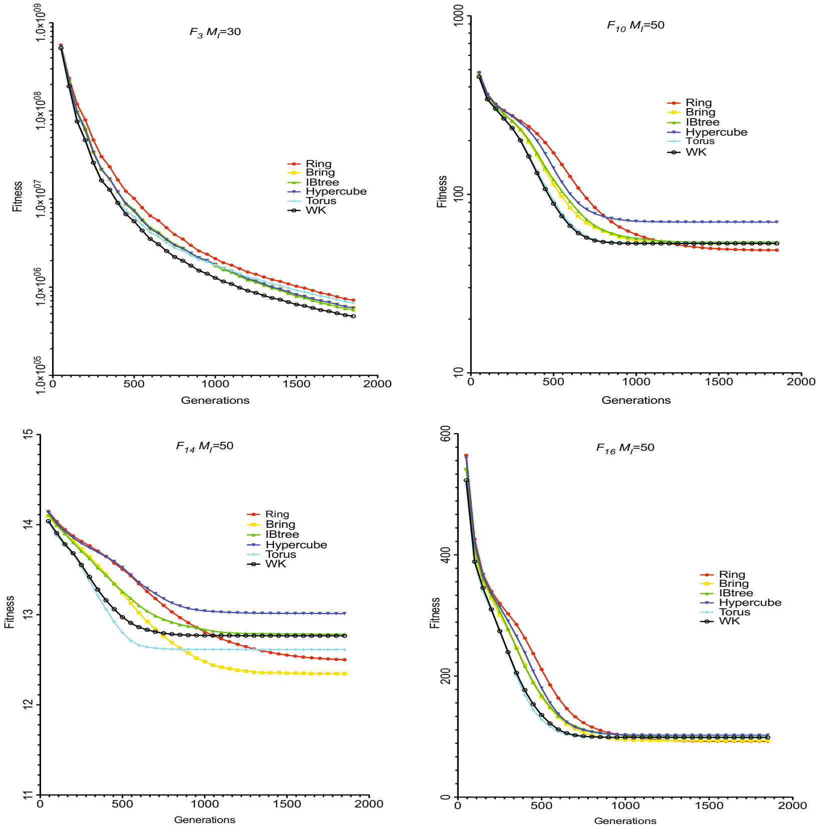

Fig. 3. Some examples of the faster convergence achieved by WK and Torus

5 Conclusions and Future Works

A distributed DE algorithm has been considered to evaluate the impact of the migration topologies on the stepping–stone model. The simulation results performed on a set of classical test functions and their statistical analysis have been shown to compare the performance of the different network topologies.

Future works will aim at carrying out a wider evaluation phase. This will be accomplished by performing sets of experiments with other DE operators, so as to ascertain that the performance are unchanged independently of the DE scheme chosen.

References

1. Holland, J.: Adaptation in natural and artificial systems. The University of Michigan Press, Ann Arbor (1975)
2. Schwefel, H.: Numerical optimization of computer models. Wiley & Sons (1981)

3. Goldberg, D.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Professional (1989)
4. Koza, J.: Genetic programming. MIT Press, Cambridge (1992)
5. Bäck, T., Fogel, D.B., Michalewicz, Z. (eds.): Handbook of Evolutionary Computation. Oxford University Press, Oxford (1997)
6. De Falco, I., Cioppa, D.A., Iazzetta, A., Tarantino, E.: An evolutionary approach for automatically extracting intelligible classification rules. Knowledge and Information Systems **7**, 179–201 (2005)
7. Cantú-Paz, E.: A summary of research on parallel genetic algorithms. Technical Report 95007, University of Illinois, Urbana-Champaign, USA (1995)
8. Mühlenbein, H.: In: Rawlins, G. (ed.) Foundations of Genetic Algorithms. Morgan Kaufmann, San Mateo (1991)
9. Tomassini, M.: Spatially structured evolutionary algorithms. Springer (2005)
10. Cantú-Paz, E.: Efficient and accurate parallel genetic algorithms, vol. 1. Kluwer Academic Publisher, Norwell (2000)
11. Alba, E., Tomassini, M.: Parallelism and evolutionary algorithms. IEEE Trans. on Evolutionary Computation **6**, 443–462 (2002)
12. Zaharie, D., Petcu, D.: Parallel implementation of multipopulation differential evolution. In: Proceedings of the Nato Advanced Research Workshop on Concurrent Information Processing and Computing, pp. 223–232. IOS Press (2003)
13. Tasoulis, D., Pavlidis, N., Plagianakos, V., Vrahatis, M.: Parallel differential evolution. Proceedings of the Congress on Evolutionary Computation. **2**, 2023–2029 (2004)
14. De Falco, I., Della Cioppa, A., Scafuri, U., Tarantino, E.: A distributed differential evolution approach for mapping in a grid environment. In: Proceedings of the Fifteenth EUROMICRO International Conference on Parallel, Distributed and Network-Based Processing, pp. 442–449. IEEE Press (2007)
15. Apolloni, J., Leguizamón, G., García-Nieto, J., Alba, E.: Island based distributed differential evolution: an experimental study on hybrid testbeds. In: Proceedings of the Eight International Conference on Hybrid Intelligent Systems, pp. 696–701. IEEE Press (2008)
16. Weber, M., Neri, F., Tirronen, V.: Distributed differential evolution with explorative-exploitative population families. Genetic Programming and Evolvable Machines **10**, 343–371 (2009)
17. Ishimizu, T., Tagawa, K.: A structured differential evolution for various network topologies. International Journal of Computers and Communications **4**, 2–8 (2010)
18. Weber, M., Neri, F., Tirronen, V.: A study on scale factor in distributed differential evolution. Information Sciences **18**, 2488–2511 (2011)
19. Price, K., Storn, R.: Differential evolution. Dr. Dobb's Journal **22**, 18–24 (1997)
20. Price, K., Storn, R.M., Lampinen, J.: Differential Evolution - A Practical Approach to Global Optmization. Springer (2005)
21. Nobakhti, A., Wang, H.: A simple self-adaptive differential evolution algorithm with application on the alstom gasifier. Applied Soft Computing **8**, 350–370 (2008)
22. De Falco, I., Della Cioppa, A., Maisto, D., Scafuri, U., Tarantino, E.: Satellite Image Registration by Distributed Differential Evolution. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS, vol. 4448, pp. 251–260. Springer, Heidelberg (2007)
23. Alba, E., Troya, J.: A survey of parallel distributed genetic algorithms. Complexity **4**, 31–52 (1999)

24. Alba, E., Luque, G.: Theoretical models of selection pressure for dEAs: topology influence. In: Proceedings of the IEEE International Conference on Evolutionary Computation, pp. 214–221 (2005)
25. De Falco, I., Cioppa, D.A., Maisto, D., Scafuri, U., Tarantino, E.: Biological invasion-inspired migration in distributed evolutionary algorithms. Information Sciences **207**, 50–65 (2012)
26. Skolicki, K., De Jong, K.: The influence of migration sizes and intervals on island models. In: Proceedings of the Conference of Genetic and Evolutionary Computation, Association for Computing Machinery Inc, pp. 1295–1302. ACM (2005)
27. Lässig, J., Sudholt, D.: Design and analysis of migration in parallel evolutionary algorithms. Soft Computing **17**, 1121–1144 (2013)
28. Suganthan, P., Hansen, N., Liang, J., Deb, K., Chen, Y., Auger, A., Tiwari, S.: Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report 201212, Zhengzhou University, China and Nanyang Technological University, Singapore (2005)
29. Rönkkönen, J., Kukkonen, S., Price, K.: Real-parameter optimization with differential evolution. In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 1, pp. 506–513. IEEE (2005)
30. Derrac, J., García, S., Molina, D., Herrera, F.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation **1**, 3–18 (2011)
31. García, S., Fernández, A., Luengo, J., Herrera, F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences **180**, 2044–2064 (2010)