

Visualising Evolutionary Pathways in Real-World Search Spaces

Paul Layzell

Hewlett-Packard Laboratories
Bristol, UK
Paul_layzell@hp.com

Abstract

A new method of visualising search spaces on a two-dimensional surface is presented, and applied to real search spaces created by the artificial evolution of electronic circuits. The visualisation method is inspired by work on search space neutrality, and its links with evolvability, both of which are discussed. The search space is transformed into a connected graph. Vertices on the graph are connected if their Hamming distance from each other is one, and a clustering algorithm reveals important inter-cluster pathways that can be reached by single mutations.

Introduction

The parametric search spaces to which evolutionary algorithms (EAs) are applied are very often represented graphically as two- or three-dimensional lines or surfaces. These plots contain peaks and troughs of varying height that represent the suitability or ‘fitness’ of a set of particular parameter values for a given task (eg. [11]). This form of visualization is useful in that it aids the understanding of certain concepts in evolutionary computation, such as ‘ruggedness’ or ‘local optima’. However, it is often misleading for the real search spaces found in the majority of applications, which possess very high dimensionality. Of course, it is impossible to map high-dimensional spaces – where each point may have several hundred adjacent neighbours – onto a 2-dimensional page without losing much of its structure. Hence, in designing such a mapping, salient characteristics of the space must be carefully picked out, and the rest discarded.

This paper presents a technique for visualizing the pathways within a search space that an evolutionary search is likely to follow. It is inspired by the concept of *neutrality* – single-bit mutations that have no effect on fitness – and its links with evolvability. The technique transforms a set of points in a given search space into vertices in a connected graph, which is displayed on a two dimensional surface. A clustering algorithm is then applied to group together sets of highly connected vertices. This has the effect of revealing pathways away from medium-fitness regions that may otherwise be considered as local optima. The more such pathways, the more amenable to evolutionary search the space can be expected to be. The technique was originally conceived to assist with the

design of evolvable architectures for electronic circuitry, and will be applied here to two examples from evolutionary electronics. However, since it does not rely on any domain-specific knowledge, it should be applicable to virtually any evolutionary task. No grand claims are made regarding the validity of this visualisation technique as a new metric for evolvability – indeed its drawbacks will be highlighted in the text. It does however represent an original way of graphically revealing important aspects of search space structure.

The next section gives a brief description of search space neutrality, and why designing an evolutionary system with high neutrality is a useful alternative to relying on domain-specific knowledge to guide the search. An example is then given showing why visualising the search space as a connected graph can aid the understanding of its evolvability. Next the technique for producing the visualisation is described in detail, with particular emphasis on the clustering process. Finally, the technique is applied to electronic search spaces and some comments made about their respective evolvability.

Rationale

Search space neutrality is rapidly finding favour as a means by which an evolutionary search can progress beyond local regions of sub-optimal fitness [1][11][4][13]. Neutral mutations are changes to an individual genotype that have no effect on the fitness of the corresponding phenotype. Paths accessible by neutral mutations can percolate through vast volumes of high-dimensional genotype space. Hence, a population of individuals can undergo a neutral drift along such pathways unencumbered by local optima, until regions of higher fitness are encountered. Consequently, many authors consider the degree of neutrality a search space possesses as having a positive correlation with its *evolvability*, particularly for those types of EA for which mutation is the primary operator, such as Evolution Strategies (ES) [10] or Harvey’s SAGA [3].

Assessing the degree of neutrality inherent in a given search space is difficult, because there are many factors determining whether or not a mutation can be expected to be neutral. For example, neutral mutations can occur in ‘junk DNA’ – parts of the genotype that do not form an active part of the evolving solution. (For example, in the

context of evolutionary robotics, the speed of a motor may be genetically specified. But if the motor is disabled elsewhere in the genotype then modifications to its speed will have no effect). However, the same part may be incorporated into the solution at a later stage, resulting in mutations of that part no longer being neutral. Mutations occurring in non-junk DNA can also be neutral if either the behaviour of the phenotype is unchanged, or if the evaluation method apportions equal fitness to different behaviours.

The visualisation method documented here derives from a study of neutrality and near-neutrality inherent in the search spaces of electronic circuits. The original intention was to ascertain visually which of a number of interconnection architectures (the reconfigurable wiring between circuit primitives) was the most appropriate for circuit design by artificial evolution. In the field of HE, the question of interconnection architecture remains open. Usually it is designed using conventional electronic engineering knowledge, (for example, limiting the connectivity between component pins, forbidding positive feedback etc.). But some authors feel that the use of such domain-specific knowledge can prejudice the search in ways incompatible with evolution, if its effect on the search space is not taken into account [12]. The visualisation method in this paper was therefore conceived as a starting point towards alternative metrics, in this case, those interconnection architectures promoting search spaces with high neutrality [8].

Visualising Evolutionary Pathways

Illustrating the concept

The method attempts to represent a portion of the search space graphically in a manner similar to figure 1, a hypothetical search space in which individuals above a certain fitness threshold are represented as vertices on a connected graph. Vertices are connected if the individuals they represent are genetically separated from each other by one unit of Hamming distance. The vertices are also shaded according to a pre-determined fitness range. In this example, low-fitness individuals are black, suboptimal ones are grey, and optimal ones are white. A clustering algorithm is then applied to the graph to create spatially distinct regions of highly connected vertex sets. The fitness threshold is necessary to ensure that the graph has structure. Clearly if all individuals in the search space were included in the graph, then each vertex would have an equal number of one-Hamming-distance neighbours, and the clustering algorithm would fail. The threshold is set low enough that individuals of that fitness or above should be found reasonably quickly by random walk. Note that while this method is inspired by work on neutrality, it utilises *ranges* of fitness rather than individuals of *equal* fitness, hence it does not attempt to reveal truly neutral networks.

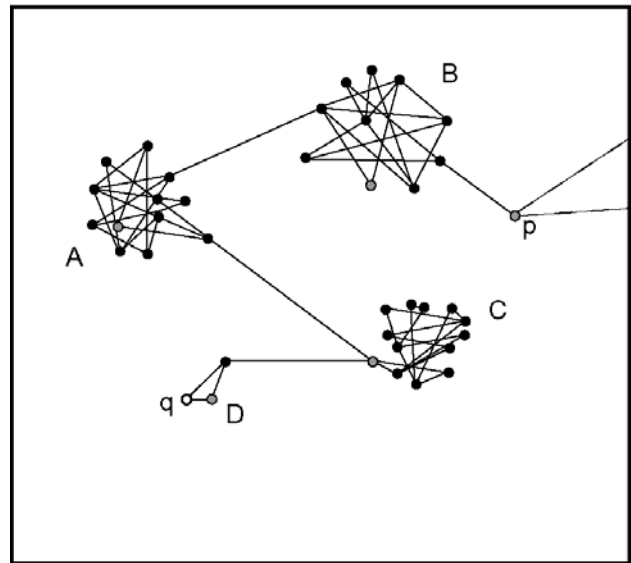


Figure 1. Hypothetical search space represented as a connected graph. Vertices on the graph represent individual genotypes, shaded according to their fitness (see text). Where two genotypes are separated by a Hamming-distance of one, their vertices are connected

The basic interpretation for such a visualisation is as follows: If an evolutionary search contains an individual lying in a cluster, then other individuals in the same cluster should be easily located by the search due to their short Hamming distance. The cluster can hence be regarded as a single individual. We would like to determine the likelihood of evolution progressing to another cluster – one which contains individuals of higher fitness, but which are on average of large Hamming distance from the original one. Even though the average Hamming distance between individuals in one cluster and those in another is large, there may be one or more pairs (with one individual in each cluster) separated by a single unit, as shown in figure 1. Where this is the case, the evolutionary search is likely to extend to the new cluster, but if it is not, a more fortuitous double mutation is required. The purpose of representing the search space in this way is therefore to render such inter-cluster pairs of individuals as apparent as possible, as they represent a path that the search can follow. Of course there are other factors affecting the search's direction, such as suboptimal individuals existing within a cluster of poor ones, but this does not hinder the *exploration* of new regions one or two mutations away; if new clusters accessible through inter-cluster paths contain similar suboptima, then the search is likely to find them.

To illustrate these points, consider a small evolutionary population lying on cluster A in figure 1. The target is the optimum individual labelled **q**, located in cluster D. There exist two inter-cluster paths by which members of the population can travel with a single-bit mutation either to cluster B or cluster C. Clearly C is the preferred choice,

from which the optimum is easily reachable. If the population migrates to **B**, a path still exists back to **A**, so all is not lost. However it may migrate further to the suboptimum **p**, lying in a cluster disconnected from **A**, **C**, or **D**. The probability of backtracking to **A** is now reduced, especially if **p** is of significantly higher fitness than any suboptima yet encountered. If this search space had contained more inter-cluster paths, **D** may have been reached by another route, implying a space more amenable to evolutionary search.

Clustering the vertices to reveal structure

Constructing the connected graph from a selection of individual points in the search space in the manner described above is unproblematic. The only difficult phase of the visualisation process is the clustering phase. The clustering algorithm used in this preliminary study is now described.

The clustering method used is a distributed algorithm, which originates from the findings of entomologists who, on observing societies of ants, have remarked that larvae and food are not scattered randomly about the nest, but are sorted into homogenous piles. Deneubourg et al. [2] proposed a behavioural model where the spatial structure of the nest emerges as a result of simple, local interactions without the need for any centralised control or global representation of the environment. In Deneubourg's model, the environment is a two-dimensional grid upon which is scattered a set of objects, each having random initial positions, and comparable with each other by an equivalence relationship. Each ant is modelled as an agent which is able to move on the grid, and displace the objects according to probabilistic rules using only local environmental information. The combined actions of a set of these agents lead to the grouping of objects which are comparable according to a measure of dissimilarity [9]. It forms one or more spatial groups such that similar objects belong to the same group, and dissimilar ones belong to different groups, each group being spatially distant from each other. This approach was later extended to work with vertices of connected graphs by defining a graph dissimilarity measure such that vertices having a number of common neighbours and few distinct neighbours share the same group [5]. This measure is suitable for unweighted, connected graphs of the form $G=(V,E)$ where V is the set of vertices and E the set of edges, such that each edge connects two vertices.

Let $\rho(v_i)$ denote the set of vertices of V which are adjacent to the vertex v_i , and include v_i :

$$\rho(v_i) = \{v_j \in V; \{v_i, v_j\} \in E\} \cup \{v_i\}$$

then the dissimilarity between two vertices v_i and v_j is given by:

$$d(v_i, v_j) = \frac{|\rho(v_i) \Delta \rho(v_j)|}{|\rho(v_i)| + |\rho(v_j)|}$$

where Δ designates the

symmetric difference (union minus intersection).

Initially, vertices and agents are laid out at random positions on the grid. Agents are able to move around the grid, pick up, carry, or drop vertices. However they are not permitted to move onto a grid position already occupied by another agent, nor are they allowed to drop a vertex onto a position occupied by another vertex. At the grid boundaries, agents are reflected. Each agent has a short-term memory containing the last m vertices it has carried and their new positions at the time of dropping.

The algorithm then proceeds in discrete time steps, t . At each t , an agent is selected at random. If a vertex lies on its position the agent can pick it up, likewise it can drop a vertex at that position if it is currently carrying one. If the agent is not carrying a vertex, it then moves r grid positions in a random direction, otherwise it moves r grid positions in a probabilistic manner towards the position of the most similar vertex in its memory to the one it is carrying. The probability $p_{pick}(v_i)$ that an agent will pick up a vertex v_i increases the more v_i is isolated, i.e. where the number of similar vertices in the immediate neighbourhood is small. By contrast, the probability $p_{drop}(v_i)$ that an agent will drop a vertex v_i increases with the number of similar vertices in the immediate neighbourhood. These probabilities are defined by

$$p_{pick}(v_i) = \left(\frac{k_p}{k_p + f(v_i)} \right)^2$$

and

$$p_{drop}(v_i) = \left(\frac{f(v_i)}{k_d + f(v_i)} \right)^2$$

where k_p and k_d are constants. The local density function f represents an estimation of the density of similar vertices in the v_i locality, defined here by an area Σ of $\sigma \times \sigma$ grid elements in which v_i lies at the centre:

$$f(v_i) = \max \left\{ 0, \frac{1}{\sigma^2} \sum_{v_j \in \Sigma} \left(1 - \frac{d(v_i, v_j)}{\alpha} \right) \right\}$$

where α is a scaling factor chosen empirically for optimal cluster separation. The performance of this algorithm has been measured with standard classes of connected graphs with up to 500 vertices, and compared with other clustering algorithms [5][6]. For more details the reader is referred to the citations above.

When applied to search space visualisation, this algorithm has the advantage that the number of clusters need not be specified *a priori*. However, there is no reason why the dissimilarity measure detailed here may not be adapted for use with other clustering algorithms.

Application to real search spaces

The visualisation technique is now applied to two real search spaces created by prior work in Hardware Evolution (HE), in which both circuit simulators and physical, reconfigurable hardware were used to evolve electronic inverter circuits with bipolar transistors [8]. In general, this research requires search spaces of the order 2^{100} to 2^{1000} in size, but by careful limitation of the interconnection architecture and minimal use of component primitives, they can be reduced to 2^8 to 2^{16} , sufficient to allow an exhaustive search. Note that although small, these are real electronic spaces and exhibit all the characteristics of the much larger spaces described above: The necessity of correct power configuration introduces epistasis into the genotype, and the circuit noise levels are not constant, but vary according to which parts are earthed (connected to 0V).

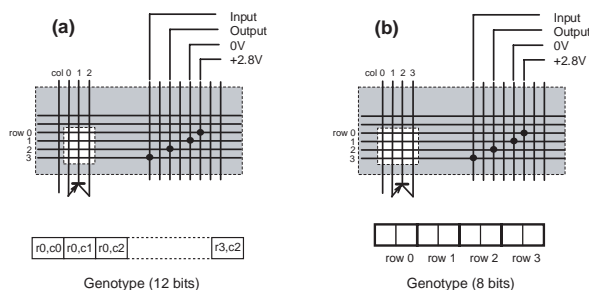


Figure 2. Two different genetic encodings for the evolution of an electronic circuit. Each is equivalent to a particular interconnection architecture (see text).

Figure 2 illustrates the evolvable architecture used for experiments with minimal search space size, and shows details of two different genetic encodings. The architecture is as follows: a matrix of wires is connected to electronic components such as transistors which are used as basic evolutionary primitives. The matrix is also supplied with input/output and power connections. Each row/column intersection within the matrix contains a programmable switch so that the connection between any row and any column may be controlled genetically. In previous examples of this system, switch matrices with up to 48 rows and columns, and up to ten transistors have been reported (eg.[7]). In the highly constrained versions of figure 2, only one transistor is allowed, and only the highlighted areas (white squares) are under genetic control, the rest being fixed open or closed. Closed switches are indicated on figure 2 by a black dot.

It is important to note that in physical implementations of this evolvable hardware, the programmable switches possess characteristics such as resistance and capacitance which affect – and are often exploited by – the evolving circuit [8][12]. Even when software simulators are used,

these characteristics are incorporated into the switch model; only more subtle characteristics such as inductance are generally ignored. Hence, the range of circuit behaviours achievable is larger than might be expected from different wiring configurations of a single primitive.

The two genetic encodings of figure 2 are analogous to different physical interconnection architectures. In (a), each of the 12 possible switches in the white highlighted area is directly mapped to one bit of the genotype, with bit values 1 & 0 specifying the on/off state of that switch. This will be referred to as the directly-mapped encoding. In (b) one switch per row is always closed and the rest open at any one time. Hence the genotype is divided into 2-bit addresses specifying by column, the closed switch for each row. An electronic equivalent of this mapping could be realised with rotary switches, the programmable versions of which are known as multiplexers. Hence it will be referred to as the multiplex-mapped encoding.

The task

The task was to evolve a digital inverter, or NOT gate. To evaluate candidate circuits, a series of 10 test inputs containing 5 1s and 5 0s (logical Highs (+5V) and Lows (+0V)) was applied sequentially to the input in alternating order. For each test input, the output voltage was measured twice (immediately on applying the input, then after a delay of 1ms) and summed. Fitness was scored as follows:

$$f = \frac{1}{10} \left(\sum_{t \in S_L} v_t - \sum_{t \in S_H} v_t \right)$$

where t signifies the test input number, S_H and S_L the set of High and Low test inputs respectively, and v_t ($t=1,2..10$) the circuit's summed output voltage for test number t . Fitness f is only ever positive when the circuit inverts, that is, when the output voltage for a Low input is higher than that for a High input. Circuits whose output is constant (for example, where the input is disconnected) score zero, and circuits which reflect the input state (for example, where input is connected directly to the output) yield negative scores.

An exhaustive search was conducted for both direct and multiplex mappings, using the above fitness function. Fitness evaluation was carried out using circuit simulators in preference to physical hardware so that the results would not be distorted by measurement or circuit noise. The resulting search spaces can now be displayed visually. Note that search spaces very similar to the two featured here have been studied in depth with a number of different metrics for evolvability [11].

Results

Figure 3 shows the search space for the direct mapping, and figure 4 the space for the multiplex mapping. The same shapes and colours are used in both of the figures to

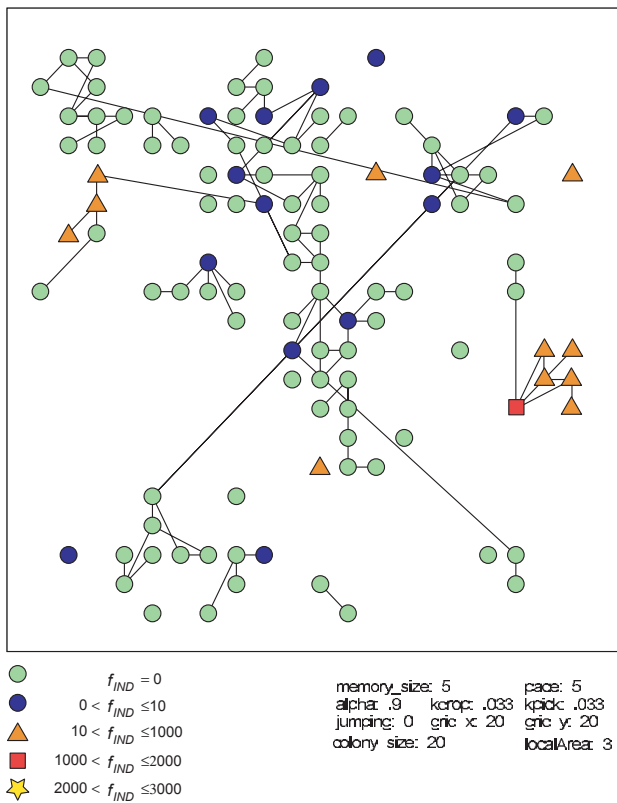


Figure 3. Connected graph visualisation of the directly-mapped search space of inverters. Directly below the graph are the parameter values used for the clustering algorithm (see text)

distinguish individuals of fitness f_{IND} lying within a given range. The fitness threshold in both cases is set at zero; only individuals of zero fitness (constant-output circuits) or better have been selected for display. The vertices shown as light circles represent exclusively individuals of zero fitness, hence sets of connected ones are true neutral networks.

Referring to figure 3 (direct mapping), we see several isolated clusters and few inter-cluster pathways. Some vertices – a few of which are suboptimal – are completely isolated, meaning that they can only be reached by single mutations of individuals with fitness less than zero. The single optimum (square, fitness 1205) lies in a distinct cluster centre-right in the diagram, which has *no* inter-cluster pathways. We can therefore speculate that an evolutionary search is unlikely to find it, being prone to wander around the suboptima (triangles) that appear in a few connected clusters. The search space created by multiplex mapping has an entirely different structure. Here we see two main clusters, predominantly individuals of zero fitness, with multiple inter-cluster paths. The optimum individual is at the top of the right-hand cluster (star, fitness 2274). Although the other cluster contains suboptima, the abundance of inter-cluster paths, and the lack

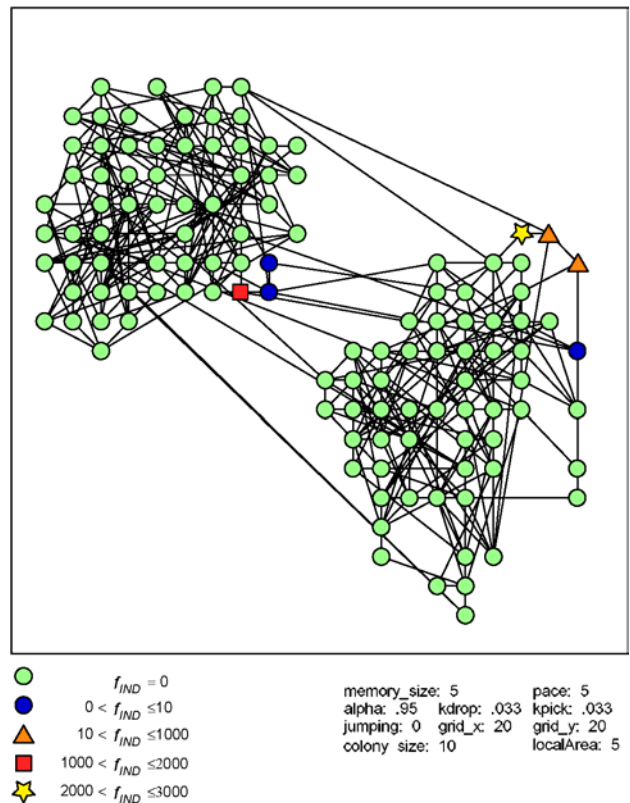


Figure 4. Visualisation of the multiplex-mapped inverter space

of isolated vertices and clusters implies far less difficulty in locating the optimum.

Conclusion

In this paper, I have shown how pathways in small evolutionary search spaces can be visualised by transforming a subset of the space into a connected graph and applying a clustering algorithm to reveal its structure. This technique has been applied to two different encodings for a real-world problem, and has revealed large differences in the nature of their corresponding search spaces. The results suggest that the multiplex-mapped encoding should be more evolvable than the directly-mapped encoding, and this is indeed the case – multiplex mapping has consistently proved much more evolvable than direct mapping for more typical large search spaces, as well as the tiny ones documented here [8]. However, as a sole indicator of evolvability, this visualisation technique has severe drawbacks: Applying the technique to the vast search spaces generated by more typical genotype lengths is currently impossible. It could be applied to sample portions, but some degree of homogeneity in the space would have to be assumed. Future work is geared towards applying the visualisation technique to sample regions, centred perhaps around sticking points observed during

evolutionary runs. It is important to re-iterate that there are many factors that influence evolvability in addition to those highlighted here. Nonetheless, this approach is a promising means of revealing structure in real search spaces, and the sort of routes an evolutionary search might take through them.

References

- [1] Barnett, L. (1997). Tangled webs. evolutionary dynamics on fitness landscapes with neutrality. Master's thesis, School of Cognitive and Computing Sciences (COGS), University of Sussex, UK.
- [2] Deneubourg, J.; Goss, S.; Franks, N.; Sendova Franks, A.; Detrain, C.; and Chretien, L. (1990). The dynamics of collective sorting: robot-like ants and ant-like robots. In 1st Int. Conf. on Simulation of Adaptive Behaviour: From Animals to Animats, pp. 356-363
- [3] Harvey, I. (1992b). Species Adaptation Genetic Algorithms: A basis for a continuing SAGA. In Varela, F. J., & Bourgine, P. (Eds.), Towards a Practice of Autonomous Systems: Proc. 1st Eur. Conf. on Artificial Life, pp. 346-354. MIT Press.
- [4] Kimura, M. (1983). *The Neutral Theory of Molecular Evolution*. Cambridge University Press, New York.
- [5] Kuntz, P., Layzell, P., & Snyers, D. (1997). A colony of ant-like agents for partitioning in VLSI technology. In Husbands, P., & Harvey, I. (Eds.), Proc. 4th European Conf. on Artificial Life (ECAL'97), pp. 417-424. MIT Press.
- [6] Kuntz, P., Snyers, D., & Layzell, P. (1999). A stochastic heuristic for visualising graph clusters in a bi-dimensional space. *Journal of Heuristics*, 5(3), 327-352.
- [7] Layzell, P. (1998b). A New Research Tool for Intrinsic Hardware Evolution. In Sipper, M., Mange, D., & P'erez-Urbe, A. (Eds.), Proc. 2nd Int. Conf. on Evolvable Systems (ICES'98), Vol. 1478 of LNCS, pp. 47-56. Springer-Verlag.
- [8] Layzell, P. (2001). Hardware Evolution: On the Nature of Artificially Evolved Electronic Circuits. Ph.D. thesis, School of Cognitive and Computing Sciences (COGS), University of Sussex, UK.
- [9] Lumer, E. and Faieta, B. (1994) Diversity and adaptation in populations of clustering ants. In Proc. of Third Conf. on Simulation of Adaptive Behaviour. pp. 499-508. MIT Press.
- [10] Schwefel, H.-P., & Rudolph, G. (1995). Contemporary evolution strategies. In Moran, F., et al. (Eds.), Advances in Artificial Life: Proc. 3rd Eur. Conf. on Artificial Life, Vol. 929 of LNAI, pp. 893-907. Springer-Verlag.
- [11] Smith, T.; Husbands, P.; Layzell, P.; and O'Shea, M. (2002). Fitness Landscapes and Evolvability. *Evolutionary Computation* 10(1) :1-34.
- [12] Thompson, A. (1998). *Hardware Evolution: Automatic design of electronic circuits in reconfigurable hardware by artificial evolution*. Distinguished dissertation series. Springer-Verlag.
- [13] Vassilev, V., & Miller, J. (2000). The advantages of landscape neutrality in digital circuit evolution. In Miller, J., Thompson, A., Thomson, P., & Fogarty, T. (Eds.), Proc. 3rd Int. Conf. On Evolvable Systems (ICES2000), Vol. 1801 of LNCS, pp. 252-263. Springer.