Towards Optimal Neuronal Wiring through Estimation of Distribution Algorithms

Laura Anton-Sanchez
Computational Intelligence Group, Departamento de Inteligencia Artificial
Universidad Politécnica de Madrid, Spain
l.anton-sanchez@upm.es

Concha Bielza
Computational Intelligence Group, Departamento de Inteligencia Artificial
Universidad Politécnica de Madrid, Spain
mcbielza@fi.upm.es

Pedro Larrañaga
Computational Intelligence Group, Departamento de Inteligencia Artificial
Universidad Politécnica de Madrid, Spain
pedro.larranaga@fi.upm.es

ABSTRACT

One of the greatest challenges of our time is to understand brain functions. Our goal is to study the existence of an optimal neuronal design, defined as the one that has a minimum total wiring. Many researchers have studied the problem of optimal wiring in neuronal trees. Here we propose a new approach. We start from point clouds formed by the branching points of real neuronal trees and we search for the optimal forest from these point clouds. To do this, we formalize the problem as a forest of degree constrained minimum spanning trees (DCMST). Since the DCMST problem is NP-hard, we will try to solve it using estimation of distribution algorithms, particularly in permutation domains.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Probabilistic algorithms; I.2.8 [**Artificial Intelligence**]: Metrics—*Problem Solving, Control Methods, and Search-Heuristic methods*

General Terms

Algorithms, Performance, Theory

Keywords

Estimation of distribution algorithms (EDAs); permutation problems; optimal wiring; neuronal trees; forest; degree constrained minimum spanning tree (DCMST)

1. NEUROSCIENCE AND OPTIMAL WIRING ANALYSIS

Neuronal circuits are composed of a large variety of branched structures (axons and dendrites). A typical neuron, see Figure 1[1], consists of a cell body or soma that contains the cell nucleus, dendrites (a large number of thin structures that arise from the soma branching multiple times, giving rise to complex 'dendritic trees'), and the axon (a special cellular extension that arises from the soma and ends with additional filaments that connect with the dendrites of other neurons).

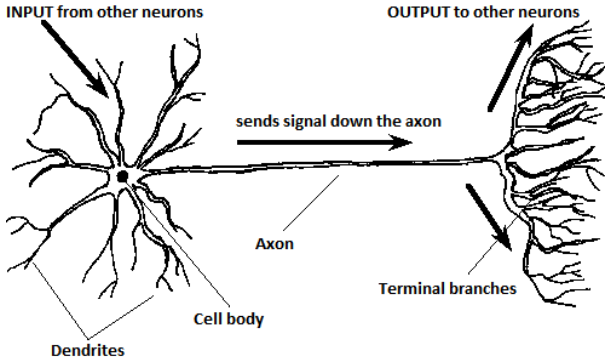

Figure 1: A typical neuron

Evolution has optimized brain design over millions of years, both to maximize its functionality and minimize its maintenance costs. That is, optimization theory may be used to study diverse features about brain design. Here we aim to explain that neuronal trees optimize brain connectivity in terms of wire length, a concept closely related to the time of signal transmission. Many studies have been performed on the study of a possible optimal wiring in neuronal trees. For example in [3] an expression of dendritic arborizations as locally optimized graphs is studied, and in [4] the authors develop a general quantitative theory relating the total length of dendritic wiring to the number of branch points and synapses.

In our approach, we start from point clouds formed by the branching points of real neuronal trees and we search for the optimal trees from these point clouds. Classical well-known algorithms, Prim [11] and Kruskal [8] algorithms, exist for building a minimum spanning tree (MST). However, if we want to build a degree constrained minimum spanning tree (DCMST), that is, an MST with constraints on the number of edges that can be incident to each node, the problem becomes NP-hard [5]. Our problem is even more complex because, in general, we will not want to obtain a single DCMST

[1]Figure from `http://www.macalester.edu`

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.
Copyright 2013 ACM 978-1-4503-1964-5/13/07 ...$15.00.

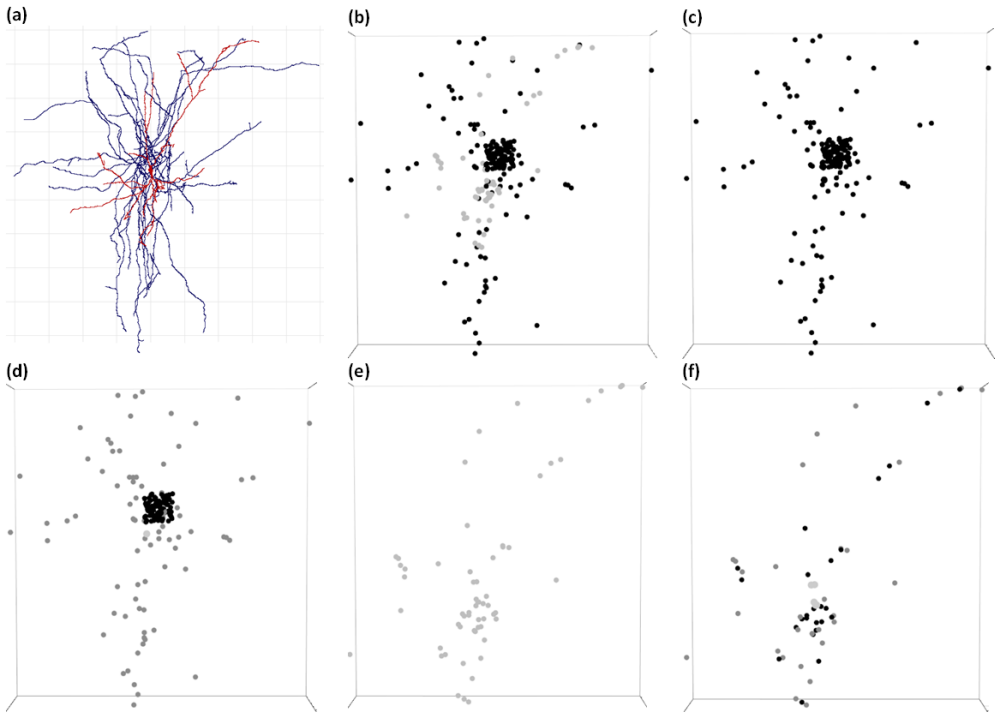

Figure 2: (a) A real neuron with its axon (blue) and dendrites (red). (b) Point cloud extracted from the axon (black) and dendrites (grey). (c) Axon point cloud. (e) Dendrites point cloud. (d)(f) Point clouds showing terminal points and roots (grey), forced in their roles, and intermediate points (black)

but a forest of them, and we also force some nodes to have specific roles in the trees.

Due to DCMST problem complexity, a number of heuristics have been applied in the literature. For example, in [1] an ant-based algorithm is proposed, tabu search is used in [13], and in [7] simulated annealing and genetic algorithms are compared. We aim to obtain our forest of DCMST using estimation of distribution algorithms (EDAs) through a novel permutation representation.

In neuroscience, when studying neuronal trees, we always work with binary trees (multifurcations barely exist), so we face an NP-hard problem where additional constraints arise. Terminal nodes of neuronal trees are connection points with other neurons; therefore, we are interested in maintaining these terminal nodes of real trees in the calculated structure. Something similar happens with the roots of real neuronal trees. Roots arise from the soma, and they are the points from which the transmission of signals to other neurons starts. For this reason, we also impose that roots of calculated trees are the same as in real trees.

A typical neuron has an axon, with its corresponding branched structure, and several dendritic trees. Figure 2(a) shows a real neuron and Figure 2(b) the point cloud formed by the roots, branching points and terminal points of its axon (black points) and dendrites (grey points). Inputs to nerve cells are usually via dendrites while axons carry output signals. Therefore, we will look at two point clouds individually.

On the one hand, we will study the point cloud formed by the root of the axon, its branching points and its terminal points, Figure 2(c), from which we will build a single binary MST keeping the root and the leaves. Then, we will compare the DCMST with the real axon. Figure 2(d) shows intermediate points of the axon in black, while the root and terminal points (points that we keep in the calculated tree) are shown in grey.

On the other hand, we will work with a point cloud consisting of all dendritic tree roots, all their intermediate points and all their terminal points. Figure 2(e) shows the point cloud formed by the five dendritic trees of this neuron. By keeping the roots, the number of constructed DCMST will be equal to the number of dendritic trees in the real neuron. As always, we will keep the leaves but, in this case, we allow mixing intermediate points from different dendritic trees. Figure 2(f) shows these intermediate points in black, while terminal points and roots are in grey. Note that if we wanted to consider each dendritic tree separately, we should build a point cloud for each of these trees as in the case of the axon where we build a single DCMST.

For the proposed problem, we have already started to work with data from six neurons (the neuron shown in Figure 2 is one of them) reconstructed by neuroscience experts at the Laboratorio Cajal de Circuitos Corticales, Madrid, Spain. We have very different types of neurons, both mouse and human neurons from distinct cortical areas and layers, with which we intend to do a major study.

1.1 Optimal Neuronal Wiring Problem as a Permutation-Based Optimization Problem

We propose to solve our problem through a permutation representation. A permutation is understood as a vector $\sigma = (\sigma_1, ..., \sigma_l)$ of the indexes $1, ..., l$ such as $\sigma_k \neq \sigma_s$ for

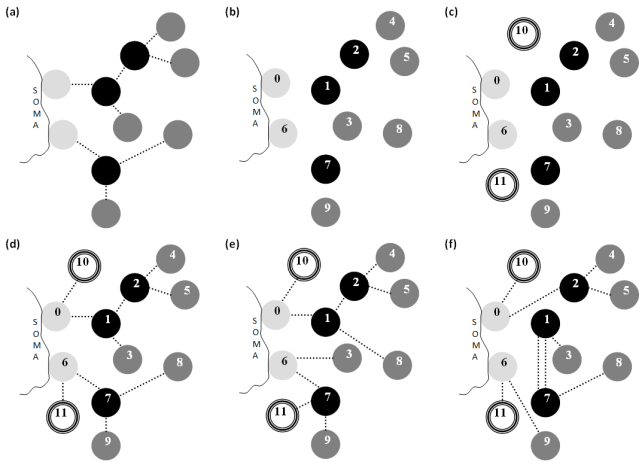

Figure 3: An example of permutation representation for two trees. (a) Two real trees. (b) Numbered point cloud. (c) Two necessary dummy nodes are added. (d) Individual equivalent to real trees, permutation $(1, 3, 4, 5, 6, 2, 7, 9, 10, 8)$**. (e) Example of correct individual, permutation** $(2, 4, 7, 5, 6, 1, 8, 3, 10, 9)$**. (f) Example of invalid individual because it has a cycle, permutation** $(9, 2, 4, 6, 5, 1, 3, 10, 7, 8)$

all $k \neq s$. We say that index s is in position k in σ when $\sigma_k = s$.

To understand this representation, consider the example in Figure 3. Figure 3(a) shows the roots (light grey), branching points (black) and terminal points (dark grey) of two small real dendritic trees of the neuron in Figure 2(a).

We enumerate the nodes, from 0 to 5 for the first tree and from 6 to 9 for the second tree, forming a single point cloud, Figure 3(b). Let's assume that n is the number of DCMSTs we wish to build ($n = 2$ in our toy example). For each dendritic tree j, let i_j be its number of intermediate nodes and t_j its number of terminal nodes. Then, tree j has $m_j = i_j + t_j + 1$ nodes (intermediate nodes, terminal nodes and one root). The point cloud will have $m = m_1 + ... + m_n$ nodes. In our example, $i_1 = 2$ (nodes 1 and 2), $t_1 = 3$ (nodes 3, 4 and 5), $i_2 = 1$ (node 7), $t_2 = 2$ (nodes 8 and 9) and the roots are nodes 0 and 6. Thus, $m_1 = 6$, $m_2 = 4$.

Using a unique permutation as explained below, we represent the n searched trees in the point cloud. For this, we require that all nodes of our binary MSTs have exactly two children (in our problem most of the nodes are binary) such that, if necessary, we add $d_j = i_j - t_j + 2$ dummy nodes to tree j (it is easy to get this result if one considers that in a binary tree with t leaves, there are exactly $t - 1$ nodes with degree two). Note that dummy nodes are always terminal nodes and they do not affect us when calculating distances in the tree. In our example we need to add two dummy nodes, $d_1 = d_2 = 1$ (nodes 10 and 11, Figure 3(c)).

The length of our permutation σ will be $l = i+t+d$, where $i = i_1+i_2+...+i_n$, $t = t_1+t_2+...+t_n$ and $d = d_1+d_2+...+d_n$ (in the example $l = 3+5+2 = 10$) and each position of the permutation represents who is the parent of each non-root node. Thus, we work with an array *parent* of length l where the roots and the i intermediate nodes of all trees will appear twice (all of them will have two children) and another array *child*, also with length l, with the i intermediate nodes, the

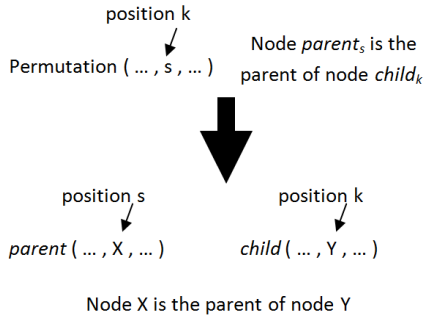

Figure 4: $\sigma_k = s$ and use of *parent* and *child*

t terminal nodes and the d dummy nodes of all trees. From these arrays, our permutations will be such that $\sigma_k = s$ will represent that the parent of $child_k$ is $parent_s$, see Figure 4. Note that we use subindex notation to denote the value of a specific array position, similarly to how we refer to a permutation position.

In the example of Figure 3 arrays *parent* and *child* could be

$parent = (0, 0, 1, 1, 2, 2, 6, 6, 7, 7)$

$child = (1, 2, 3, 4, 5, 10, 7, 8, 9, 11)$

We have obtained these arrays according to the structure of Figure 3(d) (numbered real trees with dummy nodes in those nodes that had degree 1, roots in this case). That is, in *parent* we have decided to place first the root node of the first tree and then its intermediate nodes (twice as required in this array), and then we continue analogously with the second tree. To construct *child* we have placed, once as required in this case, all non-root nodes of the first tree in increasing order and then, also in increasing order, all non-root nodes of the second tree.

Note that arrays *parent* and *child* must be fixed before starting to study permutations since these arrays will determine which DCMSTs represent each permutation. The order of the nodes in these arrays is in fact irrelevant if it holds that *parent* contains twice each root node and each intermediate node and *child* contains once all intermediate nodes, all leaves and all dummy nodes. With the arrays defined above the permutation that represents real dendritic trees, Figure 3(d), is $(1, 3, 4, 5, 6, 2, 7, 9, 10, 8)$ i.e., $parent_1$ (node 0) is the parent of $child_1$ (node 1), $parent_3$ (node 1) is the parent of node $child_2$ (node 2) and so on until the last position of the permutation which indicates that $parent_8$ (node 6) is the parent of $child_{10}$ (node 11).

The representation of binary trees by permutations has been widely studied in the literature [6] [12]. However, these representations are based on a single tree and, as mentioned, we want to be able to build several DCMSTs from the same cloud of points. As we have seen, our permutation representation can encode more than one tree simultaneously (if we want to construct a single tree, $n = 1$, the process would be analogous to the previous example where $n = 2$. It would be also easy to study permutations with $n > 2$).

We compute distances between all pairs of points and when we evaluate an individual of our EDA, represented as a permutation, a small fitness with the total distance of the tree(s) will be preferred. Note that there will be some invalid individuals. With the chosen representation, we always

keep fixed roots and leaves nodes of real neuronal trees so, if we have a correct individual, it will contain the searched number of DCMSTs. However, permutations whose representation contains some cycle represent invalid individuals.

Figure 3(e) represents an example of correct individual, two DCMSTs given by permutation $(2, 4, 7, 5, 6, 1, 8, 3, 10, 9)$, while Figure 3(f) (permutation $(9, 2, 4, 6, 5, 1, 3, 10, 7, 8)$) represents an example of invalid individual because it has a cycle between nodes 1 and 7.

2. EVOLUTIONARY COMPUTATION AND EDAS

Evolutionary computation is a discipline that creates a set of possible solutions and makes them evolve from generation to generation in order to solve optimization problems. Genetic algorithms (GAs) and estimation of distribution algorithms (EDAs) [10] are two well-known models in evolutionary computation. Both of them initialize a set of individuals and apply different operators to modify these individuals trying to find optimal solutions. GAs improve the set of solutions in each generation using crossover and mutation operators, while EDAs use probabilistic models. One of the main issues that needs to be addressed when trying to implement a GA or an EDA for a problem is the definition of an encoding of solutions. We intend to use the permutation representation described above as solution encoding.

GAs have been widely used to solve the DCMST problem and some authors also use permutation representations to solve it [7] but, unlike our representation, they represent a single tree with one of these permutations. We are interested in comparing the performance of these already used GAs with our representation using crossover and mutation operators as proposed in [9]. However, our main objective is to study the problem using EDAs in permutation domains, because EDAs for permutation-based optimization problems have not been extensively developed.

As shown in [2], most of the EDA approaches for permutation problems are adaptations of EDAs designed initially for the solution of integer or real-value domain problems. The main drawback is that those EDAs do not learn a probability distribution over a permutation space, but a distribution over an integer or real-values space. Over the past years some algorithms specifically designed for permutation-based problems have been developed. We think that we can achieve competitive results adapting these algorithms for the study of our optimal neuronal wiring problem.

3. FUTURE WORK

In addition to the discussed steps to develop our problem using evolutionary computation techniques in permutations domains, other aspects could be taken into account. For example, it could be interesting to consider a more complete fitness evaluation function for individuals of our population. One possibility might be to take also into account the minimization of the distance of each point to the root of the tree, which is closely related to minimize the time it takes for a signal to travel from the soma to each leaf.

If we consider several optimization criteria we also need to think about the convenience of using either a uniobjective problem (for example, weighting the different considered objectives) or moving towards a multiobjective problem.

In the case of dendritic trees from the same neuron, another interesting point would be not fixing in advance the number of real DCMSTs and check if the optimal found number of trees matches with reality. More insight would be needed to solve this new problem because it may require a change in the proposed representation.

4. ACKNOWLEDGMENTS

Research partially supported by the Spanish Ministry of Economy and Competitiveness, projects TIN2010-20900-C04-04 and Cajal Blue Brain.

5. REFERENCES

[1] T. N. Bui and C. M. Zrncic. An ant-based algorithm for finding degree-constrained minimum spanning tree. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, GECCO '06, pages 11–18, New York, NY, USA, 2006. ACM.

[2] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano. A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems. *Progress in Artificial Intelligence*, 1(1):103–117, 2012.

[3] H. Cuntz, F. Forstner, A. Borst, and M. Häusser. One rule to grow them all: A general theory of neuronal branching and its practical application. *PLoS Computational Biology*, 6(8):e1000877, 2010.

[4] H. Cuntz, A. Mathy, and M. Häusser. A scaling law derived from optimal dendritic wiring. *Proceedings of the National Academy of Sciences of the United States of America*, 109(27):11014–11018, 2012.

[5] M. R. Garey and D. S. Johnson. *Computers and intractability; A guide to the theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[6] G. D. Knott. A numbering system for binary trees. *Communications of the ACM*, 20(2):113–115, 1977.

[7] M. Krishnamoorthy, A. T. Ernst, and Y. M. Sharaiha. Comparison of algorithms for the degree constrained minimum spanning tree. *Journal of Heuristics*, 7(6):587–611, 2001.

[8] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. In *Proceedings of the American Mathematical Society*, volume 7, pages 48–50, 1956.

[9] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artificial Intelligence Review*, 13(2):129–170, 1999.

[10] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer, 2002.

[11] R. C. Prim. Shortest connection networks and some generalizations. *Bell System Technology Journal*, 36:1389–1401, 1957.

[12] I. Semba. Generation of binary trees from stack permutations. *Journal of Information Processing*, 5:102–105, 1982.

[13] W. Wamiliana. Solving the degree constrained minimum spanning tree problem using tabu and modified penalty search methods. *Journal Teknik Industri*, 6(1):1–9, 2004.