

On a CAD tool based on Kolmogorov's superpositions

Valeriu Beiu *

Centres for Neural-Inspired Nano Architectures(<http://www.cnina.org>), *School of EE&CS, Washington State University, Pullman, WA 99164-2752, USA.*

Based on an explicit numerical (i.e., constructive) algorithm for Kolmogorov's superpositions we will show in this paper that for obtaining minimum size circuits implementing arbitrary Boolean function, the activation function of the neurons is the identity function. Because classical Boolean implementations, as well as threshold logic implementations require exponential size in the worst case, it follows that size-optimal solutions for implementing arbitrary Boolean functions require analog circuitry. It is known that implementing arbitrary Boolean functions using classical Boolean gates (i.e., AND and OR gates) requires exponential size circuits. The bounds are still exponential if threshold logic gates are used for solving arbitrary Boolean functions. It is true that these bounds reveal exponential gaps, suggesting that threshold gate circuits with more layers (depth not constant) might have a smaller size. A different approach is to use Kolmogorov's superpositions, which shows that there are neural networks having only $2n+1$ neurons, which can approximate any function. Such a solution would clearly be size-optimal. A constructive solution for the general case was detailed in [1-4]. If we limit the functions to be *approximated* to Boolean functions, one digit of precision for the output is enough ($k=1$), which gives $\psi(0.i \text{ sub } 1) = 0.i \text{ sub } 1$, which is the identity function $\psi(x) = x$. Such a solution builds very simple 'analog neurons'. They have fan-in $2n+1$, for which the known weight bounds (holding for any fan-in $\Delta > 3$) are: $2 \sup (\Delta - 1)/2 < \text{weight} < (\Delta + 1) \sup (\Delta + 1)/2 / 2 \sup \Delta$. Thus, a precision of between Δ , and $\Delta \log \Delta$ bits per weight would be expected. The constructive solution for Kolmogorov's superpositions requires a double exponential precision for ψ , and for the weights in general. For Boolean functions precision is reduced to $(2n+2) \sup -n$, or $2n \log n$ bits per weight. Analog implementations are limited to just several bits of precision, this being one of the reasons for investigations on precision, and on algorithms relying on limited integer weights. Due to the limitation on precision, an optimal solution for implementing Boolean functions will be suggested. One should start by decomposing the given Boolean function in simpler Boolean functions, which can be efficiently implemented, based on Kolmogorov's superpositions (i.e., we have to reduce n to very small values). The partial results from this first layer of analog building blocks can be combined by using Kolmogorov's superpositions (again), leading to an analog implementation; either threshold logic gates and/or Boolean gate, leading to a mixed analog/digital solution. The final implementation will require more than two layers, but would be size/area optimal. Intuitively the idea is to "rewrite" a given computation (i.e., set of Boolean functions) in a base larger than 2, and use Kolmogorov's superpositions for the analog implementation of the digit-wise computations in this larger base.

[1] D.A. Sprecher. A universal mapping for Kolmogorov's superposition theorem. *Neural Networks*, 6(8): 1089-1094, 1993. [2] D.A. Sprecher. A numerical implementation of Kolmogorov's superpositions. *Neural Networks*, 9(5):765-772, 1996. [3] D.A. Sprecher. A numerical construction of a universal function for Kolmogorov's superpositions. *Neural Network World*, 6(4):711-718, 1996. [4] D.A. Sprecher. A numerical implementation of Kolmogorov's superpositions II. *Neural Networks*, 10(3):447-457, 1997.

* Corresponding author. Tel. +1(509)335-5223. FAX +1(509)335-3818.
Email address: vbeiu@eeecs.wsu.edu (Valeriu Beiu).