Algumas Reflexões sobre o Poder Computacional de Paradigmas Analógicos – Parte II: Aspectos Formais e Especulações

Diogo Coutinho Soriano, Vanessa Brischi Olivatto, Daniel Guerreiro e Silva, Romis Attux

Departamento de Engenharia de Computação e Automação Industrial (DCA)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (Unicamp)
Caixa Postal 6101, CEP 13083-852 – Campinas, SP, Brasil

{soriano, vanessa, danielgs, attux}@dca.fee.unicamp.br

Abstract – In this two-part work, we will discuss the idea of analog computation and aspects of the relationship between the computational power of analog paradigms and classical Turing machines. After an initial analysis of key elements of Turing's formulation, the second part presents theoretical results delimiting the potentiality of analog strategies and also a discussion about the relationship between dynamical systems and computable numbers. This is followed by a brief exposition of a list of final speculative remarks and perspectives.

Keywords – Analog computing, Turing machines, computability, neurocomputing.

1. Introdução

Nesta segunda parte, buscaremos, após uma discussão preliminar acerca do modelo computacional proposto por Turing em seu clássico trabalho de 1936 [1], apresentar alguns resultados que estabelecem comparações entre o poder computacional desse modelo e o de paradigmas mais proximamente vinculados ao *modus operandi* de dispositivos analógicos. Em seguida, exporemos um modelo simples de computação de seqüências (ou números) baseado na operação do mapa logístico [2], um dos mais conhecidos sistemas dinâmicos não-lineares de tempo discreto. Especulamos que esse modelo seja capaz de gerar um conjunto de números computáveis com cardinalidade igual à dos reais, embora ainda não apresentemos uma prova rigorosa disso. É importante ressaltar, aliás, que este trabalho como um todo é apenas um passo inicial de uma pesquisa que pretendemos conduzir e formalizar de maneira bem mais abrangente nos próximos anos.

2. Máquinas de Turing e Além

O conceito de máquina de Turing foi proposto em 1936 por Alan Mathison Turing, matemático britânico [1]. Trata-se, em termos simples, de um dispositivo teórico que manipula símbolos em uma fita (dividida em seções discretas), obedecendo a uma tabela de regras pré-definida. Essa tabela registra ações a serem tomadas de acordo com a configuração atual do dispositivo e também define transições entre elementos de um conjunto finito de estados.

A unidade idealizada é capaz de realizar as seguintes operações:

- Ler o símbolo alinhado com a cabeça de leitura/escrita;
- Imprimir um símbolo na célula lida ou apagar um símbolo nela contido;
- Efetuar uma transição entre estados (que podem, eventualmente, ser coincidentes);
- Mover a cabeça uma célula para a esquerda (E) ou direita (D). Também é possível não realizar movimento algum.

Na Tabela 1, apresentamos um exemplo de tabela de comportamento, exatamente (a menos de mudanças na notação empregada) o exemplo simples dado por Turing em [1] para computação da seqüência 01010101... a partir de uma fita em branco. Os símbolos possíveis são "0" e "1" e um símbolo lido "nenhum" denota, simplesmente, um campo em branco da fita. Não há instruções de apagamento de símbolos nesse caso, de modo que sua possível existência foi ignorada na tabela.

Estado Atual	Símbolo Lido	Símbolo Impresso	Fita	Estado Seguinte
A	nenhum	0	D	B
B	nenhum	nenhum	D	C
C	nenhum	1	D	D
D	nenhum	nenhum	D	A

Tabela 1 – Tabela de Comportamento de uma Máquina de Turing.

Turing define uma seqüência computável como sendo aquela gerada por uma máquina "livre de círculos", ou seja, por uma máquina que efetivamente computa uma seqüência ilimitada. A idéia de número computável surge diretamente

da possibilidade de considerar uma seqüência computável como representando dígitos de sua expansão.

Um ponto crucial do trabalho de Turing é a sua constatação de que o conjunto de números reais computáveis é enumerável, ou seja, corresponde a um subconjunto dos reais com cardinalidade igual à do conjunto dos números naturais. Isso decorre, essencialmente, do caráter "discreto" da tabela de comportamento da máquina e mesmo de sua operação, e, para explicitar esse caráter, Turing adota um sistema de numeração de máquinas que evoca fortemente a clássica estratégia de numeração de Gödel [3]. O sistema de Turing parte de uma padronização da forma das tabelas de comportamento e, em seguida, cria uma codificação que mapeia cada tabela em um número natural, definindo de forma rígida que a cardinalidade do conjunto de máquinas de Turing capazes de operar numa fita inicialmente em branco é igual à dos naturais. Já que o trabalho de Cantor mostra que a cardinalidade dos naturais é menor que a dos reais, haverá certamente reais não-computáveis por máquinas desse tipo.

Outro ponto importante do trabalho é a sua prova de que *não é possível decidir por meios finitos* (usando uma máquina de Turing, por exemplo) se a tabela de comportamento de uma máquina qualquer corresponde a uma operação circular ou livre de círculos, i.e., se uma determinada máquina computa ou não um número de maneira adequada. Para realizar essa prova (por mais de um caminho, aliás), Turing apresenta o belo conceito de *máquina universal*, que evoca vivamente a noção de um computador de propósito geral com programa armazenado. A prova também representa uma engenhosa aplicação do argumento diagonal, o que atrela mais uma vez o resultado aos dilemas cantorianos vinculados aos transfinitos [4]. Modernamente, o resultado de Turing, que termina por desembocar numa resposta negativa para o anseio hilbertiano expresso pelo *Entscheidungsproblem*, é interpretado em termos do chamado problema da parada de uma máquina de Turing (ou *halting problem*) [5].

2.1 – Computação Super-Turing Neural

A idéia de contrastar, de algum modo, paradigmas baseados nos reais e o poder computacional associado a dispositivos clássicos como a máquina de Turing vem atraindo a atenção de pesquisadores há mais de duas décadas (vide, por exemplo, a clássica referência [6]). Não nos proporemos a realizar uma apreciação completa dos esforços nesse campo: restringir-nos-emos, antes, a uma breve discussão sobre alguns resultados muito importantes obtidos sob a égide de um paradigma de neurocomputação e sumarizados em [7].

O modelo neurocomputacional proposto por Hava Siegelmann e Eduardo Sontag corresponde a uma arquitetura de rede recorrente operando em tempo discreto segundo a expressão [7]:

$$x_i(n+1) = \sigma\left[\sum_{j=1}^{N} a_{ij}x_j(n) + \sum_{j=N}^{M} b_{ij}u_j(n) + c_i\right], \quad i = 1, 2, \ldots \qquad (1)$$

sendo $x_i(n)$ a ativação do *i-ésimo* neurônio da rede, $u_j(n)$ uma entrada genérica e os valores a_{ij}, b_{ij} e c_i os pesos sinápticos da rede. A função de ativação é "sigmoidal" e possui a seguinte forma:

$$\sigma(x) = \begin{cases} 0, x < 0 \\ x, 0 \leq x \leq 1 \\ 1, x > 1 \end{cases} \qquad (2)$$

O que Siegelmann e Sontag mostram, essencialmente, é que a rede exposta acima, caso os pesos sinápticos se restrinjam a valores pertencentes aos racionais, é um modelo "polinomialmente relacionado" a uma máquina de Turing [7]. Portanto, em termos simples, podemos dizer que o poder computacional da rede neural composta por valores racionais é essencialmente equivalente ao de uma máquina desse tipo.

Por outro lado, ao permitir que os pesos sinápticos sejam valores reais, Siegelmann e Sontag mostram que é obtido um poder computacional *P/poly*, ou seja, tem lugar a hipercomputação. Conforme explicado em [7], a classe *P/poly* pode ser entendida em termos da operação de uma máquina de Turing com acesso a certo número de "bits adicionais de informação" provenientes de um "conselheiro externo" com um conhecimento virtualmente ilimitado.

Não desejamos aqui entrar em detalhes sobre os aspectos teóricos envolvendo a definição das classes, sendo a nossa intenção apenas ressaltar o fato de que o modelo analógico de Siegelmann e Sontag dá um "salto" de capacidade quando se assume a possibilidade de contar com valores reais. Buscaremos a seguir revisitar essa idéia abstrata a partir de outro *framework*.

3. Computação Baseada num Sistema Dinâmico Simples

As idéias expostas até aqui nos levaram a conceber um modelo de computação analógica baseado na evolução temporal do estado de um sistema não-linear cujo comportamento inclui possibilidades de convergência para pontos fixos, ciclos-limite e atratores associados a comportamento caótico: o mapa logístico [2]. Esse mapa é definido pela seguinte equação:

$$x(n+1) = \mu x(n)[1 - x(n)]. \qquad (3)$$

Para realizar computação de números ou seqüências, propomos que o intervalo unitário $I = [0,1]$, ao qual necessariamente se restringem os valores assumidos por $x(n)$, seja dividido em um número finito de regiões. Esse número define, neste modelo, a cardinalidade do alfabeto de símbolos gerados pela máquina: doravante assumiremos, por simplicidade, uma partição binária, que nos leva a um alfabeto equivalente ao conjunto $\{0,1\}$[1].

A excursão de $x(n)$ pelo espaço de fase passa, dessa forma, a equivaler à computação de uma seqüência de bits que pode ser interpretada, no espírito da abordagem de Turing [1], como um número real entre *0* e *1* (tomado a seqüencia de bits como uma seqüencia de dígitos binários de um número do tipo $0,a_1a_2a_3a_4...$). O efeito de atratores como pontos fixos e ciclos-limite se vincula essencialmente, nesse contexto, a um processo de computação de racionais (devido ao caráter repetitivo do padrão numérico gerado em regime permanente), mas o mesmo não deve ocorrer para o regime caótico.

No caso de um sistema caótico, como o engendrado pela escolha $\mu = 4$, que se caracteriza, ademais, pela realização de iterações que levam o intervalo unitário nele mesmo, esperar-se-ia que: 1) houvesse um padrão "complexo" para os bits – resultado da existência de um expoente de Lyapunov positivo e da limitação do estado a $[0,1]$ – e 2) duas condições iniciais, por mais próximas que seja, levassem à computação de seqüências distintas, o que intuitivamente se associa à idéia de dependência sensitiva às condições iniciais [2].

Sendo esses dois pontos válidos, há uma conseqüência interessante do ponto de vista computacional: cada condição inicial pertencente ao intervalo unitário levaria à computação de um número real distinto, ou, em outras palavras de uma seqüência infinita distinta. Nesse caso, seria possível estabelecer uma bijeção entre o conjunto de condições iniciais – que, no caso, tem cardinalidade igual à da reta real – e o conjunto de números computáveis segundo o paradigma exposto. Teríamos em mãos, portanto, um computador analógico capaz de gerar, de acordo com um procedimento iterativo que evoca até certo ponto a mecânica do dispositivo de Turing, um conjunto de números computáveis com cardinalidade maior que a dos naturais.

Do ponto de vista de computação analógica, poder-se-ia traçar um paralelo entre uma estratégia desse tipo e a operação de um sistema de tempo contínuo com discretização via mapas de Poincaré ou mapas estroboscópicos [2].

4. Reflexões e Especulações

Tendo em vista o que foi exposto ao longo deste trabalho, parece-nos claro que a idéia de computação analógica deva ser considerada com muita atenção tanto do ponto de vista teórico (computabilidade, estudo de processos cognitivos, análise da noção de inteligência artificial) quanto do ponto de vista prático. Em particular, desejaríamos levantar alguns pontos para reflexão que consideramos interessantes:

- De acordo com resultados como os obtidos por Siegelmann e outros [7], discutidos na seção 2.1, há um efetivo aumento de poder computacional a partir do uso de um modelo analógico. Isso se deve, do ponto de vista conceitual, à presença de valores reais na computação, já que, no caso da rede neural com parâmetros racionais, há uma equivalência relativa à máquina de Turing.
- Nesse contexto, modelos de neurocomputação analógica merecem ser alvo de cautelosa análise, uma vez que podem ser capazes de fornecer subsídios para a investigação das relações entre cognição e "*computing machinery*".
- O desenvolvimento de circuitos analógicos reconfiguráveis (FPAAs) [8] abre perspectivas muito interessantes de

[1] Há uma relação muito forte entre as idéias expostas na seção e a área de dinâmica simbólica. No entanto, nesta etapa inicial de trabalho, não nos deparamos com um resultado que seja equivalente ao que estamos apresentando, embora seja possível que isso ocorra com o aprofundamento de nosso contato com a área. Feita essa ressalva, destacamos que nossa meta aqui é fundamentalmente a de levantar uma discussão (no espírito do EADCA), sem maiores pretensões de originalidade.

computação analógica adaptativa e evolutiva, as quais podem permitir *inter alia* estudos relativos à complexidade e emergência em contextos de computação "contínua".

- Paradigmas de computação que evocam a noção de dinâmica simbólica podem trazer visões alternativas acerca de algumas concepções geradas no âmbito do paradigma de Turing. Poder-se-ia pensar, por exemplo, em investigar extensões da clássica concepção de algoritmos "discretos" e mesmo numa adaptação do conceito de complexidade de Kolmogorov [9] a grandezas relacionadas à estrutura de sistemas dinâmicos (e.g. expoentes de Lyapunov).

Agradecimentos

Os autores agradecem o apoio técnico do Prof. Marcio Eisencraft, da UFABC, e o apoio financeiro da CAPES e do CNPq.

Referências

[1] A. M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem", Proceedings of the London Mathematical Society, Ser. 2-42, pp. 230-265, 1936.

[2] L. H. A. Monteiro, *Sistemas Dinâmicos*, Editora Livraria da Física, 2006.

[3] E. Nagel, J. R. Newman, *A Prova de Gödel*, Editora Perspectiva, 2003.

[4] S. Hawking, *God Created the Integers*, Running Press, 2007.

[5] R. Penrose, *A Mente Nova do Rei*, Editora Campus, 1991.

[6] L. Blum, M. Shub, S. Smale, "On a Theory of Computation and Complexity Over the Real Numbers: NP Completeness, Recursive Functions, and Universal Machines", Bulletin of the American Mathematical Society, Vol. 21, pp. 1 – 46, 1989.

[7] H. Siegelmann, "Neural and Super-Turing Computing", Minds and Machines, Vol. 13, pp. 103-114, 2003.

[8] T. S. Hall, P. Hasler, D. V. Anderson, "Field Programmable Analog Arrays: a Floating-Gate Approach", Lecture Notes in Computer Science, Vol. 2438, pp. 133-147, 2002.

[9] T. Cover, J. A. Thomas, *Elements of Information Theory*, Wiley, 2006.