Algunos modelos de computación

Sicard, Andrés[*]
Universidad EAFIT; Medellín, Colombia

10 de Diciembre de 1999

Resumen

Se establecen cuatro divisiones de los modelos de computacin: Computacin Discreta, Computacin, ContinuaComputacin Cuntica Discreta, Computacin Cuntica Continua; y se presentan algunas relaciones entre ellas.

Los trabajos fundacionales de Kurt Gödel y Jacques Herbrand en funciones recursivas, Alonso Church y Stephen Kleene en funciones λ-definibles, Alan Turing y Emil Post en funciones computables, realizados en la década de los $30's$; establecieron las características y las condiciones que deben satisfacer los modelos clásicos de computación. En la actualidad existen varios modelos se computación que no satisfacen las condiciones impuestas por los modelos clásicos, estos nuevos modelos permiten entonces hablar de un área en creciente desarrollo denominada computación no clásica [9, 12], que abarcaría entre otros, modelos de computación ya sean, continuos, biológicos o cuánticos.

Desde el punto de vista del espacio de computación, es posible pensar en por lo menos cuatro categorías de modelos de computación:

1. Computación Discreta
   A esta categoría pertenecen los modelos convencionales de computación, es decir, modelos tales como: las máquinas de Turing, las funciones recursivas y el λ-cálculo, entre otros.

2. Computación Continua
   A esta categoría pertenecen los modelos considerados como modelos de computación continua o analógica, es decir, modelos tales como: las funciones recursivas reales, los modelos de computación sobre estructuras algebraícas (anillos), modelos de redes neuronales con pesos reales, entre otros. Característica fundamental de esta categoría, es que algunos de sus modelos presentan la propiedad de ser modelos Super-Turing.

[*]email address: asicard@sigma.eafit.edu.co

3. Computación Cuántica Discreta
   Esta categoría está formada (en términos generales) por dos modelos equivalentes de computación cuántica: Las máquinas de Turing cuánticas y los circuitos cuánticos.

4. Computación Cuántica Continua
   Para esta categoría, de acuerdo a nuestro conocimiento actual, no es posible hablar de modelos propiamente dichos, aunque si existen algunas indicaciones y sugerencias de como deberían ser éstos.

La figura 1 representa las cuatro categorías mencionadas. Para las categorías, las líneas continuas indican la existencia de modelos, las líneas punteadas por su parte, indican la no existencia de los mismos. En cuanto a las relaciones entre categorías, estas representan relaciones en cuanto el poder computacional de los modelos. Para las categorías de computación discreta y computación continua (relación 1), existen resultados de equivalencia o inclusión computacional entre los diferentes modelos propuestos; para las categorías de computación discreta y computación cuántica discreta (relación 2) también existen resultados de equivalencia computacional entre los modelos propuestos. Por otra parte, en cuanto a las relaciones entre la categoría de computación cuántica continua y las categorías de computación continua (relación 3) y computación cuántica discreta (relación 4), al no existir modelos para la primera de ellas, en el momento presente, no hay relaciones computacionales entre ellas.

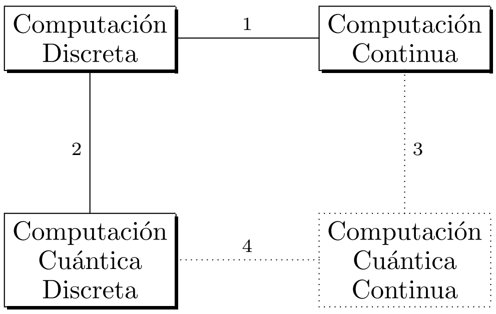

Figura 1: Semi-clasificación de las categorías de modelos de computación.

Dos de las características y condiciones impuestas al espacio de la computación por los modelos de computación clásica son: *finitud* y *discretización*. Es decir; por una parte, los objetos manipulados por una computación son finitos y discretos y por otra parte, la duración de una computación es finita y se mide en unidades discretas de tiempo.

Algunos modelos recientes de computación violan la finitud y/o discretización de los objetos de la computación. El modelo propuesto por Lenore Blum, Mike

Shub y Steve Smale [5, 4] es un modelo de computación sobre un anillo ordenado arbitrario $\mathcal{A}$; cuando $\mathcal{A} = \mathbb{Z}_{\in}$, el modelo es equivalente a una máquina de Turing, pero cuando $\mathcal{A} = \mathbb{R}$, el modelo computa funciones sobre los números reales. El ampliar el dominio de la computación de funciones sobre los números naturales a funciones sobre los números reales, exige trasladar y expandir los conceptos y métodos propios de la computación a un dominio continuo; así por ejemplo, Vasco Brattka [6] y Cristopher Moore [18] presentan (independientemente) una definición de las funciones recursivas reales con base en un conjunto de funciones iniciales sobre los números reales y algunos esquemas de creación de funciones sobre los mismos.

Otros modelos, violan la finitud y/o discretización de la evolución de la computación. Joel Hamkins y Andy Lewis [15], presentan un modelo de máquinas de Turing para un tiempo discreto pero infinito de computación; en la misma dirección que las anteriores Jack Copeland presenta las *Accelerated Turing Machines* [10, 11], estas son máquinas de Turing que realizan la siguiente operación en la mitad del tiempo que la operación anterior; Cristopher Moore [18] por su parte, presenta un modelo de computación el cual opera con un parámetro continuo de tiempo.

Los modelos de computación que permiten la manipulación de objetos continuos pertenecen a una área de la computación teórica denominada *computación continua* o *computación analógica*; sin embargo, en la actualidad no existen ni la unicidad ni se han obtenido los resultados en esta área, que sí se han obtenido en la computación discreta o computación digital [23]. Uno de los aspectos más importantes de la computación continua, es que para algunos de sus modelos, se ha demostrado que son más potentes que una máquina de Turing, es decir, -aceptando la tesis de Church-Turing- estos modelos computan lo no computable y son denominados modelos *Super-Turing* [30, 26].

Un modelo Super-Turing $\mathcal{ST}$ es construido usualmente presentando un modelo $\mathcal{T}$ equivalente a una máquina de Turing y bajo ciertas modificaciones se obtiene el modelo Super-Turing $\mathcal{ST}$. Para demostrar que $\mathcal{ST}$ es un modelo Super-Turing, se demuestra que éste es equivalente a un modelo Super-Turing ya conocido o se demuestra que éste puede resolver un problema que una máquina de Turing no puede resolver. Se han construido modelos Super-Turing al interior de las más diversas teorías, como lo indican los siguientes ejemplos:

1. Desde la teoría de las redes neuronales, Hava Siegelmann y Eduardo Sontag [26, 28, 29] presentan un modelo de red neuronal denominado *Analog Recurrent Neural Network (ARRN)*, en el cual, sí los pesos asociados a las neuronas son números racionales, la ARNN es equivalente a una máquina de Turing, pero sí los pesos son números reales, la ARRN se comporta como un modelo Super-Turing.

2. Desde la teoría de los sistemas dinámicos, Hava Siegelmann [27] construye un sistema dinámico con la propiedad de ser un modelo Super-Turing,

denominado *Analog Shift Map*. Por su parte, Petr Kurka [16] construye dos sistemas dinámicos asociados a una máquina de Turing, denominados *Turing Machine with Moving Tape (TMT)* y *Turing Machine with Moving Head (TMH)* (para éstos no se ha demostrado que sean modelos Super-Turing).

3. Mike Stannett [30] a partir de un modelo conocido como *X-machine* propone un modelo denominado *Analogous X-Machine (AXM)* que también se constituye en un modelo Super-Turing.

Por otra parte, una de las áreas de mayor investigación actual en computación es la *computación cuántica*, que correlaciona elementos de la informática teórica y de la mecánica cuántica, para producir modelos de computación que exploten las propiedades y efectos cuánticos inherentes a las partículas atómicas. El trabajo de Peter Shor [25] señaló el aspecto fundamental en el cual se diferencian la computación clásica y la computación cuántica: *la complejidad algorítmica*. Esta diferencia consiste en que es posible definir algoritmos cuánticos cuya complejidad temporal sea menor a de los algoritmos clásicos (conocidos hasta el momento). En particular Shor realizó la descripción de un algoritmo cuántico de complejidad temporal de tipo polinomial para la factorización de números enteros.

Aunque la mecánica cuántica opera naturalmente sobre objetos continuos, los modelos actuales de computación cuántica, tales como los *circuitos cuánticos* [2, 14, 24, 21] y las diferentes versiones de *máquinas de Turing cuánticas* propuestas por David Deutsch [13], Ethan Bernstein y Umesh Vazirani [3], Masanao Ozawa y Haramichi Nishimura [19, 20, 22, 21] y Dorit Aharanov [1] entre otros; son considerados como modelos de computación discretos [3].

En la actualidad existen algunas propuestas de Computación Cuántica Continua; Seth Lloyd y Samuel Braunstein [17] han propuesto un modelo de computación cuántica continua bajo ciertas limitaciones de las transformaciones que operan sobre las variables continuas, Samuel Braunstein [7] por su parte, presenta para algunas transformaciones cuánticas discretas, sus correspondientes transformaciones cuánticas continuas.

Existe un modelo de computación denominado máquinas de Hilbert [31] que permite considerar desde un mismo punto de vista, algunos modelos de computación de redes neuronales y algunos modelos de computación cuántica discreta (entre otros). Una máquina lineal [31] opera sobre un espacio vectorial y modela su dinámica por medio de operadores lineales, por su parte, una máquina topológica lineal [31] opera sobre un espacio vectorial topológico y modela su dinámica por medio de operadores lineales continuos, cuando el espacio vectorial topológico es un espacio de Hilbert, la máquina se denomina una máquina de Hilbert. Las máquinas de Hilbert pueden representar modelos discretos o continuos de computación; para el caso discreto, la máquina operaría sobre un espacio de Hilbert de dimensión n isomorfo a $\mathbb{C}^n$; para el caso continuo, la máquina operaría sobre un espacio de Hilbert separable e infinito dimensional isomorfo al espacio de sucesiones infinitas cuadrado sumables l^2.

Referencias

[1] Dorit Aharonov, *Quantum computation*, Eprint: `arXiv.org/abs/quant-ph/9812037`, 1998.

[2] Adriano Barenco, Charles H. Bennett, Richard Cleve David P. DiVincenzo, Norman Margolus Peter Shor Tycho Sleator, John Smolin, and Harald Weinfurter, *Elementary gates for quantum computation*, Phys. Rev. A **52** (1995), no. 5, 3457–3467.

[3] Ethan Bersntein and Umesh Vazirani, *Quantum complexity theory*, SIAM J. Comput. **26** (1997), no. 5, 1411–1473.

[4] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale, *Complexity and real computation*, New York: Springer-Verlag, 1998.

[5] Lenore Blum, Mike Shub, and Steve Smale, *On a theory of computation and complexity over the real numbers: NP-completeness recursive functions and universal machines*, Bull. Amer. Math. Soc. **21** (1989), no. 1, 1–46.

[6] Vasco Brattka, *Recursive characterization of computable real-valued function and relations*, Theoretical Computer Science **162** (1996), no. 1, 45–77.

[7] Samuel L. Braunstein, *Error correction for continuous quantum variables*, Phys. Rev. Lett. **80** (1998), no. 18, 4084–4087.

[8] Cristian S. Calude, J. Casti, and M. J. Dinneen (eds.), *Unconventional models of computation*, Singapure: Springer-Verlag, 1998.

[9] B. Jack Copeland, *The broad conception of computation*, American Behavioral Scientist **40** (1997), 690–716, Preprint: `www.phil.canterbury.ac.nz/philsite/people/jack_copeland` [15-May-1999].

[10] ———, *Even Turing machines can compute uncomputable functions*, Unconventional models of computation (Cristian C. Calude, J. Casti, and M. J. Dinneen, eds.), Singapure: Springer-Verlag, 1998, pp. 150–164.

[11] ———, *Super Turing-machines*, Complexity **4** (1998), no. 1, 30–32.

[12] B. Jack Copeland and Richard Sylvan, *Beyond the universal Turing machine*, Australasian Journal of Philosophy **77** (1999), 44–66.

[13] David Deutsch, *Quantum theory, the Church-Turing principle and the universal quantum computer*, Proc. R. Soc. Lond. A **400** (1985), 97–117.

[14] ———, *Quantum computational networks*, Proc. R. Soc. Lond. A **425** (1989), 73–90.

[15] Joel Hamkins and Andy Lewis, *Infinite time Turing machines*, The Journal of Symbolic Logic **65** (2000), no. 2, 567–604.

[16] Peter Kurka, *On topological dymanics of Turing machines*, Theoretical Computer Science **174** (1997), no. 1–2, 203–216.

[17] Seth Lloyd and Samuel L. Braunstein, *Quantum computation over continuous variables*, Phys. Rev. Lett. **82** (1999), no. 8, 1784–1787.

[18] Cristopher Moore, *Recursion theory on the reals and continuous-time computation*, Santa Fe, Institute, 1997.

[19] Masanao Ozawa, *Measurability and computability*, Eprint: `arXiv.org/abs/quant-ph/9809048`, 1998.

[20] ———, *Quantum Turing machines: Local transition, preparation, measurement, and halting*, Eprint: `arXiv.org/abs/quant-ph/9809038`, 1998.

[21] Masanao Ozawa and Haramichi Nishimura, *Computational complexity of uniform quantum circuit families and quantum Turing machines*, Eprint: `arXiv.org/abs/quant-ph/9906095`, 1999.

[22] ———, *Local transition function of quantum Turing machines*, Eprint: `arXiv.org/abs/quant-ph/9811069`, 1999.

[23] Roger Penrose, *Las sombras de la mente*, Coleccin: Drakontos, Barcelona: Crtica, 1996.

[24] Eleanor Rieffel and Wolfgang Polak, *An introduction to quantum computing for non-physicists*, ACM Computing Surveys **32** (2000), no. 3, 300–335.

[25] Peter W. Shor, *Polinomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM J. Comput. **26** (1997), no. 5, 1484–1509.

[26] Hava T. Siegelmann, *Computation beyond the Turing limit*, Science **268** (1995), 545–548.

[27] ———, *The simple dynamics of super Turing theories*, Theoretical Computer Science **168** (1996), no. 2, 461–472.

[28] Hava T. Siegelmann and Eduardo D. Sontag, *On the computational power of neural nets*, Proc. Fifth ACM Workshop on Computational Learning Theory, Pittsburg, July 1992, 1992.

[29] ———, *Analog computation via neural networks*, Theoretical Computer Science **131** (1994), no. 2, 331–360.

[30] Mike Stannett, *X-machines and the halting problem: building a super-Turing machine*, Formal Aspects of Computing **2** (1990), 331–341.

[31] Herbert Wiklicky, *Quantitative computation by Hilbert machines*, En: [8] (1998), 406–426.