André Kenji Horie
Paulo Muggler Moreira
Ricardo Henrique Gracini Guiraldelli
Virgílio Vettorazzo

Proposta de Metodologia para Análise da Complexidade de Sistemas Computacionais em um Âmbito Global e Evolutivo

São Paulo

2008

André Kenji Horie
Paulo Muggler Moreira
Ricardo Henrique Gracini Guiraldelli
Virgílio Vettorazzo

Proposta de Metodologia para Análise da Complexidade de Sistemas Computacionais em um Âmbito Global e Evolutivo

Monografia para o curso PCS 2058 - Engenharia de Informação.

Escola Politécnica da Universidade de São Paulo
Departamento de Engenharia de Computação e Sistemas Digitais

São Paulo

2008

Abstract

This work proposes a model for assessing complexity of computer systems in a global and evolutive scope. This will be achieved by defining relevant metrics and the way they relate among themselves, providing a quantifiable measure for complexity. Moreover, the proposed model will be applied to available data, thus creating a valid complexity curve whose tendency will be analysed.

Keywords: Computer Systems Complexity

Sumário

1 Introdução

1.1 Objetivo

Este trabalho tem por objetivo analisar a complexidade em sistemas computacionais em um âmbito global e evolutivo através da definição de uma metodologia, identificando-se métricas relevantes para o cálculo da curva de complexidade. Aplicando-se o modelo proposto, espera-se verificar a tendência da complexidade em função do tempo.

1.2 Conceitos Teóricos

Para um estudo sobre a complexidade de sistemas computacionais, vê-se a necessidade de se definir o conceito de complexidade, para então reduzir esta definição ao escopo de sistemas computacionais e, assim, poder considerar os aspectos relevantes para a criação do modelo.

1.2.1 Complexidade

Warren Weaver, em seu artigo "*Science and Complexity*" (WEAVER, 1948), introduziu o conceito de complexidade na litetura científica como o grau de dificuldade de se prever as propriedades de um sistema se as propriedades de cada parte for dada. Classifica-se então a complexidade sistêmica em organizada e desorganizada. Os sistemas de complexidade desorganizada caracterizam-se pelo número elevado de variáveis e pelo seu comportamento caótico, embora as propriedades do sistema como um todo possam ser entendidas utilizando-se métodos probabilísticos e estatísticos. A complexidade organizada, por outro lado, refere-se a interações entre as partes constituintes do sistema, sendo o comportamente deste redutíveis às interações, e não às propriedades das partes elementares. A visão proposta por este artigo influenciou fortemente o pensamento contemporâneo acerca da complexidade.

Empiricamente, observa-se que uma proporção alta dos sistemas complexos encontrados na natureza possuem uma estrutura hierárquica. Em teoria, espera-se que qualquer sistema

complexo seja hierárquico, tendo como a decomposição uma propriedade de sua dinâmica (SIMON, 1962). Esta simplifica tanto o estudo do comportamento como a descrição desses sistemas.

Em 1988, Seth Lloyd afirma que a complexidade de uma propriedade física de um objeto é função processo ou conjunto de processos responsáveis por sua criação (LLOYD, 1988). Em outras palavras, a complexidade é uma propriedade da evolução de um estado, e não do estado em si. Consequentemente, uma medida da complexidade deve classificar sistemas em estados aleatórios como de baixa complexidade, e quantificar a evolução deste sistema para seu estado final.

1.2.2 Organização de Sistemas Computacionais

O termo "arquitetura" é amplamente utilizado para se referir à estrutura na qual um sistema é organizado. Em Tecnologia da Informação, sistemas computacionais são representados pela arquitetura de hardware, de software, de rede e de informação (GENTLEMAN, 2005). Em relação ao escopo coberto por cada uma delas, observa-se que a arquitetura de hardware é essencialmente local, sendo assim definido para apenas um nó do sistema, enquanto as arquiteturas de rede e de informação requerem necessariamente a interação entre diversos componentes. A arquitetura de software, no entanto, pode tanto indicar a organização do software em apenas um componente como também em diversos componentes distribuídos, quando aplicável. Esta decomposição vê-se necessária para melhor endereçar a complexidade de cada vertente de um sistema computacional.

2 Definição do Modelo

2.1 Variáveis Relevantes

A seguir serão apresentadas as variáveis relevantes para o modelo de complexidade em termos globais.

2.1.1 Arquitetura de Hardware

Existem diversas abordagens para a medição da complexidade da arquitetura de hardware, todas elas baseando-se na complexidade como uma função do processo de construção dos componentes. É possível dizer, a fins de simplificação, que o crescimento de complexidade entre os principais componentes deve ser proporcional. Caso contrário, o gargalo de um computador seria facilmente observável em pouco tempo, o que de fato não ocorre.

Existem diversos valores representativos para o cálculo de complexidade relacionada à arquitetura de hardware. O primeiro deriva da Lei de Moore (MOORE, 1965), que afirma que o número de transistores em circuitos integrados aumenta exponencialmente, conforme ilustra a figura 1. Outros valores incluem métricas normalmente utilizadas para *benchmarks*, como por exemplo o número de instruções por segundo do processador, a capacidade de armazenamento das memórias de acesso aleatório, cache e do disco rígido. De qualquer forma, todos os estes valores devem supostamente respeitar a mesma tendência.

Desta forma, a complexidade de hardware assume a forma exponencial:

$$\mathbb{C}_{HW,local} = \alpha \cdot e^{\beta t} + \gamma \tag{2.1}$$

onde α, β e γ são fatores de ajuste.

Vale lembrar também que o escopo da arquitetura de hardware é essencialmente local. Para transpor este valor para um âmbito global, parece coerente assumir que há uma proporção direta entre o quanto a complexidade aumenta com o quanto os custos de produção totais

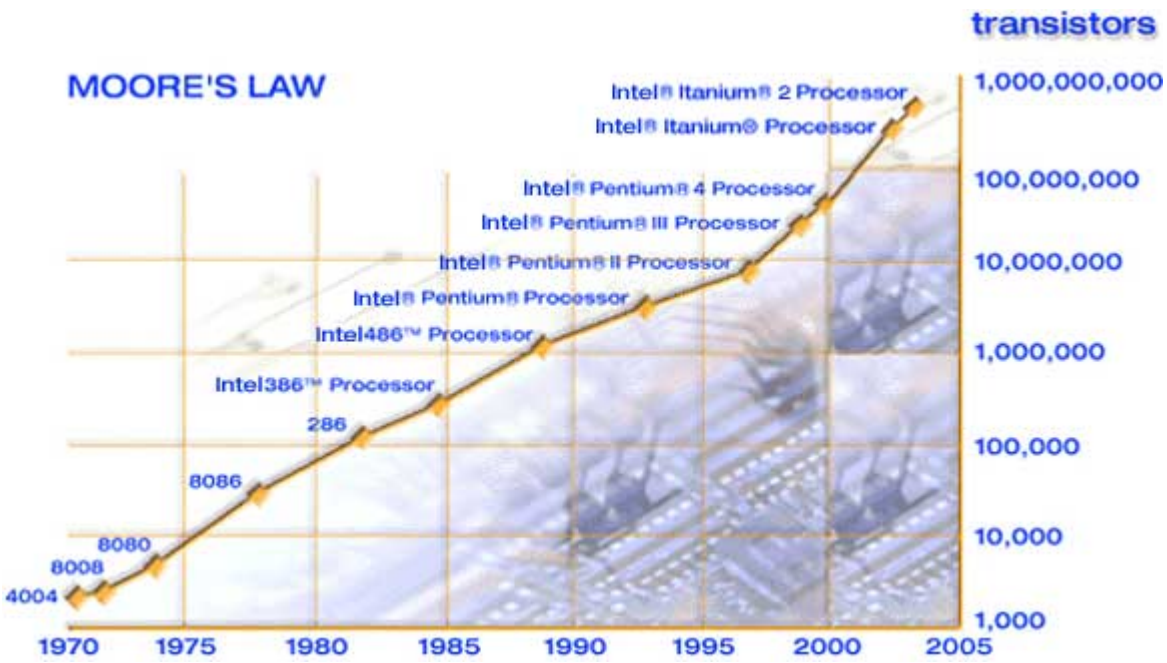

Figura 1: Lei de Moore

aumentam. Dado o custo total de produção como $C_T(Q) = C_F + C_V = C_F + Q \cdot C_{Mg}$, onde C_T é o custo total, C_F é o custo fixo, C_V é custo variável, Q é a quantidade produzida e C_{Mg} é o custo marginal. Portanto, a complexidade da arquitetura de hardware global é dada por $\mathbb{C}_{HW} = f(Q) \cdot \mathbb{C}_{HW,local}$, sendo $f(Q)$ um fator de aumento da complexidade em função da quantidade produzida. Este deve ser proporcional aos custos totais C_T e portanto, considerando-se um modelo simplificado, é linear.

2.1.2 Arquitetura de Software

(KEARNEY et al., 1986), (RANGANATHAN; CAMPBELL, 2007), (LEHMAN; RAMIL, 1998), (VASA; SCHNEIDER, 2003),(MCCABE; R, 1989). Enquanto os aspectos computacionais de complexidade temporal e espacial de algoritmos têm sido estudados extensivamente, o aspecto humano da complexidade no desenvolvimento de software permanece relativamente imaturo. Este aspecto ainda não é bem compreendido, dificultando a obtenção de estimativas diversas, o que leva frequentemente a aumentos de prazo e/ou orçamento de projetos de software, ou até ao fracasso completo dos mesmos. (RANGANATHAN; CAMPBELL, 2007) identifica cinco aspectos da complexidade em sistemas computacionais. Em seu artigo, ele propõe meios de medir cada um destes aspectos e explicita como cada um dos aspectos afeta os envolvidos nos processos de software: desenvolvedores, administradores e usuários. Os aspectos de complexidade identificados por (RANGANATHAN; CAMPBELL, 2007) são: Complexidade estrutural da tarefa; Imprevisibilidade; Complexidade de tamanho; Complexidade Caótica; Complexidade de

Algoritmo;

- Complexidade Ciclomática; - Número de Linhas de Código; - Número de Módulos;

2.1.3 Arquitetura de Rede

.

2.1.4 Arquitetura de Informação

.

2.2 Modelo Matemático

.

3 Aplicação do Modelo

3.1 Dados Históricos

3.1.1 Arquitetura de Hardware

Através dos dados históricos relacionados à arquitetura de hardware, pretende-se verificar:

Processador	MIPS	Frequência (MHz)	Ano
4004	0,06	0,1	1971
8008	0,06	0,2	1972
8080	0,64	2,0	1974
8086	0,33	5,0	1978
80286	0,90	6,0	1982
386 DX	5,0	16,0	1985
486 DX	20,0	25,0	1989
80486 DX2	50,0	50,0	1992
Pentium	60,0	60,0	1993
Pentium Pro	200,0	200,0	1995
Pentium II	300,0	300,0	1997
Pentium II Xeon	400,0	400,0	1998
Pentium III	500,0	500,0	1999
Pentium III Xeon	550,0	550,0	1999
Mobile Pentium II Xeon	400,0	400,0	1999
Pentium 4	1.500	1.500	2000
Itanium	2.500	800	2001
Pentium 4 Northwood	10.000	3.200	2003
Pentium 4E Prescott	11.000	3.800	2004

Tabela 1: Evolução dos processadores da Intel

- Se está de fato correto assumir a proporção entre o crescimento de complexidade dos diversos componentes de hardware;
- Se a tendência de evolução da complexidade segue a Lei de Moore.

A partir de dados obtidos dos próprios processadores da Intel, observa-se o número de milhões de instruções por segundo e o de frequência do clock em função do ano de introdução do processador no mercado na tabela 1.

A partir destes dados, as figuras 2a e 3b são obtidas. Claramente, observa-se nelas a tendência exponencial conforme suposta anteriormente.

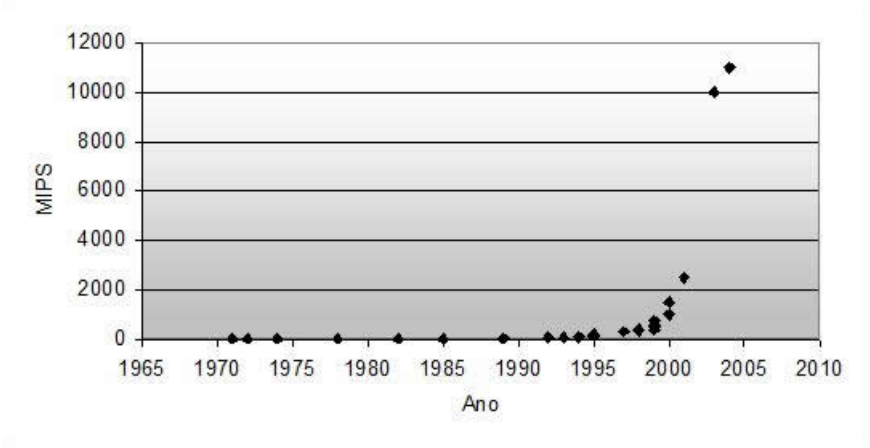

Figura 2: Tendência da evolução do MIPS para processadores Intel

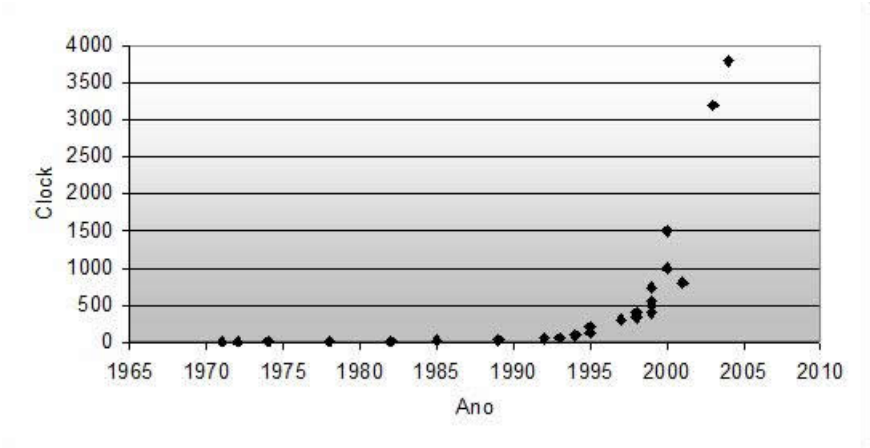

Figura 3: Tendência da evolução do clock para processadores Intel

Por fim, utilizando-se as capacidades de disco rígido e de memória de acesso aleatório como parâmetro, obtemos a mesma tendência (ver figuras 4 e 5).

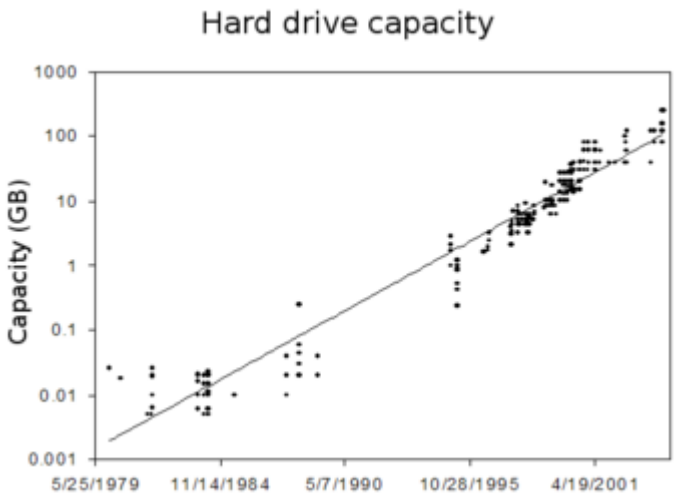

Figura 4: Tendência da evolução da capacidade de disco rígido

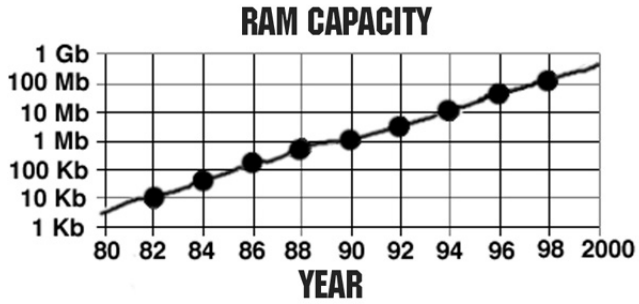

Figura 5: Tendência da evolução da capacidade de memória de acesso aleatório

3.1.2 Arquitetura de Software

.

3.1.3 Arquitetura de Rede

.

3.1.4 Arquitetura de Informação

3.2 Curva de Complexidade

.

4 Conclusão

4.1 Avaliação do Modelo

4.2 Avaliação da Curva de Complexidade

Referências

GENTLEMAN, W. M. *Dynamic Architecture*: Structuring for change. 2005.

KEARNEY, J. P. et al. Software complexity measurement. *Commun. ACM*, ACM, New York, NY, EUA, v. 29, n. 11, p. 1044–1050, 1986. ISSN 0001-0782.

LEHMAN, M. M.; RAMIL, J. F. Metrics and laws of software evolution - the nineties view. *Proceedings of the Fourth International IEEE Symposium, 1997*, IEEE, 1998.

LLOYD, S. *Black Holes, Demons and the Loss of Coherence*: How complex systems get information, and what they do with it. Tese (Doutorado em Física Teórica) — The Rockefeller University, Nova York, NY, EUA, 1988.

MCCABE, T. J.; R, C. W. B. Design complexity measurement and testing. *Communications of the ACM*, ACM, New York, NY, EUA, v. 32, n. 12, 1989.

MOORE, G. E. Cramming more components onto integrated circuits. *Electronics Magazine*, v. 38, n. 8, 1965.

RANGANATHAN, A.; CAMPBELL, R. H. What is the complexity of a distributed computing system? *Complex.*, John Wiley & Sons, Inc., New York, NY, USA, v. 12, n. 6, p. 37–45, 2007. ISSN 1076-2787.

SIMON, H. A. The architecture of complexity. *Proceedings of the American Philosophical Society*, v. 106, n. 6, p. 467–482, 1962.

VASA, R.; SCHNEIDER, J.-G. Evolution of cyclomatic complexity in object oriented software. *Proceedings of the 7TH WORKSHOP ON QUANTITATIVE APPROACHES IN OBJECT-ORIENTED SOFTWARE ENGINEERING*, 2003.

WEAVER, W. Science and complexity. *American Scientist*, v. 36, p. 536–544, 1948.