PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
DEPARTAMENTO DE ECONOMIA

MONOGRAFIA DE FINAL DE CURSO

PLATAFORMAS E COMPETIÇAO NO MERCADO DE COMPUTADORES: O CASO DA APPLE COMPUTER

Diego Navarro Pozo
Nº de matrícula: 0211592-0

Orientador: João Manoel Pinho de Mello

Junho de 2005

> I suspect, gentlemen, that you are looking at me with compassion; you tell me again that an enlightened and developed man, such, in short, as the future man will be, cannot consciously desire anything disadvantageous to himself, that that can be proved mathematically. I thoroughly agree, it can – by mathematics. But I repeat for the hundredth time, there is one case, one only, when man may consciously, purposely, desire what is injurious to himself, what is stupid, very stupid – simply in order to have the right to desire for himself even what is very stupid and not to be bound by an obligation to desire only what is sensible.
>
> **Fjodor Dostoyevsky**, *Notes from the Underground.*

Sumário

1
Economia da indústria de computação

1.1
Introdução

1.1.1
Fatos estilizados

Os mecanismos tradicionais de retornos marginais decrescentes, livre entrada e saída de firmas e agentes racionais otimizadores não parecem produzir os resultados esperados em indústrias sujeitas a determinados tipos de fricção técnica[1] que se impõem sobre as decisões de consumidores e firmas e desviam os resultados do mercado do equilíbrio Pareto-ótimo descrito pela teoria microeconômica clássica para cenários fortemente dependentes do desenvolvimento histórico dessas indústrias (*path-dependence*), bem como das características de interação tecnológica subjacentes.

O termo "fricção técnica", por analogia com a experiência física do atrito em superfícies ásperas, descreve como características tecnológicas inerentes aos bens e serviços produzidos em determinadas indústrias dificultam ou impedem o ajuste para um equilíbrio ótimo como descrito pela teoria microeconômica mais tradicional.

O termo "fricção técnica", por analogia com a experiência física do atrito em superfícies ásperas, descreve como características tecnológicas inerentes aos bens e serviços produzidos em determinadas indústrias dificultam ou impedem o ajuste para um equilíbrio ótimo como descrito pela teoria microeconômica mais tradicional.

Notoriamente, a escolha de sistemas operacionais – o software básico indispensável ao uso cotidiano de um computador – está longe de ser "competitiva" no sentido de responder bem a variações marginais de preço e qualidade: quando se decide substituir o Windows por um concorrente, têm grande peso o nosso

[1] A discussão na literatura costuma se concentrar em problemas específicos como externalidades de rede (*network externalities*) ou custos de troca (*switching costs*) sem usar este termo unificador que usaremos nesta monografia.

investimento em bens e capital humano complementares e específicos ao sistema Windows, a necessidade de interoperação e troca de arquivos num mundo em que virtualmente todos os usuários de computadores pessoais são consumidores da plataforma Windows e mesmo a compatibilidade de um sistema operacional alternativo com o hardware – i.e. o computador como peça física – de que já se dispõe.

Dado isto, o resultado do mercado depende das circunstâncias tecnológicas associadas ao problema de compatibilidade. Se um computador Macintosh consegue abrir arquivos do Microsoft Word, consegue obter uma fatia de mercado que lhe seria impossível de outro modo, posto que consegue se inserir, ao menos parcialmente, na base instalada de usuários desse editor de texto.

Na prática, dado o ciclo de renovação e a fronteira móvel representada pela lei de Moore, compatibilidade é mais importante que preço, como mostra também a experiência de 14 anos do sistema operacional Linux. Isso gera altas barreiras à entrada, posto que novos jogadores não podem forçar seu caminho para o mercado com uma política agressiva de preços.

1.1.2 Modularidade e fricção técnica

Essa classe de problemas de fricção técnica existe em virtualmente qualquer cenário em que haja grupos de bens complementares entre si, e que conjuntamente competem com grupos similares – por exemplo, meios de transporte e infra-estrutura associada – mas se tornam mais prementes à medida que a sofisticação técnica, com a subseqüente modularidade, cria questões de interação estratégica que moldam dinamicamente o resultado do "jogo" econômico. Virtualmente qualquer classe de bem ou serviço é o resultado de um processo cultural de definição e tem em sua estrutura interna uma modularidade que é consolidada em um dado arranjo monolítico, eliminando assim da discussão pragmática o problema de definir a classe de bem ou serviço para a qual se constrói um modelo econômico.

Em outras palavras, embora a consolidação da arquitetura técnica e do desenho industrial dos calçados masculinos usados na nossa cultura seja um problema econômico genuíno de modularidade gerando *path-dependence* e finalmente um equilíbrio possivelmente sub-ótimo, na prática a fricção técnica envolvida é irrelevante, por duas razões básicas:

1. A baixa densidade das opções enfrentadas no desenho industrial do produto – essencialmente, tipos de solado e materiais (solados de madeira não podem ser colados a sapatos de camurça).

2. A ordem de grandeza dos custos dessa fricção – as externalidades de rede são virtualmente todas de raiz cultural, embora seja possível pensar numa interação entre solados e pavimento, e o custo de troca e adaptação a um modelo diferente de calçado também é desprezível.

Nestes termos, o aumento da sofisticação técnica da economia – ou seja, o aumento do peso de setores tecnicamente muito sofisticados – tem três efeitos sobre o impacto econômico da fricção técnica:

1. O aumento violento da modularidade nas decisões que envolvem a delimitação e a definição de uma classe de bens ou serviços cria mais "gargalos" de fricção técnica que acumulam custos.

2. Com o aumento da modularidade surge também um incentivo maior a explorar a fricção técnica para obter um excedente econômico.

3. A sofisticação técnica afeta os custos de troca, tanto pelo aumento da modularidade como pela ordem de grandeza do investimento em capital humano complementar a uma dada configuração técnica.

De fato, embora o estudo das externalidades de rede tenha aparecido na ciência econômica já com Thorstein Veblen, é a compreensão dos problemas microeconômicos práticos associados à indústria de computação que dá o grande impulso a essa literatura. Em um manual de divulgação da literatura acadêmica na área para o público de tomadores de decisão no mundo corporativo, [Shapiro e Varian 1999] resumem a súbita premência pragmática da compreensão destas questões de forma lapidar:

> There is a central difference between the old and new economies: the old industrial economy was driven by *economies of scale*; the new information economy is driven by *economics of networks*.

1.2 Paradigmas, designs básicos e tradições

As literaturas de mudança tecnológica e economia da inovação costumam trabalhar com duas ferramentas para analisar os padrões de evolução tecnológica, os conceitos de *paradigma tecnológico* e *design básico*. Hagedoom, Carayannis e Alexander [Hagedoom, Carayannis e Alexander 2001] discutem as definições variáveis desses conceitos na literatura, e os organizam através de um paralelo ao conceito clássico de paradigma de Thomas Kuhn. O conceito kuhniano de

paradigma se referiria a "the problems and methods of a field of science and technology", e o apelo ao jargão kuhniano ressaltaria "the particular role that a disciplinary matrix or a community of practitioners plays in defining a field of technology".

A definição de paradigma tecnológico que mais se aproximaria do que estes autores procuram seria a de Giovanni Dosi – "a model and pattern of solution for selected technological problems".

Nota-se, entre o apelo kuhniano à importância fundamental de uma "comunidade" na delimitação de um paradigma e a ênfase de um conjunto de problemas a ser resolvido, a característica dual do conceito de paradigma tecnológico.

No início do desenvolvimento de um paradigma, é de se esperar que existam abordagens competindo como solução canônica para um determinado conjunto de problemas técnicos. O artigo de Hagedoom concentra-se na computação pessoal como concebida no início dos anos 90, e considera que as abordagens tecnológicas básicas que competiram para dominar o paradigma "PC" – o IBM-PC e o Macintosh. Estas abordagens tecnológicas são os designs básicos.

Nós propomos que existe um terceiro ângulo neste gênero de análise da mudança tecnológica. Por um lado, o paradigma envolve um problema a ser resolvido, e um conjunto de "interfaces" entre o problema e os designs básicos – por exemplo, o problema de tabulação de um censo, e o cartão perfurado que pode ser processado por uma máquina de tabulação mecânica ou um computador equipado com um leitor de cartões.

Por outro, o design básico é uma abordagem tecnológica associada a um problema. Assim, não é correto dizer que o Macintosh de 1984, uma solução para problemas de negócios que competia com o IBM PC para a formação de um paradigma seja o mesmo design básico do Macintosh de 1994, uma máquina primariamente usada por profissionais de design gráfico e música.

No entanto, é extremamente relevante o fato de que o Macintosh de 1984 e o de 1994 têm muito em comum, em particular todo um espectro de bens complementares e de capital humano específico formado. Isto é mais dramático no caso do UNIX, inicialmente usado em "minicomputadores" à PDP-10, em seguida em workstations à SGI/Alpha/Sun, e finalmente em PCs commoditizados — designs básicos diferentes —- e cuja aplicação começa em problemas numéricos de grande porte, passando a rodar em servidores de rede — cada vez mais importantes com a expansão da computação distribuída pela Internet — e finalmente a desktops, desde o específico (workstations pessoais, desktops corporativos de função limitada) até o geral (Macintosh a partir de 1998).

O fato é que existe uma base de software ligado à *tradição* Unix desde o

	Comunidade	Tecnologia	Aplicação
Paradigma	X		X
Design básico		X	X
Tradição	X	X	

Figura 1.1: Paradigmas, designs básicos e tradições

início dos anos 70 que pode ser trivialmente compilado de modo a rodar num PC Macintosh ou um x86 commodity rodando Linux, além de – o que é mais importante – uma extensa ecologia de engenheiros e técnicos de todos os níveis, ou seja, um grande investimento social em capital humano.

Tradições são extremamente importantes na formação de paradigmas. A força da plataforma Windows não vem apenas das externalidades de rede e *switching costs* individuais e coletivos associados à atual versão do Windows, mas do fato de que a última versão do Windows (ao tempo desta monografia, o Windows XP) roda todo o software escrito para a tradição DOS/Windows desde o início dos anos 80 – bem como as técnicas de programação associadas: um programador que tivesse sido congelado em 1985 e acordado hoje ainda pode ser colocado em frente a um terminal Windows com as ferramentas que conhece (um compilador da sua época) e produzir software que funciona hoje – i.e. que os *sunk costs* em bem complementares são continuamente reaproveitados e absorvidos por novos paradigmas ligados à mesma tradição.

Um dos temas críticos desta monografia é que a decisão da Apple de integrar o Macintosh à tradição Unix (através do NeXTStep, que traz em si toda uma outra tradição de programação própria, em torno da linguagem Objective-C, menos bem-sucedida, mas que existe desde o final dos anos 80 e tem alguma base de capital humano específico associada) muda inteiramente o cenário descrito pela *accepted wisdom* da Apple como perdedora de uma guerra de paradigmas em meados dos anos 80, como constante, por exemplo, em Hagedoom [Hagedoom, Carayannis e Alexander 2001] ou West [West 2004] – um cenário memoravelmente caracterizado por este último como *Betamax redux*.

1.3 Externalidades de rede

1.3.1 Modelo com um monopolista otimizador

De forma mais mundana, sob a presença de externalidades de rede, a utilidade para o consumidor depende não apenas da quantidade que ele consome,

mas da quantidade total de bens do mesmo tipo ou compatíveis existentes no mercado.

[Economides 1996] propõe uma função de demanda simples que depende apenas da quantidade e do tamanho esperado da base.

$$p = p^D(q, n^E) \tag{1-1}$$

Montamos nesta seção dois modelos que exploram algumas conseqüências algébricas da formulação proposta por Economides – que, apesar de simples, gera alguns fenômenos de grande interesse.

1.3.2 Equilíbrio com dois períodos

Suponhamos um modelo com dois períodos de tempo. No primeiro, os consumidores estimam a quantidade de unidades que um dado produto venderá – e conseqüentemente a base instalada.

Se o tamanho da base for igual ao número de unidades vendidas – i.e., se $n^E = q$, temos uma *curva de demanda de expectativas perfeitamente realizadas.*

Com base nesse conceito, realizamos um exercício algébrico simples sobre o modelo de Economides, com curvas de demanda lineares até onde o uma determinada quantidade $\bar{q}$, e zero a partir daí, mudando a inclinação com o tamanho da rede.[2] . Assim, a demanda por um bem com base instalada n^E é

$$p(q, n^E) = \begin{cases} n^E - \frac{n^E}{\bar{q}} q & \text{se } q \leq \bar{q} \\ 0 & \text{caso contrário} \end{cases} \tag{1-2}$$

Normalizando $\bar{q} = 1$ e fazendo $n = n^E$ (expectativas perfeitamente realizadas, obtemos:

$$p(n, n) = n - n^2 \quad \text{para } n < 1 \tag{1-3}$$

É possível encontrar os valores de n que maximizam o lucro do monopolista definindo sua função-lucro e obtendo as condições de primeira e segunda ordem.

$$\Pi(n) = p(n, n) \times n - c(n) = n^2 - n^3 - c(n)$$

$$\partial\Pi/\partial n = 2n - 3n^2 - c'(n) = 0$$

[2]Esta especificação funcional é compatível com a abordagem de *consumer choice* de [Katz e Shapiro 1985], onde a externalidade de rede é um fator aditivo simples à utilidade derivada do produto na ausência de rede instalada (i.e., $u(q, \bar{n}) = v(q) + x(\bar{n})$). Com menos qualificações, a curva de demanda com um ponto de inflexão aparece em [Varian 2003] – a partir de argumentos verbais – e [Economides 1996] – a partir de um gráfico similar à figura 1.2

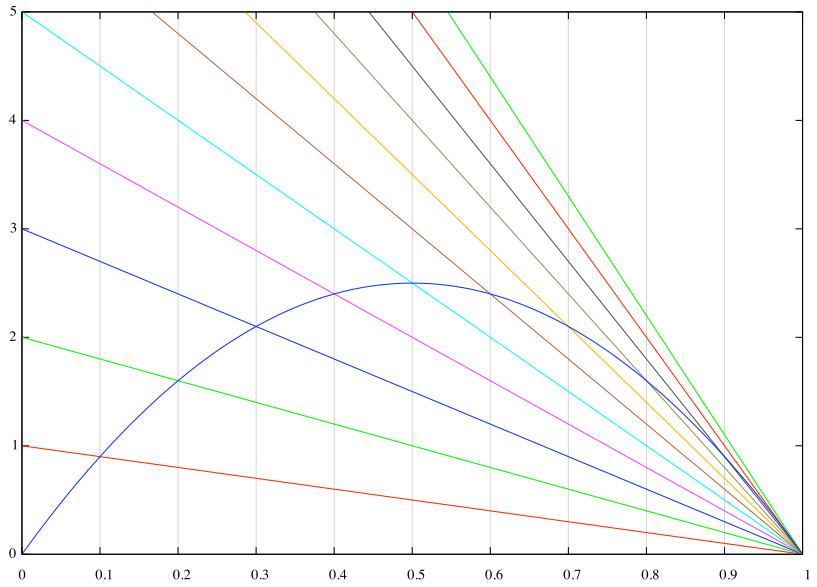

Figura 1.2: Curva de demanda com expectativas realizadas

$$\partial\Pi^2/\partial^2 n = 2 - 6n - c''(n) > 0 \Rightarrow n > \frac{c''(n)}{6}$$

Ao maximizar o seu lucro, o monopolista iguala o seu custo marginal à sua receita marginal. Sem muitas hipóteses sobre o custo marginal, é fácil ver que podem existir duas quantidades positivas que maximizam o lucro do monopolista. No entanto, no modelo de dois períodos é importante lembra que a existência de dois equilíbrios de mercado igualmente desejáveis não implica numa proposição mais forte sobre a dinâmica desse mercado. Claramente, um monopolista no equilíbrio de produção mais baixa tem um incentivo a aumentar sua produção, visto que até o outro equilíbrio, para cada unidade de produção adicional a receita marginal é maior que o custo marginal. Por outro lado, um monopolista no equilíbrio de maior produção também tem incentivo a diminuir gradualmente sua produção até encontrar o primeiro equilíbrio. *É preciso introduzir a competição (imperfeita) entre padrões técnicos para gerar uma preferência inequívoca da firma por uma fatia de mercado maior*, através das decisões dos produtores de bens complementares, dependentes do padrão técnico capitaneado pela firma.

Ainda assim, este modelo extremamente ingênuo permite imaginar um monopolista sobre computadores com interface gráfica (o Macintosh) partindo de uma fatia de mercado de 1/3 e decidindo se empreende uma ampliação do seu mercado ou se estabiliza num nível de produção menor, com lucros mais altos. Em outras palavras, *reduzir o tamanho do mercado pode ser uma decisão consciente*, ao contrário do cenário *betamax redux* de [West 2004], se abdicamos da hipótese da competição de paradigmas à [Hagedoom, Carayannis e Alexander 2001]. No entanto, esta decisão não pode ser implementada apenas com o preço, mas

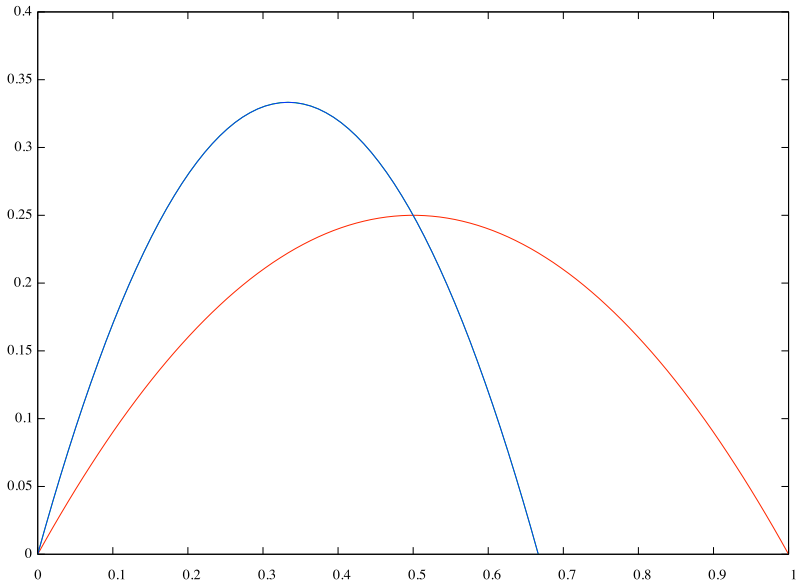

Figura 1.3: Demanda e receita marginal

também com algum choque em n^E.

Mesmo aceitando os resultados contra-intuitivos de um modelo de externalidades de rede, é preciso levar em conta o fato estilizado notado por [Beggs e Klemperer 1992] de que "managers often seem more concerned with market shares than with short-run profit".A presença de competição de paradigmas, bem como a introdução de outros problemas de fricção técnica – em particular, *switching costs* – dá mais peso ao equilíbrio de market share alta.

Talvez a falha mais crítica deste modelo seja a dinâmica ingênua de convergência para um equilíbrio de expectativas perfeitamente realizadas assumida quando se afirma que $n^E = n$. Mesmo um modelo dinâmico mais restritivo de expectativas racionais ($n_t^E = E\left(n_t | \{n_{t-1}, n_{t-2}, \ldots\}, \mathcal{I}\right)$, onde $\mathcal{I}$ é um conjunto de informação disponível) dependem de um processo de aprendizado que pode não ser possível em um sistema de trajetória não-ergódica.

Em muitos episódios históricos – como o pré-anúncio do Windows 1.0, dois anos antes da sua factibilidade tecnológica, com o propósito de desencorajar sistemas gráficos concorrentes já no mercado [Edstrom e Eller 1998] – jogadores da indústria agem especificamente no sentido de influenciar estas expectativas com um choque em $\mathcal{I}$. Esta jogada estratégica é conhecida entre os analistas da imprensa especializada como *vaporware*.

Nas palavras da direção da Apple Computer no relatório à Securities and Exchange Comission referente ao primeiro trimestre de 1998,

> The Company's future operating results and financial condition are dependent upon (...) its ability to effect a change in marketplace

perception of the Company's prospects, including the viability of the Macintosh platform.

De fato, se deixamos o mundo de dois períodos do tempo – monopolista decidindo a quantidade a ser produzida de uma vez só – obtemos um modelo dinâmico simples e muito interessante.

1.3.3 Massa crítica

Retomando a curva de demanda da equação 1-2, temos:

$$p^D = n^E - n \times n^E$$

Vamos tornar este modelo dinâmico assumindo uma hipótese muito simples[3]: $n_t^E = n_{t-1}$

$$p^D(n_t, n_{t-1}) = n_{t-1} - n_t n_{t-1} = (1 - n_t)n_{t-1} \tag{1-4}$$

O monopolista maximizador de lucro tem a seguinte função-objetivo:

$$\Pi(n) = \left[(1 - n_t)n_{t-1}\right] n_{t-1} - c(n_t) \tag{1-5}$$

Pelas condições de primeira ordem, este é máximo quando $\partial\Pi/\partial n = 0$, ou seja:

$$n_{t-1} - 2n_t n_{t-1} = \partial c/\partial n$$

Assumindo novamente uma função-custo afim da forma $c(n) = c_0 + c_1 n$, temos que

$$n_{t-1} - 2n_t n_{t-1} = c_1$$

$$n_t = \frac{n_{t-1} - c_1}{2n_{t-1}} \tag{1-6}$$

[3]É importante notar que a inclinação de cada uma das curvas de demanda (análogas às da figura 1.2) depende não do somatório das unidades vendidas ao longo da história (i.e. da base instalada), mas das unidades vendidas no último período – i.e. da*market-share*. Generalizaremos a relação dinâmica entre base instalada e *market share* na seção **??**

Uma forma de justificar a função como especificada é que computadores têm uma alta taxa de depreciação, e todos os agentes renovam seus equipamentos a cada período. Assim, podemos assumir que os períodos do modelo correspondem ao ciclo de renovação observado de fato na indústria. É possível generalizar este resultado atribuindo pesos variáveis aos instantes do passado – por exemplo, fazendo $n_t = (\sum_{i=1}^{k} w_i n_{t-i})/(\sum w_i)$ e escolhendo um peso decaindo exponencialmente como $w_i = e^{-ai}$. Não adotamos estas generalizações porque não alteram qualitativamente os resultados que encontramos.

Este modelo muito simples gera um fenômeno conhecido como *massa crítica*: essencialmente, em certo nível acontece uma mudança de regime na produção, como visto na figura 1.4, gerada simulando o modelo da equação 1-6 :

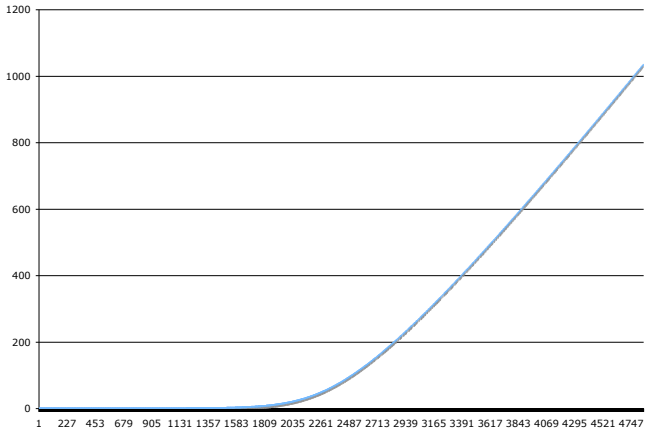

Figura 1.4: Massa crítica

O interessante é que não se trata de crescimento exponencial, mas de uma mudança entre dois regimes "localmente lineares" de aumento de vendas, como visto na figura 1.5:

Da mesma forma como o *lock-in* é um fato estilizado de mercados sujeitos a *switching costs*, a presença de uma dinâmica do tipo "massa crítica" é o fato estilizado mais notório de um mercado sujeito a externalidades de rede. Parte da literatura que discute o assunto – em geral com modelos muito mais sofisticados que o nosso – qualifica este resultado no contexto de um modelo estocástico.

De forma bastante intuitiva, notamos que a trajetória do modelo depende violentamente do nível em que a produção se encontra; em um modelo linear, um aumento súbito das vendas não levaria a uma mudança estrutural na taxa de crescimento, mas é nítido que neste modelo, um choque aleatório pode mudar toda a trajetória do sistema (*path-dependence*) e que – saindo um pouco do escopo do modelo – isso pode afetar o panorama de uma indústria em que vários paradigmas competem para alcançar massa crítica primeiro.

Analiticamente, a expressão da primeira diferença da série gerada pelo modelo (o aumento de base instalada em cada período) é:

$$n_t = \frac{n_{t-1} - c_1}{2n_{t-1}} \Leftrightarrow \Delta n_t = \frac{(n_{t-1} - n_{t-2}) \times c_1}{2n_{t-1}n_{t-2}} = \frac{c_1}{2}\left(\frac{1}{n_{t-2}} - \frac{1}{n_{t-1}}\right) \qquad (1\text{-}7)$$

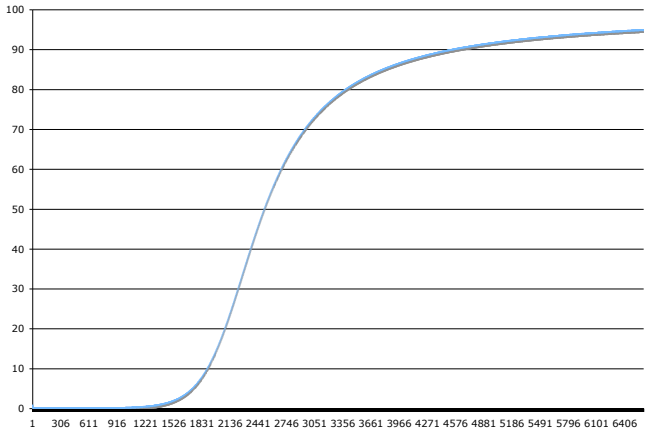

Figura 1.5: Primeira diferença da figura 1.4; nota-se uma mudança de regime no aumento de vendas, que no entanto se estabiliza.

e temos as derivadas

$$\frac{\partial\left(\Delta n_t\right)}{\partial n_{t-1}} = \frac{c_1}{\left(n_{t-1}\right)^2} > 0 \quad \forall c_1 > 0 \tag{1-8}$$

$$\frac{\partial\left(\Delta n_t\right)}{\partial n_{t-2}} = \frac{-c_1}{\left(n_{t-2}\right)^2} < 0 \quad \forall c_1 > 0 \tag{1-9}$$

que mostram que choques súbitos têm um efeito sobre a inclinação da curva de base instalada ainda maior que a lenta evolução até o mesmo nível.

1.3.4
O modelo de Shapiro & Katz

Uma abordagem alternativa à construção de uma função de demanda e o estudo de suas conseqüências é proposta por [Katz e Shapiro 1985], que constróem um modelo de equilíbrio parcial sob oligopólio. Apresentamos uma pequena variante desse modelo adaptado à nossa discussão.

Como no modelo de demanda desenvolvido na seção anterior, os consumidores primeiro formam expectativas em relação à base instalada (*network size*). Neste modelo a oferta aparece mais cedo, com as firmas tomando as expectativas dos consumidores e decidindo preços. Finalmente, os consumidores fazem suas decisões com base em um preço de reserva e no tamanho esperado da base instalada.

O modelo assume também que os consumidores são heterogêneos na sua disposição a pagar independente da externalidade de rede, mas homogêneos

no valor dado a essa externalidade de rede. Assim, o preço de reserva de um consumidor do tipo r é:

$$\hat{p_i} = r + v(y_i^e) \tag{1-10}$$

onde $v(\cdot)$ é o valor dado à externalidade de rede, e r é uniformemente distribuído entre menos infinito e um certo valor máximo A. Neste modelo, cada agente adquire o produto para o qual o excedente sobre o preço de reserva é máximo – ou seja, maximiza

$$E(i) = r + v(y_i^e) - p_i \tag{1-11}$$

Se duas firmas i e j produzem bens homogêneos, então ambas terão vendas positivas apenas se

$$p_i - v(y_i^e) = p_j - v(y_j^e) = \phi \tag{1-12}$$

onde a expressão $p_i - v(y_i^e)$ representa o preço *hedonicamente ajustado* – isto é, líquido depois das externalidades de rede.

Para um dado valor de ϕ, existem $A - \phi$ consumidores, visto que apenas consumidores com $r \geq \phi$ entram no mercado. Assim, para um dado vetor de produções por firma $(x_1, \ldots, x_n)$, o vetor de preços deve satisfazer $A - \phi = \sum_{i=1}^{n} x_i$, ou seja,

$$A + v(y_i^e) - p_i = \sum_{j=1}^{n} x_j \tag{1-13}$$

para todo i para o qual $x_i > 0$. Rearrumando os termos, obtemos

$$p_i = A + v(y_i^e) - \sum_{j=1}^{n} x_j \tag{1-14}$$

Se todas as redes individuais definidas pelos produtos são incompatíveis, a receita de cada firma é

$$R_i = x_i \left[A - \left(\sum_j x_j \right) + v(x_i^e) \right] \tag{1-15}$$

Se todos os produtos são compatíveis entre si, então a receita será

$$R_i' = x_i \left[A - \left(\sum_j x_j \right) + v \left(\sum_j x_j^e \right) \right] \tag{1-16}$$

Assim, a firma decide ser compatível se $R_i' - R_i$ excede o custo de alcançar a compatibilidade – pesquisa, licenciamento e compra de tecnologias, essencial-

mente. Para fins de notação, estes resultados para diferentes configurações de compatibilidade podem ser generalizados fazendo t_i ser a família de conjuntos que contém os números-índices das marcas compatíveis com a marca i. Assim, no caso de compatibilidade completa, $t_i = \{1, ..., n\}$ e no caso de incompatibilidade completa, $t_i = \{i\}$.

Em um equilíbrio similar ao do oligopólio de Cournot, com a restrição adicional de que as quantidades esperadas se tornem as quantidades reais, teremos a condição

$$x_i^* = A + v\left(\sum_{k \in t_i} x_k\right) - \sum_{j=1}^{n} x_j^* \tag{1-17}$$

A solução que torna a condição válida para todos os i é:

$$x_j^* = \frac{1}{n+1} \cdot \left\{ A + nv\left(\sum_{k \in t_i} x_k^e\right) - \left[\sum_{j \neq i} v\left(\sum_{k \in t_j} x_k^e\right)\right] \right\} \tag{1-18}$$

Esta equação relaciona um conjunto de quantidades esperadas $\{x_1^e, ..., x_n^e\}$ ao conjunto de quantidades de equilíbrio $\{x_1^*, ..., x_n^*\}$. O equilíbrio de expectativas plenamente realizadas acontece onde $x_i^e = x_i^*$ para todos os i.

1.4 Custos de troca

1.4.1 Origens

Em indústrias tecnicamente sofisticadas, a decisão de consumir dentre diferentes firmas, marcas ou plataformas com freqüência depende fortemente da decisão tomada no período anterior, devido à presença de custos de troca. Ao estabelecer um relacionamento com um banco, um cliente tem um custo de troca na medida em que o novo banco não sabe sua qualidade como tomador de empréstimo e por isso cobra juros mais altos. Num exemplo mais próximo, usuários do Microsoft Word enfrentam um custo altíssimo de troca gerado pelo fato de que têm boa parte de sua produção no formato eletrônico que só pode ser aberto por esse programa.

Em linhas mais gerais, costuma-se dizer que custos de troca emergem do investimento em bens complementares e incompatíveis com a competição. Segundo [Klemperer 1987], fontes comuns para esses custos de troca são:

– **Necessidade de compatibilidade com o equipamento existente.** Os exemplos da indústria de computação são inúmeros, e nós discutimos detalhadamente o problema da compatibilidade de software nesta monografia.
– **Custos de transação ao substituir fornecedores.**, o que acontece no exemplo já citado da escolha entre bancos, ou quando uma planta industrial tem que ser adaptada às necessidades de um cliente.
– **Custo do aprendizado do uso de novas marcas** – ou seja, investimento em capital humano complementar. Existe um forte incentivo para que as firmas que produzem plataformas tornem extremamente específico o capital humano complementar para elevar os custos de troca.
– **Incerteza sobre a qualidade de marcas não-testadas.**
– **Custos de troca artificiais**, através de sistemas de incentivos como cupons de descontos e programas de milhagem.
– **Custos psicológicos** (não-econômicos) de "lealdade".

Em conjunto, estas razões parecem formar uma certa lealdade a uma marca ou linha. E embora custos de transação e incerteza sobre a qualidade possam levar a resultados empíricos surpreendentes de lealdade a marcas num ambiente de bens perfeitamente substitutos (como encontrados em [Jackson e Perloff 1996]), o problema da compatibilidade técnica é chave na formação destes custos de troca.

1.4.2 O problema do lock-in

Custos de troca suficientemente altos podem significar que não existem políticas de preço de entrada agressivas o suficiente que permitam a um novo jogador disputar pelo mercado já estabelecido. Esta dependência de uma linha compatível de produtos é conhecida popularmente como *lock-in*. Independente de outros efeitos de fricção técnica, a adoção de um padrão em um dado período do tempo e o conseqüente investimento em bens complementares tornam muito difícil a adoção de um padrão técnico diferente. Por esta razão, muitas aplicações de computação ainda se encontram presas a tecnologias antigas, renovadas periodicamente.

O *lock-in* é também a razão pela qual firmas individuais optam por não participar de padrões técnicos "abertos" de base instalada mais ampla. Ao tentar impor seu protocolo proprietário como padrão *de facto*, a firma aposta em grandes excdentes econômicos a serem gerados pelo *lock-in*.

Em outras palavras, uma firma de tecnologia enfrenta um *trade-off* entre os ganhos obtidos com o *lock-in* aos seus protocolos técnicos proprietários e

a inserção em redes de usuários mais amplas. Esta é, por exemplo, a decisão conflitiva que a Apple Computer tem que assumir quando decide o grau de compatibilidade de sua linha Macintosh com o mundo commoditizado dos PCs x86.

1.5 Custos de troca coletivos

Da combinação de externalidades de rede na escolha *a priori* do consumidor e custos de troca nas suas decisões de renovação, emerge um problema de coordenação coletiva de custos de troca que fora das soluções de canto rende um problema isomórfico ao Dilema do Prisioneiro.

Supondo um mercado com dois consumidores, atualmente investidos na plataforma A, que lhes rende utilidades π_A^1 e π_A^2. Para maior simplicidade, podemos assumir que esta utilidade é líquida do preço, i.e. que $\pi_A^i = u_A^i - p_A$. Denotemos por v^1 e v^2 a externalidade de rede – a utilidade de estar na mesma plataforma que o outro jogador – e por c^1 e c^2 o custo de trocar para a plataforma B. Cada jogador decide simultanea e independentemente se muda para a plataforma B ou permanece para a plataforma A. Assim, temos uma bi-matriz típica de um jogo de coordenação:

$$\begin{bmatrix} (\pi_A^1 + v^1, \pi_A^2 + v^2) & (\pi_A^1, \pi_B^2 - c^2) \\ (\pi_B^1 - c^1, \pi_A^2) & (\pi_B^1 + v^1 - c^1, \pi_B^2 + v^2 - c^2) \end{bmatrix}$$

Para que um jogador k mude incondicionalmente para a plataforma B – i.e. independente da decisão do outro jogador, será preciso que

$$\pi_B^k - pi_A^k > c^k - v^k \tag{1-19}$$

Um desconto da firma B

Como $\pi_B^k = u_B^k - p_B$, $c^k - v^k$ pode ser interpretado também como o desconto que a firma B tem que oferecer para induzir o jogador k a mudar independentemente da decisão do outro dadas utilidades iguais, i.e.

$$u_A^k = u_B^k \Longrightarrow p_A - p_B > c^k - v^k$$

No entanto, se ambos jogadores mudam, i.e. se

$$p_A - p_B \geq max(c^1 - v^1, c^2 - v^2) \tag{1-20}$$

a perda da externalidade de rede deixa de acontecer. Assim, é vantajoso para os dois jogadores coordenar uma troca quando

$$p_A - p_B \geq max(c^1, c^2) \tag{1-21}$$

e o jogo de aceitar um desconto para mudar se torna um dilema do prisioneiro na região onde $max(c^1, c^2) < p_A - p_B < max(c^1 - v^1, c^2 - v^2)$.

O papel do pioneiro

Um resultado provocativo da condição 1-20 é que a menor externalidade de rede "sangra" para o resultado do mercado, no sentido em que, dados custos de troca iguais e compensados pela diferença de preços, o desconto oferecido pela firma entrante só precisa compensar o agente com a menor externalidade de rede para induzir a troca. Vemos isto graficamente na figura 1.6, onde Π_A e Π_B são os *pay-offs* da bi-matriz para cada jogador onde ele escolhe A e B, e o jogador muda de marca quando $\Pi_B - \Pi_A \geq 0$.

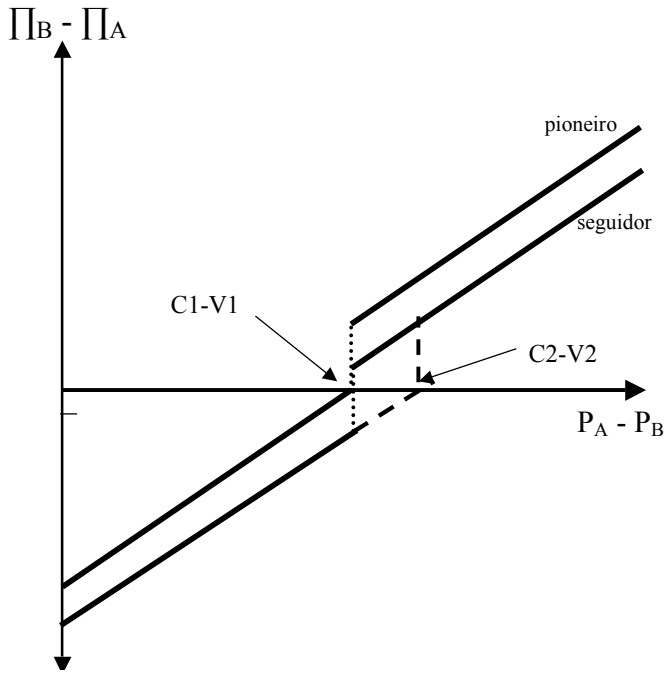

Figura 1.6: O agente com a menor externalidade de rede induz a mudança do mercado

Este é provavelmente, dentre os três fenômenos analisados, o maior mecanismo gerador de *path-dependence*. A importância do pioneiro gera um amplo repertório de estratégias para engolfar o mercado através da atração de pioneiros

com externalidades de rede mais baixas, bem como dá grande importância a certos eventos pontuais na direção do mercado. Para dar alguns exemplos,

- É comum que empresas sujeitas a estes efeitos concedam grandes descontos a estudantes. Ainda no início das suas vidas "computacionais", os estudantes têm nitidamente externalidades de rede mais baixas.
- A um custo mais baixo que obter compatibilidade completa, algum grau de compatibilidade pode ser oferecido a um segmento do mercado de modo a induzir um choque em u_B^k. Por exemplo, usuários em potencial do Linux tipicamente têm externalidades de rede mais baixas em suas funções de utilidade, e é possível induzí-los a migrar para a plataforma oferecendo compatibilidade com arquivos do Microsoft Word, sem incorrer nos custos de obter compatibilidade com outros softwares líderes de mercado. Se um grande número de usuários migra para o Linux porque o preço é mais baixo e a compatibilidade parcial oferecida lhe é suficiente para compensar sua externalidade de rede, outros usuários podem ter incentivo a migrar e incorrer nos custos de troca de lidar com novos softwares de planilha eletrônica, computação numérica, etc.
- Uma forma de entrar num mercado mais amplo pode ser começar por um submercado bem mais específico, sem um "vencedor" claro ainda, e construir uma base instalada para a plataforma. Esta era, por exemplo, a estratégia da BeInc. de Jean-Louis Gassée, que começou se projetando como sistema operacional para mídia, embora fosse funcional em todos os outros aspectos. Esta também parece ser claramente a estratégia da Sun Microsystems para a linguagem Java, que começou como uma linguagem para *applets* que rodam dentro de uma página web, e vem ganhando progressivamente espaço como linguagem de programação para aplicativos multiplataforma independentes do browser.

1.6 Economia positiva e normativa da fricção técnica

1.6.1 Fricção técnica como falha de mercado, e a pragmática dos mecanismos regulatórios

Nesses termos, os problemas de fricção técnica, ao gerar os problemas microeconômicos associados, podem ser encarados como uma fonte importante de *falhas de mercado*. Na abordagem de [Stiglitz 2000], que reflete o pensamento

mainstream da microeconomia do bem-estar, o conceito de falha de mercado é apresentado como conseqüência da ausência de uma ou mais hipóteses críticas para os Teoremas do Bem-Estar, que a partir de um mercado competitivo geram equilíbrios Pareto-ótimos nas trocas (taxas marginais de substituição iguais para todos os indivíduos), produção (taxas marginais de substituição técnica iguais em todas as firmas) e cesta de produção (taxa marginal de transformação da economia é igual à taxa marginal de substituição).

Assim, é importante ter clareza dos problemas de análise econômica positiva associados às falhas de mercado da fricção técnica por duas razões: para compreender de forma normativa e positiva a dinâmica do processo regulatório que começa a se tornar importante na última década, e para separar o que é uma falha de mercado genuína que gera resultados ineficientes mesmo dados agentes racionais e o que é meramente comportamento infra-racional dos agentes em resposta à crescente complexidade das opções que enfrentam.

Na prática normativa, os mecanismos de regulação econômica têm enfrentado o problema de agir sobre as falhas de mercado ligadas à fricção técnica acabam se concentrando sobre um de seus efeitos possíveis, a concentração do mercado em poucas firmas que derivam disso grandes excedentes econômicos. Um exemplo recente e extremo disso foi o prolongado processo do Departamento de Justiça americano contra a Microsoft, amplamente discutido em [Bresnahan 2001]; putativamente "culpada" de empregar com grande habilidade todos os recursos de aquisição de excedente que a fricção técnica de sua indústria permite – ao menos dentro de um esquema normativo que valoriza a eficiência de mercado e a consecução dos teoremas do Bem-Estar – a Microsoft foi acusada de exercer poder de monopólio, o que era perfeitamente improcedente dada a diversidade de sistemas operacionais concorrendo nos magros pontos percentuais de fatia de mercado não ocupados pelo seu principal produto, o Windows.

As narrativas em [Bresnahan 2002a] e [Bresnahan 2002b] enfatizam bastante essa estratégia pragmática de usar o dispositivo das leis anti-truste para conter a *juggernaut* Microsoft. Esta estratégia se repete em outros episódios de órgãos regulatórios tentando conter as falhas mais evidentes resultantes da fricção técnica — mais recentemente, a determinação da União Européia que se produza uma versão do Windows sem determinados recursos de multimídia. É difícil identificar se este divórcio entre a análise positiva e a prática normativa é uma estratégia legal um tanto cínica com o objetivo de conter a Microsoft *de qualquer forma*, ou uma falha de identificação do problema de economia positiva subjacente.

Nos termos de uma análise econômica normativa pura, tais regulações são ilegítimas, posto que atacam uma conseqüência do problema sem atacar o problema subjacente, e podem vir a ser usadas para intervir em casos em

que a condição de monopólio não é resultado de falhas de mercado, mas de características técnicas propriamente ditas. Falando do exemplo específico, a falha de mercado não é que a Microsoft domine o mercado de software de reprodução de mídia (música e filmes, principalmente), mas que seu domínio nesse mercado se deva à não-disponibilidade dos protocolos de compressão de dados proprietários da Microsoft em plataformas alternativas, e do poder que a empresa detém de impor esses protocolos como padrões *de facto* dado seu peso precedente no mercado correlato de sistemas operacionais com a plataforma Windows.

A teoria normativa prática da regulação do mercado de computação parece se concentrar na otimalidade local de um arranjo batizado em [Bresnahan 2001] como "divided technical leadership" (DTL). O objetivo da autoridade regulatória seria facilitar a emergência de ordens em que cada "estágio" modular de um processo de produção suposto linear e acumulativo (chips, sistemas operacionais e aplicativos, por exemplo) é dominado por uma firma – e logo por um padrão técnico – existindo espaço para a mudança técnica nas descontinuidades de coordenação entre os agentes dominantes em cada fase do processo.

Nesse modelo, usado tanto pela defesa como pela promotoria do caso americano contra a Microsoft, é ótimo que computadores e sistemas operacionais sejam dominados por um padrão (Intel x86 e Microsoft Windows). O cerne do caso americano era a inclusão do *browser* Internet Explorer como parte do sistema operacional Windows, o que para a acusação consistiria de um desvio do cenário de DTL. Em outras palavras, a Microsoft estaria alavancando seu domínio no mercado de sistemas operacionais para obter também domínio em um "mercado" putativo de browsers – muito embora todos os browsers estivessem gratuitamente disponíveis para *download* e a estratégia do maior concorrente da Microsoft nessa seara, a Netscape, fosse alavancar o domínio no mercado de browsers para conquistar o segmento de servidores web.

Dentro de um argumento microeconomicamente mais estrito que identifique a raiz da evidente ineficiência do domínio da Microsoft com falhas de mercado associadas à fricção técnica que não permite a livre entrada e saída de firmas no mercado de sistemas operacionais e, num futuro orwelliano, de browsers. De fato, a ineficiência nesse mercado é ligada à incompatibilidade entre produtos gerando externalidades de rede na escolha *a priori* entre marcas e padrões, custos de troca na renovação de linhas e problemas de coordenação dados os custos coletivos de troca que poderiam eliminar o problema de externalidades de rede.

Assim, enquanto todos os browsers compartilharem a suíte de protocolos comuns à rede (TCP/IP, HTTP, XML ,CSS), abertos por terem sido formados por órgãos técnicos de engenharia antes da comercialização da Internet, não há sub-otimalidade num domínio hipotético de um browser que não implique em um

desvio gradual para protocolos crescentemente incompatíveis com o padrão *de jure* que se imponha como padrão *de facto* levando, só então, a uma sub-otimalidade do mercado na medida em que padrões proprietários impedem a competição.

É nossa opinião normativa – compartilhada por órgãos técnicos responsáveis por protocolos *de jure* como o IEEE, o IETF, o W3C – que um esforço regulatório informado pela teoria econômica da fricção técnica deveria se concentrar sobre problemas específicos de compatibilidade e interação técnica, e não sobre seus efeitos observados. Em outras palavras, faz mais sentido corrigir o mercado e deixá-lo agir que agir sobre seus resultados para corrigir problemas subjacentes.

No caso europeu da inclusão de *software* para a reprodução de filmes e música no sistema operacional Windows, a falha de mercado concreta não está em que a Microsoft controle também o mercado de reprodução de mídia em PCs, mas que os formatos em que filmes e música são distribuídos seja proprietário à Microsoft e inacessível em sistemas operacionais concorrentes.

1.7 Progresso tecnológico e produtividade

É difícil negar o impacto da informática e da computação sobre a produtividade da economia e logo sobre a capacidade da oferta de longo prazo. Os modelos mais tradicionais de desenvolvimento se apóiam em hipóteses simples para a dinâmica do progresso tecnológico, mas se sob a presença das falhas de mercado apontadas na indústria de computação os resultados obtidos são fortemente *path-dependent*, então é possível que grande parte do fator tecnológico presente nos modelos de desenvolvimento econômico a longo prazo seja fortemente não-linear – sensível a microeventos históricos.

Assim, não é improvável que a compreensão da dinâmica neste setor possa ser crítica para entender a dinâmica de mais longo prazo da produtividade e do crescimento econômico. Estimando econometricamente uma função de produção neoclássica "setorializada", [Oliner e Sichel 2000] atribuem à tecnologia da informação aproximadamente dois terços do crescimento da produtividade do trabalho nos Estados Unidos. Apenas com o comércio eletrônico, [Litan e Rivlin 2001] encontra um aumento de 0.4 pontos percentuais à tendência de crescimento do PIB potencial. Numa discussão mais conceitual, [Brynjolfsson e Hitt 2000] lista razões pelas quais a computação – através da transformação organizacional que traz no seu uso real típico – torna a operação das empresas mais eficiente.

1.8 Compatibilidade na indústria de software

Numa indústria densamente modular como a de computação, empresas e consumidores enfrentam opções de compatibilidade na formação das classes de produtos para as quais se estabelece um mercado. Discutimos nesta seção os aspectos técnicos da compatibilidade de software como subsídio para a discussão posterior sobre as opções estratégicas enfrentadas pela Apple Computer.

1.8.1 Para o usuário final

Compatibilidade binária

Programas vendidos para um sistema funcionam diretamente em outro, sem necessidade de recurso a nenhuma camada de emulação, nem acesso ao código-fonte – que pode conter *trade secrets* de como o programa funciona, e por isso em geral não é divulgado.

Exemplos: MS-DOS → Windows; Linux → Solaris.

Código-fonte

Com o código-fonte do programa, que revela todos os segredos de funcionamento do programa a qualquer programador "alfabetizado" na linguagem em questão, é possível compilar o programa para o novo sistema operacional sem alterações. Normalmente programas com código-fonte aberto estão disponíveis em praticamente qualquer plataforma.

Exemplos: Unix. Em particular, Linux → Mac OS X

Reimplementações de APIs por terceiros

O conceito de *application programmer interface* (API) é essencial para entender aspectos mais sutis da compatibilidade técnica na indústria de software. Essencialmente, uma API é um conjunto de funções pré-definidas que os programadores de aplicativos utilizam como peças modulares de seu código-fonte. Assim, se as APIs estão disponíveis em vários sistemas operacionais, em geral a portabilidade é trivial. No entanto, a maior parte das APIs são desenvolvidas juntamente com os sistemas operacionais e são específicas a eles (Win32/Windows, Cocoa/Mac OS X).

No entanto, o comportamento das APIs a que o programa chama pode, em tese ,ser reproduzido por um novo conjunto de APIs para o sistema alvo. Quando feito com pleno conhecimento das especificações, pode funcionar muito bem (subsistemas POSIX para Windows); um caso mais fraco, que constitui compatibilidade parcial é o de reimplementações feitas sem o conhecimento pleno de uma especificação técnica – como os módulos de compatibilidade com o Windows disponíveis para Linux – Wine, Crossover, Cedega.

Exemplos: Unix → Windows (Interix); Windows → Linux (Wine)

Emulação

O comportamento do *hardware* de uma máquina é simulado por software. Assim, o sistema operacional para o qual o programa foi compilado pode rodar por inteiro, e executar o programa. Baixa performance.

Exemplos: Windows → Mac (Virtual PC)

1.8.2 Para o fabricante de software

Código-fonte

Vantagens Possibilidade de auditoria para aplicações críticas governamentais e militares. Dado um padrão técnico amplo sem compatibilidade instalada (como o Unix), ampla base potencial de usuários sem esforço adicional de produção.

Desvantagens Revela o funcionamento interno do programa que pode conter segredos de algoritmos – não patenteáveis na maior parte do mundo.

Exemplos Linux, BSD, e grande parte do software para essas plataformas.

Distribuição apenas de binários; código-fonte escrito contra uma API padrão

Vantagens Protege os segredos contidos no código-fonte enquanto ainda garante uma ampla base instalada.

Desvantagens Esforço de compilar e dar suporte a todos os sistemas. APIs padrão podem ser limitadoras, e pode ser impossível construir o programa sem criar APIs próprias, já não caracterísitcas das plataformas subjacentes.

Exemplos Software comercial para Unix, em geral para grandes aplicações de engenharia.

Distribuição apenas de binários; código-fonte escrito contra uma API própria

Vantagens Não depende de padrões técnicos, freqüentemente defasados em relação às necessidades do software. Grande portabilidade.

Desvantagens A API própria precisa ser portada para cada plataforma-alvo. Necessidade de compilar e manter suporte para cada um dos sistemas alvo.

Exemplos Adobe Photoshop.

Distribuição apenas de binários; código-fonte escrito contra uma API presente em apenas um sistema.

Vantagens Maior facilidade de manter e dar suporte; evita-se o custo de lidar com muitas plataformas. Comportamento muito mais previsível. As APIs específicas de cada plataforma costumam também ser muito ricas e facilitar em muito o esforço de programação.

Desvantagens Nenhuma compatibilidade com outros sistemas, salvo sob emulação

Exemplos A grande maioria do software comercial para Windows ou Macintosh.

2
História e cultura dos paradigmas tecnológicos

2.1
A Lei de Moore

Em um profético artigo de meados da década de 60, o então engenheiro da Fairchild Electronics e futuro fundador da Intel Gordon Moore [Moore 1965] - postula que a complexidade de um circuito integrado, relativo ao custo dos componentes, duplicaria a cada 24 meses. Nas palavras de Moore,

> Integrated circuits will lead to such wonders as home computers – or at least terminals connected to a central computer – automatic controls for automobiles and personal portable communication equipment. (. . .) Computer will be more powerful, (. . .) built at lower costs and with faster turn-around.

As formulações modernas ([Research 2005]) da lei de Moore diminuem mais ainda o prazo de duplicação, a partir da observação empírica, e separam a previsão original em duas afirmações distintas e relacionadas:

1. O poder de computação concentrado em uma dada unidade de espaço duplica a cada 18 meses.

2. O poder de computação disponível a um dado custo duplica a cada 18 meses.

Na expressão de Christopher Evans (citado por [Dawkins 1998]),

> Today's car differs from those of the immediate post-war years on a number of counts . . . But suppose for a moment that the automobile industry had developed at the same rate as computers and over the same period: how much cheaper and more efficient would the current models be? If you have not already heard the analogy the answer is shattering. Today you would be able to buy a Rolls-Royce for 1.35

> pounds, it would do three million miles to the gallon, and it would deliver enough power to drive the Queen Elizabeth II. And if you were interested in miniaturization, you could place half a dozen of them on a pinhead.

A lei de Moore – em termos econômicos o postulado de que o progresso tecnológico no setor de eletrônicos é exponencial – veio a assumir uma significância cultural mítica, como profecia fundadora da explosão social da computação, e assume características de profecia auto-realizável, posto que os jogadores individuais da indústria assume que todos seus competidores são capazes de cumprir a profecia e logo investem enormes quantidades de recurso para se manter dentro dela.

Essa característica de profecia auto-realizável pode estar introduzido um viés na pesquisa tecnológica que prioriza a capacidade de computação sobre características inicialmente secundárias, mas que dada uma utilidade marginal decrescente para a velocidade e poder de computação ganham importância relativa maior, como o consumo de energia ou a dissipação térmica. Alguns comentaristas, como [Tuomi 2002] e [Twist 2005] especulam que pode estar havendo uma distorção ainda maior, dado que à medida que o custo por unidade de computação decai exponencialmente, o investimento necessário para cumprir a profecia da lei de Moore se torna cada vez mais caro.

2.2 O início da computação

Originalmente usado como sinônimo de "calculista" – uma pessoa cuja função é realizar operações aritméticas –, o uso do termo "computador" para dispositivos mecânicos de cálculo é registrado pelo *Oxford English Dictionary* a partir de 1897.

A ciência da computação costuma classificar computadores segundo sua capacidade. Por essa abordagem, computadores de uma única função surgem já na era clássica, como os mecanismos de engrenagens para calcular o movimento dos astros encontrados na ilha de Antikhytera, datados do século I a.C. Computadores capazes de avaliar um número limitado de funções surgem no século XIX, a partir do analisador diferencial de Charles Babbage. Finalmente, computadores de função geral podem resolver qualquer problema expresso na forma de um programa, dados os limites de capacidade de armazenamento.

Em um artigo [Turing 1936] de 1936, o matemático Alan Turing demonstra que um computador de função geral pode emular qualquer outro computador. Diz-se do dispositivo, sistema lógico ou linguagem de programação que goza desta

propriedade que é Turing-completo. Assim, adota-se a propriedade de Turing-completitude como a marca divisória pela qual se considera um dispositivo um computador moderno. Demonstrar que um dispositivo é de função geral é um procedimento matemático não-trivial, e assim, o privilégio de ter construído o primeiro computador moderno é freqüentemente reatribuído. Em 1998, provou-se [Rojas 1998] que o Z3 de Konrad Zuse, de 1941, é o computador Turing-completo mais antigo que conhecemos, mas o primeiro computador que foi construido sabendo-se ser Turing-completo foi o ENIAC de 1946.

A necessidade do computador emerge de problemas numéricos de grande porte, freqüentemente analiticamente insolúveis. Talvez o impulso mais importante à computação numérica na primeira metade do século XX tenha sido o método para obter soluções para a equação de Naviers-Stokes desenvolvido por Lewis Fry Richardson [Richardson 1922] em 1922, inoperável quando proposto e usado para a previsão metereológica até hoje. No entanto, sua aplicabilidade a problemas mais prosaicos de planejamento rapidamente se torna evidente, e poucos anos depois do ENIAC já começa a emergir uma indústria nascente de computação.

Paralelamente ao desenvolvimento dos computadores, prosperava toda uma indústria de máquinas de processamento automático de dados baseadas em cartão perfurado. Inventadas pelo estatístico Herman Hollerith para lidar com o problema de tabular o censo de 1890, elas em breve seriam usadas nas grandes empresas para a automação de problemas de contabilidade e administração de inventários. Essencialmente, o cartão agia como isolante elétrico, e quando perfurado, codificava uma informação a partir da qual a máquina (um dispositivo elétrico simples) abria um dentre vários compartimentos e armazenava o cartão, além de avançar um contador mecânico. A empresa que Hollerith fundou a partir do seu invento viria a se tornar a International Business Machines (IBM) [IBM History and Archives].

O primeiro computador comercialmente disponível – bem como o primeiro projetado e destinado para aplicações de processamento comercial de dados foi o UNIVAC I, produzido pela Remington Rand. Simbolicamente, o primeiro computador comercial da História é vendido ao US Census Bureau — como havia sido a primeira máquina de processamento automático de dados de Herman Hollerith – e 46 unidades foram vendidas a entidades como a Comissão de Energia Atômica e a Força Aérea americana, bem como a empresas como a Nielsen, a companhia de seguros Prudential, a General Electric e a DuPont.

No melhor estilo *arma virumque cano*, várias das questões de interação estratégica que emergem dos problemas de fricção técnica que estamos discutindo já surgem aqui. O UNIVAC competia diretamente com as máquinas de automação

comercial baseadas em cartão perfurado (feitas principalmente pela IBM), mas usava um sistema de armazenamento magnético décadas antes de seu tempo – quando muitas empresas já tinham muita informação armazenada em cartões, gerando um *switching cost* considerável [Unisys History Newsletter 2001]. Ao entrar no negócio de computadores, a IBM continuaria usando cartões perfurados como mídia para os novos equipamentos até meados da década de 70, e a Remington Rand seria obrigada a produzir conversores de fita magnética para cartão perfurado.

Por outro lado, a Remington Rand revela uma clarividência surpreendente quando adota uma estratégia reveladora de uma intuição quanto à questão das externalidades de rede ao adotar inicialmente o preço de US$159 mil – bem abaixo do custo de produzir a unidade – e ir aumentando gradualmente até chegar a US$ 1,5 milhão, sobrevivendo depois depois de ter estado à beira da falência e vindo a se tornar uma das sete companhias de computação de grande porte (além da IBM) que dominaram o mercado nos anos 60 e 70.

Mais importante, – embora não ligado diretamente aos problemas de fricção técnica – o poder e as possibilidades da propriedade de Turing-completitude são rapidamente percebidas. Não só a máquina originalmente concebida para cálculo numérico de grande porte rapidamente é adotada como substituto para os dispositivos mecânicos de processamento de dados, mas toda sorte de aplicações originais surgem nos primeiros anos. A DuPont usa o UNIVAC I para engenharia de projetos, estatística e econometria. A Igreja Episcopal completa em 300 horas uma edição com índice remissivo da Bíblia – trabalho que anteriormente levara 30 anos entre 1864 e 1894 para ser concluído. De forma muito clara, o impacto social da computação muito além de sua origem como facilitador de cálculo numérico revela-se explosivo mesmo dados os imensos custos de entrada e a escala mínima de operação imposta na década de 50.

2.3 O paradigma do time-sharing

Essa barreira à entrada cria uma demanda por um esquema que tornem mais direto e menos burocrático o acesso aos serviços de computação. O modelo monolítico dos primeiros mainframes implica em que apenas uma tarefa pode ser executada de cada vez, devendo ser inteiramente completada antes que outros possam trabalhar.

Numa instância clássica de simbiose entre a academia e as corporações – fenômeno que se repete inúmeras vezes ao longo da história desta indústria – as primeiras soluções de compartilhamento de um computador – i.e., de *time-*

sharing – surgem da demanda de cientistas da computação trabalhando em projetos acadêmicos de inteligência artificial, e são projetados e implementados em protótipo por esses cientistas.

[McCarthy 1983][1] narra o processo de adaptação de um computador existente (o IBM 7090) a um modelo de time-sharing, que envolve modificações de hardware propostas pelos cientistas do MIT e realizadas pelos engenheiros da IBM. O MIT também desenvolveria o primeiro sistema operacional para *time-sharing*, o CTSS.

Mais tarde, a Digital Equipment Corporation decidiria adotar o modelo de hardware para *time-sharing* projetado no MIT para o seu PDP-1, um computador que viria a fazer grande sucesso na academia e que é essencial no surgimento do Unix.

Com o amadurecimento da idéia do *time-sharing* na computação comercial, um consórcio de empresas se formaria para o desenvolvimento de um novo sistema operacional, o Multics. Quando a AT&T (e seu Bell Labs, um laboratório de pesquisa com muitas características de centro acadêmico) se retiram do consórcio Multics, alguns engenheiros e cientistas passam a trabalhar em um sistema operacional experimental - que serviria, inicialmente, de campo de teste para idéias em design de sistemas operacionais – que recebeu o nome irônico de "Unics". Este sistema viria a se tornar o sistema operacional mais popular do mundo por décadas – esencialmente, até o advento da computação pessoal – e é um testemunho da importância de uma *tradição* como descrevemos anteriormente que, com a adoção do Unix para o novo sistema operacional da Microsoft, ainda seja o modelo dominante de design de sistemas operacionais.

O Unix também é marcado por um extremo pragmatismo e um distanciamento gradual das abstrações mais puristas da academia de ciência da computação. Nas palavras de [Raymond 2003],

> Where Multics had been a large project with thousands of pages of technical specifications written before the hardware arrived, the first running Unix code was brainstormed by three people and implemented by Ken Thompson in two days – on an obsolete machine that had been designed to be a graphics terminal for a "real" computer.

O Unix, o primeiro sistema operacional portável a qualquer arquitetura de hardware subjacente[2] se desenvolveria pelos dez anos seguintes no ambiente

[1] Significativamente, John McCarthy é também quem demonstrou que uma linguagem de programação inteira pode ser construída a partir de funções e operadores, sem as características "imperativas" da maioria das linguagens de programação que existem até hoje, e com isso cria o instrumental básico (a linguagem Lisp) para a pesquisa em inteligência artificial, sendo celebrado por [Graham] como o "Euclides da programação de computadores"

[2] Antes, sistemas operacionais eram escritos especificamente para um tipo de computador.

acadêmico, chegando a ser essencialmente o mesmo paradigma tecnológico reproduzido pelo Linux e pelo Macintosh pós-1997 por volta de 1979.

Em 1980, a Agência de Projetos de Pesquisa Avançados do Departamento de Defesa decide usar o Unix como base para o seu novo protocolo de comunicações, o TCP/IP, que viria a substituir o UDP usado na Arpanet – a rede militar de computadores na origem da Internet – citando a falta de flexibilidade dos sistemas operacionais comerciais disponíveis então. Ainda segundo [Raymond 2003], esta fusão salvaria as duas tecnologias da destruição em face da ascensão dos microcomputadores pessoais que, estando longe de ter o poder de computação dos minicomputadores e workstations de *time-sharing*, estavam assumindo o espaço fora da academia que se esperava que o Unix assumisse.

2.4 O advento da computação pessoal

O primeiro computador pessoal de que se tem notícias é o kit de construção do MITS Altair 8800 de 1975. Um pacote de peças eletrônicas com as quais um computador – uma unidade de processamento, sem memória, armazenamento, entrada ou saída – pdoa ser montado, seguindo instruções a serem publicadas na revista *Popular Electronics* ao longo de meses. Apesar da sua extrema complexidade (o projeto requeria que o comprador soldasse centenas de fios) e da necessidade de complementos caros para qualquer trabalho sério – o computador precisava de um dispositivo de entrada e saída externo, e mesmo a cinco vezes o preço do computador em si, esses dispositivos só estariam disponíveis em um ano, as vendas excederam completamente as expectativas em dez vezes. Com suas aproximadamente 2000 unidades vendidas, o Altair era o modelo individual de computador mais vendido da História, e mostrou a existência de uma demanda reprimida por computadores de menor porte mesmo dadas custos draconianos de entrada em qualificação. O Altair se comunicava com um dispositivo de entrada e saída similar às workstations de *time-sharing*, um teletipo ASR-33 que combinava um teclado, uma impressora e um leitor/perfurador de cartões.

A MITS comprara para este computador um arranjo de licenciamento via *royalties* para um interpretador da linguagem de programação BASIC escrito por um jovem programador de 17 anos chamado Bill Gates. Divergências surgidas na contabilidade dos royalties – que não contavam as cópias "compartilhadas" em grupos de usuários – levariam o jovem Gates a formar uma empresa para vender software diretamente aos usuários finais[3], iniciando também o modelo de

[3]Essa empresa, a Microsoft, viria a se tornar o jogador dominante na indústria de software por mais de 20 anos, e seu virtual monopólio em uma ampla gama de linhas de produto parece

distribuição de software que ainda é o paradigma da indústria hoje em dia.

O Apple II, o primeiro modelo comercial da Apple Computer, representa um passo além. A arquitetura física de um computador pessoal moderno é essencialmente definida num salto discreto quando do aparecimento desta máquina – monitor, teclado, armazenamento e processamento numa unidade — um grande salto dos computadores de *time-sharing* operados através de teletipos ligados remotamente a uma unidade de processamento central, ou dos kits para computador pessoal populares nos anos anteriores.

Acessível pelo preço a uma família de classe média – apenas 20 anos depois do Univac I – o Apple II essencialmente inicia a revolução da computação pessoal, vindo a trazer os dois produtos de software que permanecem ainda hoje como usos essenciais da computação – o processador de texto e a planilha eletrônica. Devido a esta última, computadores pessoais da Apple também entrariam no mundo corporativo, iniciando uma cadeia de mudanças nos procedimentos organizacionais nas empresas cujos resultados são vistos em contraste agora.

O sucesso do Apple II não só criaria toda uma indústria de computadores pessoais no fim dos anos 70, como acabaria levando a IBM a criar o IBM PC, que se tornariam avassaladoramente bem-sucedido a ponto de levar quase toda a indústria de *hardware* a se concentrar em produzir computadores compatíveis com a linha. A IBM acabaria por perder o controle do design técnico da linha "IBM-compatível" à medida que o que no início eram clones ganham poder de mercado. Atualmente, o PC dessa linha é definido virtualmente pelo sistema operacional, no sentido em que a decisão de compra de hardware se orienta pela demanda por equipamento compatível com o Microsoft Windows.

O PC moderno segue um padrão *de facto*, mas não tem uma entidade organizadora que defina sua tecnologia. De maneira simplificada, pode-se considerar que é uma combinação de um conjunto de instruções para o processador (o x86, inicialmente proprietário da Intel mas clonado por outras companhias como a AMD, a Cyrix e a Transmeta), um conjunto de interfaces para periféricos (ISA, PCI, IDE, ATA, SCSI, USB) e um conjunto de sistemas operacionais comuns – atualmente, essencialmente o Windows. Este arranjo fantasticamente descentralizado tem a vantagem social de produzir um mercado commoditizado em que as firmas derivam pouco excedente econômico, num resultado próximo ao de competição perfeita, mas gera um centro de gravidade tecnológico para o rumo da indústria do qual é dificílimo sair. E de fato a diversidade virtualmente infinita de configurações de hardware tem conseqüências sérias sobre a complexidade e a confiabilidade de um sistema operacional moderno, que deve incluir uma coleção de *drivers* (módulos de software que se comunicam com os periféricos do computa-

inabalável até hoje.

dor) extremamente abrangente, o que por sua vez impõe uma barreira de entrada no mercado de sistemas operacionais significativamente maior que o problema da compatibilidade com os aplicativos.

É possível enxergar este problema como um processo composto de *path-dependence*: por externalidades de rede, o sistema operacional dominante é escolhido; os fabricantes de hardware optam por participar da rede fazendo seus periféricos compatíveis com o sistema operacional dominante, ou incorrem no custo de produzir *drivers* para ele – onde a demanda por sistemas alternativos em geral não é o suficiente para que as firmas de hardware se ocupem disso – o que torna o investimento em hardware algo específico e complementar ao sistema operacional, gerando *lock-in* e mais externalidades de rede.

Esta característica "topológica" da rede que constitui um PC é a falha essencial do argumento de *divided technical leadership* de [Bresnahan 2001]; onde este enxerga uma espécie de simbiose modular, existe na verdade uma relação de comensalismo que impede a competição no nível mais alto da cadeia. Para alguns analistas e jogadores da indústria, a relação do sistema operacional com a ecologia de fabricantes de hardware é ainda mais importante que o problema de compatibilidade com aplicativos geralmente enfatizado na literatura. É célebre, por exemplo, a declaração de Marc Andreessen, fundador da Netscape e um dos iniciadores do movimento contemporâneo de separar as APIs do sistema operacional subjacente, de que o Windows não passa de um conjunto de drivers ([Schmaleense 2000]).

O arranjo alternativo – a integração vertical dos componentes de um sistema de computação em um bloco monolítico e densamente integrado é putativamente superior em termos técnicos – e vigora como arranjo dominante nos primeiros 30 anos da computação mas traz o forte custo social do grande *lock-in* ocorrido quando do compromisso com uma solução tecnológica interamente proprietária.

2.5 A Era da Internet

Criada para aplicações militares e gestada por mais de uma década como uma rede acadêmica de computadores, a Internet se torna comercialmente disponível por volta de 1994. Concebida desde seu início com protocolos técnicos abertos e amplamente divulgados através do mecanismo dos RFCs, a rede mundial transforma em grande medida a relação de produção e consumo desses padrões tecnológicos. Em linhas gerais, esse fenômeno pode ser caracterizado por

1. Uma aumento significativo da demanda por computadores dados todos os novos usos de comunicação e entretenimento. Significativamente, estando

além do controle proprietário das firmas, esses novos usos criados para a computação criam oportunidades de entrada para firmas que consigam obter compatibilidade com os protocolos abertos e amplamente documentados usados na Internet; a Microsoft controla o formato do Microsoft Word e gera excedente econômico com o *lock-in* associado, mas não o formato de texto usado em *web sites*, e virtualmente qualquer sistema operacional concorrente ao Windows pode fazer grande parte do que é o uso de um computador hoje.

2. Um conjunto de novas soluções para diversas aplicações, independentes de plataforma e não sujeitas ao *lock-in* ao sistema operacional. É cada vez mais comum mover aplicativos que teriam sido específicos a uma plataforma para um modelo distribuído apoaido nos protocolos técnicos abertos da Internet. Um formulário eletrônico de imposto de renda, por exemplo, que antes da Internet teria sido um programa executável para alguma plataforma em específico, hoje pode ser trivialmente colocado num site web.

3
Estratégia e incerteza para a Apple Computer

3.1
Características tecnológicas da linha Macintosh

A linha Macintosh consiste de um pequeno número de modelos de computadores de especificações técnicas bem-definidas , desenhados para rodar um sistema operacional produzido pela própria Apple Computer, o Mac OS X.

O processador usado pelos computadores da linha é uma versão para desktops da arquitetura projeta pela IBM para suas aplicações de maior porte também, o PowerPC. O Mac OS X, um Unix herdeiro da versão de Berkeley do sistema, procura conciliar conceitos caros à era "clássica" do Mac como transparência e representação visual de conceitos com a tradição acumulada do sistema Unix subjacente.

Por rodar em um computador com um processador diferente, o Mac OS X está, em tese, sujeito a restrições de compatibilidade com aplicativos ainda mais sérias que as enfrentadas por sistemas operacionais alternativos para a plataforma x86 como o Linux, visto que não basta reimplementar as APIs do Windows para que seja possível executar aplicativos para tal sistema.

Este fato tem sido contrabalançado pela ampla disponibilidade de software profissional para o Mac em paralelo às versões Windows. A maior parte deste software gira em torno às aplicações específicas populares do Mac, em setores de criação – Pro Tools, Pagemaker/InDesign, Photoshop, Final Cut – mas algum software para desktops domésticos e corporativos mais usuais também se encontra disponível – notavelmente, o Microsoft Office.

3.2
Opções tecnológicas enfrentadas pela Apple Computer

A adoção do Unix como plataforma-base para a nova versão do sistema operacional do Macintosh pode ser explicada pelas questões de compatibilidade gerando fricção técnica que discutmos no primeiro capítulo desta monografia.

Postulamos, assim, um *trade-off* entre retornos de externalidade de rede, ao se inserir – ao menos parcialmente – na rede mais ampla dos computadores pessoais e os retornos com o *lock-in* à plataforma Mac.

3.2.1 Opções tecnológicas e demanda

A Apple e o padrão Unix

O principal mercado consumidor da Apple continua sendo o de profissionais de cinema, áudio, fotografia e editoração. No entanto, com a entrada na rede de *workstations* Unix, o Macintosh tem atraído boa parte da comunidade científica que realiza trabalhos de computação numérica. Significativamente, a Apple tem feito incursões até no restrito mercado de supercomputadores, o que representa se não uma mudança de foco, uma diversificação surpreendente do *core business* da companhia, classicamente associada à facilidade de uso para profissionais criativos.

Por outro lado, a compatibilidade com o Unix desincentiva o desenvolvimento de programas específicos para o Macintosh, e com isso diminui os custos de troca para usuários atuais de aplicações científicas, o que reduz um grande retorno de *lock-in*, ao menos para esse mercado. É também significativo para os custos de troca enfrentados pelos usuários de Macintosh que grande parte dos softwares profissionais originalmente disponíveis apenas para computadores Apple foram gradualmente ganhando versões para o Microsoft Windows.

Uma terceira alternativa possível seria afastar gradualmente o Mac OS X da compatibilidade plena com outros sistemas Unix à medida que se torna dominante nesse segmento do mercado. Notoriamente, a última versão do sistema operacional já introduz modificações nas camadas mais internas do sistema – inclusive no *kernel* XNU – para dar suporte aos novos recursos de busca da interface gráfica.

A Apple e o PowerPC

Quando da adoção do processador PowerPC em meados dos anos 90, esta linha encontrava-se tecnologicamente muito à frente da linha x86. Esse lag tecnológico foi se fechando ao longo da última década, ao mesmo tempo que o processador PowerPC não conseguia ganhar espaço no mercado de workstations dada a penetração de desktops cada vez mais poderosos.

Em junho de 2005, a Apple anunciou que a partir de 2007 realizaria a transição gradual da linha Macintosh para processadores Intel. Isso repre-

senta quebrar a compatibilidade com os computadores Macintosh da última década, mas também ganhar em opções de compatibilidade com as plataformas disponíveis sobre o processador x86 – em especial o Windows.

Trata-se de uma opção tecnológica arriscada. Se um Macintosh consegue rodar programas Windows ou o sistema operacional Windows inteiro, os custos de troca incorridos por um usuário contumaz de software Windows na hora de abandonar a linha Macintosh são virtualmente nulos A Apple enfrenta o problema de incentivar o desenvolvimento de software específico para o Macintosh, que não rodará em outros sistemas operacionais disponíveis para processadores x86, dado que com uma camada de compatibilidade Windows é preferível produzir diretamente para Windows e atingir os usuários de ambos sistemas.

3.2.2 O problema de supply chain

Uma das razões que pode ter motivado a Apple a trocar de tecnologia de processadores pode estar ligado ao seu desconfortável papel de oligopsonista nesse mercado. Para ser capaz de efetuar uma expansão das vendas além de certo patamar, a Apple precisa obter o compromisso de fornecedores que podem estar menos dispostos a adaptar plantas de produção para o hardware altamente específico de um Macintosh.

Por outro lado, um fornecedor dos componentes mais sofisticados – como a IBM, que fabrica os processadores PowerPC para a Apple – pode estar menos disposto a continuar fazendo pesquisa e desenvolvimento dado o mercado restrito da produção. Isso explicaria o fechamento do gap tecnológico entre a linha PowerPC e os processadores Intel encontrados nos computadores comuns ao longo dos últimos dez anos.

A troca para processadores Intel aliviaria esses dois problemas, na medida em que a Apple pode contar com o volume de prodção da Intel para aumentar sua própria linha de produção, e pode contar com o esforço de pesquisa e desenvolvimento da Intel que frutifica em um mercado bem mais amplo.

3.2.3 Interação com mercados correlatos

A Apple Computer vem desenvolvendo um negócio paralelo de grande sucesso financeiro – atualmente bem maior do que o obtido com os computadores Macintosh – na distribuição online de música e reprodutores portáteis de áudio.

Um fenômeno explosivo de 7 trimestres para cá, a combinação do reprodutor iPod com o software iTunes afirmou-se rapidamente como um novo paradigma tecnológico na distribuição, armazenamento e reprodução de entretenimento, dando extrema visibilidade à Apple no mercado consumidor *mainstream*.

O iPod tem complementaridades interessantes com os computadores Macintosh, embora seja compatível com PCs commoditizados. Uma questão empírica que essa complementaridade levanta, e que não pôde ser estudada devido à escassez de dados tão recentes, é se a explosão nas vendas de eletrônicos para consumidor da Apple tem algum efeito sobre as vendas da linha Macintosh.

Como firma de eletrônicos de consumidor, a Apple não tem a opção de ser incompatível com o PC sob pena de ser substituída rapidamente por clones no mercado de reprodutores portáteis de música digital, e não há razão técnica pura para que um usuário de iPod deseje um Macintosh.

No entanto, a exposição à marca e às tecnologias da empresa, bem como a percepção pública da viabilidade da Apple no mercado mais amplo de tecnologia, podem ter grande efeito se corretamente administradas.

Além disso, o fluxo de caixa gerado pelo iPod pode ser quase inteiramente investido em pesquisa e desenvolvimento para o Macintosh, que seria virtualmente subsidiado pelas margens de lucro no setor de música até uma possível expansão e conquista do mercado.

3.3 Conclusões e possibilidades estratégicas futuras

O senso comum, sancionado por parte da literatura tanto de economia da computação como de negócios e estratégia, sugere que a Apple e o Macintosh são vítimas do processo de feedback positivo inerente à peculiar estrutura tecnológica da indústria de computação.

No entanto, uma análise mais atenta dos problemas tecnológicos relevantes para a produçcão desses problemas de fricção técnica gerando *path-dependence* e feedback positivo revela opções mais sutis de estratégia sendo feitas pela administração da empresa que podem passar desapercebidas sem uma análise atenta do problema tecnológico de compatibilidade.

Acreditamos que o julgamento sumário de [West 2004] (resumido na expressão "Betamax redux") não é compatível com uma análise mais atenta da modularidade envolvida nas opções tecnológicas que formam as estratégias das firmas no mercado de computação pessoal, bem como da história e estrutura de outros segmentos do mercado mais amplo de computação com o qual cada vez mais se confunde dado o mecanismo aparentemente inexorável da lei de Moore.

Pensamos, finalmente, com base nos estudos empíricos citados e uma visão mais ampla dos rumos do progresso tecnológico, que indústrias sujeitas a *path-dependence* e fricção técnica terão cada vez mais peso na economia como um todo, e a compreensão dos fenômenos de fricção técnica, bem como das opções tecnológicas subjacentes a esses fenômenos crescerá cada vez mais em importância ao entender o processo mais longo de desenvolvimento econômico.

Referências Bibliográficas

[Beggs e Klemperer 1992]BEGGS, A.; KLEMPERER, P. Multi-period competition with switching costs. *Econometrica*, v. 60, p. 651–666, May 1992.

[Bresnahan 2001]BRESNAHAN, T. The right remedy. 2001. Disponível em: <http://www.stanford.edu/ tbres/Microsoft/The%20Right%20Remedy.pdf>.

[Bresnahan 2002a]BRESNAHAN, T. Economist amicus curiae brief. *United States District Court for the District of Columbia*, 2002a.

[Bresnahan 2002b]BRESNAHAN, T. A remedy that falls short of restoring competition. *Antitrust*, 2002b. Disponível em: <http://www.stanford.edu/ tbres/Microsoft/anti-bre.pdf>.

[Brynjolfsson e Hitt 2000]BRYNJOLFSSON, E.; HITT, L. M. Beyond computation: Information technology, organizational transformation and business performance. *Journal of Economic Perspectives*, 2000.

[Cusumano, Mylondais e Rosenbloom 1992]CUSUMANO, M. A.; MYLONDAIS, Y.; ROSENBLOOM, R. S. Strategic maneuvering and mass-market dynamics: The triumph of vhs over beta. *Business History Review*, v. 66, 1992.

[David 1985]DAVID, P. A. Clio and the economics of qwerty. *American Economic Review*, v. 75, 1985.

[David 2000]DAVID, P. A. Path dependence, its critics and the quest for 'historical economics. *Evolution and Path Dependence in Economic Ideas: Past and Present*, 2000.

[Dawkins 1998]DAWKINS, R. Science and sensibility. In: LECTURE, Q. E. H. (Ed.). *Sounding the Century - What will the 20th century leave to its heirs?* [s.n.], 1998. Disponível em: <http://www.simonyi.ox.ac.uk/dawkins/WorldOfDawkins-archive/Dawkins/Work/Articles/1998-03-24science_and_sensibility.shtml>.

[Economides 1996]ECONOMIDES, N. The economics of networks. *International Journal of Industrial Organization*, 10 1996.

[Edstrom e Eller 1998]EDSTROM, J.; ELLER, M. *Barbarians led by Bill Gates*. [S.l.]: Henry Holt, 1998.

[Farrell e Shapiro 1989]FARRELL, J.; SHAPIRO, C. Optimal contracts with lock-in. *American Economic Review*, 1989.

[Graham]GRAHAM, P. The roots of lisp. Disponível em: <http://www.paulgraham.com/rootsoflisp.html>.

[Hagedoom, Carayannis e Alexander 2001]HAGEDOOM, J.; CARAYANNIS, E.; ALEXANDER, J. Strange bedfellows in the personal computer industry: technology alliances between ibm and apple. *Research Policy*, 2001.

[IBM History and Archives]IBM History and Archives. *http://www.ibm.com/ibm/history/*.

[Jackson e Perloff 1996]JACKSON, T. W.; PERLOFF, J. M. Personal computer brand loyalty. *Dept. of Agricultural and Resource Economics Working Papers, Univ. of California at Berkeley*, 1996. Disponível em: <http://ideas.repec.org/p/are/cudare/790.html>.

[Katz e Shapiro 1985]KATZ, M.; SHAPIRO, C. Network externalities, competition and compatibility. *American Economic Review*, v. 73, 1985.

[Klemperer 1987]KLEMPERER, P. The competitiveness of markets with switching costs. *RAND Journal of Economics*, v. 18, 1987.

[Kurzweil 2000]KURZWEIL, R. *The Age of Spiritual Machines: when computers exceed human intelligence*. [S.l.]: Penguin, 2000.

[Kurzweil 2001]KURZWEIL, R. The law of accelerating returns. *KurzweilAI Research*, 2001. Disponível em: <http://www.kurzweilai.net/articles/art0134.html?printable=1>.

[Liebowitz e Margolis 1994]LIEBOWITZ, S.; MARGOLIS, S. E. The fable of the keys. *Journal of Law and Economics*, v. 33, 1994.

[Liebowitz e Margolis 1999]LIEBOWITZ, S.; MARGOLIS, S. E. *Winners, Losers & Microsoft: Competition and Antitrust in High Technology*. [S.l.: s.n.], 1999.

[Litan e Rivlin 2001]LITAN, R. E.; RIVLIN, A. M. Projecting the economic impact of the internet. *American Economic Review*, Papers and proceedings of the 130th annual meeting of the American Economic Association, 2001.

[McCarthy 1983]MCCARTHY, J. Reminiscenses on the history of time-sharing. 1983. Disponível em: <http://www-formal.stanford.edu/jmc/history/timesharing/timesharing.html>.

[Moore 1965]MOORE, G. E. Cramming more components onto integrated circuits. *Electronics Magazine*, 1965.

[Nelson e Winter 1977]NELSON, R.; WINTER, S. In search of an useful theory of innovation. *Research Policy*, v. 6, 1977.

[Oliner e Sichel 2000]OLINER, S. D.; SICHEL, D. E. The resurgence of growth in the late 1990s: Is informaiton technology the srory? *Journal of Economic Perspectives*, 2000.

[Raymond 2003]RAYMOND, E. S. *The Art of Unix Programming*. [S.l.]: Addison-Wesley, 2003.

[Raymond 2003]RAYMOND, E. S. The art of unix programming. In: _____. *The Art of Unix Programming*. [S.l.]: Addison-Wesley, 2003. cap. 20.

[Research 2005]RESEARCH, I. Moore's law. *http://www.intel.com/research/silicon/mooreslaw.htm*, 2005.

[Richardson 1922]RICHARDSON, L. F. *Weather prediction by numerical process*. [S.l.]: Cambridge University Press, 1922.

[Rojas 1998]ROJAS, R. How to make zuze's z3 a universal computer. *IEEE Annals of the History of Computing*, 1998.

[Schmaleense 2000]SCHMALEENSE, R. Antitrust issues in schumpeterian industries. *American Economic Review*, v. 90, 2000. Disponível em: <http://ideas.repec.org/a/aea/aecrev/v90y2000i2p192-196.html>.

[Schnaars 1994]SCHNAARS, S. P. *Managing Imitation Strategies*. [S.l.]: The Free Press, Macmillan Inc., 1994.

[Shapiro e Varian 1999]SHAPIRO, C.; VARIAN, H. *Information Rules: a strategic guide to the network economy*. [S.l.]: Harvard Business School Press, 1999.

[Spolsky 2002]SPOLSKY, J. *Strategy Letter V*. [S.l.], 2002. Disponível em: <http://www.joelonsoftware.com/printerFriendly/articles/StrategyLetterV.html>.

[Stiglitz 2000]STIGLITZ, J. *Economics of the Public Sector*. [S.l.]: W.W. Norton & Company, 2000.

[Tuomi 2002]TUOMI, I. The lives and deaths of moore's law. *First Monday*, v. 7, n. 11, Novembro 2002. Disponível em: <http://firstmonday.org/issues/issue7_11/tuomi/index.html>.

[Turing 1936]TURING, A. On computable numbers: with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 1936.

[Twist 2005]TWIST, J. The law that has driven digital life. *BBC News*, 2005. Disponível em: <http://news.bbc.co.uk/2/hi/science/nature/4449711.stm>.

[Unisys History Newsletter]UNISYS History Newsletter. Disponível em: <http://www.cc.gatech.edu/services/unisys-folklore/>.

[Unisys History Newsletter 2001]UNISYS History Newsletter. v. 5, n. 1, Jan 2001. Disponível em: <http://www.cc.gatech.edu/gvu/people/randy.carpenter/folklore/v5n1.html>.

[Varian 2003]VARIAN, H. R. *Microeconomia - Princípios Básicos*. [S.l.]: Campus, 2003.

[West 2004]WEST, J. Strategy in transition. In: _____. [S.l.]: Oxford, 2004. cap. The fall of a Silicon Valley Icon: was Apple really Betamax redux.

As referências à Internet foram todas conferidas e estavam disponíveis no dia 25 de junho de 2005.