Computação Quântica: uma perspectiva de Computação Paralela

William Alexandre Miura Gnann[1]

[1]Instituto de Matemática e Estatística – Universidade de São Paulo
São Paulo – SP

gnann@ime.usp.br

Sumário

1. Introdução

De uns tempos para cá, sobretudo depois de os fabricantes de processador praticamente chegarem ao limite físico de *clock*, a ideia de se paralelizar processamento ganhou cada vez mais força. Hoje em dia existem diversas tecnologias para paralelizar indo desde processadores com alguns núcleos até redes com milhares de computadores contendo eventualmente aceleradoras com milhares de núcleos com um poder de processamento que parece oriundo de histórias de Asimov.

No começo dos anos 2000, ainda era comum os fabricantes de processador compararem seu produto pelo *clock* - quantidade de instruções que ele consegue executar em um segundo. Contudo, depois de anos de desenvolvimento, chegou-se ao momento onde o esforço para executar mais instruções por ciclo era monumental, pois a temperatura era alta o suficiente para não garantir precisão no processamento. Os fabricantes decidiram trabalhar outros aspectos na arquitetura dos processadores, como, por exemplo, trabalhar com mais núcleos. Outra tendência que se desenvolveu bastante desde os anos 2000 foi o uso de processadores gráficos. O que antigamente era usado para arte gráfica ou para os maravilhosos *games* passou, também, a ser visto como uma forma extremamente poderosa de se fazer processamento.

Os processadores gráficos que compõem as placas gráficas surgiram como especialistas em fazer conta, afinal, a computação gráfica, grosso modo, pode ser reduzida a um sem fim de contas. Um exemplo dessas contas é determinar novas coordenadas para algum objeto a partir de um espaço tridimensional. Podemos organizar essas contas para que elas sejam facilmente sejam executadas em paralelo. Com efeito, se tivéssemos n pessoas com calculadoras poderíamos dar uma linha da matriz e uma cópia do vetor para cada uma dessas pessoas e cada uma delas conseguiria determinar o trecho das novas coordenadas. Poderíamos, ainda, dividir as parcelas que sobraram entre as pessoas de modo que elas concluíssem outra parte das contas em paralelo. Faríamos essa redução até restar o vetor final. Supondo todas essas pessoas saudáveis, elas quase certamente venceriam

uma competição contra apenas uma pessoa que também teria de fazer todas as contas usando uma calculadora simples. Chamamos esse ganho de desempenho de *speedup*.

Paralelamente, a Computação Quântica também vem se desenvolvendo, embora ainda não se tenha conhecimento de um computador quântico completo. Alguns fabricantes alegam possuir modelos com 1000 *qubits* hoje em dia, contudo diversos pesquisadores discordam da eficácia desses modelos cheios de *qubits*. Até o presente momento, não há evidência de que se tenha atingido o *speedup* nesses computadores quânticos badalados.

A ideia Computação Quântica foi introduzida por Richard Feynman no artigo "Simulating Physics with Computers"[7] onde o autor discute o problema em simular fenômenos quânticos com computadores classicos em termos de desempenho. Para simular um sistema quântico com n variáveis seria necessária a quantidade proibitiva de 2^n variáveis num computador comum! Logo a maneira natural de simular um sistema quântico deve ser justamente utilizar um computador que consiga tirar proveito das propriedades quânticas.

Para entendermos melhor o modelo quântico, precisamos antes visitar alguns conceitos que ajudam a compreender toda essa maluquice. O primeiro deles é o de transformações lineares.

Uma transformação linear é uma função, $T : A \mapsto B$, essencialmente com duas propriedades:

1. $T(v + w) = T(v) + T(w)$
2. $T(\alpha v) = \alpha T(v)$, com α escalar

No contexto que estamos estudando, podemos representar tais transformações por matrizes e o objeto onde as aplicamos como vetores. Por exemplo:

$$\underset{T}{\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}} \underset{v}{\begin{pmatrix} 1 \\ 0 \end{pmatrix}} = \underset{T(v)}{\begin{pmatrix} 1 \\ 1 \end{pmatrix}}$$

Elas aparecem em modelos utilizados em diversas áreas como Biologia, Economia, Física, Probabilidade etc. A partir dessa poderosa notação, conseguimos modelar processos como a transição probabilística entre estados, um desses modelos são as Cadeias de Markov que será visto adiante.

2. Uma pitada de probabilidade

No dia-a-dia, costumamos inferir de forma completamente amadora se irá, ou não, chover. É mais esperado, por exemplo, que chova num dia nublado do que num dia de céu de brigadeiro. Isso induz um modelo simples com três estados: *ensolarado*, *nublado* e *chuvoso*.

Naturalmente, poderíamos utilizar as queridas transformações linerares para simular esse modelo e conseguir "prever" se vai chover amanhã com perguntas do tipo: *"Dado que está chovendo hoje, será que vai chover amanhã?"*

Na figura acima, a coordenada de cima do vetor representa o estado "ensolarado", a do meio, o estado "nublado" e a de baixo, o estado "chuvoso". Ou seja, depois do primeiro dia chuvoso, a chance de um novo dia chuvoso será de 0.4.

	ensolarado	nublado	chuvoso
ensolarado	0.5	0.4	0.1
nublado	0.25	0.5	0.25
chuvoso	0.3	0.3	0.4

Tabela 1. Probabilidades de transição de estados.

$$\begin{pmatrix} 0.5 & 0.25 & 0.3 \\ 0.4 & 0.5 & 0.3 \\ 0.1 & 0.25 & 0.4 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.3 \\ 0.4 \\ 0.4 \end{pmatrix}$$

Figura 1. Exemplo do modelo de previsão de chuva.

Para saber a probabilidade de chover num futuro mais distante, podemos simplesmente aplicar a transformação a quantidade desejada de vezes, n, e obteremos as probabilidades para depois de n dias[1].

$$\begin{pmatrix} 0.5 & 0.25 & 0.3 \\ 0.4 & 0.5 & 0.3 \\ 0.1 & 0.25 & 0.4 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.34884 \\ 0.41860 \\ 0.23256 \end{pmatrix}$$

Figura 2. Previsão de chuva para 10 dias no futuro.

É importante destacar que não é qualquer transformação linear que serve para modelar esse tipo de processo probabilístico e, naturalmente, não é qualquer vetor ao qual podemos aplicar tais transformações. Todos eles precisam respeitar a restrição de que representam probabilidades, logo a soma de todos os elementos do vetor deve ser 1[2] e o análogo ocorre com as colunas da matriz.

O leitor mais curioso já percebeu que nosso modelo, embora limitado, tem uma característica curiosa: a partir de um dado dia, não há mais mudança no vetor de probabilidades. Chamamos esse vetor que não se altera de distribuição estacionária. Essa propriedade não é exclusiva do nosso modelo, ela se manifesta em diversas cadeias de Markov.

3. Um punhado de interferência

Os fenômenos ondulatórios desempenham um papel importantíssimo hoje em dia: diversos meios de comunicação, como a Internet, fazem uso de suas propriedades. Contudo, fazer um uso eficiente desse meio traz uma série de desafios como garantir que os dados enviados pelas ondas cheguem corretamente do outro lado, pois há diversos empecilhos como o meio de propagação, a potência de emissão e, principalmente, a interferência.

As ondas quando em propagação funcionam como se fossem somadas umas as outras, logo é de se imaginar que duas ondas iguais intensifiquem a intensidade e que

[1] Ou n transições de estado.

[2] Claro que todos os elementos devem ser não negativos.

duas ondas completamente opostas se anulem. A interferência é justamente essa relação que acontece entre as ondas.

4. Uma dose de quântica

Provavelmente, algumas pontas ficaram soltas até o presente momento no texto. O que é um *qubit*? Por que existe essa necessidade de trabalhar com uma quantidade tão grande de variáveis para simular um sistema quântico? Será que esses computadores "mágicos" conseguem, de fato, algum *speedup* se comparados aos primos clássicos?

Primeiramente, o modelo de quântica assemelha-se muito à probabilidade, contudo, existem alguns fatores que diferem, e muito, a quântica da mera probabilidade. Os estados quânticos são representados por números complexos e não por números reais entre 0 e 1, portanto, é possível que haja interferência entre estados quando aplicamos alguma transformação sobre eles. Repetindo: é possível que dois estados quânticos se anulem depois de uma transformação tal qual duas ondas!

Um sistema quântico, assim como os estados em probabilidade, pode ter seus estados representados por um vetor. Entretanto, cada uma das combinações possíveis de estados requer um representante no "vetor quântico". Como precisamos de *todas* as combinações, precisaremos, para n estados, de 2^n representantes. Isso nos permite chegar ao conceito de *qubit*.

Análogo ao *bit*, o *qubit* é a unidade básica de computação em computação quântica[3]. Como um *qubit* corresponde a um *bit*, precisaremos de um vetor complexo com 2^1 coordenadas. Denotamos as coordenadas pela notação *ket*, onde $|n\rangle$ representa cada um dos $lg(n) + 1$ estados.

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$$

Figura 3. Um *qubit* genérico.

Não podemos escolher qualquer valor para os α_i - chamados de fase. Para ser consistente com os experimentos de física quântica, é preciso, ainda, que os α_i sejam tais que a soma dos quadrado de seus módulos seja 1. Ou seja, para cada número complexo α_i, somamos a sua magnitude e essa soma precisa ser 1. Fazendo a ponte com probabilidade, a probabilidade preserva a soma dos elementos - chamada de norma 1 - enquanto a quântica preserva a soma do quadrado dos elementos - chamada de norma 2.

No caso de um *qubit*:

$$|\alpha_0| + |\alpha_1| = 1$$

Naturalmente, podemos associar diversos *qubits* para formar registradores quânticos. Novamente, eles serão representados por uma quantidade exponencial de estados e sua norma 2 terá de ser 1.

Até poderíamos utilizar essa notação para trabalhar com computação clássica. O que muda é que à cada passo de computação somente um dos α_i poderia ser 1 enquanto

[3]Não há nenhum impedimento do ponto de vista físico para trabalharmos com mais do que 0 e 1 por *qubit*, contudo parece mais simples trabalhar, por ora, dessa forma.

$$|\psi\rangle = \alpha_0 |00\rangle + \alpha_1 |01\rangle + \alpha_2 |10\rangle + \alpha_3 |11\rangle$$

Figura 4. Sistema com dois *qubits*.

todos os outros deveriam ser 0. A propriedade de podermos trabalhar com mais de uma combinação de estados por passo de computação é chamada de *sobreposição*.

Dado que a norma 2 de um vetor quântico soma 1, podemos interpretá-la como uma probabilidade: a probabilidade de encontrarmos o estado $|n\rangle$ ao observarmos o sistema. Uma observação é justamente uma transformação não-reversível onde um sistema quântico colapsa para apenas uma combinação de estados.

Resta-nos, agora, conseguir transformar esses *bits* quânticos de alguma forma como transformamos os *bits* comuns através de portas lógicas. As portas "lógicas" quânticas comportam-se como transformações lineares. Mas não são quaisquer transformações lineares! Elas precisam preservar a norma 2, pois um *qubit* tem uma distribuição de probabilidade. Tais transformações são conhecidas por *unitárias*.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Figura 5. Transformação de Hadamard.

A transformação de Hadamard é uma das mais conhecidas e utilizadas em computação quântica. Sua principal aplicação é deixar um sistema quântico em sobreposição, não obstante também ser utilizada como intermediária para derivar outras transformações.

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Figura 6. Transformação de Hadamard sobre 1 *qubit*.

Tudo isso sugere um modelo razoavelmente simples para se fazer computação quântica.

1. Garantimos que o sistema ficará em sobreposição;
2. Aplicamos diversas transformações lineares de forma a maximizar a probabilidade de observar o estado correto. A interferência faz o seu papel nesse momento;
3. Observamos o sistema que colapsará, com alta probabilidade, para o estado correto.

Ou seja, o modelo de computação quântica nada mais é que uma forma de massagear os *qubits* a fim de aumentarmos a probabilidade de observar o desejado enquanto diminuímos a probabilidade de observar errado. Vamos ver como esse modelo se aplica ao Algoritmo de Grover.

5. Algoritmo de Grover

Inspirado, provavelmente, pelo modelo de computação descrito acima, nasceu o Algoritmo de Grover. Ele promete, e cumpre, encontrar um dado elemento, e, num banco

de dados fora de ordem em $O(\sqrt(n))$. Isso significa que consumiremos um tempo da ordem de $\sqrt(n)$ para encontrar um elemento num banco de dados de tamanho n. Classicamente, esse resultado demora tempo da ordem do tamanho do banco de dados, pois podemos resolver o problema meramente inspecionando todos os elementos.

Embora a promessa seja essa, o jeito de o algoritmo atuar é expressivamente diferente do algoritmo trivial. Primeiramente, o banco de dados é dado de maneira implícita, isto é, em vez de colocarmos a mão no banco para encontrarmos o elemento e, colocamos a mão numa função f definida como:

$$f(x) = \begin{cases} -x & \text{se } x = e \\ x & \text{c.c.} \end{cases}$$

Dada f podemos definir uma transformação U que inverterá o valor apenas quando $x = e$. Chamamos essa transformação de *oráculo quântico*.

$$\begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ ou } \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ ou } \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ ou } \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Figura 7. Exemplos de oráculo quântico com 2 *qubits*.

No algoritmo, a única função do oráculo é fazer o que ele faz: inverter a fase. Isso garante que haja uma distinção entre o elemento que desejamos em relação aos outros elementos, contudo, como não há alteração na probabilidade de se observar o elemento correto, precisaremos de mais processamento.

A próxima fase consiste essencialmente em tirar proveito da inversão de fase afetando diretamente a probabilidade de observar o índice correto. O grande truque é refletir as fases a partir do valor da média, μ, das fases. Caso o valor da fase seja superior à média, a diferença será subtraída, caso contrário, a diferença será adicionada. Logo a fase mais distante tende a ser justamente aquela que foi refletida, afinal um dos elementos teve o seu sinal trocado. Contudo, sem a devida configuração inicial, ainda não há garantia acerca do sucesso do algoritmo.

A reflexão pode ser feita a partir da transformação de Hadamard e de uma outra transformação, R, que é representada por uma matriz com 1 na primeira posição e -1 em todas as outras.

$$H \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} H$$

Figura 8. Transformação de reflexão pela média.

A título de curiosidade, utiliza-se, na literatura, a notação *braket* para representar R que fica com a seguinte cara: $2\left|0\right\rangle\left\langle 0\right| - I$.

Chamamos a sequência de inversão e reflexão de *difusão de Grover*. Ao fim de uma difusão, a probabilidade de observar o índice i, dada uma configuração inicial, é aumentada. Para garantir esse aumento, basta inicializar os *qubits* com fases todas iguais. Como a norma 2 deve somar 1, todas as fases serão iguais a $\frac{1}{\sqrt{n}}$. No fim de uma operação de difusão, o i-ésimo elemento distará de $\frac{2}{\sqrt{n}}$ dos outros.

Seja a média μ, α_i o i-ésimo elemento e α_k os outros elementos. A média μ será a soma de $n-1$ elementos iguais a $\frac{1}{\sqrt{n}}$ com o invertido $-\frac{1}{\sqrt{n}}$.

$$\mu = (n-1)\frac{1}{\sqrt{n}} - \frac{1}{\sqrt{n}} = \frac{n-2}{\sqrt{n}}$$

Agora, precisaremos calcular $2\mu - \alpha i$ e $2\mu - \alpha k$ para ver de quanto será o incremento.

$$\alpha_i^{novo} = 2\mu - \alpha_i = 2\frac{n-2}{\sqrt{n}} + \frac{1}{\sqrt{n}} = \frac{2n-4+1}{\sqrt{n}} = \frac{2n-3}{\sqrt{n}}$$

$$\alpha_k^{novo} = 2\mu - \alpha_k = 2\frac{n-2}{\sqrt{n}} - \frac{1}{\sqrt{n}} = \frac{2n-4-1}{\sqrt{n}} = \frac{2n-5}{\sqrt{n}}$$

$$\alpha_i^{novo} - \alpha_k^{novo} = \frac{2}{\sqrt{n}}$$

Resta-nos saber quantas vezes precisamos repetir esse procedimento a fim de maximizar a probabilidade de observar o estado correto. Foi provado que essa quantidade é de $\frac{\pi}{4}\sqrt{n}$. Se executarmos mais vezes, a amplitude será refletida para baixo, pois o comportamento da probabilidade em função do número de execuções é senoidal.

Portanto, o Algoritmo de Grover garante um *speedup* quadrático na tarefa de encontrar um elemento num banco de dados desordenado.

```
Grover(U)
    for i = 1 to len(q)
        q[i] <- 1/sqrt(n)
    for i = 1 to pi*(sqrt(n)/4)
        q <- inverte(q) #oráculo
        q <- reflete(q) #H*(2|0><0|-I)*H
    return observa(q)
```

6. Conclusão

Embora não se possa dizer que existe de fato "Computação Paralela" em Computação Quântica, existe um paralelismo natural quando levamos em conta o que é um *qubit*. Tudo isso, claro, devido principalmente à sobreposição das combinações de estado que faz tratarmos um conjunto de *bits* como um vetor de tamanho exponencial. Essa propriedade, aliás, constrói a classe BQP, *Bounded error, quantum, polynomial time*, que difere da conhecida classe P[4], pois oferece algoritmos como o de Grover com um claro *speedup* em relação às versões clássicas.

[4]A menos de $P = NP$.

A utilização de transformações lineares como forma de modificar os *qubits* sugere o uso de paralelismo para sua simulação, contudo, isso não resolve o tamanho exponencial das instâncias. Por outro lado, justifica a utilização de máquinas do TOP500 para a simulação de sistemas quânticos.

De qualquer forma, ainda hoje em dia a computação quântica engatinha. Já que os mais promissores modelos não apresentaram algoritmos com um aumento de velocidade considerável.

7. Referências

1. Scott Aaronson. 6.845 Quantum Complexity Theory, Fall 2010. (Massachusetts Institute of Technology: MIT OpenCourseWare), http://ocw.mit.edu (Accessed 21 Jun, 2015). License: Creative Commons BY-NC-SA
2. Lov K. Grover, Jaikumar Radhakrishnan - arXiv:quant-ph/0407122
3. http://twistedoakstufidios.com/blog/Post2644_grovers-quantum-search-algorithm
4. http://kukuruku.co/hub/quantum-computing/
5. http://www.hpcwire.com/off-the-wire/d-wave-systems-breaks-1000-qubit-quantum-computing-barrier/
6. http://phys.org/news/2014-06-independent-group-d-wave-quantum-spee html
7. http://www.cs.berkeley.edu/~christos/classics/Feynman.pdf
8. http://math.nist.gov/quantum/zoo/