Computadores auto replicantes

Em 1952, 7 anos após o final da II Grande Guerra, Turing já havia construído diversos computadores (a bomba em Bletchley Park, ACE e MUC em Manchester...) já havia transitado pela Inteligência Artificial (seu artigo seminal *Computing Machinery and intelligence* foi publicado em 1950) e agora publicou *The chemical basis of morphogenesis.* tratando do desenvolvimento de padrões em organismos por crescimento. O artigo propõe a questão: *Como as coisas crescem ?*, *Como a matéria ganha forma ?* Deve-se lembrar que nesse exato momento, em Cambridge, Crick e Watson estavam tentando resolver esse mesmo problema do ponto de vista microbiológico, tendo acabado por chegar à hélice dupla do DNA. Mas Turing, como matemático brilhante que era, estava atacando o problema de um ponto de vista matemático. É claro que só nos resta especular, mas o que mais teria saído da cabeça desse sujeito se ele não tivesse morrido em 1953, com pouco mais de 40 anos ?

Outra pessoa importante, John Von Neumann, prestes a morrer também (em 1957) inventou uma máquina que era capaz não apenas de computar qualquer função computável mas também de se auto-reproduzir. De maneira não surpreendente, o método proposto para a máquina gerar descendência tem semelhança com os métodos humanos. Ainda que existam apenas rascunhos dessa máquina de como implantar uma coleção enorme de 29 autômatos de estado. Mesmo que essa máquina tivesse sido construída, ela provavelmente não seria muito impressionante de observar, já que apenas os padrões de seus estados e não qualquer comportamento físico é que seriam reproduzidos. Eis o que Von Neumann tinha em mente: O plano é dividido em infinitas células quadradas e cada célula é ocupada pelo mesmo autômato, digamos J. No início, muitos desses autômatos estão em um estado especial dormente, enquanto os demais estão em algum estado indeterminado. O estado de cada autômato no instante de tempo $t+1$ depende estritamente do seu estado no tempo t, bem como do estado de seus quatro vizinhos também no tempo t. Por emular uma máquina de Turing completa, a máquina de Neumann pode computar qualquer função computável, mas ela também pode duplicar um padrão de estados. Na verdade ela pode gerar um padrão de estados representando qualquer Máquina de Turing específica em qualquer lugar do grid. Por óbvio, após criada, esta máquina pode ser disparada para fazer a computação para a qual foi desenhada.

Este dispositivo conceitual é conhecido como computador construtor universal de Von Neumann (ou UCC).

Em meados dos anos 60, E.F.Codd melhorou o design de Neumann em muitos aspectos, sendo o principal a redução de 29 para 8 estados. Seria necessário um livro de bom tamanho para explicar a máquina de Codd em detalhes, logo apenas uma descrição da mesma segue. Entretanto, para prover algum nível desse detalhe, um aspecto específico dessa máquina (conhecido como construtor de caminhos) será estudado.

Um quadro geral da UCC de Codd, dentro de uma célula do grid, devidamente adormecida, há uma caixa preta, além de um número de fitas. Dentro da caixa preta, há um enorme diagrama de estados, que estão em contínua mudança durante a operação da máquina (qualquer semelhança com a vasta circuitaria de um moderno computador, não é coincidência).

A descrição de uma máquina de turing arbitrária (programa) é colocada na fita da programação, e os dados nos quais essa máquina de turing deve operar são colocados na fita de dados. A caixa preta (aqui chamada de controle e execução da UCC) lê a fita da programação e simula a máquina de Turing definida, lendo e escrevendo na fita de dados. Ao fazer isso um caminho deve ser marcado pela UCC nas duas fitas em questão. Os caminhos são extendidos ou retraídos à medida que a simulação da máquina de Turing ocorre. Padrões de estados representando símbolos podem ser lidos ou escritos ao atravessar esse caminho.

A UCC é também uma construtora universal. Dada a descrição de uma particular máquina de Turing celular na fita da programação, a ucc pode construir essa máquina pela extensão do caminho de construção numa área adjacente dormente à ucc de modo que, depois de pronta, quando ativada ela compute a função própria daquela MT. Na verdade, a UCC pode construir exatamente uma cópia de si mesma dessa maneira.

A máquina a ser construída pela UCC é descrita numa linguagem especial que informa como a construção deve ser feita em termos do deslocamento do caminho de construção. Especificamente para se assegurar que certas células objetivo serão colocadas em seu estado inicial, o caminho é extendido até aquela célula e um sinal especial é propagado pelo caminho até afetar a transição apropriada na célula objetivo. Falando em transições, ainda que cada célula no espaço tenha um autômato de 8 estados, o alfabeto é muito maior. De fato, como cada um dos quatro vizinhos pode estar em um estado (além do estado da célula em questão tem-se

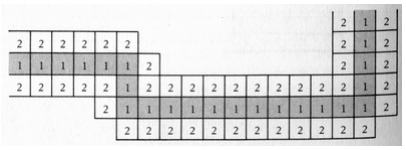
$$8^5 = 32.768$$

possíveis combinações de sinais aos quais cada autômato pode responder entrando em um determinado estado. Codd não chegou a especificar todas essas trasições. Definiu só um pequeno conjunto.

Como caminhos são criados e usados

Os estados de uma célula serão conhecidos como $0, 1, 2, 3, 4, 5, 6$ e 7. O estado quiescente (dormente) é o 0. Se uma célula está no estado 0 e seus 4 vizinhos também, ela permanece no estado 0.

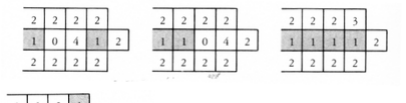
Um caminho consiste de uma configuração de 1 rodeado de 2 (veja a figura abaixo)



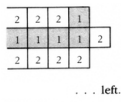
Para extender o caminho ao longo da direção corrente, o padrão 0607 é propagado ao longo do caminho. Quando o final do caminho é encontrado uma nova célula 1 é adicionada ao caminho, bem como 3 novas células 2 são adicionadas ao isolamento. Veja na figura 2 na transparência (não cabe aqui). Note que cada um dos tempos envolve transições em poucos autômatos.

Por exemplo, a célula marcada em negrito no segundo desenho está no estado 1 e rodeada por células nos estados 7,2,2,2 tomados na ordem convencional. Sob a as regras de transição que governam a máquina UCC de Codd, então ocorrerá uma transição no próximo instante de tempo $t+1$. Quando o estado é 1 e os seus vizinhos são 7,2,2,2 então a célula deve entrar no estado 7.

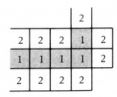
Além disso, precisa-se ordens para virar à esquerda ou à direita. Por exemplo, virar à esquerda é obtido propagando o padrão 04 ao longo do caminho. Veja as figuras



e



. . . left.



The turn completed

A seguir, um 05 é transmitido seguido de um 06 resultando finalmente na configuração mostrada na figura assinalada com "the turn completed", que pode ser novamente extendida através do comando 0607 como se viu acima.

Existem sinais que permitem extender a construção, e existem sinais que determinam a retração do caminho. Ainda que apenas partes dessa máquina tenham sido simuladas em computador para verificar a correção de seus componentes, ela foi especificada em suficiente detalhe para alguém que se disponha a gastar algumas centenas de anos construindo um dispositivo físico de tamanho suficiente para contê-la. Seja como for é provavelmente o maior e mais complicado dispositivo computacional já concebido.

Se não se exigir que um computador auto-replicante seja um construtor universal, o autômato requerido fica muito mais simples. Por exemplo, em 1985, Christopher Langton descobriu um autômato auto replicante que requer apenas 8 estados por célula. Ele publicou uma tabela listando 219 transições que governam o crescimento de uma única célula em direção a uma colônia. Em 1989, John Byl, matemático canadense, achou uma máquina auto-replicante mais simples: suas células tinham apenas 6 estados, sua tabela de transições 56 entradas e a semente inicial era composta de 11 células .

	2	2	
2	3	2	
2	3	4	2
	2	5	

A tabela de transições de Byl aparece a seguir; O formato é $CNESO \rightarrow C$ (central, norte, este, sul, oeste $\rightarrow$ novo central)

CNESW → C	10000 → 0	20000 → 0	30003 → 0	31235 → 5	40252 → 0
00003 → 1	10001 → 0	20015 → 5	30011 → 0	31432 → 1	40325 → 5
00012 → 2	10003 → 3	20022 → 0	30012 → 1	31452 → 5	4xxxx → 3
00013 → 1	10004 → 0	20202 → 0	30121 → 1	3xxxx → 3	
00015 → 2	10033 → 0	20215 → 5	30123 → 1		50022 → 5
00025 → 5	10043 → 1	20235 → 3	31122 → 1	40003 → 5	50032 → 5
00031 → 5	10321 → 3	20252 → 5	31123 → 1	40043 → 4	50212 → 4
00032 → 3	11253 → 1	2xxxx → 2	31215 → 1	40212 → 4	50222 → 0
00042 → 2	12453 → 3		31223 → 1	40232 → 4	50322 → 0
0xxxx → 0	1xxxx → 4	30001 → 0	31233 → 1	40242 → 4	5xxxx → 2

☞ Para você fazer

Você deve calcular o estado da grade no instante $t+1$, considerando que a grade dada está no estado t. Use as regras de transição do autômato de John Byl, dadas acima. Após calcular o novo estado da grade, SOME os números encontrados e responda esse número. Para o segundo exercício, recomendo fortemente a construção de um programa que faça a contagem, pois fazê-la à mão é pedir para errar...

Para este exercício considere que a matriz pedida está rodeada por zeros adicionais que não são mostrados (esta generalização é necessária para calcular os estados das células limítrofes da grade).

```
1. Considere a seguinte grade 4 x 4

  3  0  0  0
  3  2  0  2
  5  4  0  4
  0  0  0  0

2. E a seguinte grade 15 x 15

  1 5 5 0 1 4 0 0 2 0 0 0 0 3 0
  4 0 5 5 0 3 1 0 3 4 1 0 2 4 0
  0 2 0 0 0 5 0 3 0 0 0 0 3 3 0
  4 0 4 2 0 0 0 0 0 0 0 3 0 0 0
  0 3 4 2 5 0 3 0 0 0 0 0 0 0 0
  3 0 0 3 5 0 5 4 5 0 0 0 0 0 2
  1 4 0 0 5 0 0 0 0 0 4 1 0 0 0
  0 1 5 4 0 3 2 0 5 0 0 0 1 1 4
  0 1 0 4 5 3 0 0 3 0 0 5 0 0 0
  5 5 2 0 0 2 0 0 0 3 0 0 0 0 4
  5 2 2 1 3 0 1 1 0 0 5 0 3 0 0
  3 4 0 0 2 0 5 0 3 0 0 0 1 0 2
  0 0 0 0 0 0 0 2 0 0 0 3 5 0 0
  5 0 4 0 3 0 0 3 2 0 0 0 0 4 0
  2 0 3 3 0 3 0 0 5 0 0 3 5 0 4
```

1	2

Para saber mais:
CODD, E. F. **Cellular Automata**. New York: Academic Press, 1968.
NEUMANN, J von. **The Theory of Self-Reproduction Automata**. Urbana: University of Illinois Press. 1966.
DEWDNEY, A. K. **The (new) Turing Omnibus**. New York: A.W.H. Freeman, 1989.