Não tão básicos assim

Autômatos Celulares

Ideia Geral

- Grid de células (vamos trabalhar só com a retangular)
- Estados: cada célula tem dois (viva / morta) ou mais estados
- Vizinhança: células próximas
- Regras: numa dada geração, estado depende do estado das células vizinhas na geração anterior

Possibilidades de exploração

- Mexer com o número de estados
- Mudar as configurações da vizinhança

E mais um monte de coisa que não vamos ver hoje:

- Trabalhar com grid não retangular (hexagonal, por exemplo)
- 3D
- Regras probabilísticas
- Memória maior: as n gerações anteriores afetam a geração seguinte
- E por aí vai =)

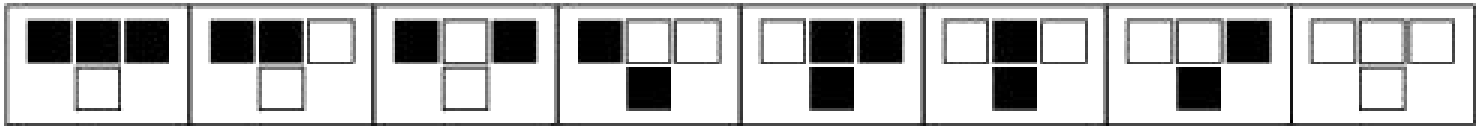
Referências

- Baixe os códigos que vou mostrar durante a apresentação: tinyurl.com/processing-lhc

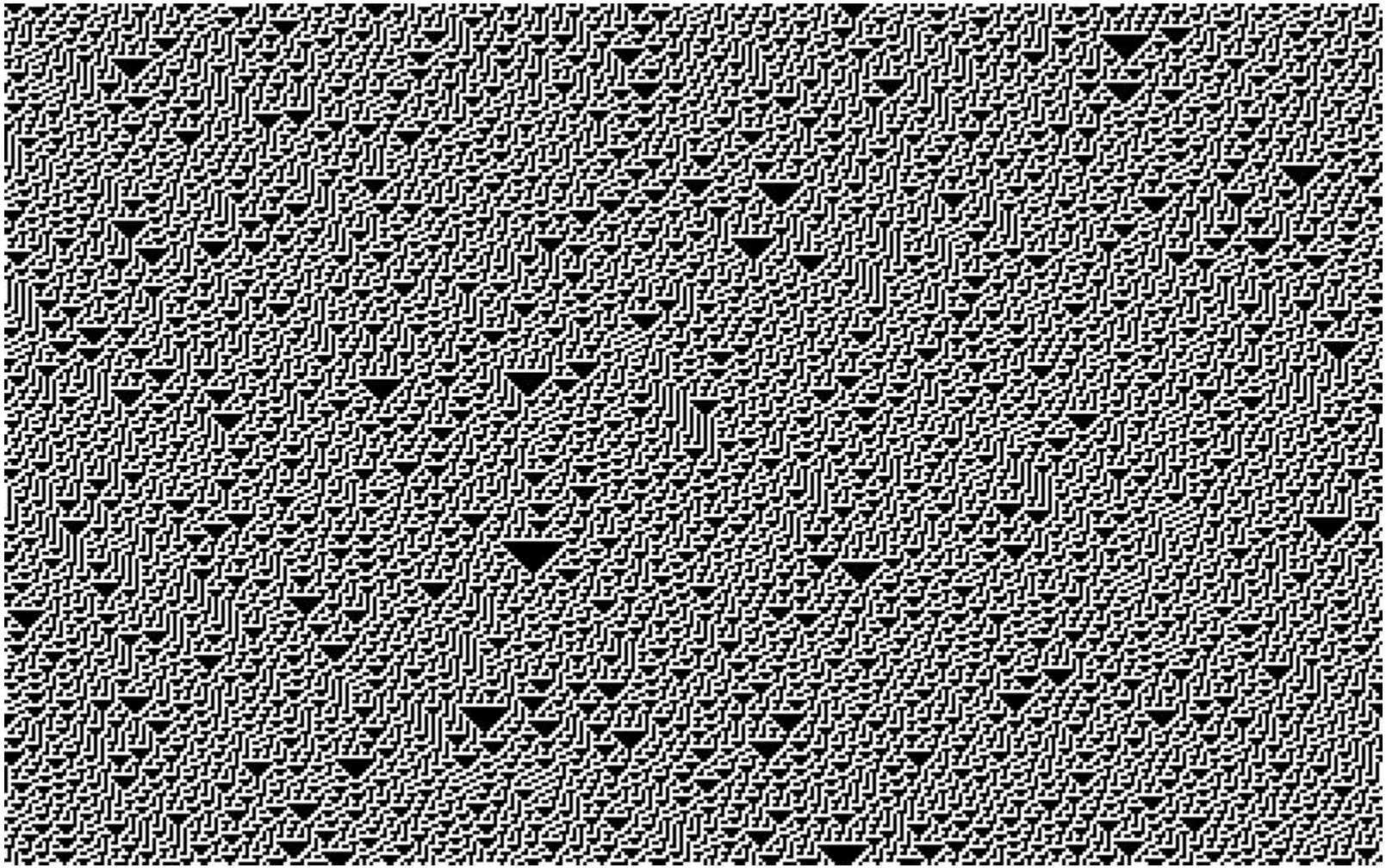
- Kellie Evans, “Larger than Life: Digital Creatures in a Family of Two-Dimensional Cellular Automata”
- Softology: softologyblog.wordpress.com
- Slackermanz: reddit.com/user/slackermanz

Regras de Wolfram

- Unidimensional
- 2 estados
- Vizinhança: 3 células
- Exemplo: regra 30



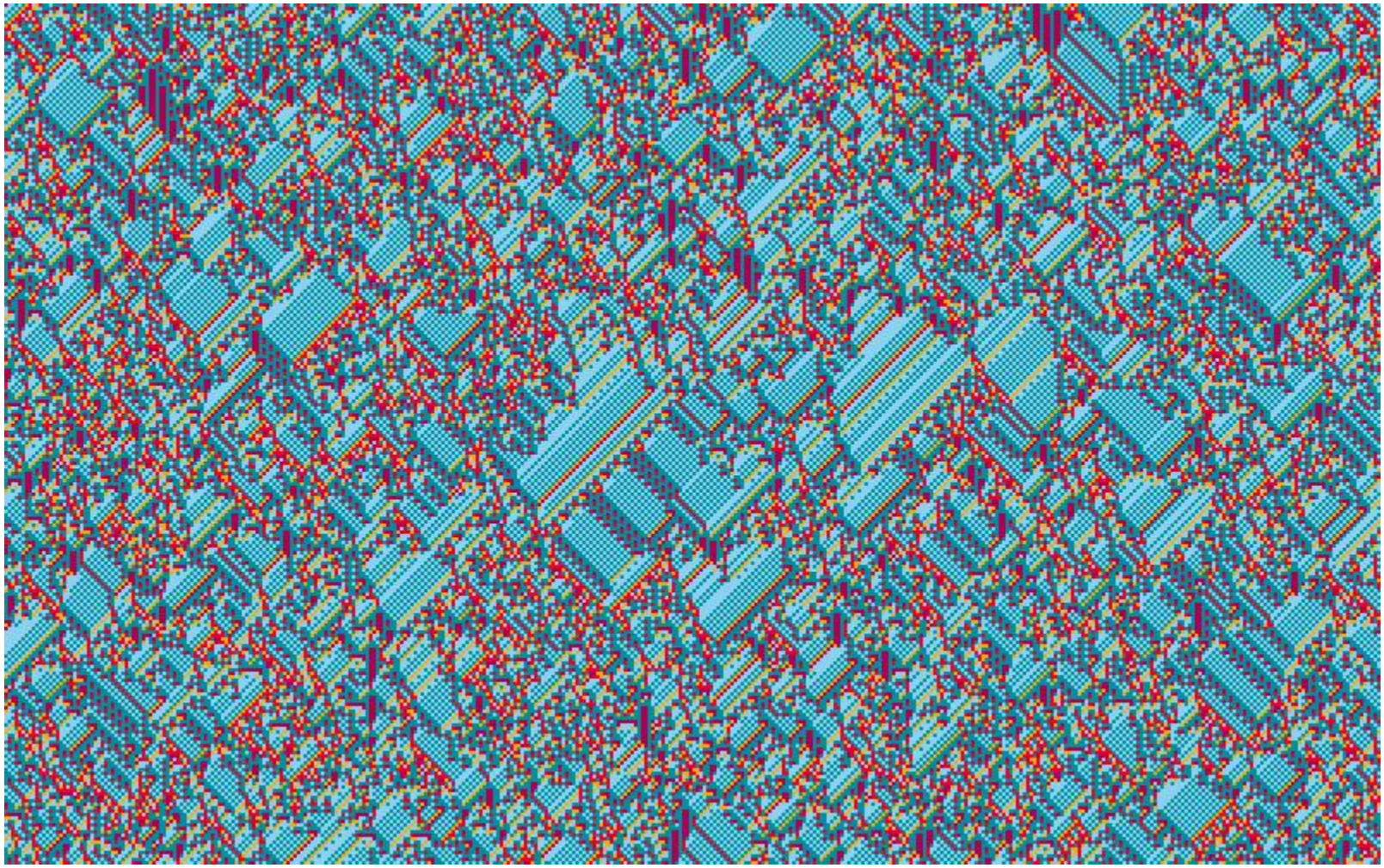
Regras de Wolfram: regra 30



Além das Regras de Wolfram

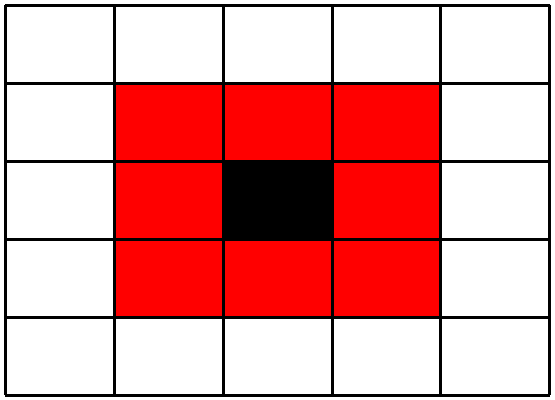
- Unidimensional
- **Em vez de 2 estados, vamos fazer 5!**
- Vizinhança: 3 células
- Agora, temos 5^3 configurações possíveis para a vizinhança
  - Ideia: gerar as regras aleatoriamente (ou nem tanto) para facilitar a exploração
- Não podemos mais acessar a regra via condicional
  - Solução: configuração da vizinhança como um número na base 5

Além das Regras de Wolfram



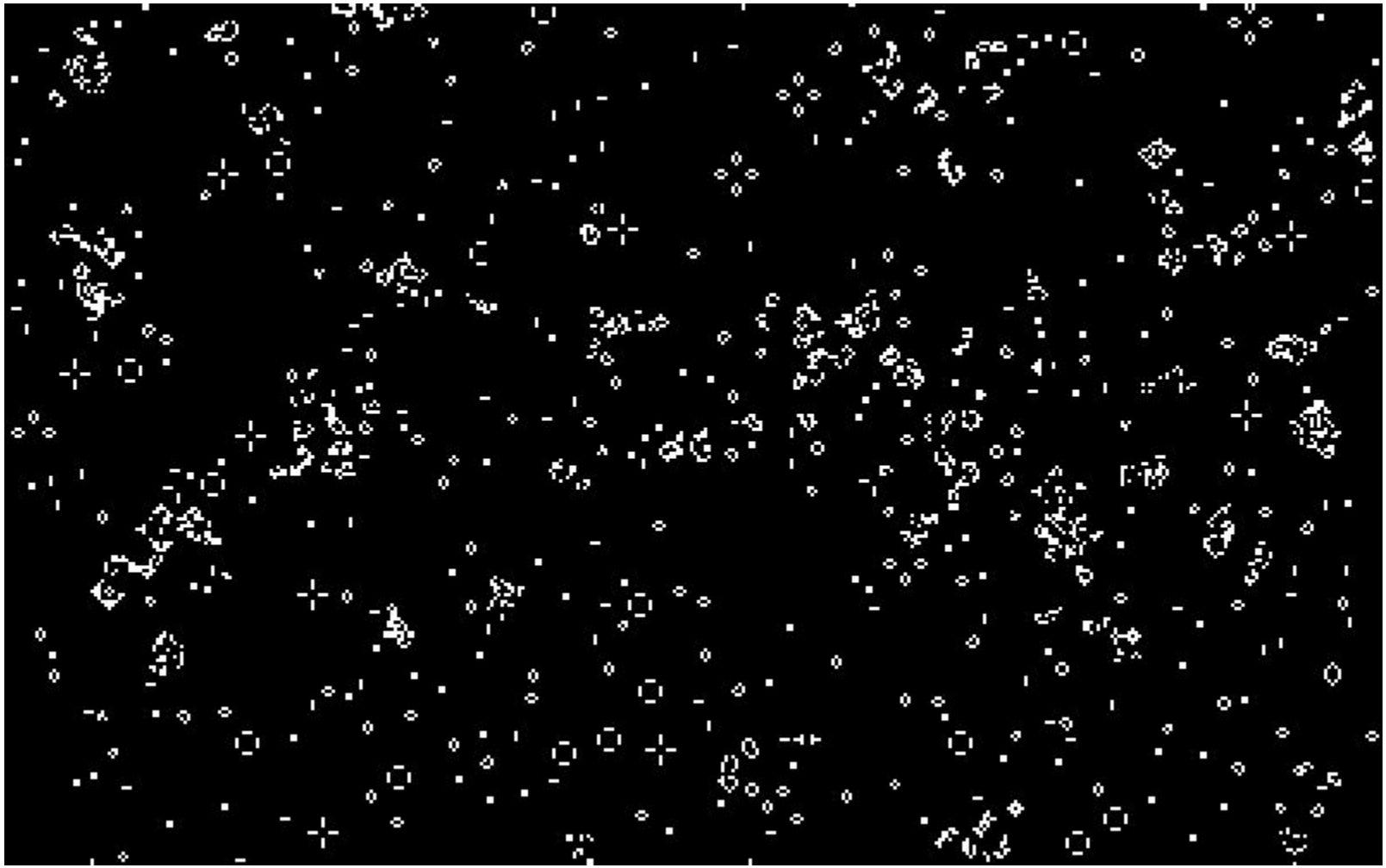
Jogo da Vida de Conway

- **Duas dimensões**
- 2 estados
- **Vizinhança: 8 células**



- Regras:
  - Se a célula está viva e tem 2 ou 3 vizinhos, ela permanece viva. Caso contrário, ela morre
  - Se a célula está morta e tem exatamente 3 vizinhos, ela fica viva. Caso contrário, permanece morta

Jogo da Vida

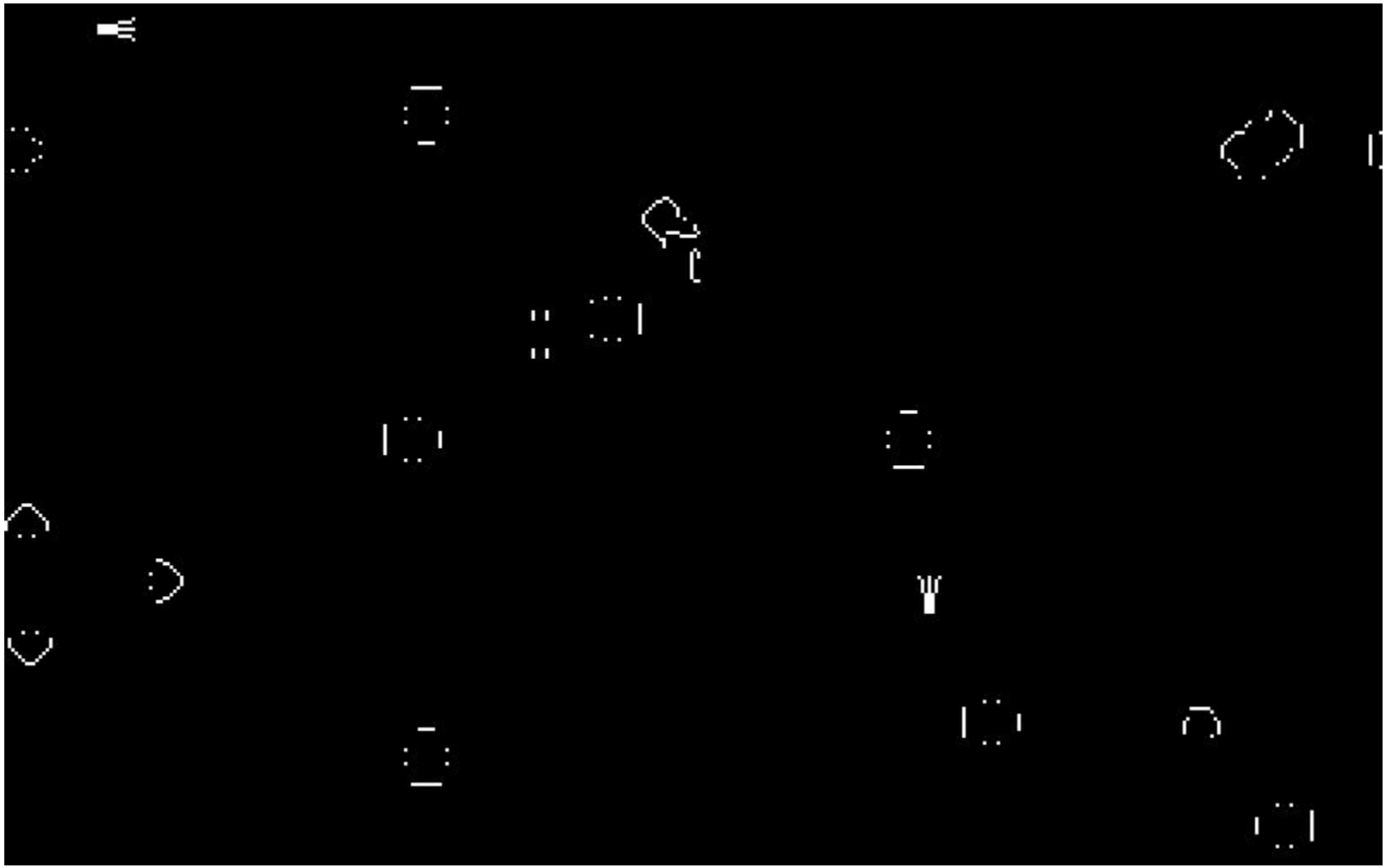


Maior que a Vida

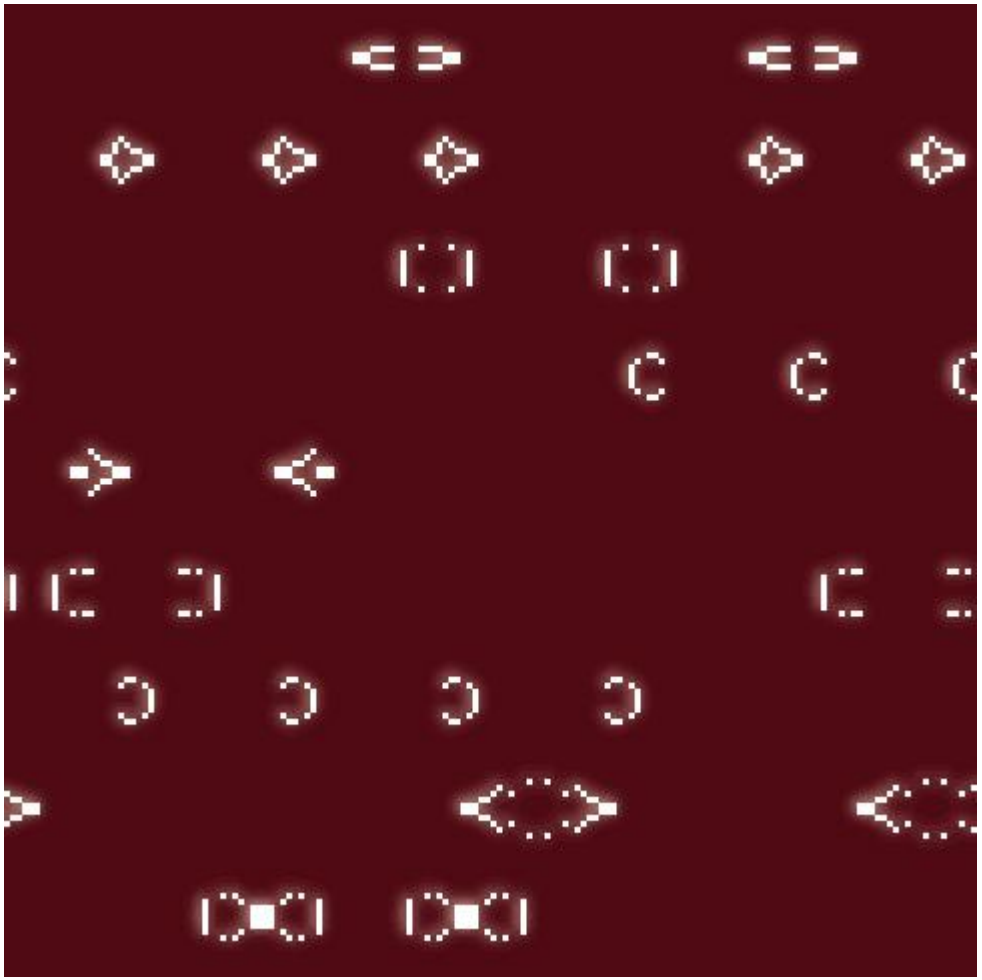
- Duas dimensões
- 2 estados
- **Vizinhança é um quadrado de N x N células**

- Regras:
  - Se a célula está viva e tem um certo número de vizinhos, ela permanece viva. Caso contrário, ela morre
  - Se a célula está morta e tem um certo número de vizinhos, ela fica viva. Caso contrário, permanece morta

Maior que a Vida

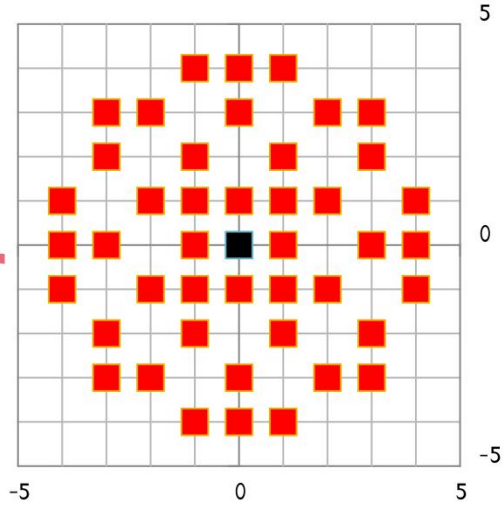


Maior que a Vida



Vizinhanças esquisitas

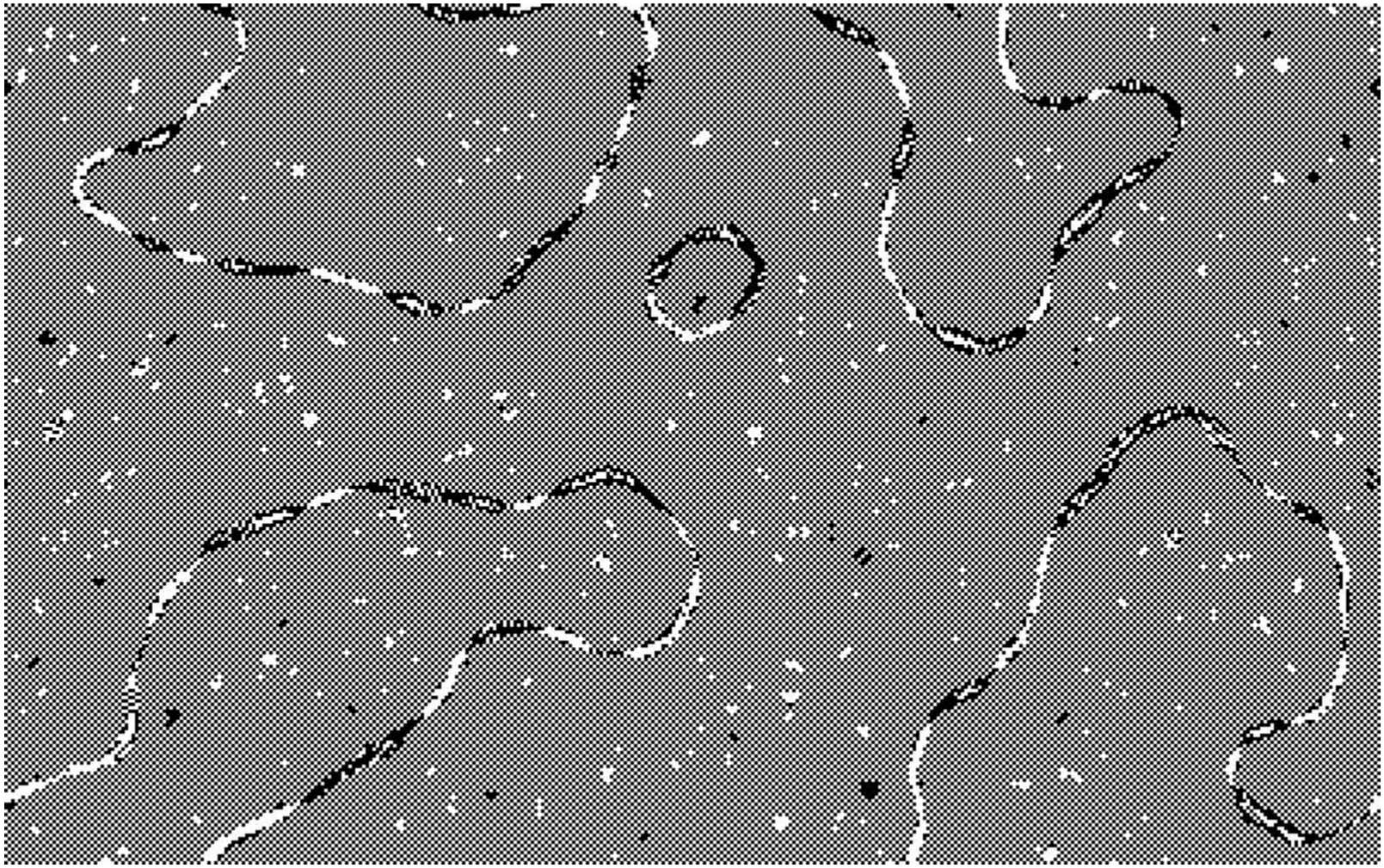
- Duas dimensões
- 2 estados
- **Vizinhança: formato irregular**



Regras

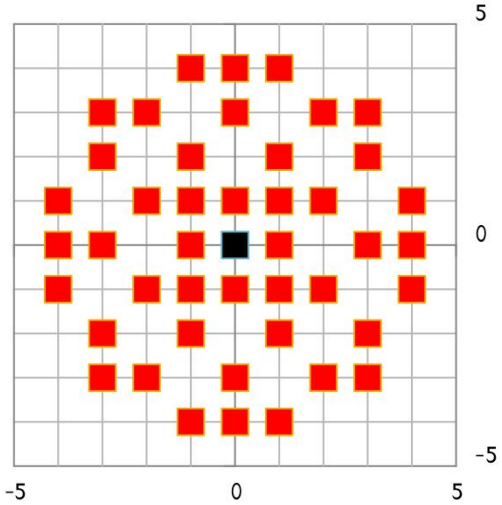
- Se o número de vizinhos estiver dentro do intervalo 1, a célula permanece viva se estava viva, e fica viva se estava morta
- Se o número de vizinhos estiver dentro do intervalo 2, a célula morre se estava viva, e permanece morta se estava morta
- Caso contrário, o seu estado não se altera

Vizinhanças esquisitas



Vizinhanças esquisitas com regras bizarras

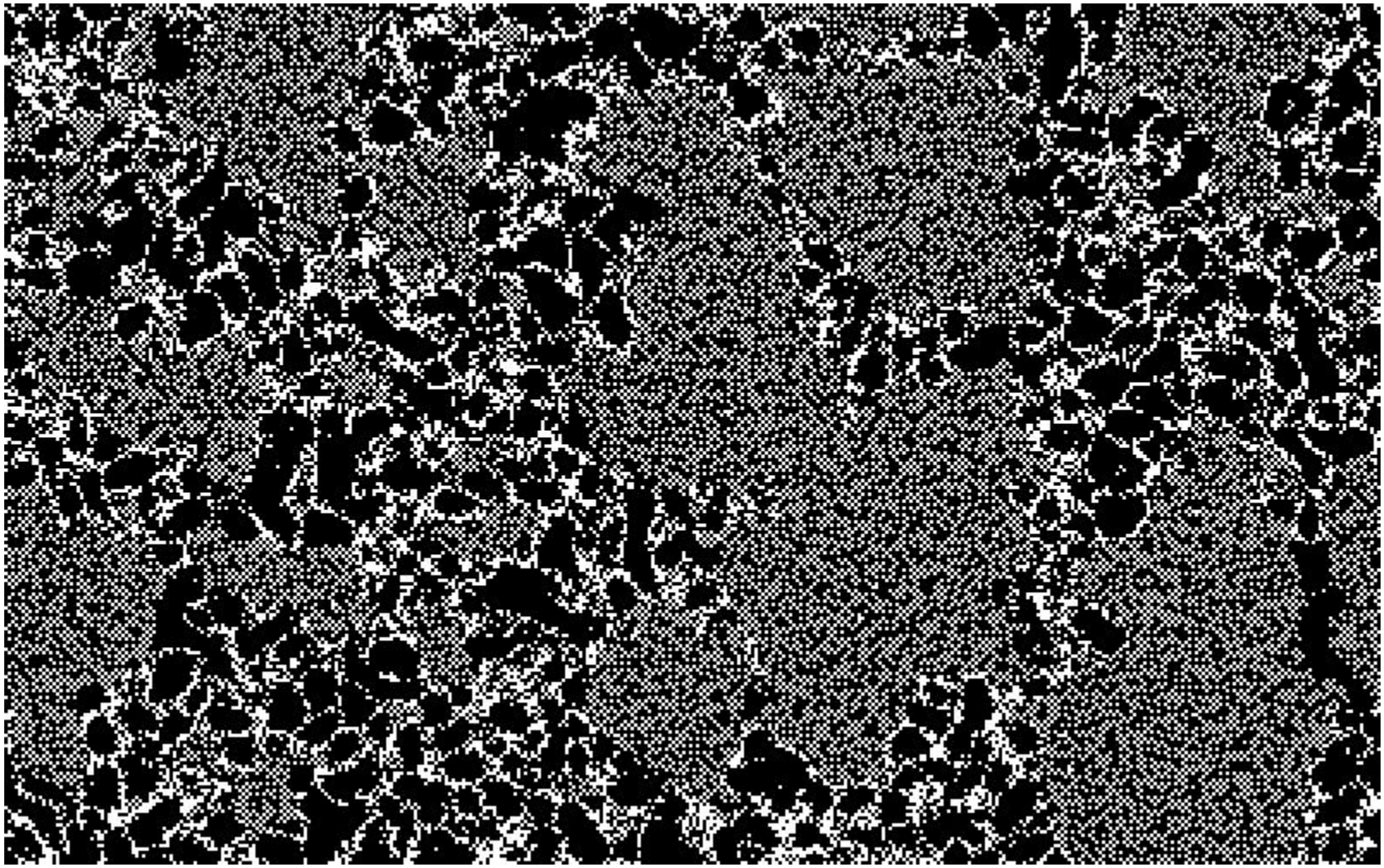
- Duas dimensões
- 2 estados
- Vizinhança: formato irregular



Regras

- N intervalos
- Se o número de vizinhos estiver dentro de certos intervalos, a célula permanece viva se estava viva, e fica viva se estava morta
- Se o número de vizinhos estiver dentro dos demais intervalos, a célula morre se estava viva, e permanece morta se estava morta
- Caso contrário, o seu estado não se altera

Vizinhanças esquisitas com regras bizarras



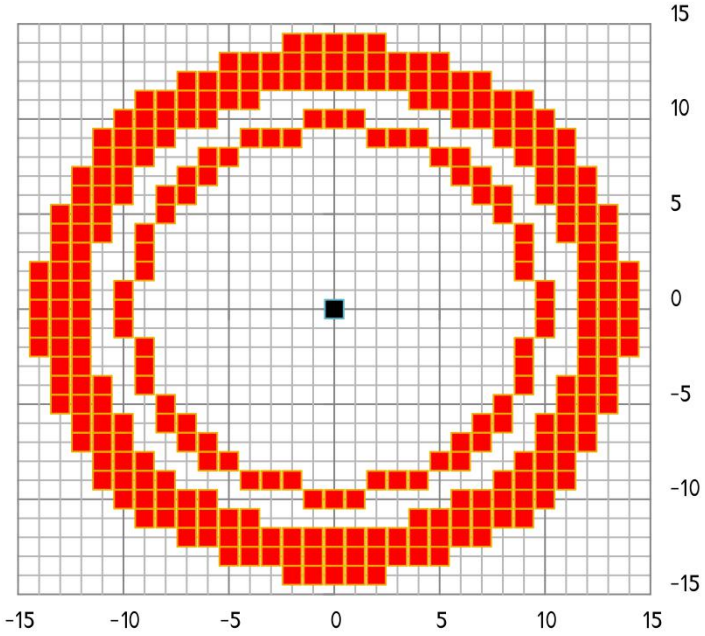
Múltiplas vizinhanças

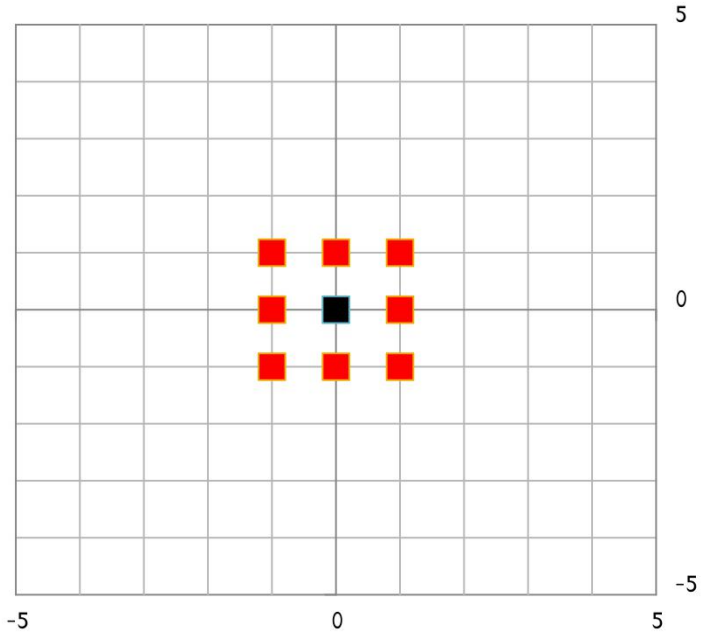
- Duas dimensões
- 2 estados
- **Várias!**

Regras

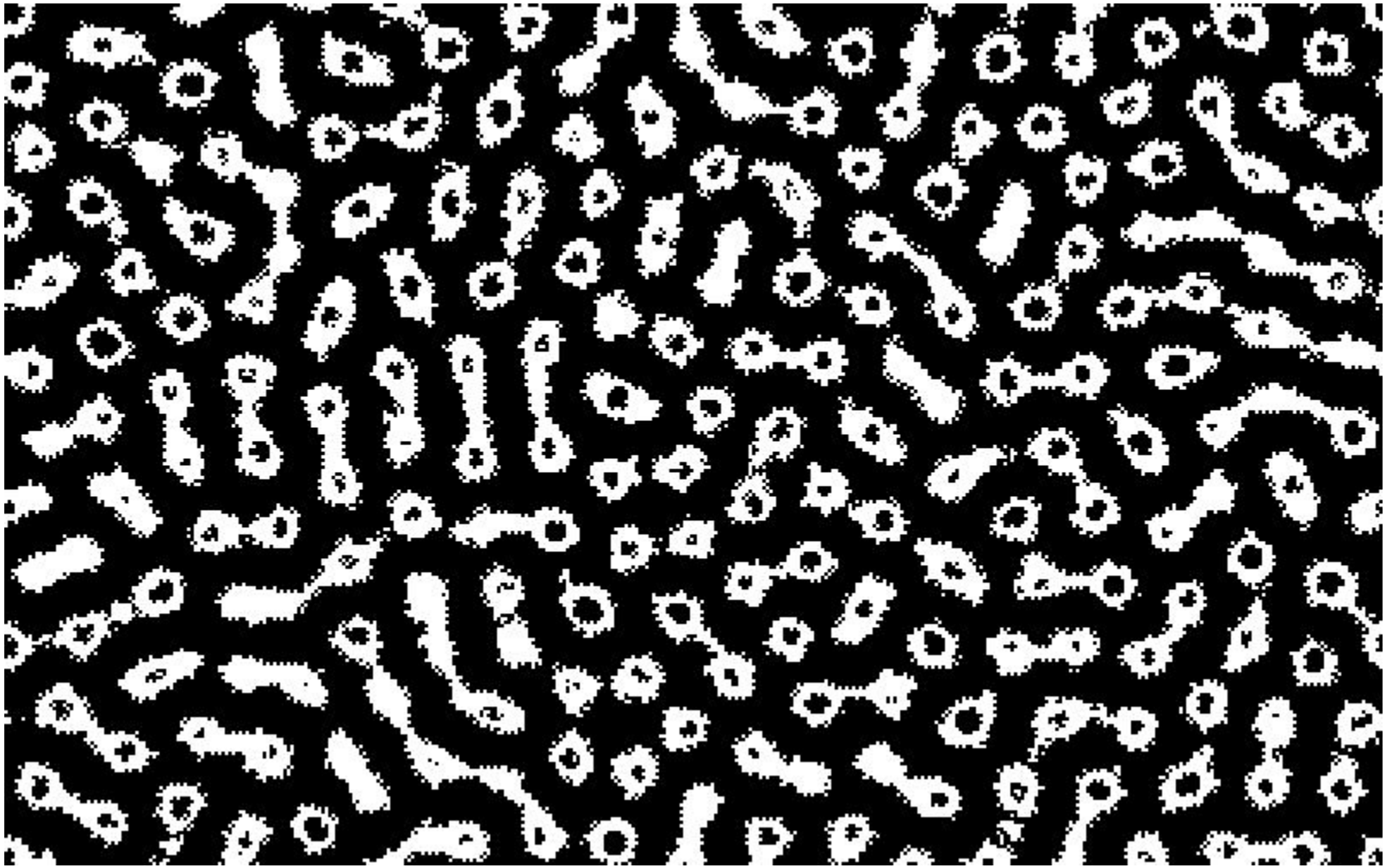
- Aplicadas sequencialmente, vizinhança a vizinhança
- Vários intervalos para cada vizinhança
- Se o número de vizinhos estiver dentro de certos intervalos, a célula permanece viva se estava viva, e fica viva se estava morta
- Se o número de vizinhos estiver dentro dos demais intervalos, a célula morre se estava viva, e permanece morta se estava morta
- Caso contrário, o seu estado não se altera

Múltiplas vizinhanças

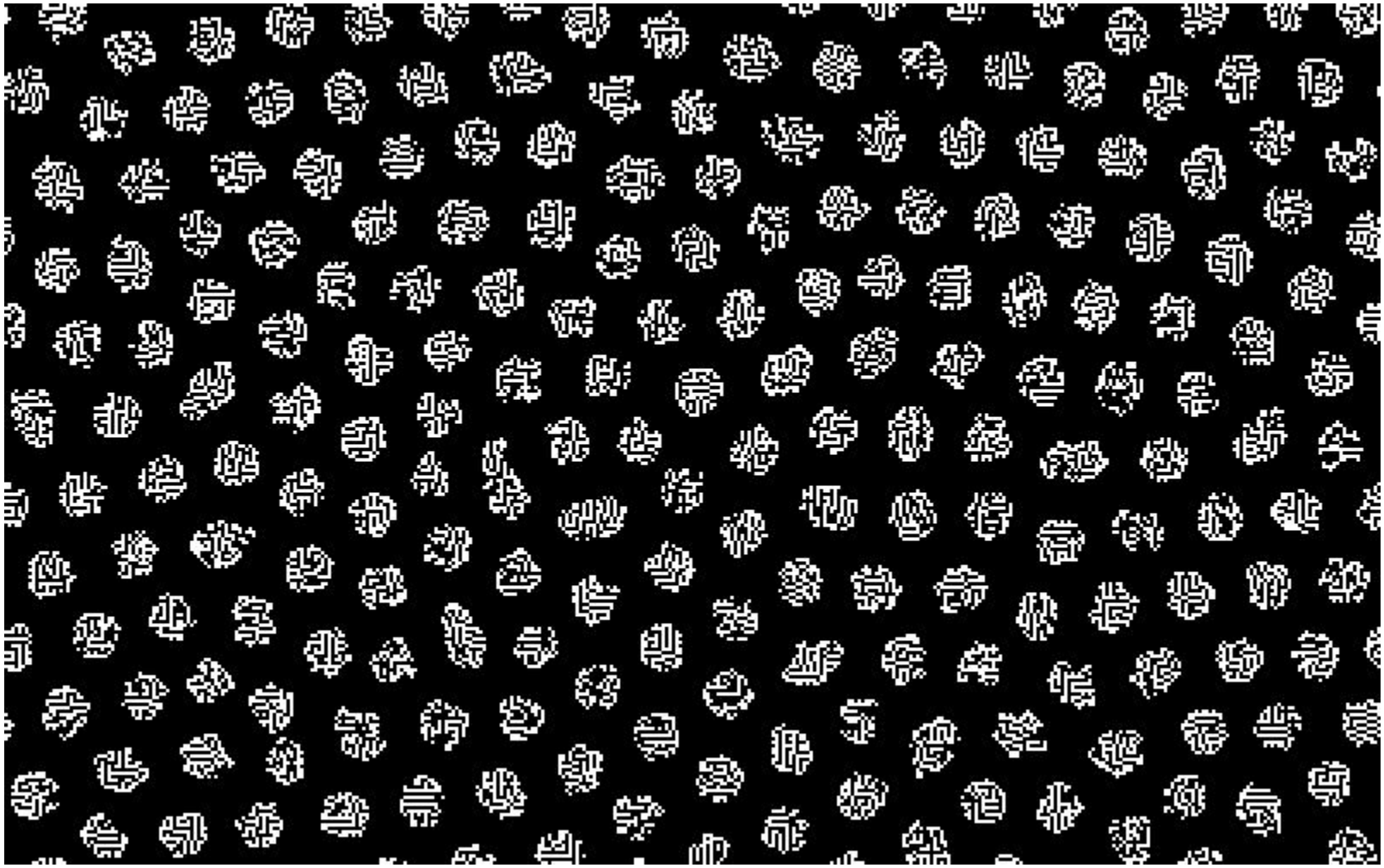




Múltiplas vizinhanças



Múltiplas vizinhanças

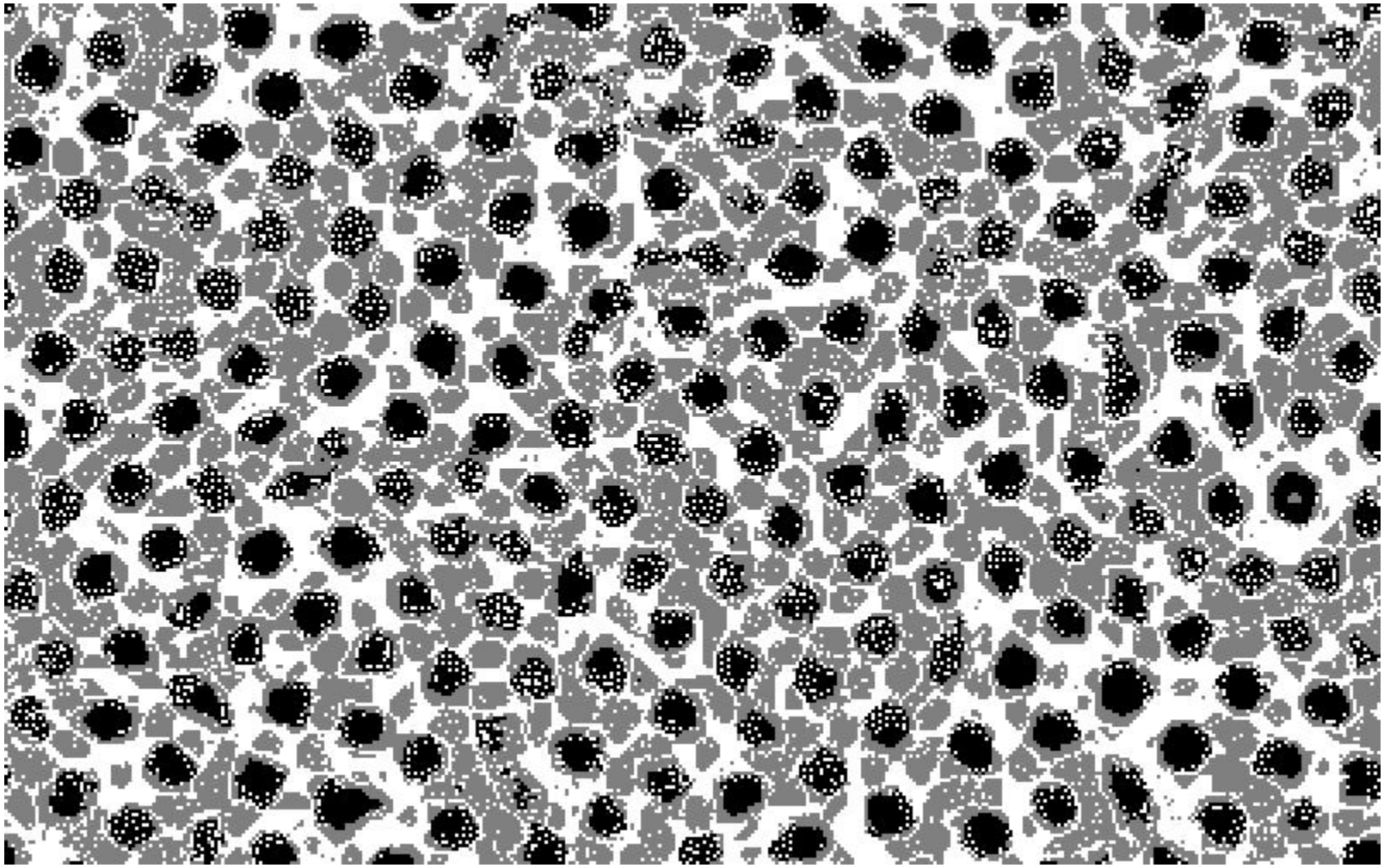


Múltiplas vizinhanças: 3 estados

- Duas dimensões
- **3 estados**
- Várias!

Como as regras são dadas por intervalos, em vez de trabalhar com "0" / "1" / "sem alteração" como resultados possíveis em cada intervalo, podemos usar "0" / "1" / "2" / "sem alteração".

Múltiplas vizinhanças: 3 estados



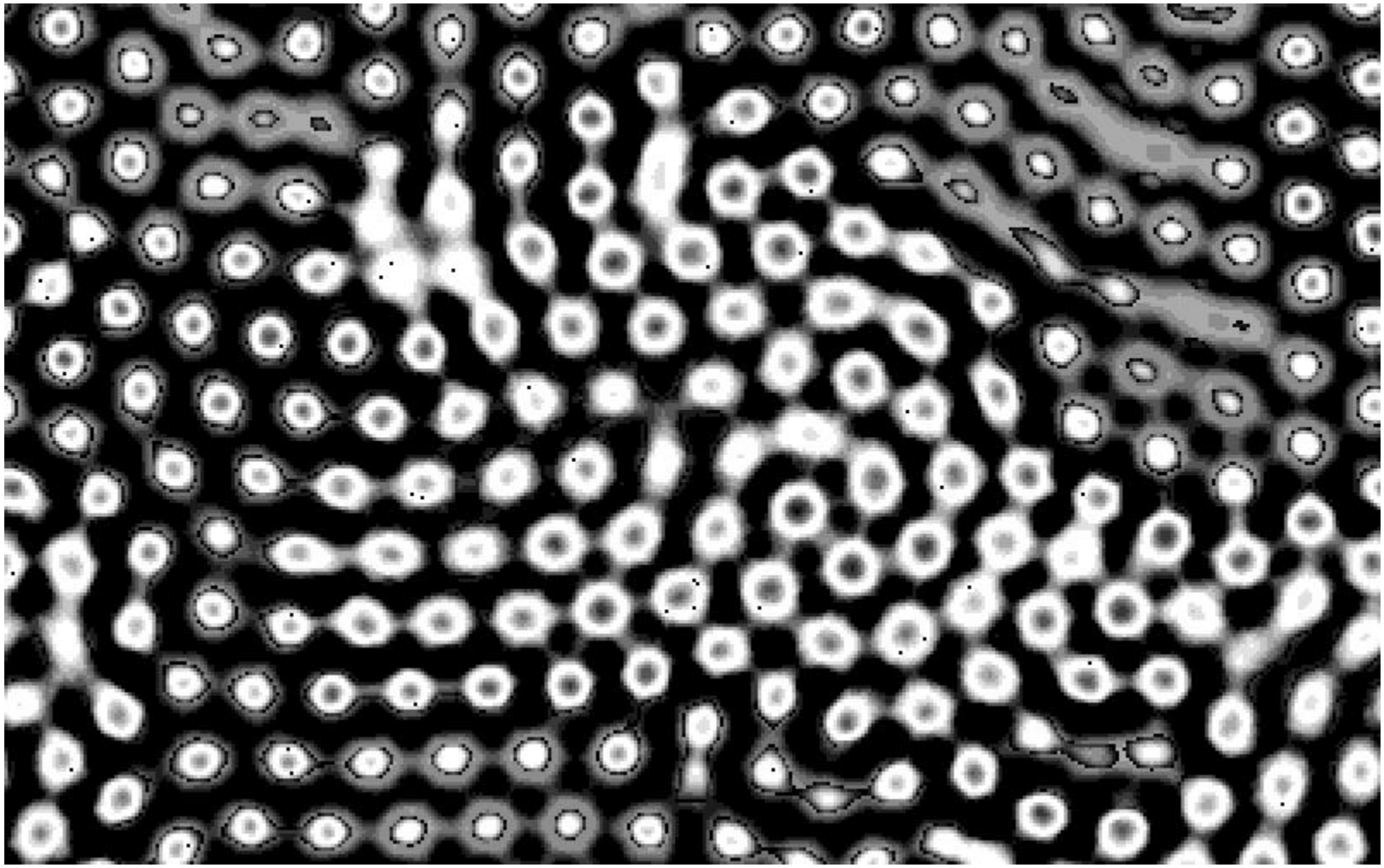
Múltiplas vizinhanças: muitos estados!

- Duas dimensões
- 10 estados
- Várias!

A abordagem anterior funciona para 3-4 estados, mas não para um número grande.
Solução: regra dos senos

- Em vez de fazer intervalos, para cada vizinhança, fazemos $s = \sin(\beta \sum e_i)$
- Se s é menor que zero, o estado não se altera
- Caso contrário, o novo estado é uma função de s

Múltiplas vizinhanças: muitos estados!



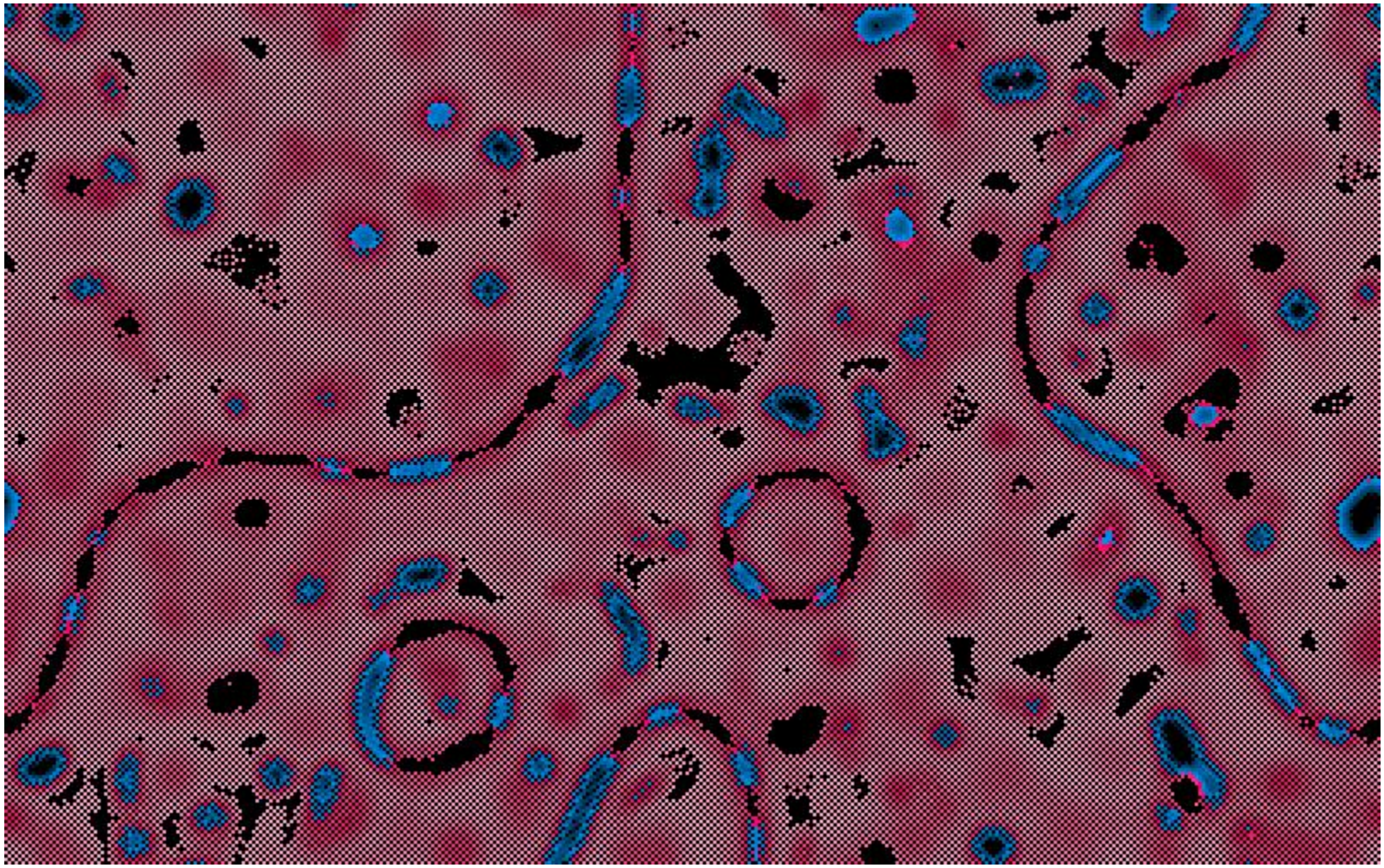
Múltiplas vizinhanças: muitos estados!

- Duas dimensões
- 60 estados
- Várias!

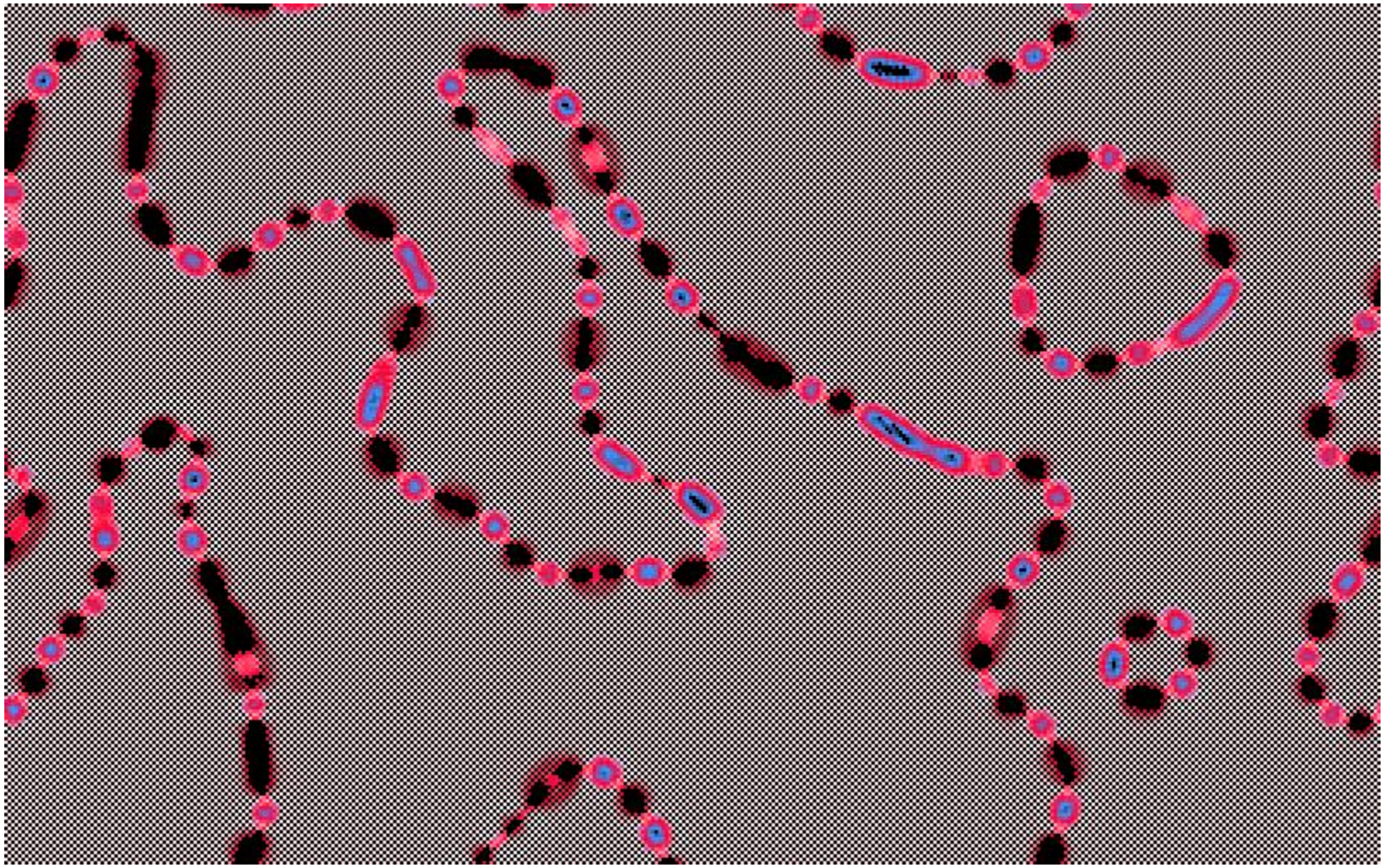
A abordagem anterior funciona para 3-4 estados, mas não para um número grande.
Solução: regra dos senos

- Em vez de fazer intervalos, para cada vizinhança, fazemos $s = \sin(\beta \sum e_i)$
- E, porque não, fazemos bruxaria com s

Múltiplas vizinhanças: muitos estados!



Múltiplas vizinhanças: muitos estados!



Múltiplas vizinhanças: muitos estados!