How To Color a French Flag

Biologically Inspired Algorithms for Scale-Invariant Patterning

Bertie Ancona, Ayesha Bajwa, Nancy Lynch, and
Frederik Mallmann-Trenn

Massachusetts Institute of Technology

The slides are a courtesy of Ayesha, the figures are a courtesy of Bertie.

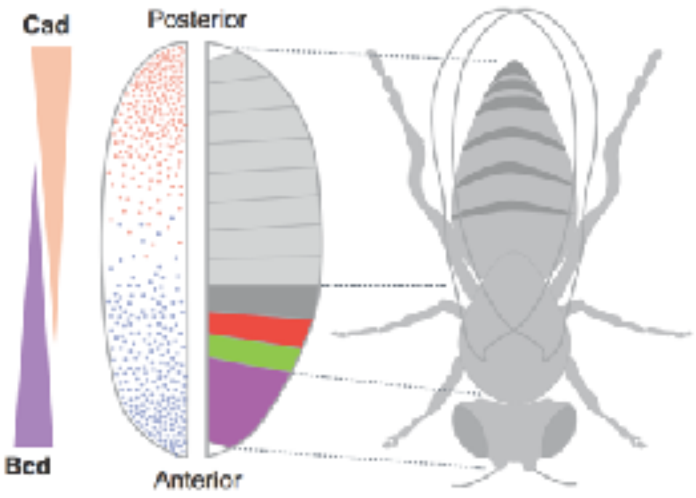
Bertie Ancona

Ayesha Bajwa

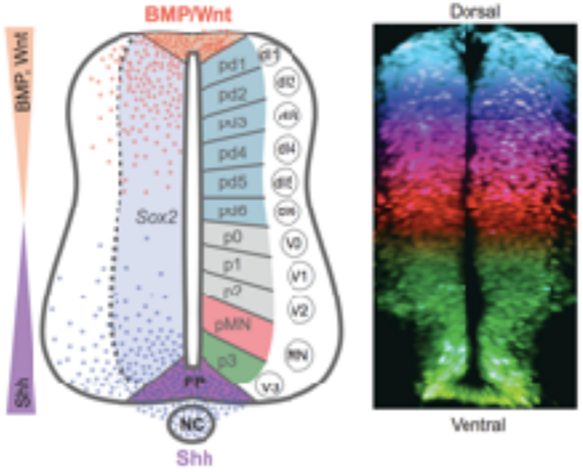
Nancy Lynch

Developmental Patterning

How do tissues form in a scale-invariant way as organisms develop?



Fly Embryo Patterning
(Briscoe & Small 2015)

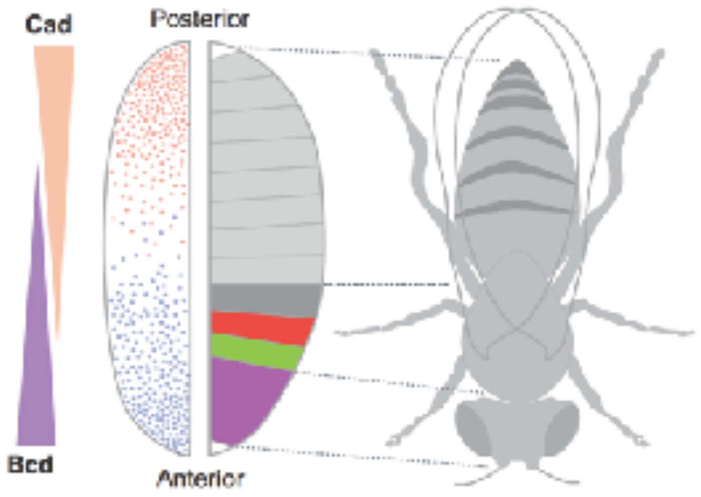


Vertebrate Neural Tube
Patterning
(Briscoe & Small 2015)

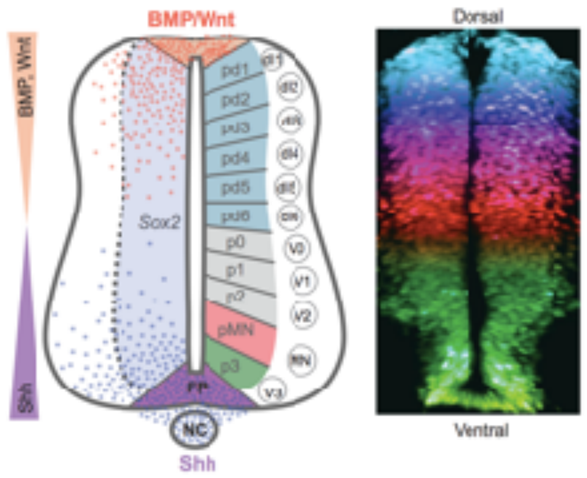
(precursor to the central nervous system)

“The French Flag Problem” (Wolpert 1969)

- Cells determine their type from positional information
- Positional information is thought to come from graded chemical signals known as *morphogen gradients*
- Later verified experimentally (Nüsslein-Volhard & Wieschaus 1980)



Fly Embryo Patterning
(Briscoe & Small 2015)

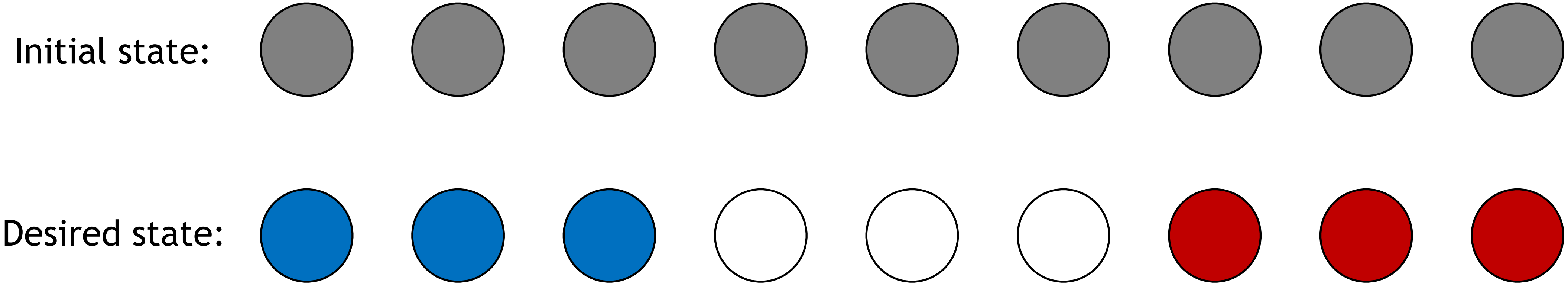


Vertebrate Neural Tube Patterning
(Briscoe & Small 2015)

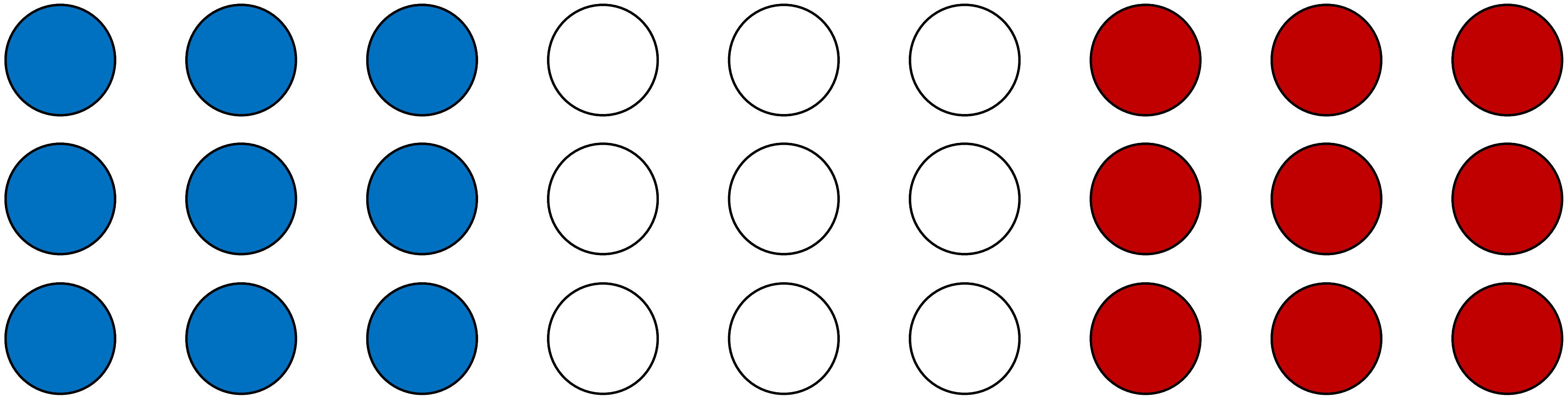
1D: The French Ribbon

Analogy to cellular differentiation: cells are agents which are initially uncolored and must choose a color, from 3 (or *k)* colors.

$$\max\{|b - w|, |b - r|, |w - r|\} \leq 1$$



2D: The French Flag

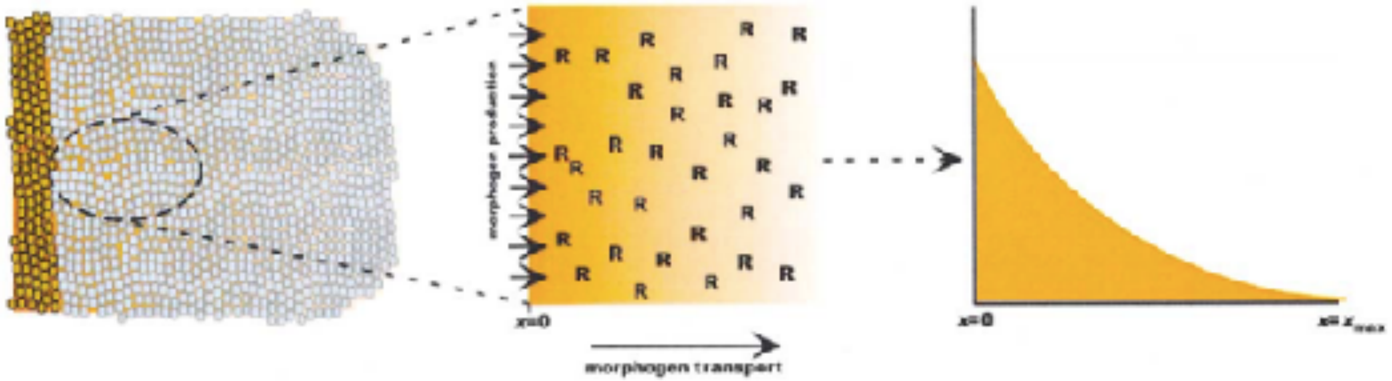


Generalization from 1D case (3 or *k* colors).

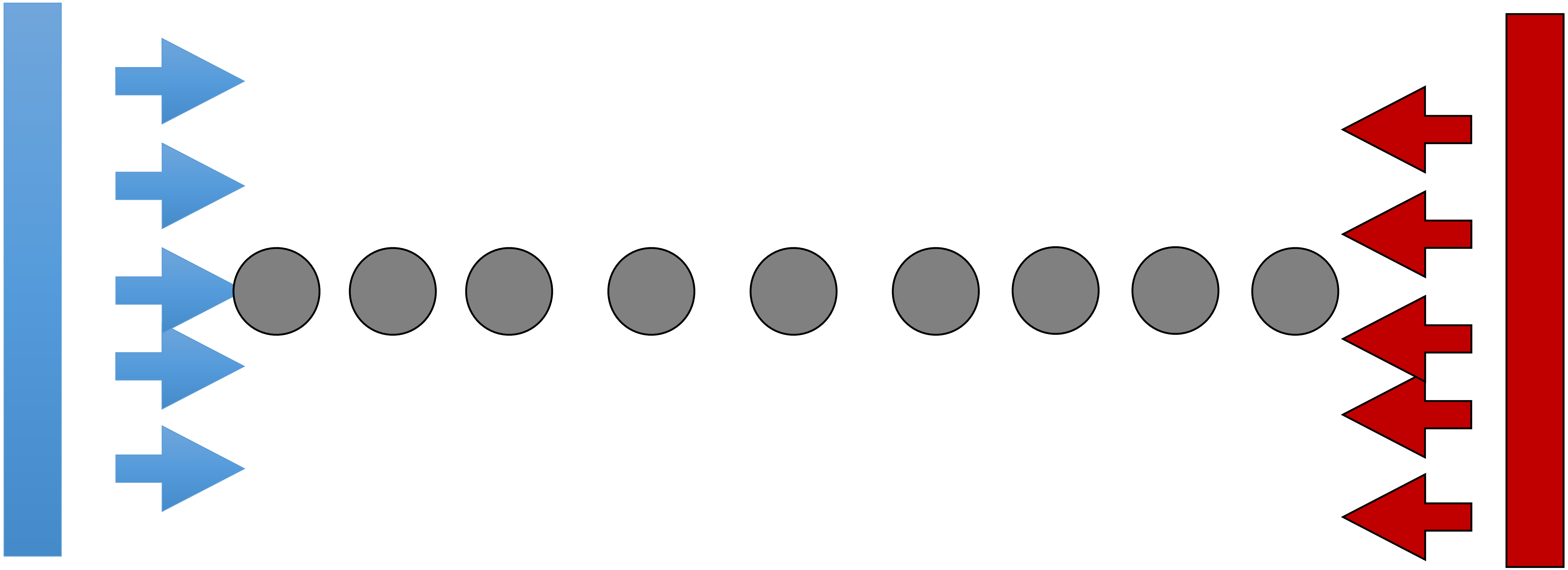
See paper for definition of ε-approximate flag and approximate algorithms (Ancona et al. 2019)

Concentration Model (Synchronous)

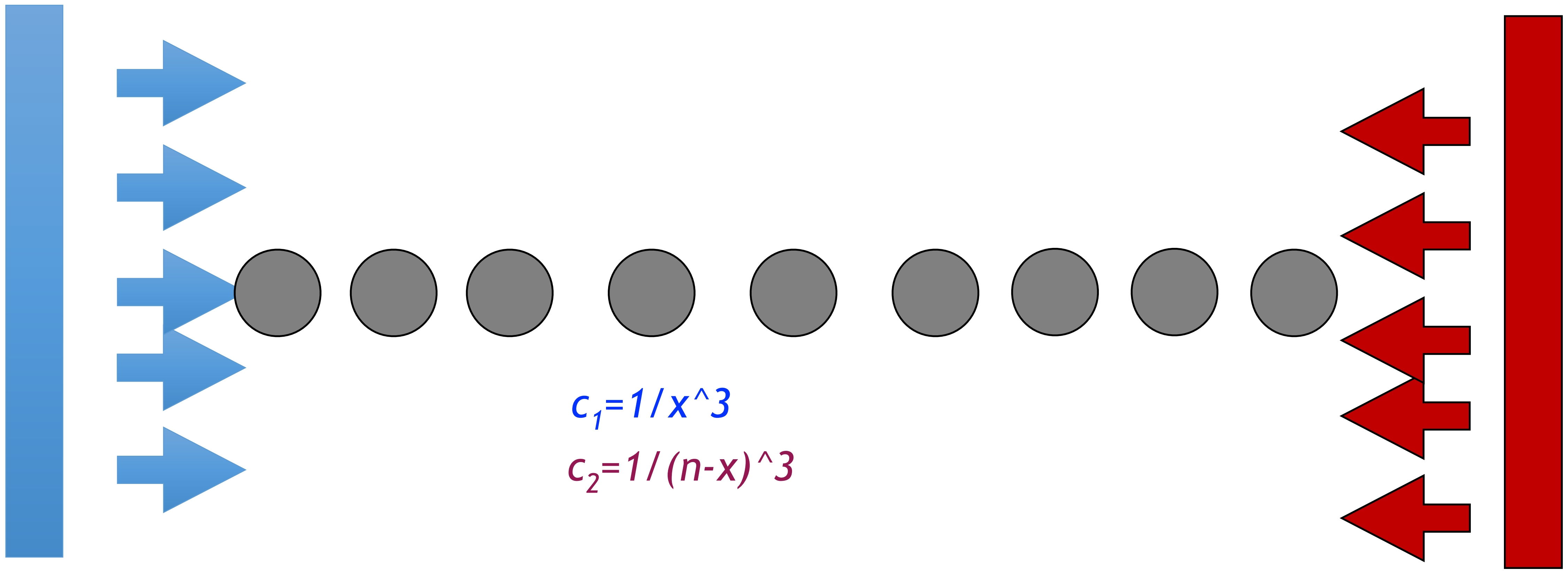
- Provides positional information to agents directly via measured morphogen concentrations
- Morphogens released by designated *source* agents
- Assume steady state concentrations, agents know gradient function
- Compare to biological morphogen gradients



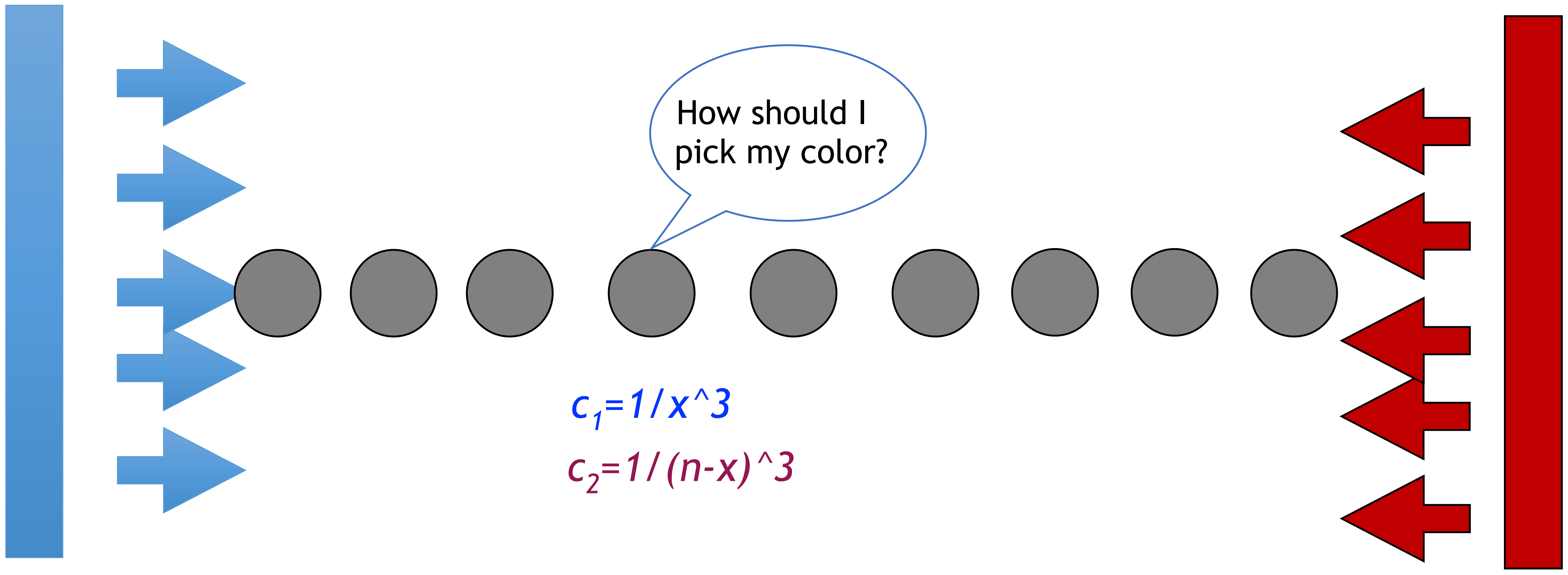
(Lander et al. 2002)



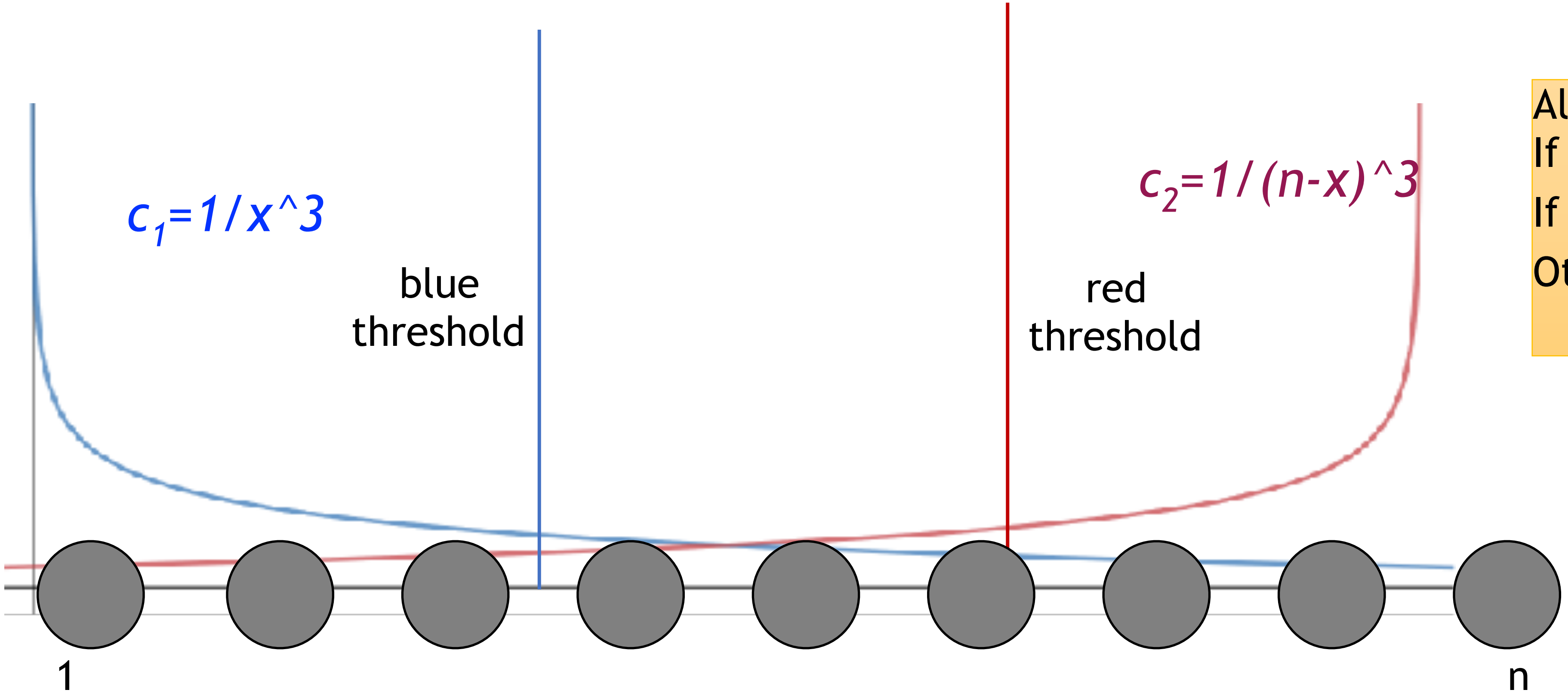
Agents receive steady state input concentrations c_1 and c_2 and know the gradient function (e.g. inverse power law)



Agents receive steady state input concentrations c_1 and c_2 and know the gradient function (e.g. inverse power law)



1D Concentration Algorithm



Algorithm for each agent:
If $c_1 / c_2 >= 8$, choose blue
If $c_1 / c_2 <= 1/8$, choose red
Otherwise choose white

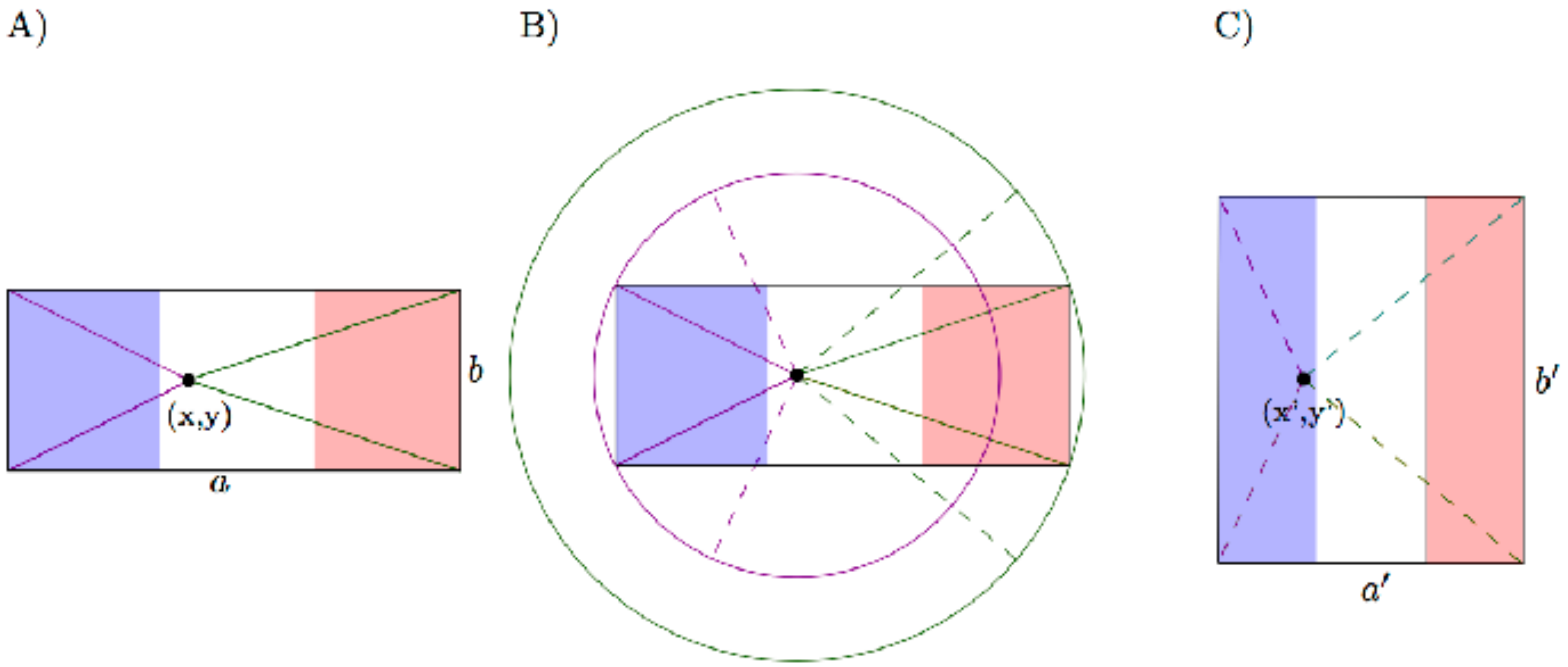
Independent of n!!!

Theorem:

The simple algorithm on the last slide solves the French ribbon problem for any n.

What about the 2D version? The French flag problem?

2D Concentration Impossibility



Given a flag, we construct a new flag with a different aspect ratio so that two agents which are supposed to be different colors can't distinguish settings – can't color correctly.

Message Passing Model (Synchronous)

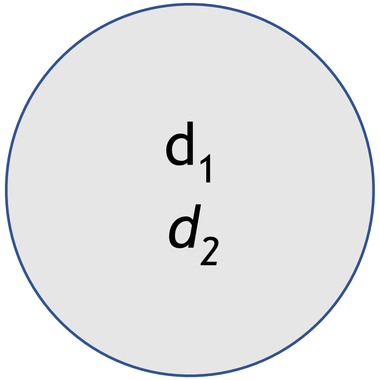
- Similar to LOCAL model from distributed computing
- Difference: agents know directions (up, down, left, right) and which of their neighbors exist
- All agents initially asleep except the *starting agent*, which awakens other agents as algorithms progress
- Comparable to biological cell-cell communication

Exact Count

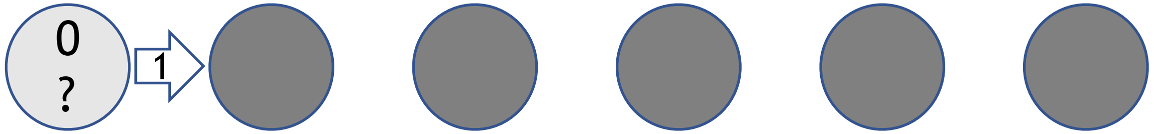
- Agents calculate the distances to the end points d_1 and d_2

Algorithm for each agent:
If d_1 / d_2 >= 2, choose blue
If d_1 / d_2 <= 1/2, choose red
Otherwise choose white

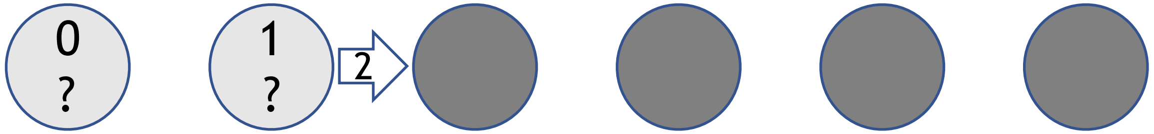
Exact Count:



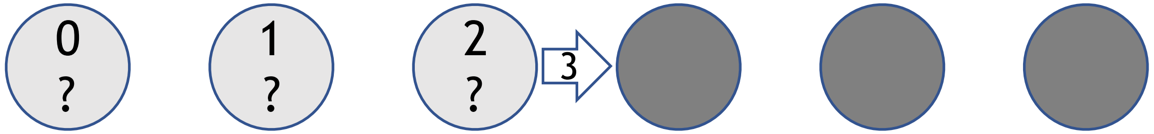
Exact Count: Round 1



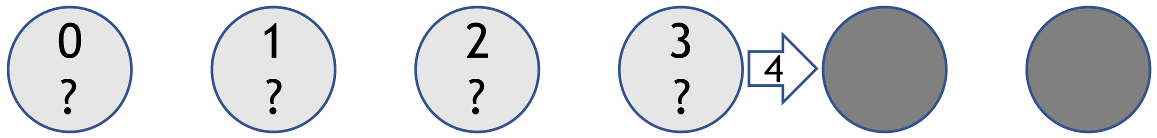
Exact Count: Round 2



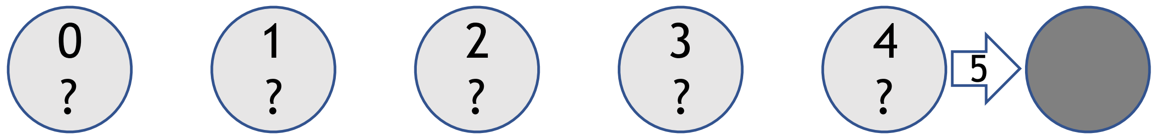
Exact Count: Round 3



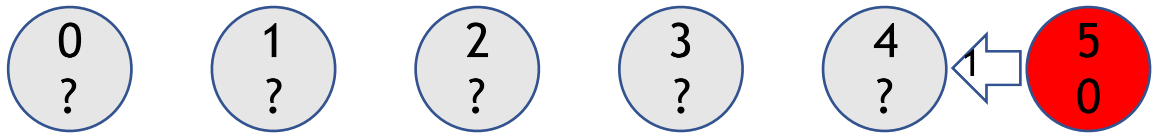
Exact Count: Round 4



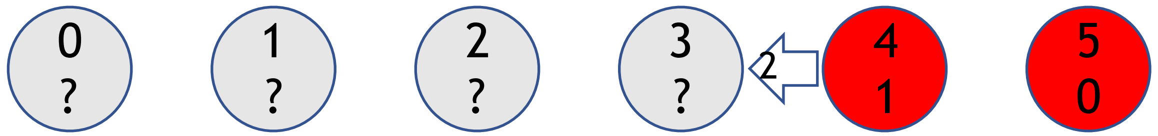
Exact Count: Round 5



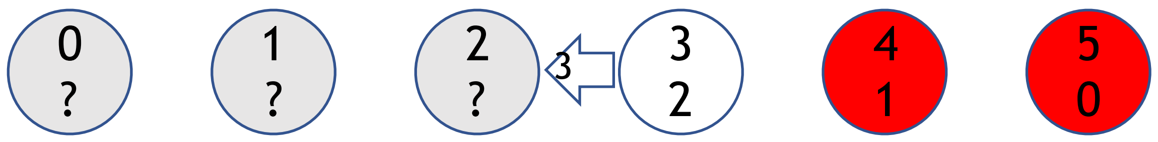
Exact Count: Round 6



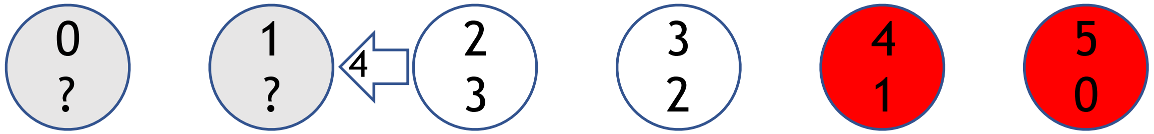
Exact Count: Round 7



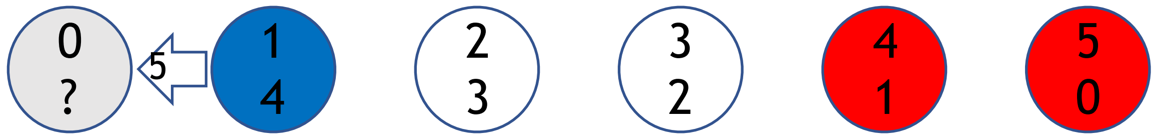
Exact Count: Round 8



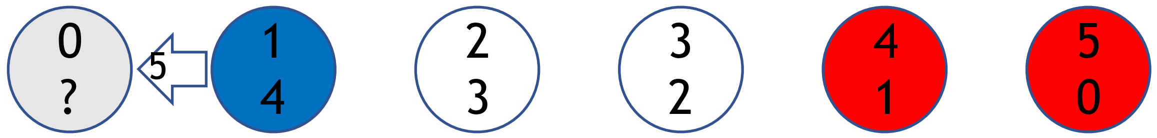
Exact Count: Round 9



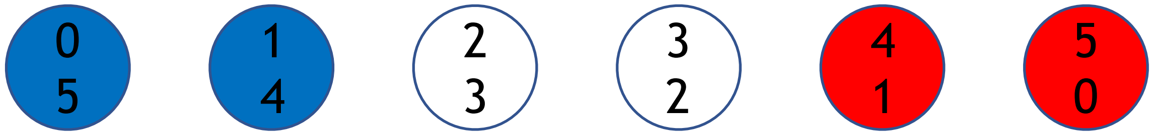
Exact Count: Round 10



Exact Count: Round 10



Exact Count: Round 11



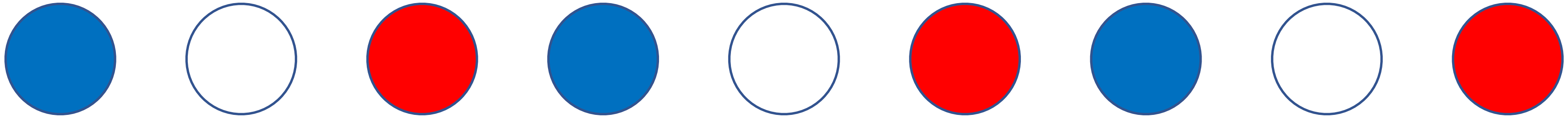
Exact Count

- Naïve algorithm complexity
  - $O(\log n)$ bits per message
  - $O(\log n)$ memory per agent
- Can be used asynchronously
- Could also start anywhere (not just at the border)

- Can we decrease the message complexity further?

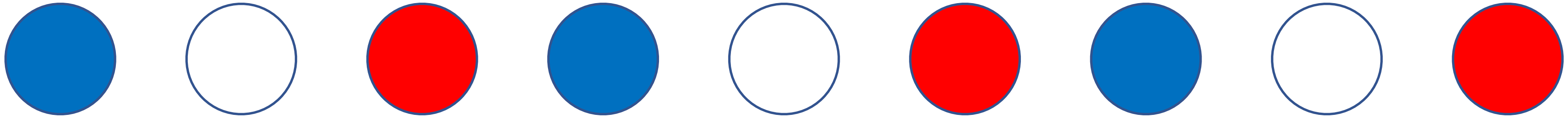
Bubble Sort

- Rough idea:
- First: alternate colors

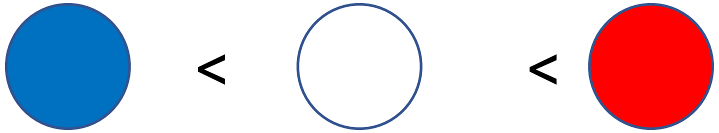


Bubble Sort

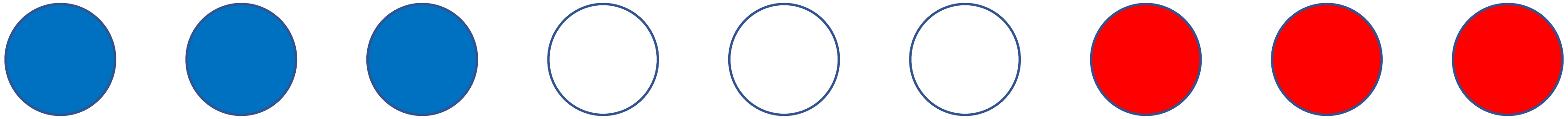
- Rough idea:
- First: alternate colors



- Bubble sort (swap locally) using



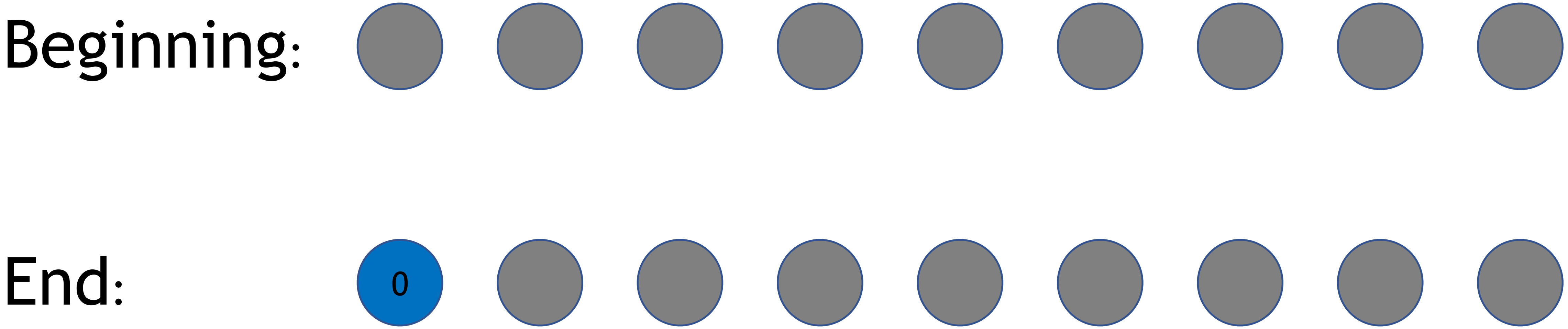
- Result:



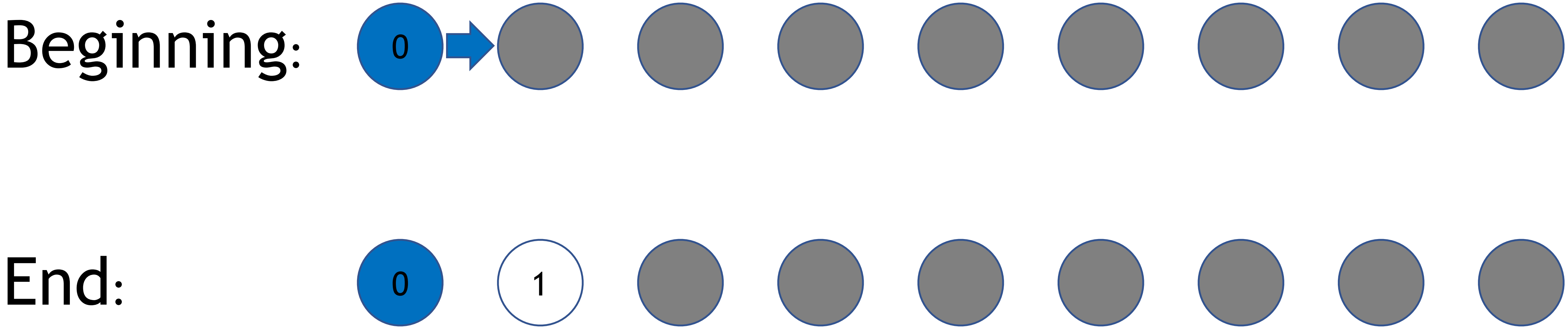
Bubble Sort

- Assigns colors in required ratios and performs distributed bubble sort
- Define which pairs can swap in a given round:
  - Either odd or even agents can swap in each round (parity)
  - Agents may only swap towards the right
- $O(\log k)$ memory per agent and bits per message – constant!

Bubble Sort: Round 0



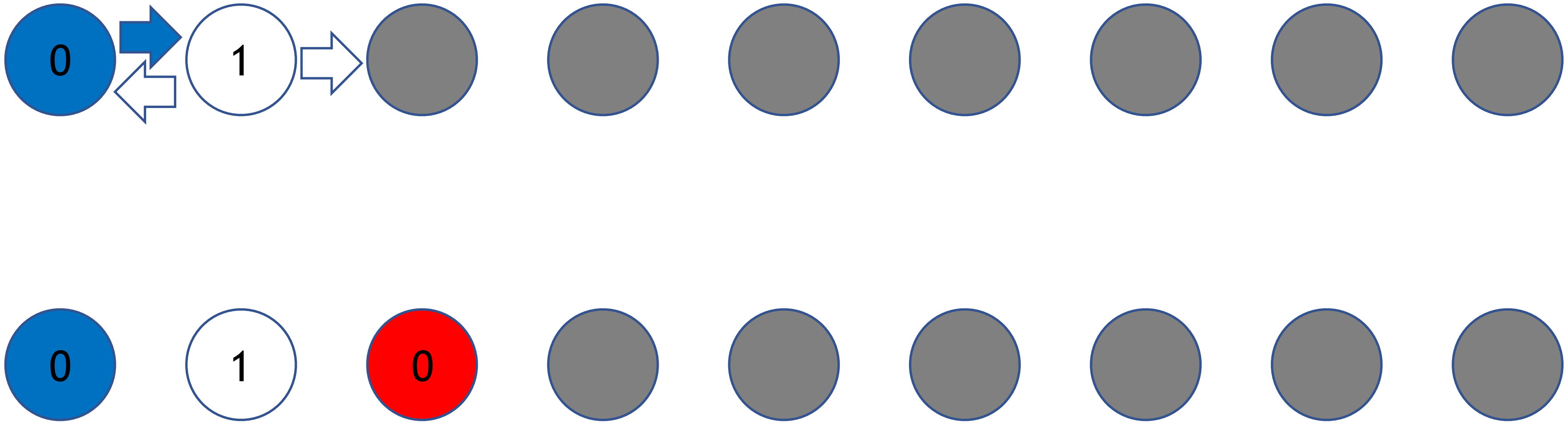
Bubble Sort: Round 1



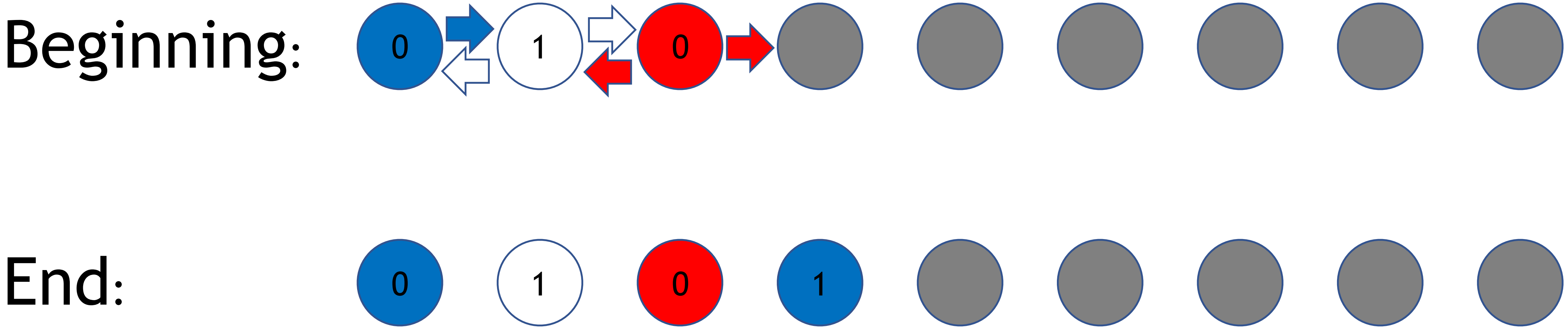
Bubble Sort: Round 2

Beginning:

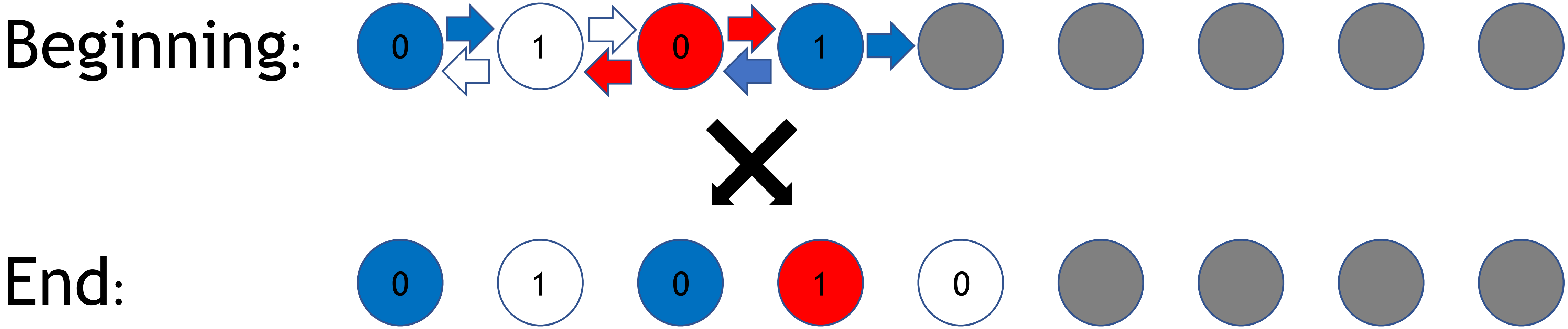
End:



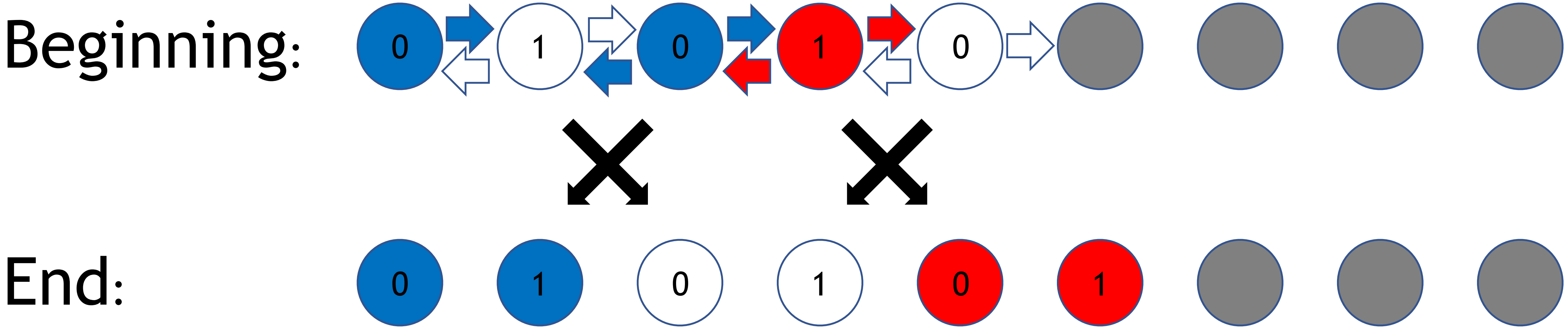
Bubble Sort: Round 3



Bubble Sort: Round 4



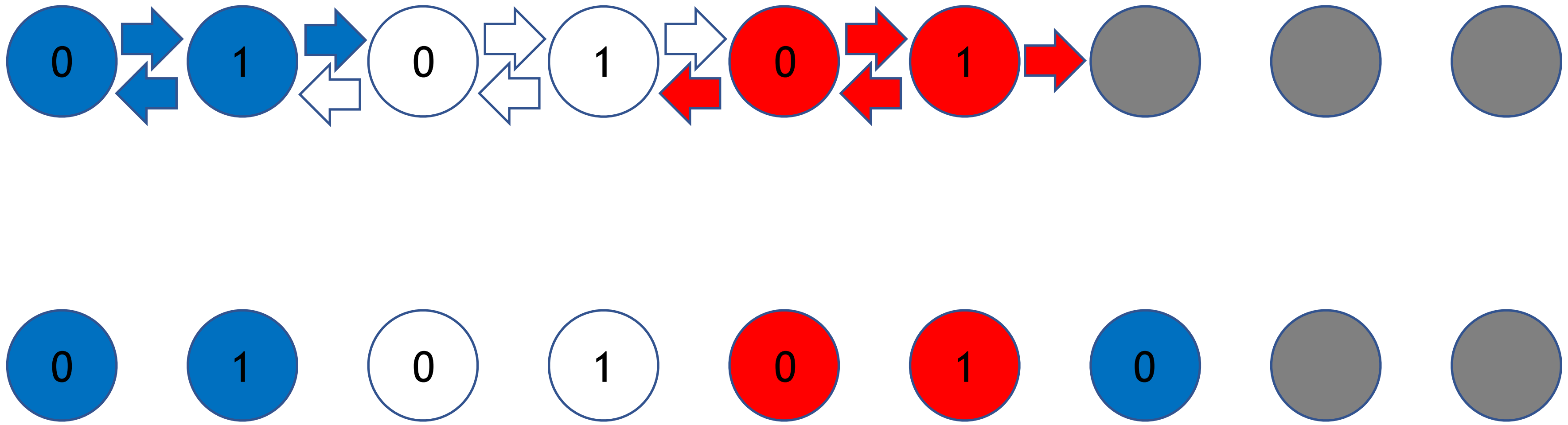
Bubble Sort: Round 5



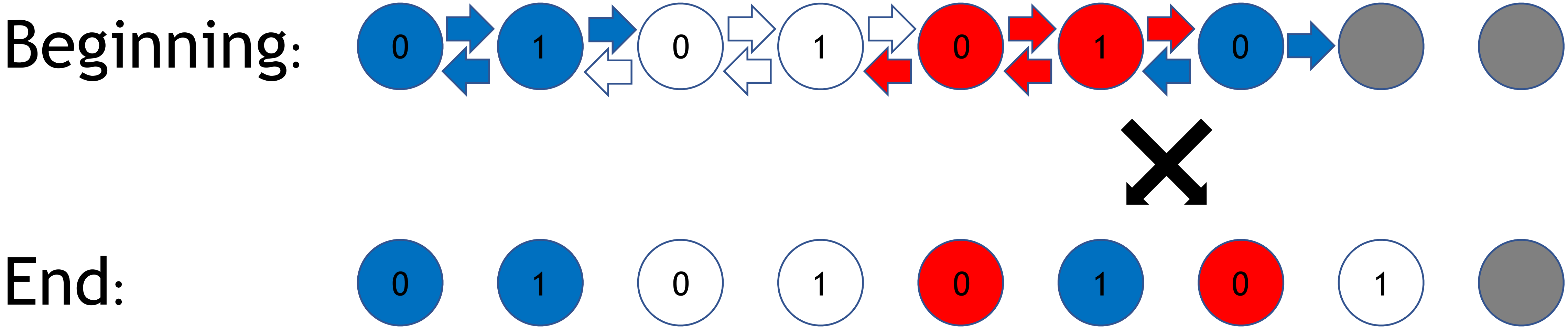
Bubble Sort: Round 6

Beginning:

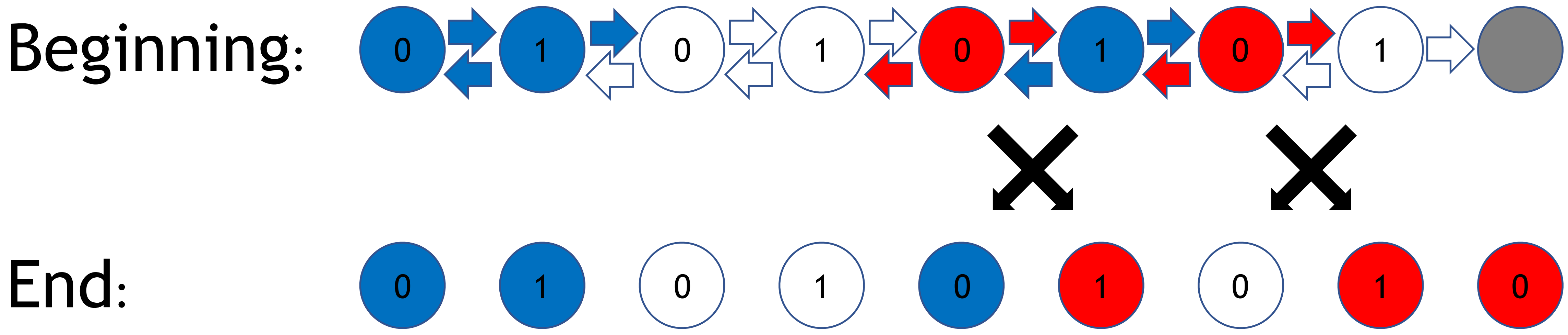
End:



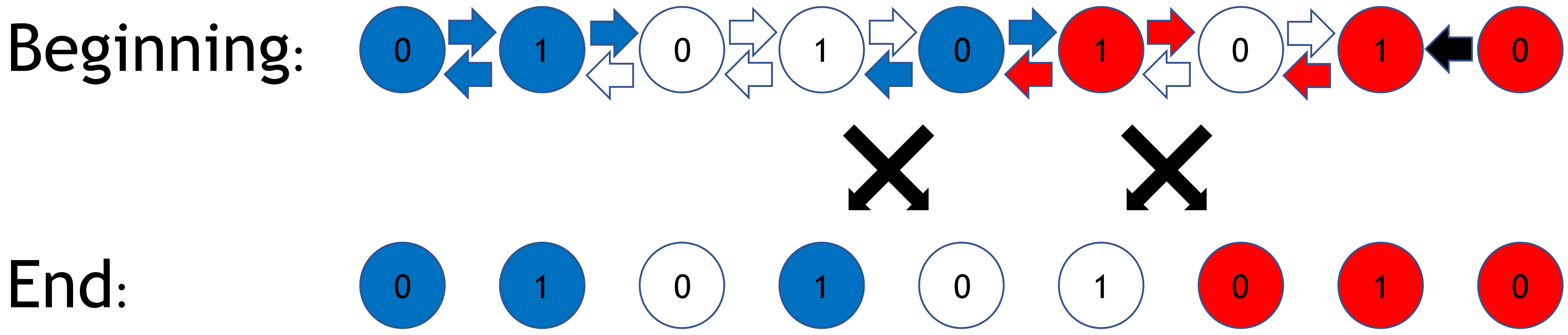
Bubble Sort: Round 7



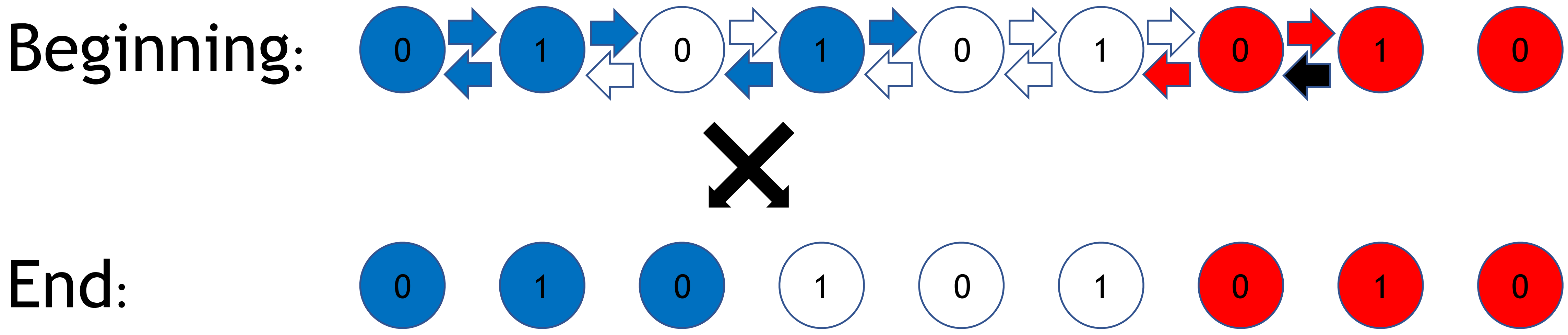
Bubble Sort: Round 8



Bubble Sort: Round 9

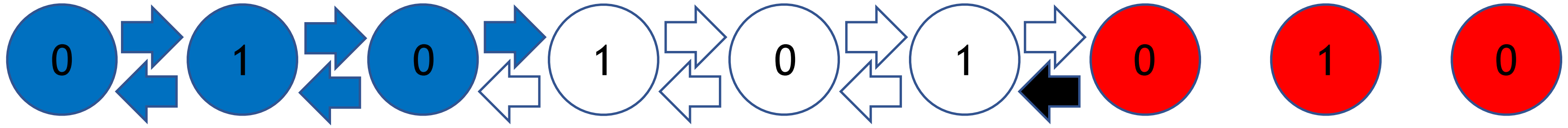


Bubble Sort: Round 10

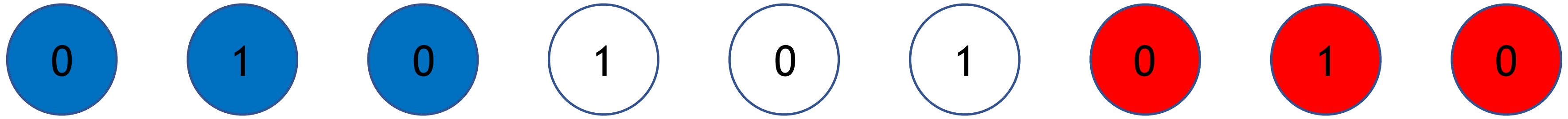


Bubble Sort: Round 11

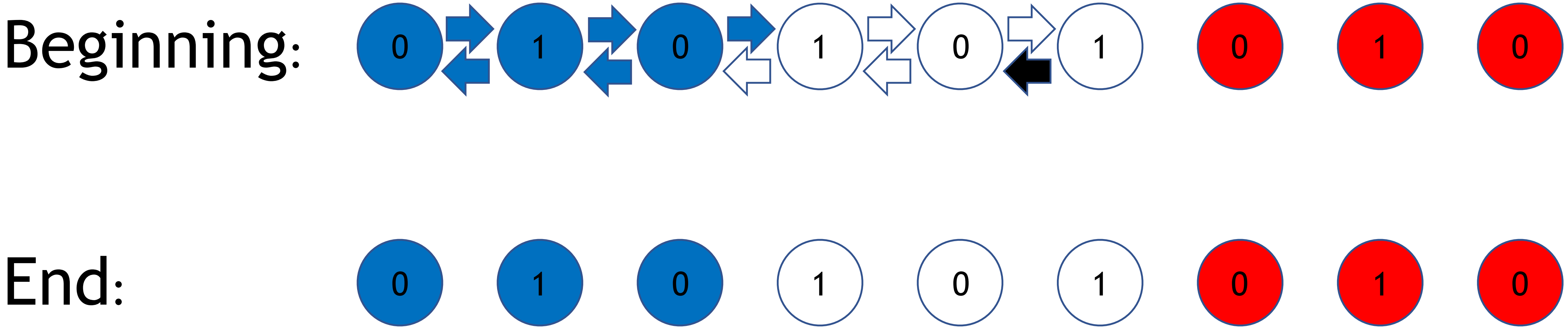
Beginning:



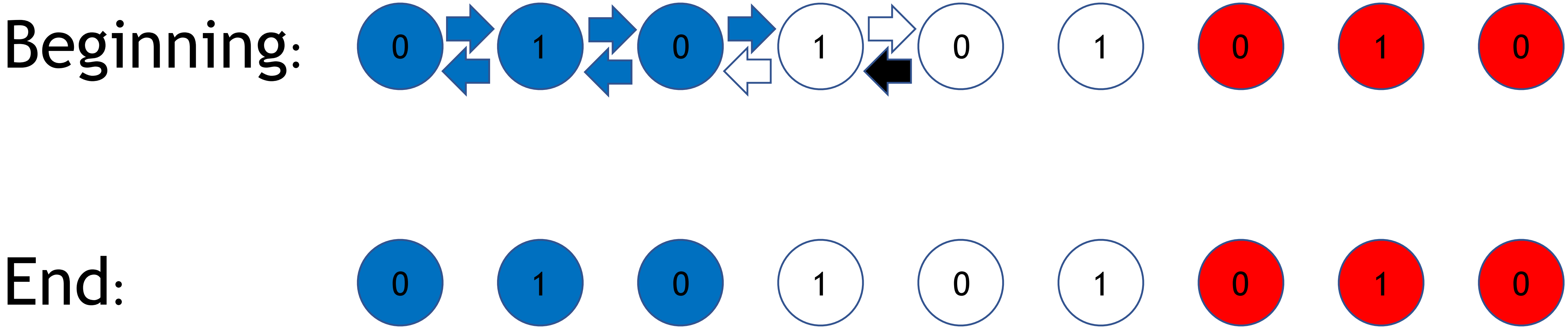
End:



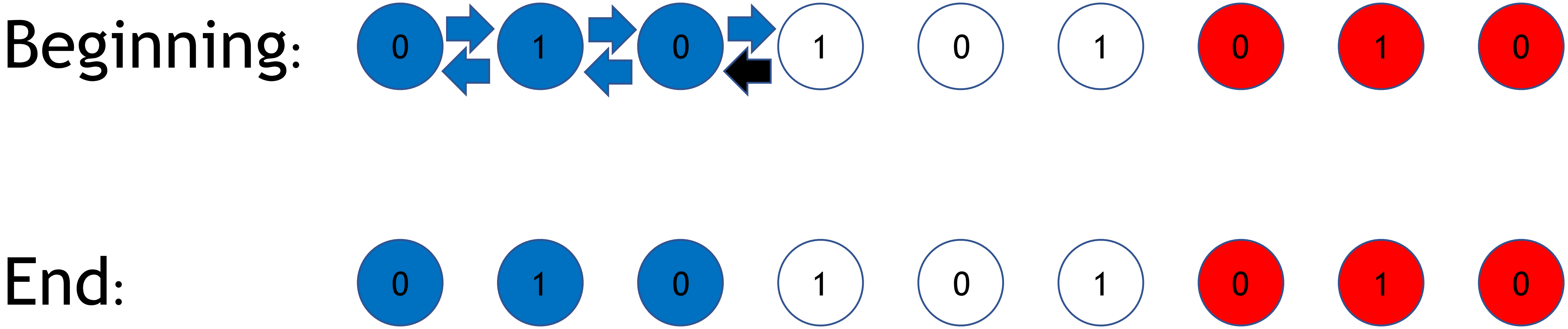
Bubble Sort: Round 12



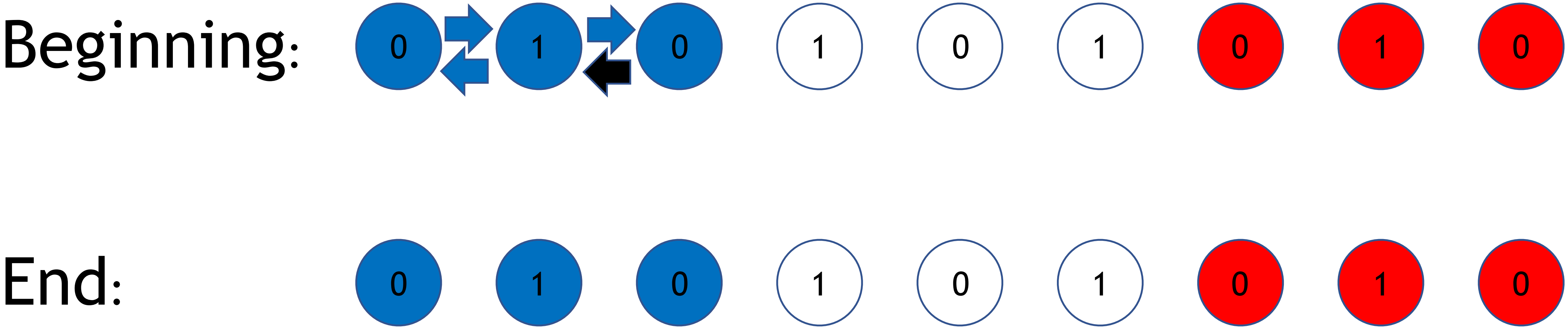
Bubble Sort: Round 13



Bubble Sort: Round 14



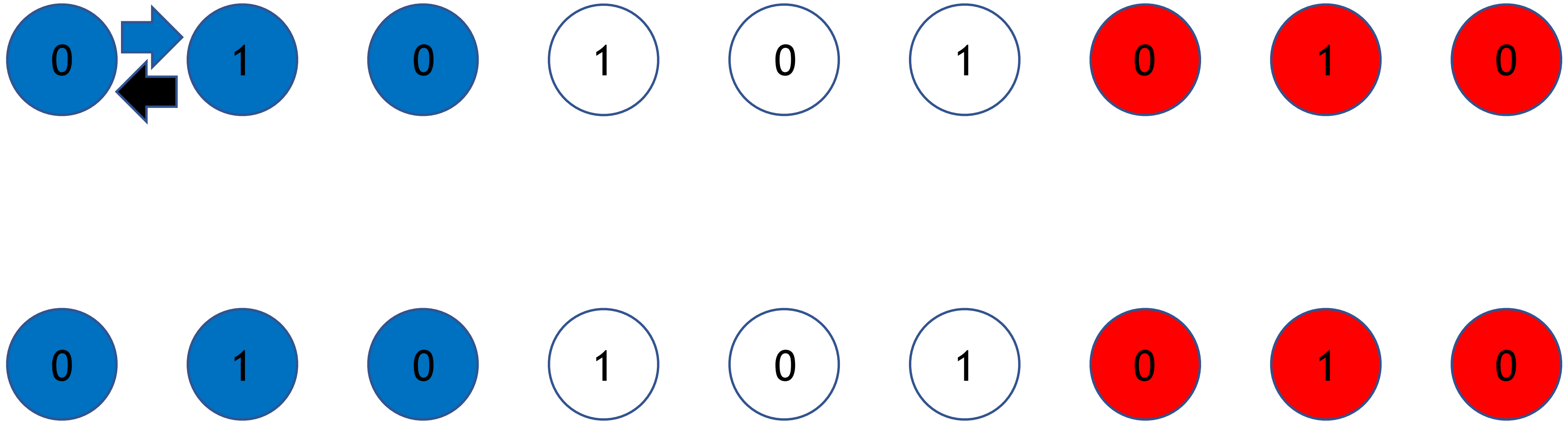
Bubble Sort: Round 15



Bubble Sort: Round 16

Beginning:

End:



1D Message Passing Algorithms

Algorithm	Rounds	Memory per agent	Bits per message	Exact
Exact Count	$O(n)$	$O(\log n)$	$O(\log n)$	✓
Exact Silent Count	$O(n)$	$O(\log n)$	$O(1)$	✓
Bubble Sort	$O(n)$	$O(\log k)$	$O(\log k)$	✓
Approx Count	$O(n)$	$O(\log\log n)$	$O(\log\log n)$	X

Other Counting Algorithms

- *Exact Silent Count*
  - $O(1)$ bits per message
  - $O(\log n)$ memory per agent
  - uses absence of a message as info by keeping track of rounds
- *Approx Count*
  - $O(\log\log n)$ bits per message
  - $O(\log\log n)$ memory per agent
  - uses approximate counting (Morris 1978, Flajolet 1985)

Conclusions

- Two models and approaches: concentration and message passing
- Concentration model: 1D vs 2D
  - 1D concentration model can use relative distance alone
  - 2D concentration model requires additional positional information
- Message passing model: standard tradeoffs
  - *Exact Count* seems fairly obvious
  - *Exact Silent Count* reduces message size (constant)
  - *Approx Count* improves space complexity (loglog$n$)
  - *Bubble Sort* has good space complexity (constant)

Merci!

Questions?

Related Work

Ancona, A., Bajwa, A., Lynch, N., Mallmann-Trenn, F. How to color a French flag–biologically inspired algorithms for scale-invariant patterning. arXiv preprint arXiv:1905.00342 (2019)

Briscoe, J., Small, S. Morphogen rules: design principles of gradient-mediated embryo patterning. Development 142(23), 3996–4009 (2015)

Flajolet, P. Approximate counting: A detailed analysis. BIT 25 (1), 113–134 (1985)

Lander, A ., Nie, Q., Wan, F. Do morphogen gradients arise by diffusion? Developmental Cell, 2(6), 785-796 (2002)

Morris, R. Counting large numbers of events in small registers. Commun. ACM, 21(10):840–842 (1978)

Nüsslein-Volhard, C., Wieschaus, E. Mutations affecting segment number and polarity in drosophila. Nature 287(5785), 795–801 (1980)

Wolpert, L. Positional information and the spatial pattern of cellular differentiation. Journal of Theoretical Biology 25 (1), 1 – 47 (1969)