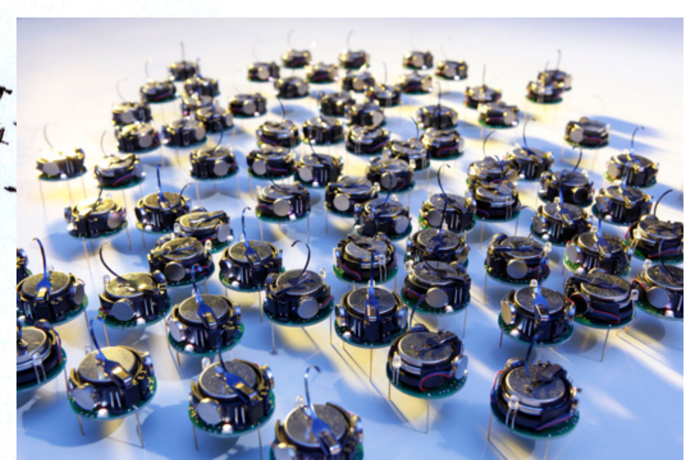
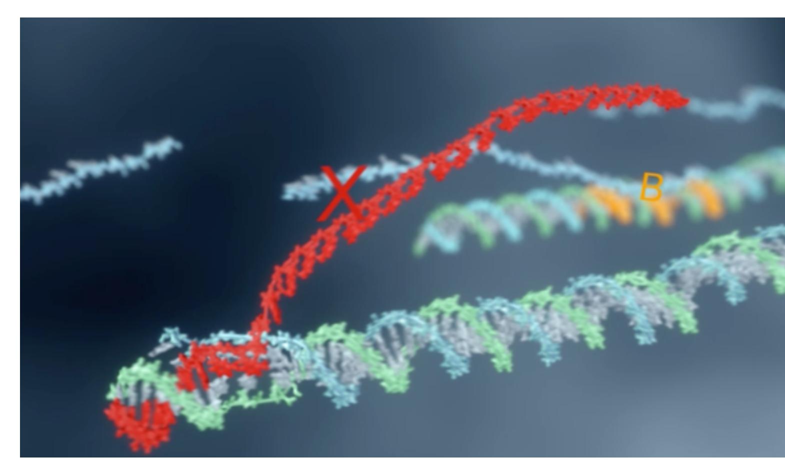
Computation with Anonymous Finite-State Agents

EE382N
Embedded System Design and Modeling
Guest Lecture

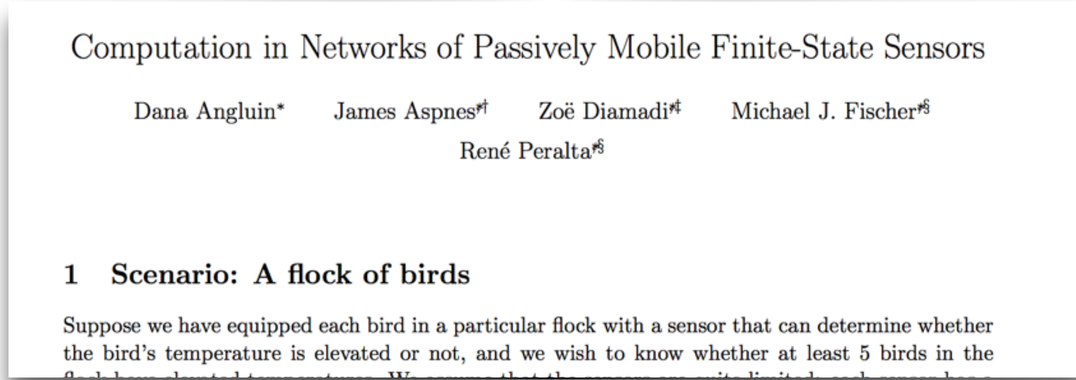
David Soloveichik





Outline

- Population protocols model
- Examples of "deterministic" computation
- Formally defining "deterministic computation": stable computation
- Time model and computational complexity
- Consensus / approximate majority algorithm
- Biological connections
- Programming molecular interactions


Computation in Networks of Passively Mobile Finite-State Sensors

Dana Angluin* James Aspnes*† Zoë Diamadi*‡ Michael J. Fischer*§
René Peralta*§

1 Scenario: A flock of birds

Suppose we have equipped each bird in a particular flock with a sensor that can determine whether the bird's temperature is elevated or not, and we wish to know whether at least 5 birds in the

"Population Protocols"

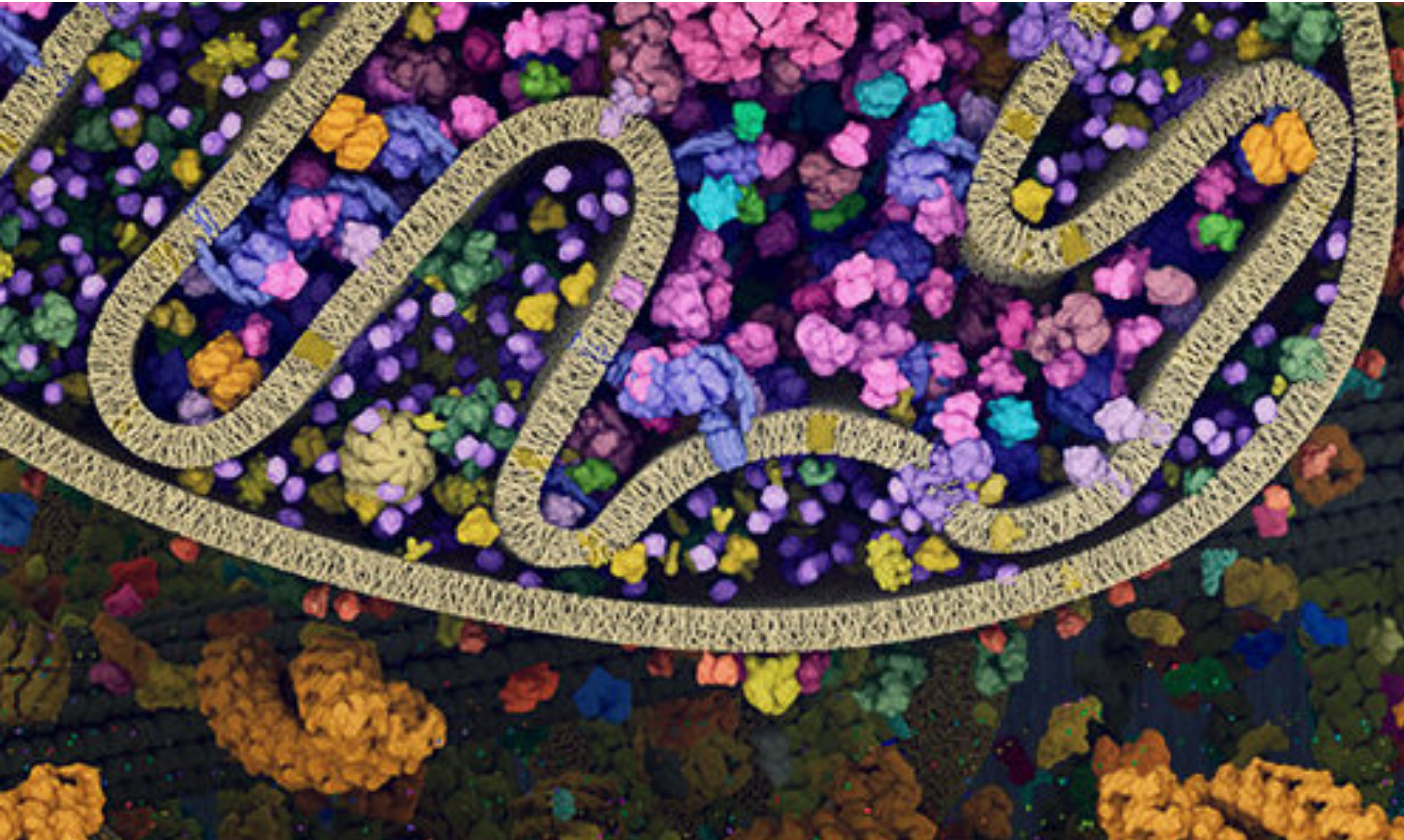
"Suppose we have equipped each bird in a particular flock with a sensor that can determine whether the bird's temperature is elevated or not, and we wish to know whether at least 5 birds [or at least 5%] in the flock have elevated temperatures."

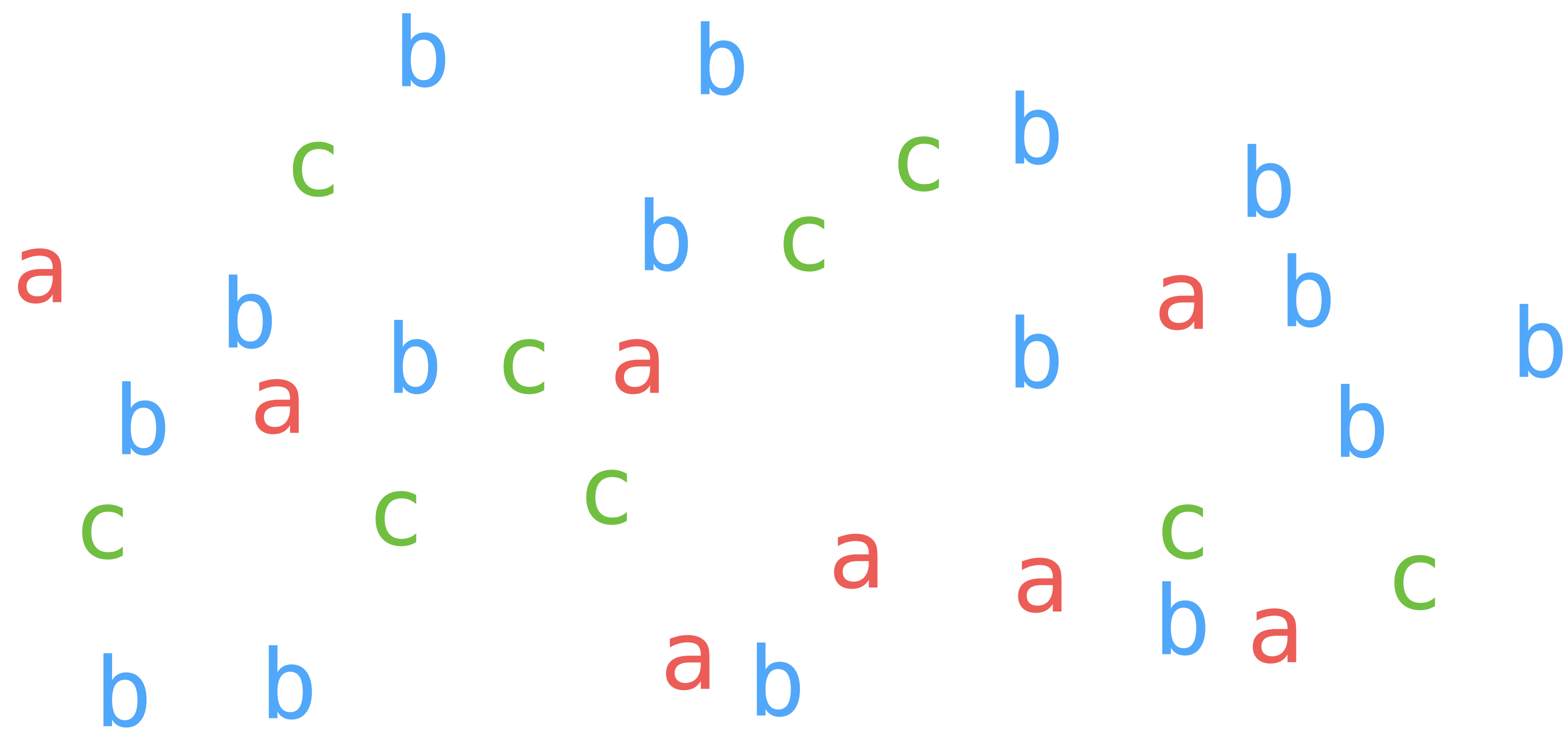
"In systems consisting of **massive amounts of cheap, bulk-produced hardware**, or of small mobile agents that are tightly constrained by the systems they run on, the resources available at each agent may be **severely limited**."

Designer does not have control over interactions between agents (i.e., which agents interact next).

[Angluin, Diamadi, Fischer, Peralta PODC 2004]

Agents ↔ Molecules



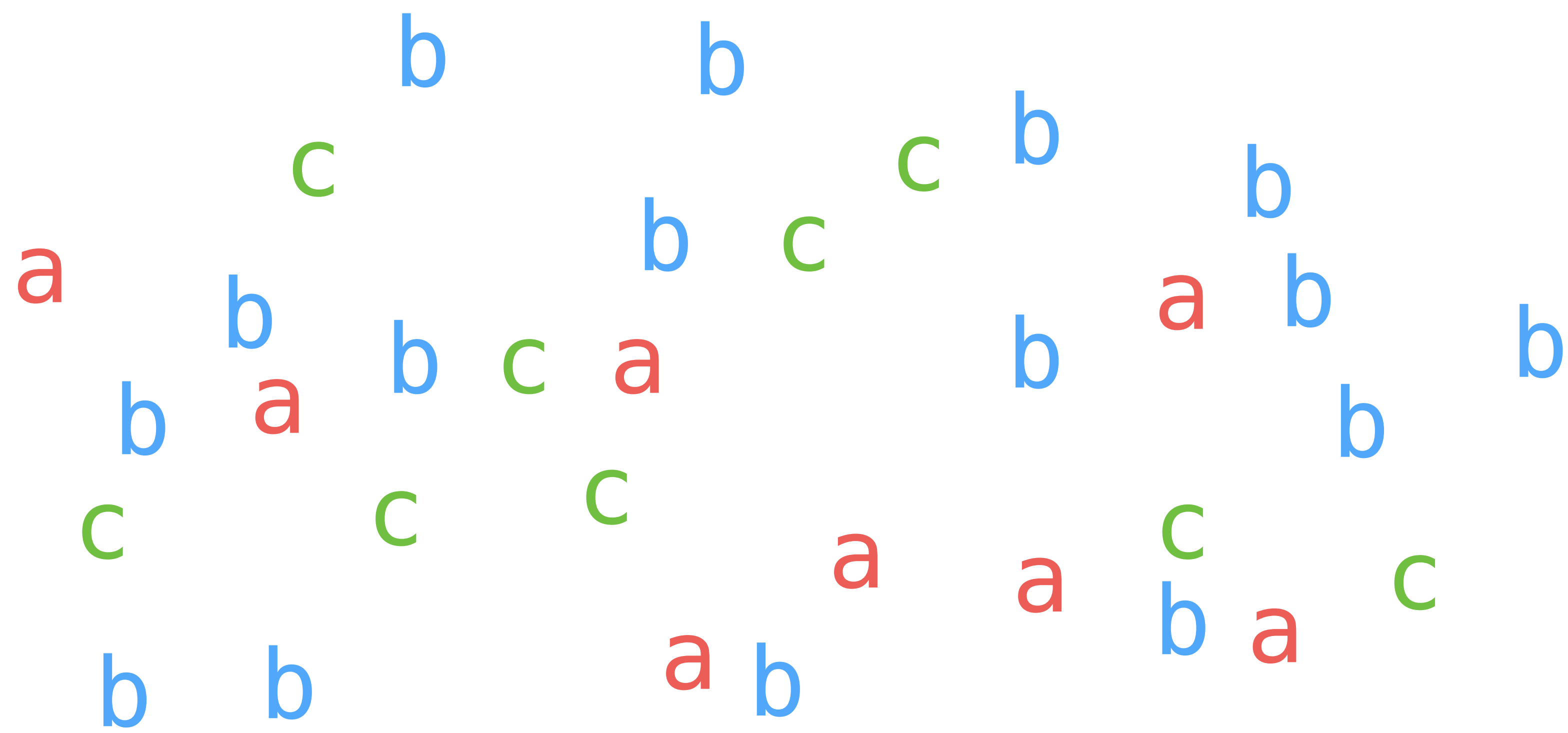


n = number of agents

<u>anonymous</u> <u>finite-state</u>

- no unique id
- can't tell if interacts twice with the same agent

- agents have finite memory (independent of n)



n = number of agents

fully connected interaction graph

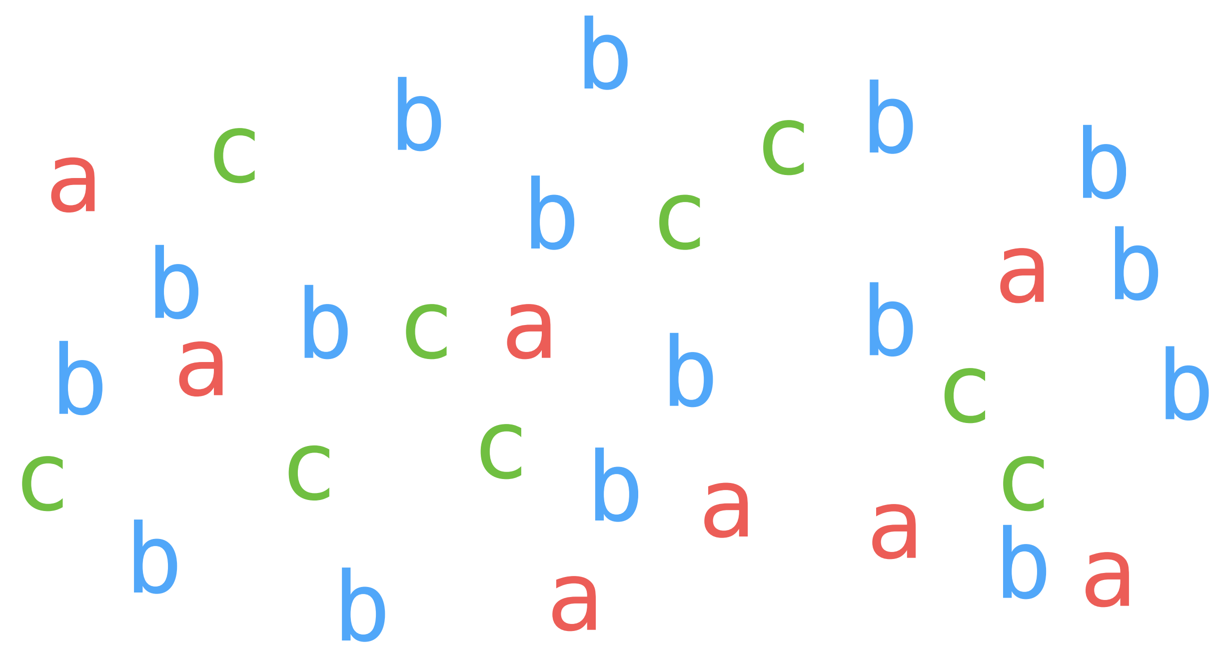
- any pair of agents could interact next
- physical intuition: interact when happen to come close, but no control over movement

"Well-mixed" interaction model: any two agents equally likely to interact next

Configuration: counts of agents in each state

Transition (interaction) rules: describes how states of two interacting agents change

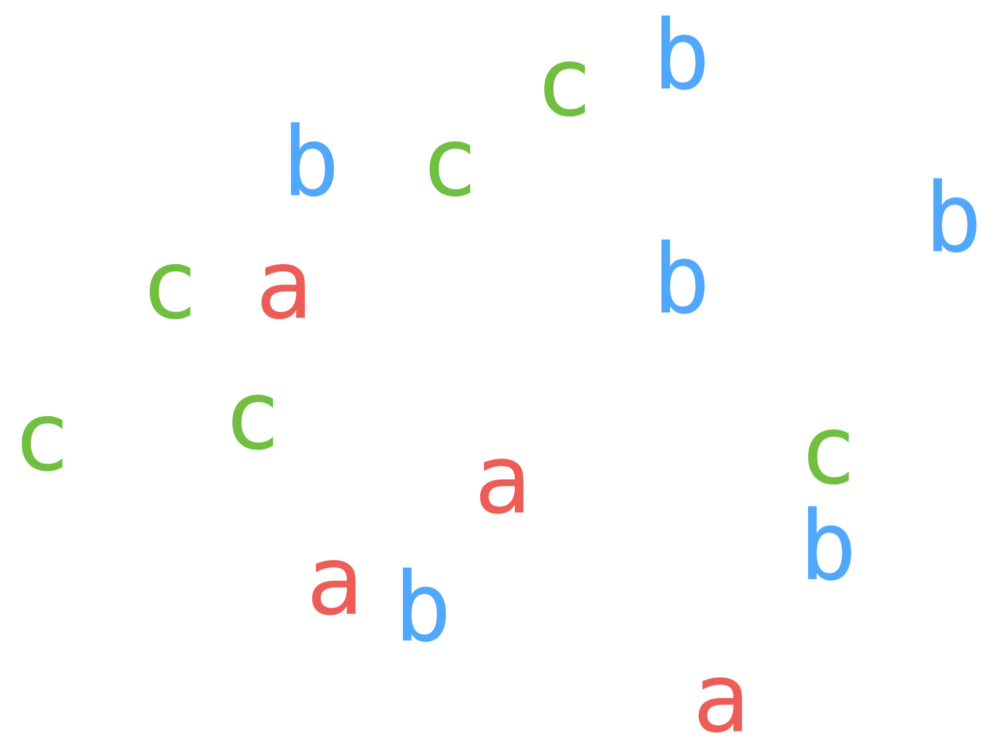
a, a → c, c
a, c → a, a
b, c → b, b



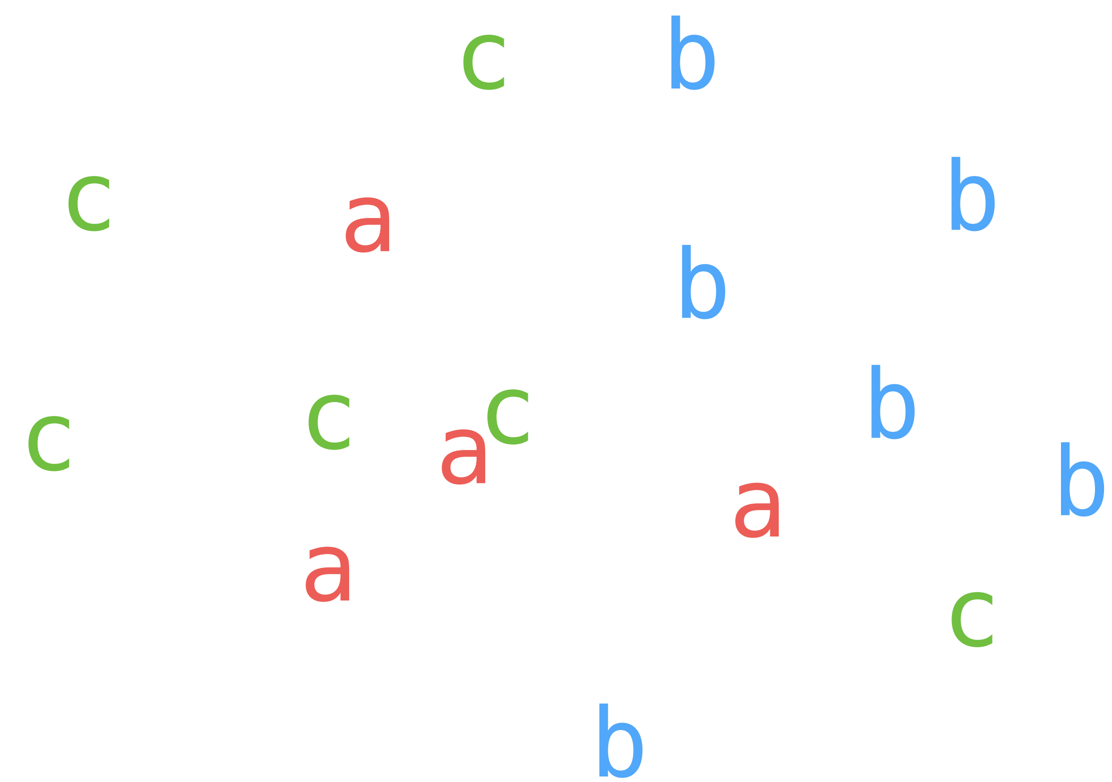
8a 16b 9c

a,a → c,c
a,c → a,a
b,c → b,b

4a 6b 6c

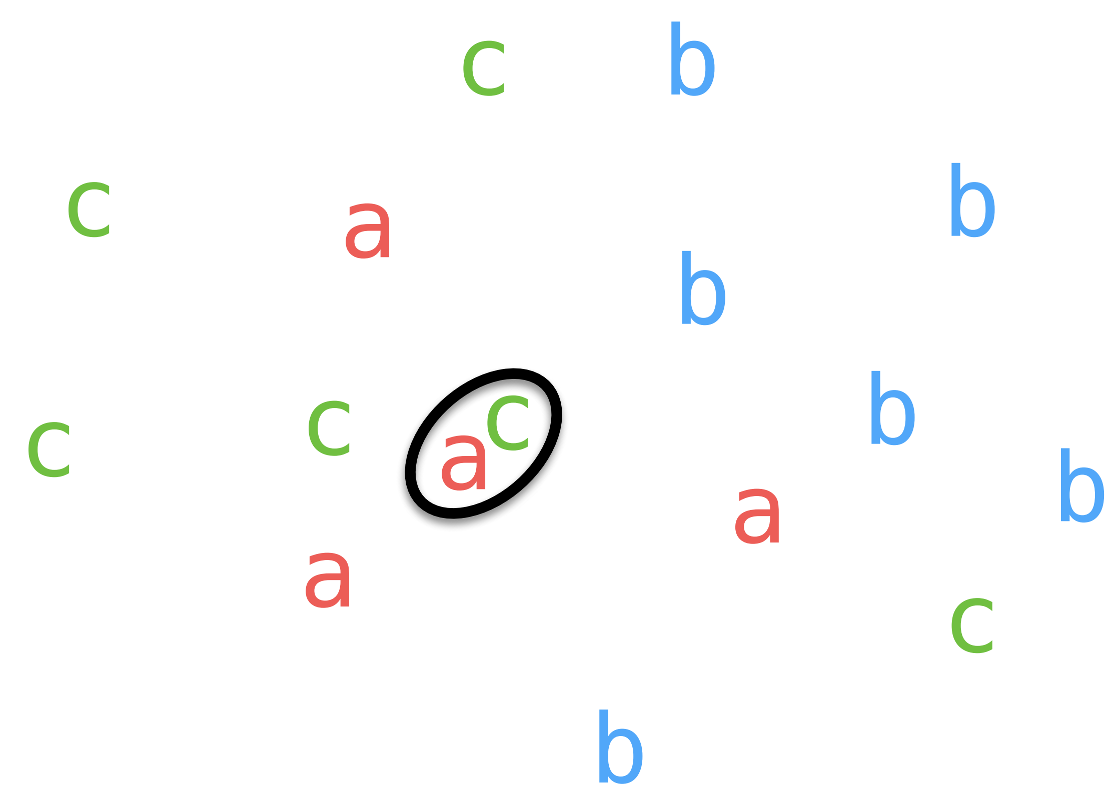


a, a → c, c
a, c → a, a
b, c → b, b



4a 6b 6c

a,a → c,c
a,c → a,a
b,c → b,b



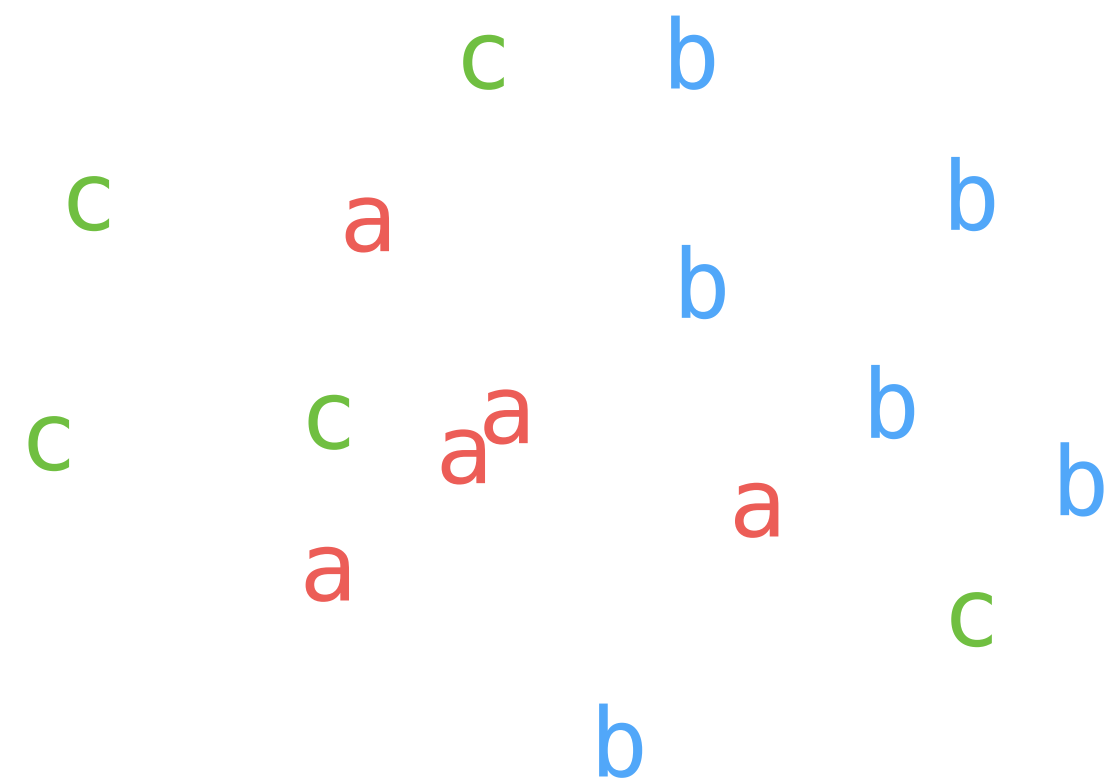
4a 6b 6c

a,a	→	c,c
a,c	→	a,a
b,c	→	b,b

4a 6b 6c

⬇

5a 6b 5c



$a, a \rightarrow c, c$

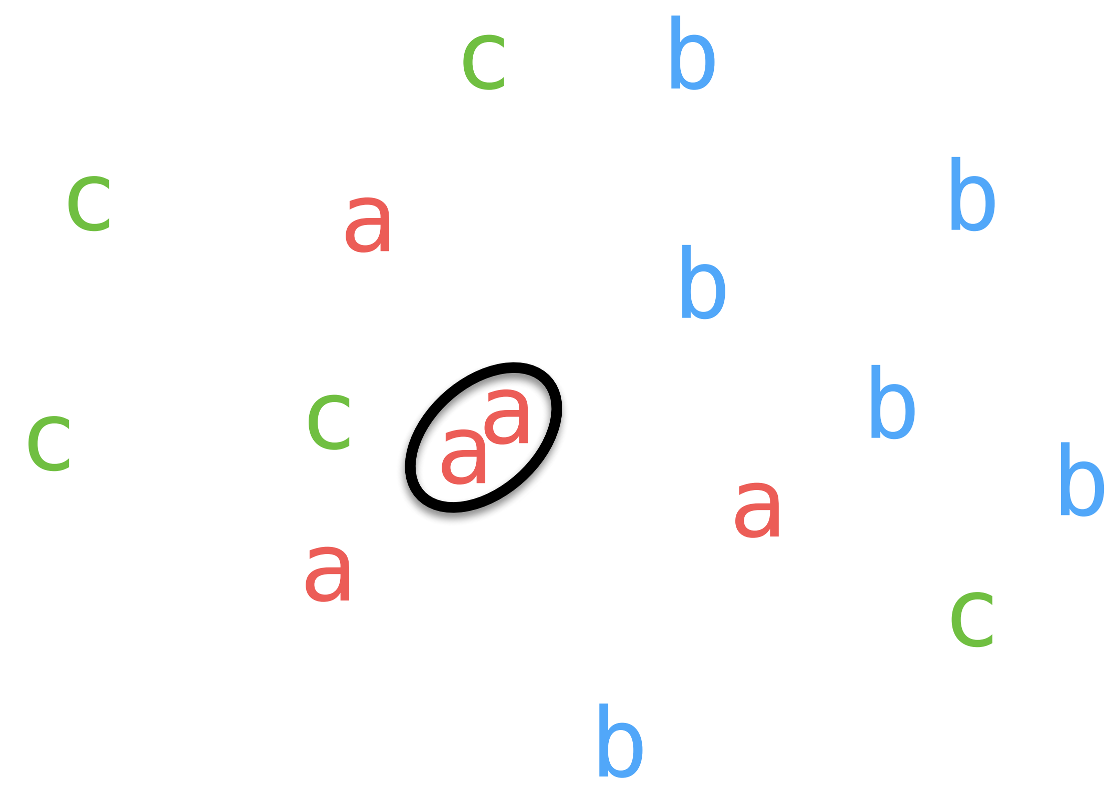
$a, c \rightarrow a, a$

$b, c \rightarrow b, b$

4a 6b 6c

⬇

5a 6b 5c



$a, a \rightarrow c, c$

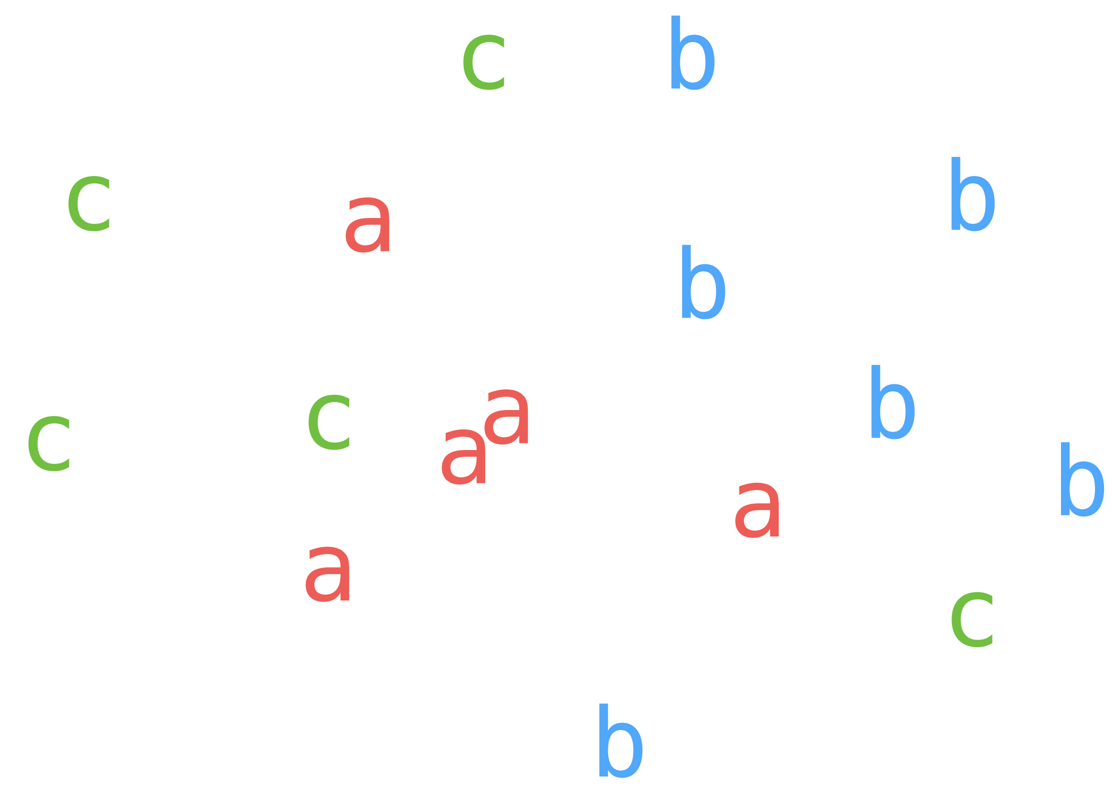
$a, c \rightarrow a, a$

$b, c \rightarrow b, b$

4a 6b 6c

⬇

5a 6b 5c

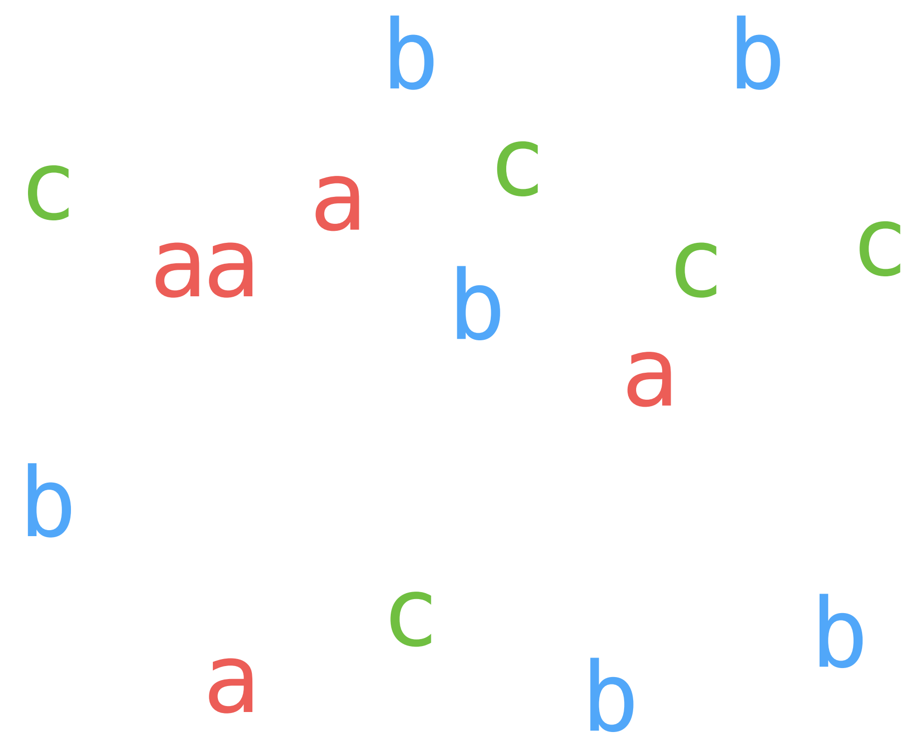


a,a → c,c
a,c → a,a
b,c → b,b

4a 6b 6c

⬇

5a 6b 5c

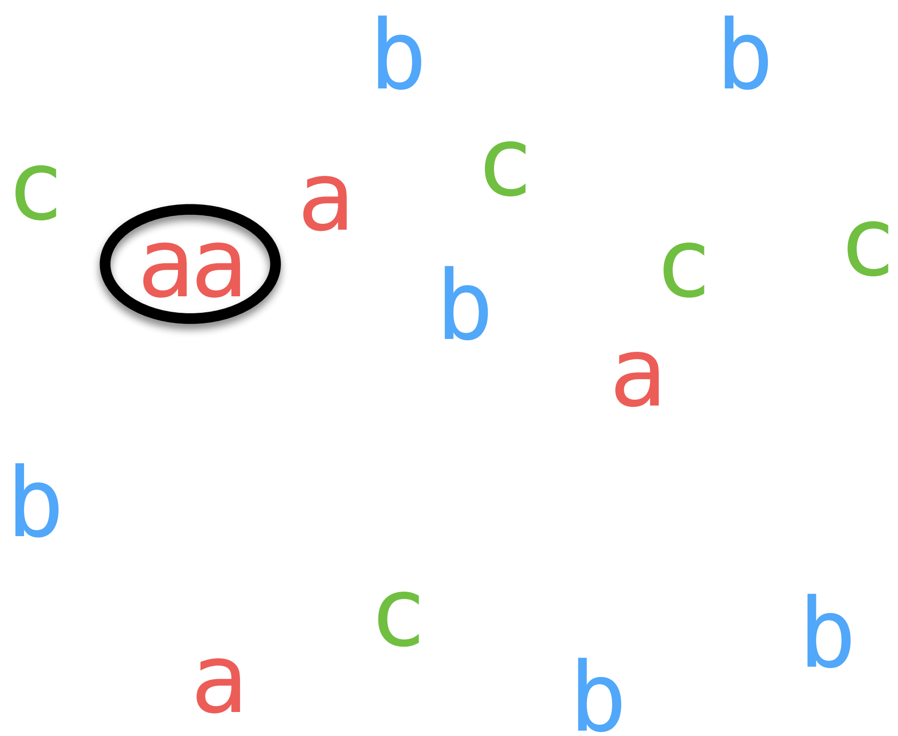


a,a → c,c
a,c → a,a
b,c → b,b

4a 6b 6c

↓

5a 6b 5c



a,a	→	c,c
a,c	→	a,a
b,c	→	b,b

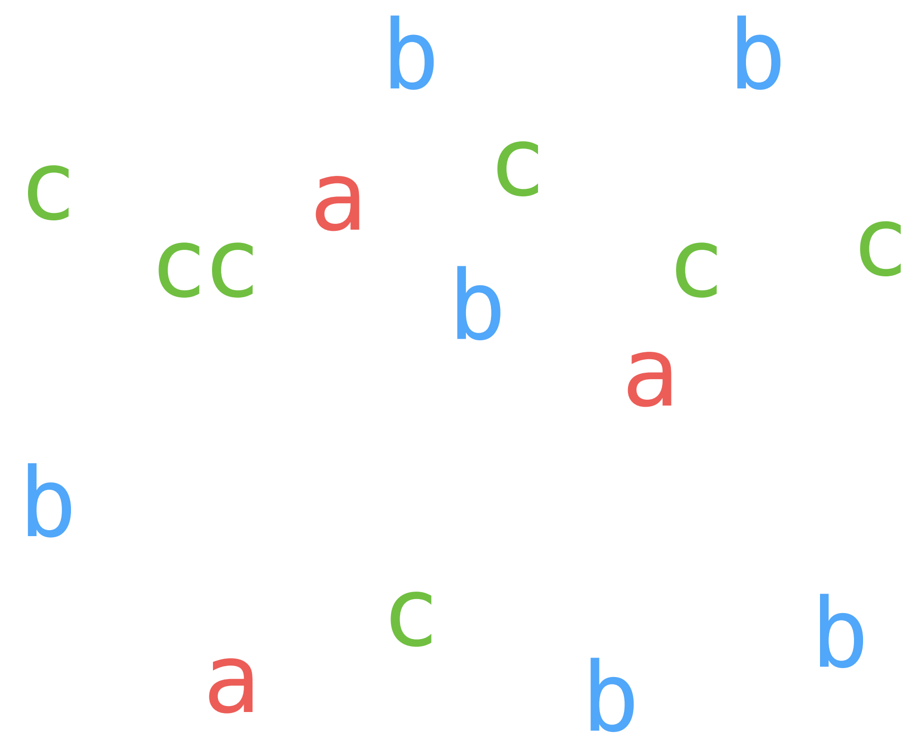
4a 6b 6c

⬇

5a 6b 5c

⬇

3a 6b 7c



a,a → c,c
a,c → a,a
b,c → b,b

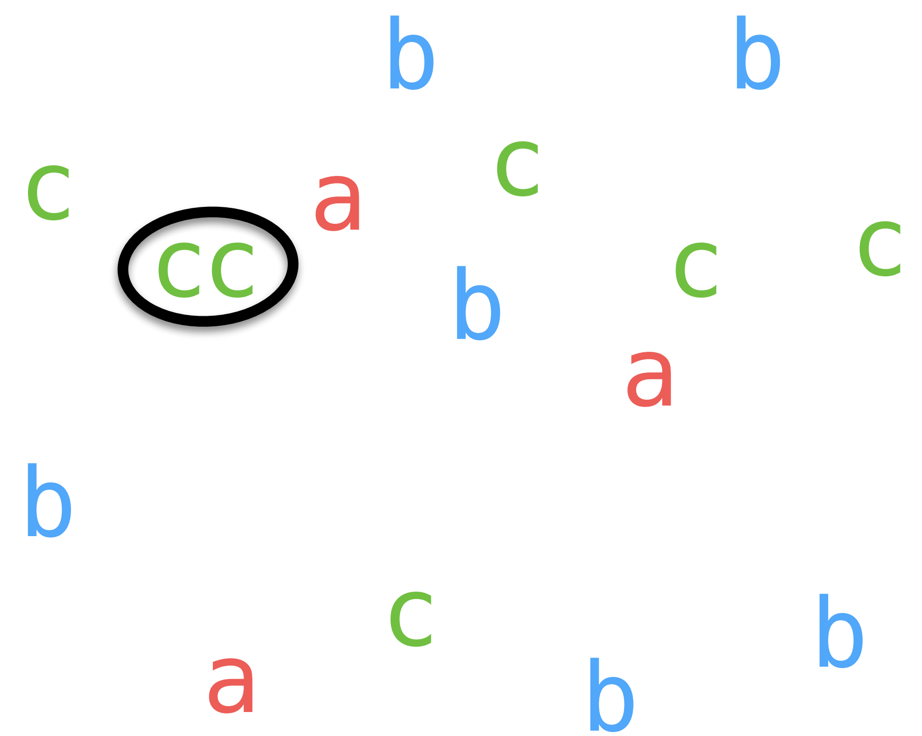
4a 6b 6c

⬇

5a 6b 5c

⬇

3a 6b 7c



a,a → c,c
a,c → a,a
b,c → b,b

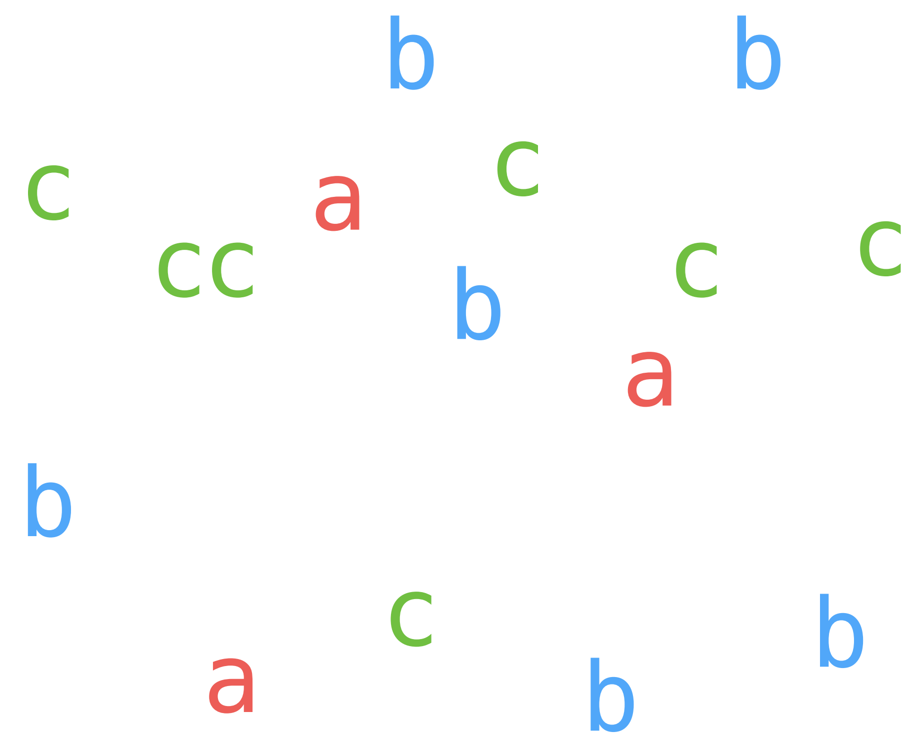
4a 6b 6c

↓

5a 6b 5c

↓

3a 6b 7c

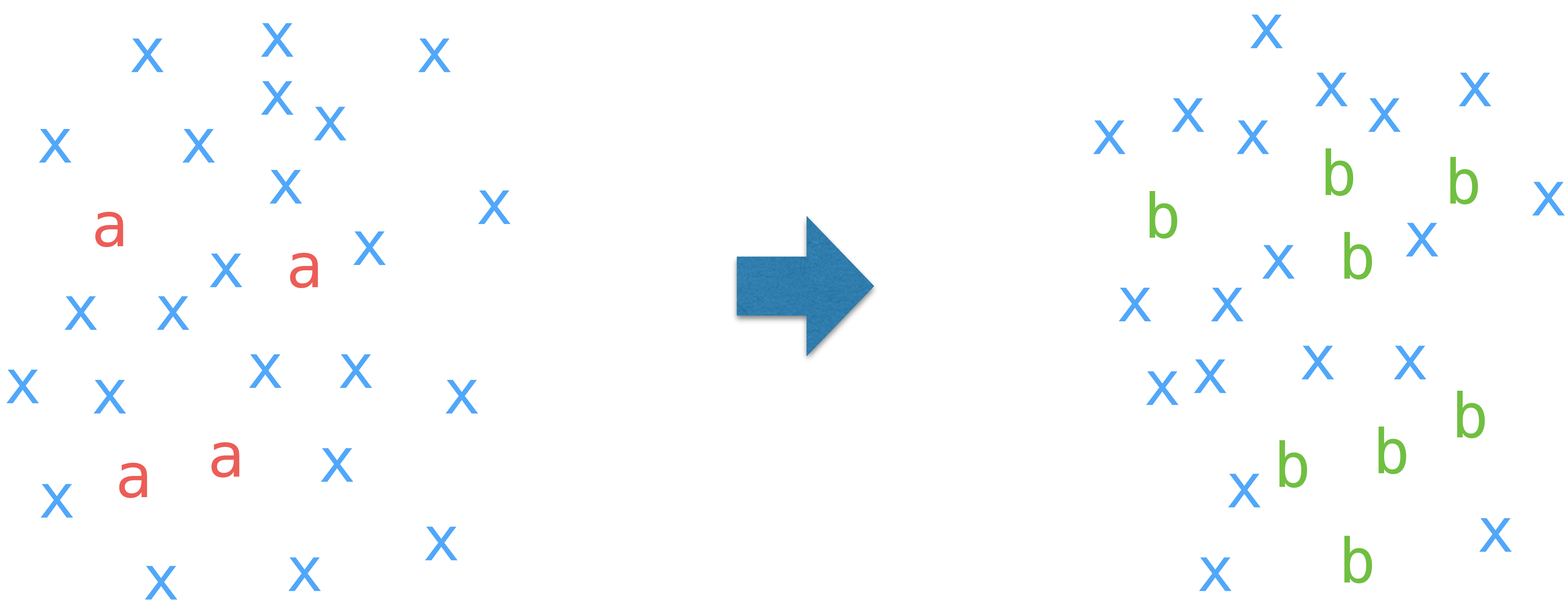


Outline

- Population protocols model
- Examples of "deterministic" computation
- Formally defining "deterministic computation": stable computation
- Time model and computational complexity
- Consensus / approximate majority algorithm
- Biological connections
- Programming molecular interactions

Example: Multiplication by 2

$$b = a \cdot 2$$

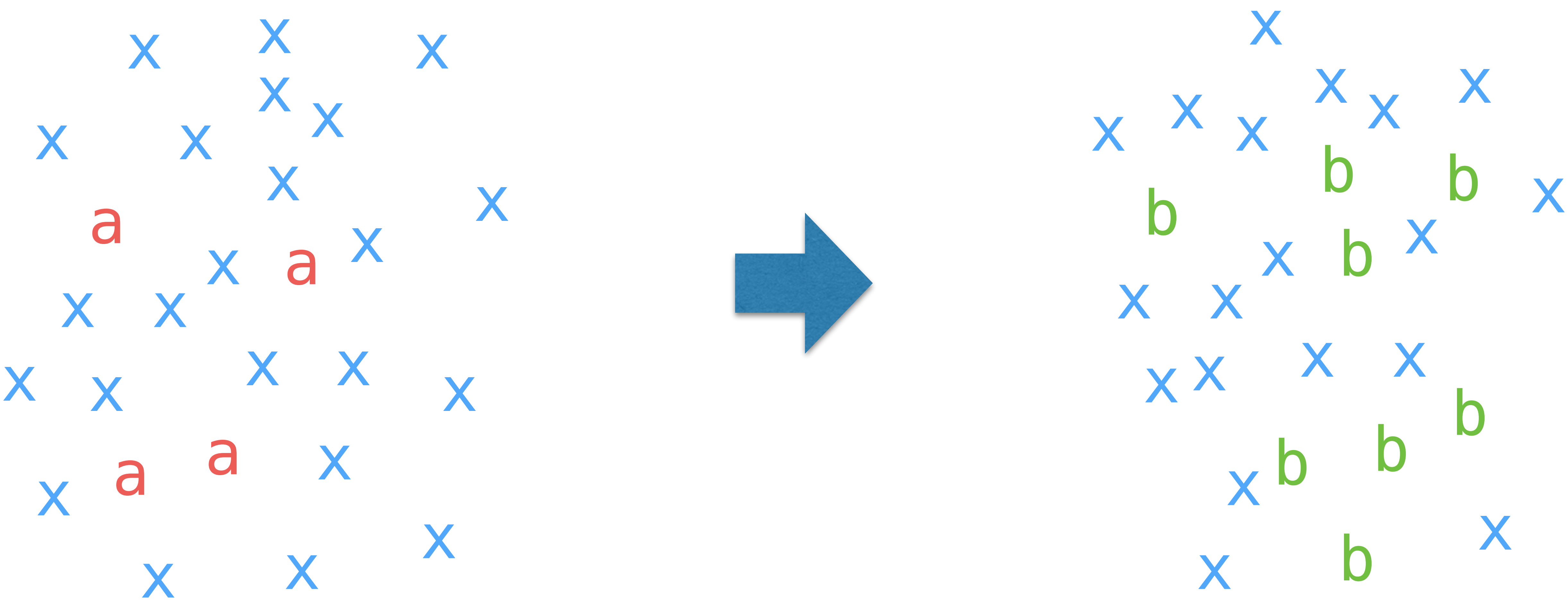


4 a

8 b

Example: Multiplication by 2

$$b = a \cdot 2$$



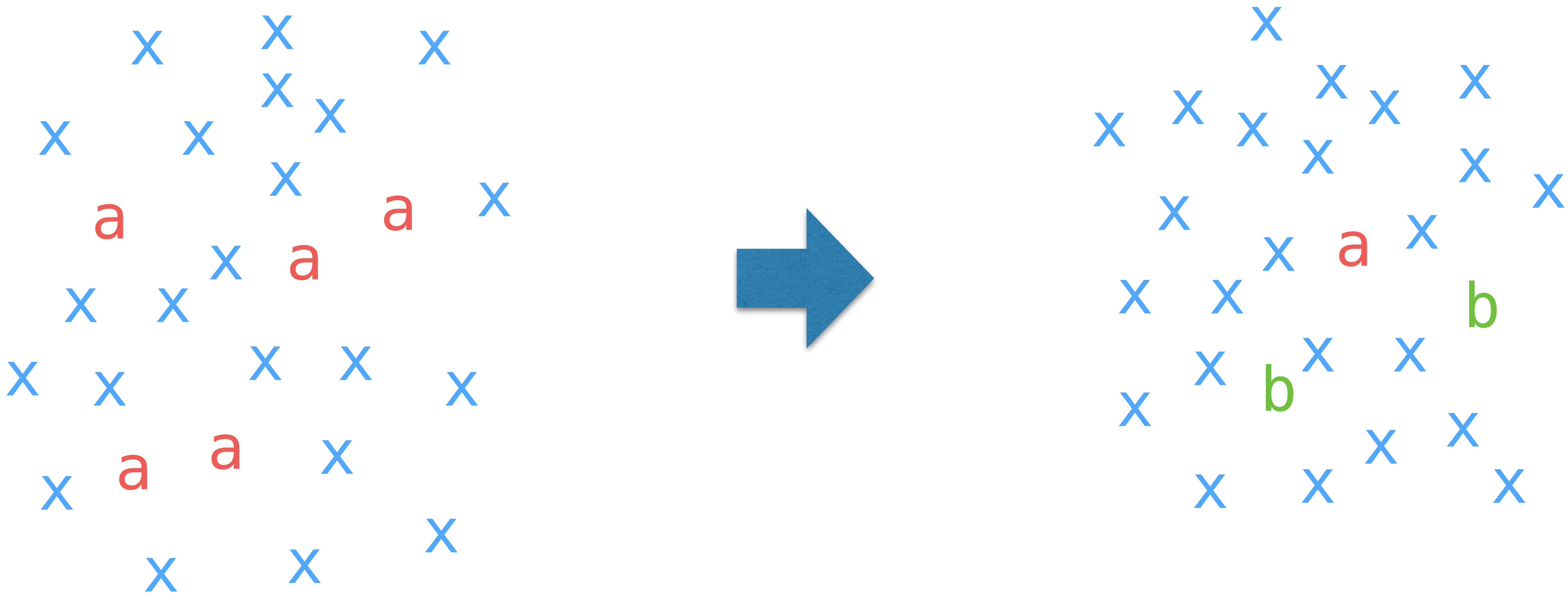
$$\boxed{a, x \rightarrow b, b}$$

4 a

8 b

Example: Division by 2

$$b = \lfloor a/2 \rfloor$$

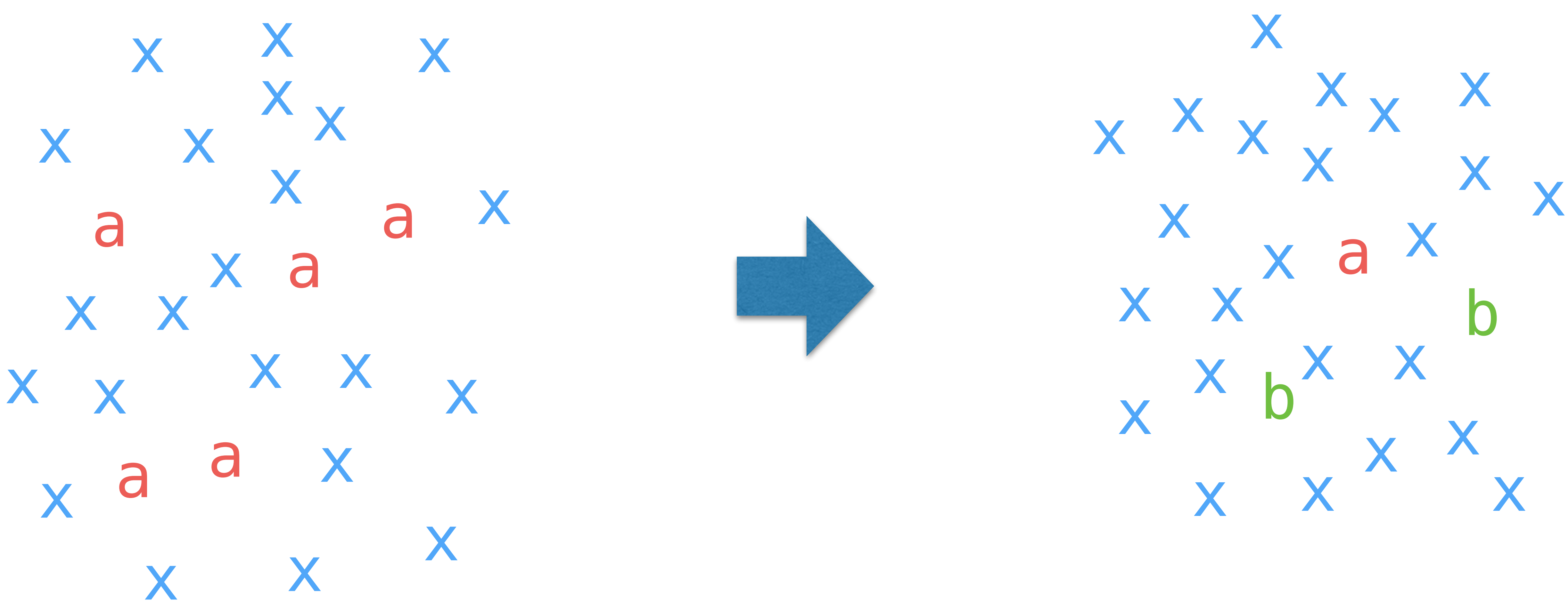


5 a

2 b

Example: Division by 2

$$b = \lfloor a/2 \rfloor$$



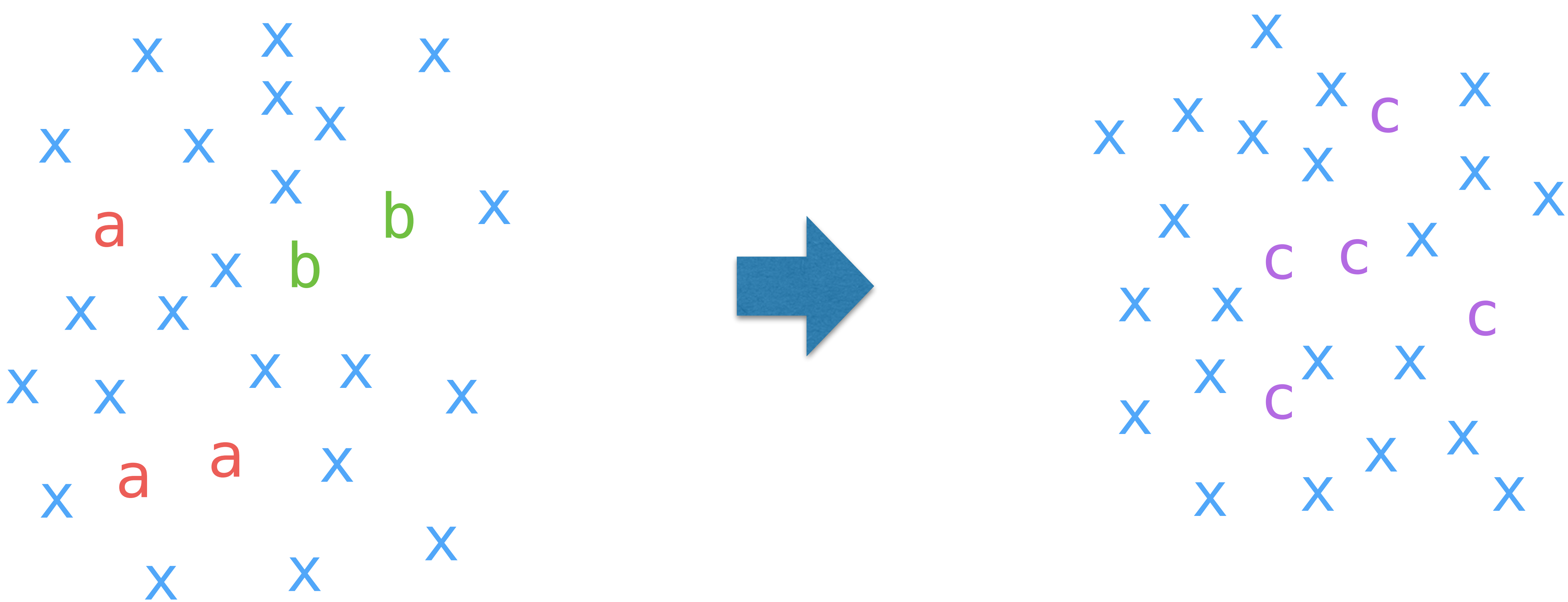
$$\boxed{a, a \rightarrow b, x}$$

5 a

2 b

Example: Addition

$$c = a + b$$



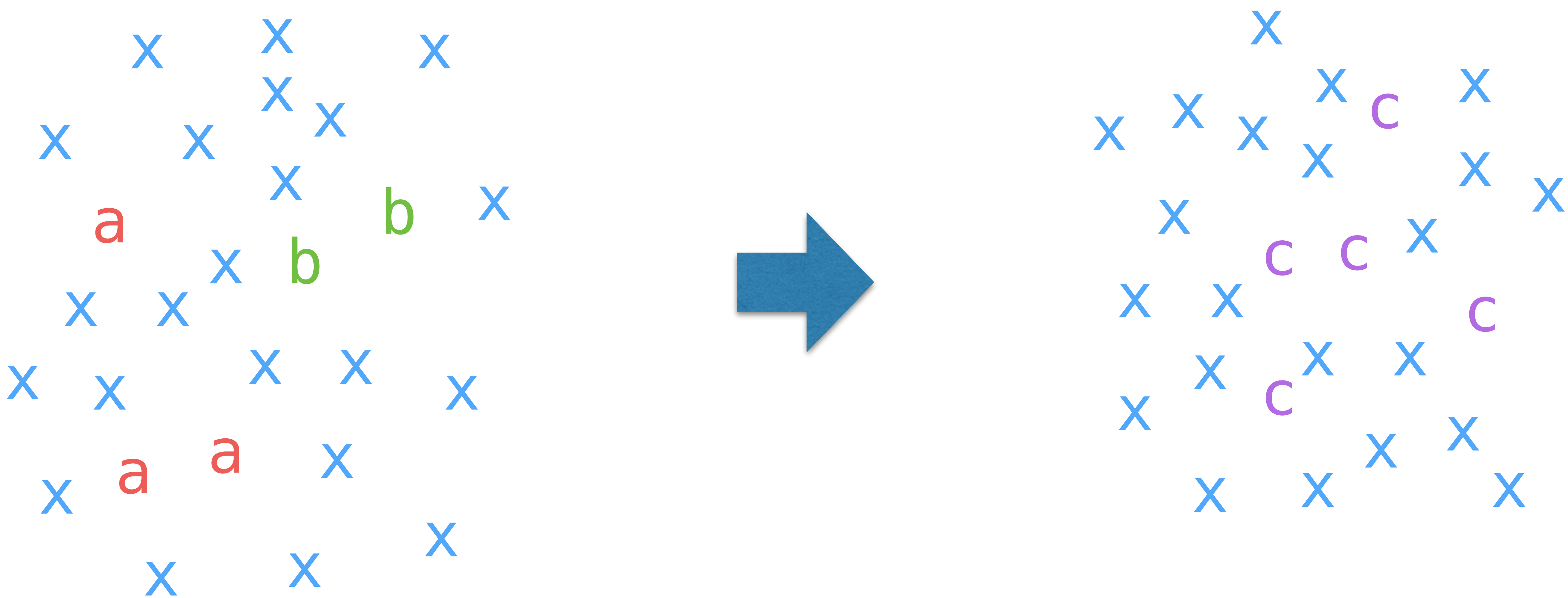
3 a

2 b

5 c

Example: Addition

$$c = a + b$$



3 a

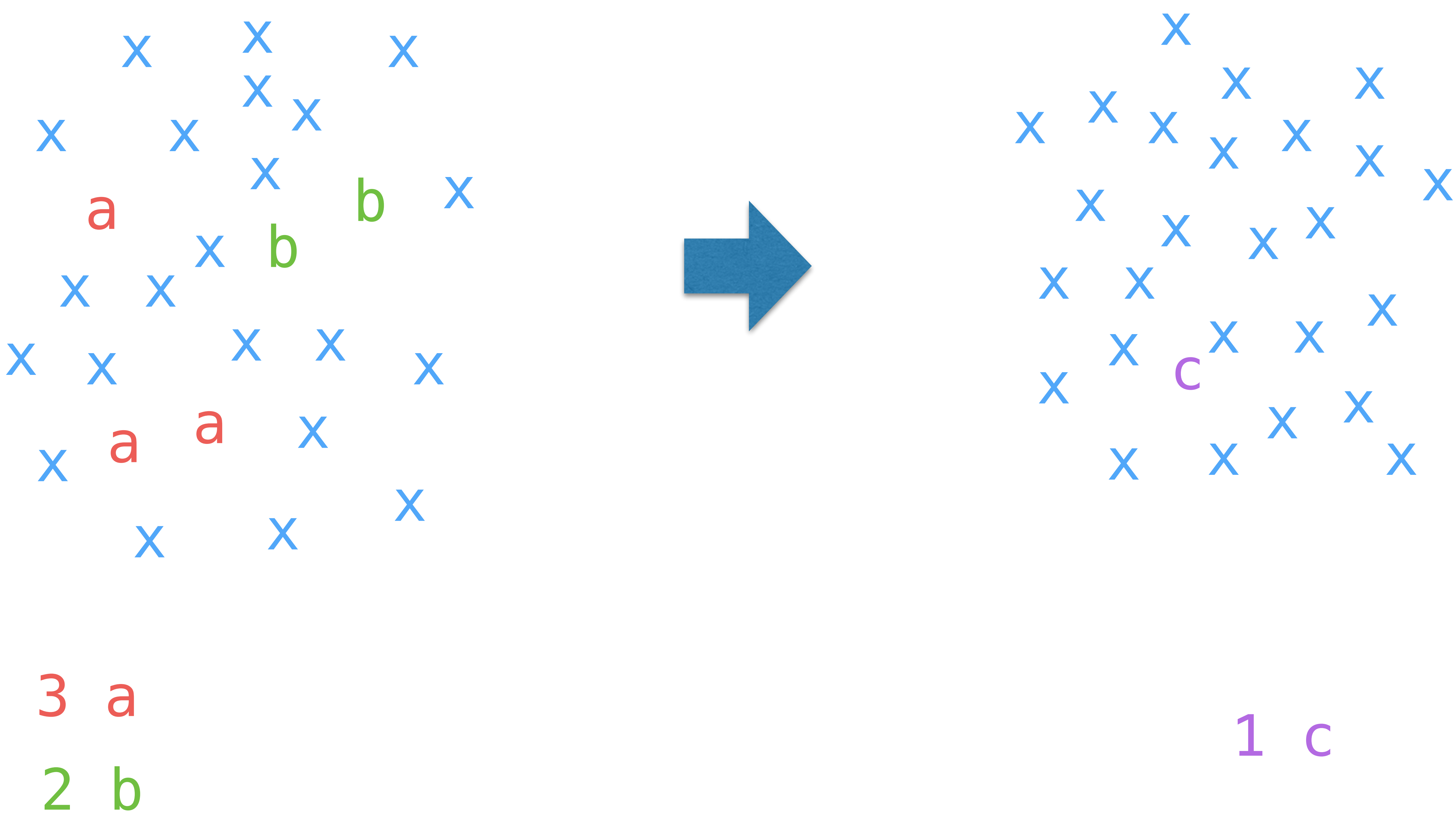
2 b

```
a,x → c,x
b,x → c,x
```

5 c

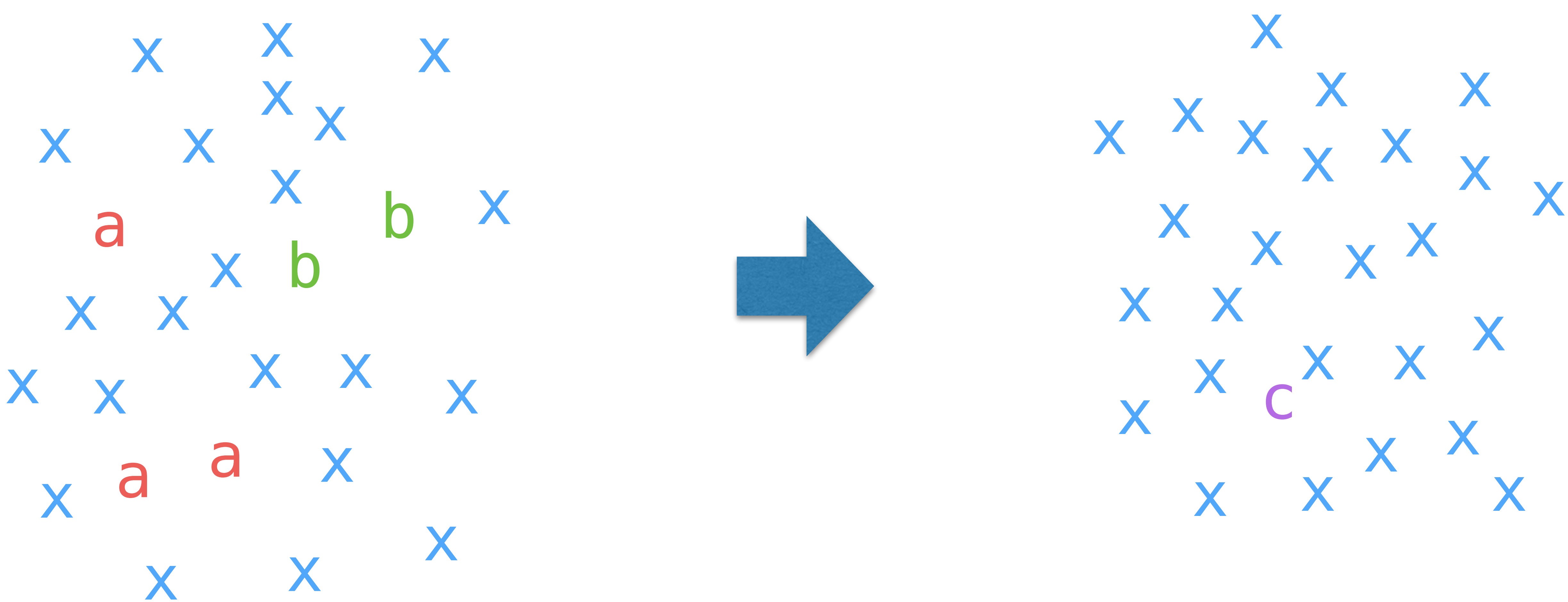
Example: Subtraction

$c=a-b$ (assume $a \geq b$)



Example: Subtraction

$c = a - b$ (assume $a \geq b$)



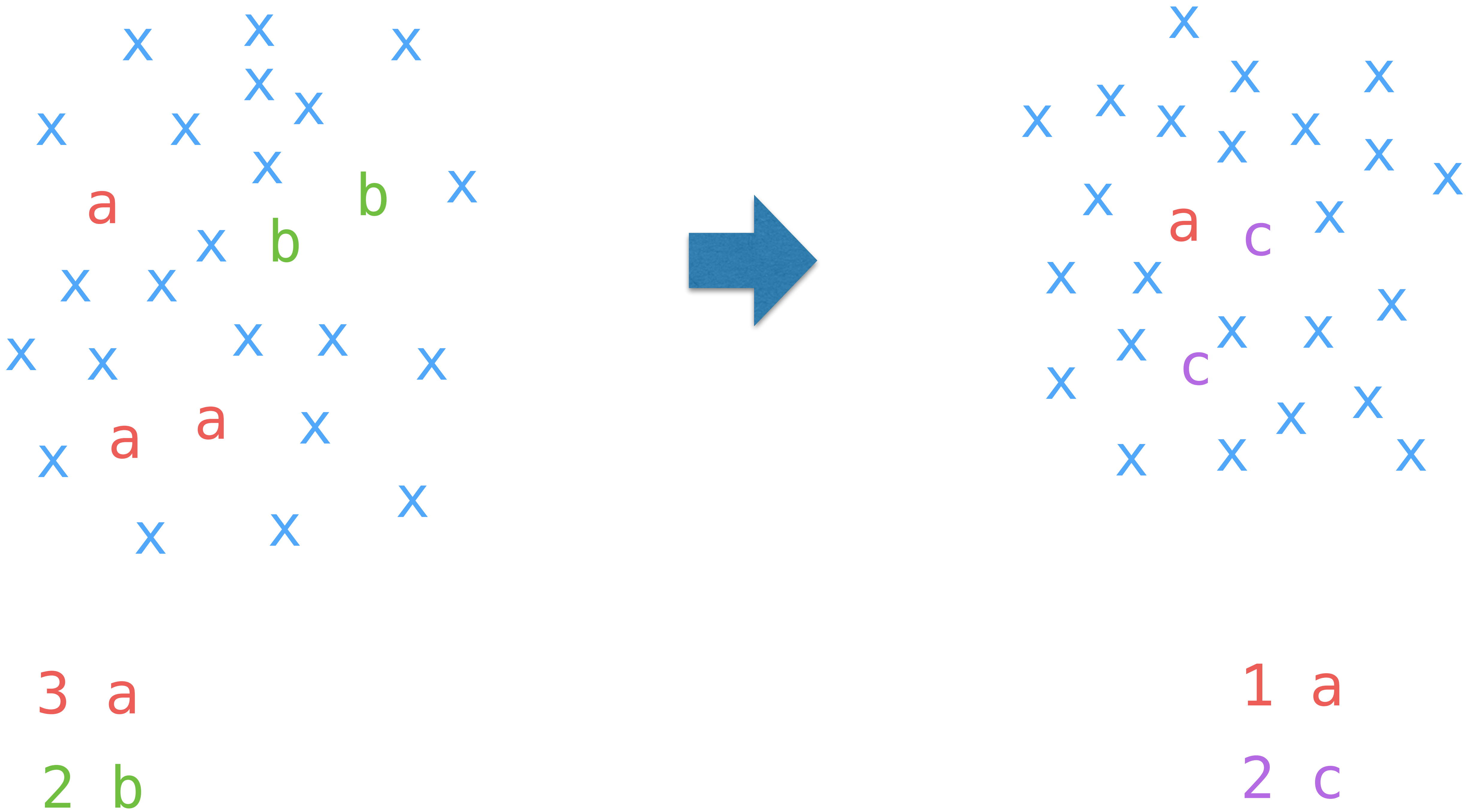
a, x → c, x
c, b → x, x

3 a
2 b

1 c

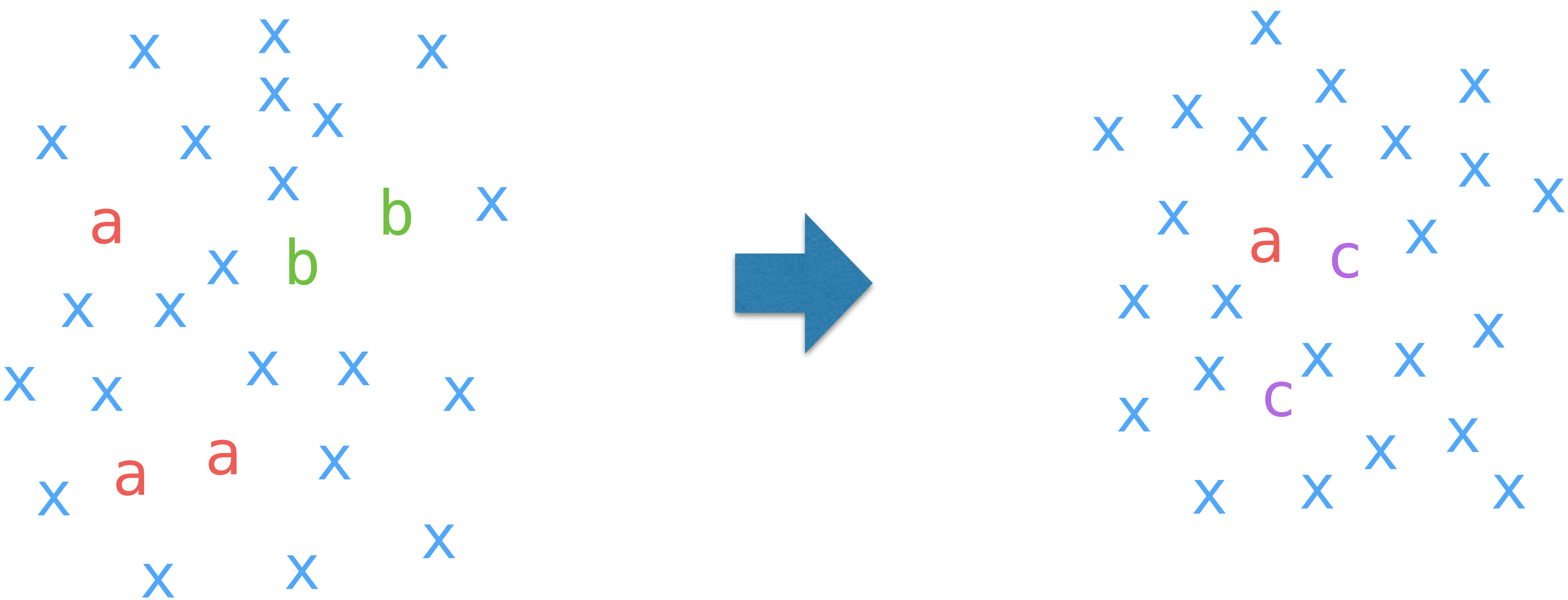
Example: Minimum

$$c = \min(a, b)$$



Example: Minimum

$$c = \min(a, b)$$



3 a

2 b

$\boxed{a, b \rightarrow c, x}$

1 a

2 c

Example: Maximum

$c = \max(a, b)$

Example: Maximum

$$c = \max(a, b)$$

Note: $\max(a,b) = a + b - \min(a,b)$

Example: Maximum

$c = \max(a,b)$

Note: $\max(a,b) = a + b - \min(a,b)$

$c = a+b$

```
a,x → c,x
b,x → c,x
```

$c = \min(a-b)$

```
a,b → c,x
```

$c = a-b$

```
a,x → c,x
c,b → x,x
```

Example: Maximum

c=max(a,b)

Note: max(a,b) = a + b - min(a,b)

c=a+b

```
a,x → c,x
b,x → c,x
```

c=min(a-b)

```
a,b → c,x
```

c=a-b

```
a,x → c,x
c,b → x,x
```

```
a,x → c,a'
b,x → c,b'
a',b' → k,x
c,k → x,x
```

Output Non-Monotonicity Makes Composition Tricky

d=max(a,b)

```
a,x  →  d,a'
b,x  →  d,b'
a',b'  →  k,x
d,k  →  x,x
```

e=min(d,c)

```
d,c  →  e,x
```

How do you compose these to compute

e=min(max(a,b),c) ?

$$e = \min(\max(a, b), c)$$

```
a,x  →  d,a'
b,x  →  d,b'
a',b'  →  k,x
d,k  →  x,x

d,c  →  e,x
```

$$e = \min(\max(a, b), c)$$

```
a,x → d,a'
b,x → d,b'
a',b' → k,x
d,k → x,x

d,c → e,x
```

incorrect

$$e = \min(\max(a, b), c)$$

$a, x \rightarrow d, a'$
$b, x \rightarrow d, b'$
$a', b' \rightarrow k, x$
$d, k \rightarrow x, x$

$d, c \rightarrow e, x$

incorrect

$a, x \rightarrow d, a'$
$b, x \rightarrow d, b'$
$a', b' \rightarrow k, x$

$d, c \rightarrow e, x$

$k, x \rightarrow k', c$
$k', e \rightarrow x, x$

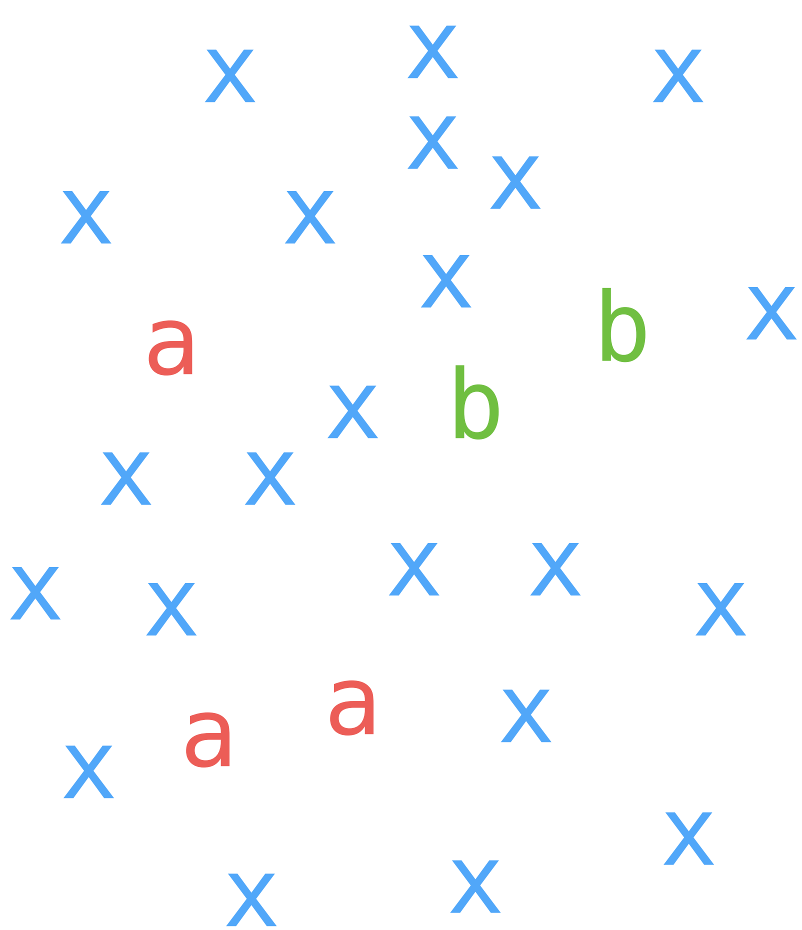
correct

Example: Testing Equality

y means *a*=*b*
n means *a*≠*b*

(assume *a*,*b*>*0*)

Output goal: get to a stable configuration where there are agents in state *y* or state *n* but not both.



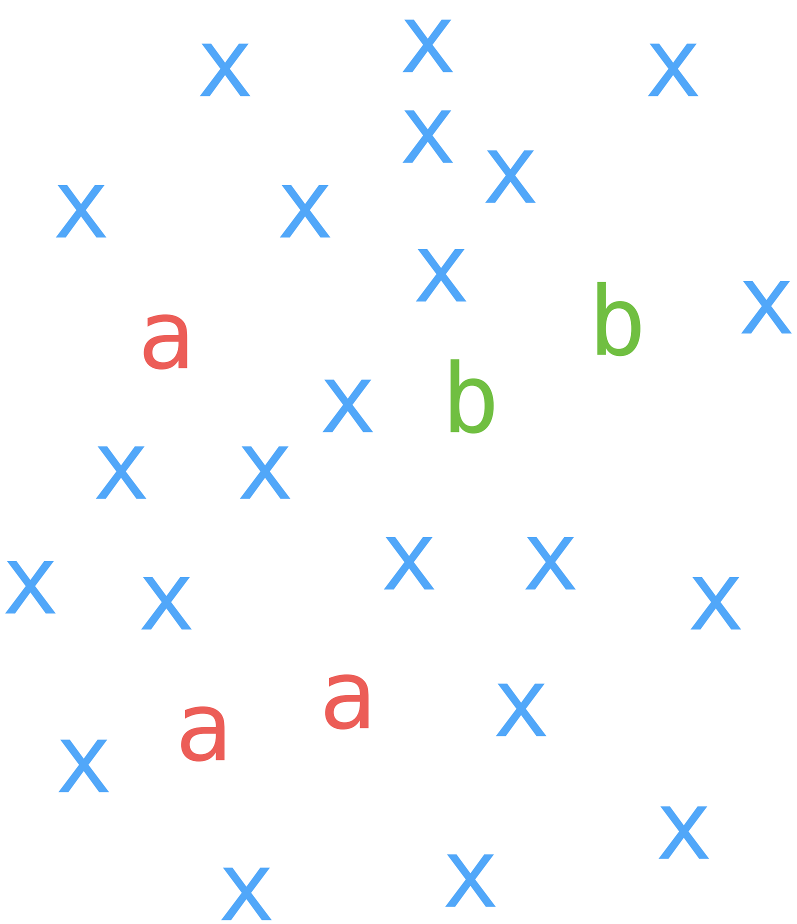
Example: Testing Equality

y means $a=b$

n means $a \neq b$

(assume $a,b>0$)

Output goal: get to a stable configuration where there are agents in state *y* or state *n* but not both.



$$a,b \rightarrow y,x$$
$$y,n \rightarrow y,x$$
$$a,y \rightarrow a,n$$
$$b,y \rightarrow b,n$$

Defn: Output Stable Configurations

For a configuration $\mathbf{x}$, let $\Psi(\mathbf{x})$ be in the input value and $\Phi(\mathbf{x})$ is the output value.
$\Psi(\mathbf{x})$ or $\Phi(\mathbf{x})$ may be undefined ($\perp$).

We say configuration $\mathbf{x}$ is <u>output-stable</u> if for all configurations $\mathbf{y}$ reachable from $\mathbf{x}$, output value $\Phi(\mathbf{y}) = \Phi(\mathbf{x})$.

Defn: Stable Computation

We say the population protocol <u>stably computes</u> the function or predicate f if:

For every configuration $\mathbf{x}$ with input value $\Psi(\mathbf{x}) \neq \perp$, for every configuration $\mathbf{w}$ reachable from $\mathbf{x}$, there is an output-stable configuration $\mathbf{y}$ reachable from $\mathbf{w}$ with output value $\Phi(\mathbf{y}) = f(\Psi(\mathbf{x}))$.

[Angluin, Aspnes, Eisenstat, Ruppert 2007]

Outline

- Population protocols model
- Examples of "deterministic" computation
- Formally defining "deterministic computation": stable computation
- Time model and computational complexity
- Consensus / approximate majority algorithm
- Biological connections
- Programming molecular interactions

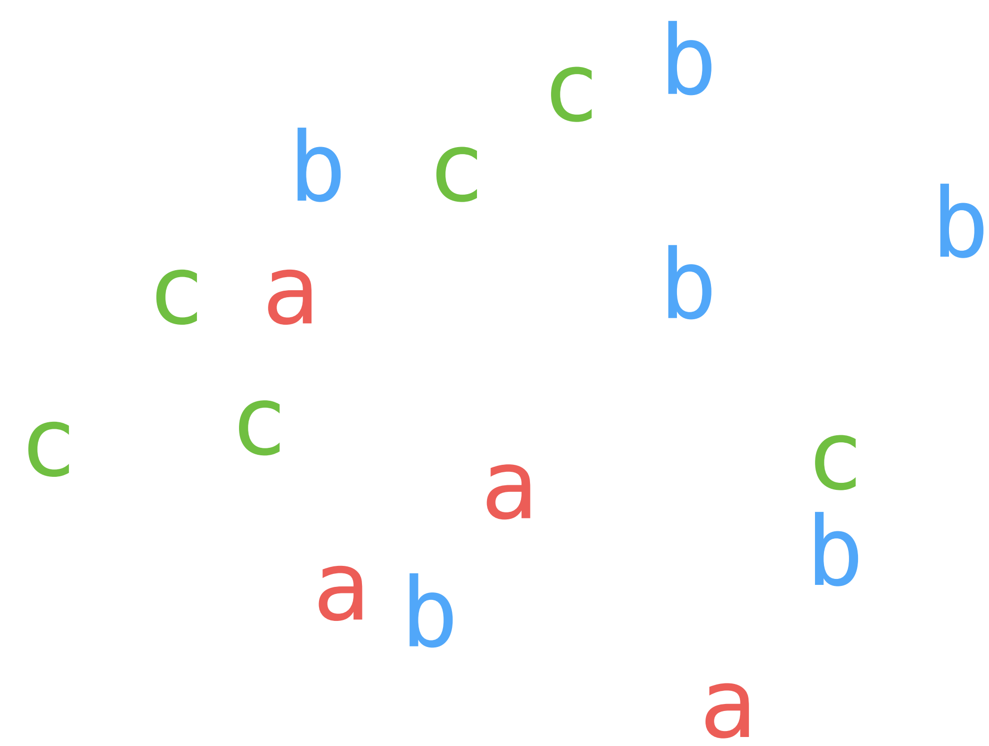
"Well-Mixed" Stochastic Model

any two agents equally likely to interact next

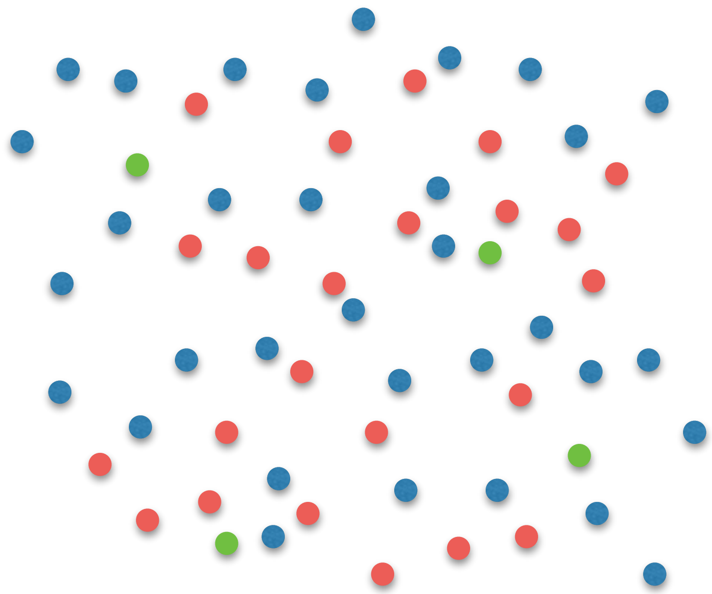
a,a → c,c

a,c → a,a

b,c → b,b



Measuring Time Complexity of Population Protocols



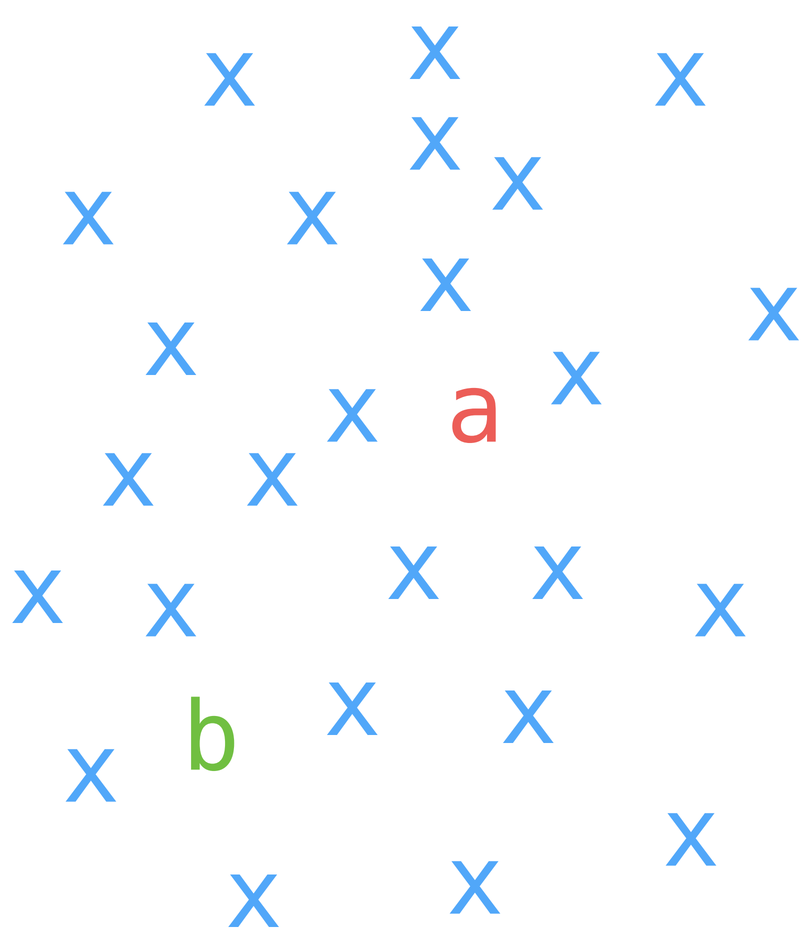
n = total number of agents

Natural parallel model: each agent interacts with $\Theta(1)$ other agents in one unit of time

Thus there are $\Theta(n)$ total interactions per unit of time

Expected Time for "Direct Communication"

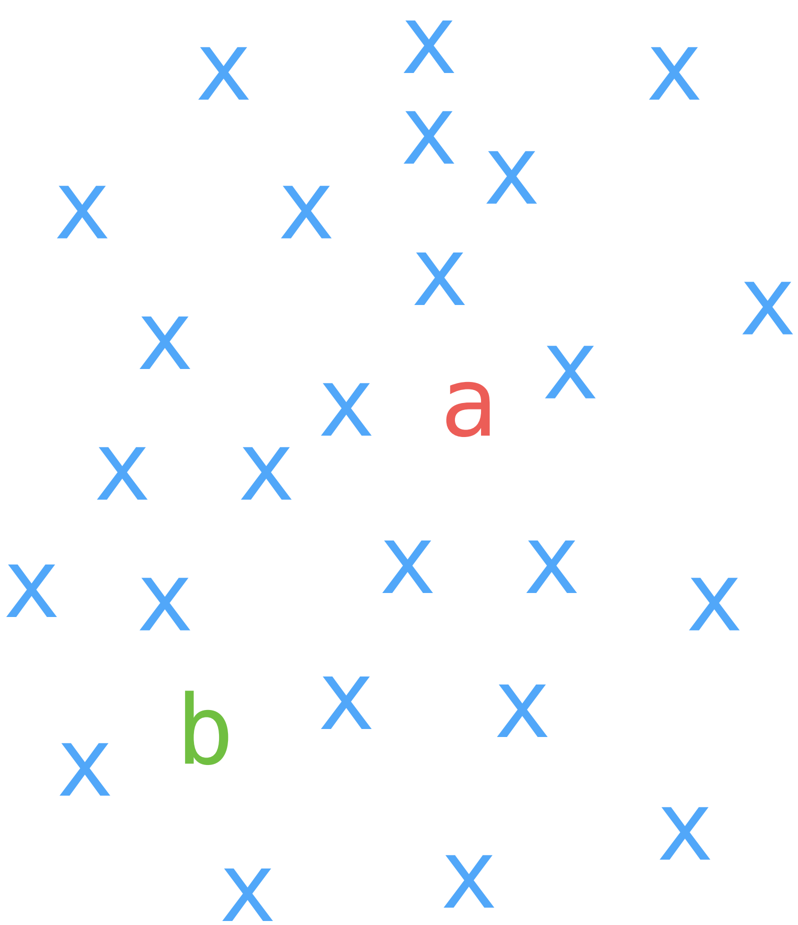
n agents total



$a, b \rightarrow a, y$

Expected Time for "Direct Communication"

n agents total

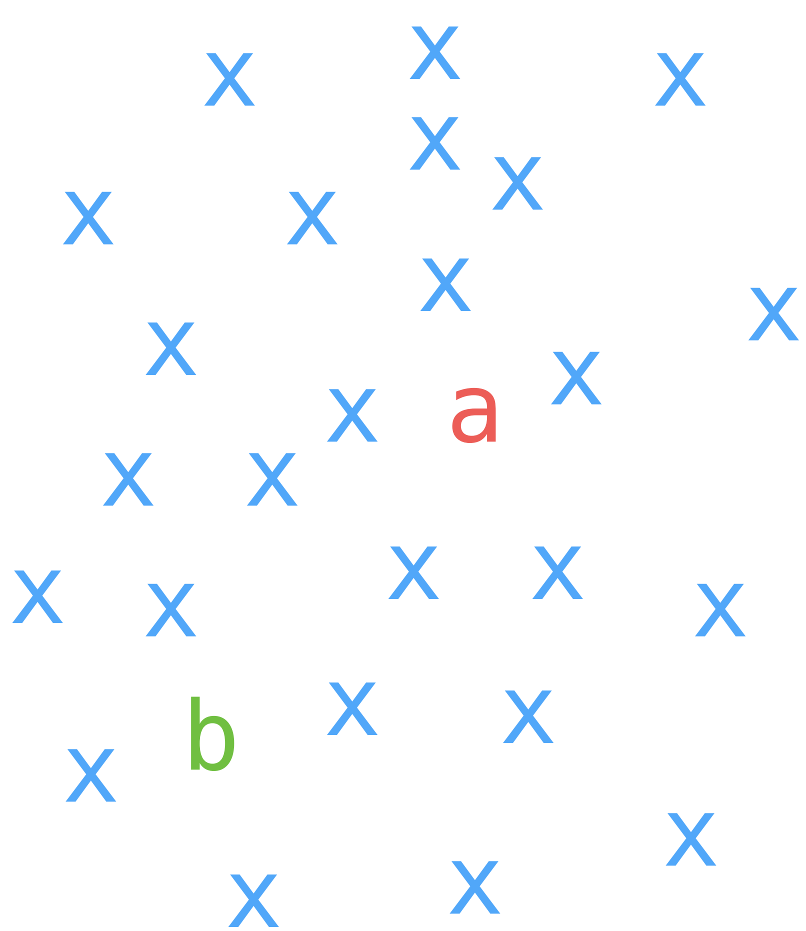


`a,b → a,y`

probability that the next interaction involves a and b:

Expected Time for "Direct Communication"

n agents total



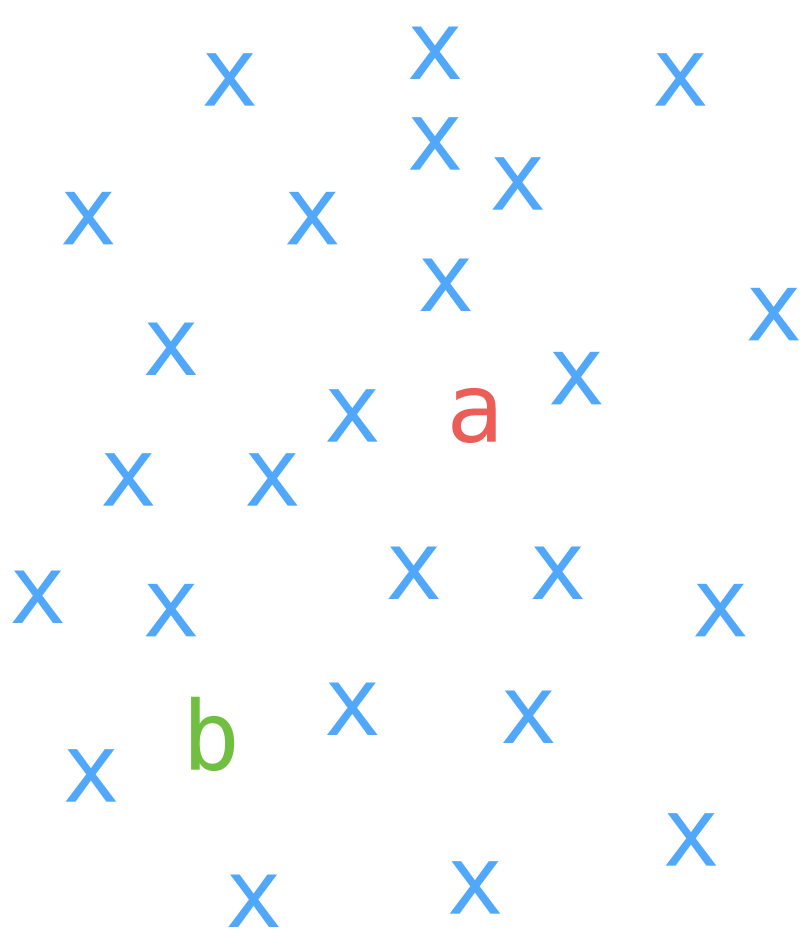
probability that the next interaction involves a and b:

$$2/n^2$$

$$\boxed{a, b \rightarrow a, y}$$

Expected Time for "Direct Communication"

n agents total



$\boxed{a, b \rightarrow a, y}$

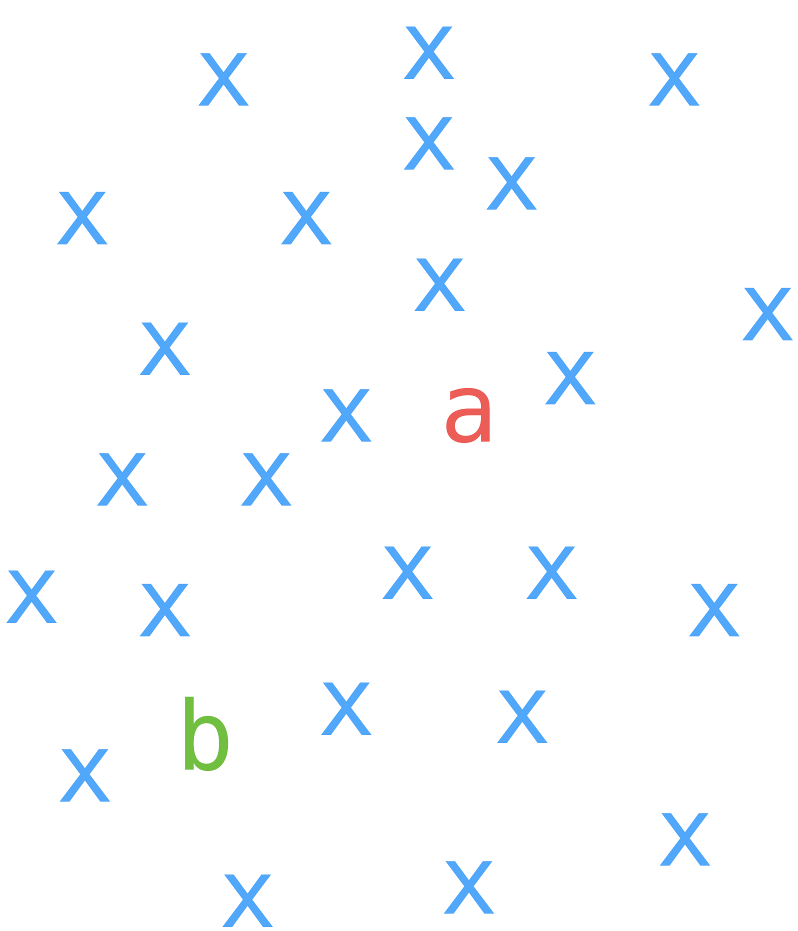
probability that the next interaction involves a and b:

$2/n^2$

expected number of interactions until a and b interact:

Expected Time for "Direct Communication"

n agents total



$\boxed{a, b \rightarrow a, y}$

probability that the next interaction involves a and b:

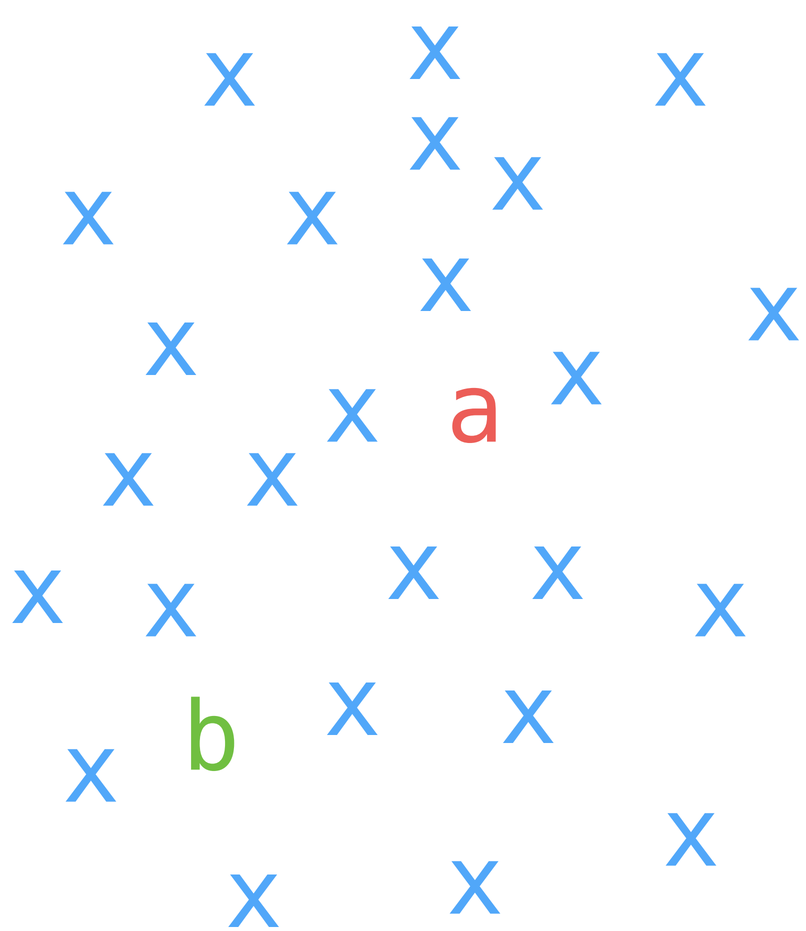
$2/n^2$

expected number of interactions until a and b interact:

$n^2/2$

Expected Time for "Direct Communication"

n agents total



`a,b → a,y`

probability that the next interaction involves a and b:

$2/n^2$

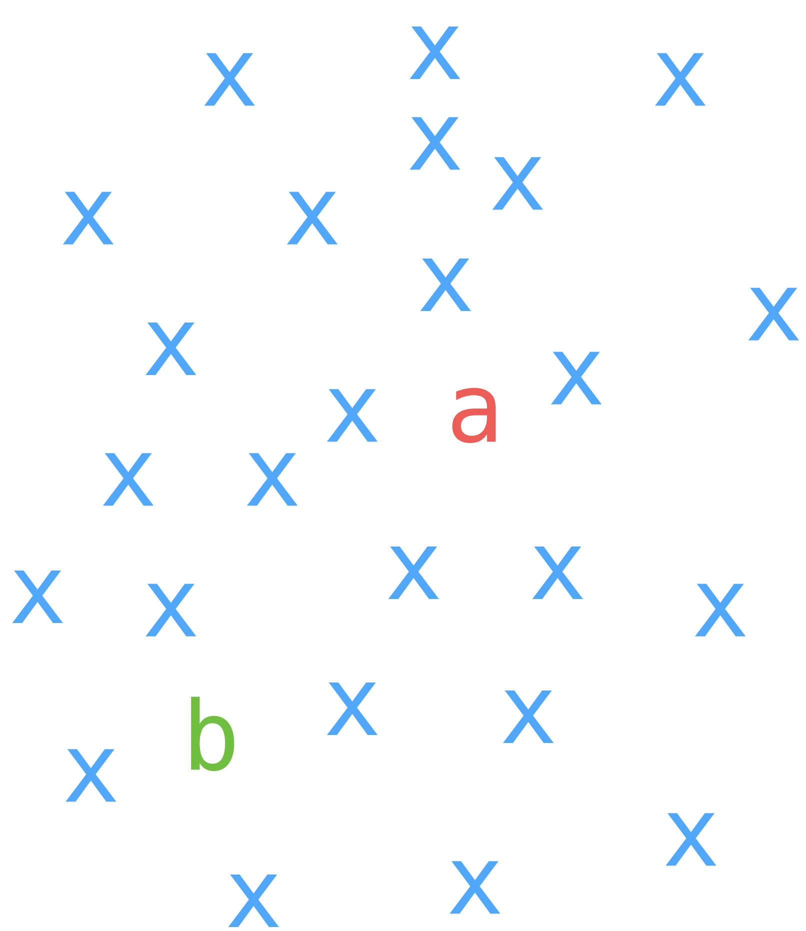
expected number of interactions until a and b interact:

$n^2/2$

corresponding expected time:

Expected Time for "Direct Communication"

n agents total



$$\boxed{a, b \rightarrow a, y}$$

probability that the next interaction involves a and b:

$2/n^2$

expected number of interactions until a and b interact:

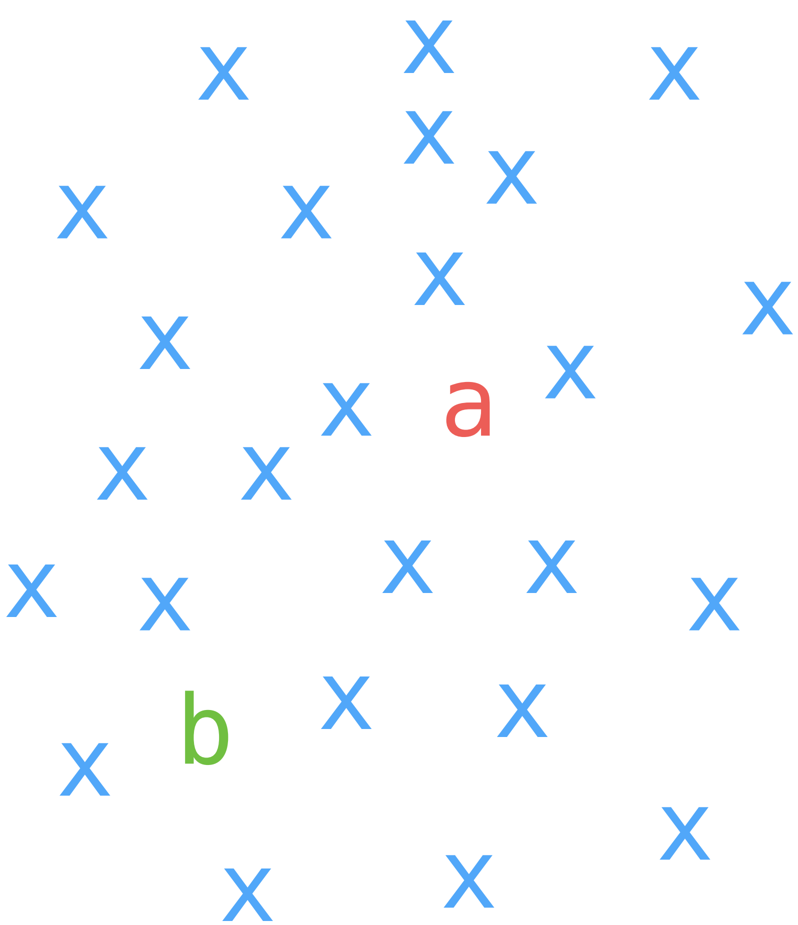
$n^2/2$

corresponding expected time:

$\Theta(n)$

Expected Time for "Communication by Epidemic"

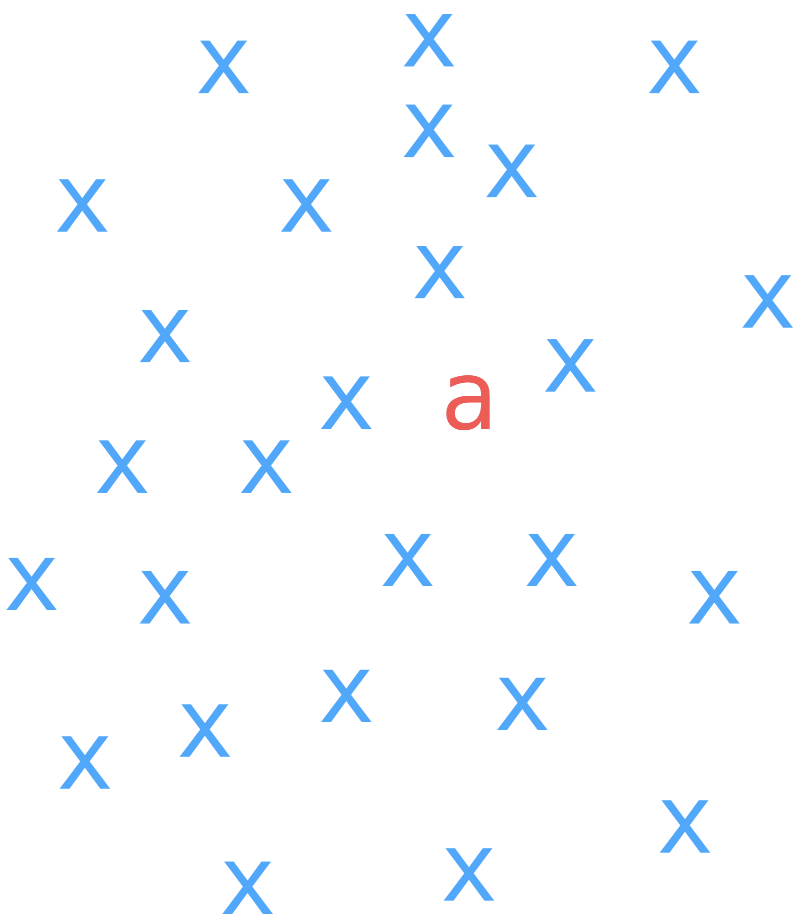
n agents total



$$a, x \rightarrow a, a$$
$$a, b \rightarrow a, y$$

Expected Time for "Communication by Epidemic"

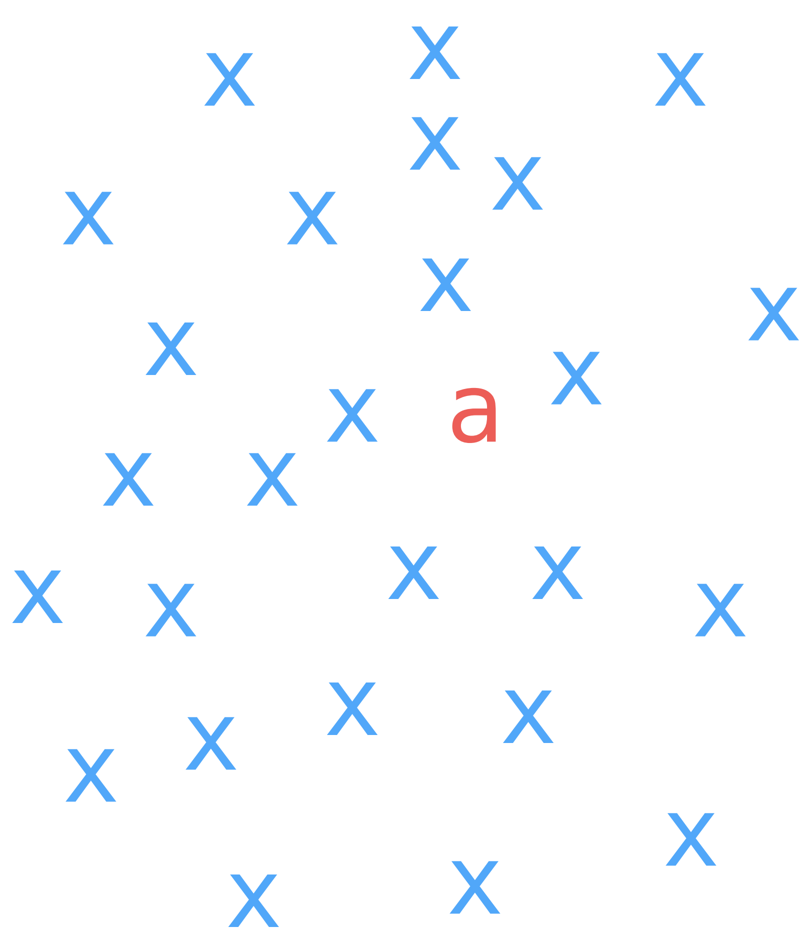
n agents total



$$\boxed{a, x \rightarrow a, a}$$

Expected Time for "Communication by Epidemic"

n agents total

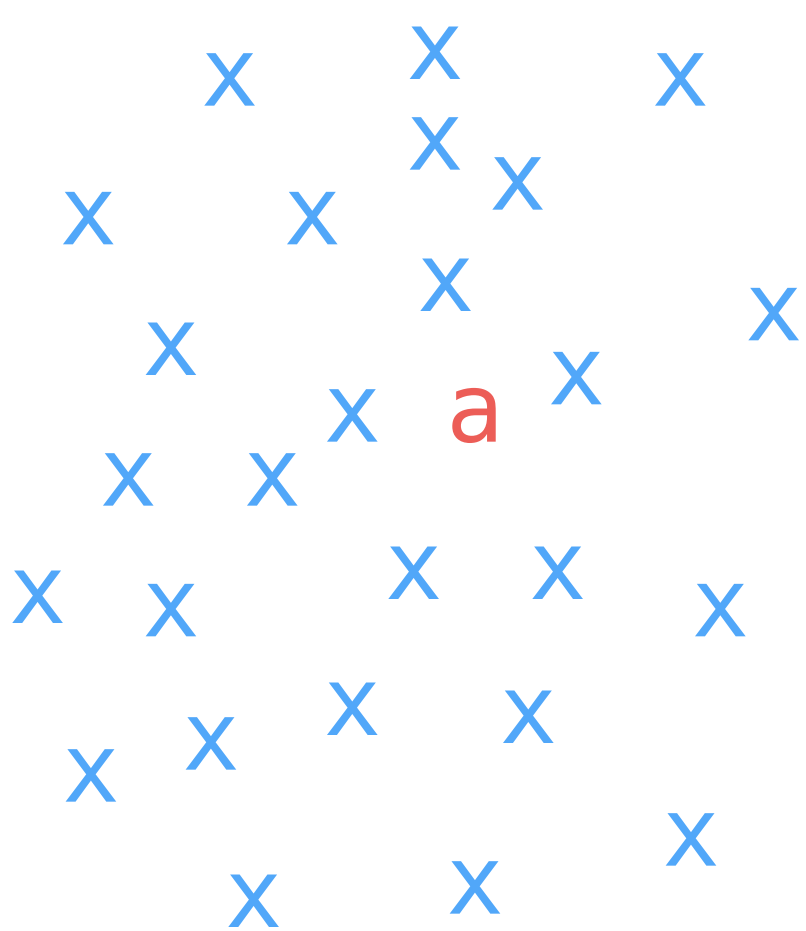


$\boxed{a, x \rightarrow a, a}$

probability that the next interaction involves a and x:

Expected Time for "Communication by Epidemic"

n agents total



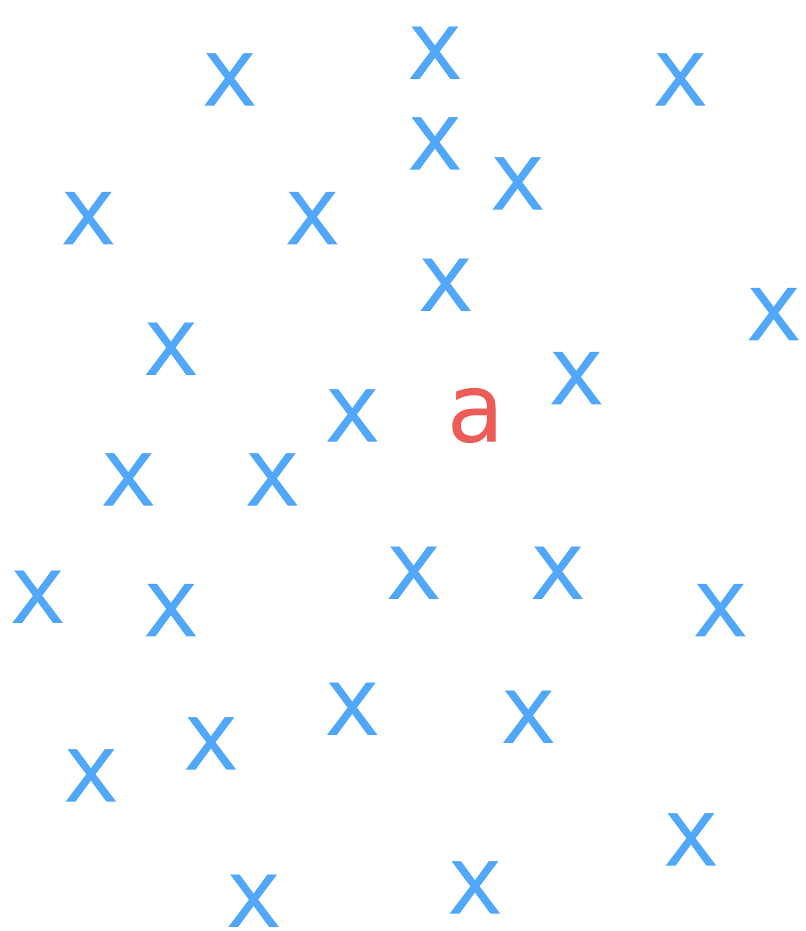
$$\boxed{a, x \rightarrow a, a}$$

probability that the next interaction involves a and x:

$$2ax/n^2 = 2a(n\text{-}a)/n^2$$

Expected Time for "Communication by Epidemic"

n agents total



$$\boxed{a, x \rightarrow a, a}$$

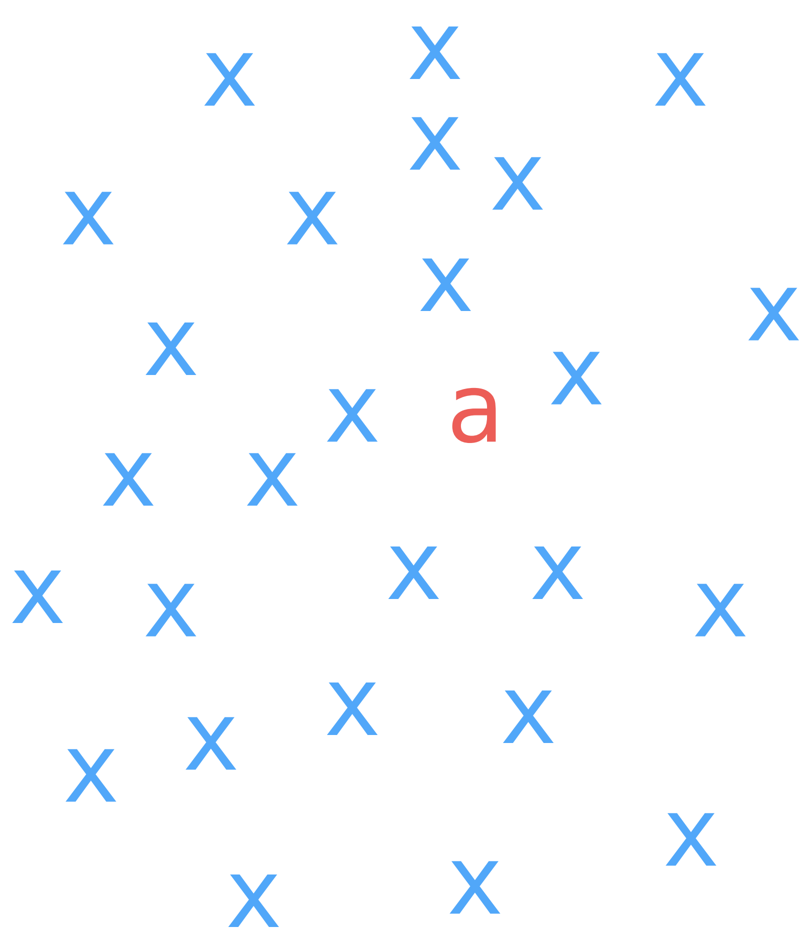
probability that the next interaction involves a and x:

$$2ax/n^2 = 2a(n-a)/n^2$$

expected number of interactions until *a* increases by 1:

Expected Time for "Communication by Epidemic"

n agents total



$$\boxed{a, x \rightarrow a, a}$$

probability that the next interaction involves a and x:

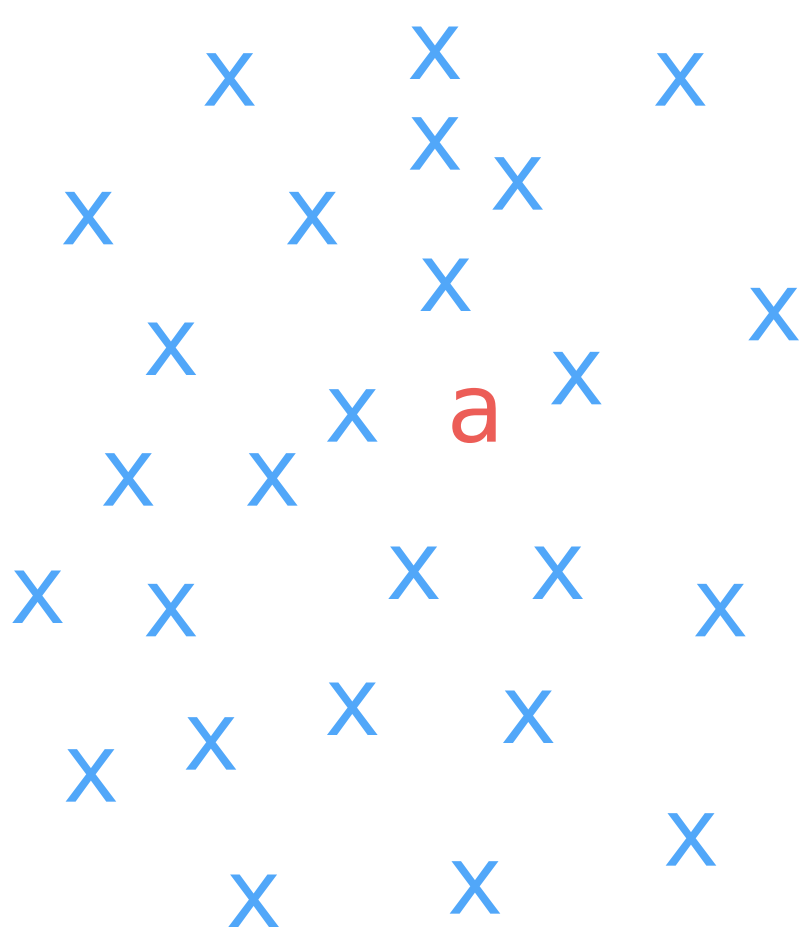
$$2ax/n^2 = 2a(n-a)/n^2$$

expected number of interactions until *a* increases by 1:

$$n^2/(2a(n\text{-}a))$$

Expected Time for "Communication by Epidemic"

n agents total



$\boxed{a, x \rightarrow a, a}$

probability that the next interaction involves a and x:

$$2ax/n^2 = 2a(n-a)/n^2$$

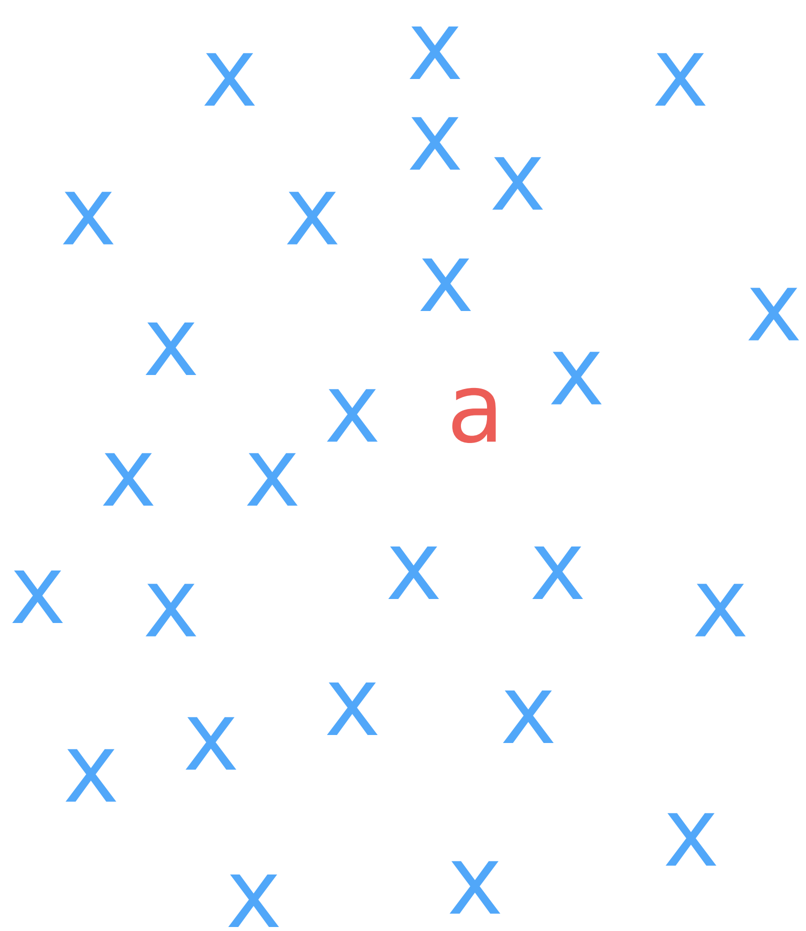
expected number of interactions until *a* increases by 1:

$$n^2/(2a(n\text{-}a))$$

expected number of interactions until all x become a:

Expected Time for "Communication by Epidemic"

n agents total



$$\boxed{a, x \rightarrow a, a}$$

probability that the next interaction involves a and x:

$$2ax/n^2 = 2a(n\text{-}a)/n^2$$

expected number of interactions until *a* increases by 1:

$$n^2/(2a(n\text{-}a))$$

expected number of interactions until all x become a:

$$\frac{n^2}{2}\sum_{a=1}^{n-1}\frac{1}{a(n-a)} = ?$$

Fact: $\sum_{a=1}^{n} \frac{1}{a} = O(\log n)$

Fact: $\sum_{a=1}^{n} \frac{1}{a} = O(\log n)$

$$\frac{n^2}{2} \sum_{a=1}^{n-1} \frac{1}{a(n-a)}$$

Fact: $\sum_{a=1}^{n} \frac{1}{a} = O(\log n)$

$$\frac{n^2}{2} \sum_{a=1}^{n-1} \frac{1}{a(n-a)} \quad = \frac{n^2}{2} \left(\sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)} + \sum_{a=\frac{n-1}{2}+1}^{n-1} \frac{1}{a(n-a)} \right)$$

Fact: $\displaystyle\sum_{a=1}^{n} \frac{1}{a} = O(\log n)$

$$\frac{n^2}{2} \sum_{a=1}^{n-1} \frac{1}{a(n-a)} = \frac{n^2}{2} \left(\sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)} + \sum_{a=\frac{n-1}{2}+1}^{n-1} \frac{1}{a(n-a)} \right)$$

$$= \frac{n^2}{2} \left(\sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)} + \sum_{x=\frac{n-1}{2}}^{1} \frac{1}{(n-x)x} \right)$$

change of variables: *x*=*n*-*a*

Fact: $\sum_{a=1}^{n} \frac{1}{a} = O(\log n)$

$$\frac{n^2}{2} \sum_{a=1}^{n-1} \frac{1}{a(n-a)} = \frac{n^2}{2} \left(\sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)} + \sum_{a=\frac{n-1}{2}+1}^{n-1} \frac{1}{a(n-a)} \right)$$

$$= \frac{n^2}{2} \left(\sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)} + \sum_{x=\frac{n-1}{2}}^{1} \frac{1}{(n-x)x} \right)$$

$$= n^2 \sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)}$$

change of variables: x=n-a

Fact: $\displaystyle\sum_{a=1}^{n} \frac{1}{a} = O(\log n)$

$$\begin{aligned}
\frac{n^2}{2}\sum_{a=1}^{n-1}\frac{1}{a(n-a)} &= \frac{n^2}{2}\left(\sum_{a=1}^{\frac{n-1}{2}}\frac{1}{a(n-a)} + \sum_{a=\frac{n-1}{2}+1}^{n-1}\frac{1}{a(n-a)}\right)\\
&= \frac{n^2}{2}\left(\sum_{a=1}^{\frac{n-1}{2}}\frac{1}{a(n-a)} + \sum_{x=\frac{n-1}{2}}^{1}\frac{1}{(n-x)x}\right)\\
&= n^2\sum_{a=1}^{\frac{n-1}{2}}\frac{1}{a(n-a)}\\
&< n^2\sum_{a=1}^{\frac{n-1}{2}}\frac{1}{a(n/2)} = 2n\sum_{a=1}^{\frac{n-1}{2}}\frac{1}{a}
\end{aligned}$$

change of variables: x=n-a

Fact: $\sum_{a=1}^{n} \frac{1}{a} = O(\log n)$

$$\frac{n^2}{2} \sum_{a=1}^{n-1} \frac{1}{a(n-a)} = \frac{n^2}{2} \left(\sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)} + \sum_{a=\frac{n-1}{2}+1}^{n-1} \frac{1}{a(n-a)} \right)$$

$$= \frac{n^2}{2} \left(\sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)} + \sum_{x=\frac{n-1}{2}}^{1} \frac{1}{(n-x)x} \right)$$

change of variables: x=n-a

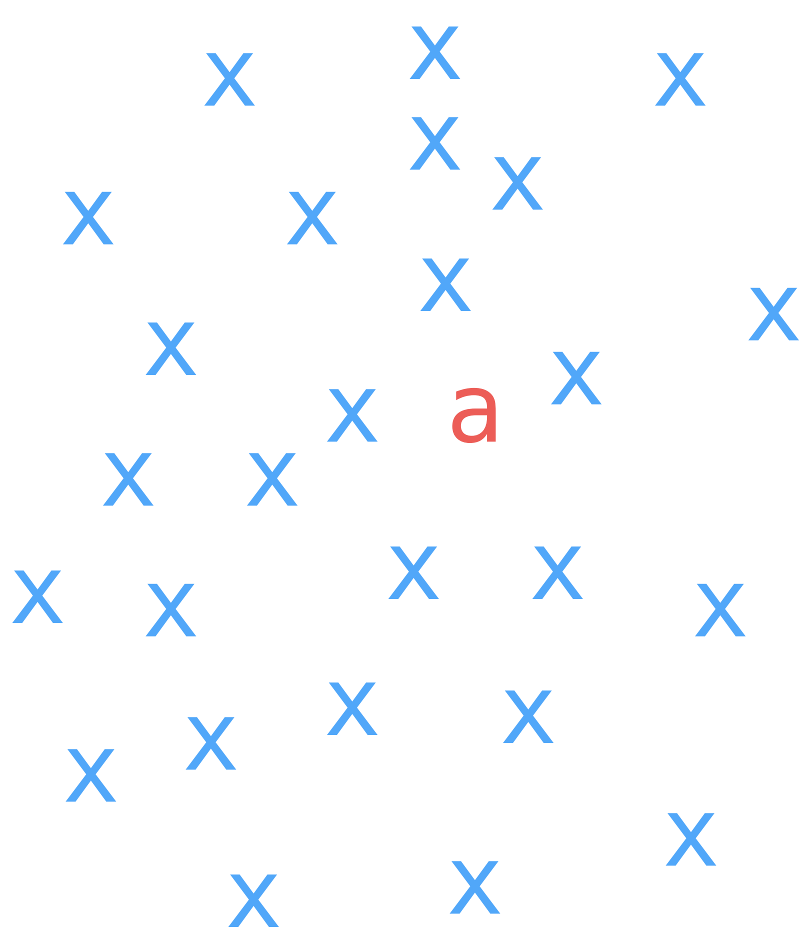
$$= n^2 \sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n-a)}$$

$$< n^2 \sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a(n/2)} = 2n \sum_{a=1}^{\frac{n-1}{2}} \frac{1}{a}$$

$$= O(n \log n)$$

Expected Time for "Communication by Epidemic"

n agents total



$$\boxed{a, x \rightarrow a, a}$$

probability that the next interaction involves a and x:

$$2ax/n^2 = 2a(n\text{-}a)/n^2$$

expected number of interactions until *a* increases by 1:

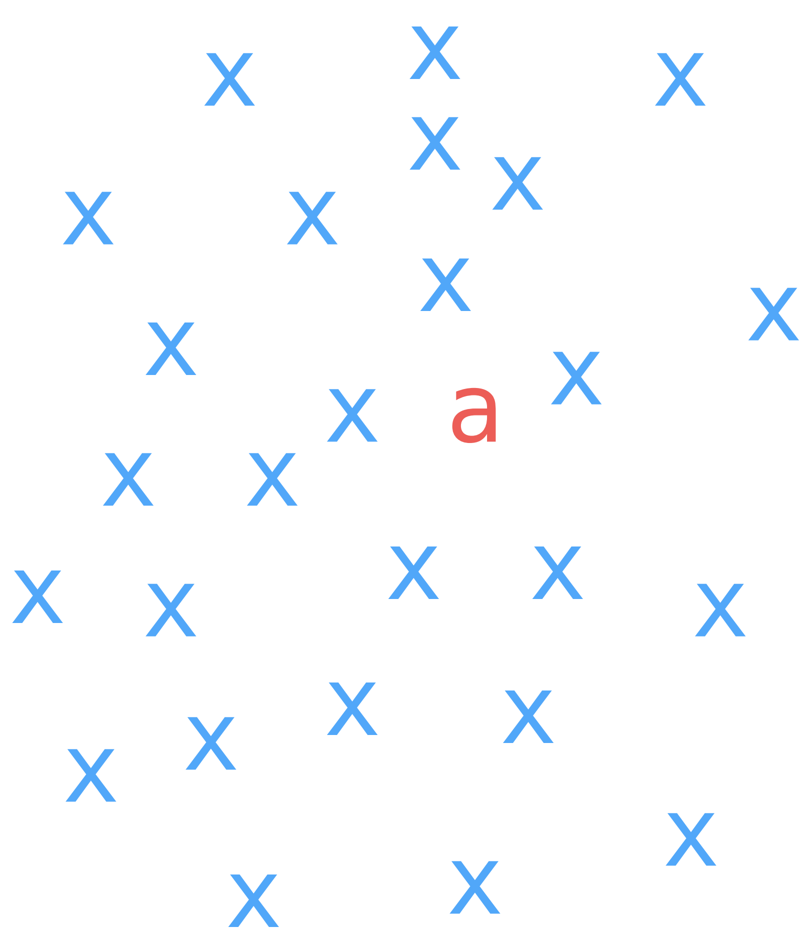
$$n^2/(2a(n\text{-}a))$$

expected number of interactions until all x become a:

$$\frac{n^2}{2}\sum_{a=1}^{n-1}\frac{1}{a(n-a)} = O(n \log n)$$

Expected Time for "Communication by Epidemic"

n agents total



$$\boxed{a, x \rightarrow a, a}$$

probability that the next interaction involves a and x:

$$2ax/n^2 = 2a(n\text{-}a)/n^2$$

expected number of interactions until *a* increases by 1:

$$n^2/(2a(n\text{-}a))$$

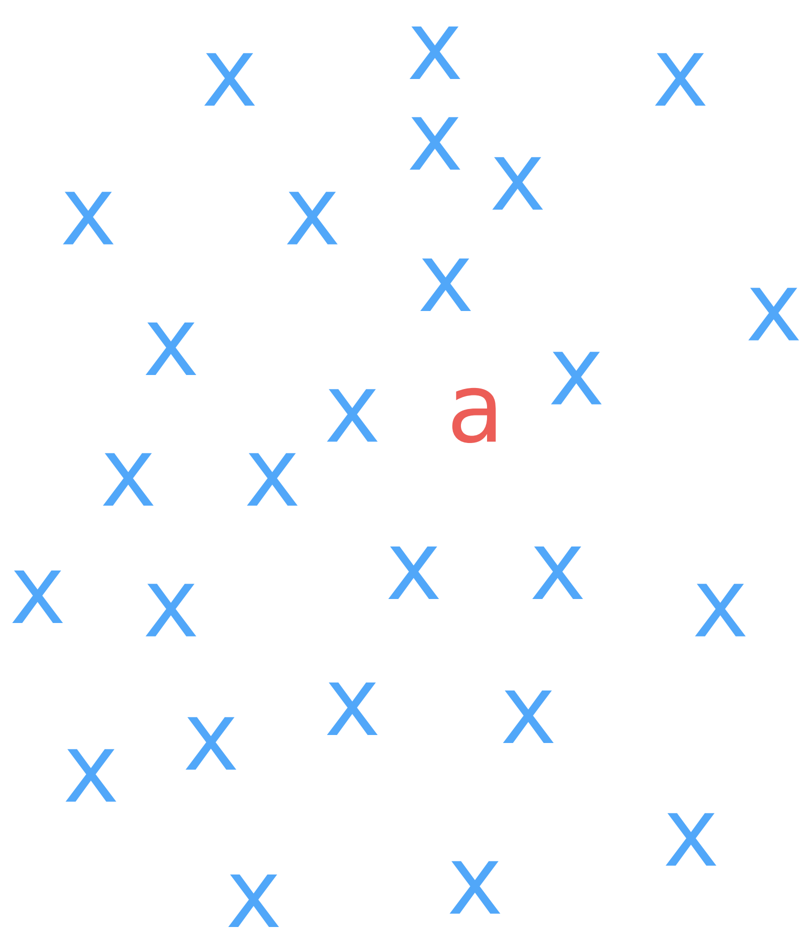
expected number of interactions until all x become a:

$$\frac{n^2}{2}\sum_{a=1}^{n-1}\frac{1}{a(n-a)} = O(n \log n)$$

corresponding expected time:

Expected Time for "Communication by Epidemic"

n agents total



$$a, x \rightarrow a, a$$

probability that the next interaction involves a and x:

$$2ax/n^2 = 2a(n\text{-}a)/n^2$$

expected number of interactions until *a* increases by 1:

$$n^2/(2a(n\text{-}a))$$

expected number of interactions until all x become a:

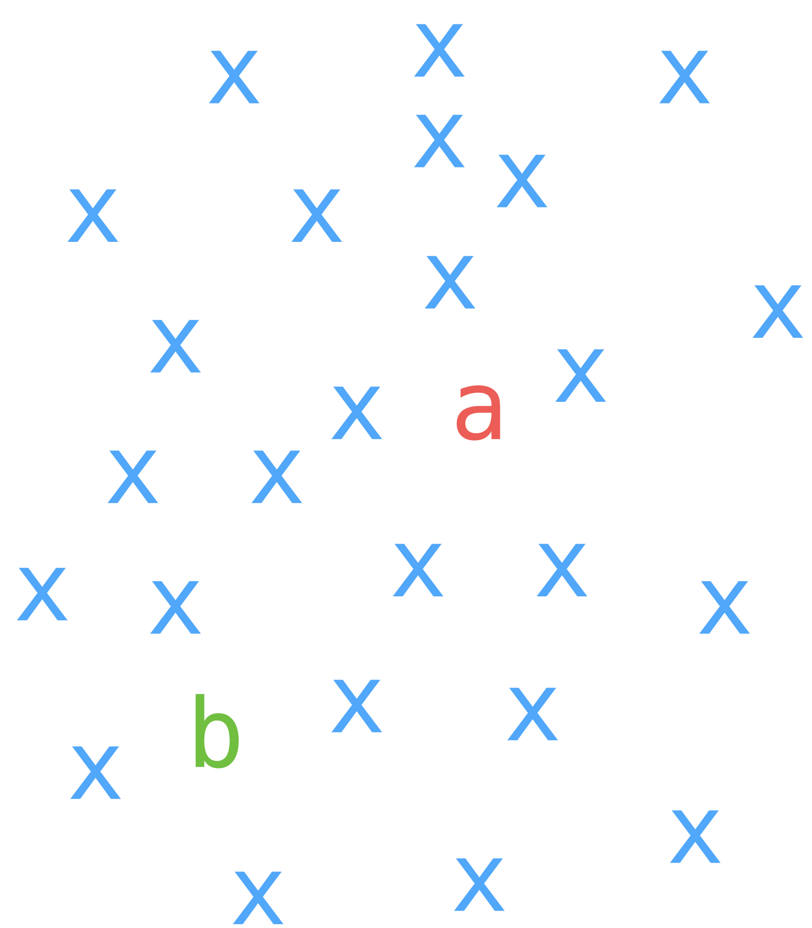
$$\frac{n^2}{2}\sum_{a=1}^{n-1}\frac{1}{a(n-a)} = O(n \log n)$$

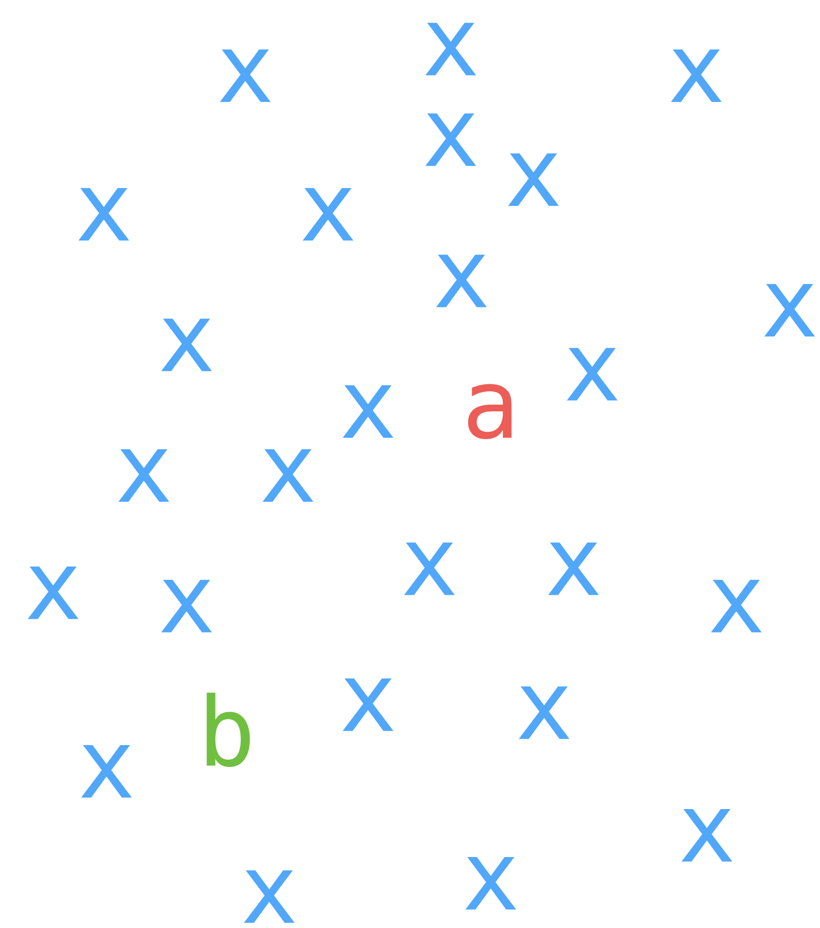
corresponding expected time:

$O(\log n)$

Exponential Difference Between Direct vs Epidemic

n agents



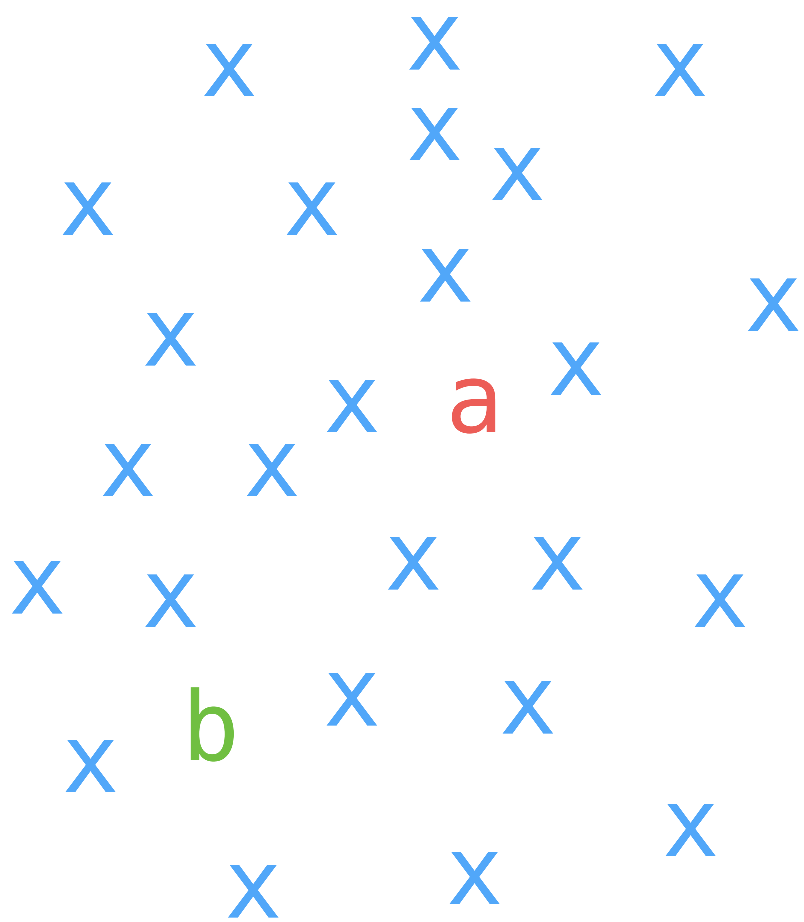


$$\boxed{a, b \rightarrow a, y}$$

$$\boxed{\begin{array}{l} a, x \rightarrow a, a \\ a, b \rightarrow a, y \end{array}}$$

expected time: $\Theta(n)$ $\Theta(\log n)$

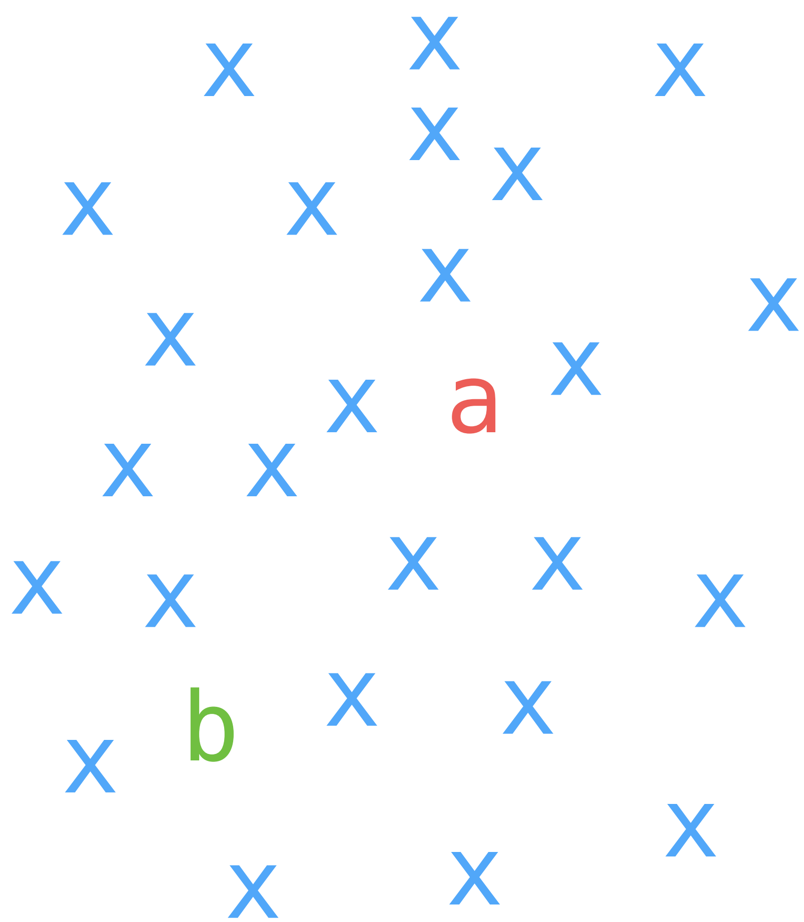
Produce y iff at least 1a and 1b



$$a, x \rightarrow a, a$$
$$a, b \rightarrow a, y$$

$\Theta(\log n)$
expected time

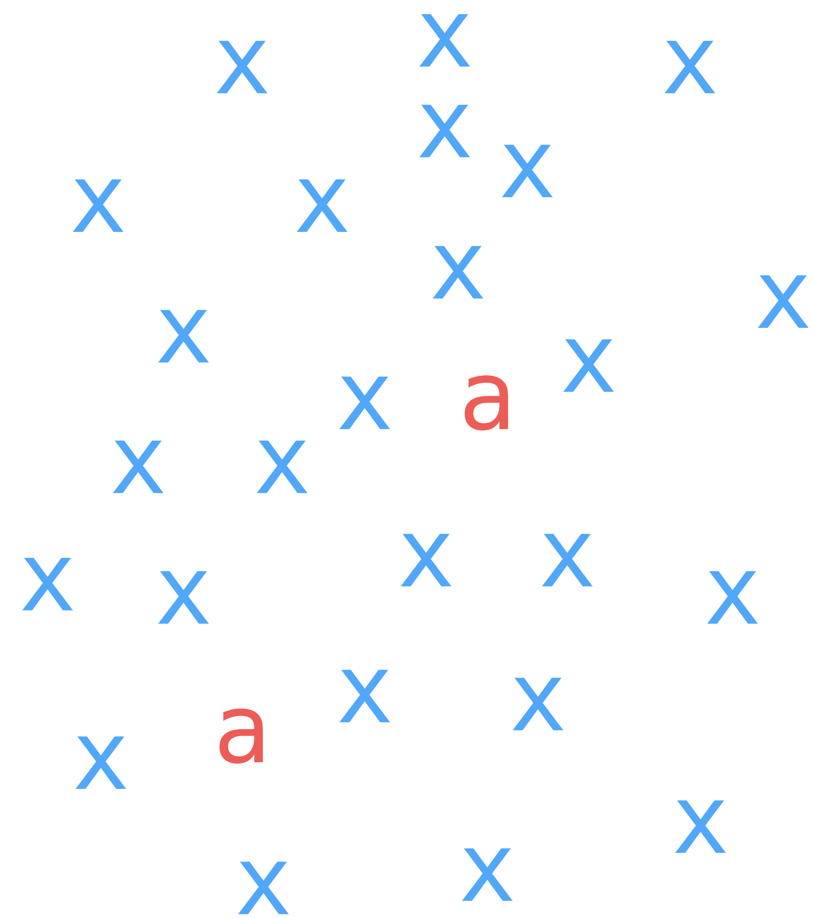
Produce y iff at least 1a and 1b



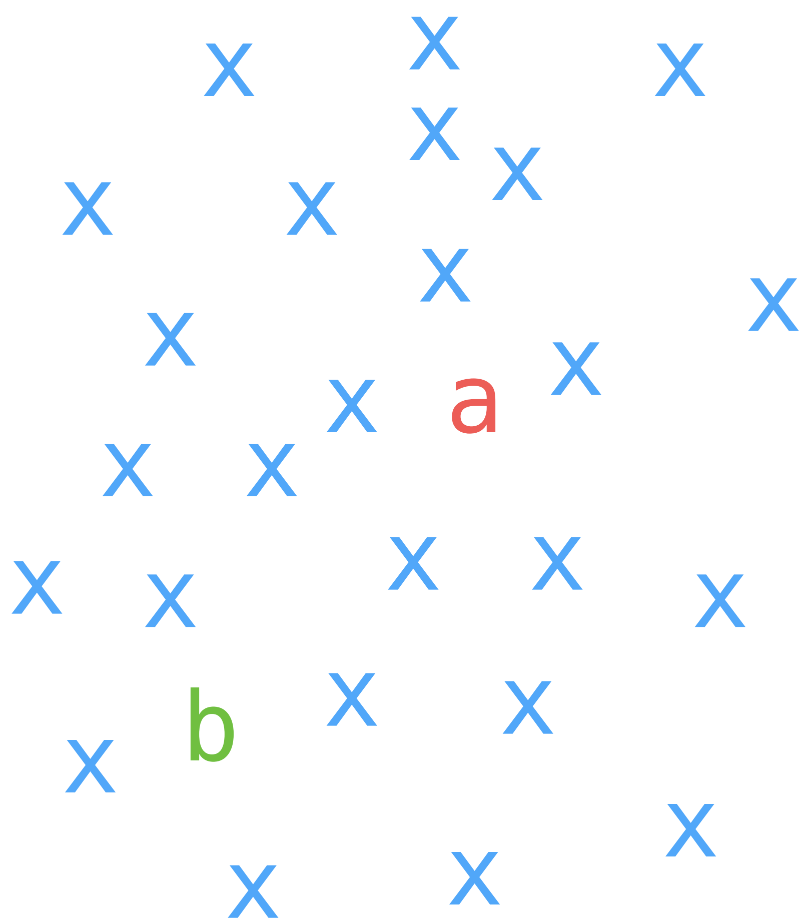
$$a, x \rightarrow a, a$$
$$a, b \rightarrow a, y$$

$\Theta(\log n)$
expected time

Produce y iff at least 2a



Produce y iff at least 1a and 1b

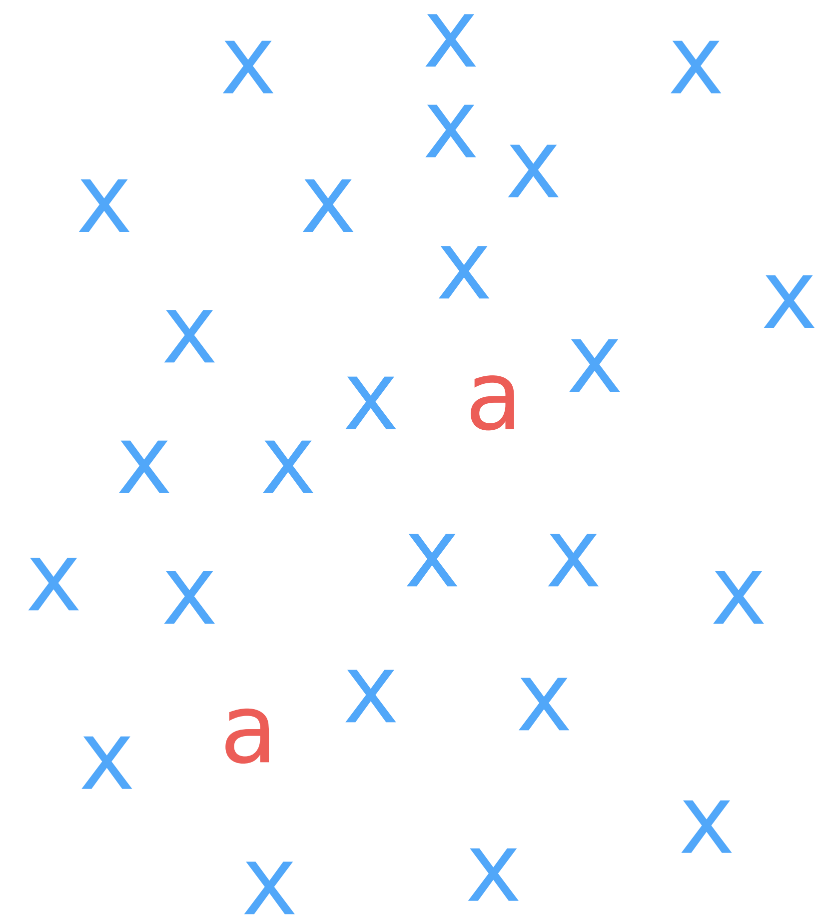


$a, x \rightarrow a, a$

$a, b \rightarrow a, y$

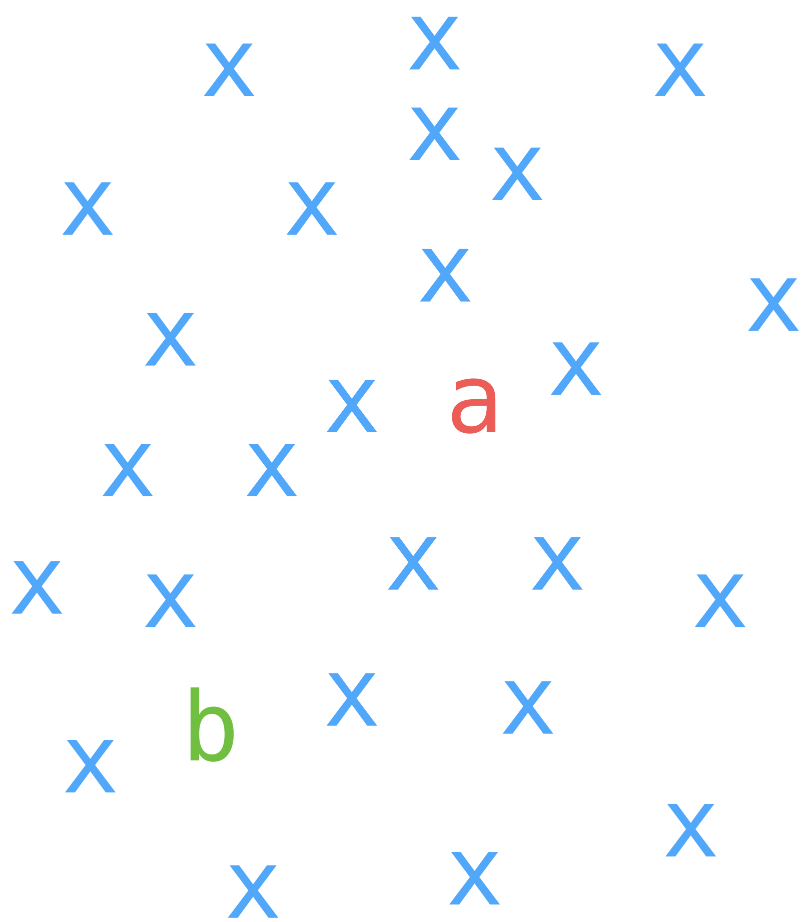
$\Theta(\log n)$ expected time

Produce y iff at least 2a



$a, a \rightarrow a, y$

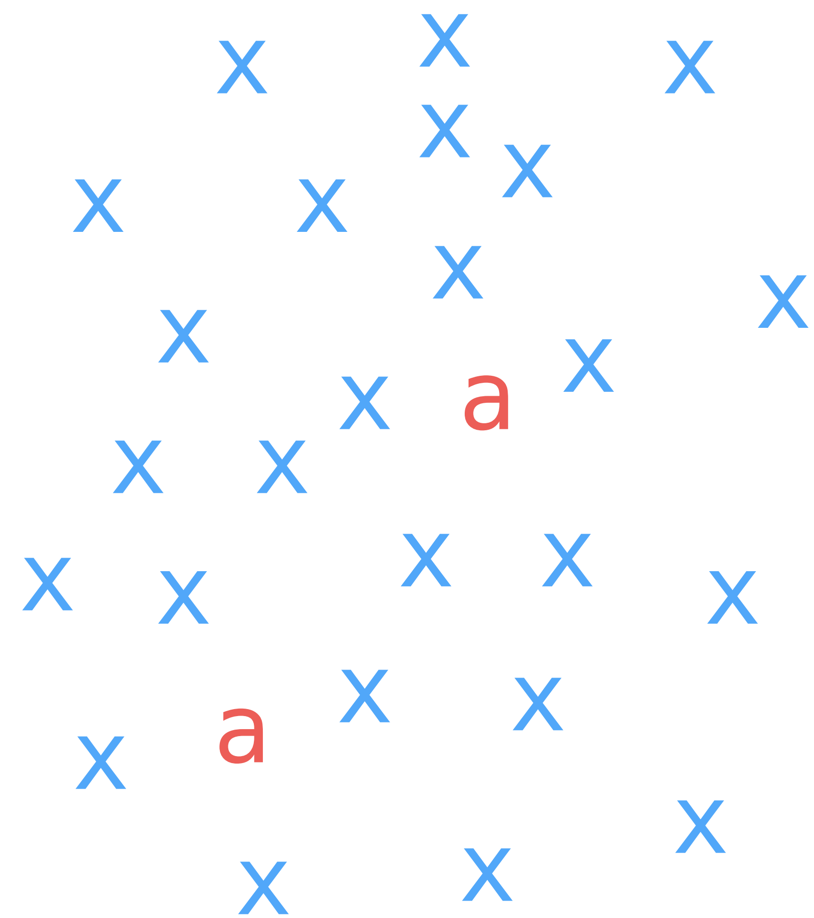
Produce y iff at least 1a and 1b



$$\boxed{\begin{array}{l} a, x \rightarrow a, a \\ a, b \rightarrow a, y \end{array}}$$

$\Theta(\log n)$
expected time

Produce y iff at least 2a



$$\boxed{\begin{array}{l} a, x \rightarrow a, a \\ a, a \rightarrow a, y \end{array}}$$

Outline

- Population protocols model
- Examples
- Definition of "stable computation" (captures the computation style of examples)
- Time model and computational complexity
- Consensus / approximate majority algorithm
- Biological connections
- Programming molecular interactions

Approximate Majority Population Protocol (aka Consensus)

$x, y \rightarrow x, b$

$x, y \rightarrow b, y$

$x, b \rightarrow x, x$

$y, b \rightarrow y, y$

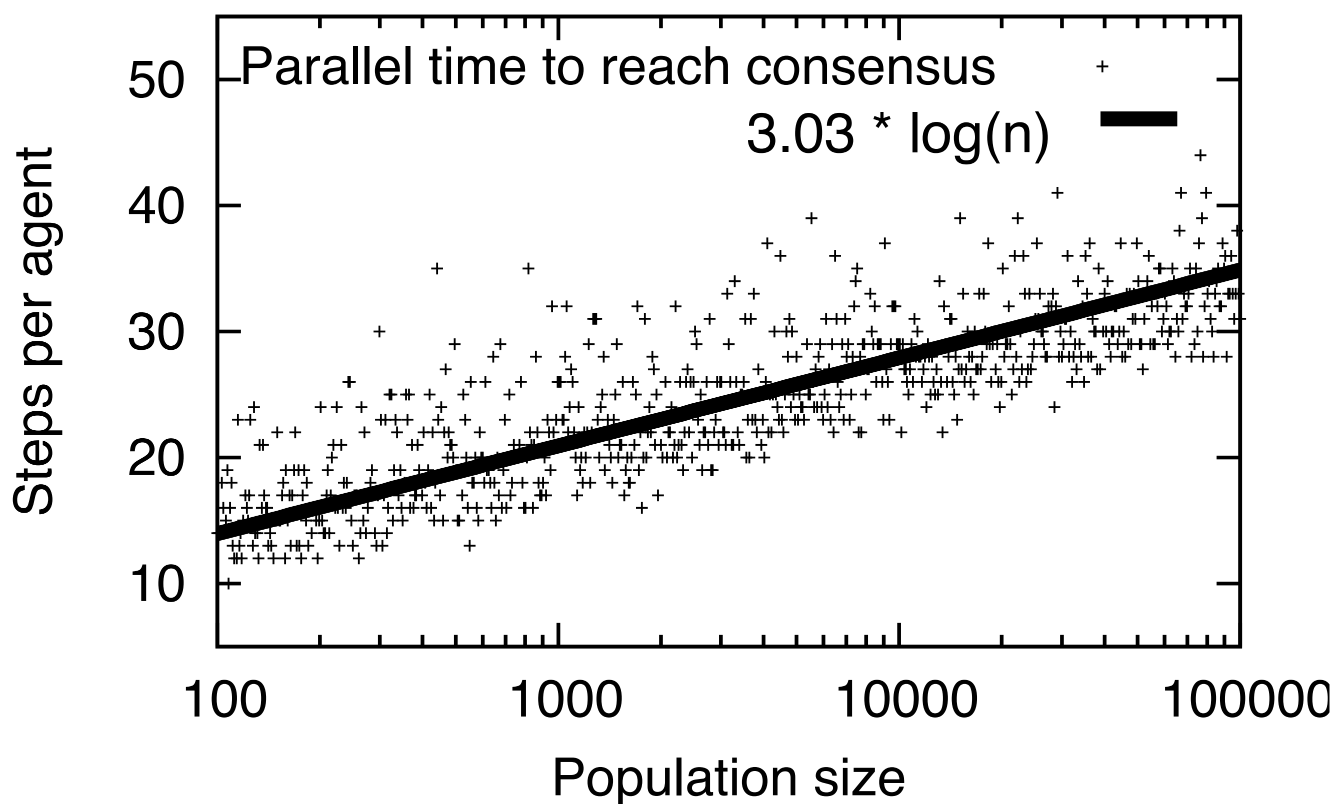
[run simulation]

[Angluin, Aspnes, Eisenstat DISC'07]

Approximate Majority Population Protocol (aka Consensus)

The expected time to converge is provably Θ(log *n*)

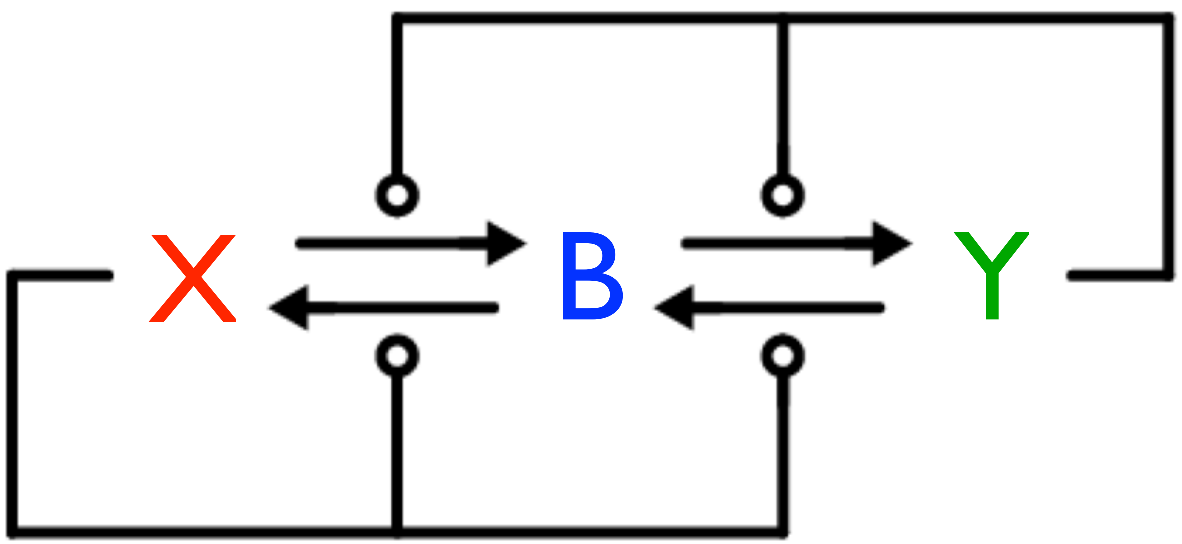
starting configuration: half x and half y



[Angluin, Aspnes, Eisenstat DISC'07]

Approximate Majority Population Protocol in Biology

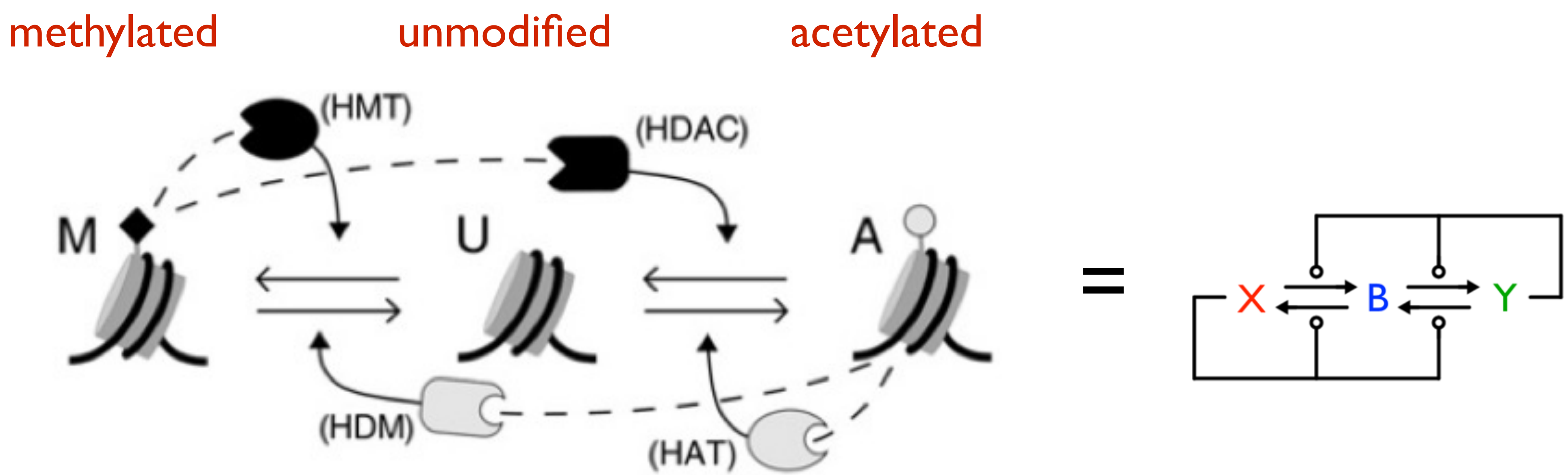
X + Y → X + B
X + Y → B + Y
X + B → X + X
Y + B → Y + Y

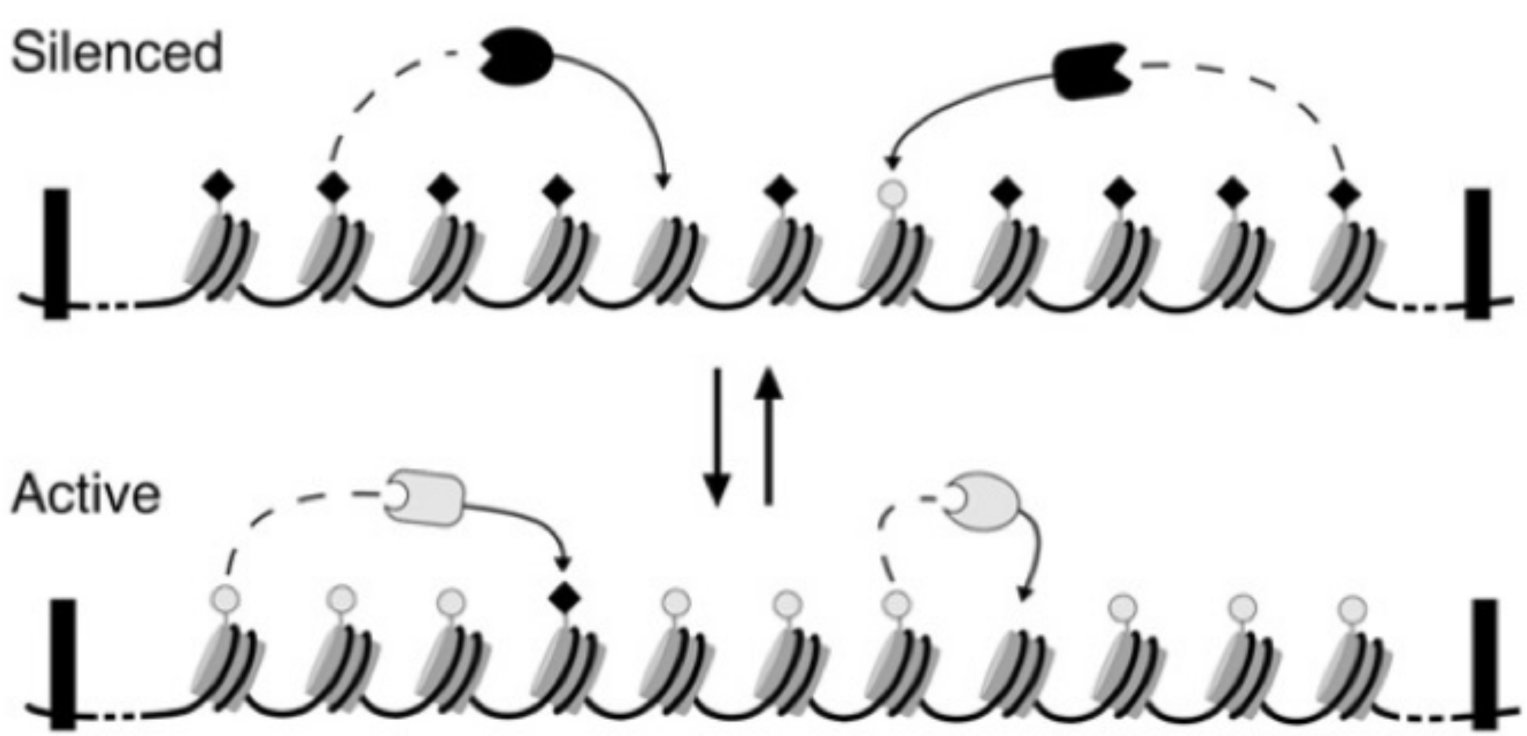


n = total number of molecules (X, Y, B)

Approximate Majority Population Protocol in Biology

"Epigenetic Memory by Nucleosome Modification"

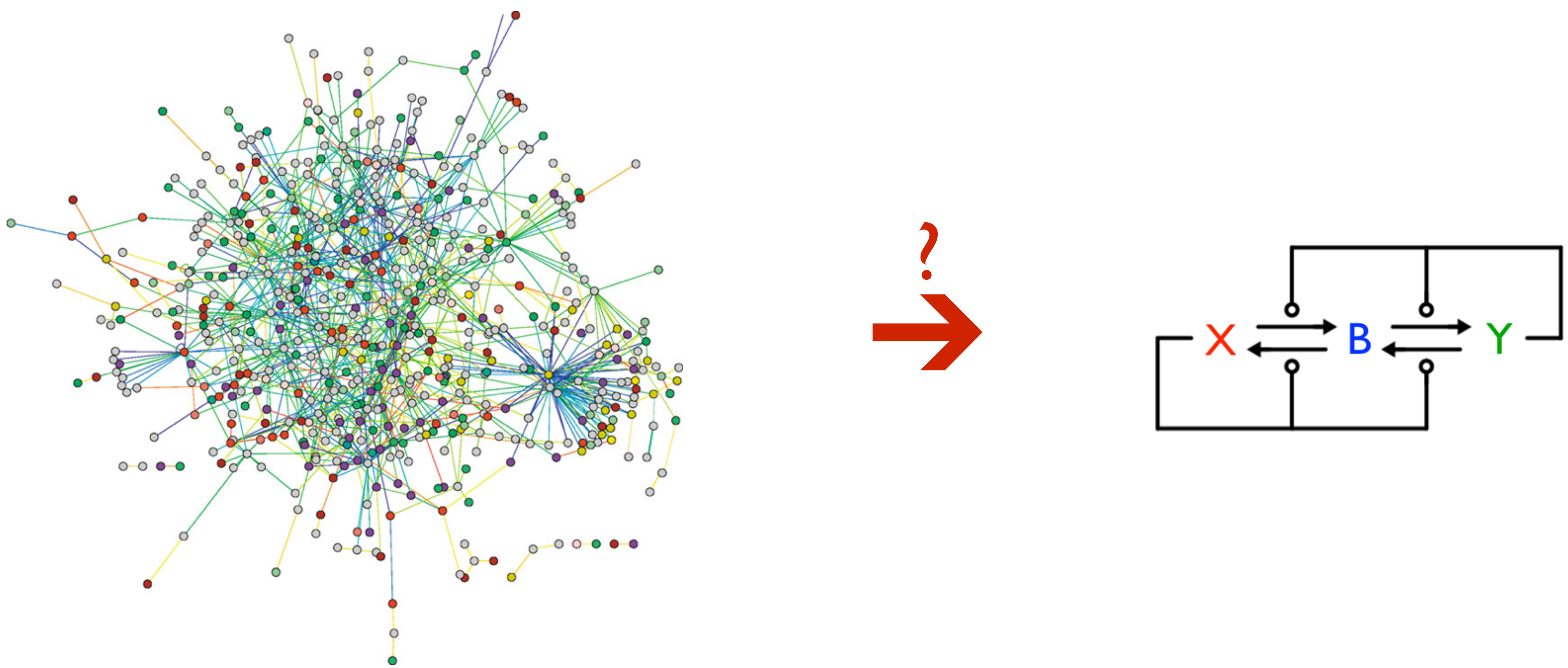




Dodd, Micheelsen, Sneppen, Thon, *Cell* 129, 813-822 (2007)

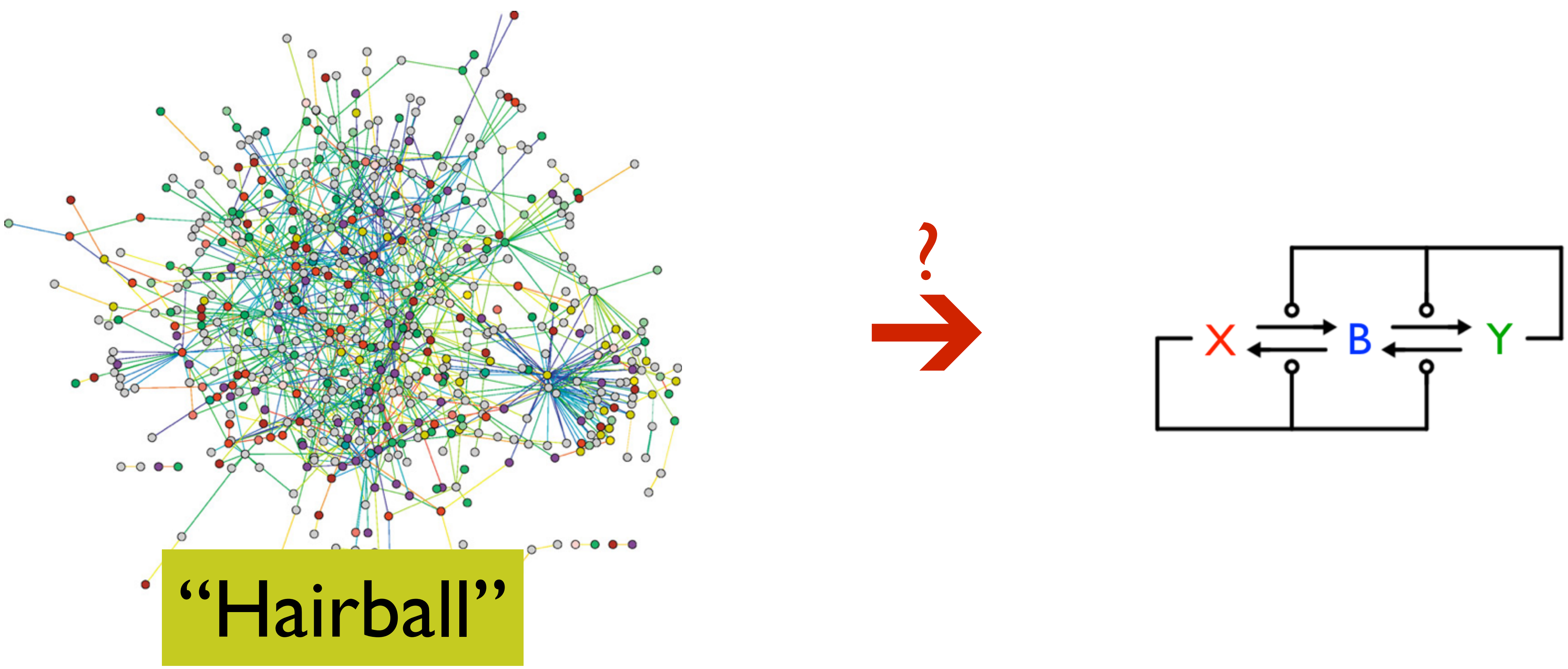
How Can We Identify Algorithms in Biology?

Does a biologically messy network X "implement" some ideal algorithm Y?



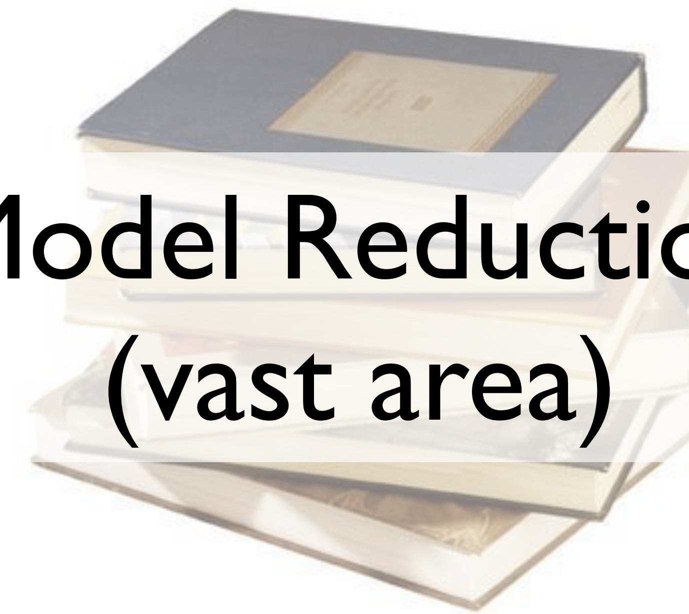
How Can We Identify Algorithms in Biology?

Does a biologically messy network X "implement" some ideal algorithm Y?



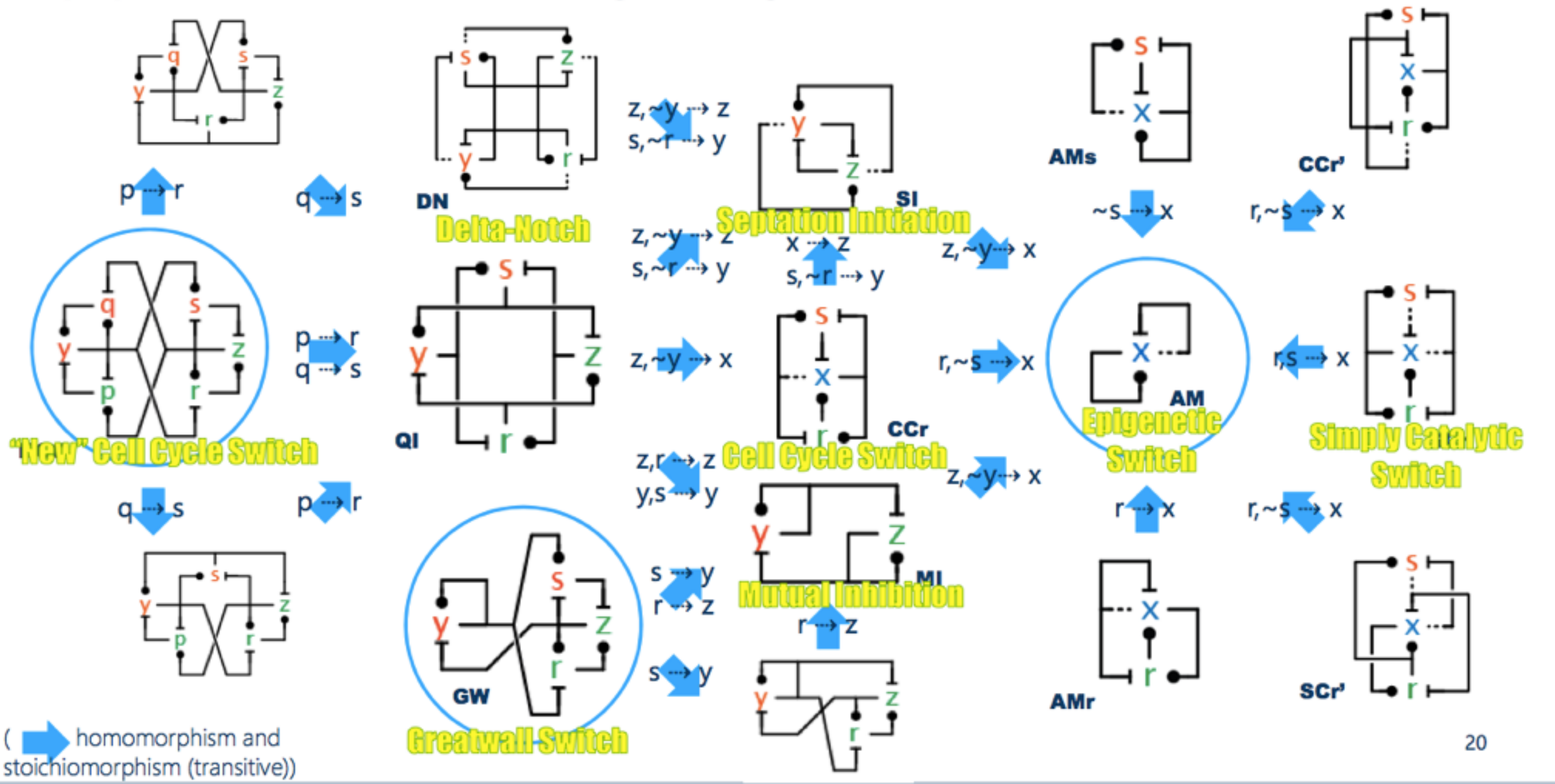
How Can We Identify Algorithms in Biology?

Intermediary Species → Model Reduction (vast area)



Symmetries → CRN Morphisms

Approximate Majority Emulation Zoo



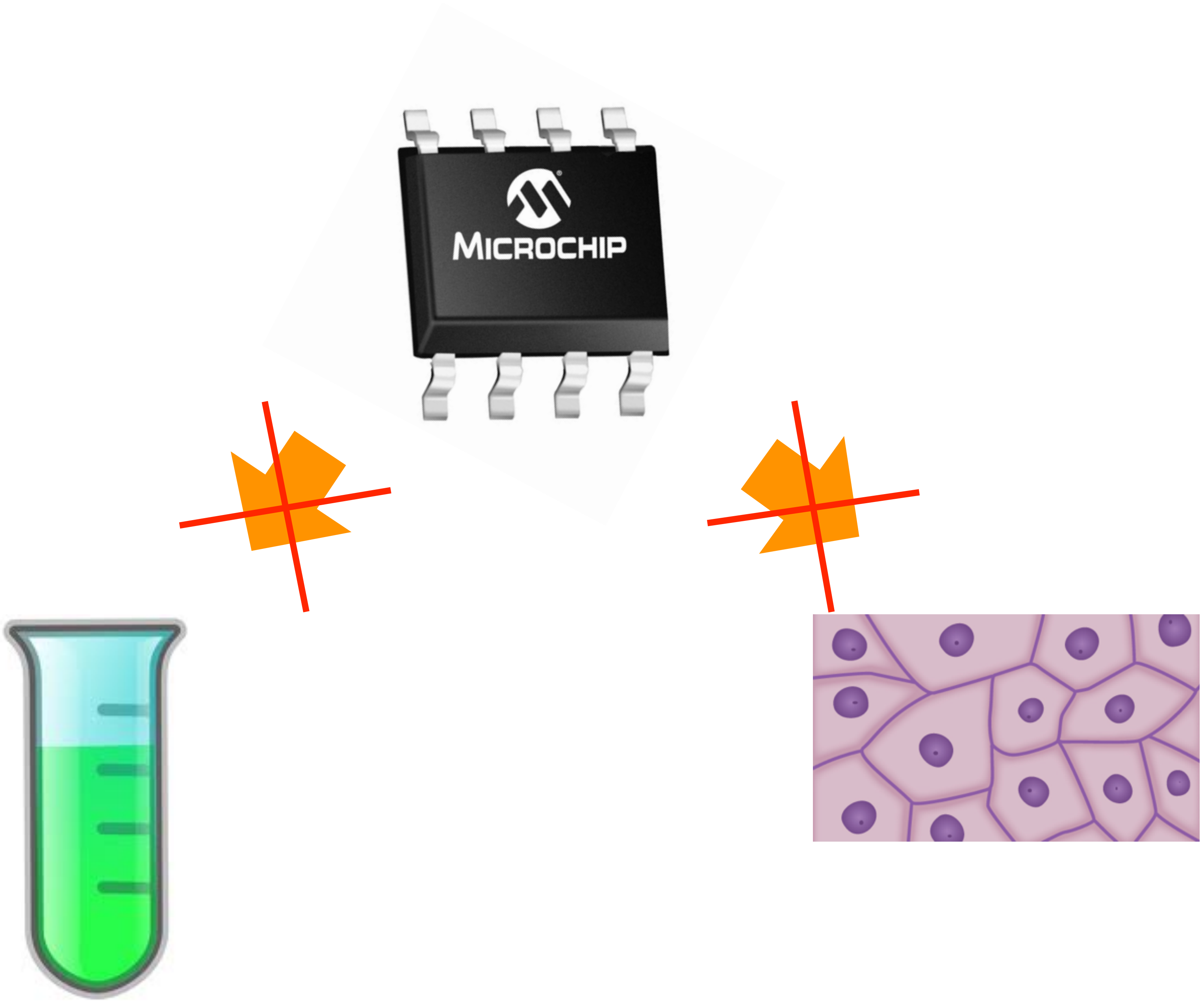
[Cardelli, "Morphisms of reaction networks that couple structure to function" 2014]

Outline

- Population protocols model
- Examples of "deterministic" computation
- Formally defining "deterministic computation": stable computation
- Time model and computational complexity
- Consensus / approximate majority algorithm
- Biological connections
- Programming molecular interactions

Why Compute with Molecules?

- Embed programming into environments not compatible with electronics



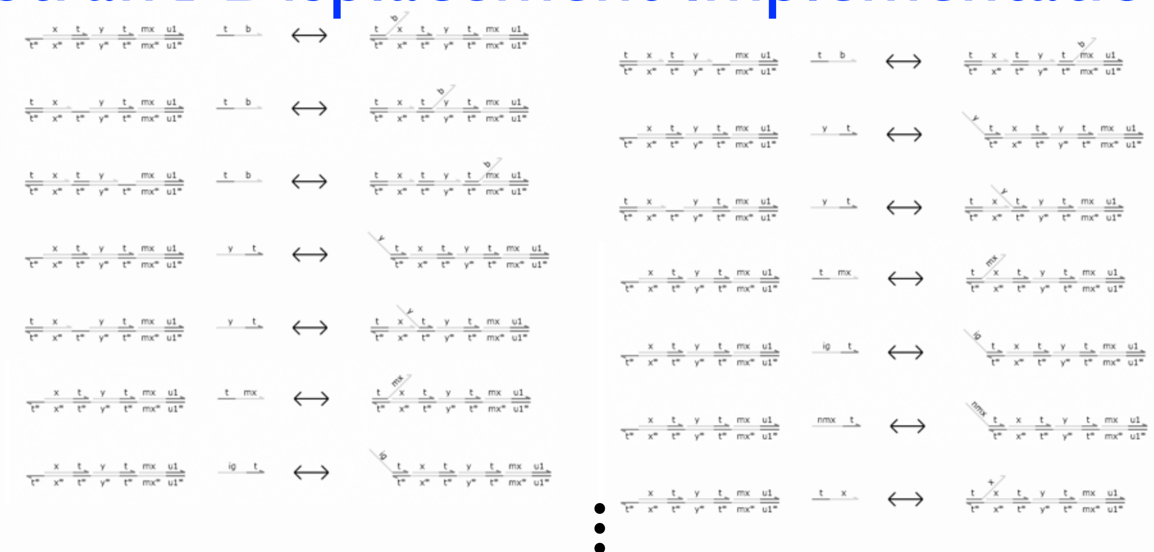
Strand Displacement Implementation of the Approximate Majority Network

Goal: Approximate Majority

$$X + Y \rightarrow B + Y$$
$$X + Y \rightarrow X + B$$
$$B + X \rightarrow X + X$$
$$B + Y \rightarrow Y + Y$$

compile

Strand Displacement Implementation



"3 rules" reactions

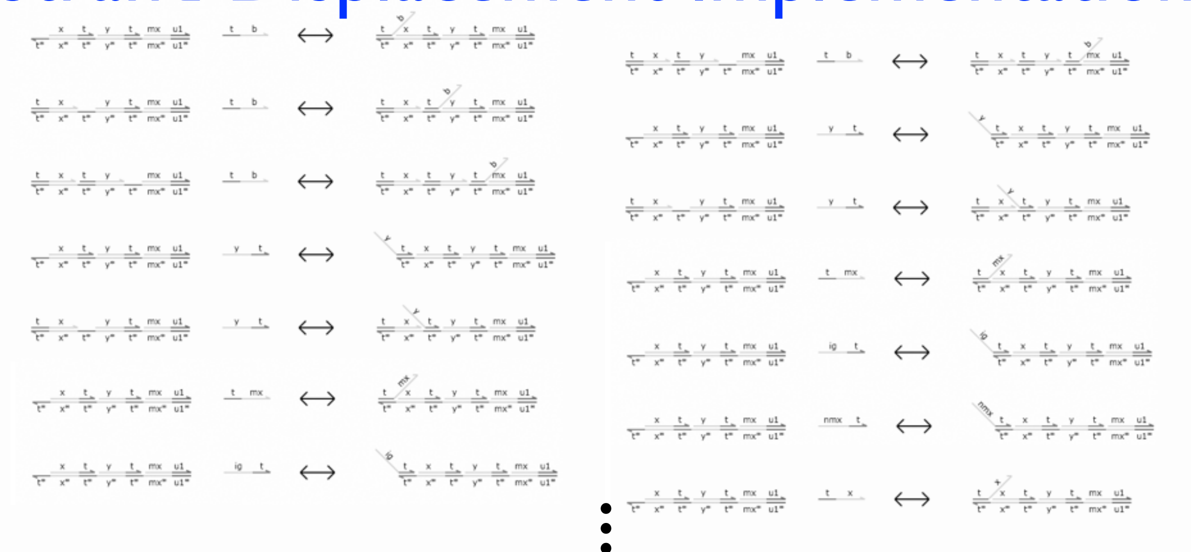
Strand Displacement Implementation of the Approximate Majority Network

Goal: Approximate Majority

$$X + Y \rightarrow B + Y$$
$$X + Y \rightarrow X + B$$
$$B + X \rightarrow X + X$$
$$B + Y \rightarrow Y + Y$$

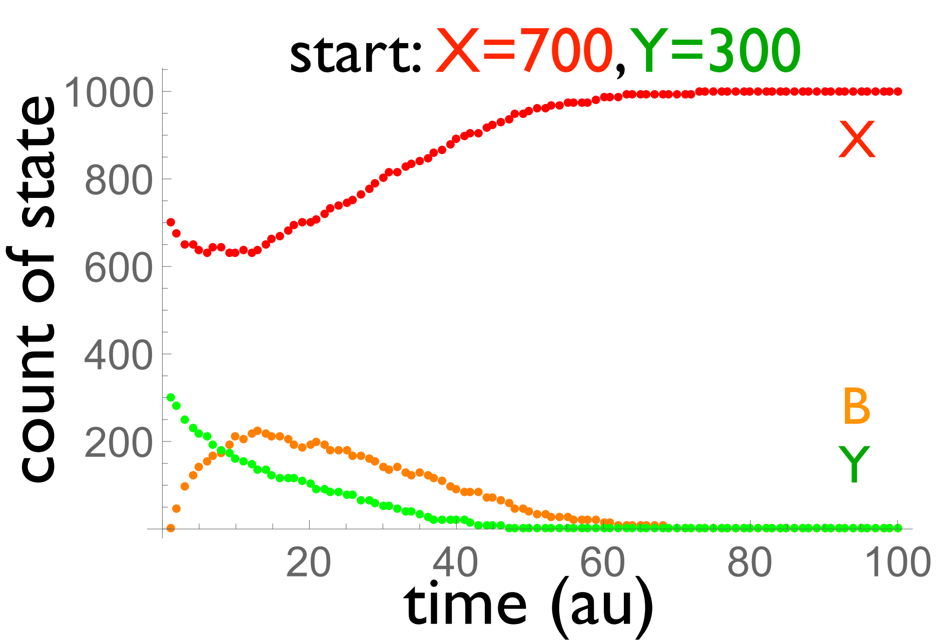
compile

Strand Displacement Implementation

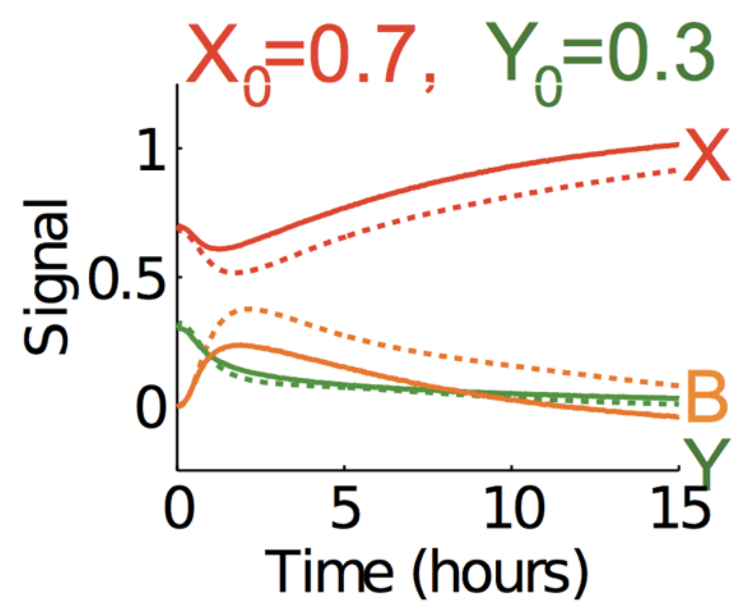


"3 rules" reactions

Ideal



Test tube



Chen, Dalchau, Srinivas, Phillips, Cardelli, Soloveichik, Seelig, *Nature Nanotechnology* 2013

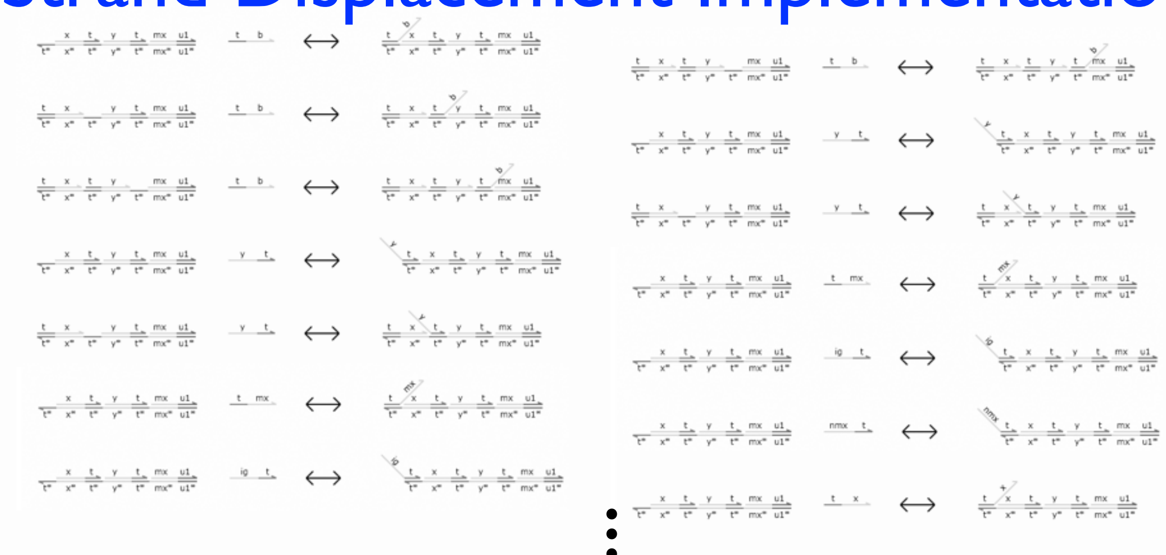
Strand Displacement Implementation of the Approximate Majority Network

Goal: Approximate Majority

$$X + Y \rightarrow B + Y$$
$$X + Y \rightarrow X + B$$
$$B + X \rightarrow X + X$$
$$B + Y \rightarrow Y + Y$$

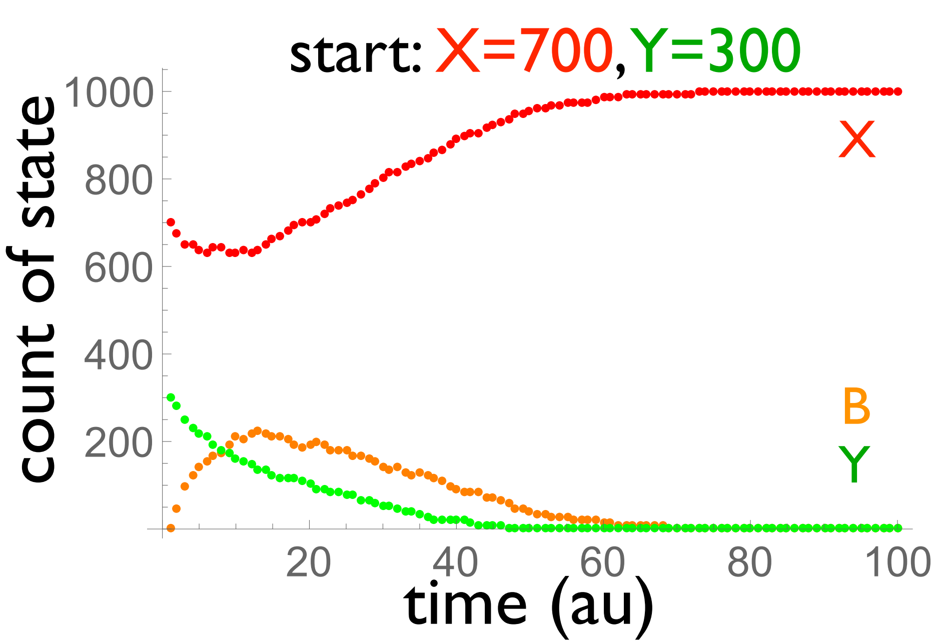
compile

Strand Displacement Implementation



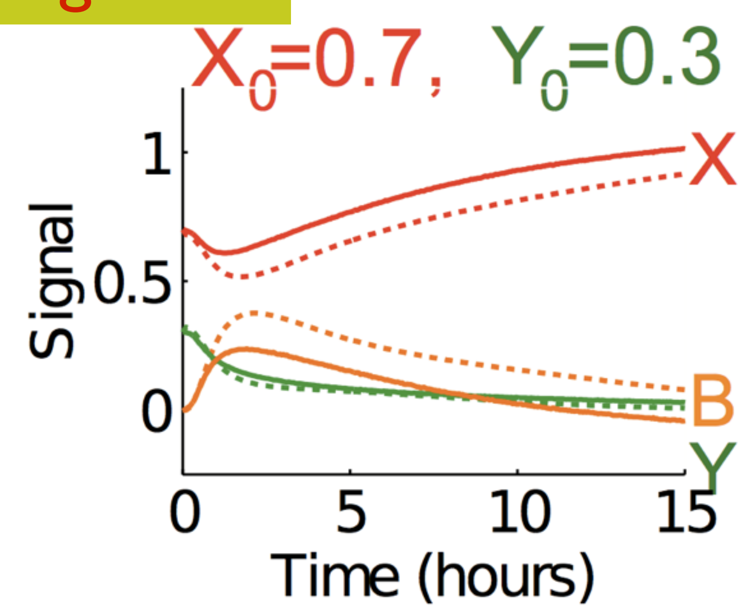
“3 rules” reactions

Ideal



Test tube

10^{11} agents!



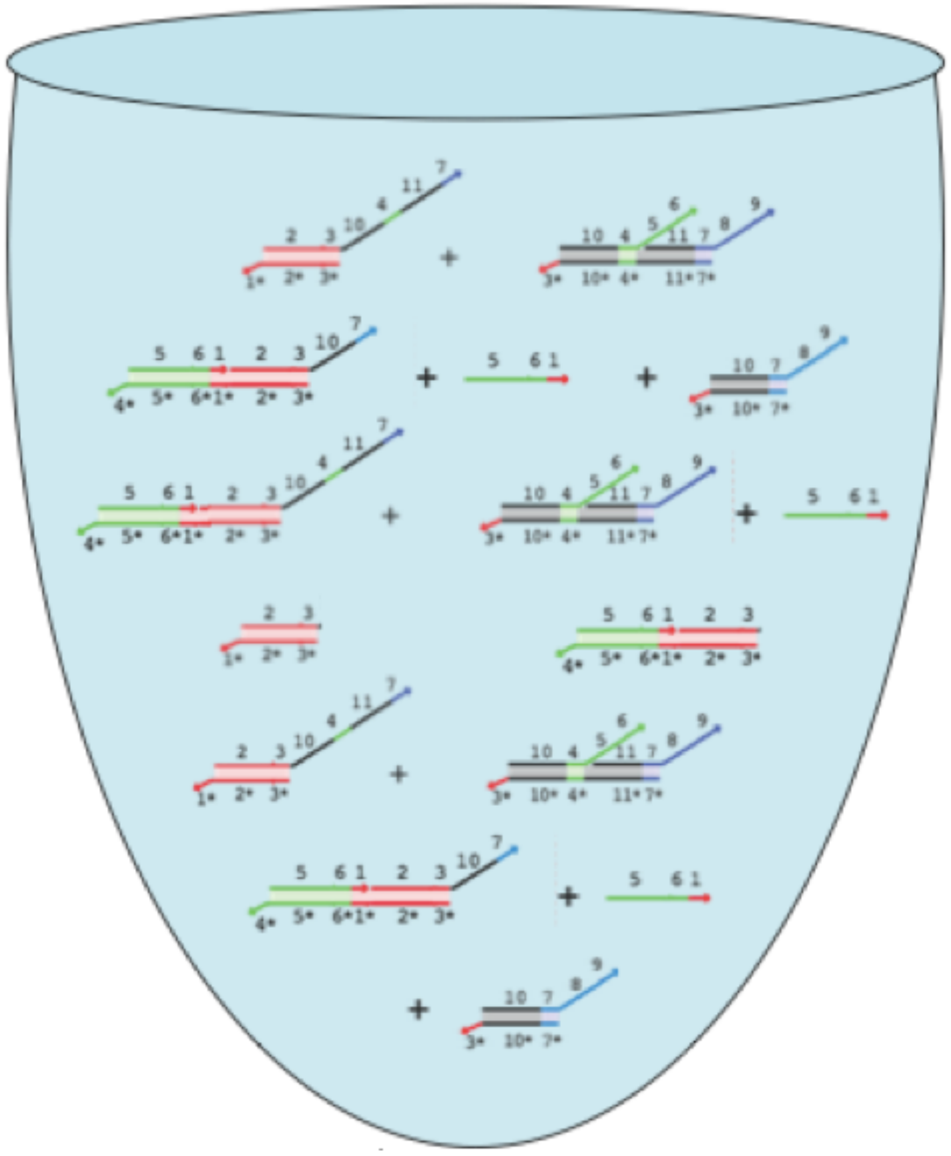
Chen, Dalchau, Srinivas, Phillips, Cardelli, Soloveichik, Seelig, *Nature Nanotechnology* 2013

Every goal reaction corresponds to a set of implementation reactions

$$X3 + X4 \xrightarrow{k_1} X5$$

$$X5 \xrightarrow{k_2} X1$$

$$X1 + X2 \xrightarrow{k_3} X3$$



Every goal reaction corresponds to a set of implementation reactions

$$X3 + X4 \xrightarrow{k_1} X5$$

$$X5 \xrightarrow{k_2} X1$$

$$X1 + X2 \xrightarrow{k_3} X3$$

$$X3 + g1 \xrightarrow{k_4} i + g2$$
$$i + g2 \xrightarrow{k_5} X3 + g1$$
$$i + x4 \xrightarrow{k_6} j + w1$$
$$j + g3 \xrightarrow{k_7} X5 + w2$$

$$X5 + g4 \xrightarrow{k_8} k + w3$$
$$k + g5 \xrightarrow{k_9} X1 + w4$$

$$X1 + g6 \xrightarrow{k_{10}} l + g7$$
$$l + g7 \xrightarrow{k_{11}} X1 + g6$$
$$l + X2 \xrightarrow{k_{12}} m + w5$$
$$m + g8 \xrightarrow{k_{13}} X3 + w6$$

How can you tell that an implementation of a reaction is correct? Can be tricky!

[Shin, Thachuk, Winfree, VEMDP 2014]

How can you tell that an implementation of a reaction is correct? Can be tricky!

Goal reactions

$A \rightarrow B + C$

Implementation

$A \rightarrow i1 + B$

$i1 + B \rightarrow A$

$i1 \rightarrow C$

[Shin, Thachuk, Winfree, VEMDP 2014]

How can you tell that an implementation of a reaction is correct? Can be tricky!

Goal reactions

1. $A \rightarrow B + C$
2. $B + D \rightarrow B + E$
3. $A + E \rightarrow F$

Implementation

1.1. $A \rightarrow i1 + B$
1.2. $i1 + B \rightarrow A$
1.3. $i1 \rightarrow C$

[Shin, Thachuk, Winfree, VEMDP 2014]

How can you tell that an implementation of a reaction is correct? Can be tricky!

Goal reactions

1. $A \rightarrow B + C$

Implementation

1.1. $A \rightarrow i1 + B$

1.2. $i1 + B \rightarrow A$

1.3. $i1 \rightarrow C$

Ex. Error

$\{1\ A, 1\ D\}$

2. $B + D \rightarrow B + E$

3. $A + E \rightarrow F$

[Shin, Thachuk, Winfree, VEMDP 2014]

How can you tell that an implementation of a reaction is correct? Can be tricky!

Goal reactions

1. $A \rightarrow B + C$
2. $B + D \rightarrow B + E$
3. $A + E \rightarrow F$

Implementation

1.1. $A \rightarrow i1 + B$
1.2. $i1 + B \rightarrow A$
1.3. $i1 \rightarrow C$

Ex. Error

$\{1\ A, 1\ D\}$

$\Downarrow$ 1.1

$\{1\ i1, 1\ B, 1\ D\}$

$\Downarrow$ 2

$\{1\ i1, 1\ B, 1\ E\}$

$\Downarrow$ 1.2

$\{1\ A, 1\ E\}$

$\Downarrow$ 3

$\{1\ F\}$

[Shin, Thachuk, Winfree, VEMDP 2014]

Conclusions

- Population protocols model considers extremely weak agents, no control over interactions
- Application domains: sensor networks, molecular computation
- Complex global behavior possible: arithmetic, boolean predicates, consensus, etc.
- Time complexity: exponential difference between certain tasks, many open questions
- "Programming language" for chemistry?

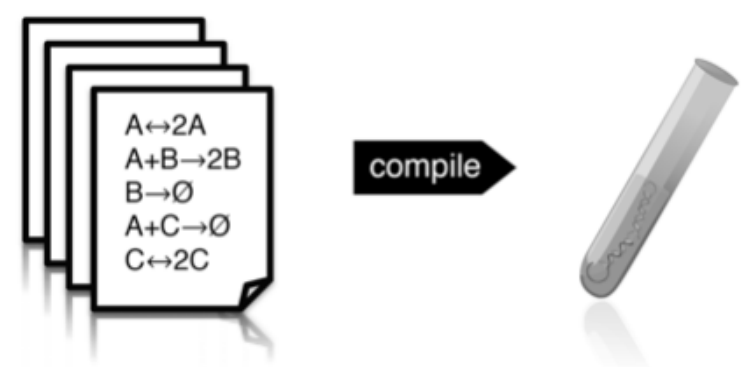
More: Google "population protocols", "chemical reaction networks"

New grad course!

[EE381V] Programming with Molecules

Graduate Course :: Spring 2016

Instructor: David Soloveichik
Lecture: T, TH 12:30PM-2PM
Classroom: CBA 4.338



Description

We will discuss paradigms for programming complex behavior in (bio)chemical systems. Similar to how digital circuits and automata (e.g., finite state machines) are fundamental abstractions for electronic computation, we are interested in models of computation as embedded in the chemical world. The motivating natural phenomena include biological self-organization and information processing in chemical pathways in cells. Applications of rationally designed molecular systems will be introduced from synthetic biology, and DNA bioengineering and nanotechnology. Topics will include algorithmic tile-assembly and cellular automata, discrete and continuous chemical reaction networks, population protocols, and strand displacement cascades. We will study chemical computation by reasoning, simulation, and formal proofs about these and other models. Besides chemistry, we find applications in distributed computing settings where weak computational agents must operate in a disordered environment (e.g., sensor networks). The course will consist of a combination of lectures, paper discussions, and group projects.