The Curse of Hamilton’s Chairs

Ahmed Shalaby
2nd year PhD

Supervisor: Damien Woods

Hamilton Institute
Maynooth University
National University of Ireland Maynooth
Fondúireacht Eolaíochta Éireann sfi Dá bhfuil romhainn
European Innovation Council
Funded by the European Union
erc

The Curse of Hamilton's Chairs

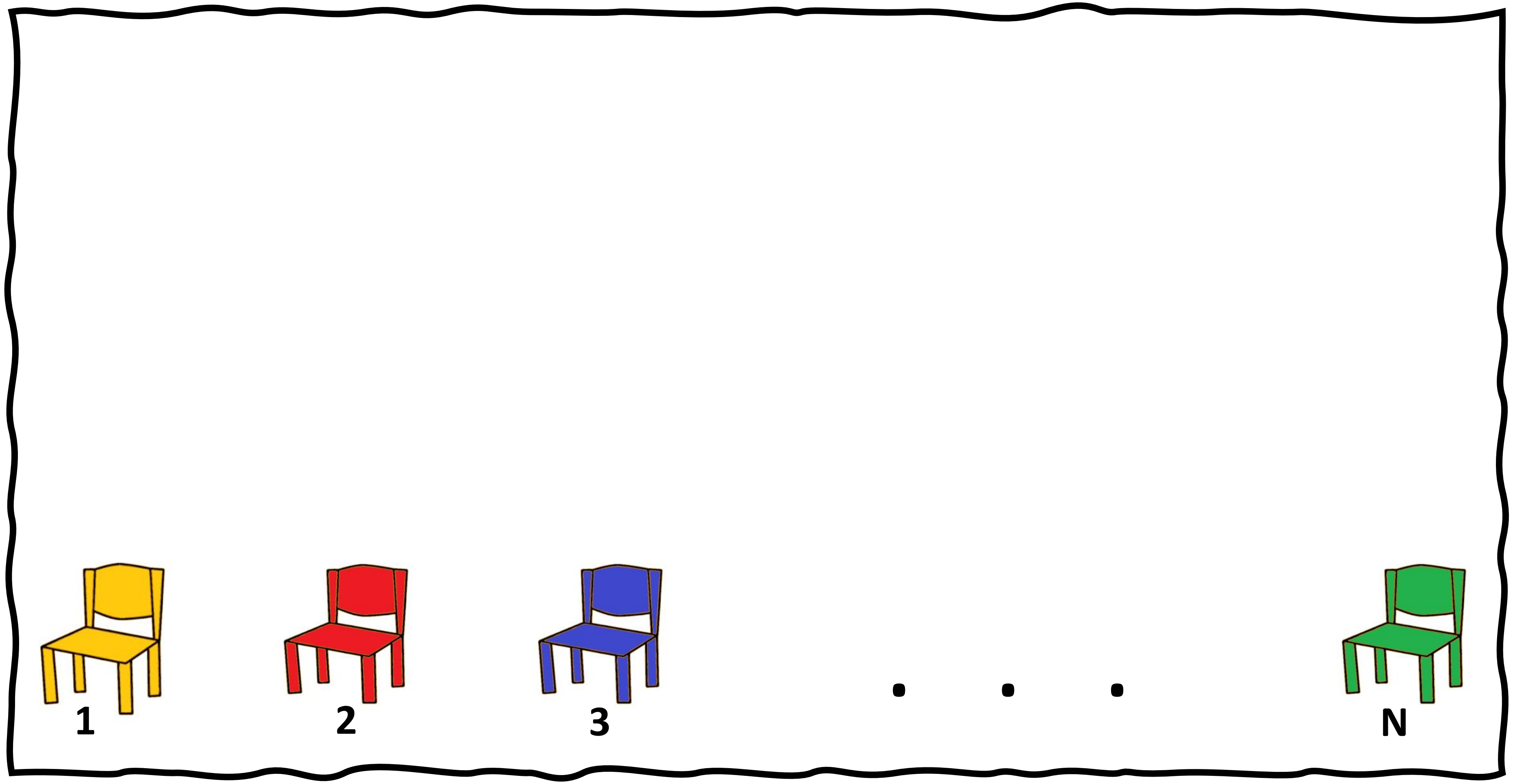
Thermodynamics of a multistranded one-dimension Scaffolded DNA Computer

Ahmed Shalaby
2nd year PhD

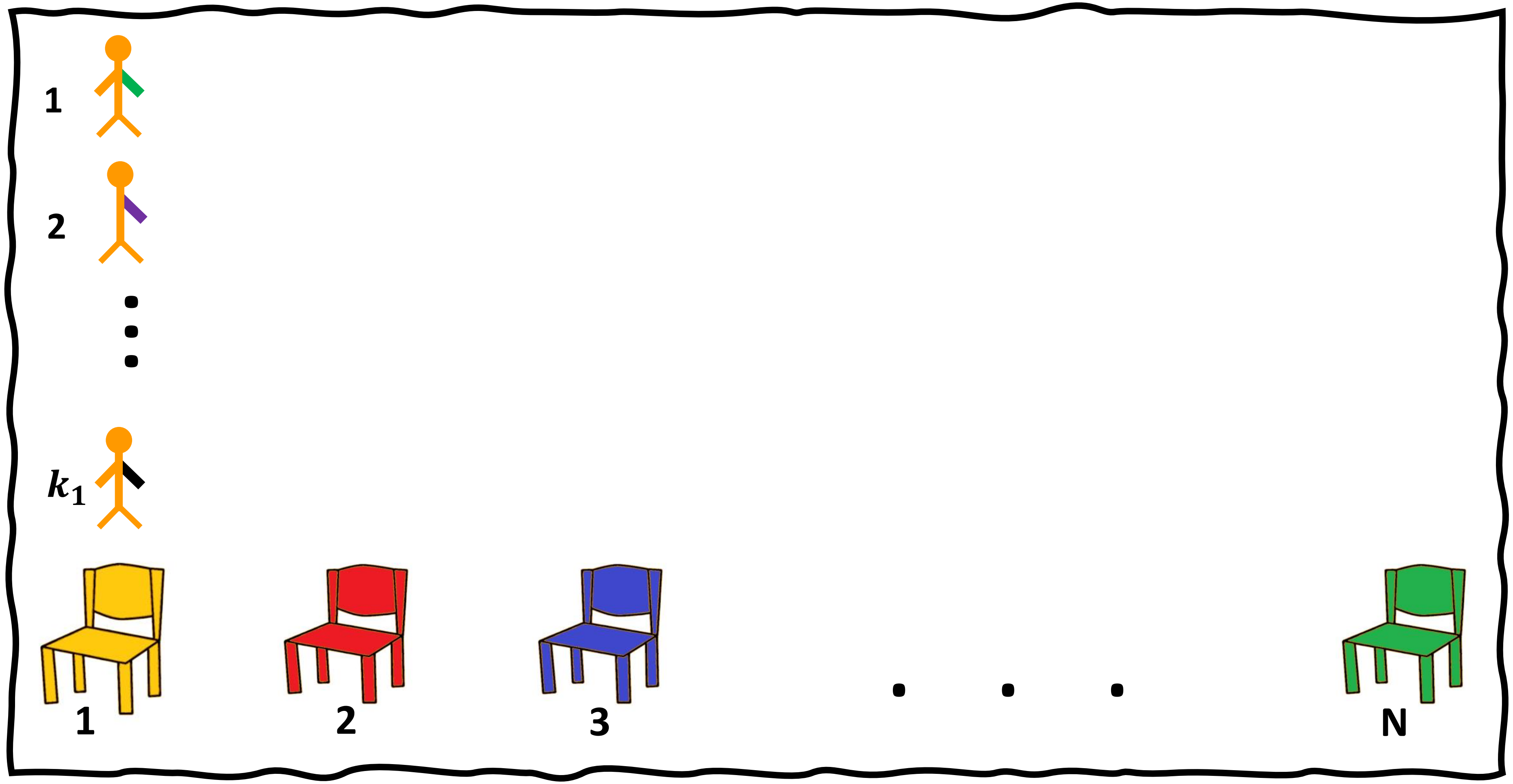
Supervisor: Damien Woods


EOLAS

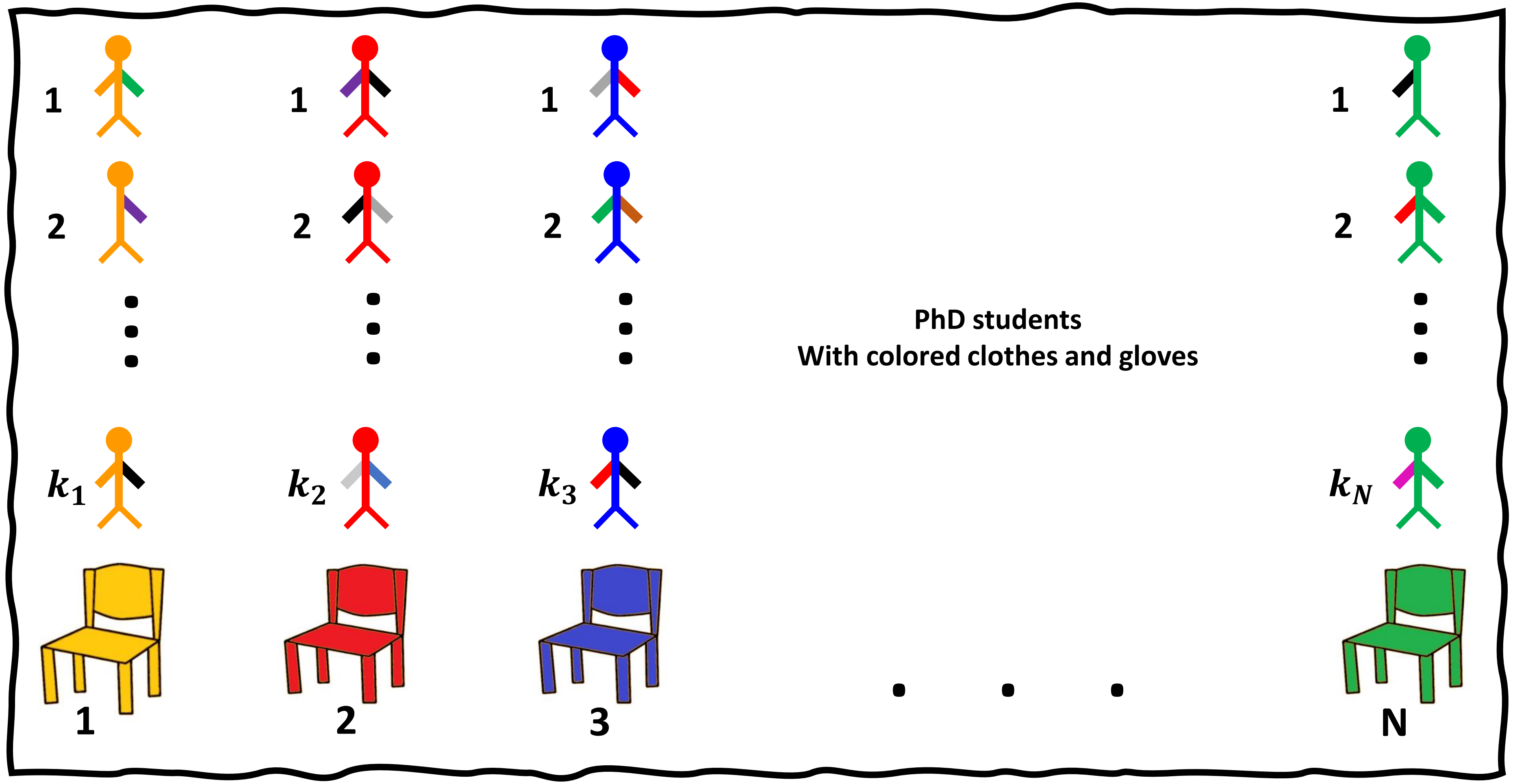
Let’s discover the rules of the game



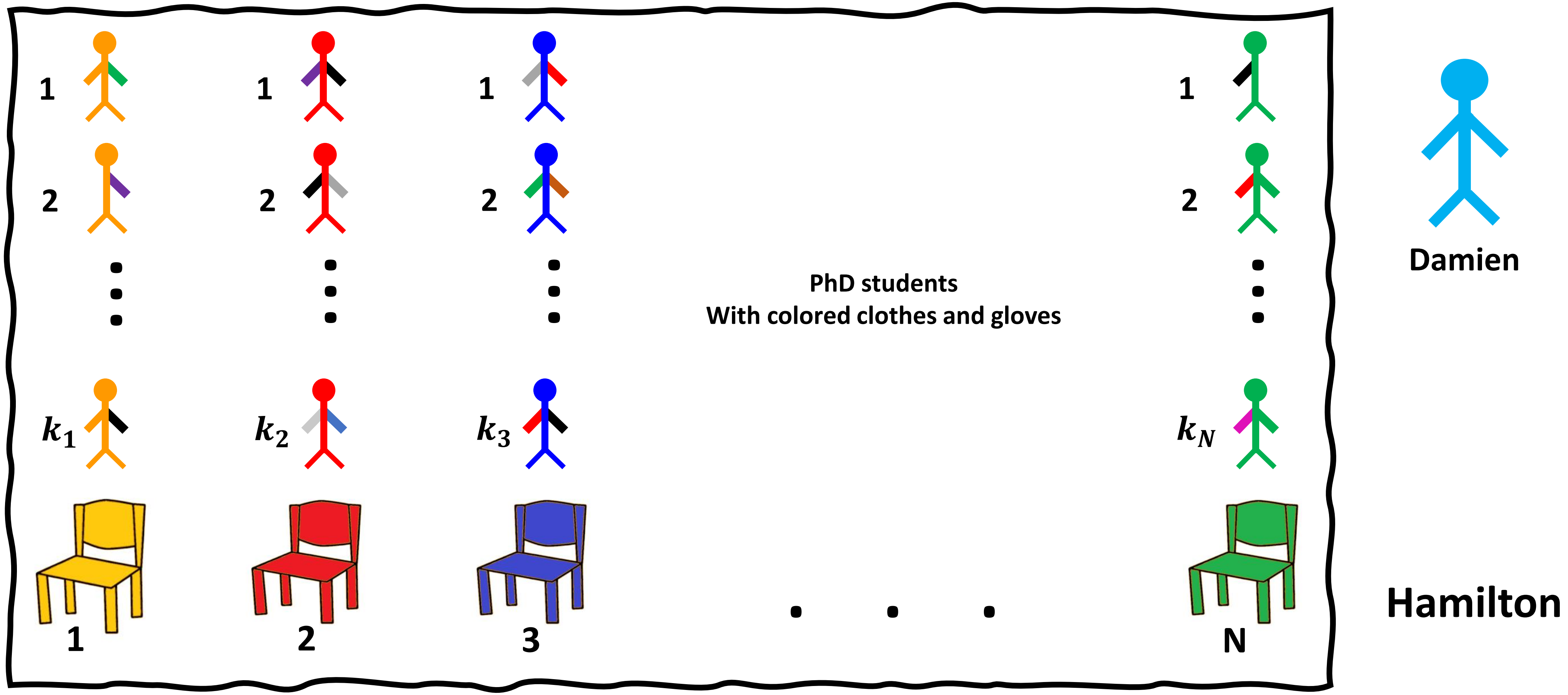
Hamilton

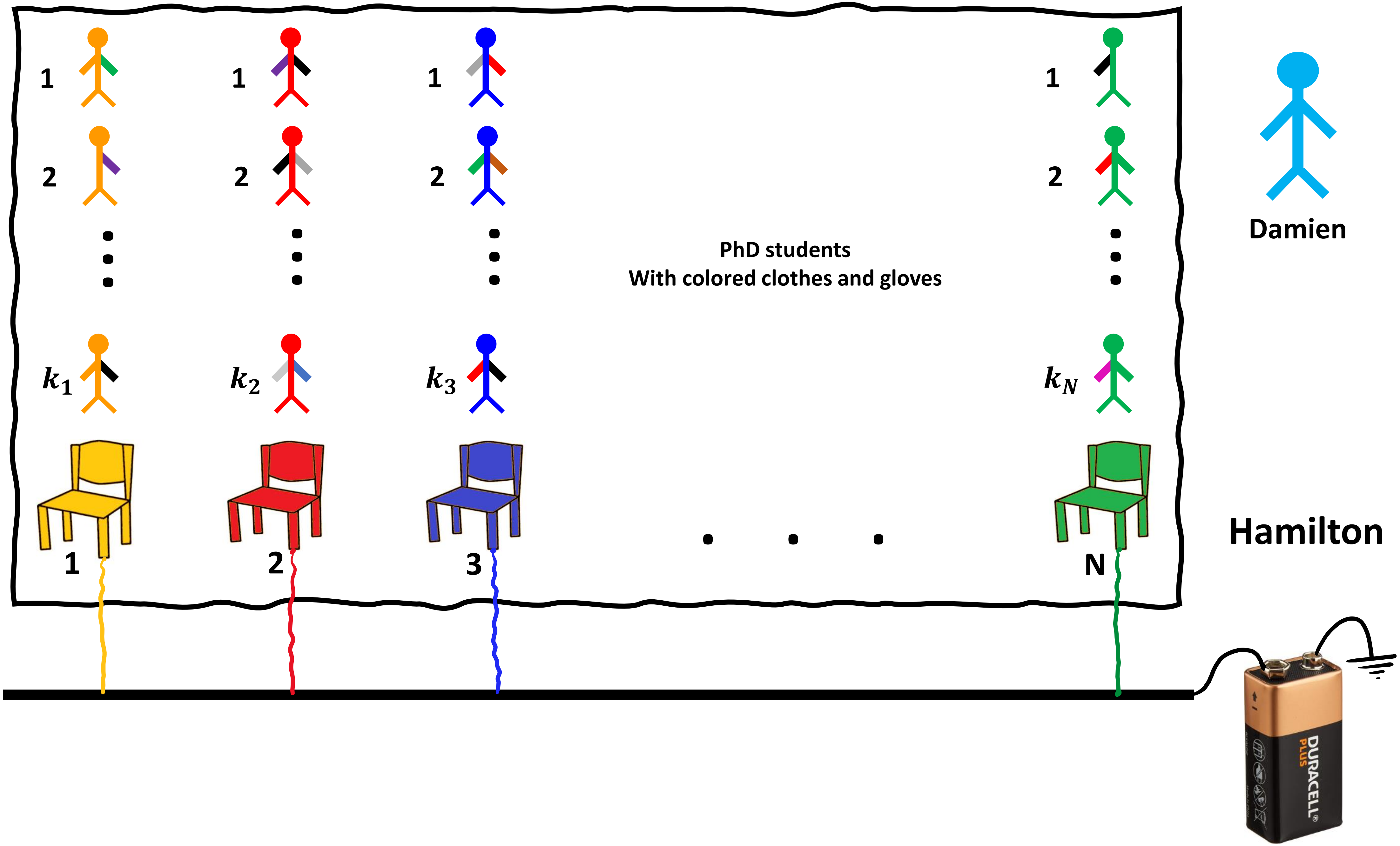


Hamilton

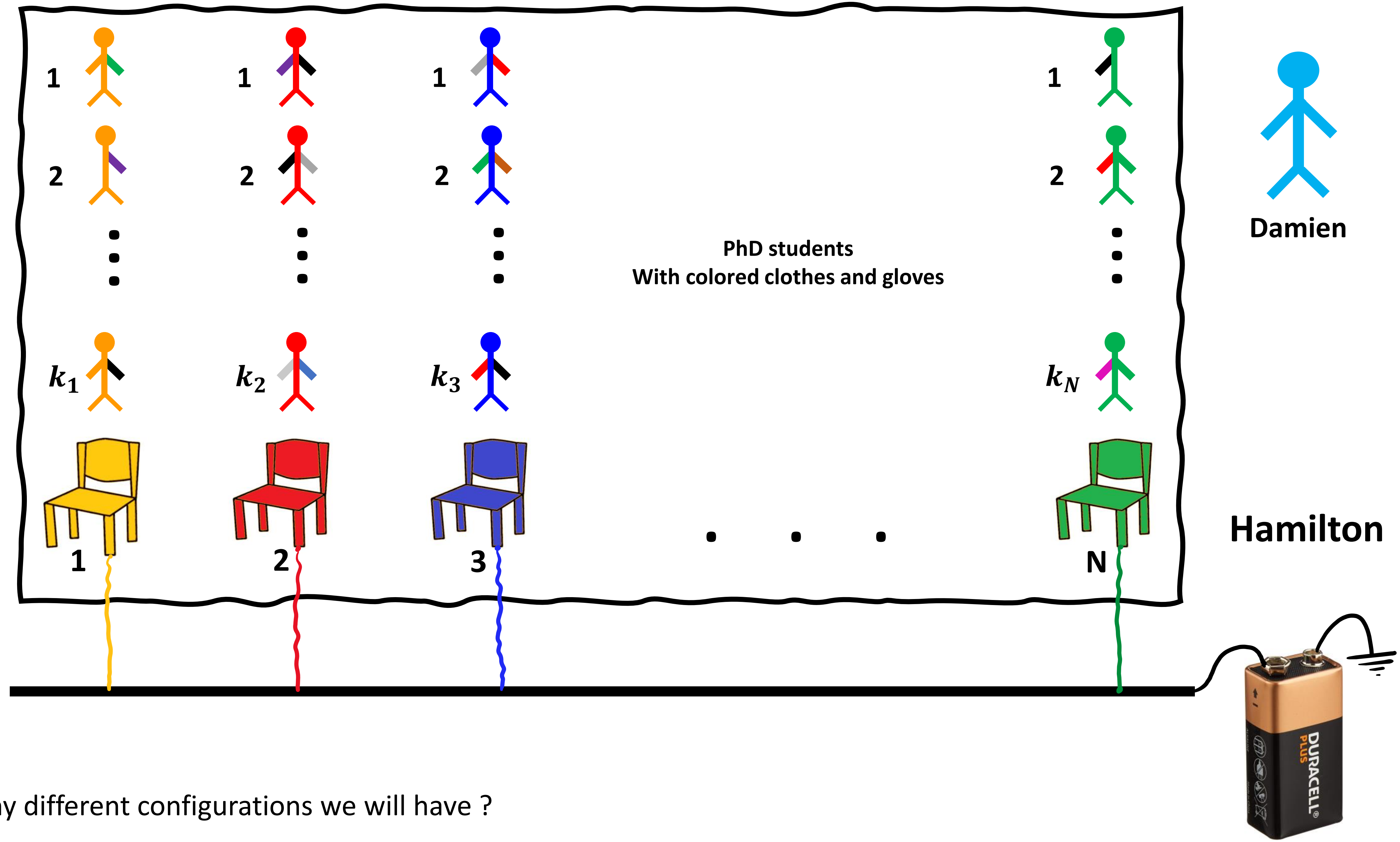

1
1
1
1
2
2
2
2
k_1
k_2
k_3
k_N
PhD students
With colored clothes and gloves
1
2
3
N

Hamilton

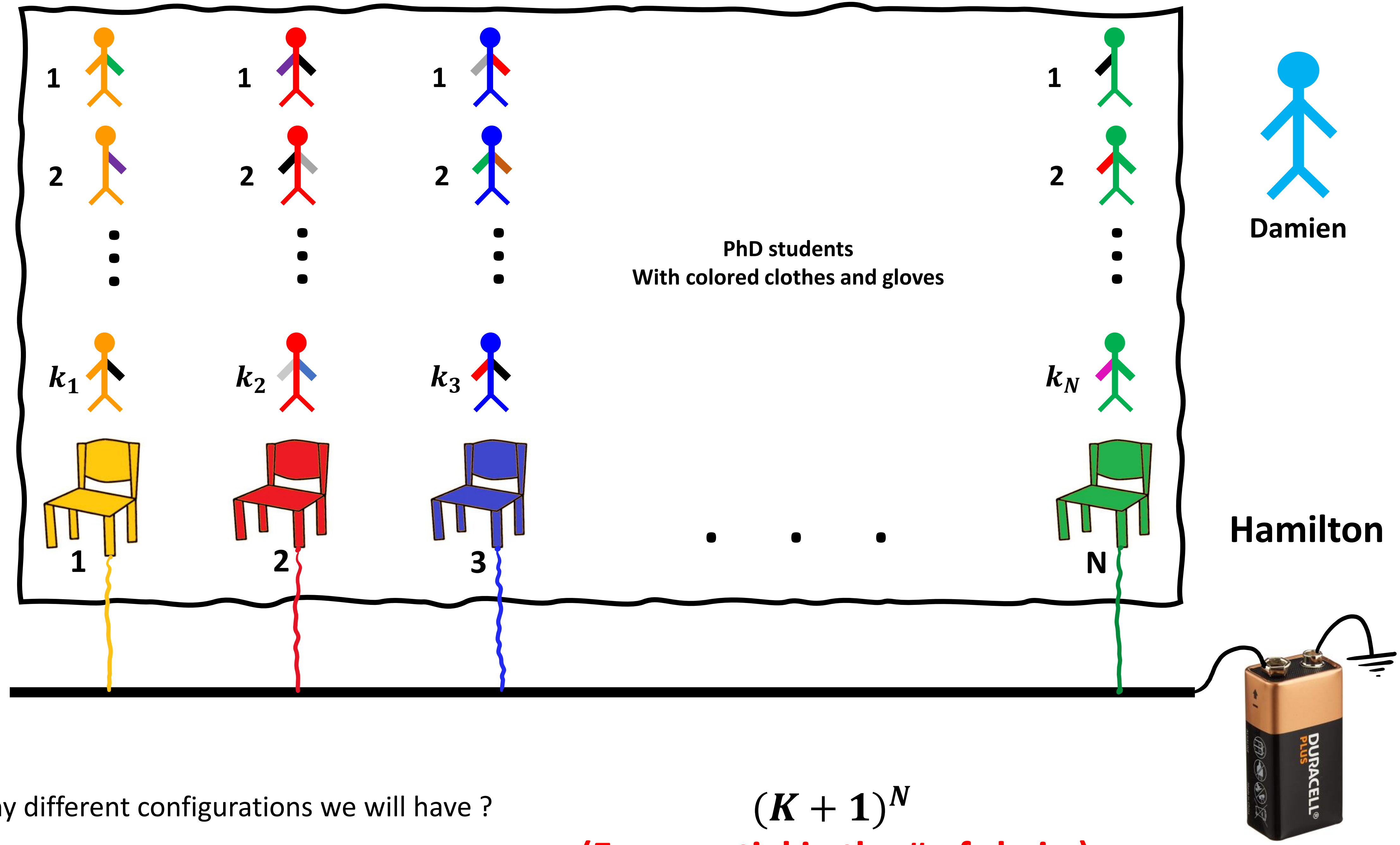

1
2
k_1
k_2
k_3
k_N
PhD students
With colored clothes and gloves
Damien
1
2
3
N
Hamilton



1
2
k_1
k_2
k_3
k_N
PhD students
With colored clothes and gloves
Damien
Hamilton
1
2
3
N
DURACELL
PLUS



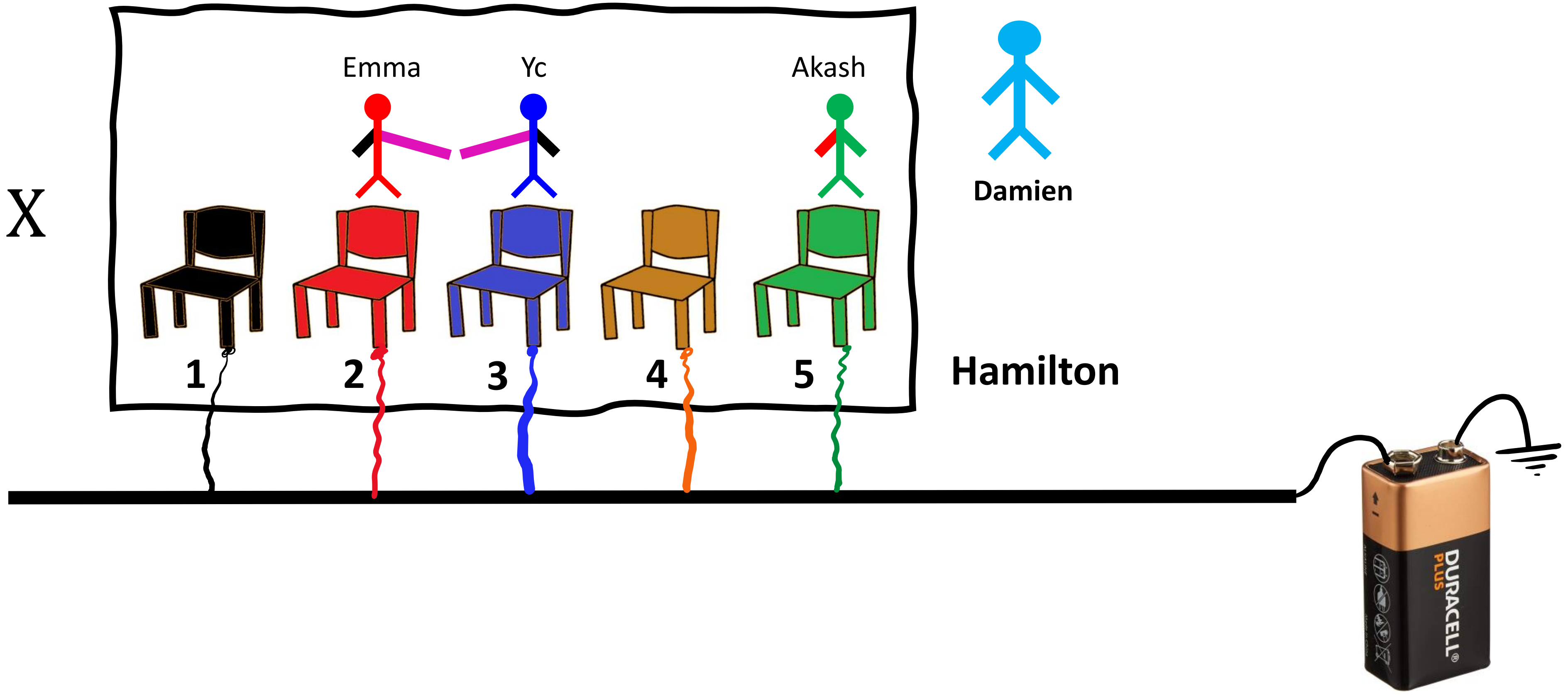
- How many different configurations we will have ?

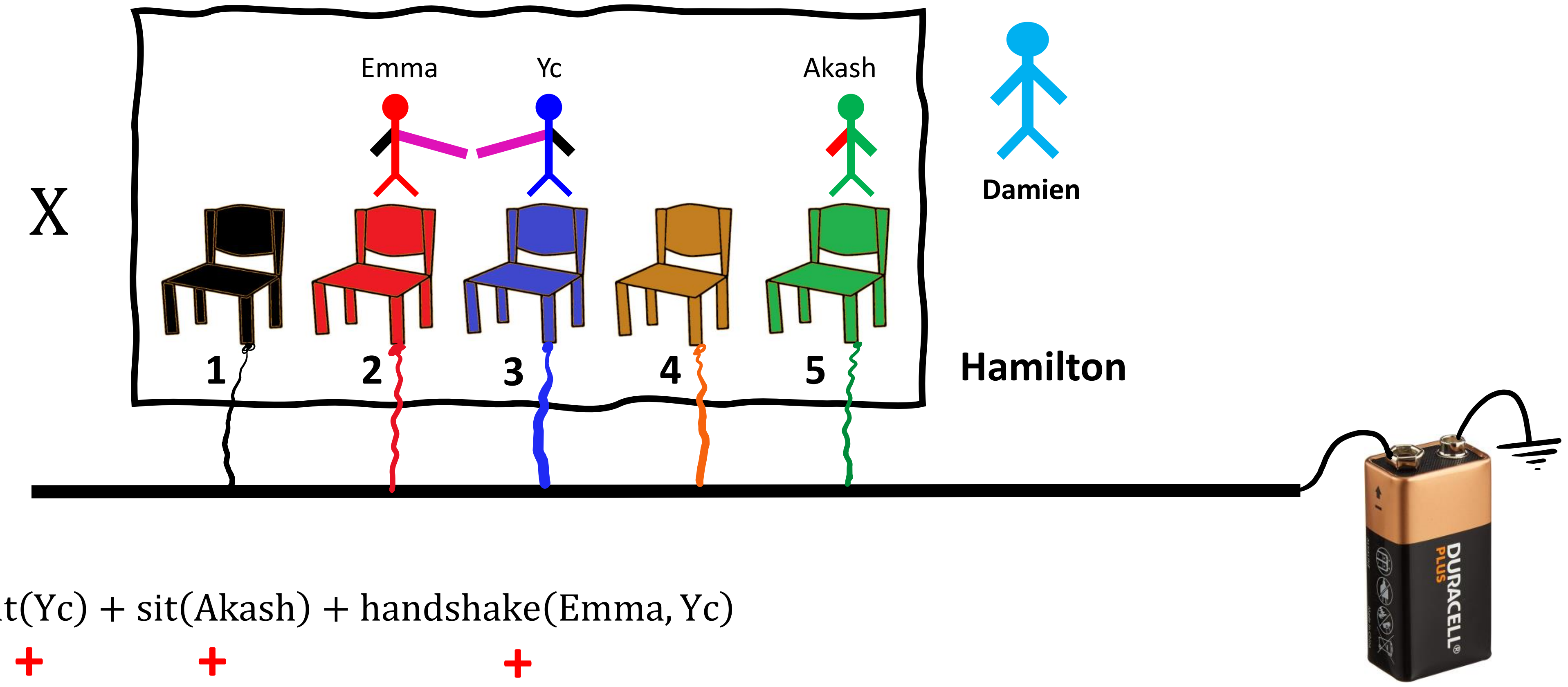


- How many different configurations we will have ?

$$(K+1)^N$$

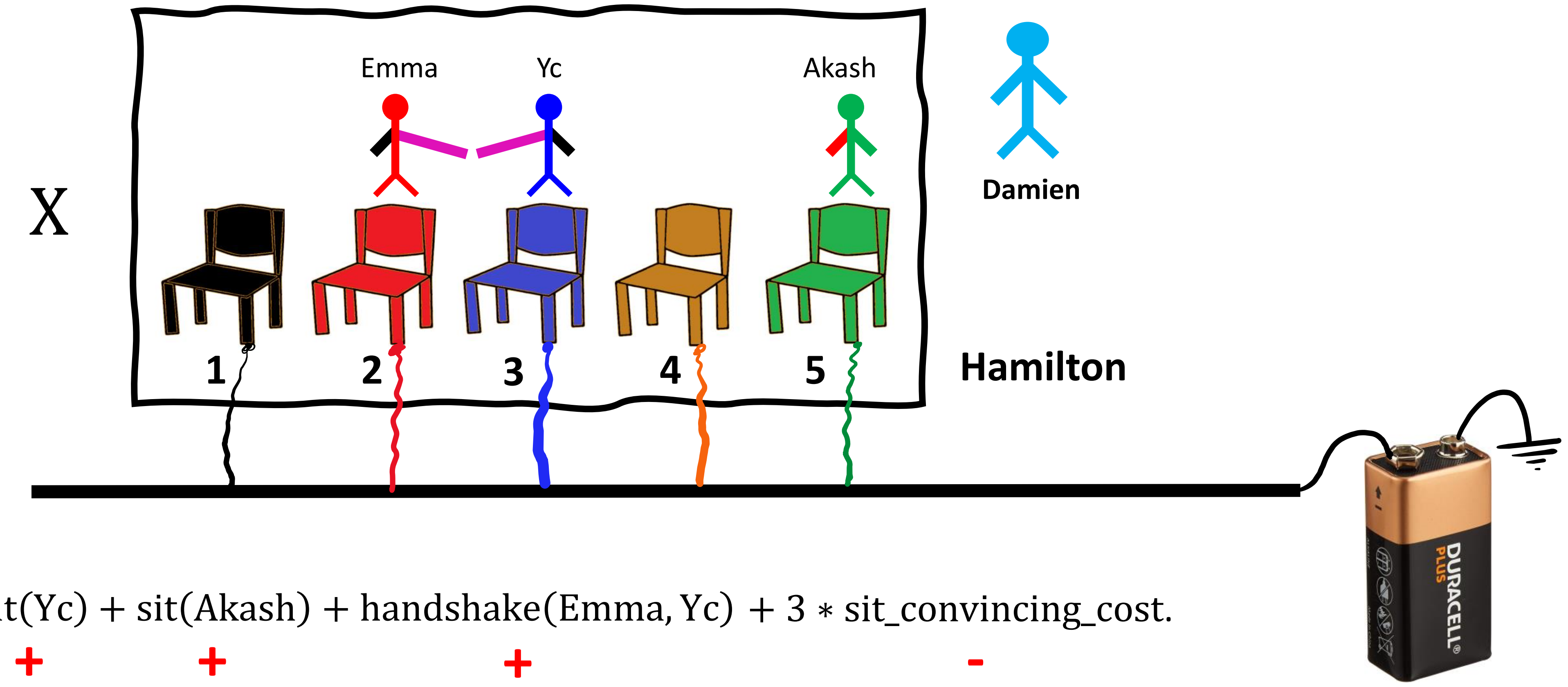
(Exponential in the # of chairs)


X
Emma
Yc
Akash
Damien
1
2
3
4
5
Hamilton
DURACELL PLUS



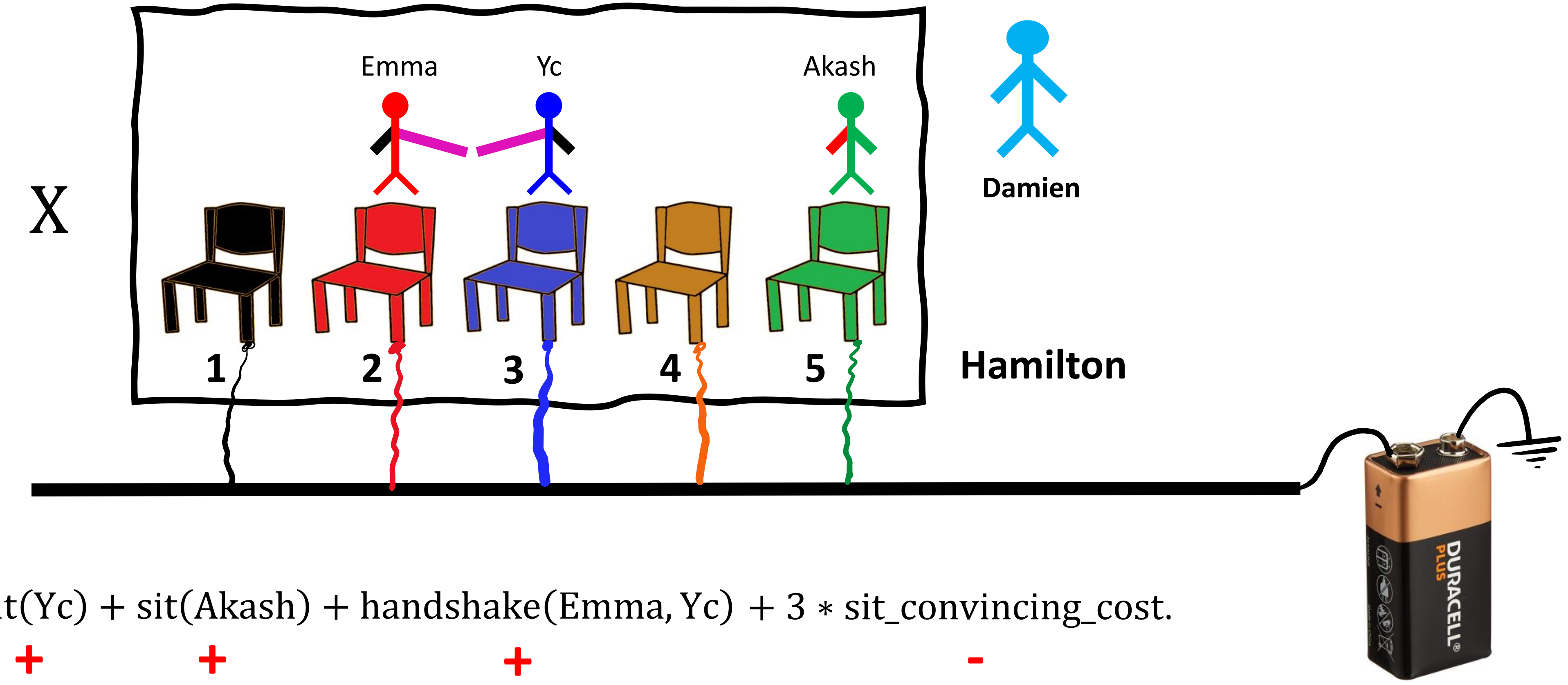
$$E(X) = sit(Emma) + sit(Yc) + sit(Akash) + handshake(Emma, Yc)$$

+ + + +



$$E(X) = \text{sit}(\text{Emma}) + \text{sit}(\text{Yc}) + \text{sit}(\text{Akash}) + \text{handshake}(\text{Emma}, \text{Yc}) + 3 * \text{sit\_convincing\_cost}.$$

\+ \+ \+ \+ -

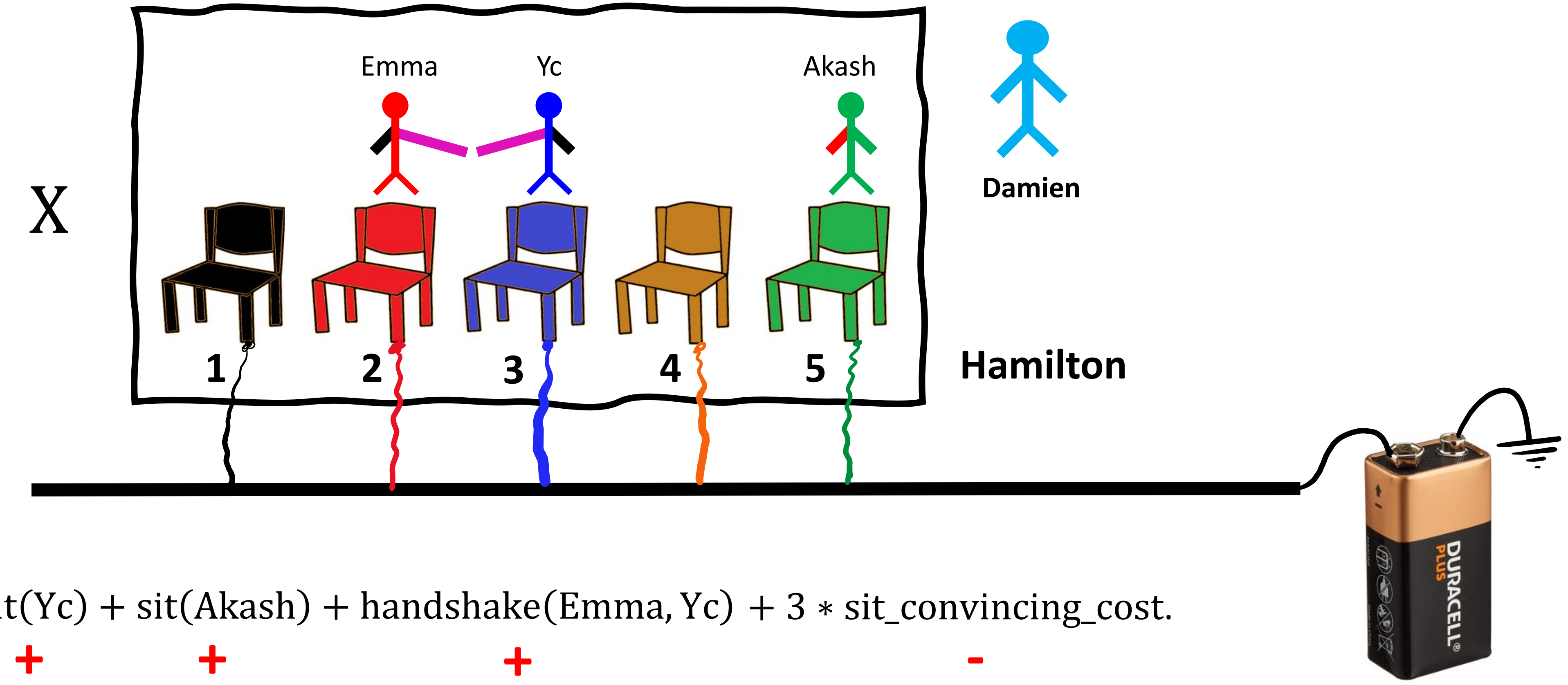


$$E(X) = \text{sit}(\text{Emma}) + \text{sit}(\text{Yc}) + \text{sit}(\text{Akash}) + \text{handshake}(\text{Emma}, \text{Yc}) + 3 * \text{sit\_convincing\_cost}.$$

\+ \+ \+ \+ -

We further assume the following:

- $|\text{sit}(p)| > |\text{sit\_convincing\_cost}|$. (Damien always gains by convincing a PhD student to sit)



$$E(X) = \text{sit}(\text{Emma}) + \text{sit}(\text{Yc}) + \text{sit}(\text{Akash}) + \text{handshake}(\text{Emma}, \text{Yc}) + 3 * \text{sit\_convincing\_cost}.$$

+ + + + -

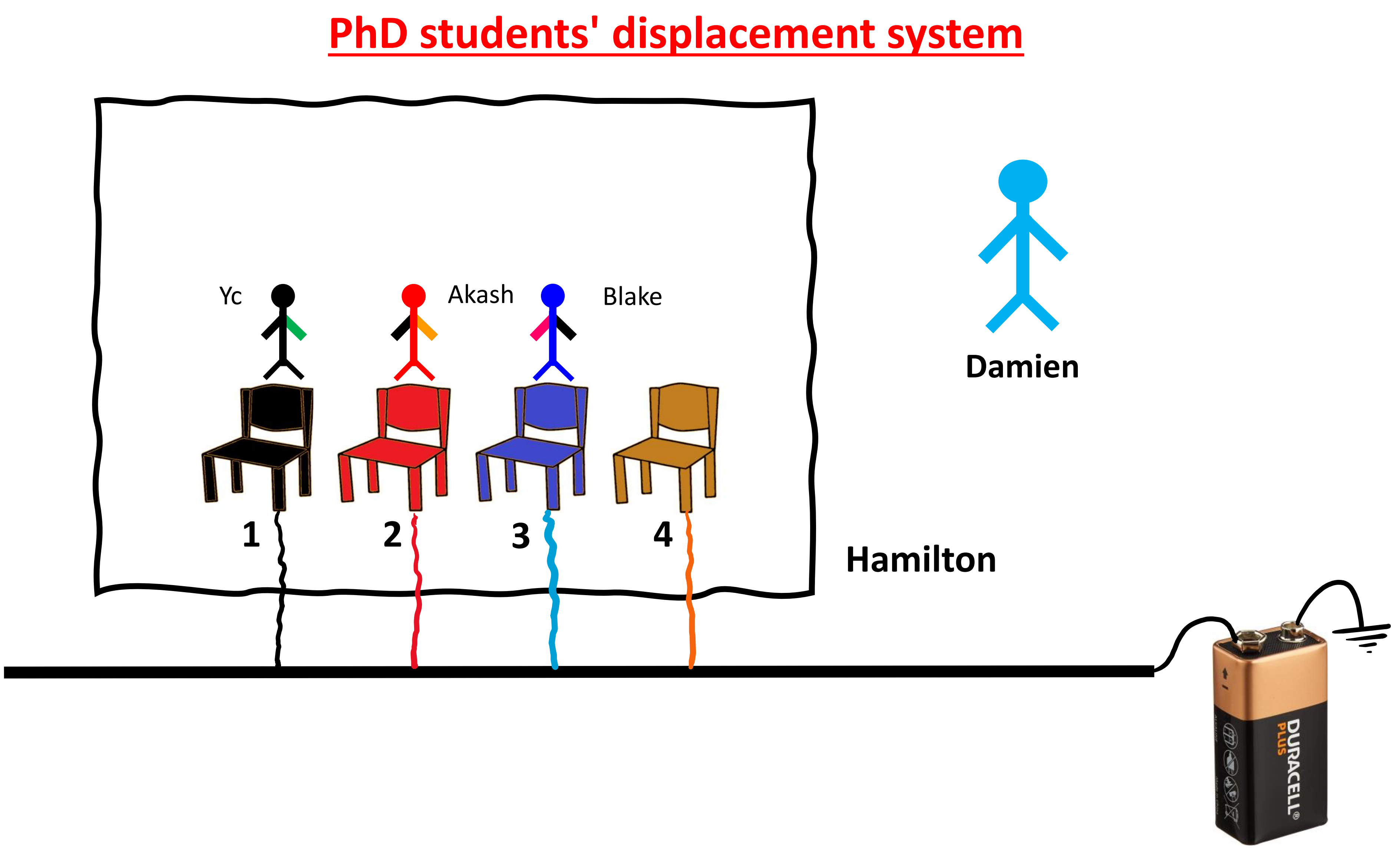
$$E(X) = \sum_{p \in X} \text{sit}(p) + \sum_{p_i, p_{i+1} \in X} \text{handshake}(p_i, p_{i+1}) + l * \text{sit\_convincing\_cost}.$$

configuration X of size l PhD students

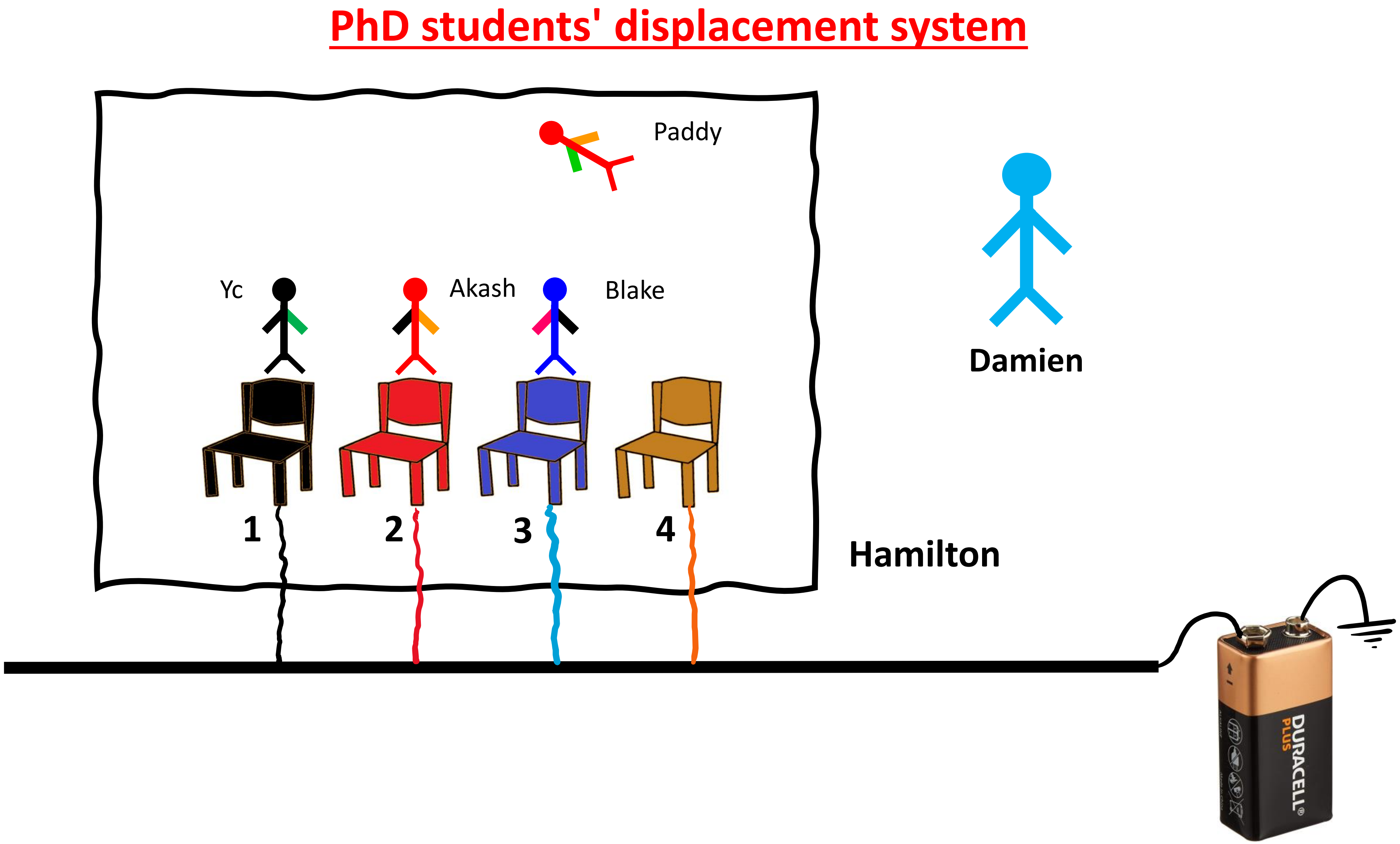
We further assume the following:

- $|\text{sit}(p)| > |\text{sit\_convincing\_cost}|$. (Damien always gains by convincing a PhD student to sit)

Built–in self improvement mechanism

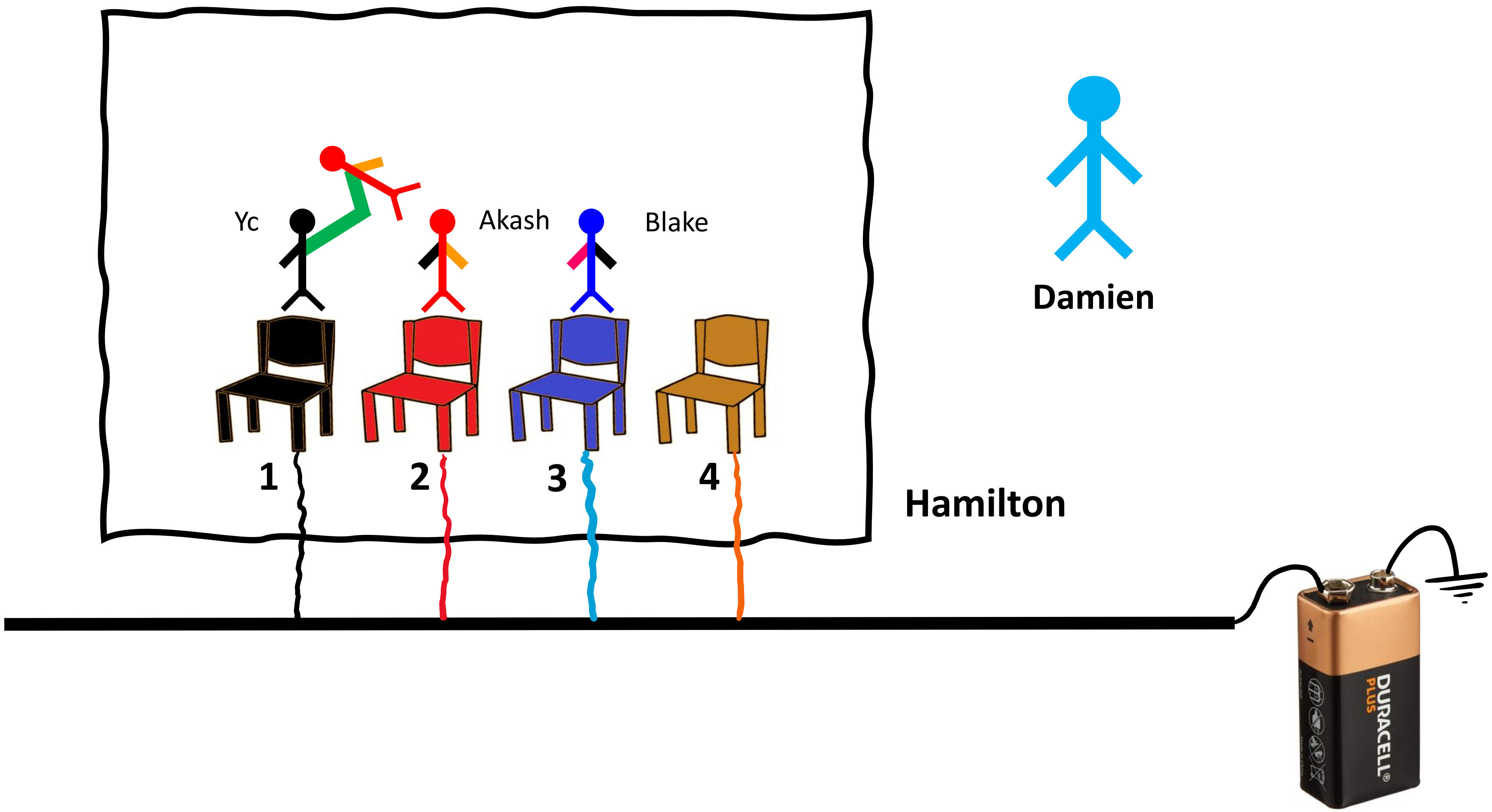


Built–in self improvement mechanism

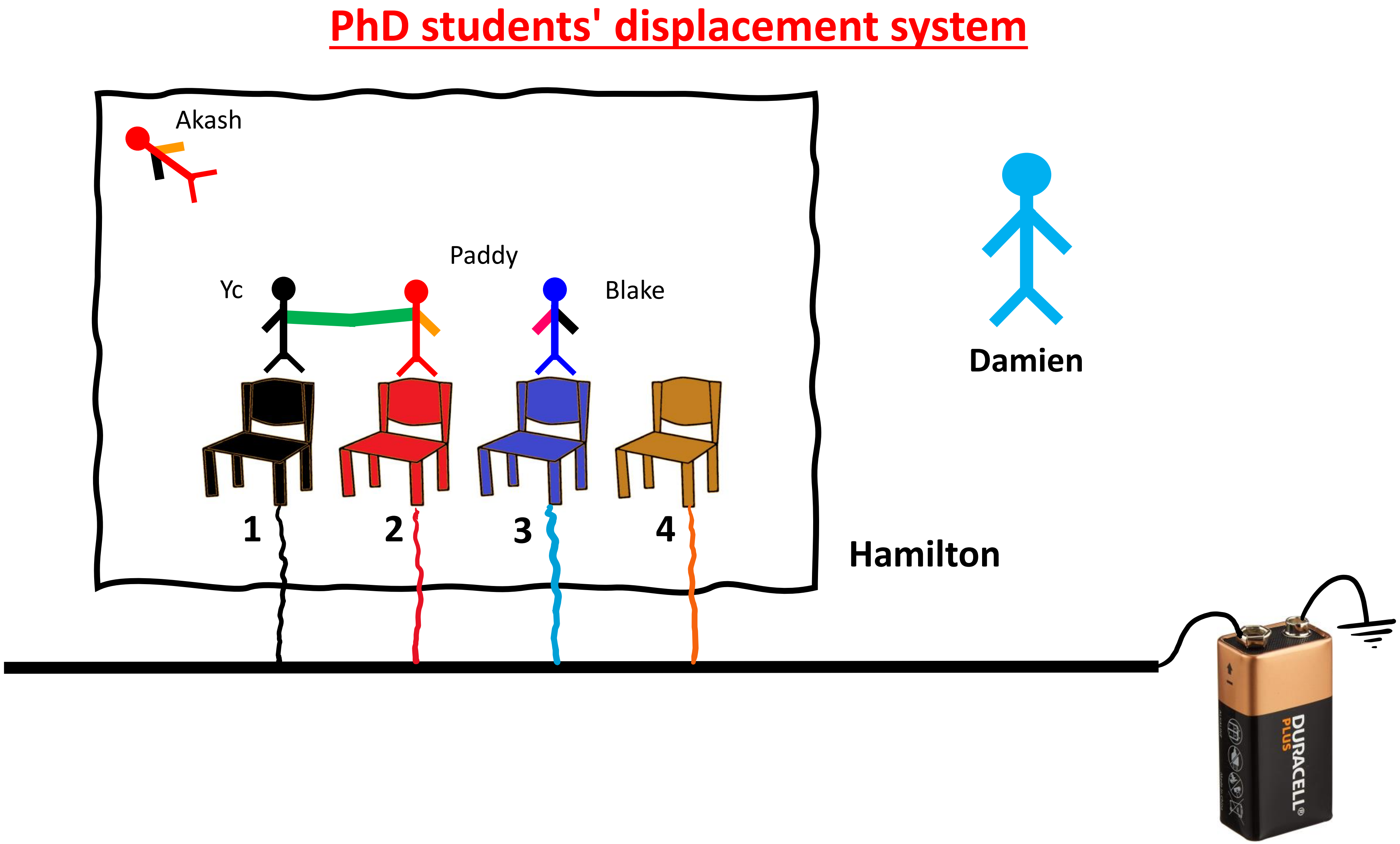


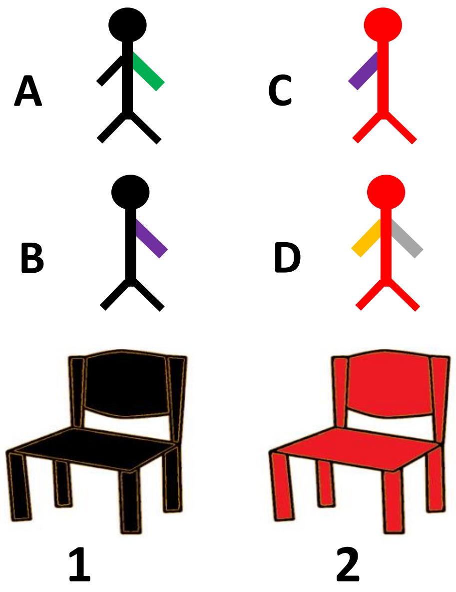
Built–in self improvement mechanism

PhD students' displacement system



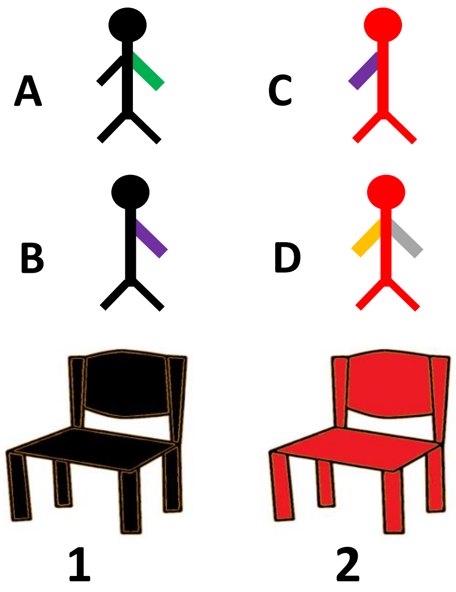
Built–in self improvement mechanism



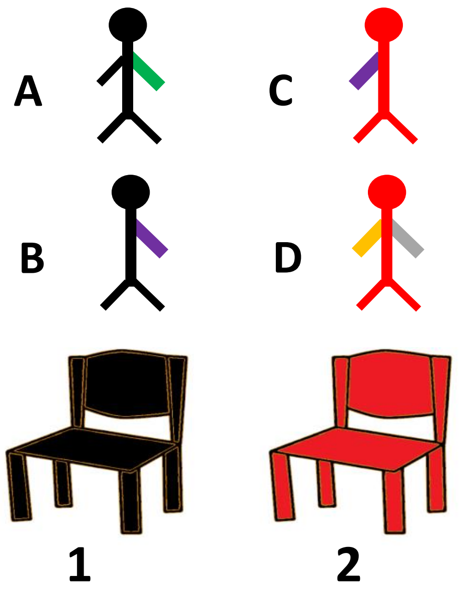


- sit(A) = sit(B) = 3
- sit(C) = sit(D) = 3
- handshake(B, C) = 1
- sit_convincing_cost = -1

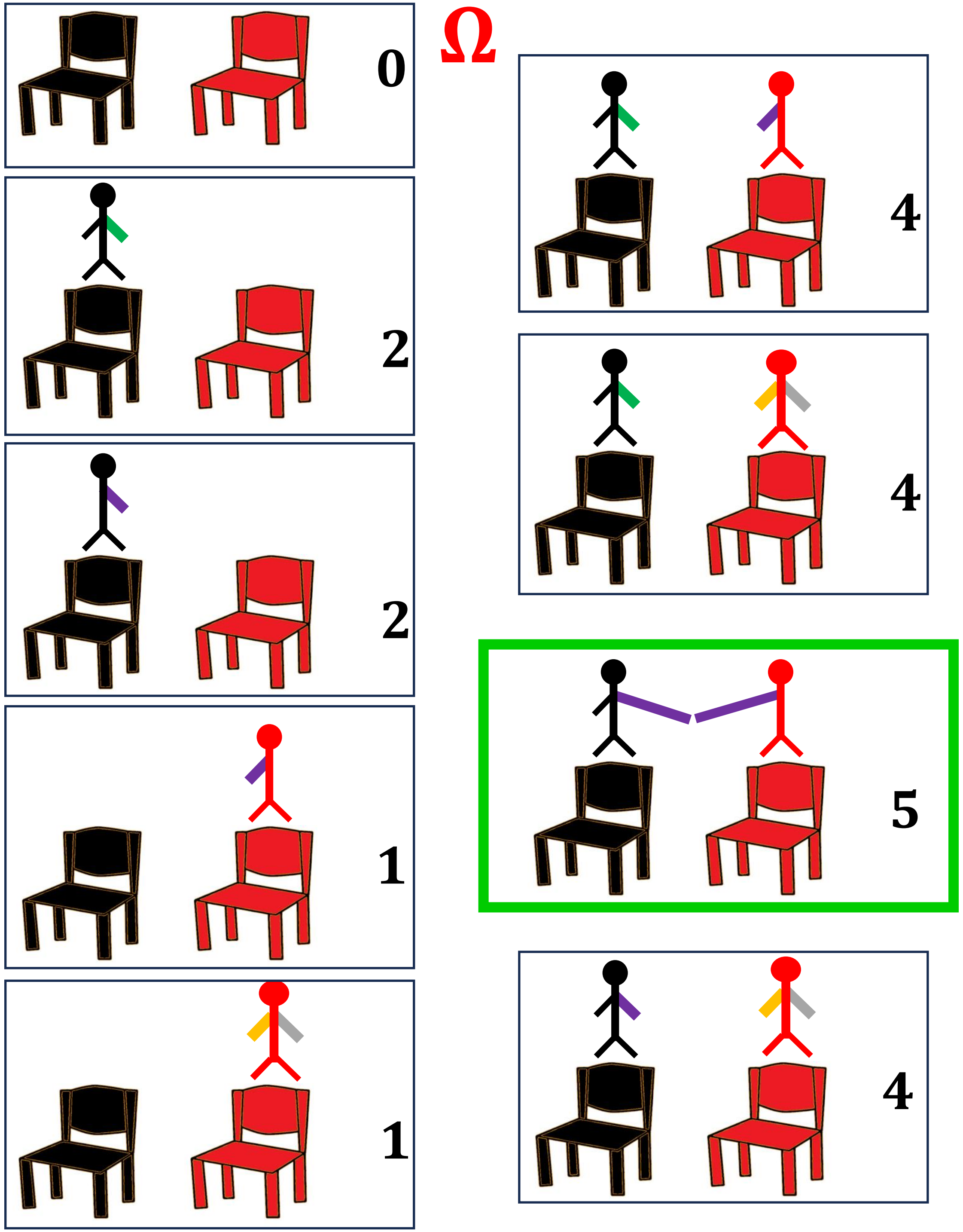
Ω

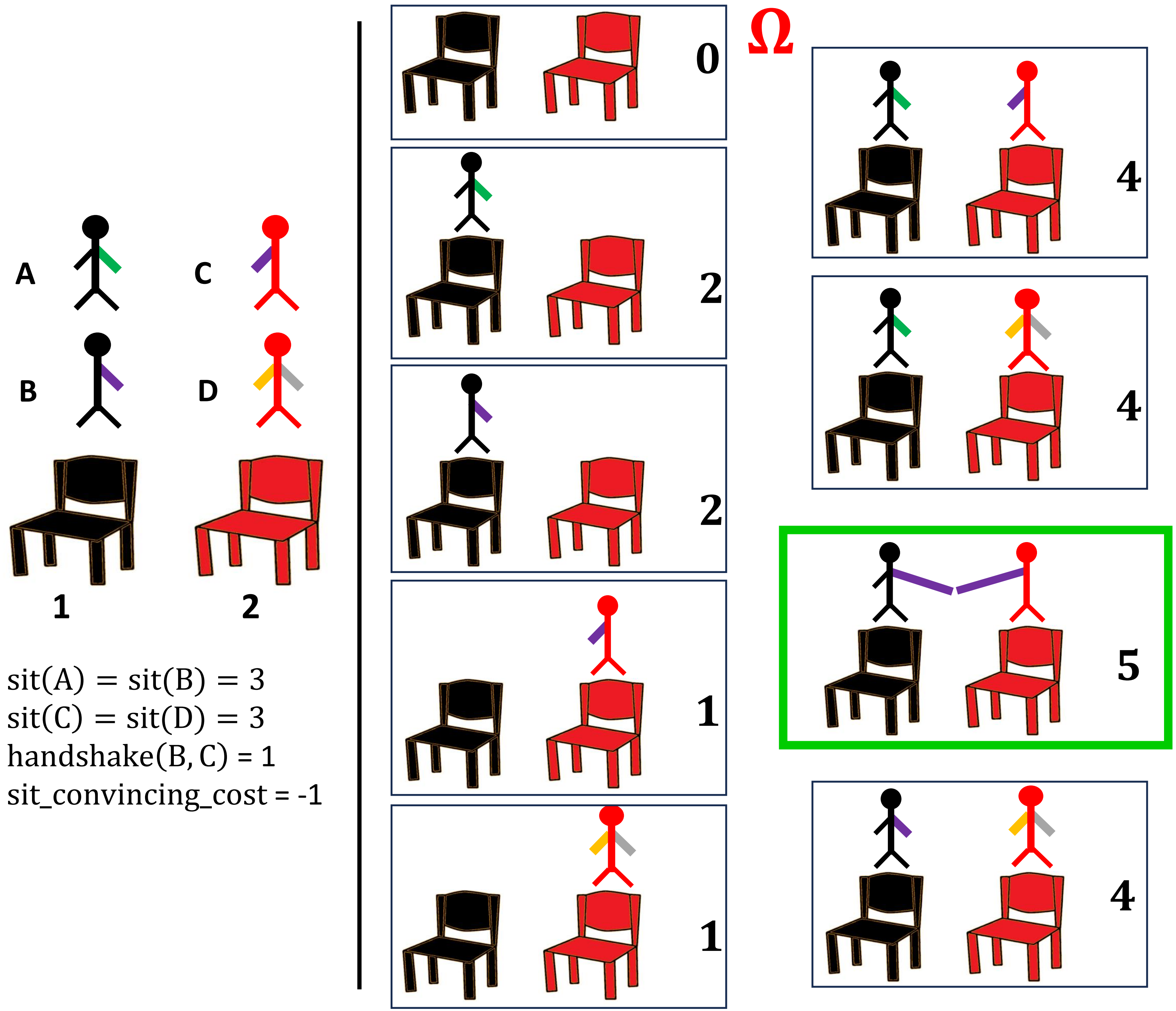


- sit(A) = sit(B) = 3
- sit(C) = sit(D) = 3
- handshake(B, C) = 1
- sit_convincing_cost = -1



- sit(A) = sit(B) = 3
- sit(C) = sit(D) = 3
- handshake(B, C) = 1
- sit_convincing_cost = -1

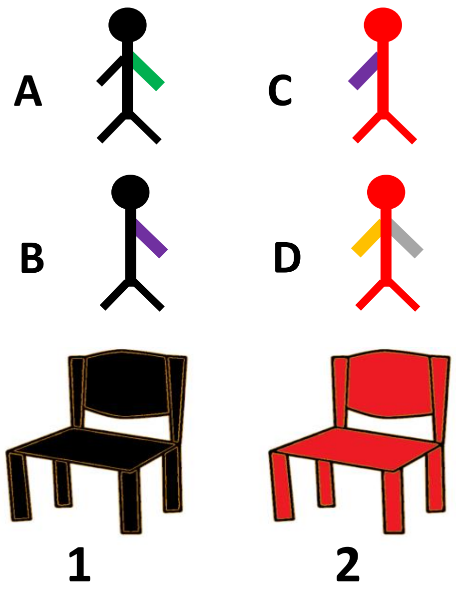




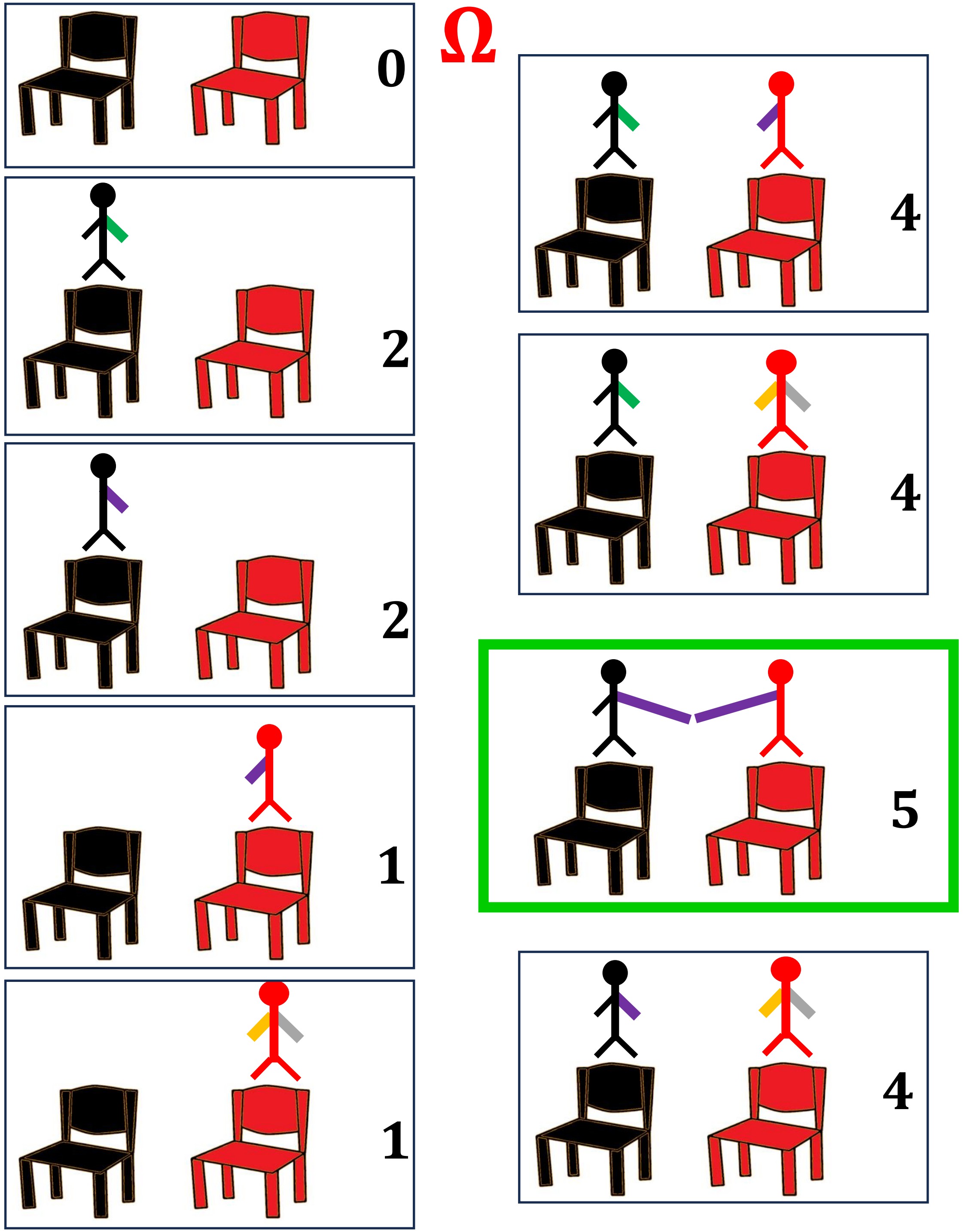
- $\text{sit}(A) = \text{sit}(B) = 3$
- $\text{sit}(C) = \text{sit}(D) = 3$
- $\text{handshake}(B, C)$ = 1
- sit_convincing_cost = -1

The Maximum Energy M

$$\boldsymbol{M} \coloneqq \max_{\boldsymbol{X} \in \boldsymbol{\Omega}} \{\boldsymbol{E}(\boldsymbol{X})\}$$

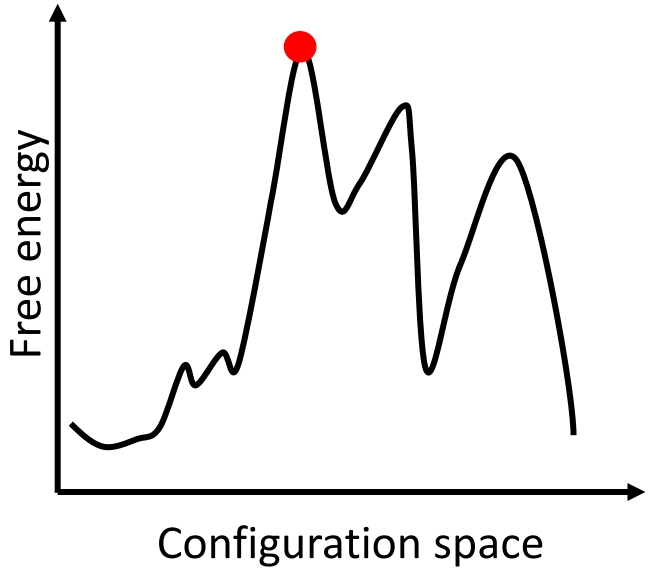


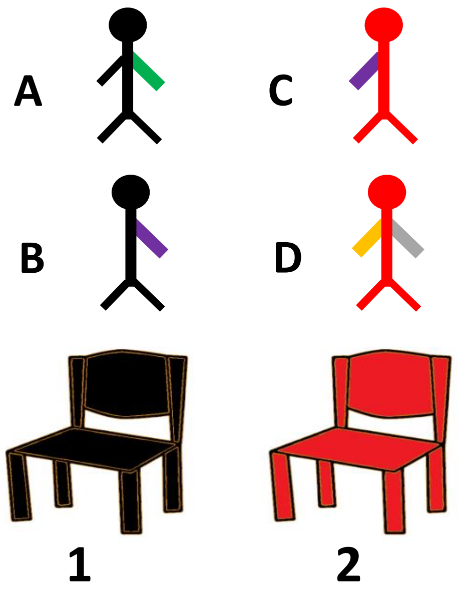
- $\text{sit}(A) = \text{sit}(B) = 3$
- $\text{sit}(C) = \text{sit}(D) = 3$
- handshake(B, C) = 1
- sit_convincing_cost = -1



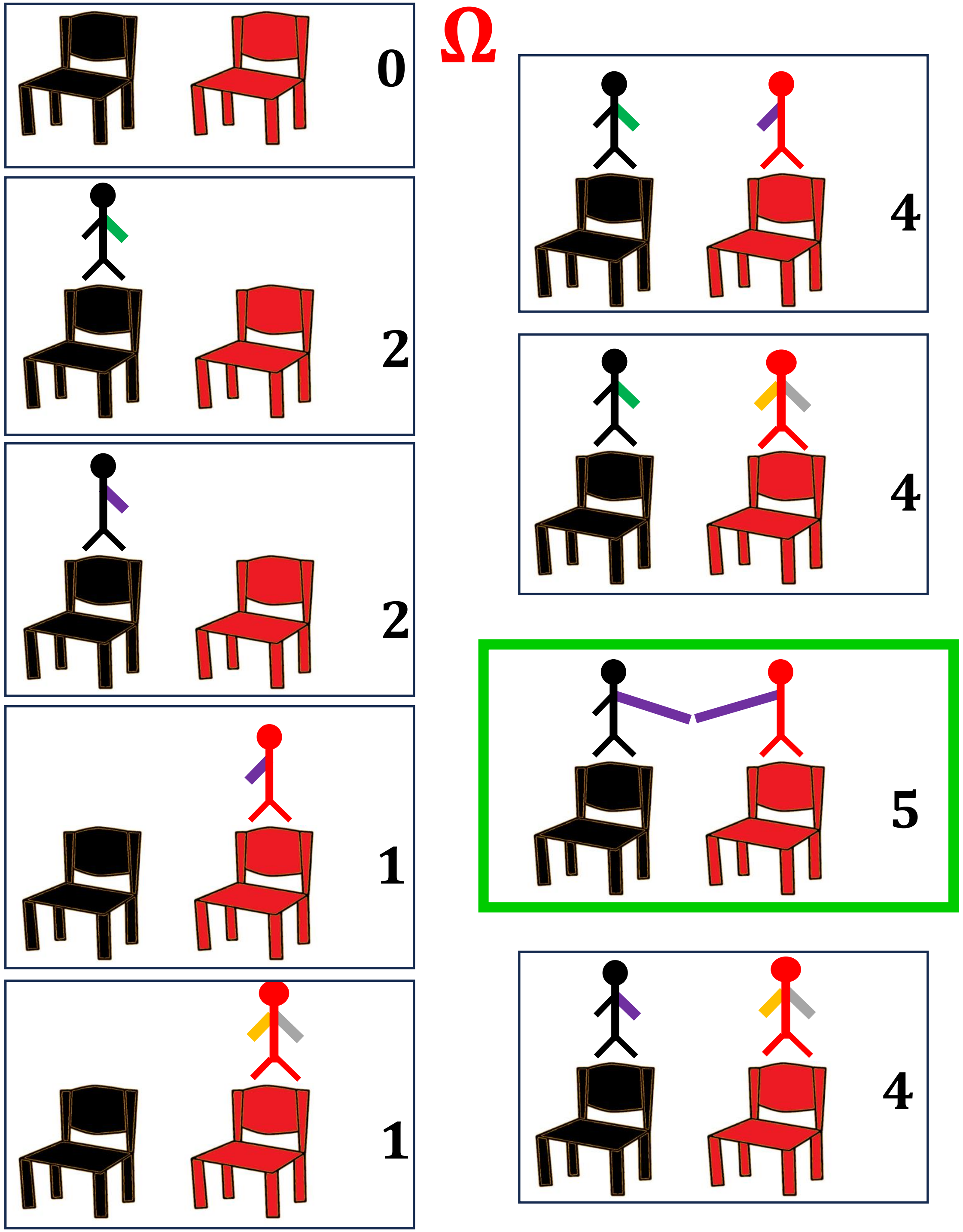
The Maximum Energy M

$$M := \max_{X \in \Omega} \{E(X)\}$$



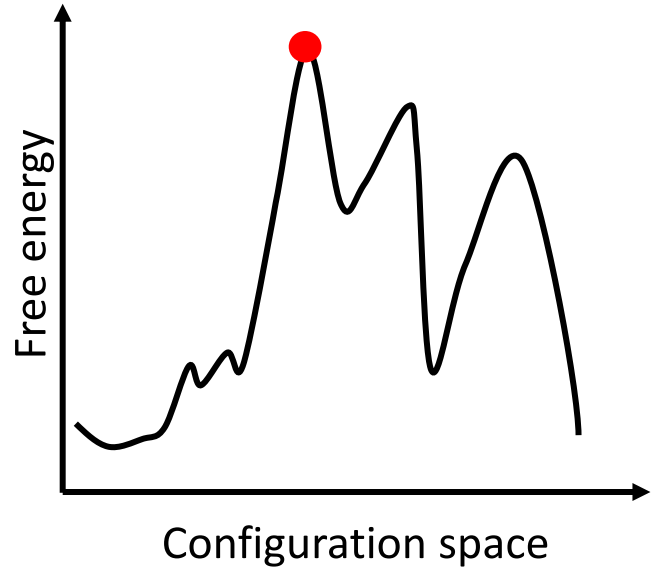


- $\mathrm{sit}(A) = \mathrm{sit}(B) = 3$
- $\mathrm{sit}(C) = \mathrm{sit}(D) = 3$
- $\mathrm{handshake}(B, C) = 1$
- sit_convincing_cost = -1



The Maximum Energy M

$$\boldsymbol{M} \coloneqq \max_{\boldsymbol{X} \in \boldsymbol{\Omega}} \{\boldsymbol{E}(\boldsymbol{X})\}$$



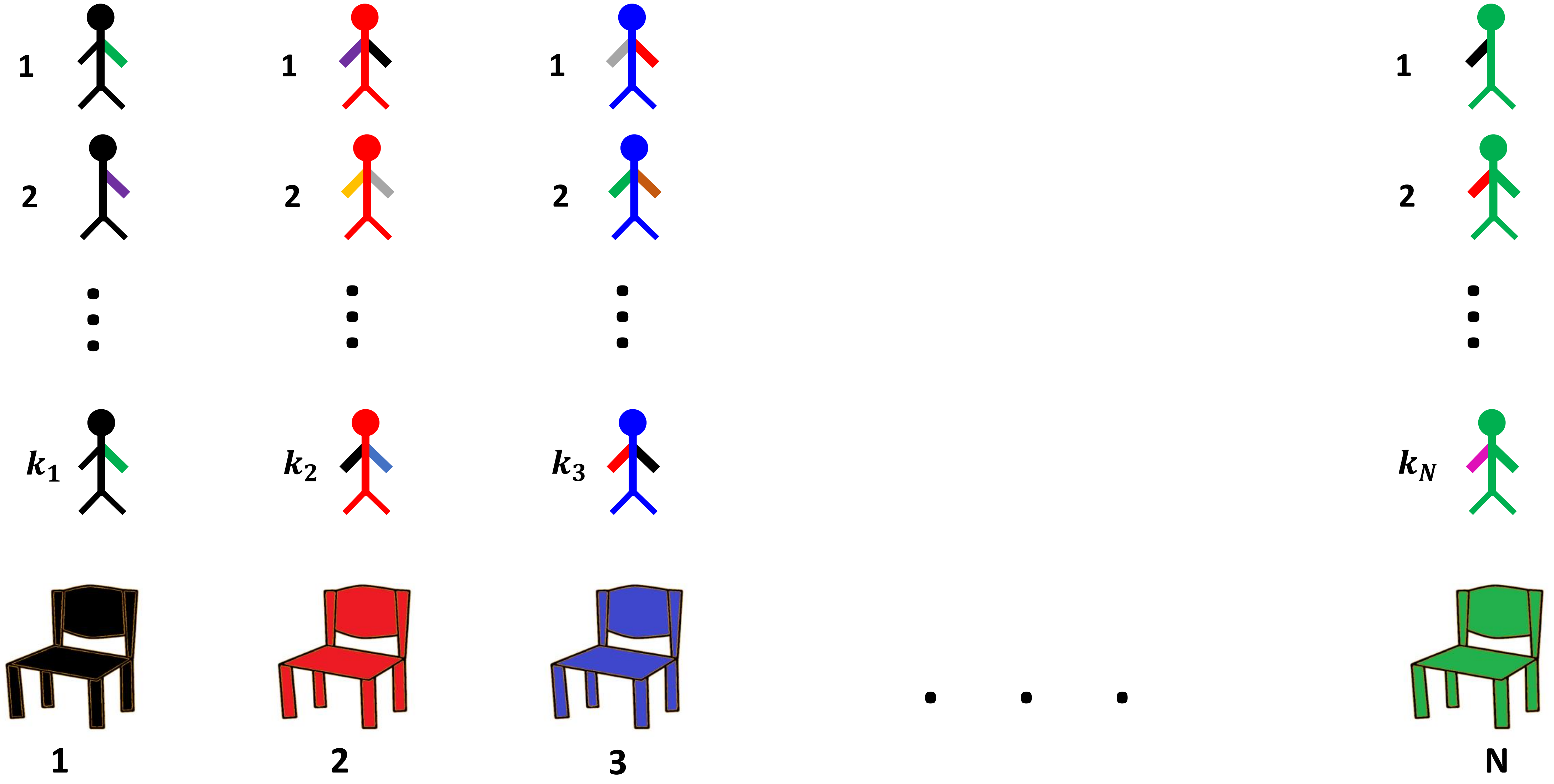
The Partition function Q

$$\boldsymbol{Q} \coloneqq \sum_{\boldsymbol{X} \in \boldsymbol{\Omega}} \boldsymbol{e}^{\frac{\boldsymbol{E}(\boldsymbol{X})}{\boldsymbol{c}}}$$

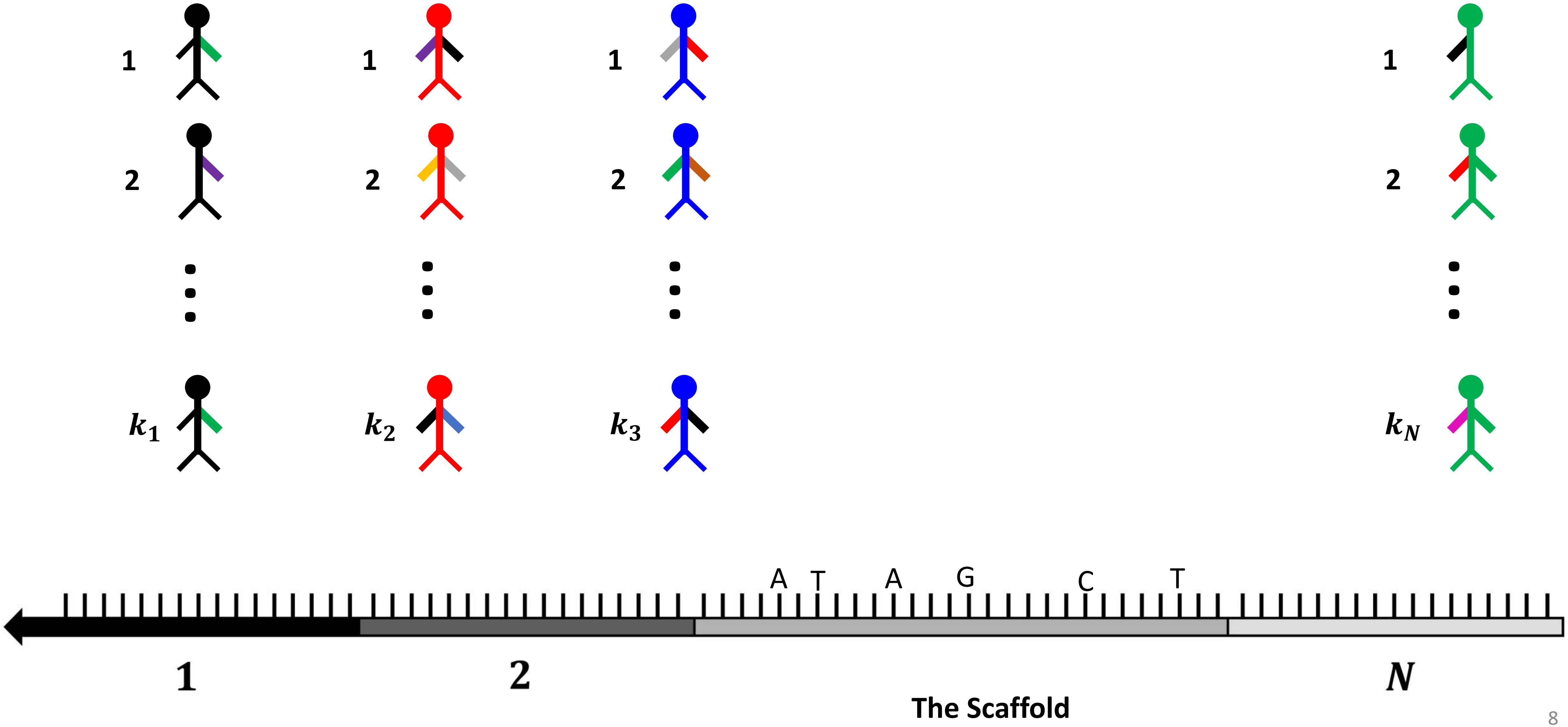
Where c is some specific constant

Why are we interested in this ?

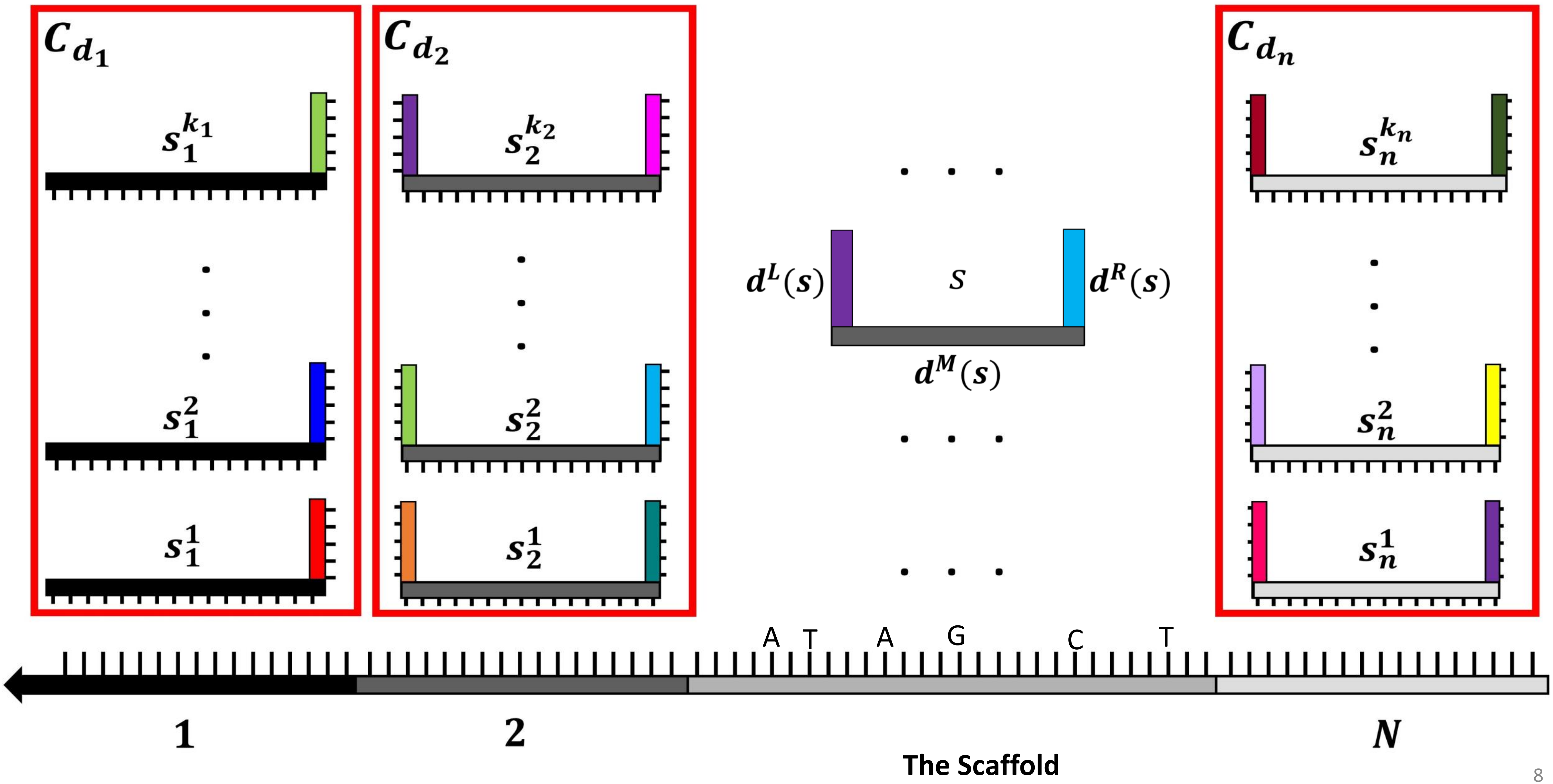
Why are we interested in this ?



1
2
k_1
1
2
k_2
1
2
k_3
1
2
k_N
1
2
3
N

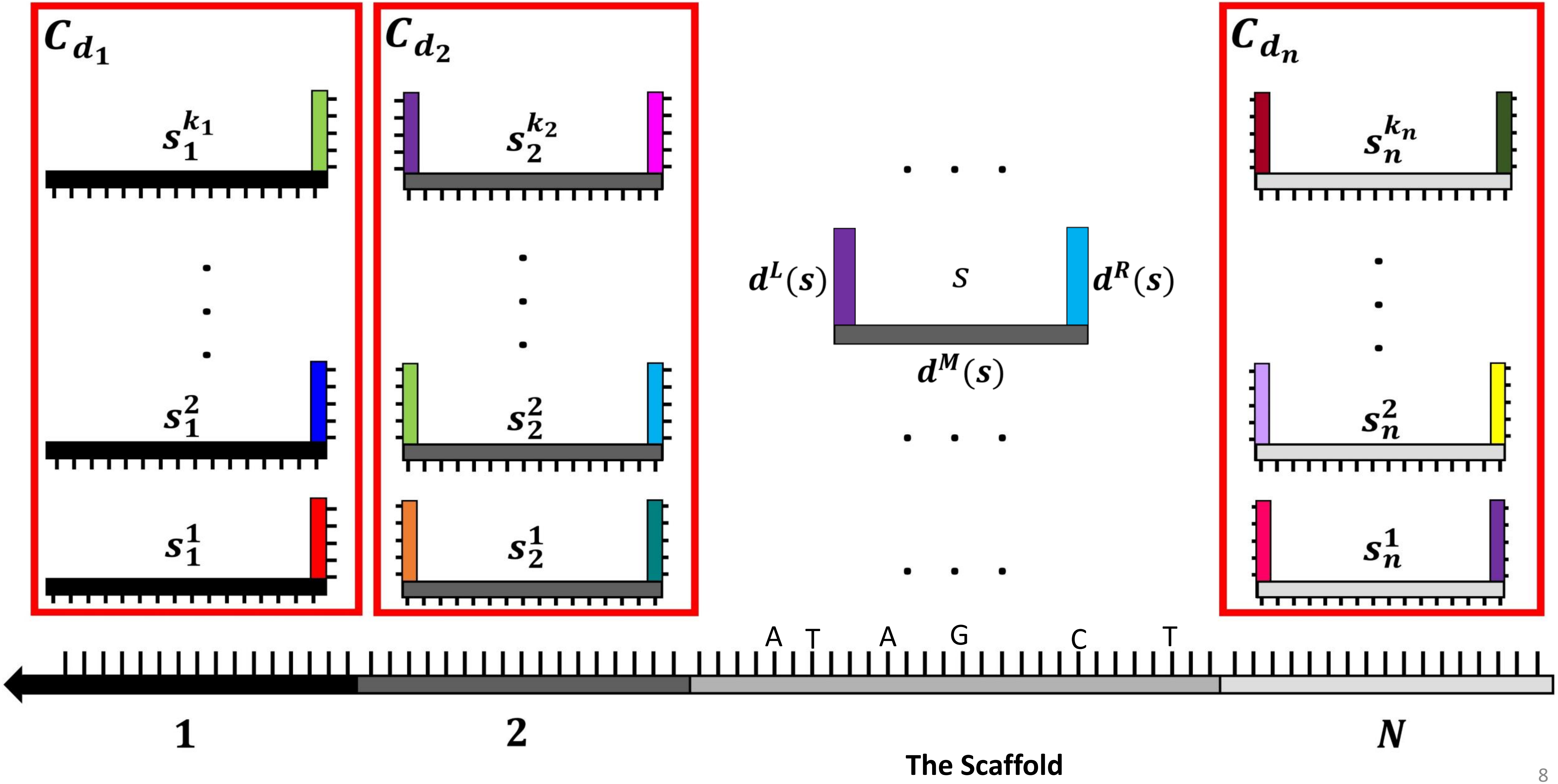


1
2
k_1
1
2
k_2
1
2
k_3
1
2
k_N
A T A G C T
1
2
N
The Scaffold



C_{d_1}
$s_1^{k_1}$
s_1^2
s_1^1
C_{d_2}
$s_2^{k_2}$
s_2^2
s_2^1
$d^L(s)$
s
$d^R(s)$
$d^M(s)$
C_{d_n}
$s_n^{k_n}$
s_n^2
s_n^1
A T A G C T
1
2
N
The Scaffold

This is 1D Scaffolded DNA Computer



Built–in algorithmic self correction mechanism

DNA strand displacement

Built–in algorithmic self correction mechanism

DNA strand displacement

1D Scaffolded DNA Computer Energy model and thermodynamic features

For any configuration X of size l :

Chair $\mathrm{E}(X) = \sum_{p \in X} \mathrm{sit(p)} + l * sitcost + \sum_{p_i, p_{i+1} \in X} \mathrm{handshake}(p_i, p_{i+1}).$

1D Scaffolded DNA Computer Energy model and thermodynamic features

For any configuration X of size l :

Chair $\mathrm{E}(X) = \sum_{p \in X} \mathrm{sit(p)} + l * sitcost + \sum_{p_i, p_{i+1} \in X} \mathrm{handshake}(p_i, p_{i+1}).$

↕ ↕ ↕

DNA $$\Delta G^{\mathcal{S}}(X) = \sum_{s \in X} \Delta G(d^M(s)) + l \cdot \Delta G^{\text{assoc}} + \sum_{s_i, s_{i+1} \in X} \Delta G(d^R(s_i), d^L(s_{i+1})).$$

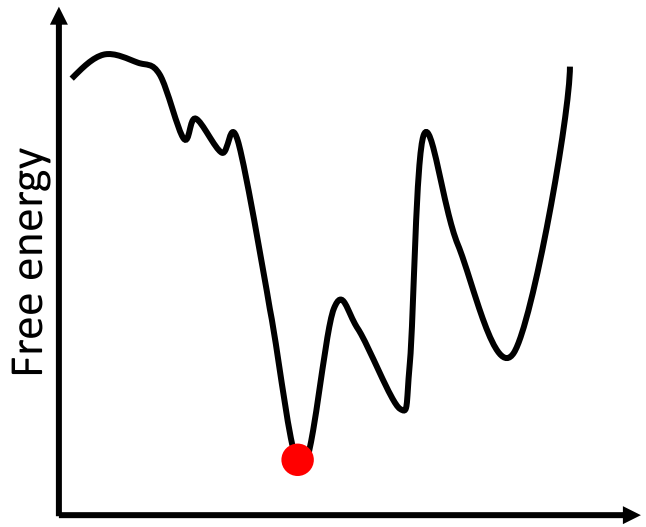
1D Scaffolded DNA Computer Energy model and thermodynamic features

For any configuration X of size l :

Chair $\mathrm{E}(X) = \sum_{p \in X} \mathrm{sit}(\mathrm{p}) + l * sitcost + \sum_{p_i, p_{i+1} \in X} \mathrm{handshake}(p_i, p_{i+1}).$

↕ ↕ ↕

DNA $\Delta G^{\mathcal{S}}(X) = \sum_{s \in X} \Delta G(d^M(s)) + l \cdot \Delta G^{\mathrm{assoc}} + \sum_{s_i, s_{i+1} \in X} \Delta G(d^R(s_i), d^L(s_{i+1})).$



$$\mathrm{MFE} = \min_{S \in \Omega} \Delta G(S)$$

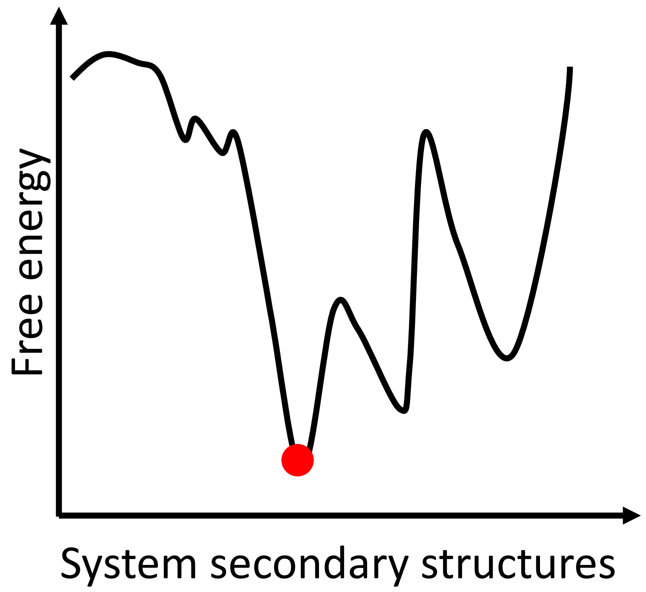
Minimum Free Energy

1D Scaffolded DNA Computer Energy model and thermodynamic features

For any configuration X of size l :

Chair $$\mathrm{E}(X) = \sum_{p \in X} \mathrm{sit(p)} + l * sitcost + \sum_{p_i, p_{i+1} \in X} \mathrm{handshake}(p_i, p_{i+1}).$$

DNA $$\Delta G^{\mathcal{S}}(X) = \sum_{s \in X} \Delta G(d^M(s)) + l \cdot \Delta G^{\text{assoc}} + \sum_{s_i, s_{i+1} \in X} \Delta G(d^R(s_i), d^L(s_{i+1})).$$



$$\mathrm{MFE} = \min_{S \in \Omega} \Delta G(S)$$

Minimum Free Energy

Boltzmann weighted sum

$$Q = \sum_{S \in \Omega} e^{-\Delta G(S)/k_{\mathrm{B}}T}$$

Partition Function

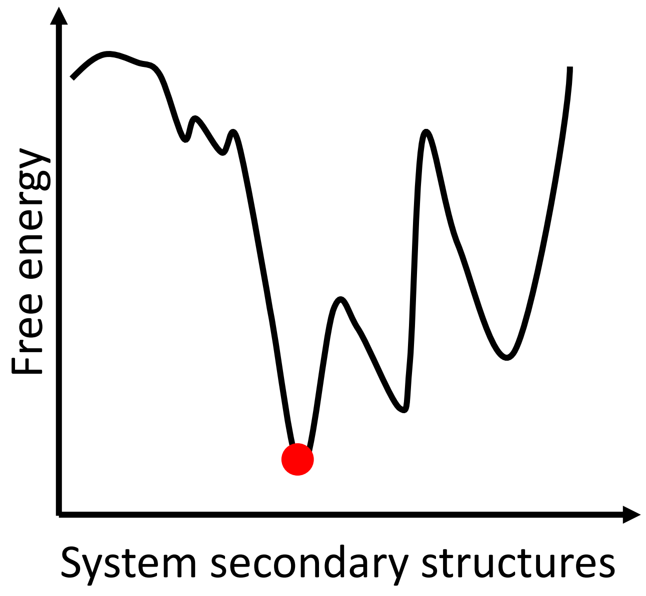
1D Scaffolded DNA Computer Energy model and thermodynamic features

For any configuration X of size l :

Chair $\mathrm{E}(X) = \sum_{p \in X} \mathrm{sit(p)} + l * sitcost + \sum_{p_i, p_{i+1} \in X} \mathrm{handshake}(p_i, p_{i+1}).$

$\updownarrow$ $\updownarrow$ $\updownarrow$

DNA $\Delta G^{\mathcal{S}}(X) = \sum_{s \in X} \Delta G(d^M(s)) + l \cdot \Delta G^{\mathrm{assoc}} + \sum_{s_i, s_{i+1} \in X} \Delta G(d^R(s_i), d^L(s_{i+1})).$



$$\mathrm{MFE} = \min_{S \in \Omega} \Delta G(S)$$

Minimum Free Energy

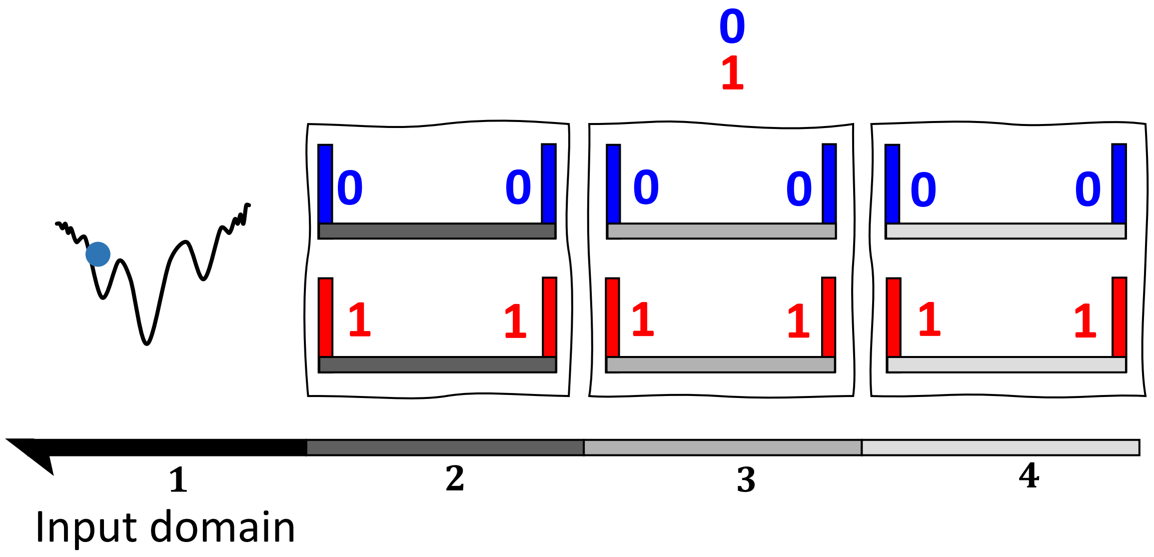
Boltzmann weighted sum

$$Q = \sum_{S \in \Omega} e^{-\Delta G(S)/k_{\mathrm{B}}T}$$

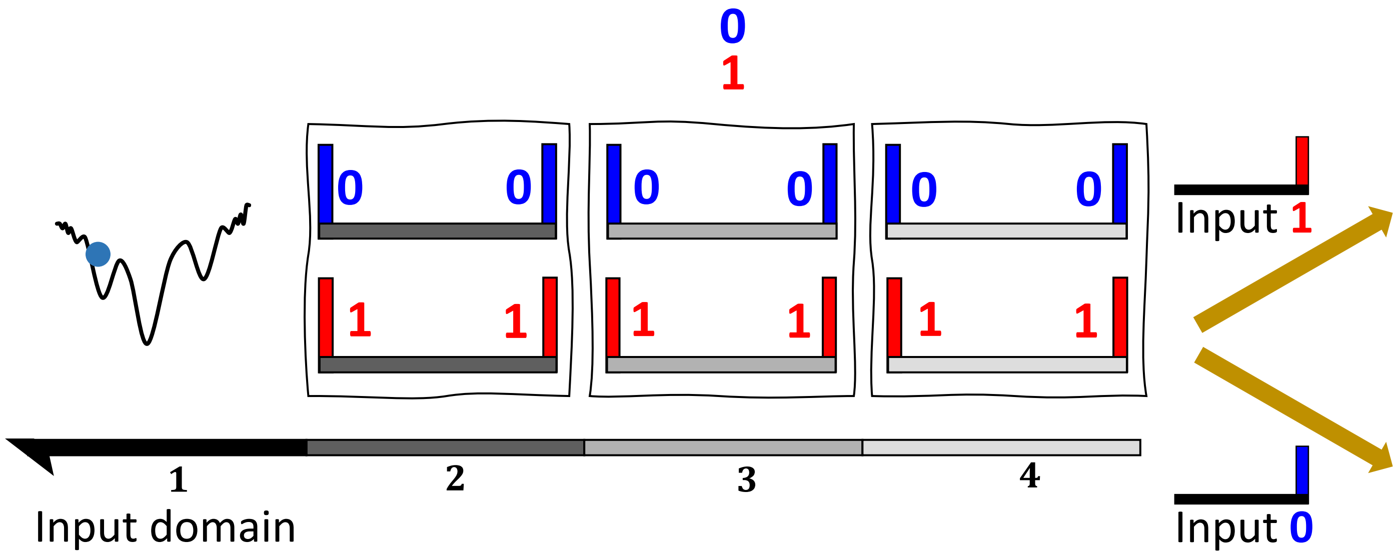
Partition Function

$$\Rightarrow \quad \Pr[S] = \frac{e^{-\Delta G(S)/k_{\mathrm{B}}T}}{Q}$$

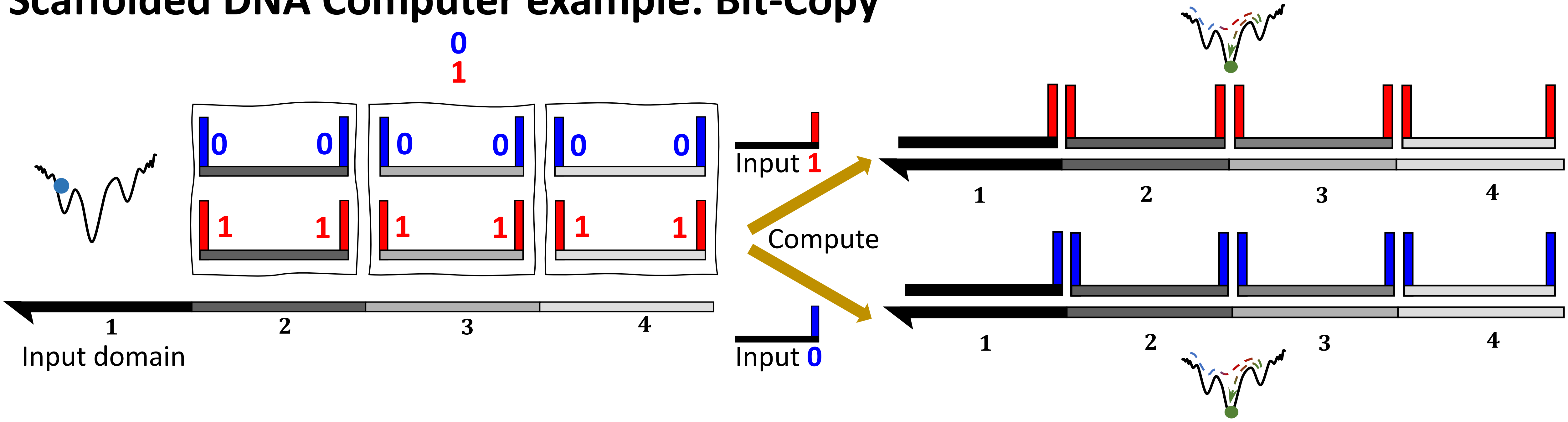
Scaffolded DNA Computer example: Bit-Copy



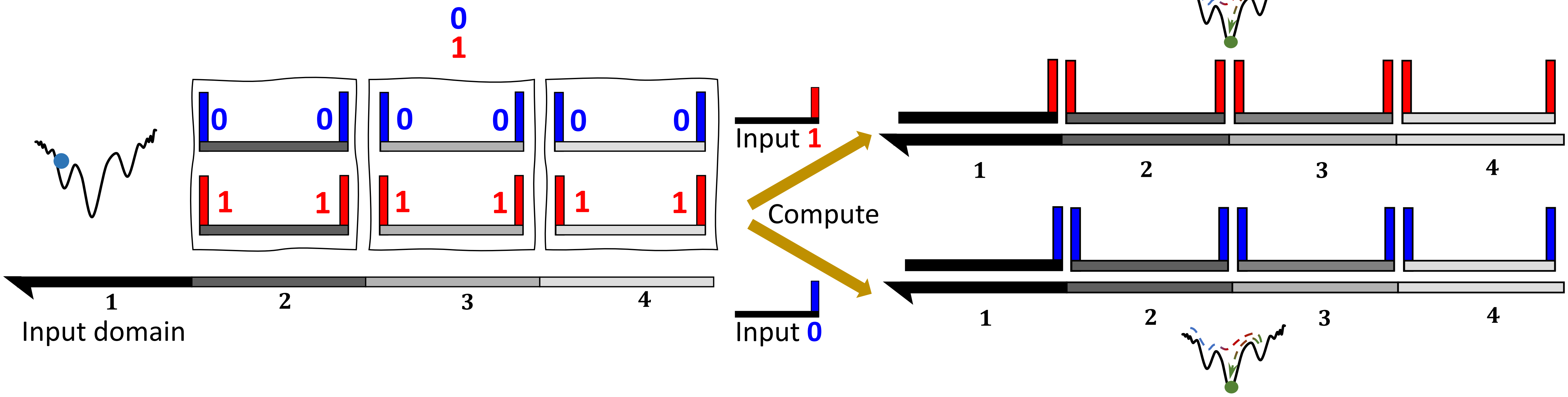
Scaffolded DNA Computer example: Bit-Copy



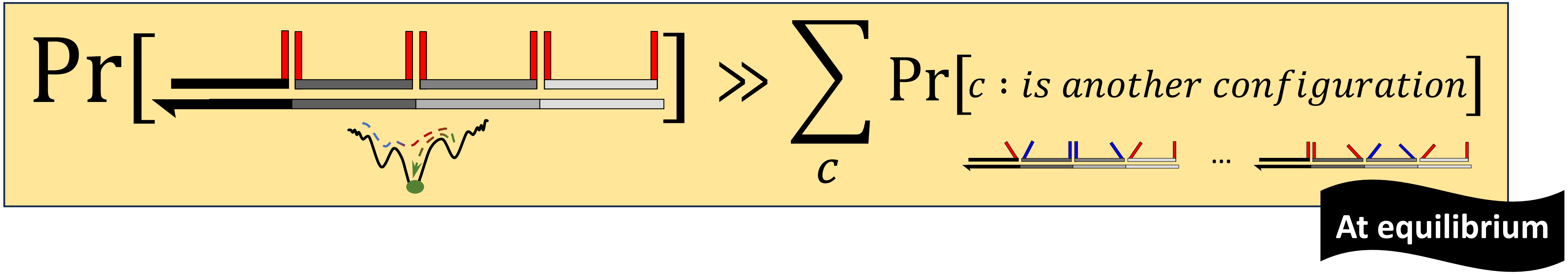
Scaffolded DNA Computer example: Bit-Copy



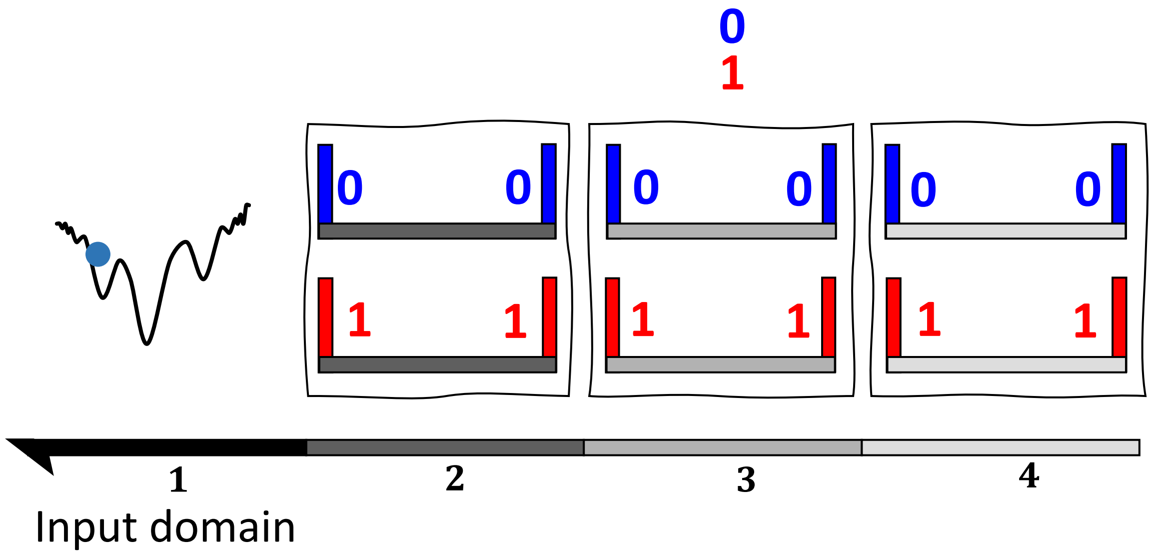
Scaffolded DNA Computer example: Bit-Copy

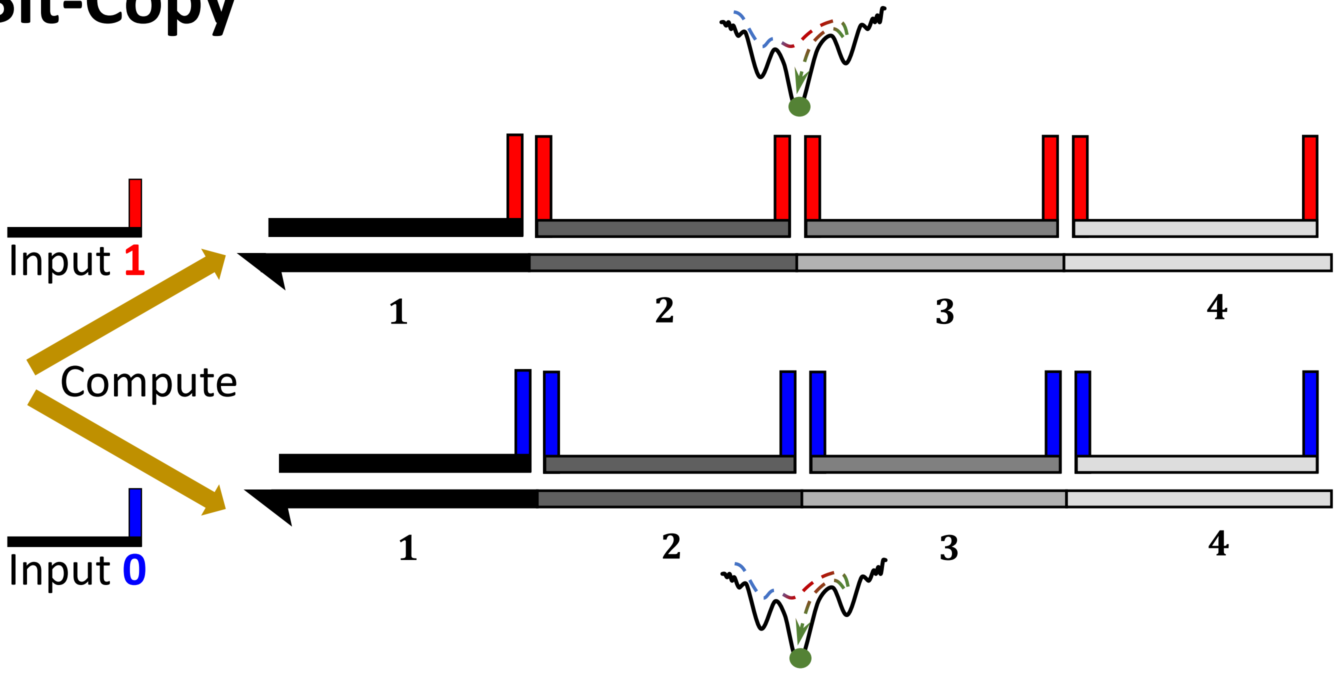


GOAL

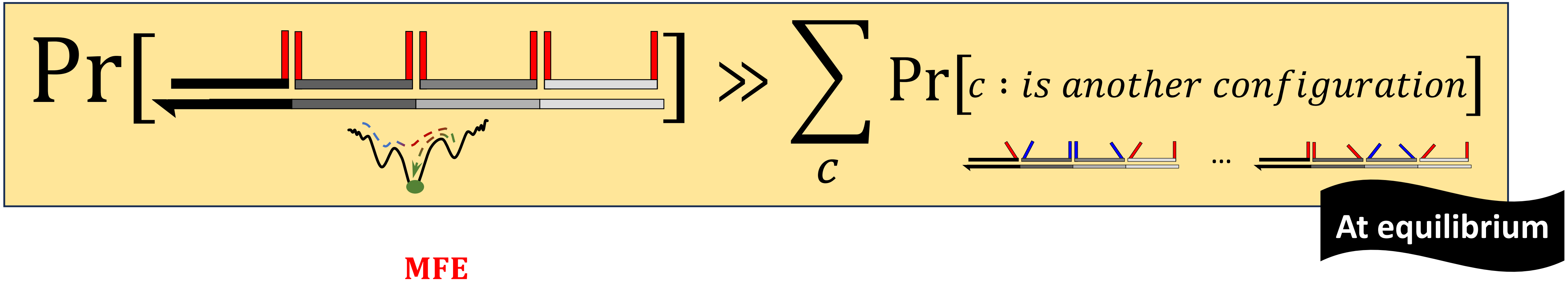


Scaffolded DNA Computer example: Bit-Copy





GOAL



MFE

$$\Pr[\text{target}] = \frac{e^{-\Delta G(\text{target})\,/k_{\mathrm{B}}T}}{Q}$$

Computational complexity of Minimum Free Energy and the Partition Function

Input Type	MFE	Partition Function
Single Strand	$O(n^3)$	$O(n^3)$
Multiple Strands, Bounded ($\leq s$)	?	$O(n^3)(s-1)!$
Multiple Strands, Unbounded	$NP - \text{Complete}$	?

n bases, s strands

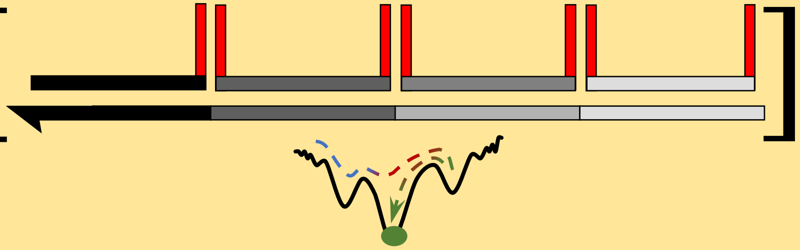
Computational complexity of Minimum Free Energy and the Partition Function

Input Type	MFE	Partition Function
Single Strand	$O(n^3)$	$O(n^3)$
Multiple Strands, Bounded ($\leq s$)	?	$O(n^3)(s-1)!$
Multiple Strands, Unbounded	$NP-\text{Complete}$	?

n bases, s strands

GOAL

At equilibrium

Pr[] $\gg \sum_{c} \Pr[c : \textit{is another configuration}]$

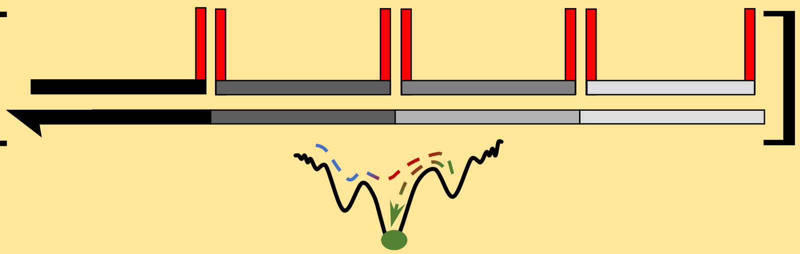
Computational complexity of Minimum Free Energy and the Partition Function

Input Type	MFE	Partition Function
Single Strand	$O(n^3)$	$O(n^3)$
Multiple Strands, Bounded ($\leq s$)	?	$O(n^3)(s-1)!$
Multiple Strands, Unbounded	$NP -$ Complete	?

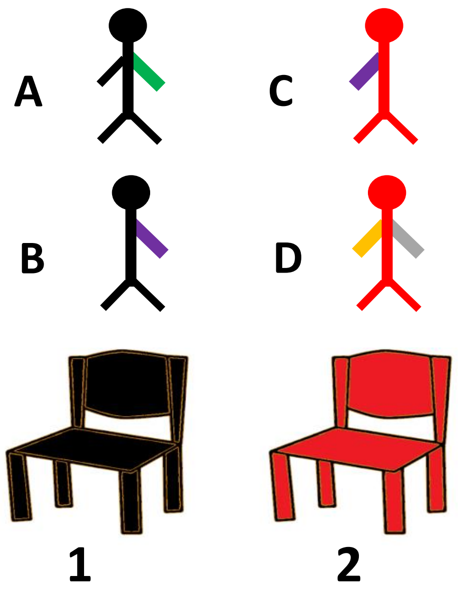
n bases, s strands

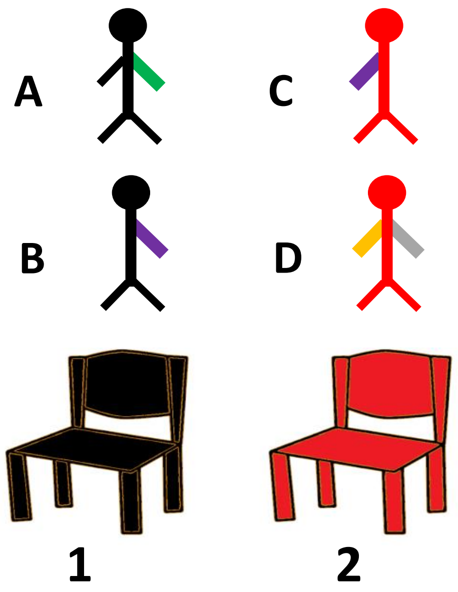
GOAL

At equilibrium

$\Pr[$  $] \gg \sum_{c} \Pr[c : is\ another\ configuration]$

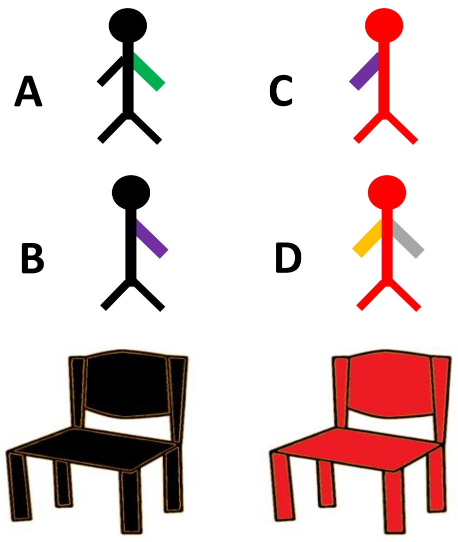
Efficiently


A
C
B
D
1
2



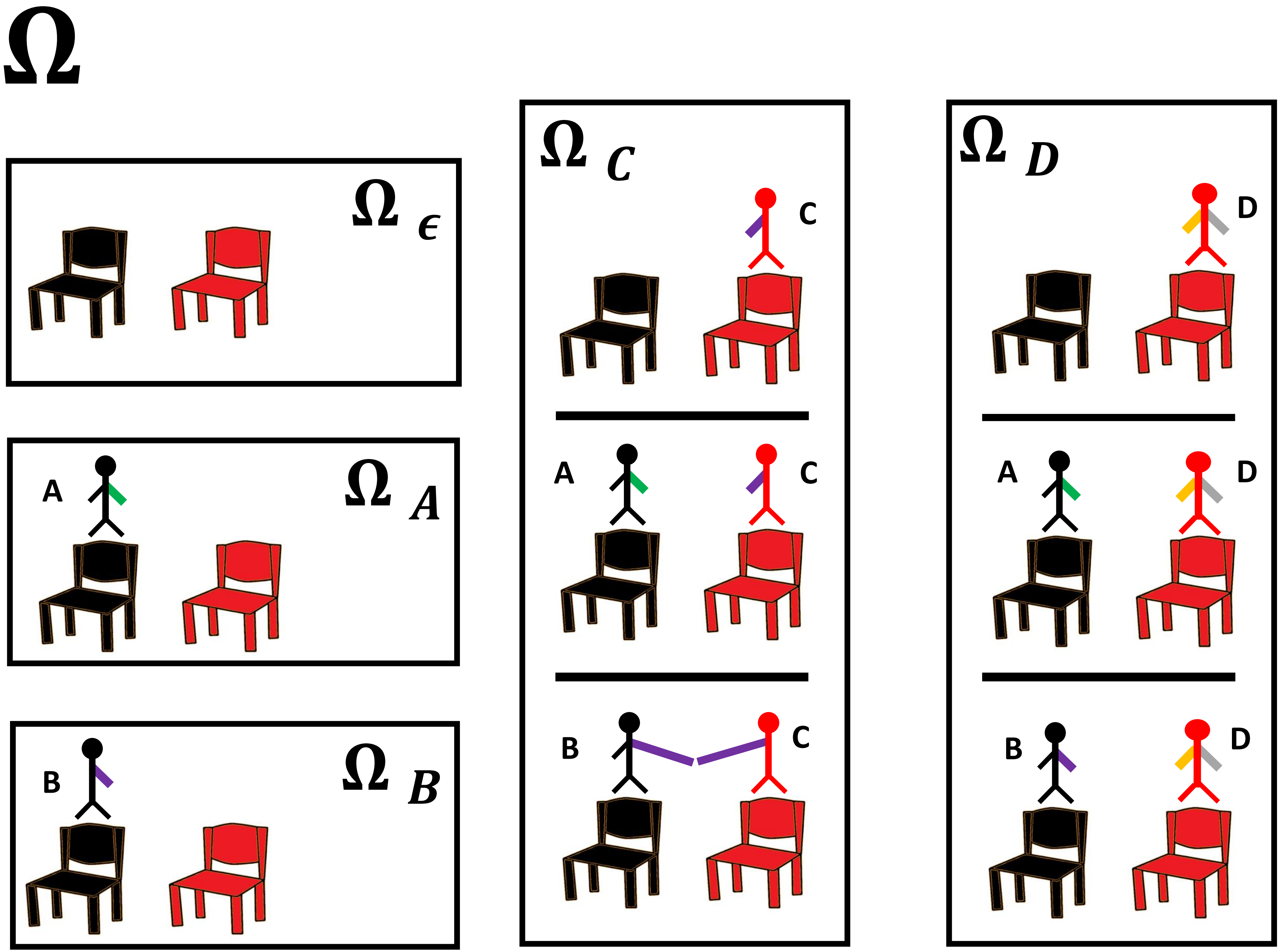
$$|\Omega| = (\boldsymbol{k} + \mathbf{1})^{\boldsymbol{N}} = \mathbf{9}$$

$$Q = \sum_{p \in \boldsymbol{\Omega}} e^{-\Delta G(p)/k_{\mathrm{B}}T}$$

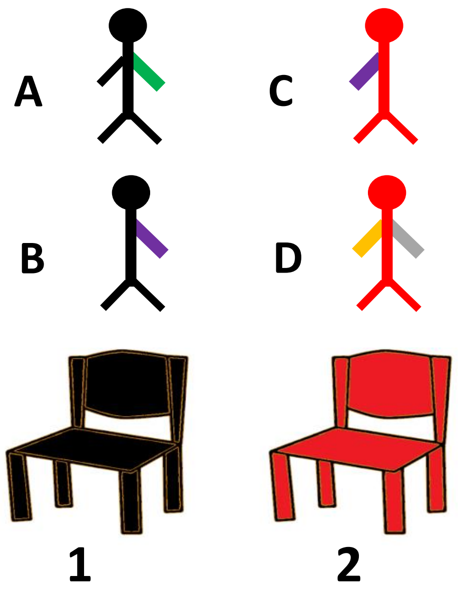


$$|\Omega| = (\boldsymbol{k}+\mathbf{1})^{N} = \mathbf{9}$$

$$Q = \sum_{p \in \boldsymbol{\Omega}} e^{-\Delta G(p)/k_{\mathrm{B}}T}$$

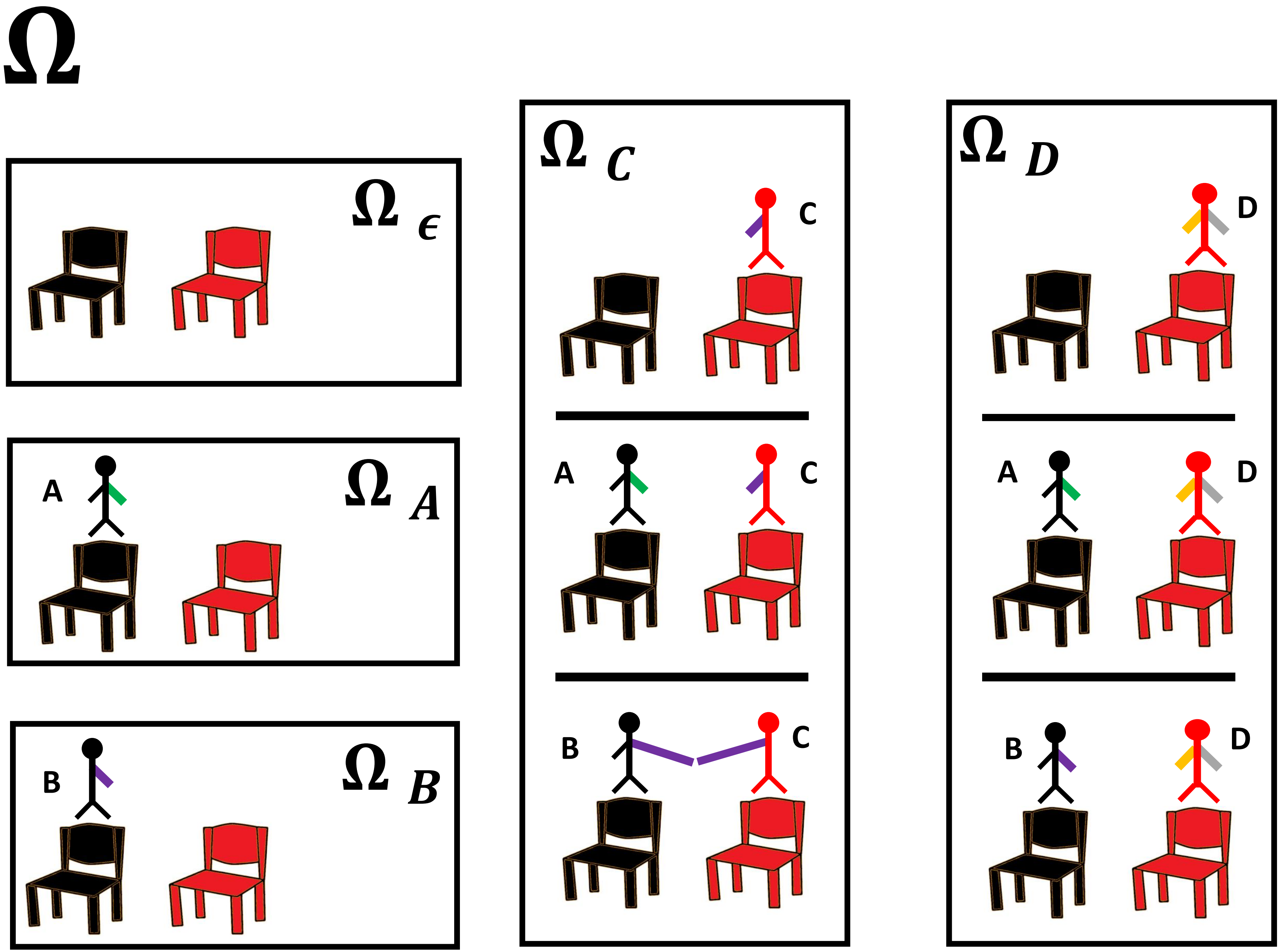


Is there recursive way to build these classes from themselves?

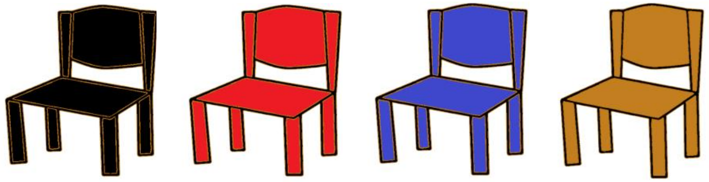


$$|\Omega| = (\boldsymbol{k}+\mathbf{1})^{N} = \mathbf{9}$$

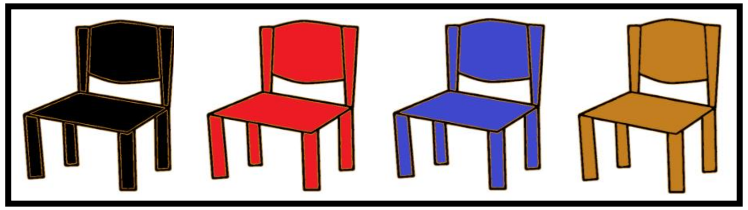
$$Q = \sum_{p\in\boldsymbol{\Omega}} e^{-\Delta G(p)/k_{\mathrm{B}}T}$$



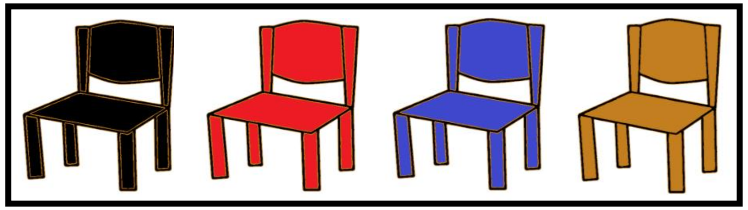
0



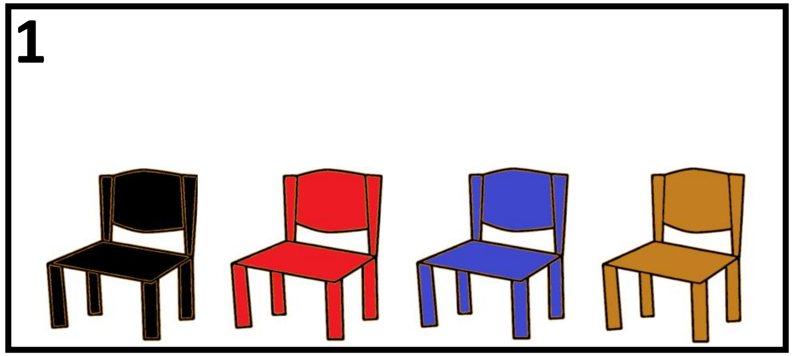
0

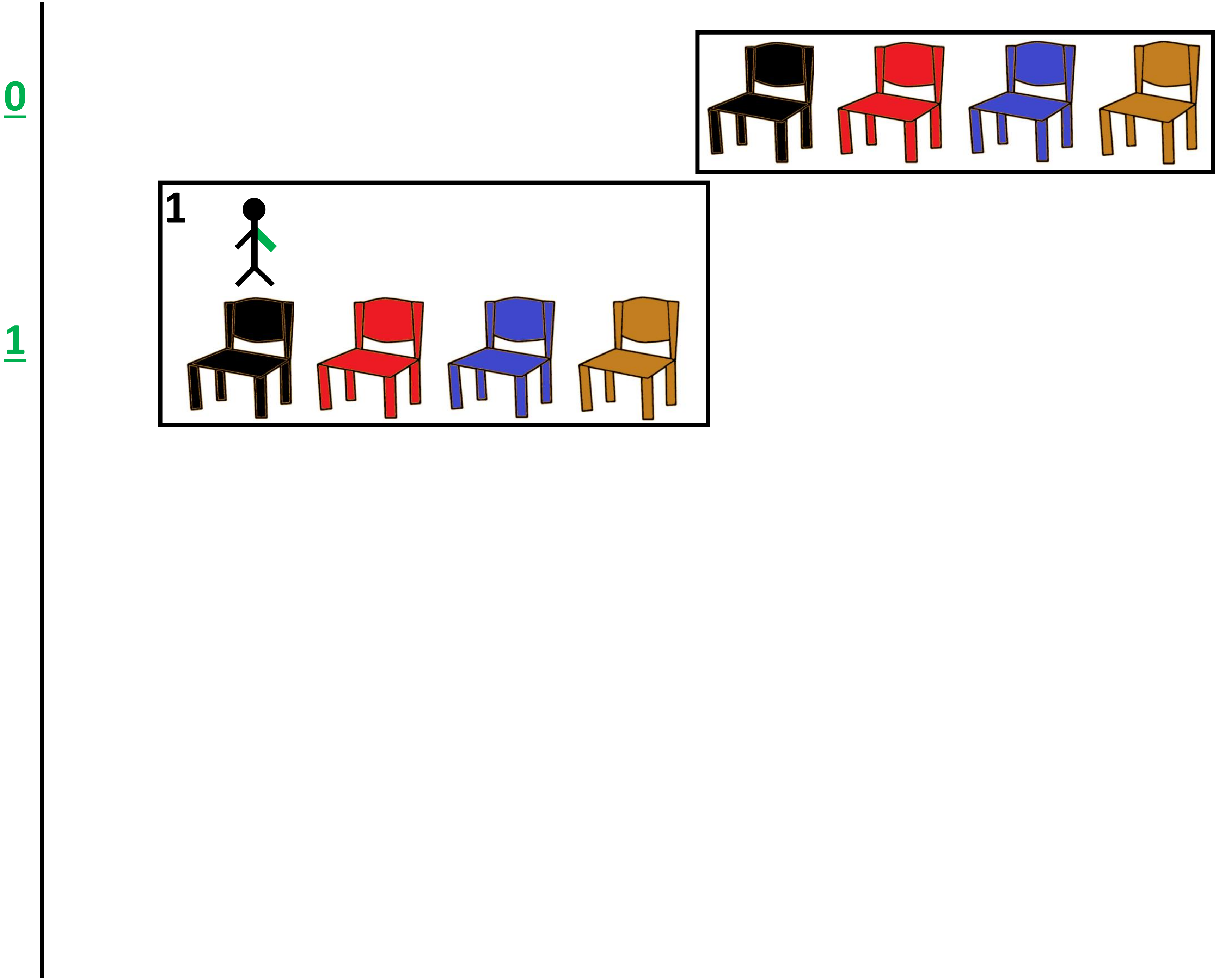


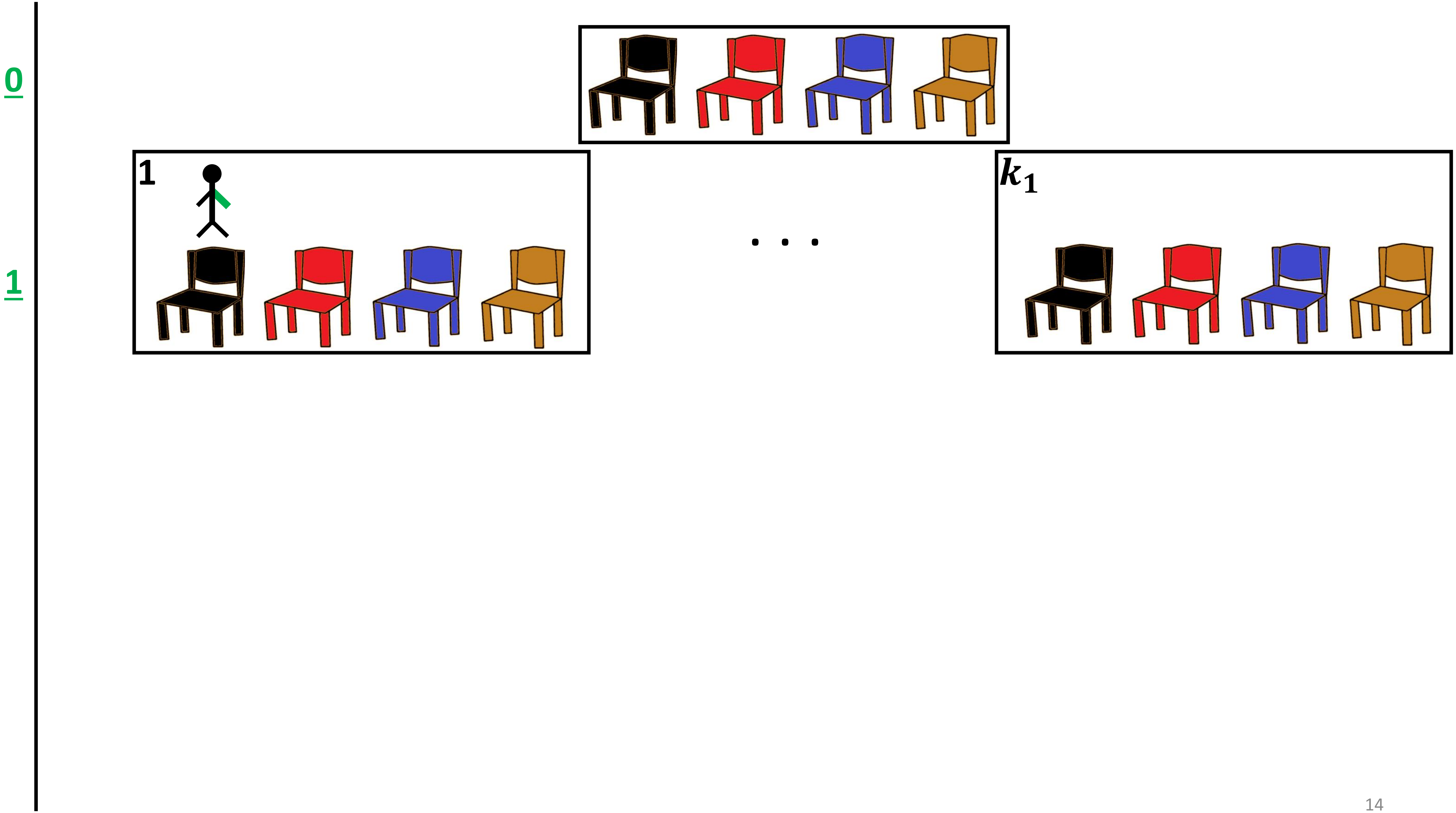
0



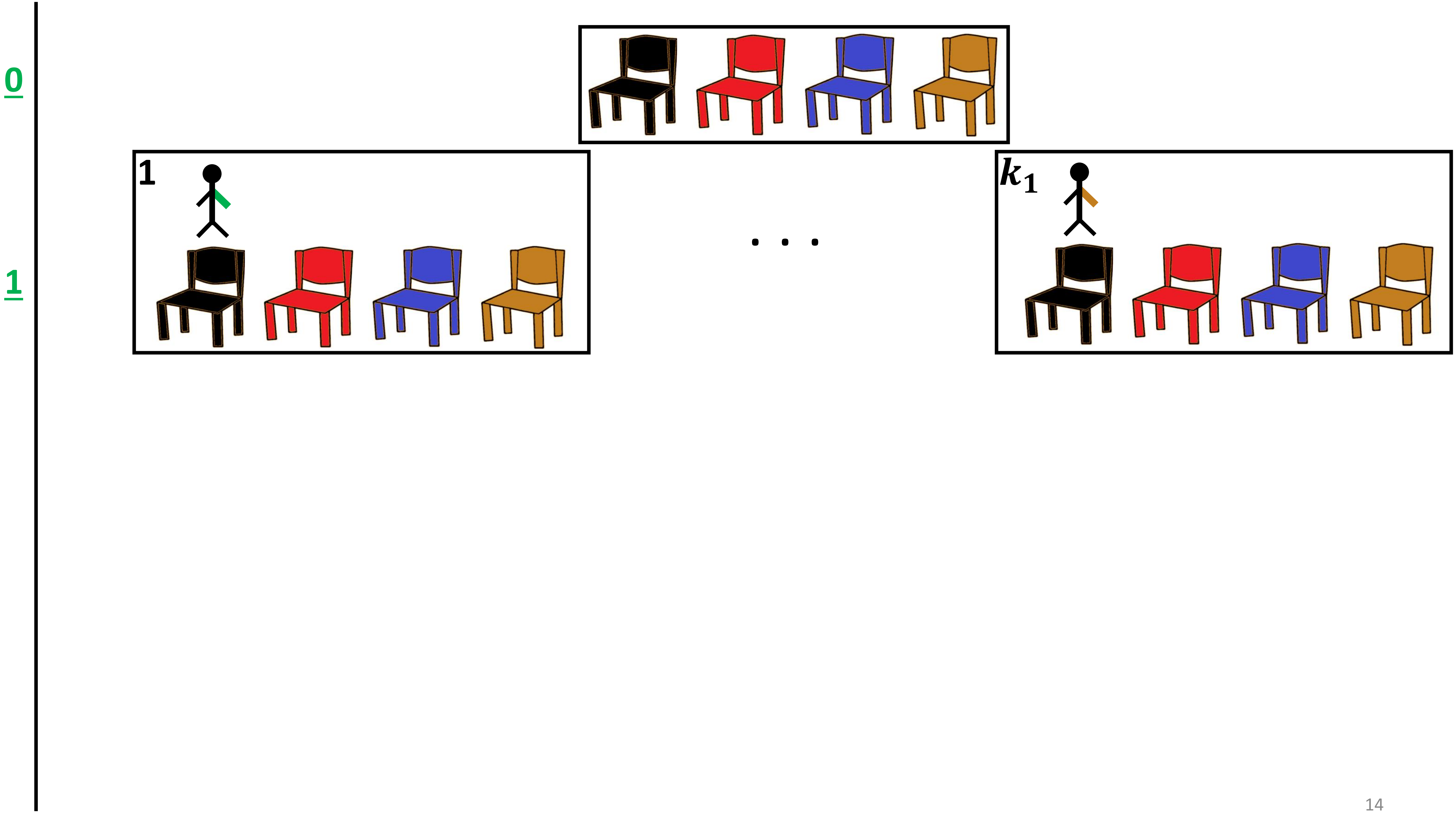
1



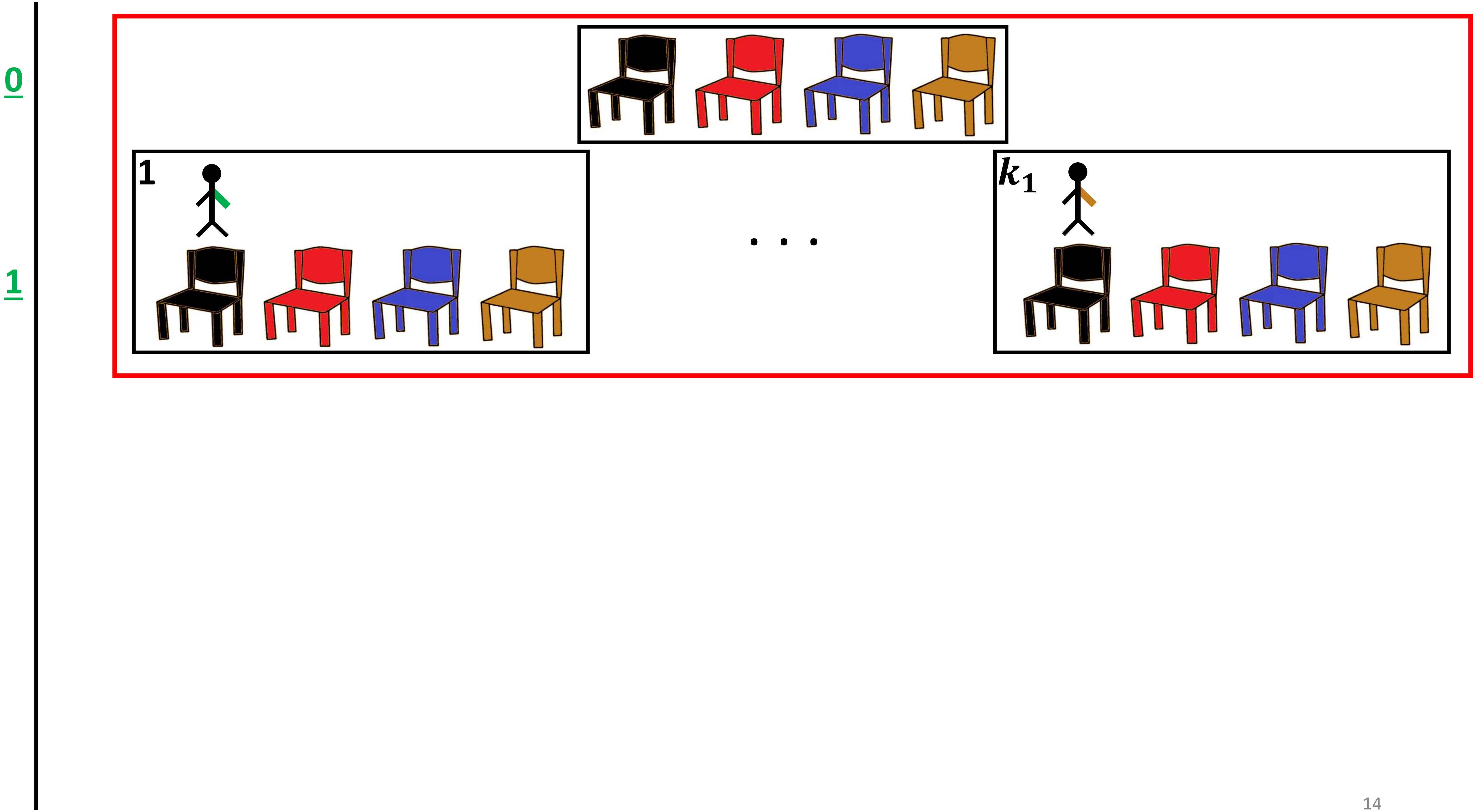

0
1
1



0
1
1
. . .
k_1



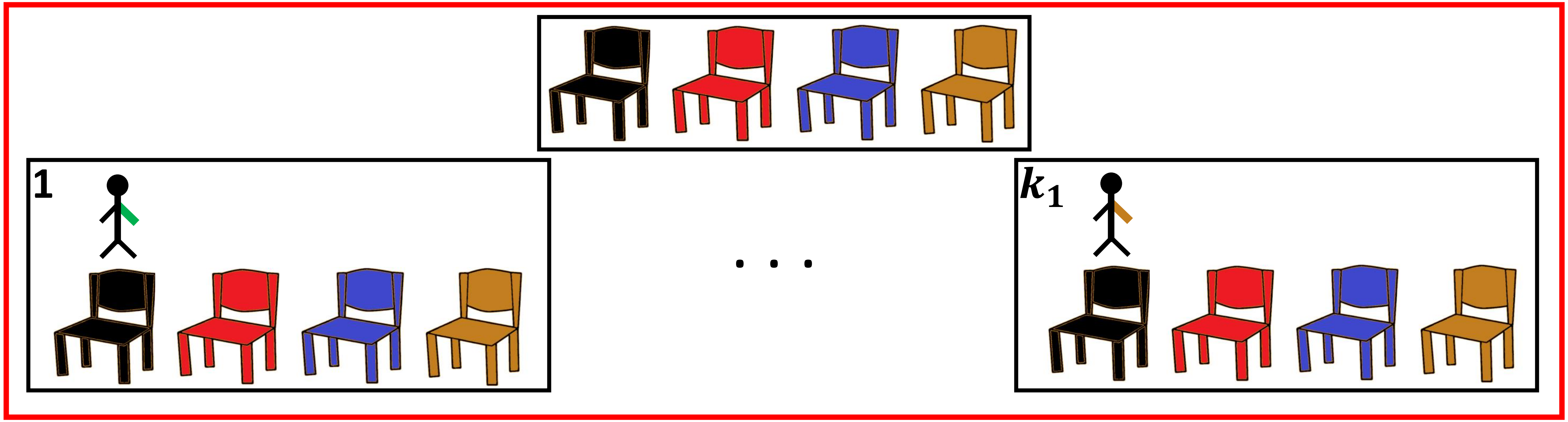
0
1
1
. . .
k_1

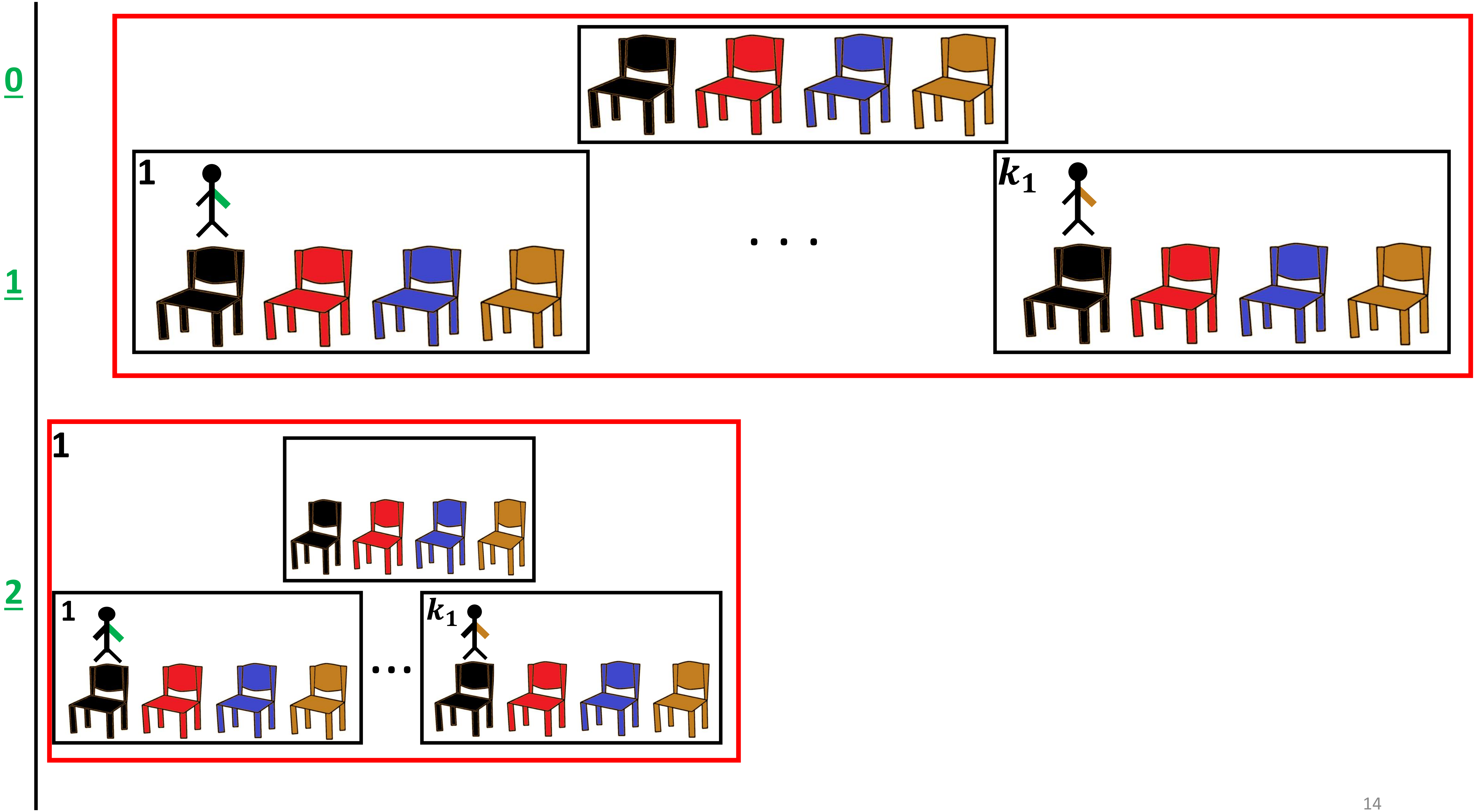

0
1
1
k_1
. . .

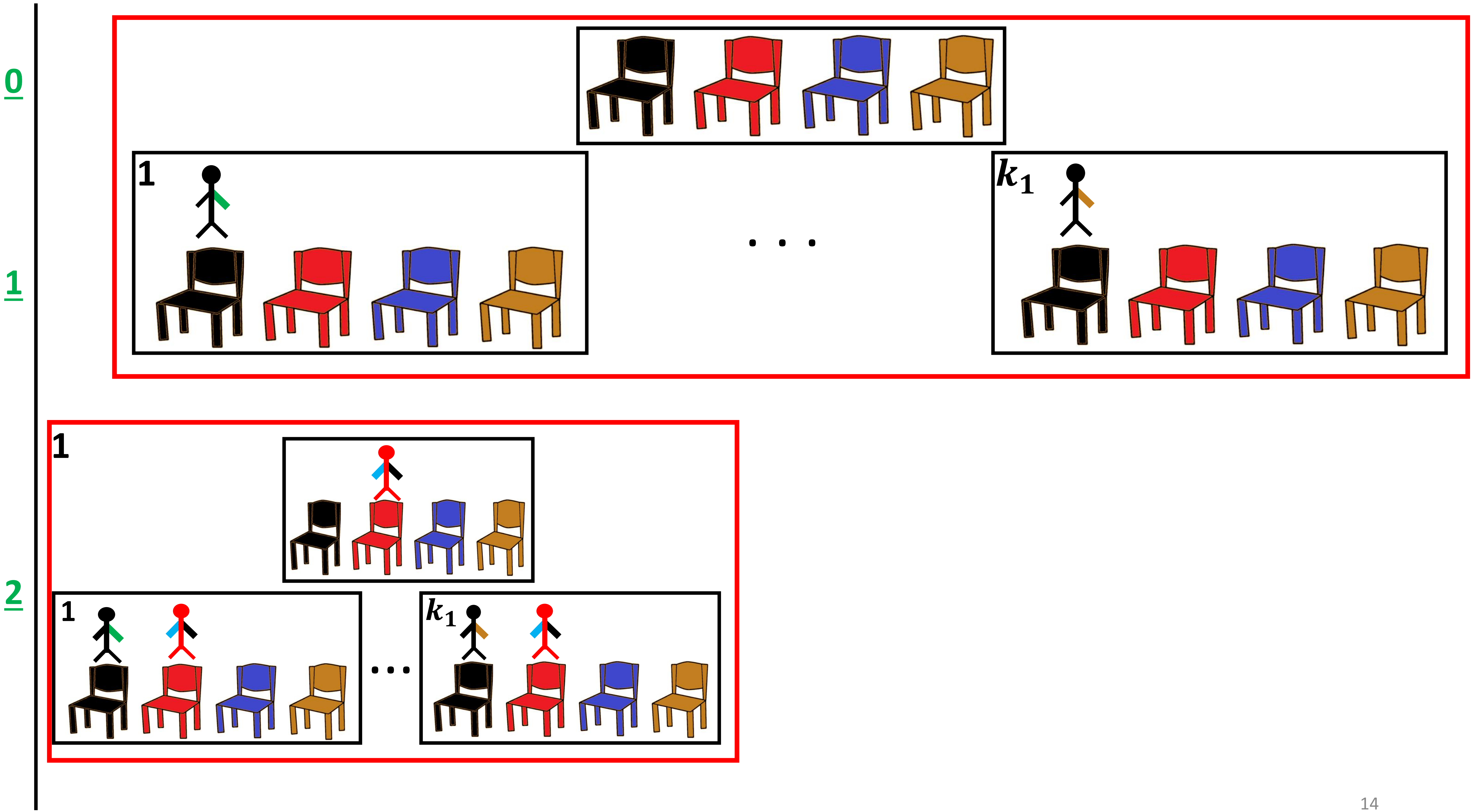
0

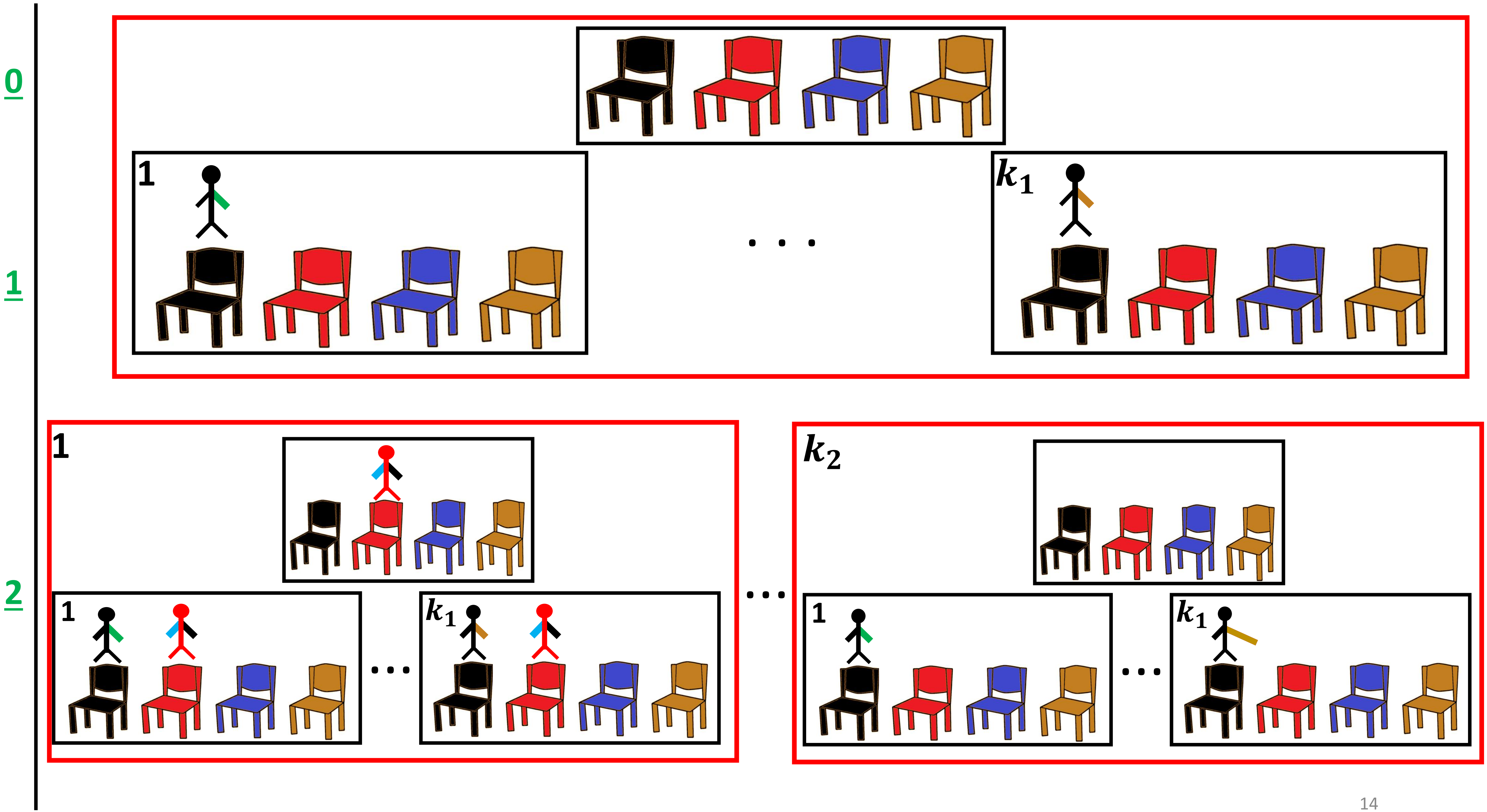
1

2

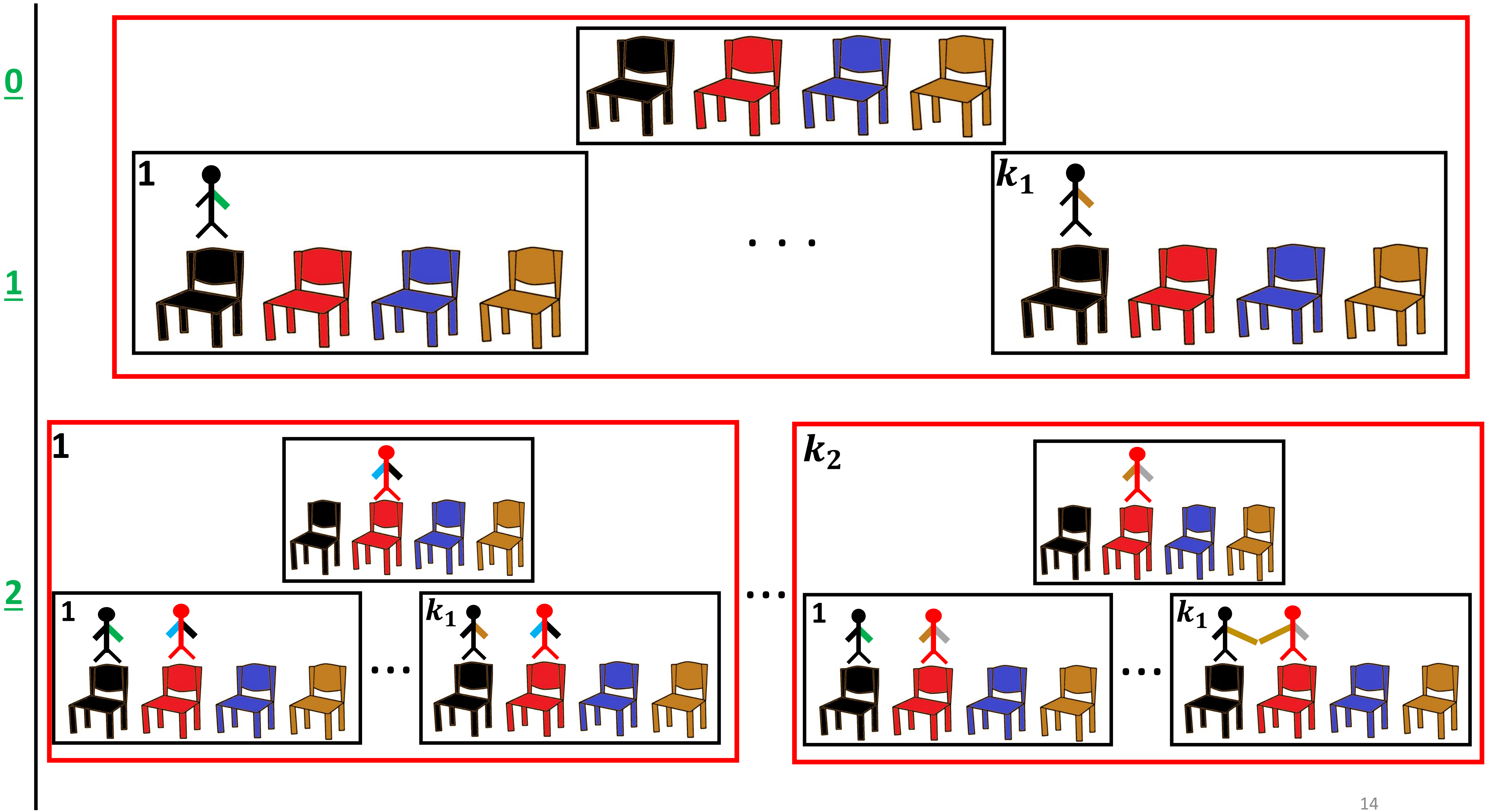


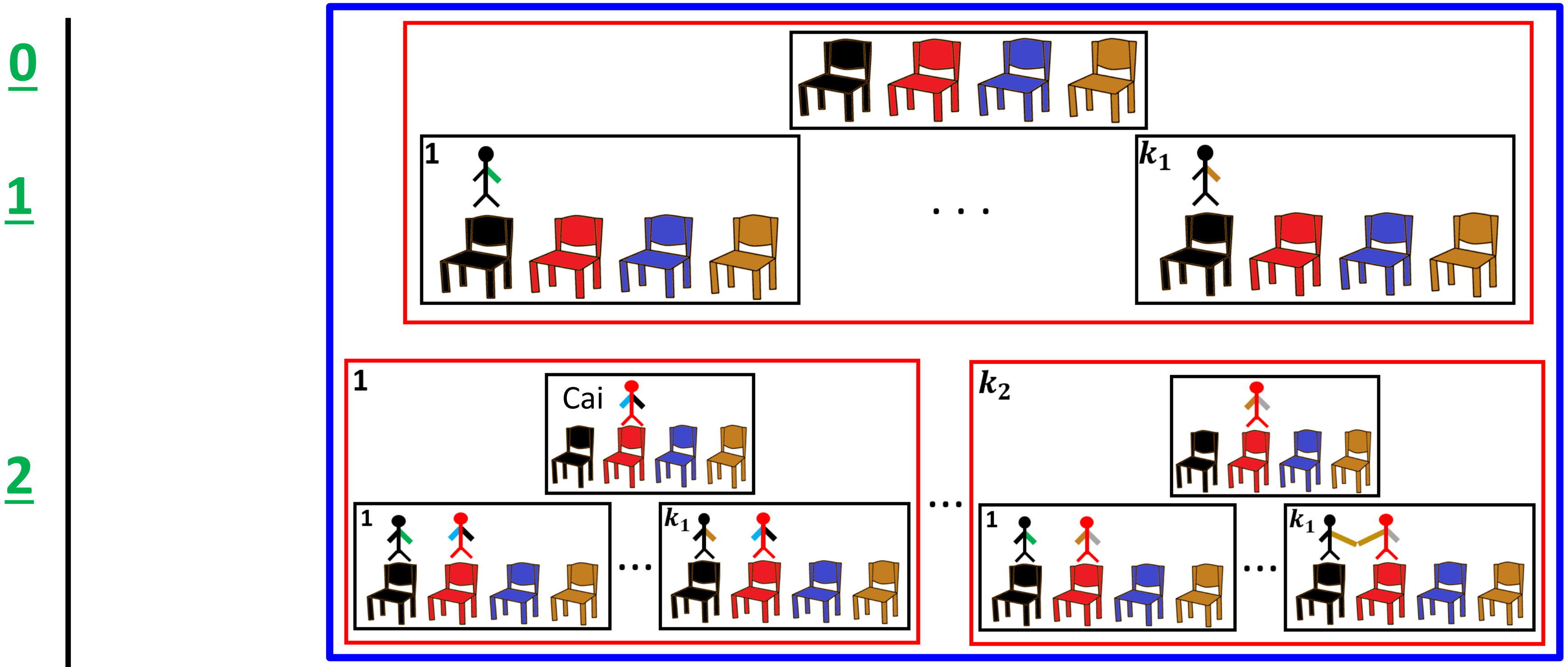

0
1
2
1
k_1
1
1
k_1

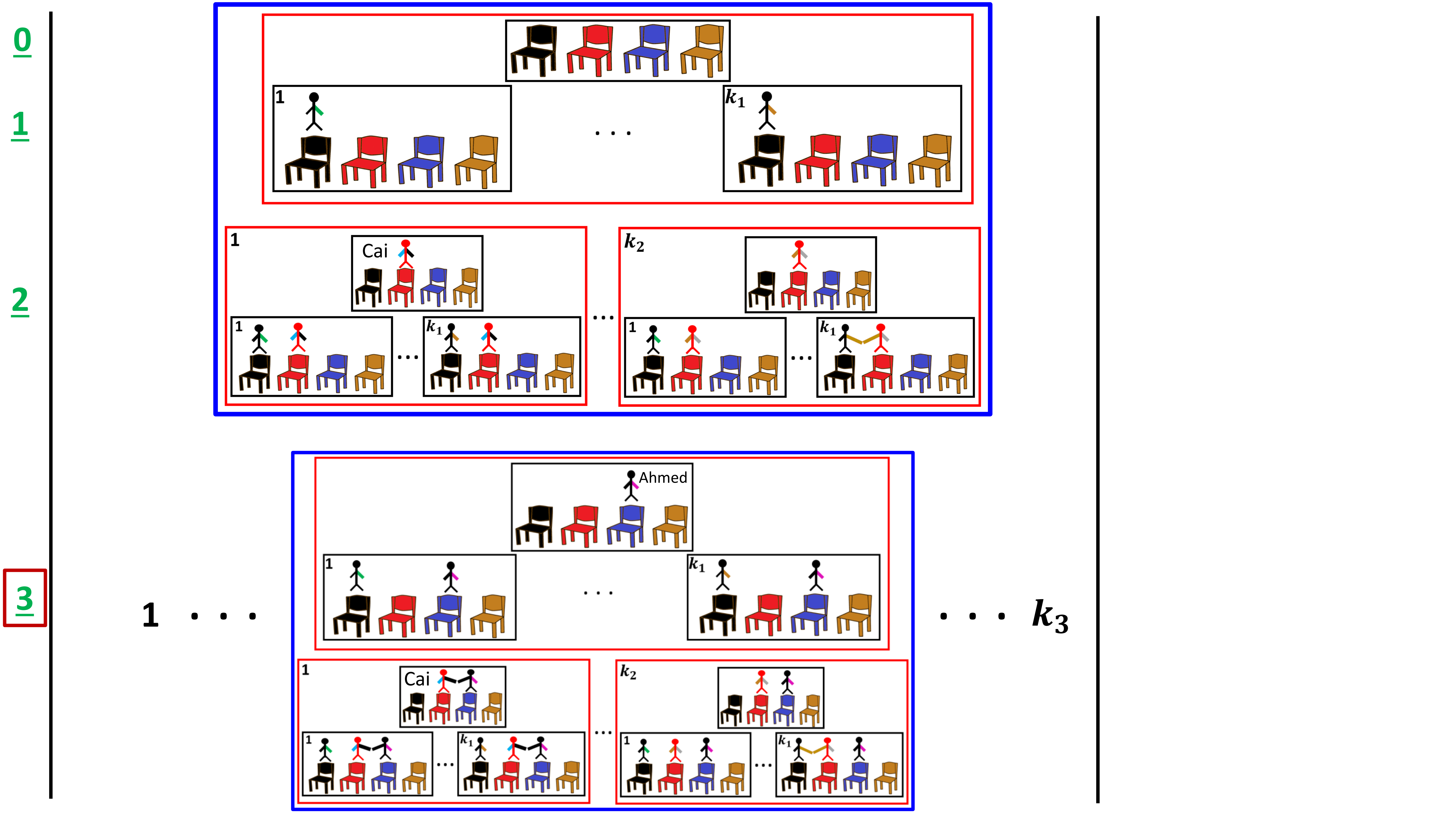

0
1
2
1
k_1
. . .
1
1
...
k_1

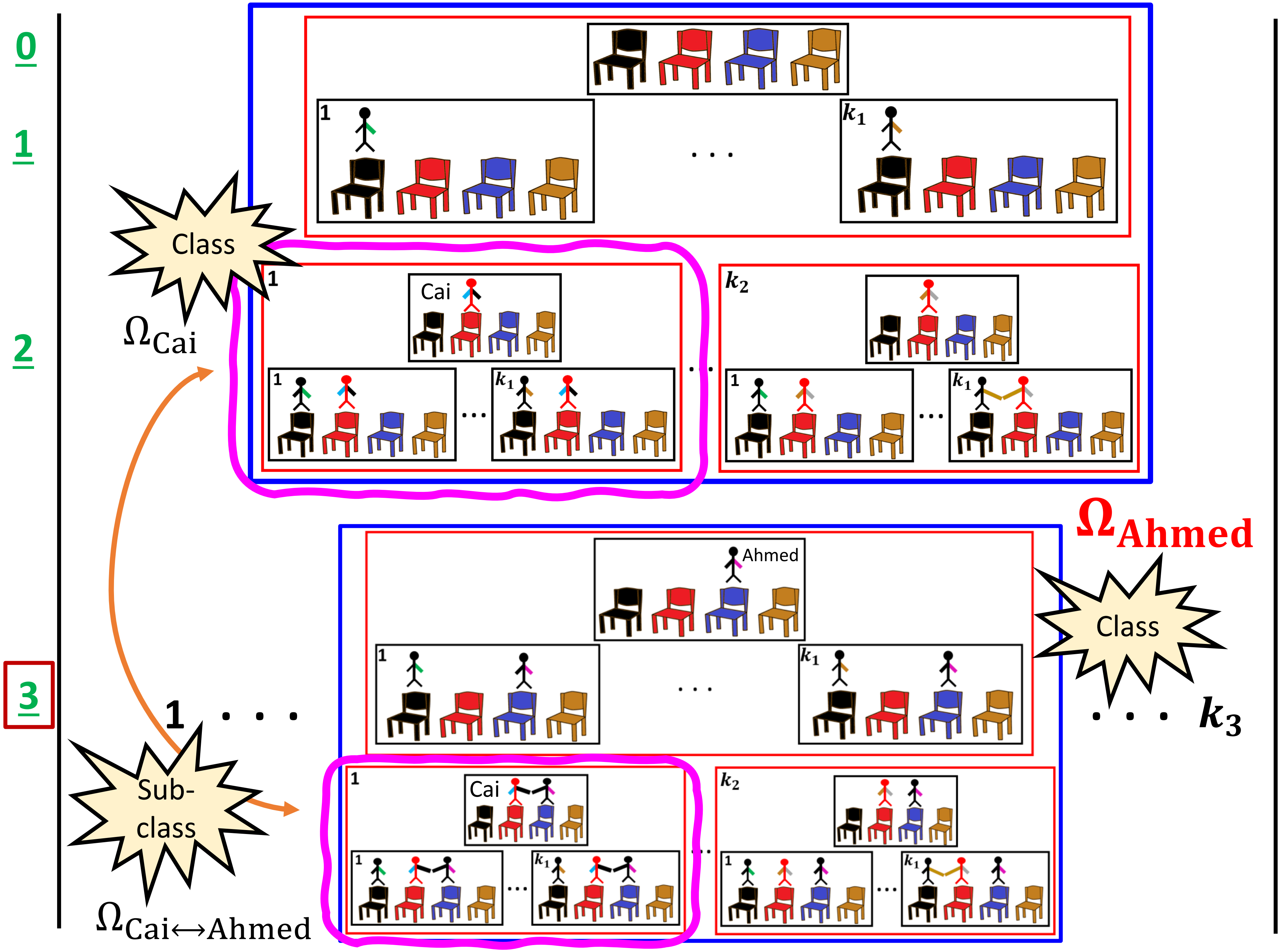


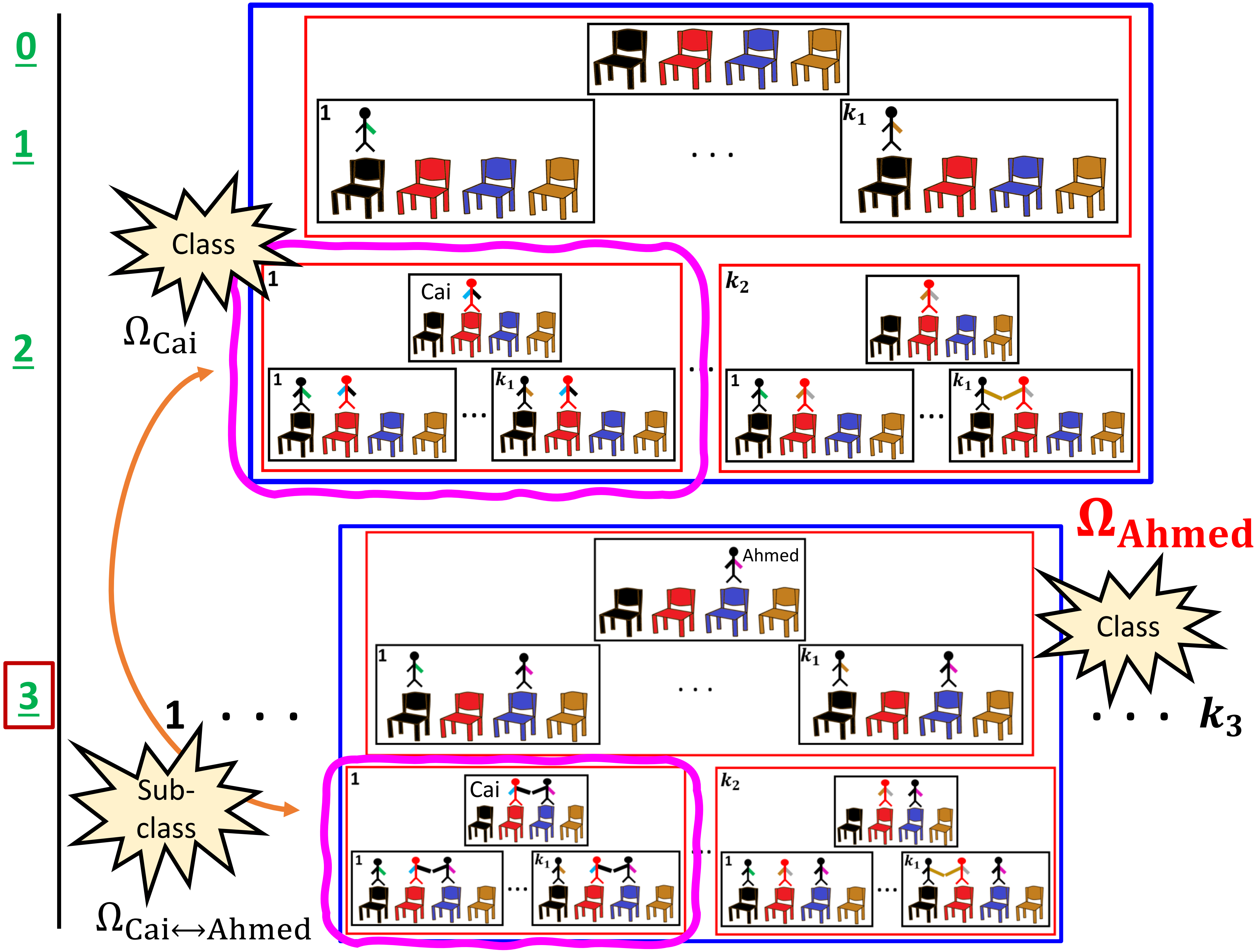
0
1
2
1
k_1
. . .
1
k_2
1
k_1
1
k_1


0
1
2
1
k_1
. . .
1
k_2
1
k_1
1
k_1


0
1
2
1
k1
1
Cai
k2
1
k1
1
k1


0
1
2
3
1
k_1
Cai
k_2
Ahmed
1 . . .
. . . k_3


0
1
2
3
Class
Ω_{Cai}
Cai
k_1
k_2
Ω_{Ahmed}
Ahmed
Class
k_3
Sub-class
$\Omega_{Cai \leftrightarrow Ahmed}$



Layer = chair

There is a 1-1 correspondence between each **class** of higher layers and a **sub-class** of any class in the current layer.

$$\Omega_{\text{Cai}}$$

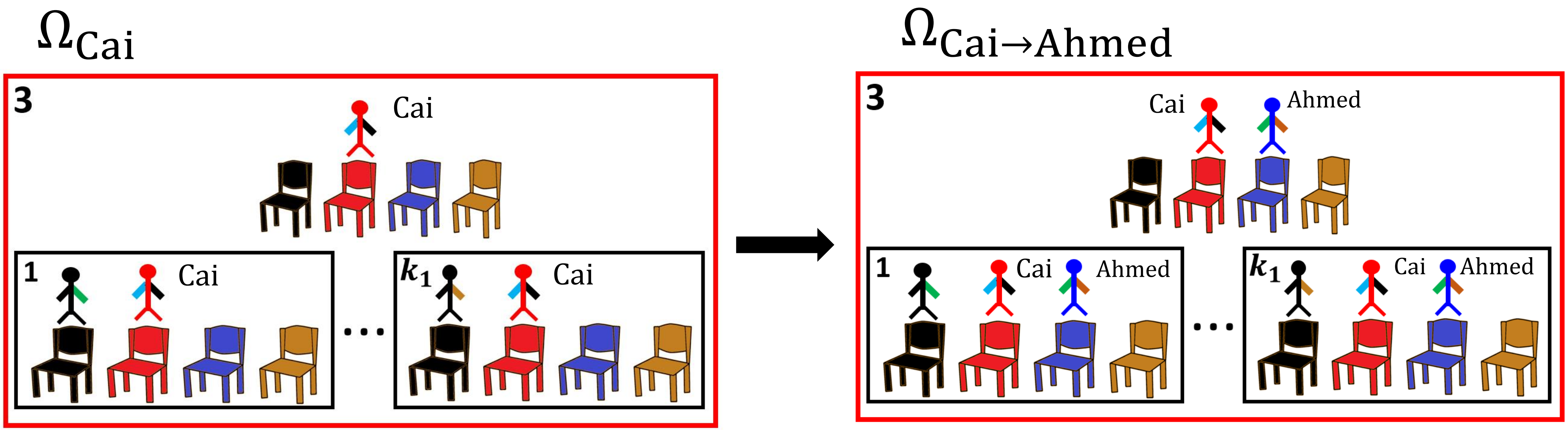
$$\updownarrow$$

$$\Omega_{\text{Cai}\leftrightarrow\text{Ahmed}} \subset \Omega_{\text{Ahmed}}$$

Can we use this inductive construction process to propagate information through this hierarchy ?

Can we use this inductive construction process to propagate information through this hierarchy ?

1 - Propagate information from exactly the previous layer (Direct neighbour handshaking possibility).



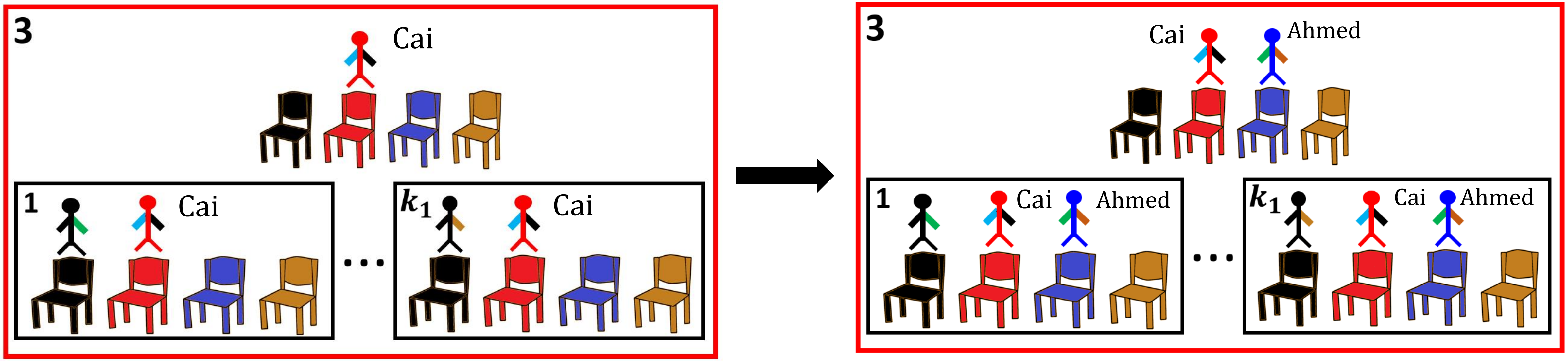
Information given: $Q(\Omega_{\mathbf{Cai}})$

Information needed: $Q(\Omega_{\text{Cai}\leftrightarrow\text{Ahmed}})$

Can we use this inductive construction process to propagate information through this hierarchy ?

1 - Propagate information from exactly the previous layer (Direct neighbour handshaking possibility).

Ω_{Cai} $\quad\quad\quad$ $\Omega_{\text{Cai}\rightarrow\text{Ahmed}}$



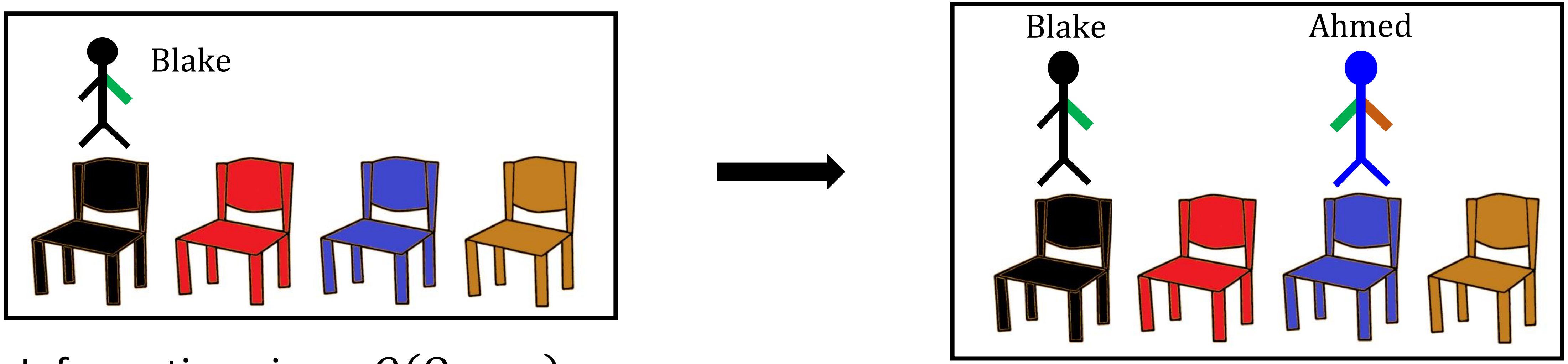
Information given: $Q(\Omega_{\textbf{Cai}})$

Information needed: $Q(\Omega_{\text{Cai}\leftrightarrow\text{Ahmed}})$

$$Q(\Omega_{\text{Cai}\leftrightarrow\text{Ahmed}}) = \ldots = e^{\frac{\text{sit(Ahmed)} + \text{Sit\_Cost}}{c}} * e^{\frac{\text{handshake}(\textit{Cai},\text{Ahmed})}{c}} * Q(\Omega_{\textbf{Cai}})$$

2 - Propagate information from the rest (No handshaking possibility).

Ω_{Blake} $\Omega_{\text{Blake}\leftrightarrow\text{Ahmed}}$

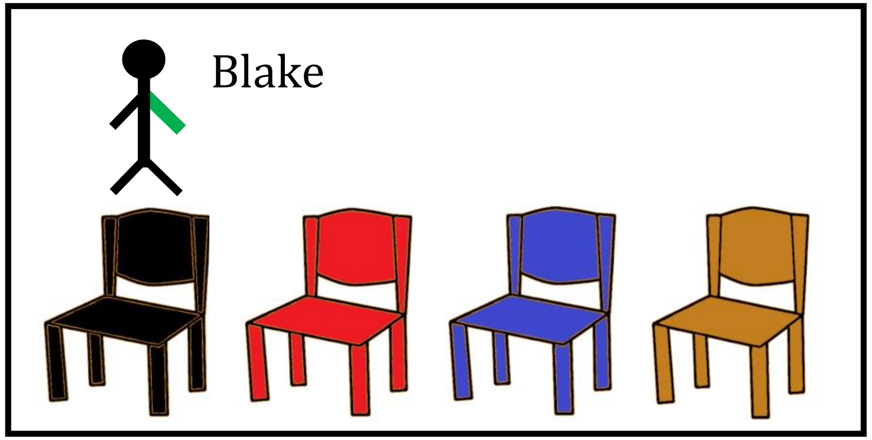


Information given: $Q(\Omega_{\textbf{Blake}})$

Information needed: $Q(\Omega_{\text{Blake}\rightarrow\text{Ahmed}})$

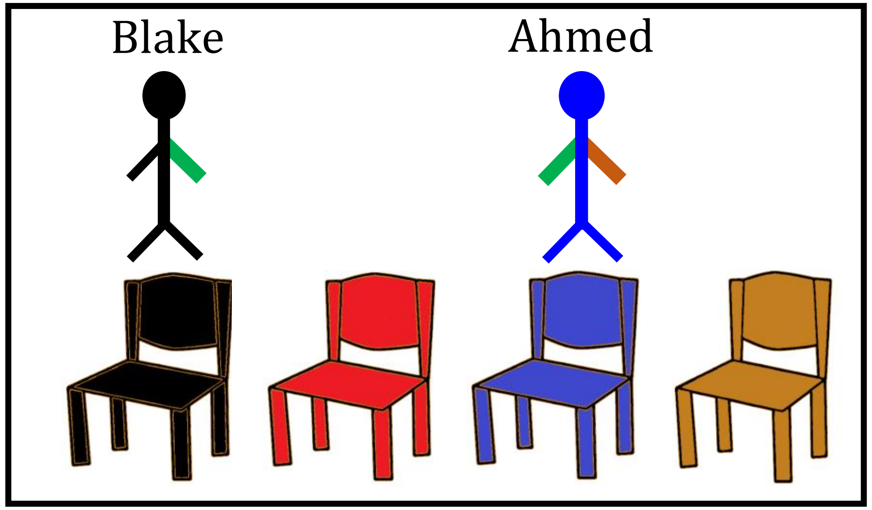
2 - Propagate information from the rest (No handshaking possibility).

Ω_{Blake}



Information given: $Q(\Omega_{\textbf{Blake}})$

$\Omega_{\text{Blake}\leftrightarrow\text{Ahmed}}$



Information needed: $Q(\Omega_{\text{Blake}\rightarrow\text{Ahmed}})$

$$Q(\Omega_{\text{Blake}\leftrightarrow\text{Ahmed}}) = \ ... \ = e^{\frac{\text{sit(Ahmed)+sit\_cost}}{c}} * Q(\Omega_{\text{Blake}})$$

$$Q(\Omega_p) = e^{\frac{sit(p)+sit\_cost}{c}} * \left[\left(\sum_{\substack{p'is \\ direct\ neighbour}} e^{\frac{handshake(p',p)}{c}} * Q(\Omega_{\boldsymbol{p'}})\right) + 1 + \sum_{\substack{p'is\ not \\ direct\ neighbour}} Q(\Omega_{\boldsymbol{p'}})\right]$$

$$Q = 1 + \sum_{p} Q(\Omega_p)$$

$$Q(\Omega_p) = e^{\frac{sit(p)+sit\_cost}{c}} * \left[\left(\sum_{\substack{p' is \\ direct\ neighbour}} e^{\frac{handshake(p',p)}{c}} * Q(\Omega_{\boldsymbol{p'}}) \right) + 1 + \sum_{\substack{p' is\ not \\ direct\ neighbour}} Q(\Omega_{\boldsymbol{p'}}) \right]$$

▶ **Theorem 2.** There is an $O(k^2N)$ time algorithm for the domain-level partition function for a 1D SDC of length N with $\leq k$ computation strands competing at each scaffold domain.

$$Q(\Omega_s) = e^{\frac{-\Delta G(M(s))+\Delta G^{\text{assoc}}}{k_B T}} * \left[\sum_{s' \in LD_s} \left(e^{\frac{-\Delta G(R(s'), L(s))}{k_B T}} * Q(\Omega_{s'}) \right) + 1 + \sum_{\substack{s' \prec s \\ s' \notin LD_s}} Q(\Omega_{p'}) \right]$$

$$Q = 1 + \sum_{s \in T} Q(\Omega_s)$$

Algorithm 2 1D SDC partition function algorithm. The proof of Theorem 2 argues that this algorithm returns Z^S as defined in Equation (6). Note that arrays are indexed from 1, and recall that $k_1, \ldots, k_N$ are the counts of competing strands at scaffold domains $d_1, \ldots, d_N$, and we let s_i^j be the j^{th} strand competing at domain d_i. See Figure 9.

$Z_{\text{curr}} = [0, 0, \ldots, 0]$ ▷ size $k = \max(k_1, \ldots, k_N)$, current (partial) partition function
$Z_{\text{prev}} = [0, 0, \ldots, 0]$ ▷ size $k = \max(k_1, \ldots, k_N)$, previous (partial) partition function
$Z^S \leftarrow 1$; $\text{sum}_a \leftarrow 0$
for $i \leftarrow 1 \ldots N$ **do**
  $\text{sum}_a \leftarrow \text{sum}_a + \sum_{i \in \{1,\ldots,k\}} Z_{\text{prev}}[i]$ ▷ sum_a: rightmost summation Equation (7)
  $Z_{\text{prev}} \leftarrow Z_{\text{curr}}$
  $Z_{\text{curr}} = [0, 0, \ldots, 0]$
  for $j \leftarrow 1 \ldots k_i$ **do** ▷ each iteration computes Equation (7) for a strand
    $t_1 = e^{-(\Delta G(d^{\text{M}}(s_i^j)) + \Delta G^{\text{assoc}})/k_B T}$
    if $i = 1$ **then** ▷ first domain where is no neighbors at all
      $Z_{\text{curr}}[j] = t_1$
    else
      $t_2 \leftarrow 0$
      for $m \leftarrow 1 \ldots k_{i-1}$ **do**
        $t_2 \leftarrow t_2 + \left(e^{-(\Delta G(d^{\text{R}}(s_{i-1}^m), d^{\text{L}}(s_i^j)))/k_B T} \right) \cdot Z_{\text{prev}}[m]$
      end for
      $Z_{\text{curr}}[j] \leftarrow t_1 + t_2 + \text{sum}_a$
    end if
    $Z^S \leftarrow Z^S + Z_{\text{curr}}[j]$ ▷ computing Equation (6)
  end for
end for
return Z^S

▶ **Theorem 1.** There is an $O(k^2N)$ time algorithm to determine the domain-level MFE for a 1D SDC of length N with $\leq k$ computation strands competing at each scaffold domain.

$$M^{\mathcal{S}} = \min_{s \in C_{d_N}} \{M_s^{\mathcal{S}}\}$$

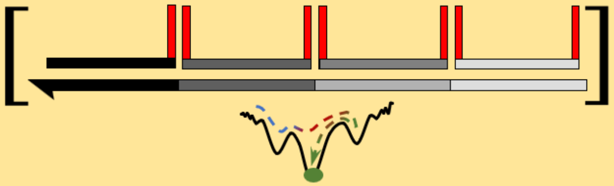
$$M_s^{\mathcal{S}} = \Delta G(d^{\mathrm{M}}(s)) + \Delta G^{\mathrm{assoc}} + \min_{s' \in L_s} \{M_{s'}^{\mathcal{S}} + \Delta G(d^{\mathrm{R}}(s'), d^{\mathrm{L}}(s))\}$$

Algorithm 1 1D SDC MFE algorithm. The proof of Theorem 1 proof shows that this algorithm returns the value $M^{\mathcal{S}}$ defined in Equation (4). Note that arrays are indexed from 1. Notation: $k_1, \ldots, k_N$ are the counts of competing strands at scaffold domains $d_1, \ldots, d_N$. Let s_i^j be the j^{th} strand competing at domain d_i.

$M_{\mathrm{curr}} = [0, 0, \ldots, 0]$ ▷ size $k = \max(k_1, \ldots, k_N)$ for current MFEs
$M_{\mathrm{prev}} = [0, 0, \ldots, 0]$ ▷ size $k = \max(k_1, \ldots, k_N)$ for previous MFEs
for $i \leftarrow 1 \ldots N$ **do** ▷ index scaffold domains
    $M_{\mathrm{prev}} \leftarrow M_{\mathrm{curr}}$
    for $j \leftarrow 1 \ldots k_i$ **do** ▷ index computational strands at scaffold domain d_i
        if $i = 1$ **then** ▷ first scaffold domain, has no left neighbour
            $M_{\mathrm{curr}}[j] \leftarrow \Delta G(d^{\mathrm{M}}(s_i^j))$
        else
            ▷ $O(k)$ steps to choose min and bind scaffold + entropic penalty
            $M_{\mathrm{curr}}[j] \leftarrow [\min_{m \in \{1,2,\ldots,k_{i-1}\}} \left(M_{\mathrm{prev}}[m] + \Delta G\left(d^{\mathrm{R}}(s_{i-1}^m), d^{\mathrm{L}}(s_i^j)\right)\right) + \Delta G(d^{\mathrm{M}}(s_i^j)) + \Delta G^{\mathrm{assoc}}]$
        end if
    end for
end for
$M^{\mathcal{S}} \leftarrow \min_{k' \in \{1,2,\ldots,k_N\}} M_{\mathrm{curr}}[k']$ ▷ $O(k)$ steps implement Equation (4) giving $M^{\mathcal{S}}$
return $M^{\mathcal{S}}$

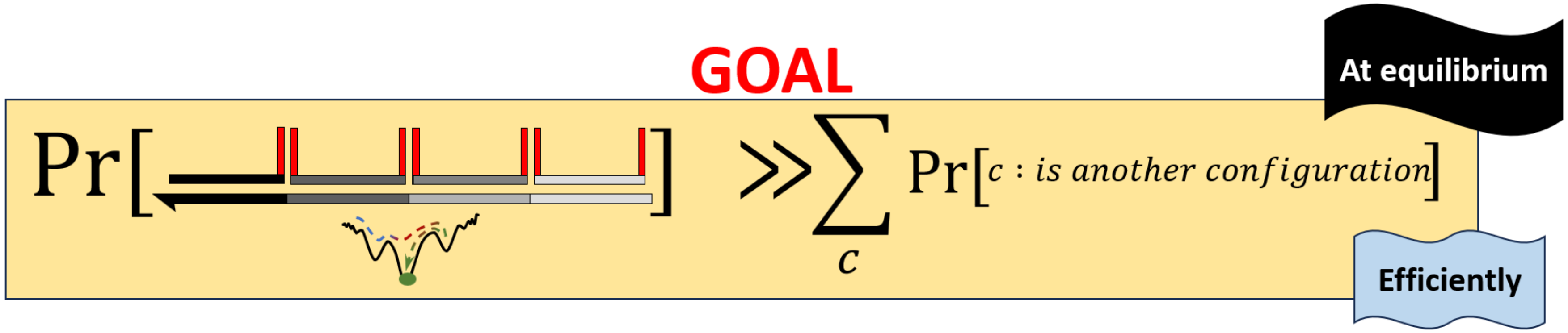
GOAL

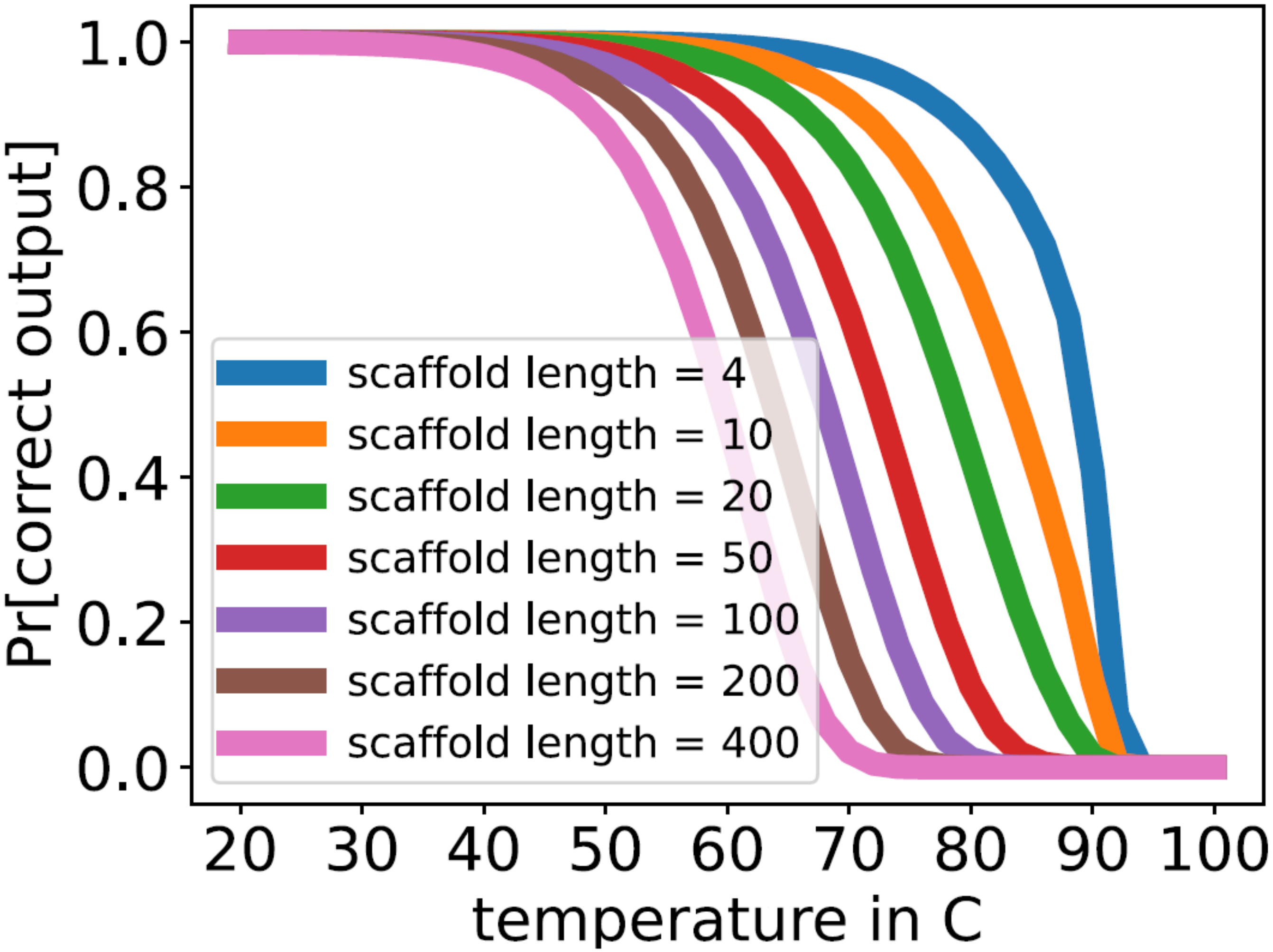
At equilibrium

$\Pr[$  $]$ $\gg \sum_{c} \Pr[c : is\ another\ configuration]$

Efficiently

GOAL





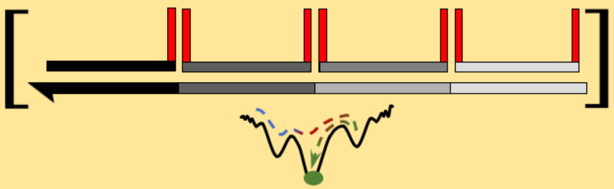
Benefits of Domain Based models !!

Previous: 1 simulation of length 13
Now: 280 simulations in 20 min [800 strands]

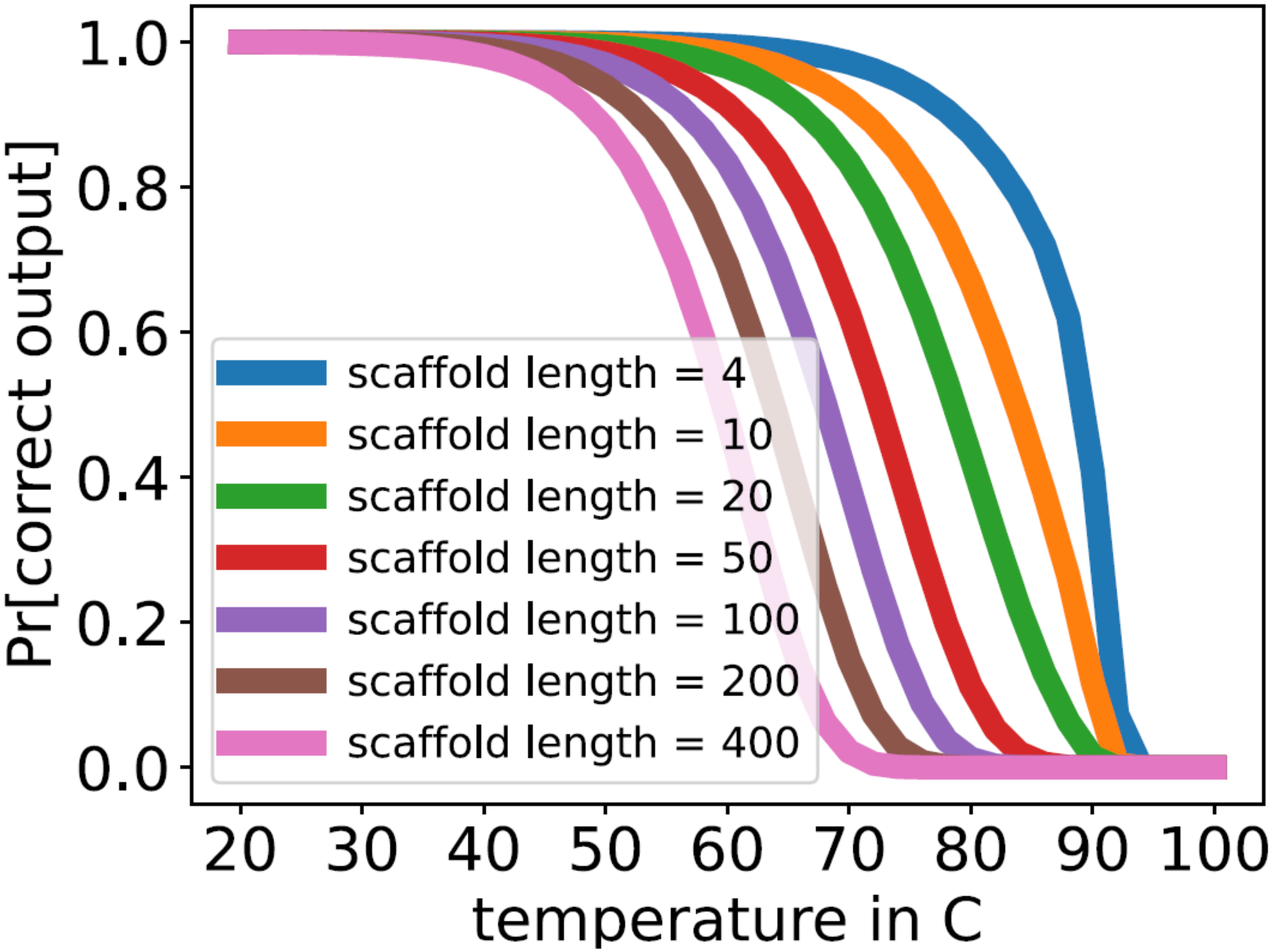
At equilibrium

GOAL

$$\Pr[\text{ }] \gg \sum_{c} \Pr[c : is\ another\ configuration]$$



Efficiently



Benefits of Domain Based models !!

Previous: 1 simulation of length 13

Now: 280 simulations in 20 min [800 strands]

Thanks

Hamilton Institute

Maynooth University
National University of Ireland Maynooth

dna.hamilton.ie

We're hiring!

Postdoc, PhD

sfi Science Foundation Ireland

erc

European Innovation Council

Funded by the European Union