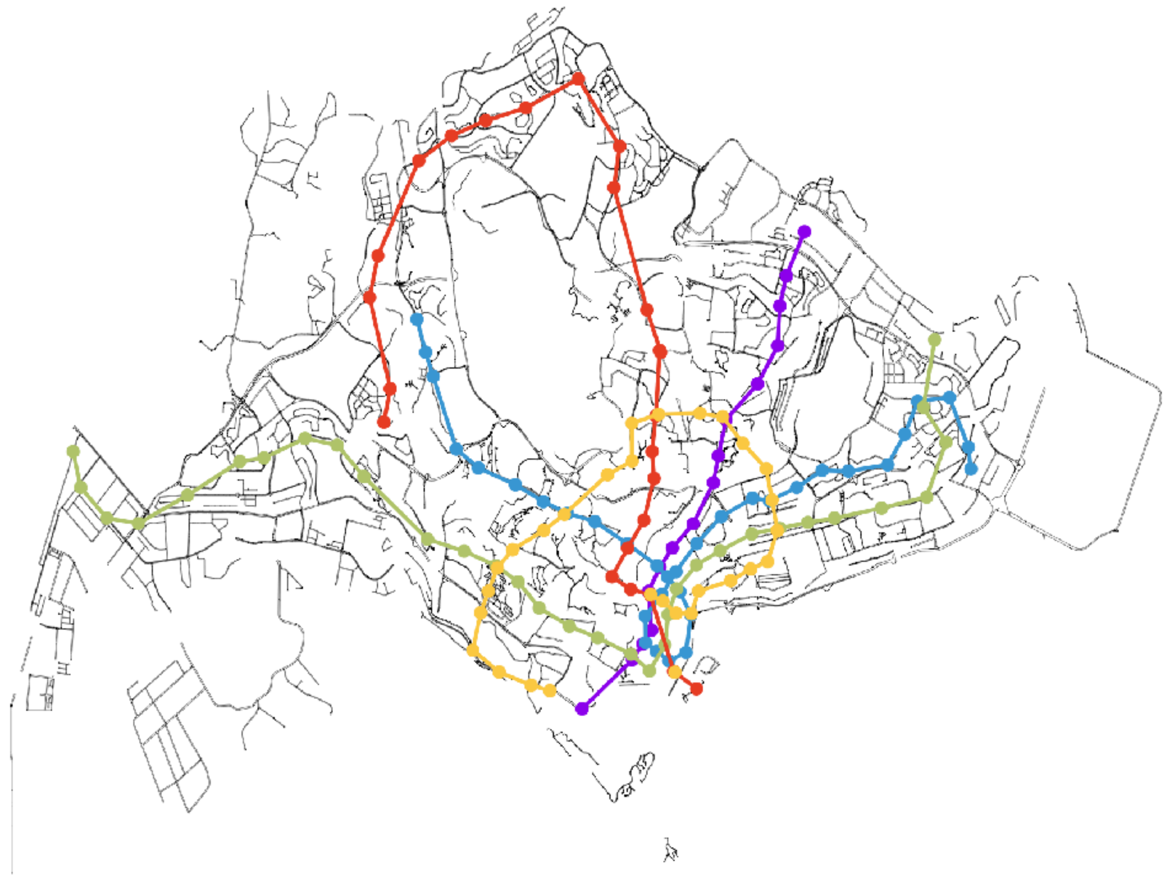
CSCI 357: Algorithmic Game Theory

Lecture 16: Voting & Social Choice 3

Shikha Singh

Announcements and Logistics

- Mask policy in CS 357:
  - Starting Thursday April 14, masking is optional in lecture
  - Starting tomorrow, masking is optional in office hours
- HW 7 (partner assignment) is due this Thursday at 11 pm
- Midterm 2 will be a 24-hour take home
  - Tentative date April 24 - 25
  - LaTeX solutions and submit via Gradescope
  - Cumulative with focus on topics not covered in Midterm 1

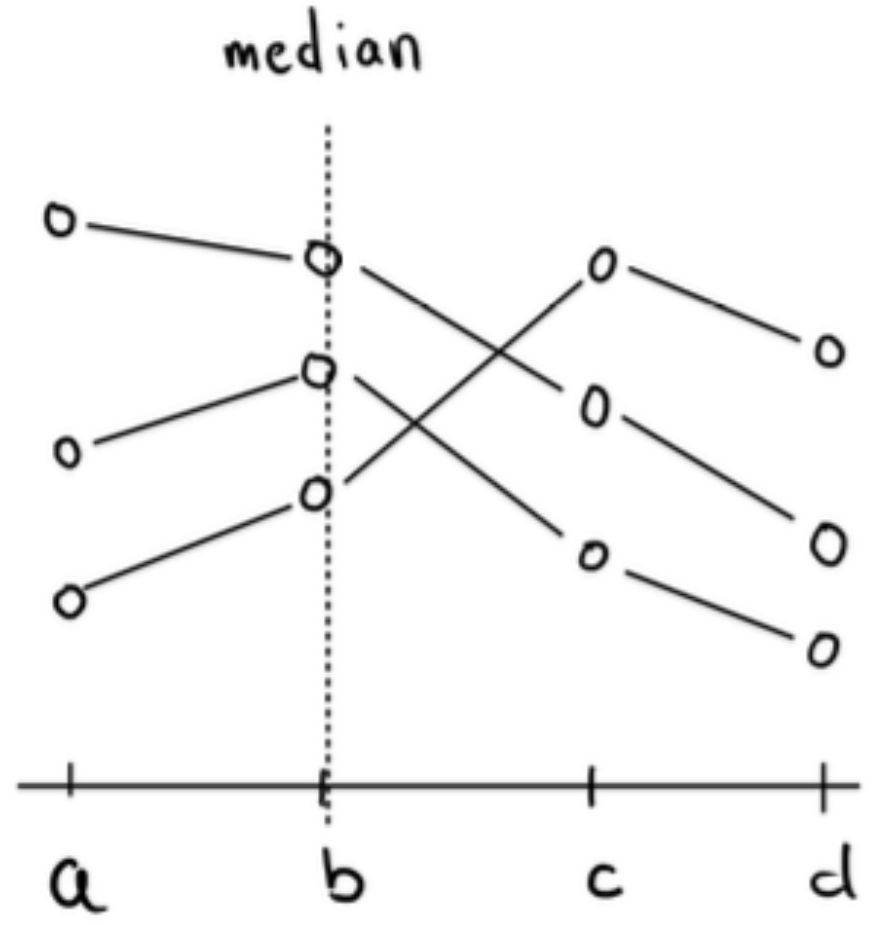
Questions?

Overview of Remaining Weeks

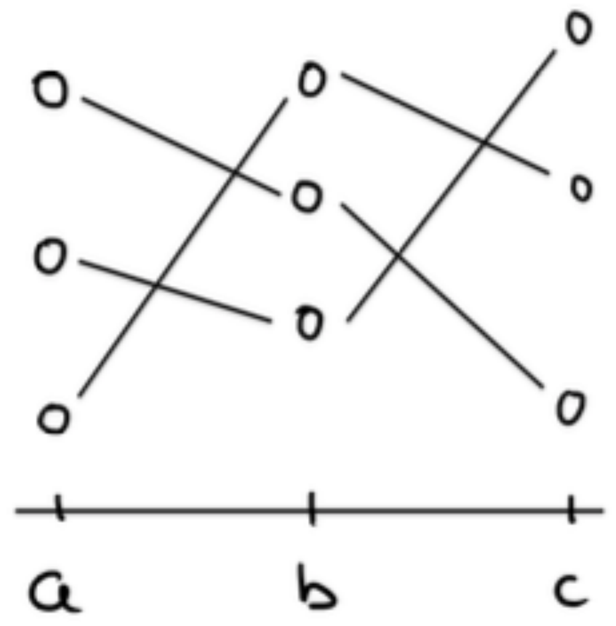
	Monday	Tuesday	Wednesday	Thursday	Friday
Week 7	Voting & Social Choice			HW 6 due	
Week 8	Extensive-form and Repeated games			HW 7 due	
Week 9	Incentives in P2P systems		HW 8 due	Project proposal due	
Week 10	Midterm 2/ No class			Misc	Project checkpoint 1
Week 11	Misc		Misc	Project checkpoint 2	
Week 12	Project presentations: Checkpoint 3				
Finals period	Final Project Report Due				

Last Time

- Proved **Gibbard-Satterthwaite theorem**.
    - When there are 3 or more alternatives, a voting rule is strategyproof and onto if and only if it is dictatorial
- If we do not require onto, what is the space of strategyproof voting rules?
    - Constant or majority of two alternatives
- Mechanism design: all the ways to circumvent GS theorem
    - Mechanism design with money (first half of course)
    - Restricted preferences: single-peaked or others



Single-peaked preferences



Not single-peaked

Today

- Look at the hardness of manipulation as a potential way to circumvent GS
- Computational viewpoint of Voting
- Brief look at fair division
- Wrap up this unit and set up next: extensive form games

Circumventing GS: Complexity

- So far we did not put any restrictions on the strategies voters can use
- Suppose we restrict to strategies that are "efficient" to compute
- f-manipulation problem:
  - Input: A set of preference lists $P_2, \ldots, P_n$ of voters $1, \ldots, n$
  - Goal: Compute a preference list P_1 (a possible misreport) such that 1's favorite candidate a wins: $f(P_1, \ldots, P_n) = a$
- **Questions**:
  - Is it always possible to find such a list?
  - How computationally difficult is it to solve f-manipulation problem?

Greedy Manipulation: Borda

- Suppose you are trying to solve the f-manipulation problem in Borda

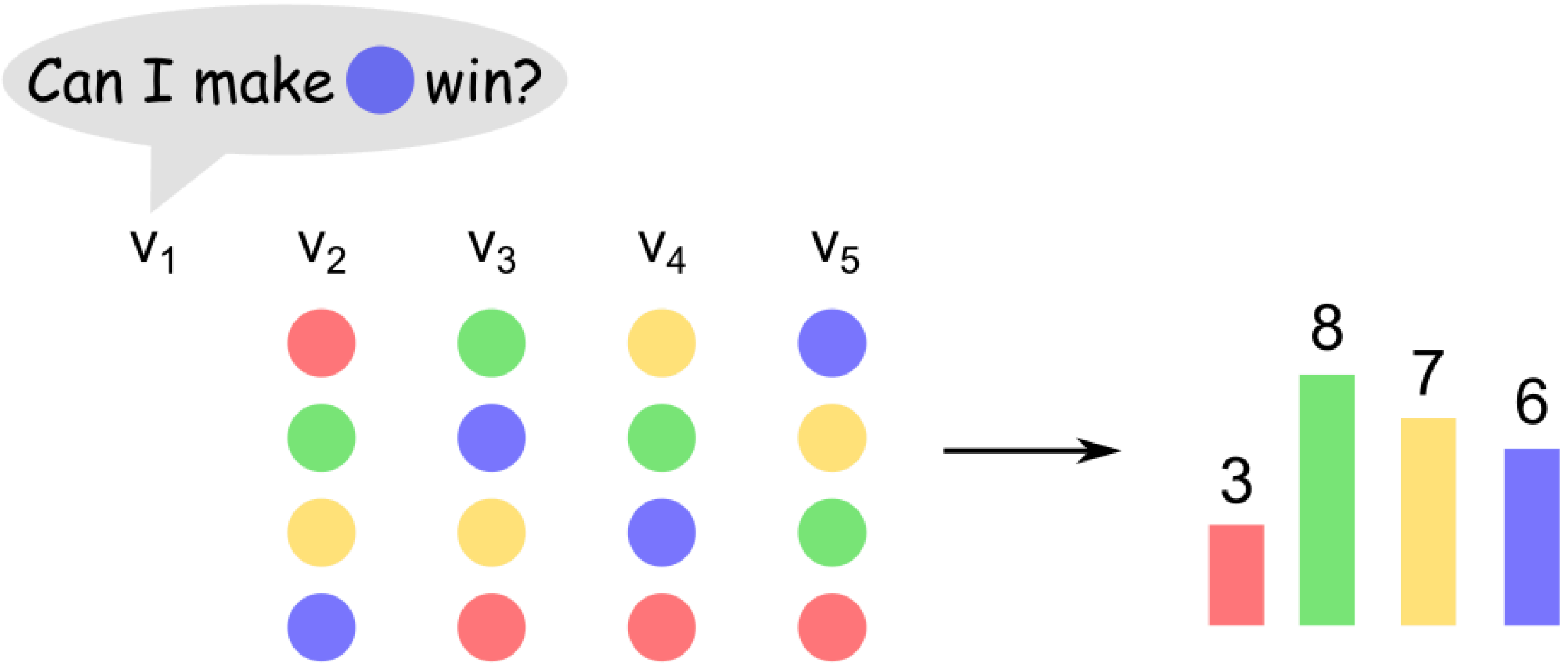


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Greedy Manipulation: Borda

- Suppose you are trying to solve the f-manipulation problem in Borda

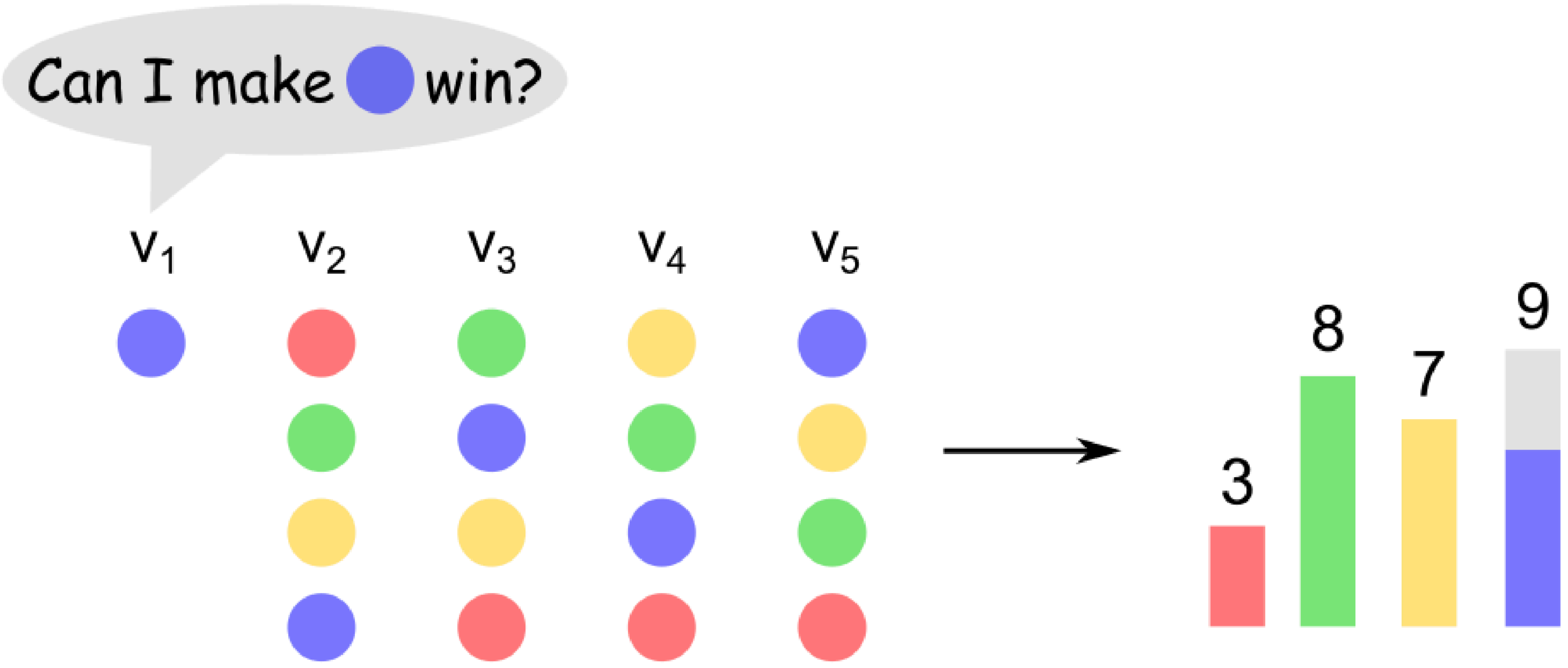


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Greedy Manipulation: Borda

- Suppose you are trying to solve the f-manipulation problem in Borda

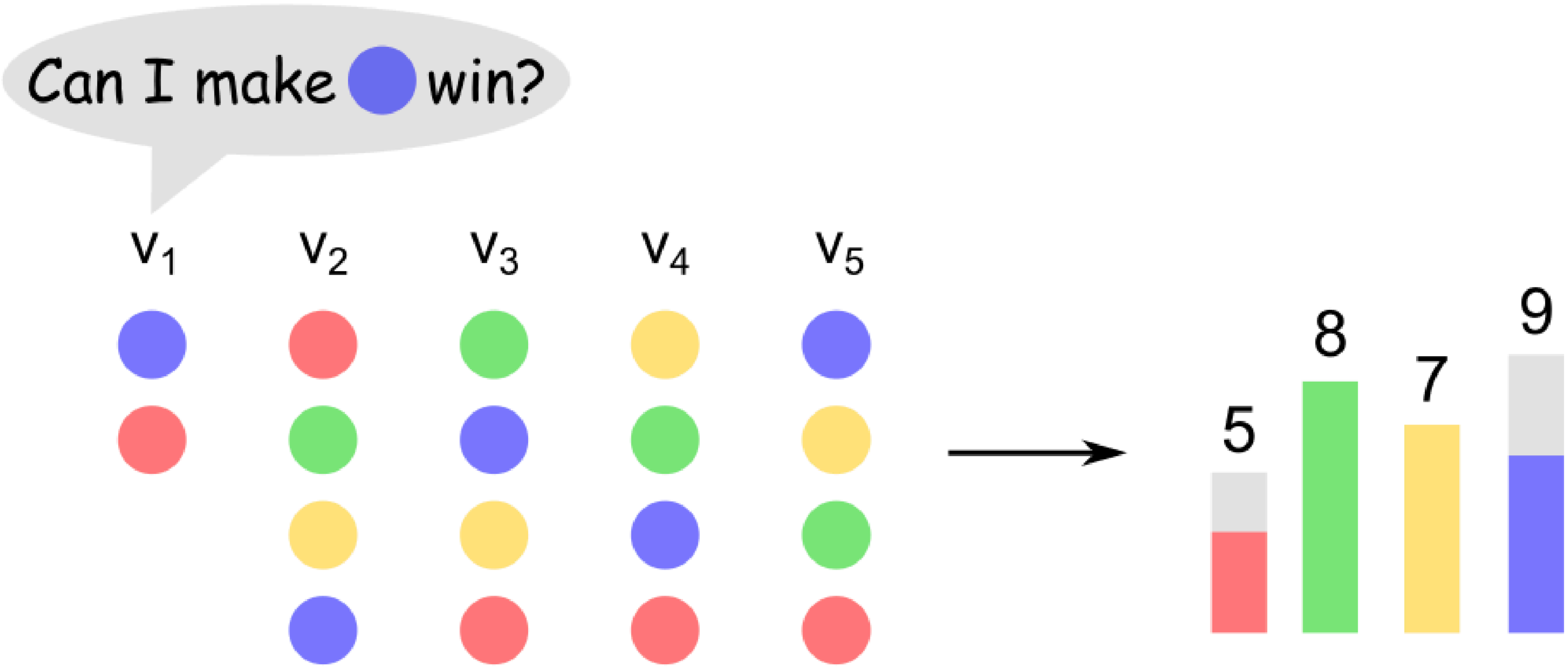


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Greedy Manipulation: Borda

- Suppose you are trying to solve the f-manipulation problem in Borda

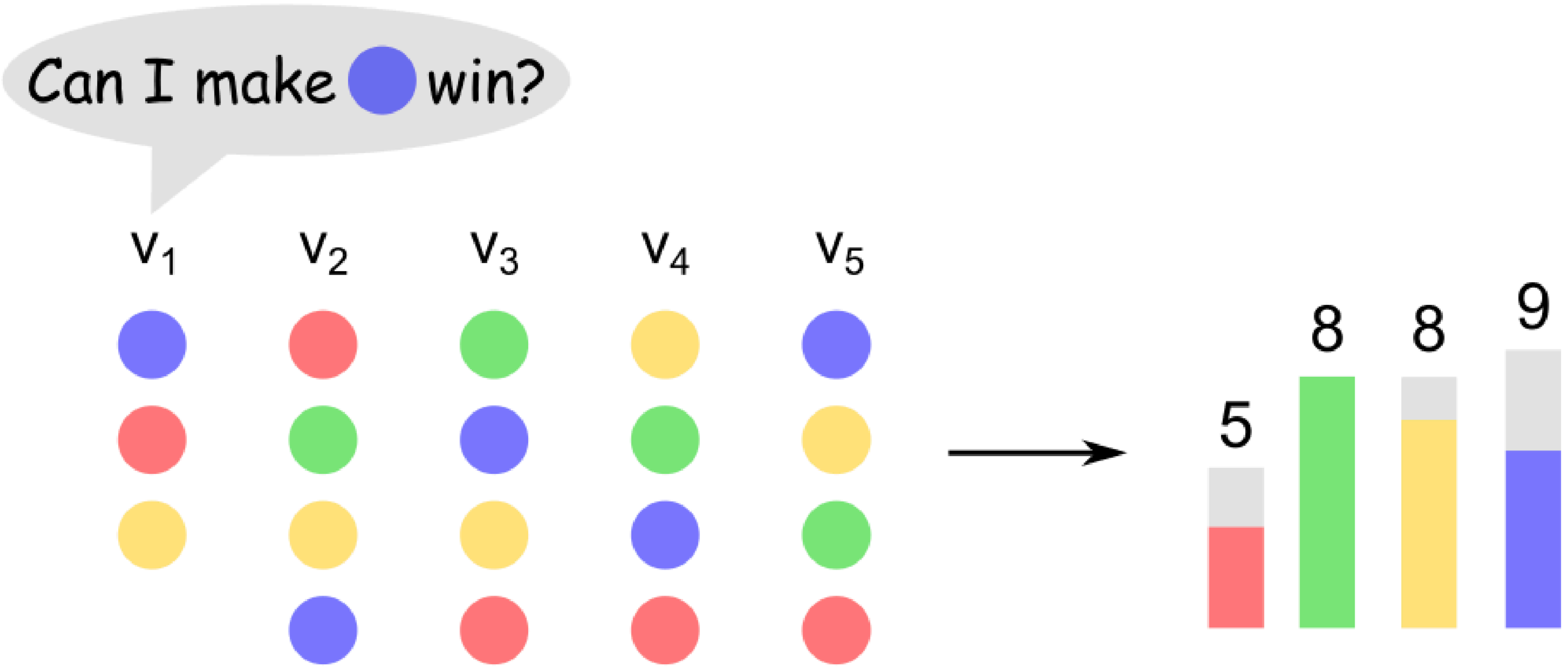


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Greedy Manipulation: Borda

- Suppose you are trying to solve the f-manipulation problem in Borda

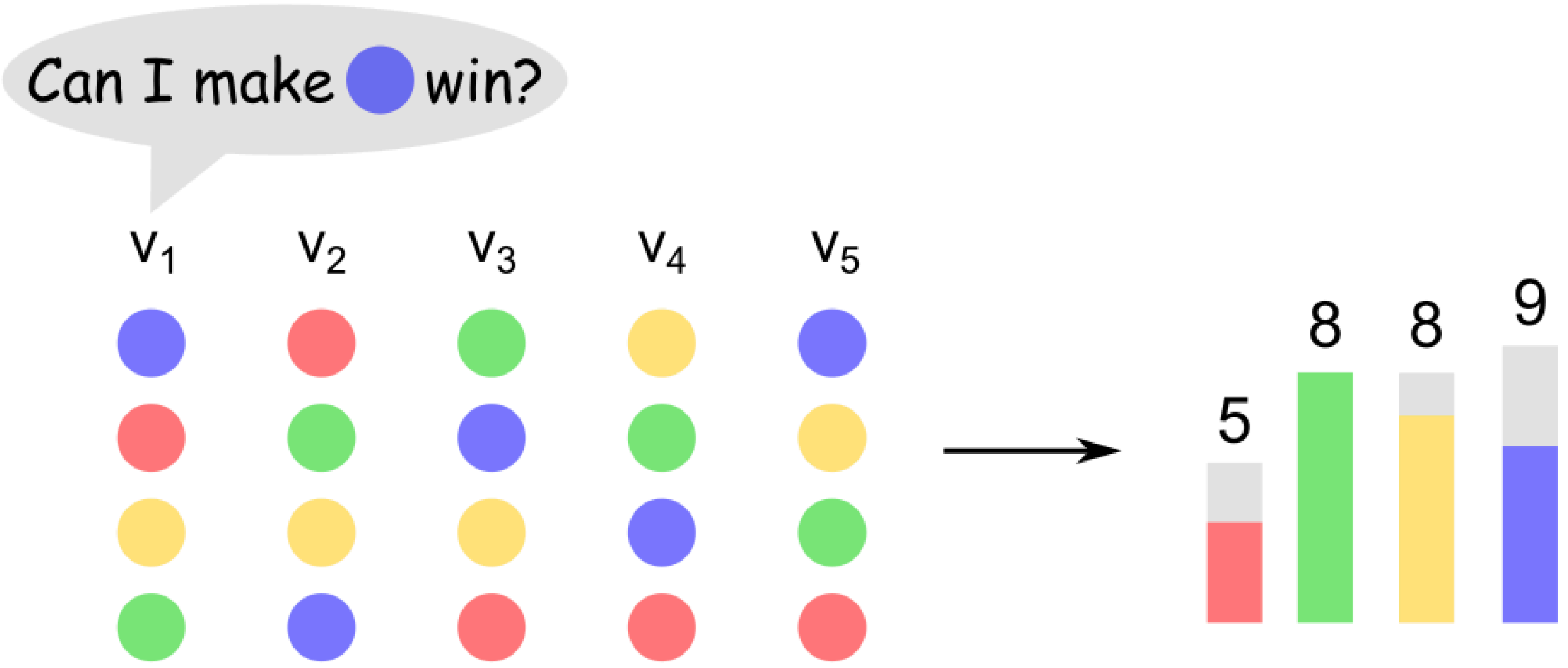


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Greedy Strategy

- Rank 1's favorite candidate a at the top of P_1
- While there are unranked candidates:
  - If a candidate b can be "safely placed" in the next position without preventing a from winning, do so
- Otherwise, return False (not possible to make a win)
- In Borda rule: placing candidates in the reverse Borda score satisfies this
- Homework 7: Show that greedy is optimal: always solves f-manipulation whenever it is possible
- Does the greedy strategy always work?

Ranked-Choice Voting

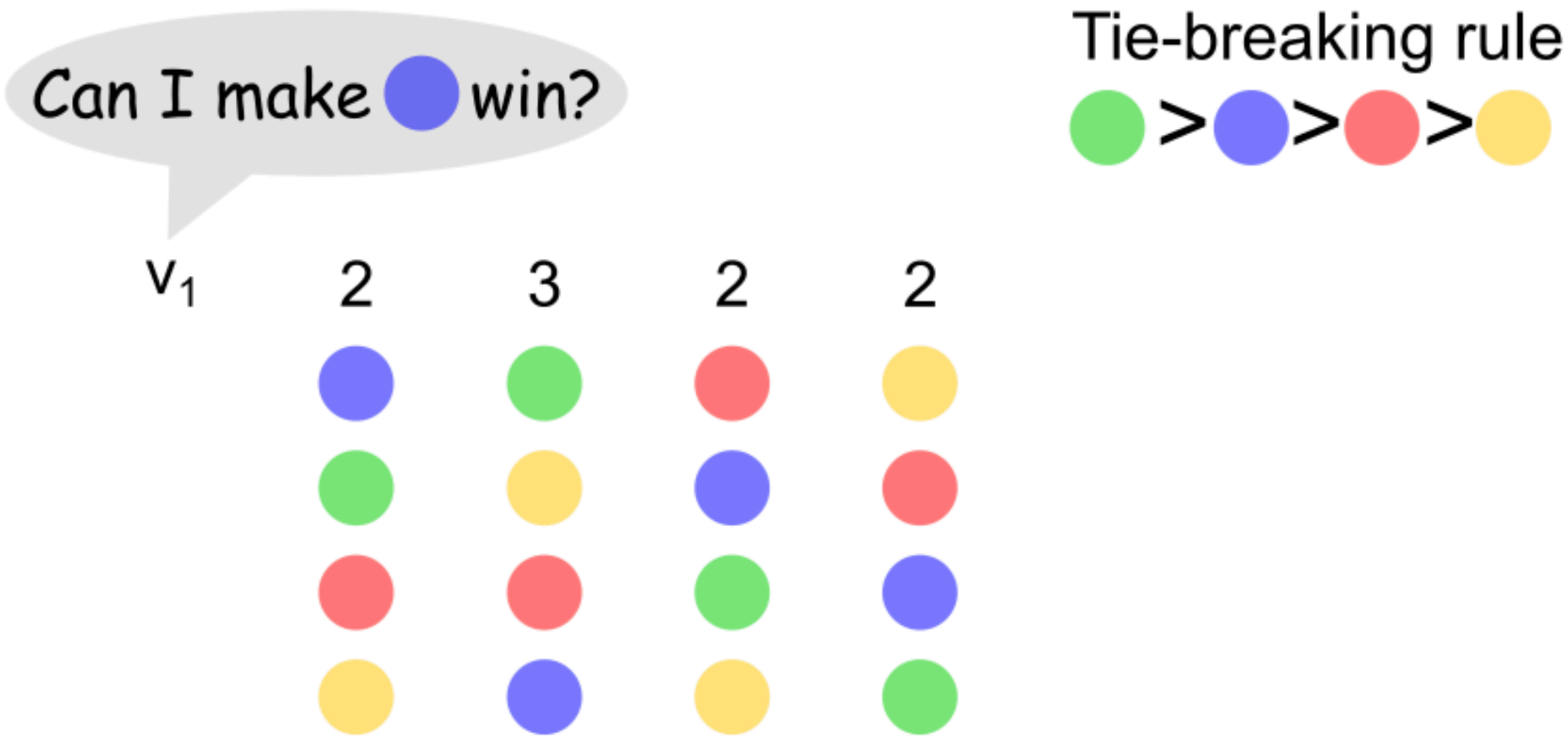


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

Tie-breaking rule

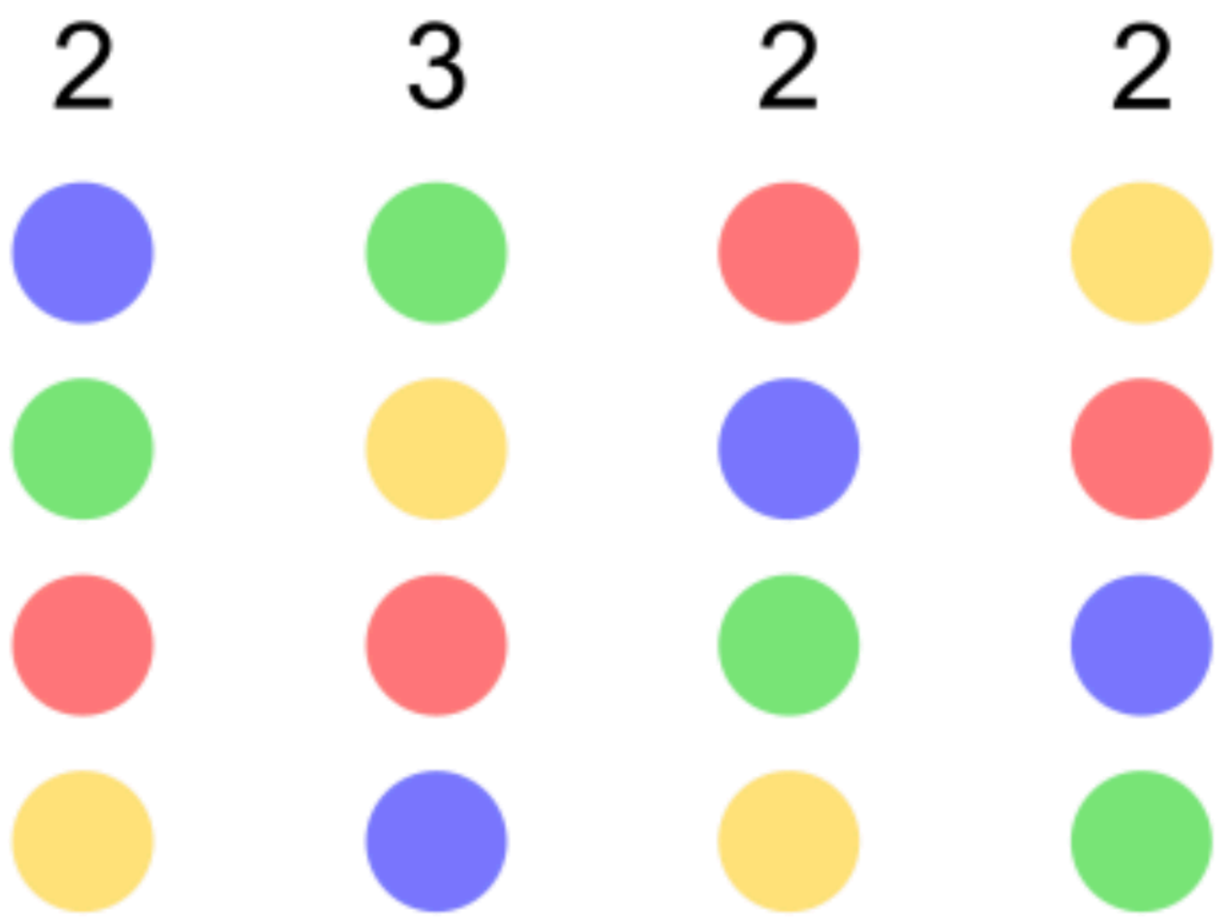


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

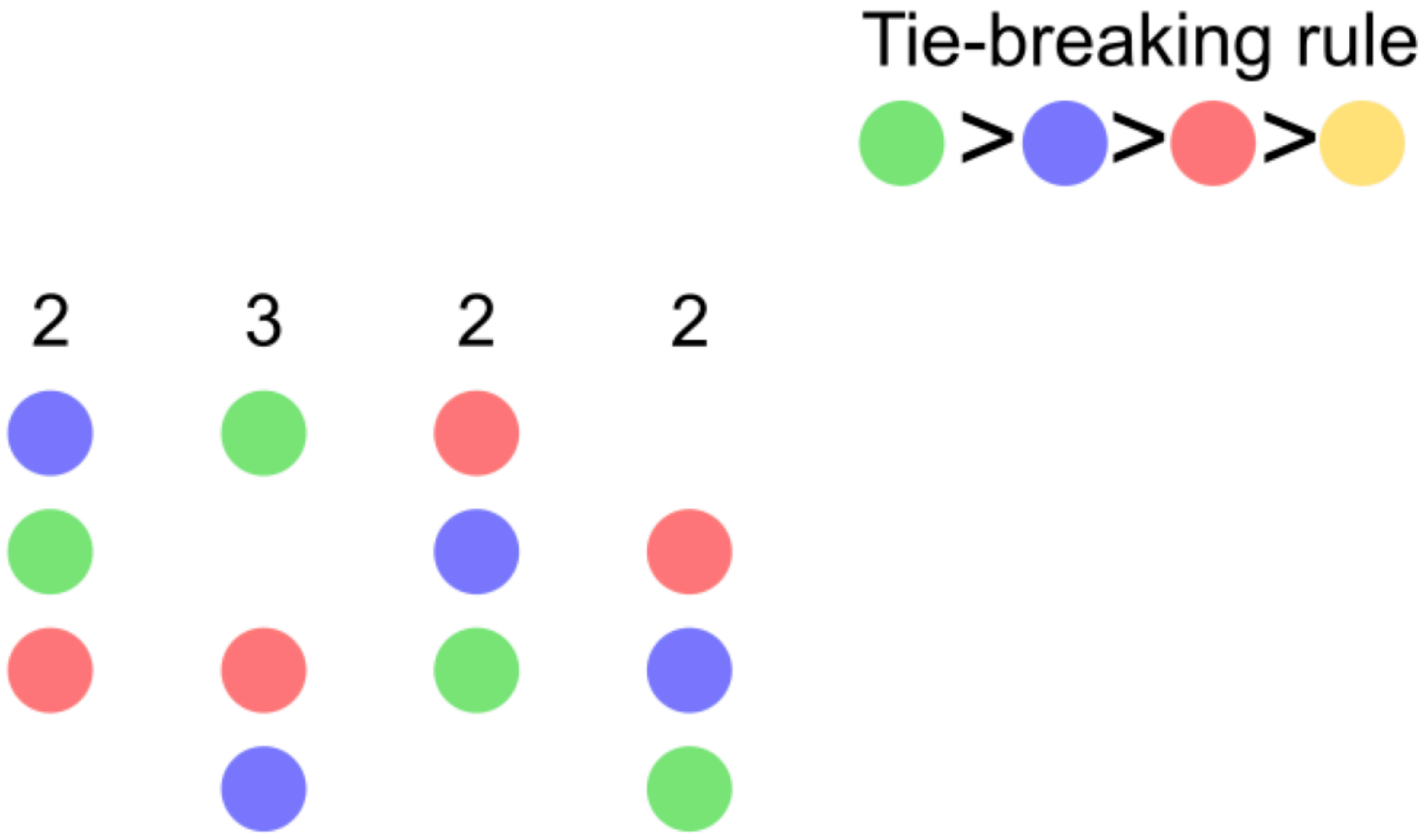
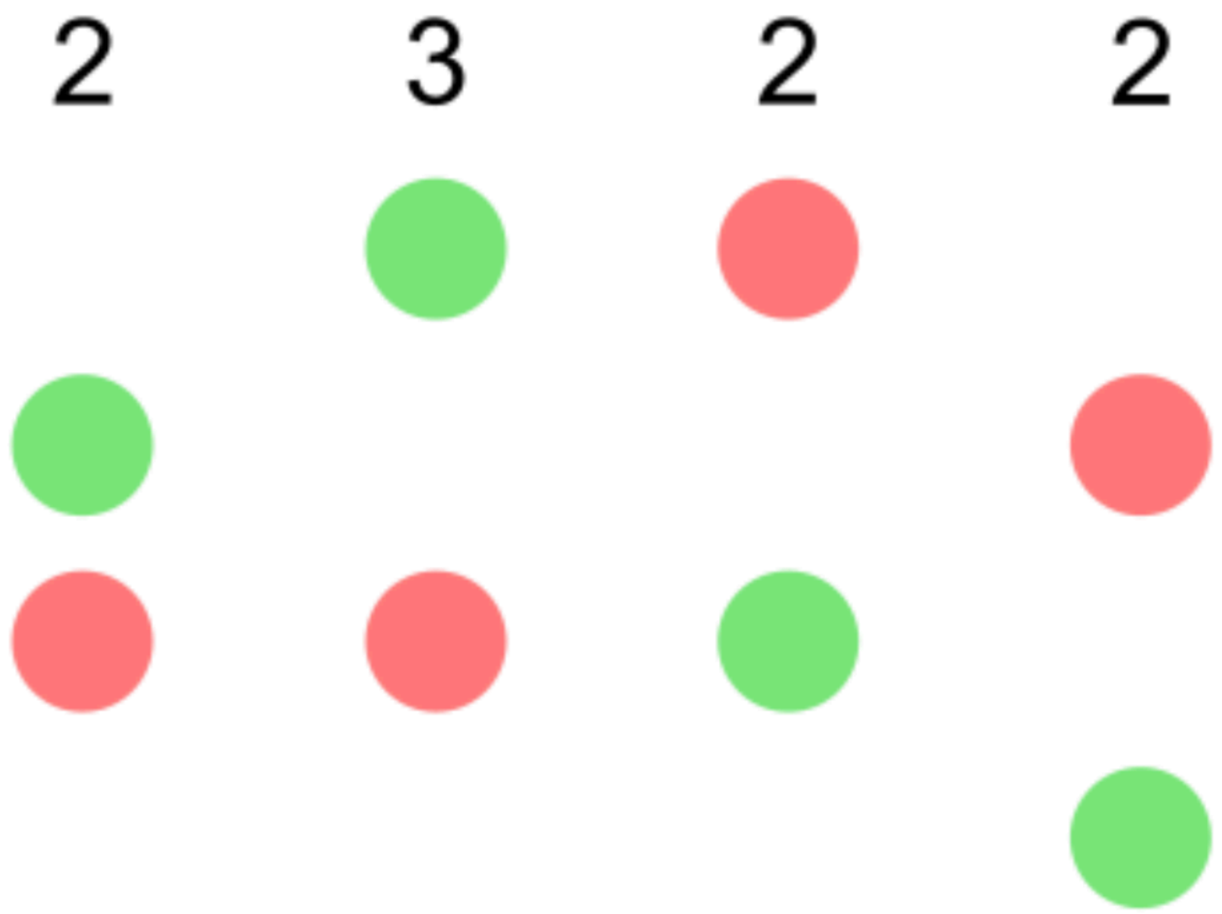


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting



Ranked-choice winner :

Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

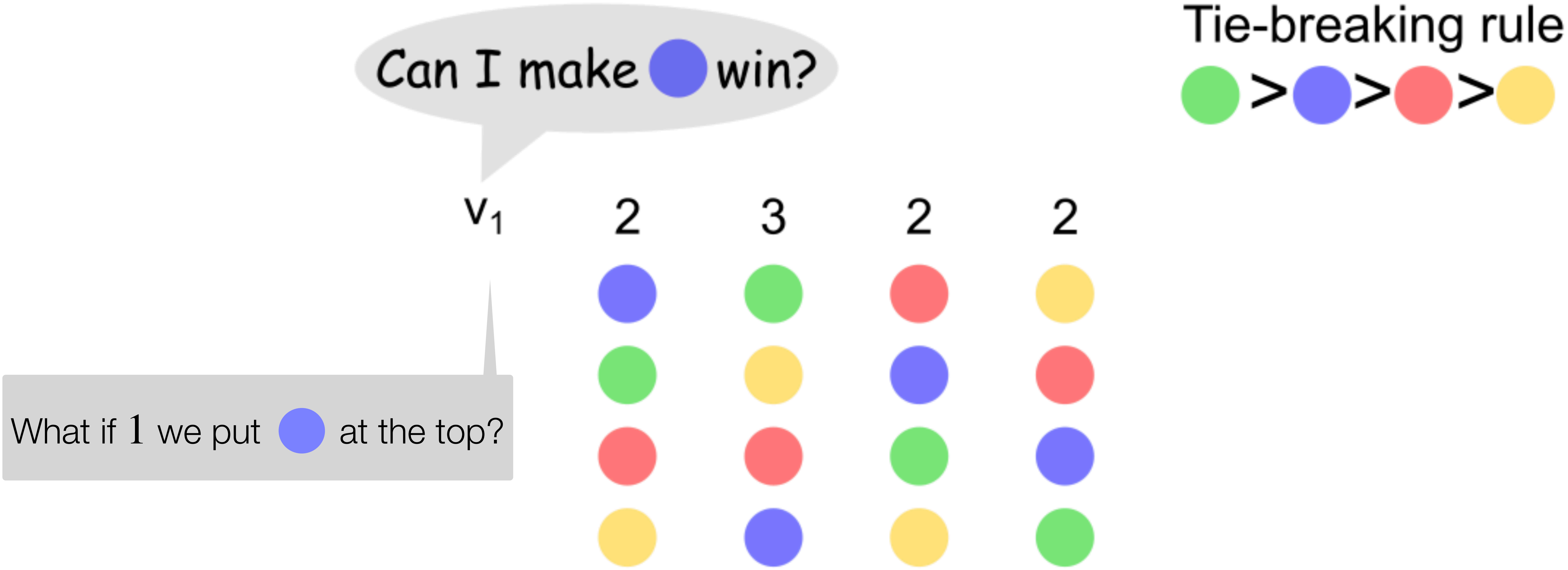
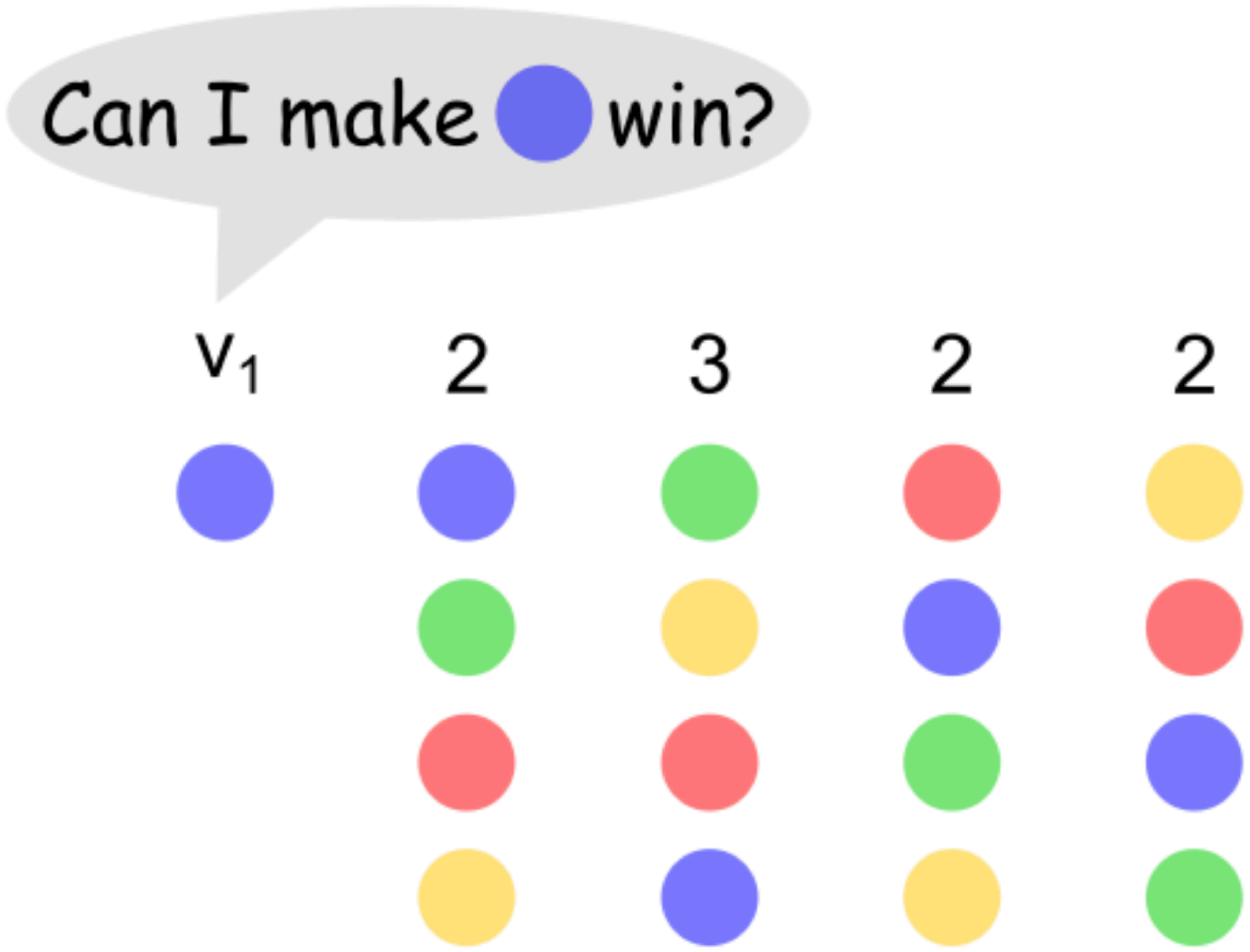


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

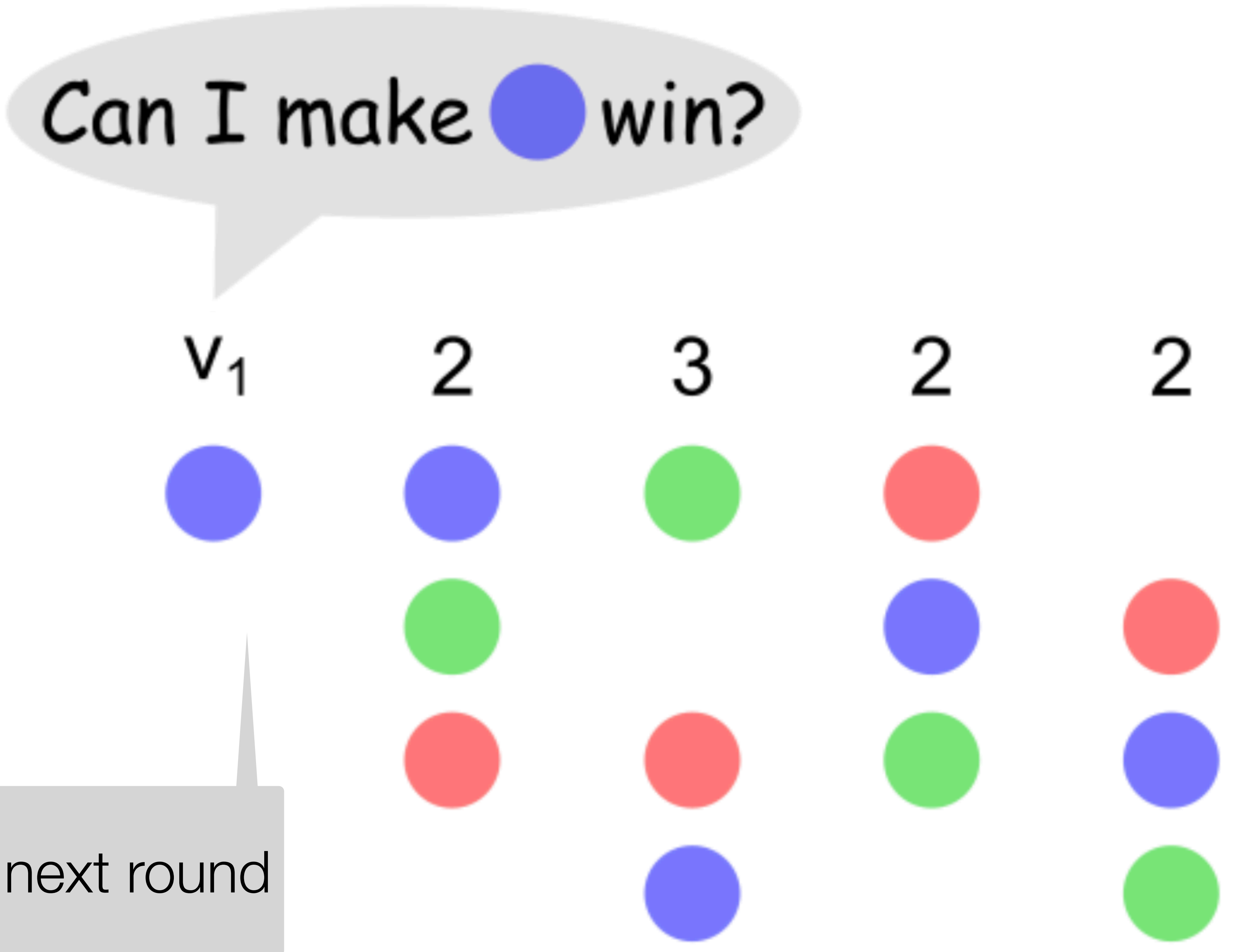


Tie-breaking rule

Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

Tie-breaking rule



eliminated in the next round

Does there exist a way to get to win?

Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

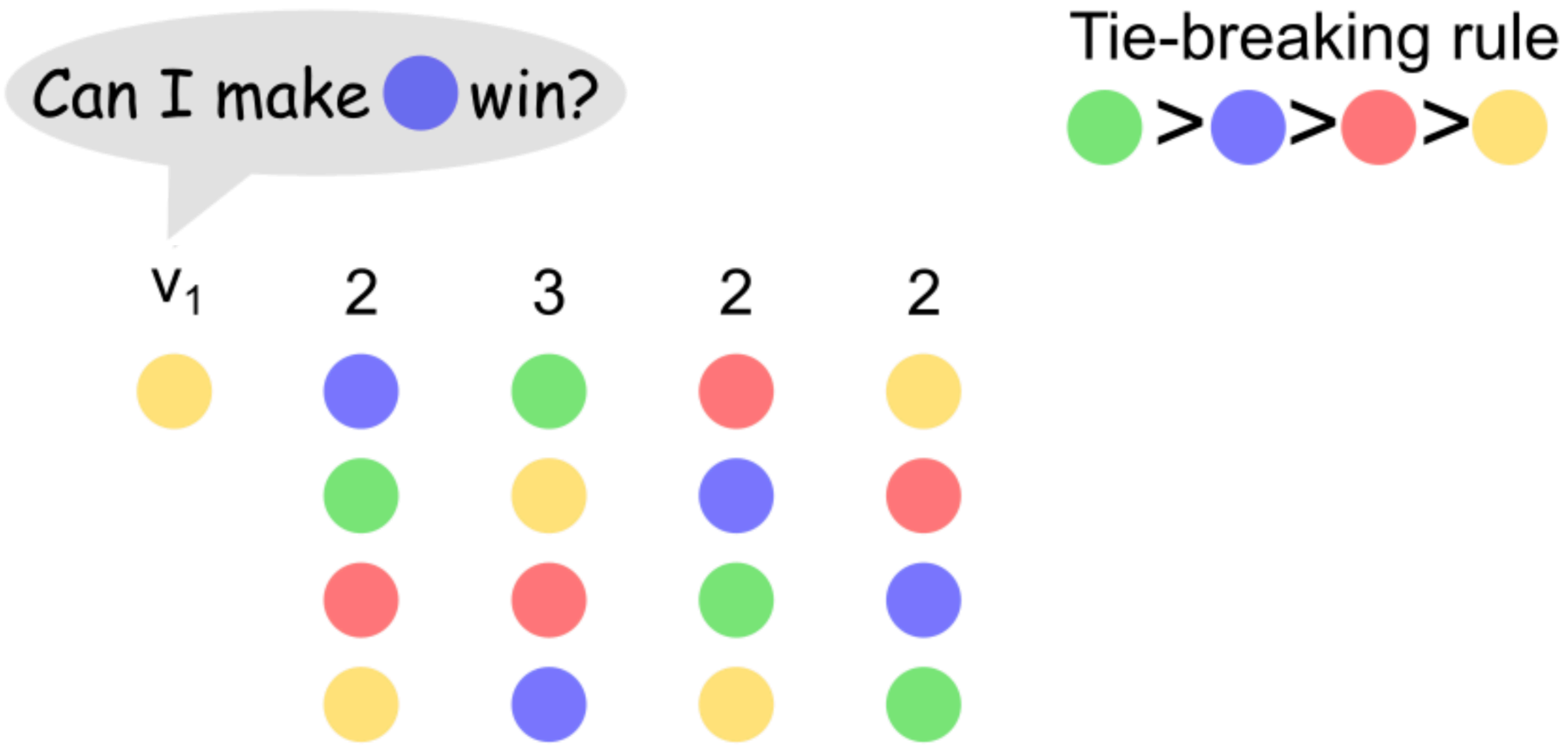


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

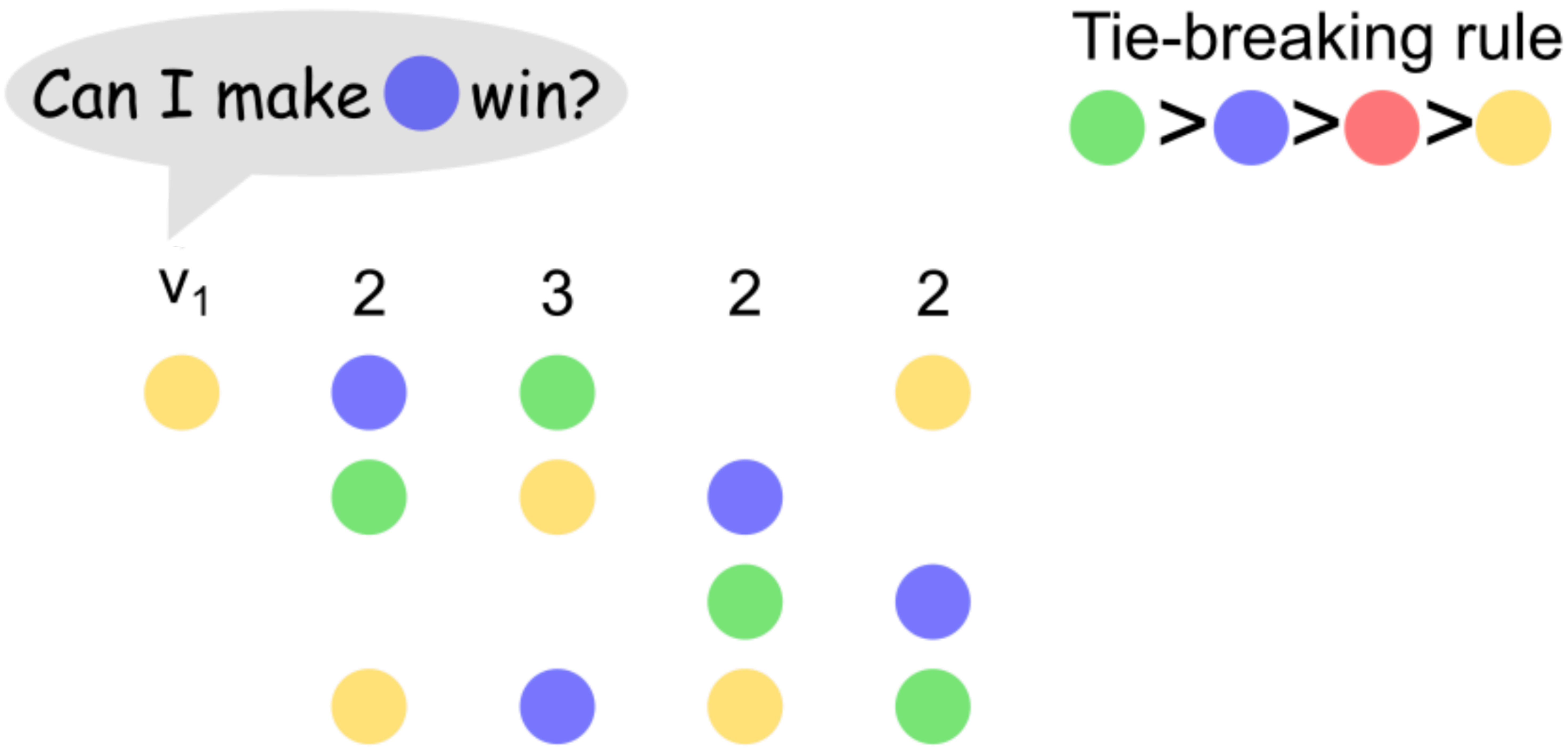


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting

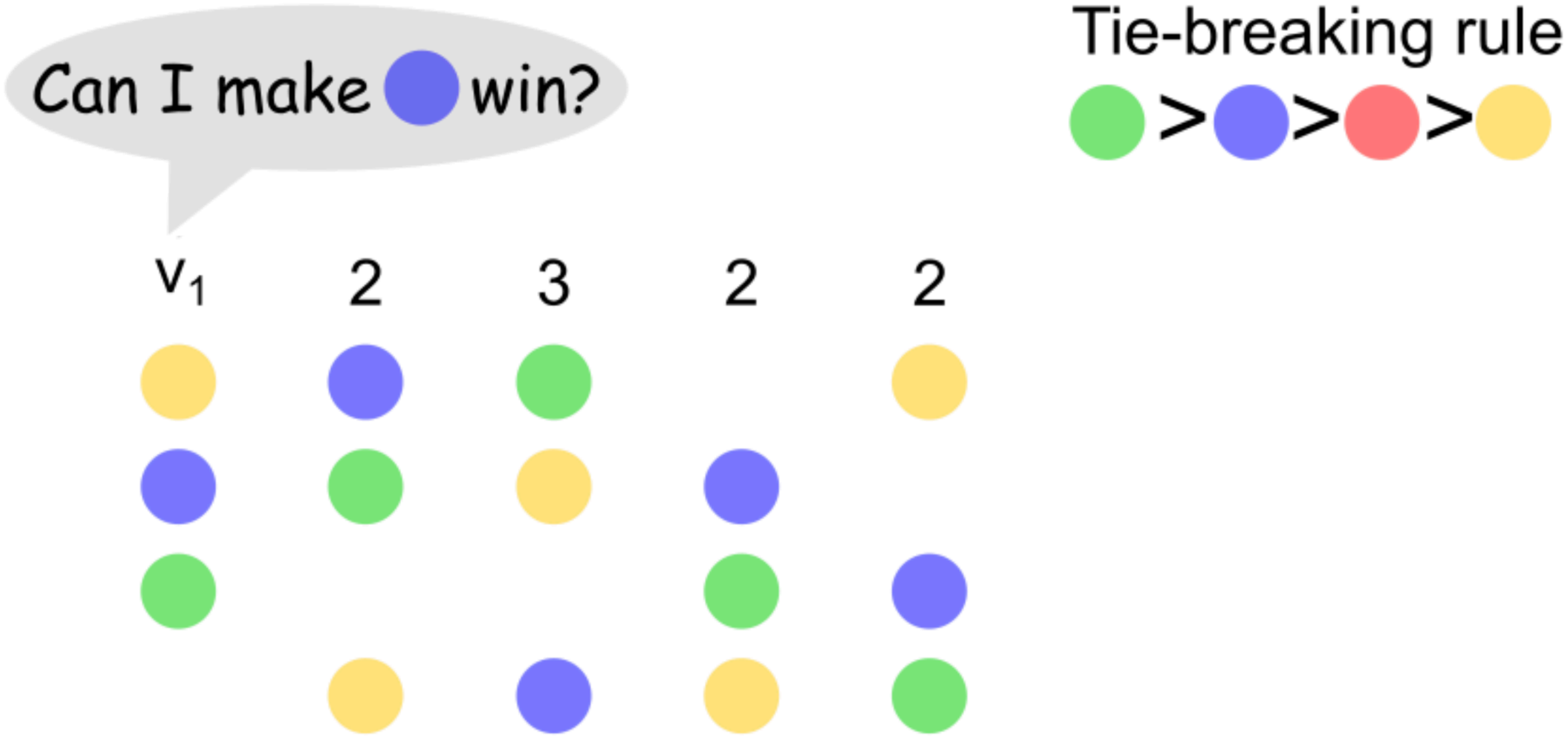
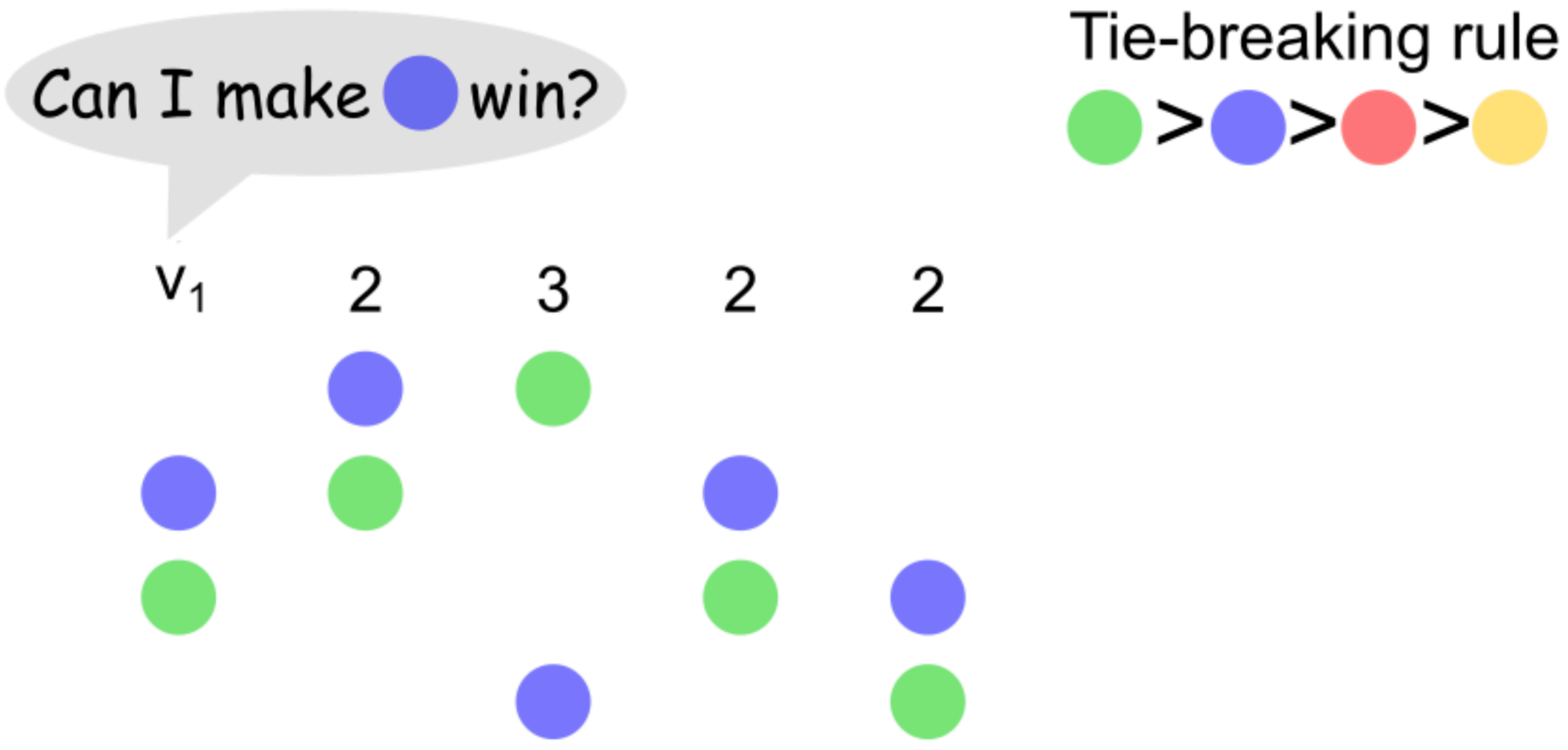


Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

Ranked-Choice Voting



Ranked-choice winner :

Image credit: https://rohitvaish.in/Teaching/2022-Spring/Slides/Lec%202.pdf

When Does Greedy Manipulation Work?

- [Bartholdi, Tovey, Trick '89] Characterized voting rules where f-manipulation is solvable in polynomial time.

The Computational Difficulty of Manipulating an Election*

J. J. Bartholdi III, C. A. Tovey, and M. A. Trick**

School of Industrial and Systems Engineering, Georgia Institute of Technology,
Atlanta, GA 30332, USA

Received June 9, 1987 / Accepted July 29, 1988

Abstract. We show how computational complexity might protect the integrity of social choice. We exhibit a voting rule that efficiently computes winners but is computationally resistant to strategic manipulation. It is *NP*-complete for a manipulative voter to determine how to exploit knowledge of the preferences of others. In contrast, many standard voting schemes can be manipulated with only polynomial computational effort.

When Does Greedy Manipulation Work?

- [Bartholdi et al' 89] Greedy strategy can correctly solve f-manipulation for any voting rule f that can be stated as scoring rule $s(P) \to \mathbf{R}$
  - **Max-score winner:** the candidate with the largest $s(P, i)$ wins
  - **Monotonicity of score**: Suppose a candidate b is preferred over the set S under profile P and S' under P' and suppose $S' \subseteq S'$, then score $s(P, x) \leq s(P', x)$
- Moreover if f can be computed in polynomial time then the manipulation problem is polynomial-solvable
- Turns, out these conditions hold for Plurality and Borda (also Copeland)
  - Copeland rule winner: who beats most others under head-to-head comparison
- Does not hold for Ranked-choice

Ranked-Choice Voting

- f-manipulation is **NP hard** in ranked-choice voting, even if you know everyone's preferences
- Reasonable to assume profitable manipulations are not likely in such a voting rule
- However, NP hardness is a worst-case notion of difficulty
- Most instances are not worst case!

Single transferable vote resists strategic voting

John J. Bartholdi III[1] **and James B. Orlin**[2]

[1] School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA
[2] Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Received December 24, 1990 / Accepted May 12, 1991

Abstract. We give evidence that Single Tranferable Vote (STV) is computationally resistant to manipulation: It is NP-complete to determine whether there exists a (possibly insincere) preference that will elect a favored candiate, even in an election for a single seat. Thus strategic voting under STV is qualitatively more difficult than under other commonly-used voting schemes. Furthermore, this resistance to manipulation is inherent to STV and does not depend on hopeful extraneous assumptions like the presumed difficulty of learning the preferences of the other voters. We also prove that it is NP-complete to recognize when an STV election violates monotonicity. This suggests that non-monotonicity in STV elections might be perceived as less threatening since it is in effect "hidden" and hard to exploit for strategic advantage.

Hardness of Manipulation

- Interesting open problem to design voting rules that are hard to manipulate on average
- Very nice and readable article about manipulation in voting

AI’s War on Manipulation: Are We Winning?

Piotr Faliszewski and Ariel D. Procaccia

"The most controversial part of the approach is that it relies on NP-hardness as a measure of computational difficulty.The issue is that NP-hardness is a worst-case notion and the fact that a problem is NP-hard simply means that it has some dif]cult instances and not that necessarily the ones typically occurring in practice are hard to solve. "

Approximate Approaches

- In the vein of approximate solutions in algorithms, one can try to relax the strategyproofness conditions
  - Consider "milder" notions of incentive compatibility

Approximate Strategyproofness

Benjamin Lubin
School of Management
Boston University
blubin@bu.edu

David C. Parkes
School of Engineering and Applied Sciences
Harvard University
parkes@eecs.harvard.edu

July 24, 2012

Abstract

The standard approach of mechanism design theory insists on equilibrium behavior by participants. This assumption is captured by imposing *incentive constraints* on the design space. But in bridging from theory to practice, it often becomes necessary to relax incentive constraints in order to allow tradeoffs with other desirable properties. This paper surveys a number of different options that can be adopted in relaxing incentive constraints, providing a current view of the state-of-the-art.

Voting in CS Applications

Voting in CS

- In a democracy, voting serves as a way to reach consensus between differing opinions
- In CS, we often use voting as a way to aggregate rankings
    - To recover the "ground truth" from noisy, imperfect estimates
- Voters are effectively cooperating to figure out the objective correct answer: e.g., the true ranking of a set of Web pages by relevance
- Rank-aggregation problem:
    - Given different rank orderings, output a final ranking of alternatives that best captures the input orderings
    - Objective: minimizes some notion of "distance"

Kemeny Rule

- **Kendall tau distance:** the Kendall tau distance between two ranked lists is the total number of rank disagreements over all unordered pairs
- Also called "bubble sort distance”: Kendall tau distance between two ordered lists: number of “swaps” needed to go from one to the other
- For example, consider two ranked lists
    - $L = (b, e, d, a, c)$ and $L' = (b, a, e, d, c)$
- What is the Kendall tau distance between L and L'?
    - Two because they disagree on pairs (a, e) and (a, d)
- **Kemeny rule.** Given preference lists $L = (L_1, \ldots, L_n)$, the Kemeny rule selects a ranked list $L*$ of alternatives that minimizes the Kendall tau distance between $L*$ and L_i summed over all agents i.

Computational Considerations

- **Theorem**. The problem of determining the social rank order in the Kemeny rule is NP hard.
- This isn't really a problem for cases where the number of candidates won't grow too large
- But, Kemeny rule is often used for rank aggregation in CS applications and there scalability is a real concern
- In practice, good heuristics exist to solve this problem
    - Integer linear programming and branch and bound methods

2001

Rank Aggregation Revisited

Cynthia Dwork* Ravi Kumar† Moni Naor‡ D. Sivakumar§

Abstract

The *rank aggregation* problem is to combine many different rank orderings on the same set of candidates, or alternatives, in order to obtain a "better" ordering. Rank aggregation has been studied extensively in the context of social choice theory, where several "voting paradoxes" have been discovered. The problem also arises in many other settings:

Sports and Competition: How to determine the winner of a season, how to rank players or how to compare players from different eras?

Machine Learning: Collaborative filtering and meta-search;

Statistics: Notions of Correlation;

Database Middleware: Combining results from multiple databases.

A natural step toward aggregation was taken by Kemeny. Informally, given orderings $\tau_1, \ldots, \tau_k$ on (partial lists of) alternatives $\{1, 2, \ldots, n\}$, a *Kemeny optimal* ordering σ minimizes the sum of the "bubble sort" distances

$$\Sigma_{i=1}^{k} K(\sigma, \tau_i) .$$

Thus, intuitively, Kemeny optimal solutions produce "best" compromise orderings. However, finding a Kemeny optimal aggregation is NP-hard [4].

In this work we revisit rank aggregation with an eye toward reducing search engine spam in meta-search. We note the virtues of Kemeny optimal aggregation in this context, strengthen the NP-hardness results, and, most importantly, develop a natural relaxation called *local Kemeny optimality* that preserves the spam-fighting capabilities of Kemeny optimality at vastly reduced cost. We show how to efficiently take *any* initial aggregated ordering and produce a maximally consistent locally Kemeny optimal solution.

We therefore propose a new approach to rank aggregation: begin with any desirable initial aggregation and then "locally Kemenize" it. We also propose the use of Markov chains for obtaining the initial aggregation, and suggest four specific chains for this purpose.

2020

How to aggregate Top-lists: Approximation algorithms via scores and average ranks

Claire Mathieu * clairemmathieu@gmail.com

Simon Mauras * simon.mauras@irif.fr

Abstract

A top-list is a possibly incomplete ranking of elements: only a subset of the elements are ranked, with all unranked elements tied for last. Top-list aggregation, a generalization of the well-known rank aggregation problem, takes as input a collection of top-lists and aggregates them into a single complete ranking, aiming to minimize the number of upsets (pairs ranked in opposite order in the input and in the output). In this paper, we give simple approximation algorithms for top-list aggregation.

AN ALGORITHMIC VIEW OF VOTING*

2016

RONALD FAGIN†, RAVI KUMAR‡, MOHAMMAD MAHDIAN‡, D. SIVAKUMAR‡, AND ERIK VEE‡

Abstract. We offer a novel classification of voting methods popular in social choice theory. Our classification is based on the more general problem of *rank aggregation* in which, beyond electing a winner, we also seek to compute an aggregate ranking of all the candidates; moreover, our classification is offered from a computational perspective—based on whether or not the voting method generalizes to an aggregation algorithm guaranteed to produce solutions that are near optimal in minimizing the distance of the aggregate ranking to the voters' rankings with respect to one of three well-known distance measures: the Kendall tau, the Spearman footrule, and the Spearman rho measures. We show that methods based on the average rank of the candidates (Borda counting), on the median rank of the candidates, and on the number of pairwise-majority wins (Copeland) all satisfy the near-optimality criterion with respect to each of these distance measures. On the other hand, we show that natural extensions of each of plurality voting, single transferable voting, and Simpson–Kramer minmax voting do not satisfy the near-optimality criterion with respect to these distance measures.

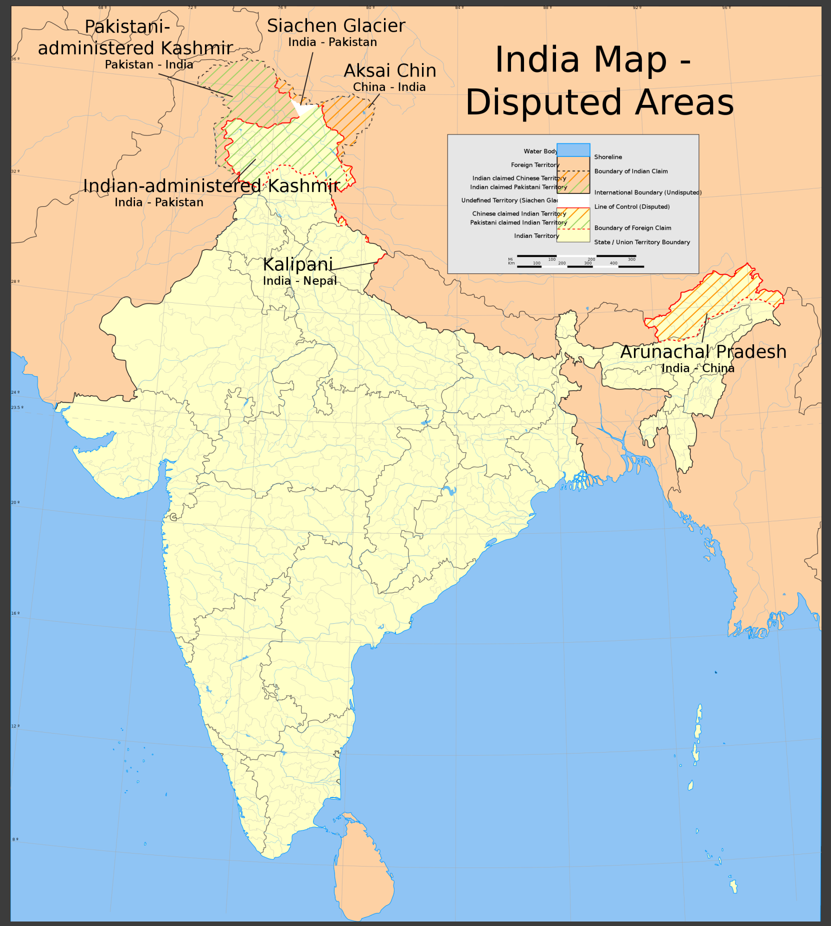
Comparison of preferential electoral systems [hide]

System	Monotonic	Condorcet winner	Majority	Condorcet loser	Majority loser	Mutual majority	Smith	ISDA	LIIA	Independence of clones	Reversal symmetry	Participation, consistency	Later-no-harm	Later-no-help	Polynomial time	Resolvability
Schulze	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No	No	Yes	Yes
Ranked pairs	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes	Yes
Tideman's Alternative	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	No	No	Yes	Yes
Kemeny–Young	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	No	No	Yes
Copeland	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes	No	No	No	Yes	No
Nanson	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes	No	No	No	Yes	Yes
Black	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	Yes	No	No	No	Yes	Yes
Instant-runoff voting	No	No	Yes	Yes	Yes	Yes	No	No	No	Yes	No	No	Yes	Yes	Yes	Yes
Smith/IRV	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	No	No	Yes	Yes
Borda	Yes	No	No	Yes	Yes	No	No	No	No	No	Yes	Yes	No	Yes	Yes	Yes
Baldwin	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	Yes	Yes
Bucklin	Yes	No	Yes	No	Yes	Yes	No	No	No	No	No	No	No	Yes	Yes	Yes
Plurality	Yes	No	Yes	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes
Contingent voting	No	No	Yes	Yes	Yes	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Coombs[4]	No	No	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No	Yes	Yes
MiniMax	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	Yes	Yes
Anti-plurality[4]	Yes	No	No	No	Yes	No	No	No	No	No	No	Yes	No	No	Yes	Yes
Sri Lankan contingent voting	No	No	Yes	No	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Supplementary voting	No	No	Yes	No	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Dodgson[4]	No	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	No	Yes

Fair Division

Fair Division

Divisible



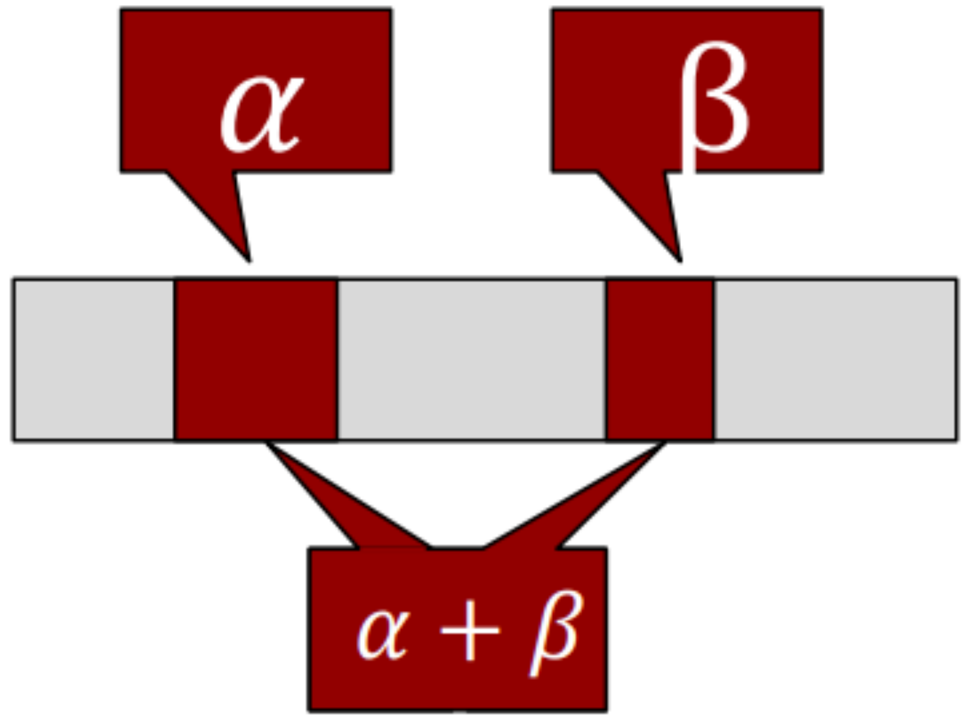
Indivisible

Cake Cutting Problems

- Fairly dividing a **heterogeneous**, **divisible** resource among agents with differing preferences
  - **heterogenous:** equal amounts of the resource can have different values for different agents
  - **divisible**: any fractional allocation is feasible
- Resource is often a cake (hence the name)
- In practice, can be processing time on a compute cluster (with some times of the day more valuable than others)

Fair Division Model

- **Line representation.** Let the good/cake be the unit interval $[0,1]$
- Each player i has a valuation function v_i: the value $v_i(S)$ for any subset
  - Assume v_i is normalized with $v_i([0,1]) = 1$
  - v_i is additive on disjoint subsets: $v_i(A) + v_i(B) = v_i(A \cup B)$
- Goal is a fair division, we need a notion of fairness
- **Proportional.** An allocation $A_1, \ldots, A_n$ of cake to n players is proportional if $v_i(A_i) \geq 1/n$ for every player
- **Envy free.** An allocation $A_1, \ldots, A_n$ of cake to n players is envy free if $v_i(A_i) \geq v_i(A_j)$ for every pair i, j of players
    - Envy free $\Longrightarrow$ Proportionality (stronger notion)

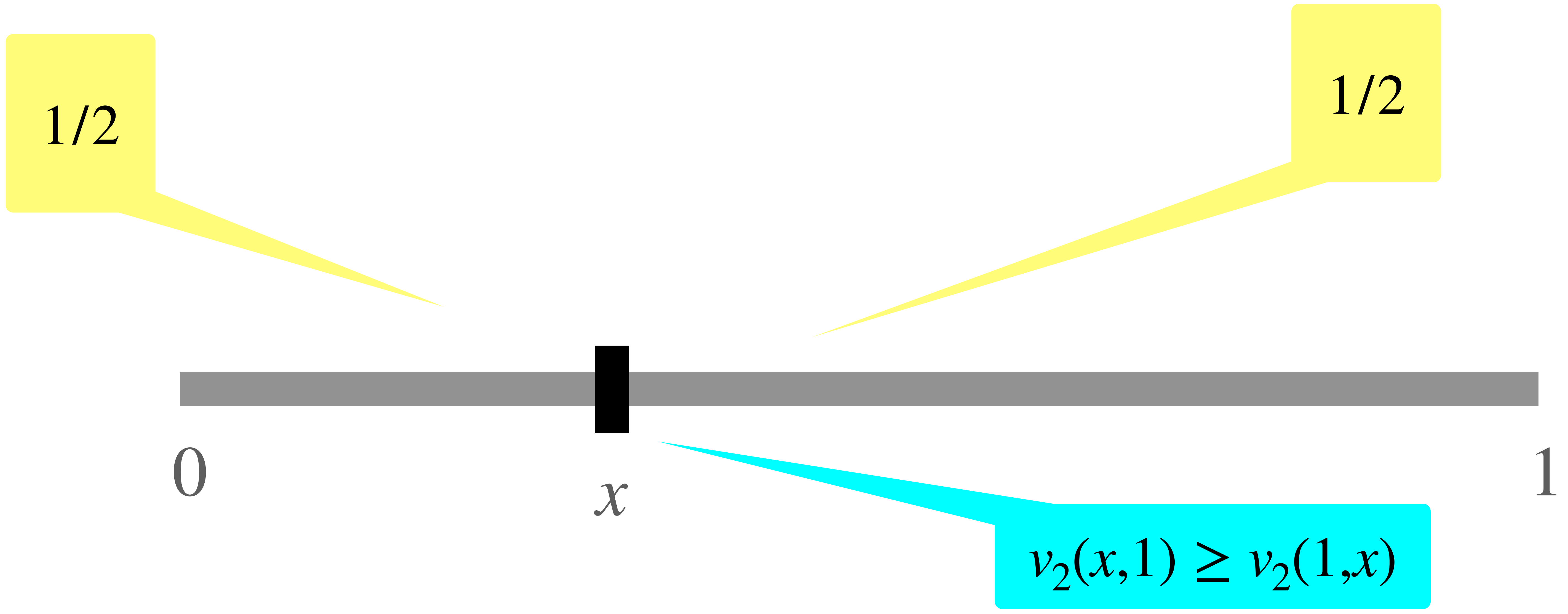


Two Agents

- Suppose we only have two agents, can you suggest a natural protocol that is proportional and envy free
  - Both properties are equivalent for $n = 2$ case

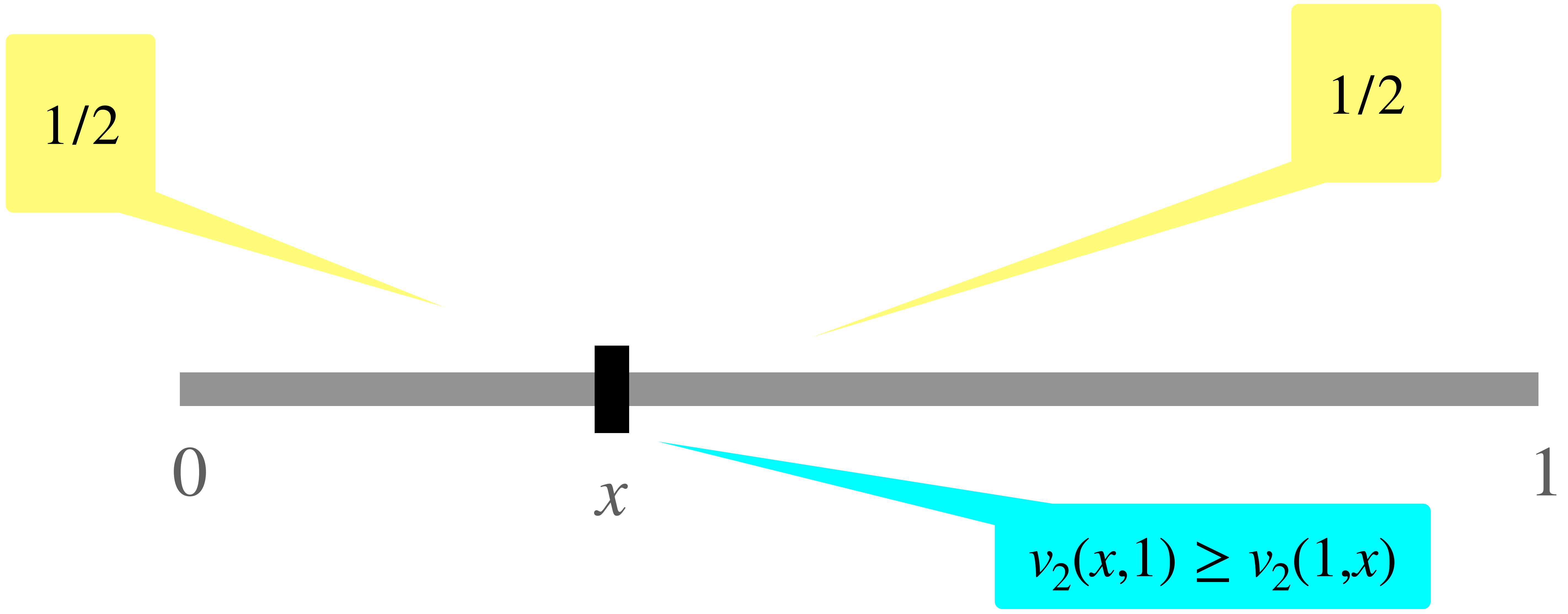
Two Agents: Cut and Choose

- Player 1 splits the good into two equally-valuable pieces A and B (according to v_1)
- Player 2 picks whichever A, B she likes better (according to v_2)



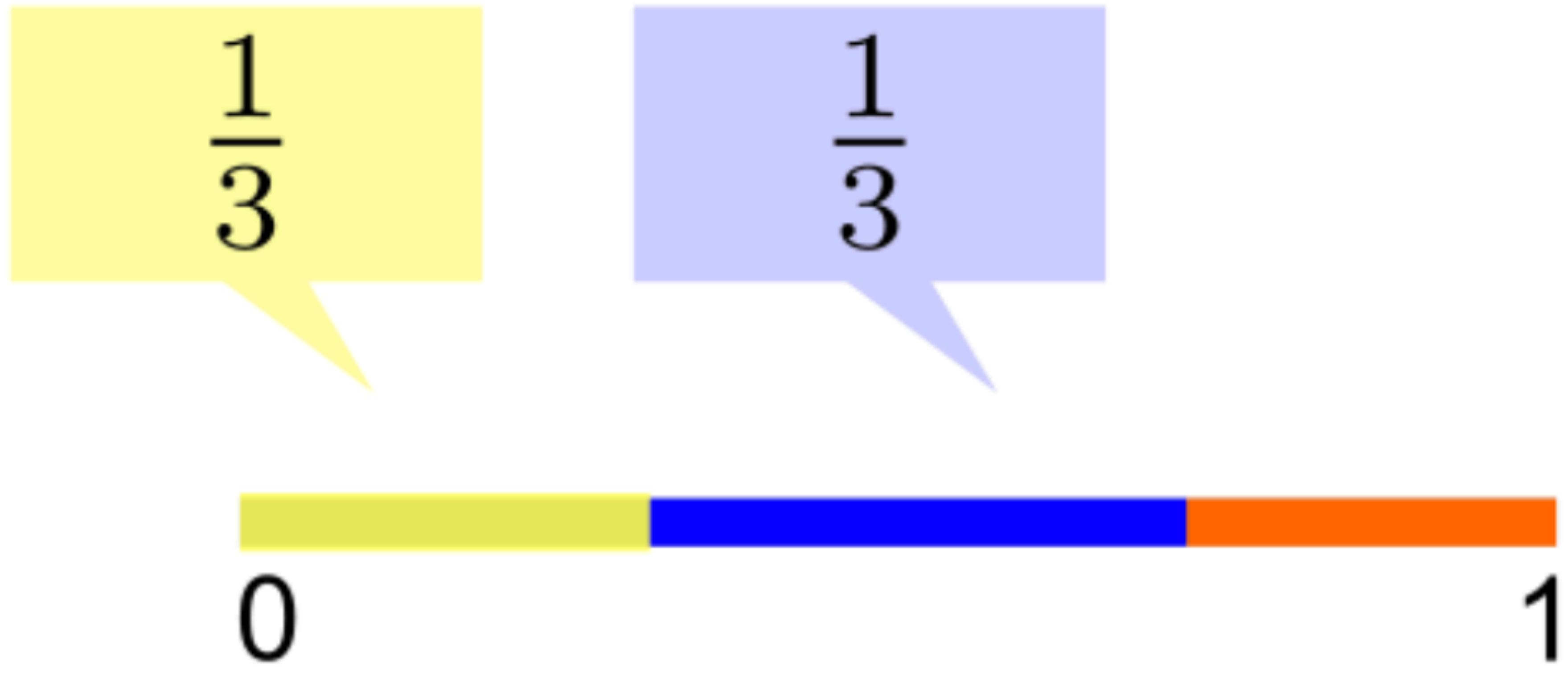
Cut and Choose Protocol

- Is proportional: player 1 gets exactly half, player 2 gets at least $1/2$
- Envy free: player 2 gets favorite piece, player 1 values each the same



Proportionality: n players

- A referee gradually moves the knife from left to right
- As soon as the knife reaches a point s.t. the piece to the left is equal to $1/n$ of some players value
  - Give the piece to that player
  - Delete that player and its share and recurse
  - (Ties are broken in a coordinated way)
- Why is this proportional?
  - Every player except last gets $1/n$
  - Last player gets at least $1/n$



Three Players: Envy Free

- Even with three players, guaranteeing envy-free ness gets tricky
- 3 player case: Selfrige and Conway's protocol
- Nice exposition in:
  https://www.quantamagazine.org/new-algorithm-solves-cake-cutting-problem-20161006/


ALGORITHMS

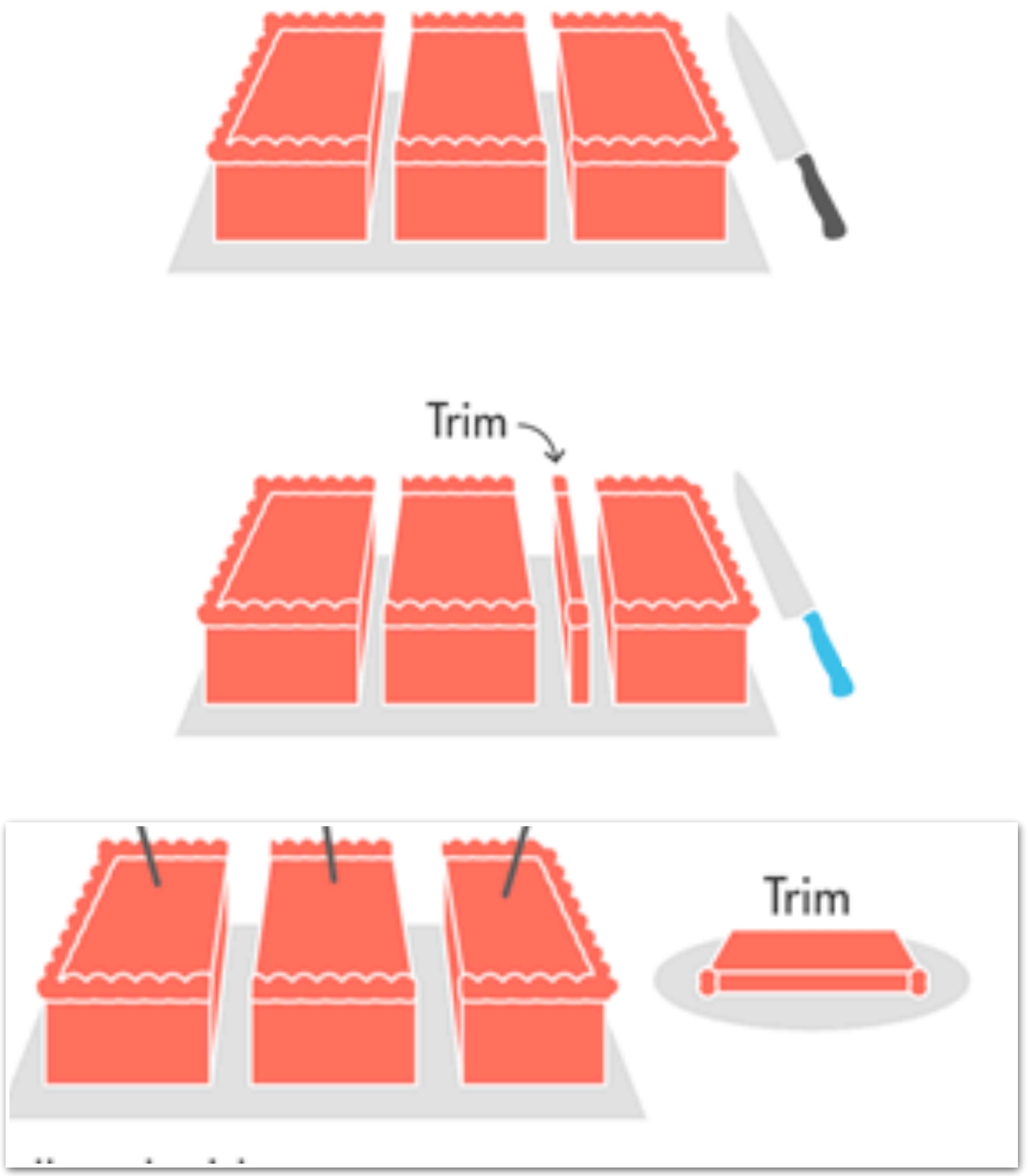
How to Cut Cake Fairly and Finally Eat It Too

31

Computer scientists have come up with an algorithm that can fairly divide a cake among any number of people.

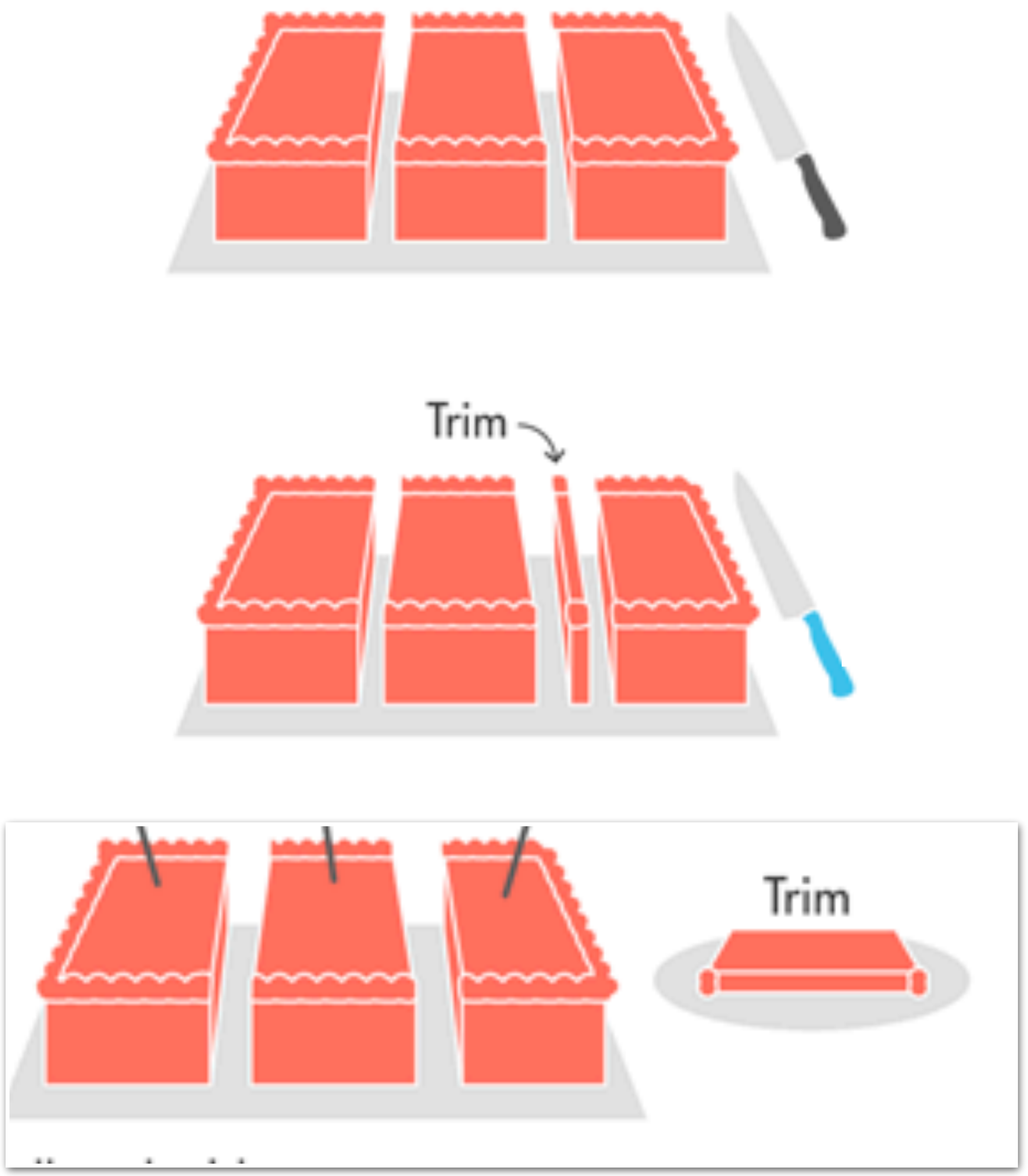
Selfridge & Conway: 3 Players

- **Phase 1**.
  - A divides the cake into 3 equal pieces (according to v_A)
  - B trims its favorite piece to create a tie with its favorite
    - M: main cake, S: trim
    - Now agents choose their favorite piece from M in the order:
      - C, then B, then A
      - Condition: B must choose trimmed piece if C does not
      - Let T be owner of trimmed piece (has to be C or B)
      - Let T' be other among them, that is, $T' = (B \cup C) \backslash T$



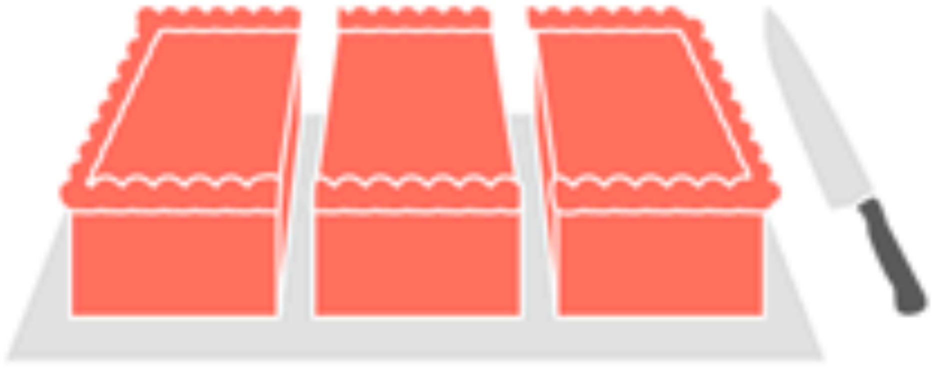
Selfridge & Conway: 3 Players

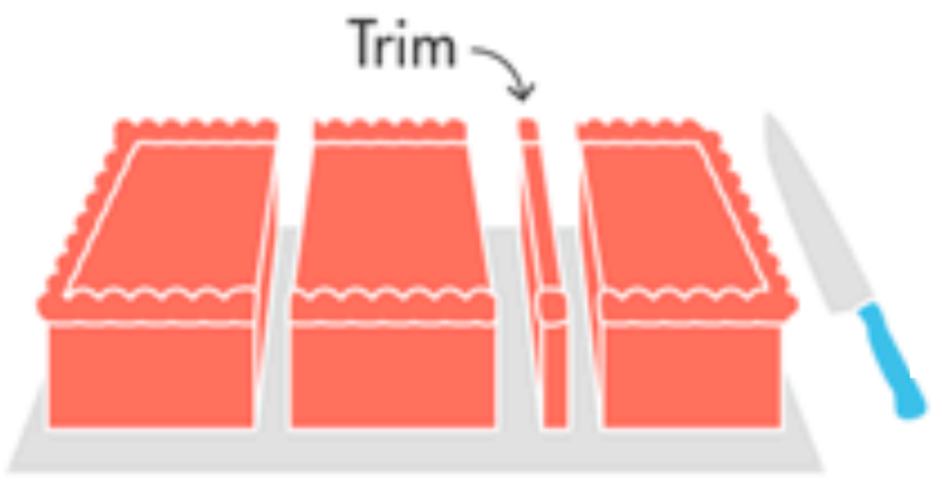
- **Phase 2**.
  - T' divides trim S into 3 equal pieces (according to $v_{T'}$)
  - The agents pick their favorite remaining piece from S in order:
    - T, then A, then T'
- Is every part of the cake allocated?
- Is this division, envy free for C?
  - In M, C gets first pick
  - In S, if $C = T'$, each piece is equal
  - In S, if $C = T$, then picks first

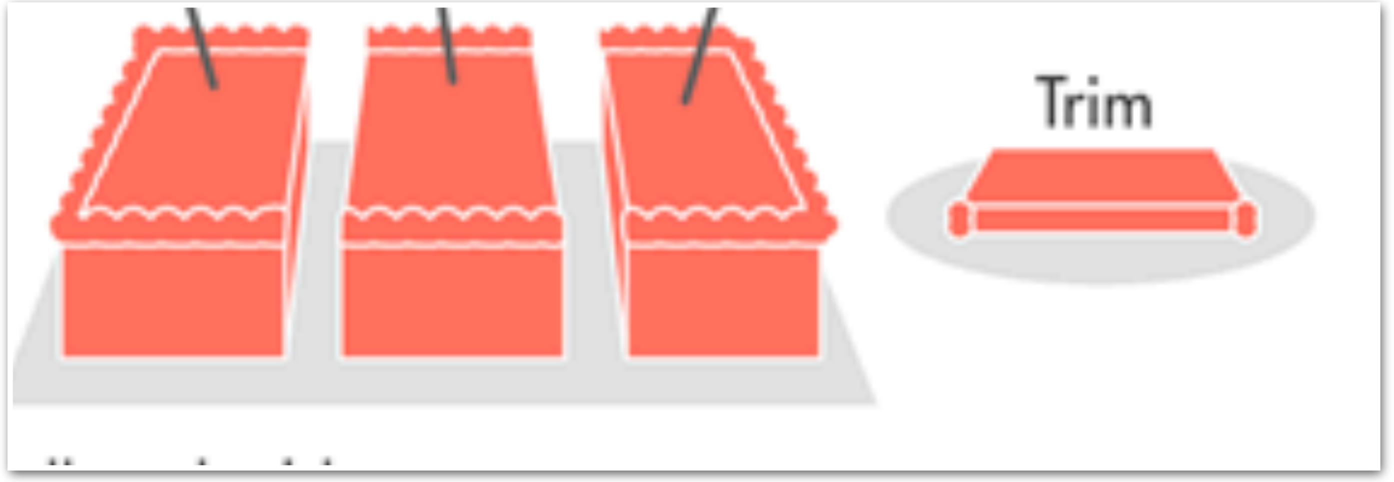


Selfridge & Conway: 3 Players

- Is this division envy free for B?
    - In main cake, B has two pieces of equal value, so does not envy C who goes first
    - Does not envy A because chooses before A
- In trim S, cases:
    - If $B = T'$, cuts S into equal pieces
    - If $B = T$, then chooses first from S
- Finally, lets think about A who goes first in Phase 1
    - Envy free piece in M (never gets trimmed piece)
    - Why envy free piece in S (goes before T'

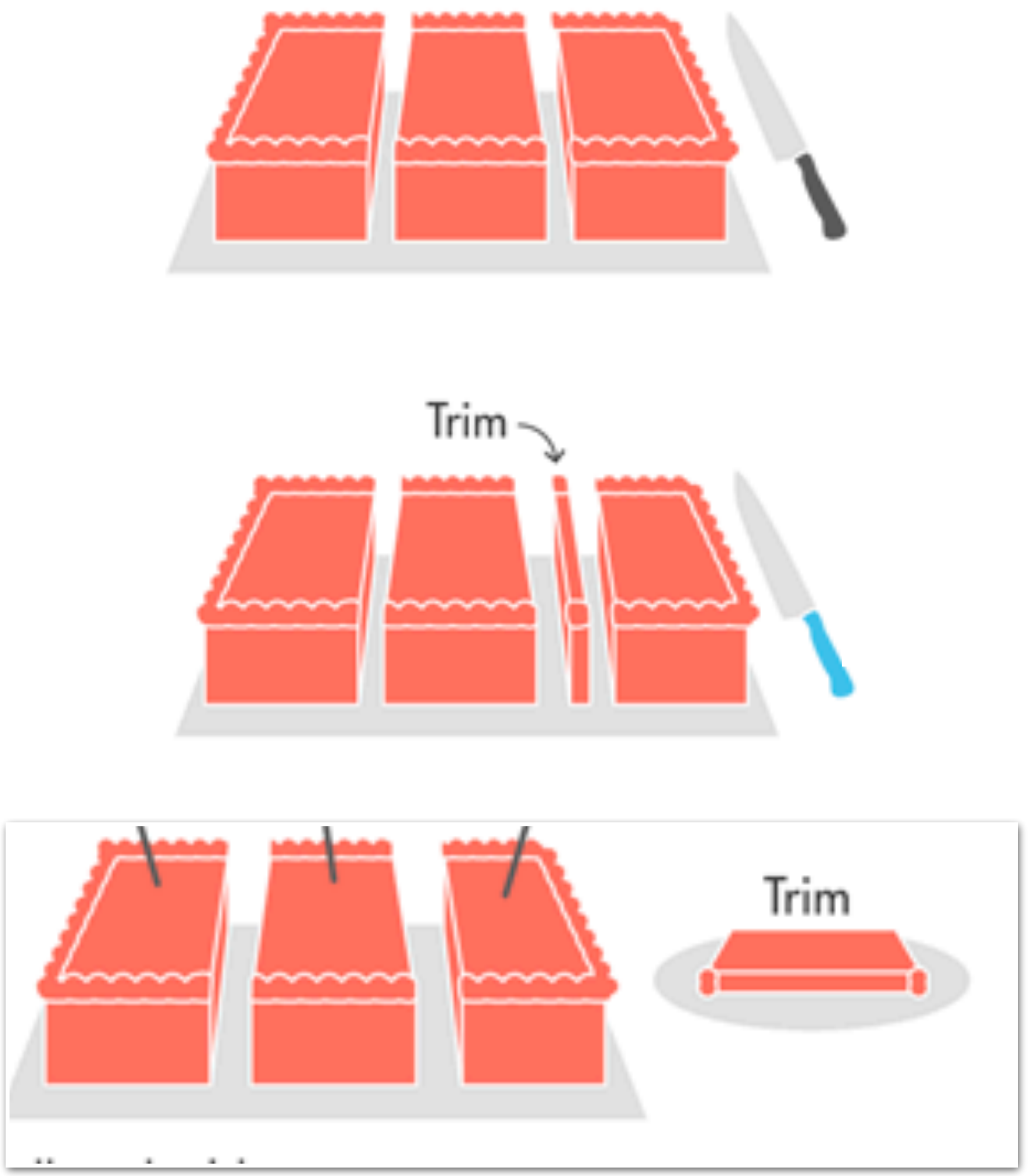






Selfridge & Conway: 3 Players

- Finally, lets think about A who goes first in Phase 1
- Envy free wrt M
    - Was the cutter and never gets trimmed piece
- Envy free wrt S
    - Does not go envy T' because chooses before T'
    - Does not envy T, why?
        - Irrevocable advantage from Phase 1



Story: Envy Free Cake Cutting

- **Question**. Given $n > 3$ agents, does there exist an envy-free cake cutting algorithm?
- [Brams and Taylor '95]. gave am envy-free protocol for any number of players but the number of steps were unbounded; depending on the choice of valuations, the protocol not guaranteed to terminate
- **Next open problem**: is there an envy-free protocol that terminates in $f(n)$ steps, where n is the number of players
    - Big open question for a couple of decades; many experts believed that no such protocol existed
- Recently in 2016, breakthrough result by Aziz & Mackenzie
    - Gave a 4-player protocol that terminated in at most 203 cuts
    - Extended the result to n players, can you guess the number of cuts needed?

Envy Free Cake Cutting: n Players

- For the n-player case, the best known upper bound on the number of cuts is

$$n^{n^{n^{n^{n^{n}}}}}$$

- It is a tower of 6 n's!
- As for lower bound on the number of cuts
  - The best known is $\Omega(n^2)$ [Procaccia 2009]
- **Open problem**. Can we do better in any direction?
- Is it possible to find a polynomial time algorithm for envy-free cake cutting?

Sequential Games