STCS Vigyan Vidushi 2025

Computational Barriers to Manipulation

Rohit Vaish

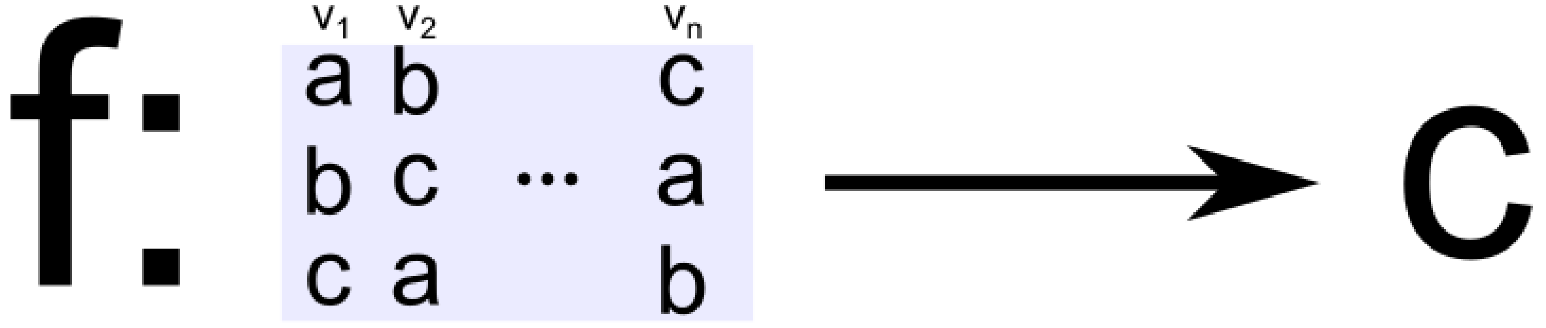
Last Time

[Gibbard'73; Satterthwaite'75]

Any onto and non-dictatorial voting rule must be manipulable.

VOTING RULE

A mapping from preference profiles to candidates.



f-Manipulation

f-Manipulation

Input:

- A set of candidates and a set of voters $v_1, v_2, \ldots, v_n$

f-Manipulation

Input:

- A set of candidates and a set of voters $v_1, v_2, \ldots, v_n$
- Votes $P_2, \ldots, P_n$ of all non-manipulating voters $v_2, \ldots, v_n$

f-Manipulation

Input:

- A set of candidates and a set of voters $v_1, v_2, \ldots, v_n$
- Votes $P_2, \ldots, P_n$ of all non-manipulating voters $v_2, \ldots, v_n$
- Manipulator v_1's favorite candidate c

f-Manipulation

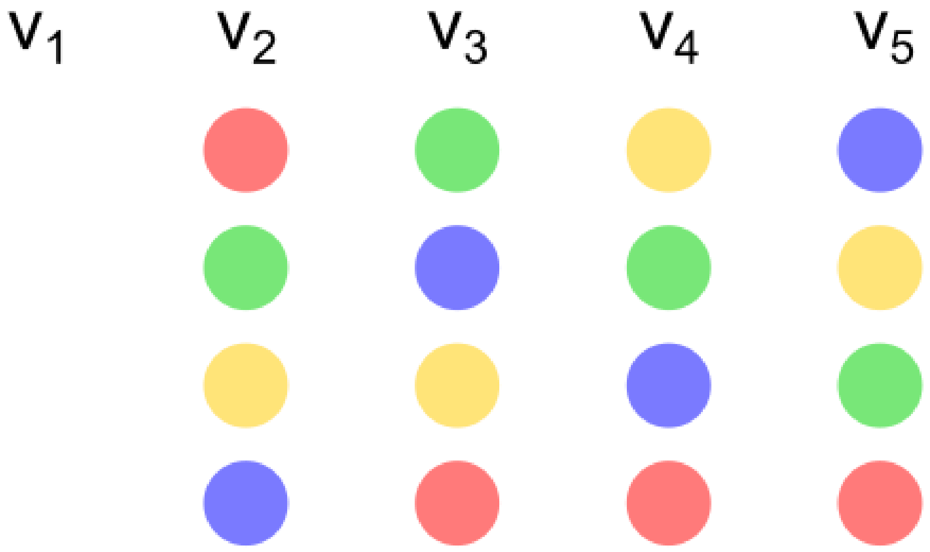
Input:

- A set of candidates and a set of voters $v_1, v_2, \ldots, v_n$
- Votes $P_2, \ldots, P_n$ of all non-manipulating voters $v_2, \ldots, v_n$
- Manipulator v_1's favorite candidate c

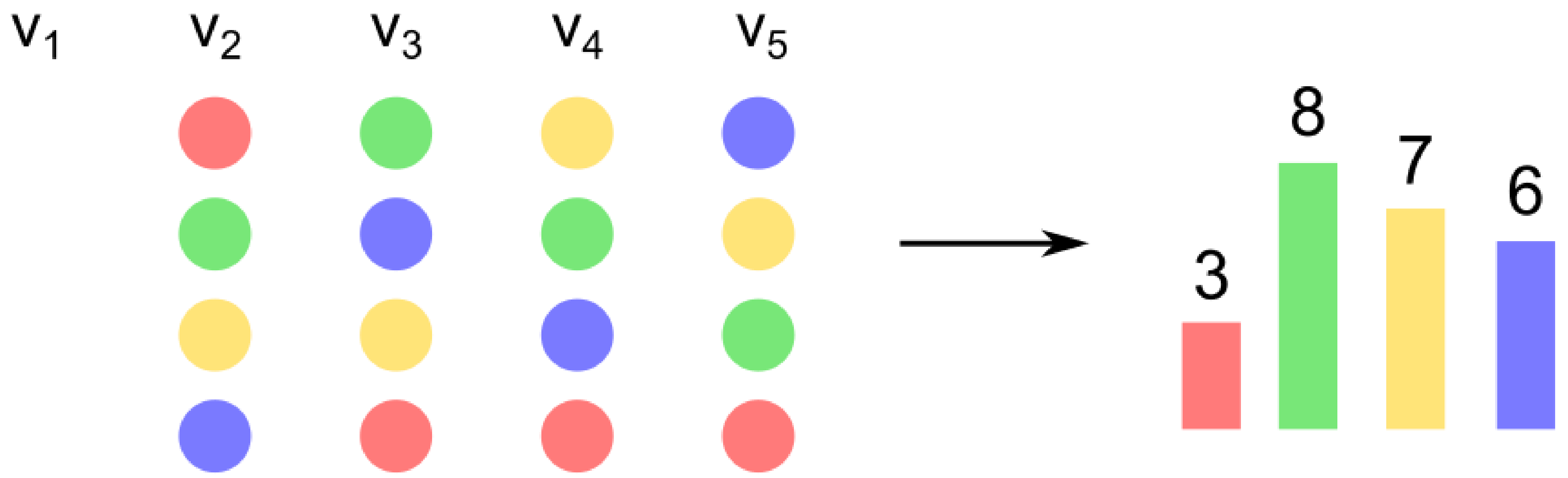
Question:

Does there exist a vote P_1 of the manipulator v_1 such that $f(P_1, P_2, \ldots, P_n) = c$?

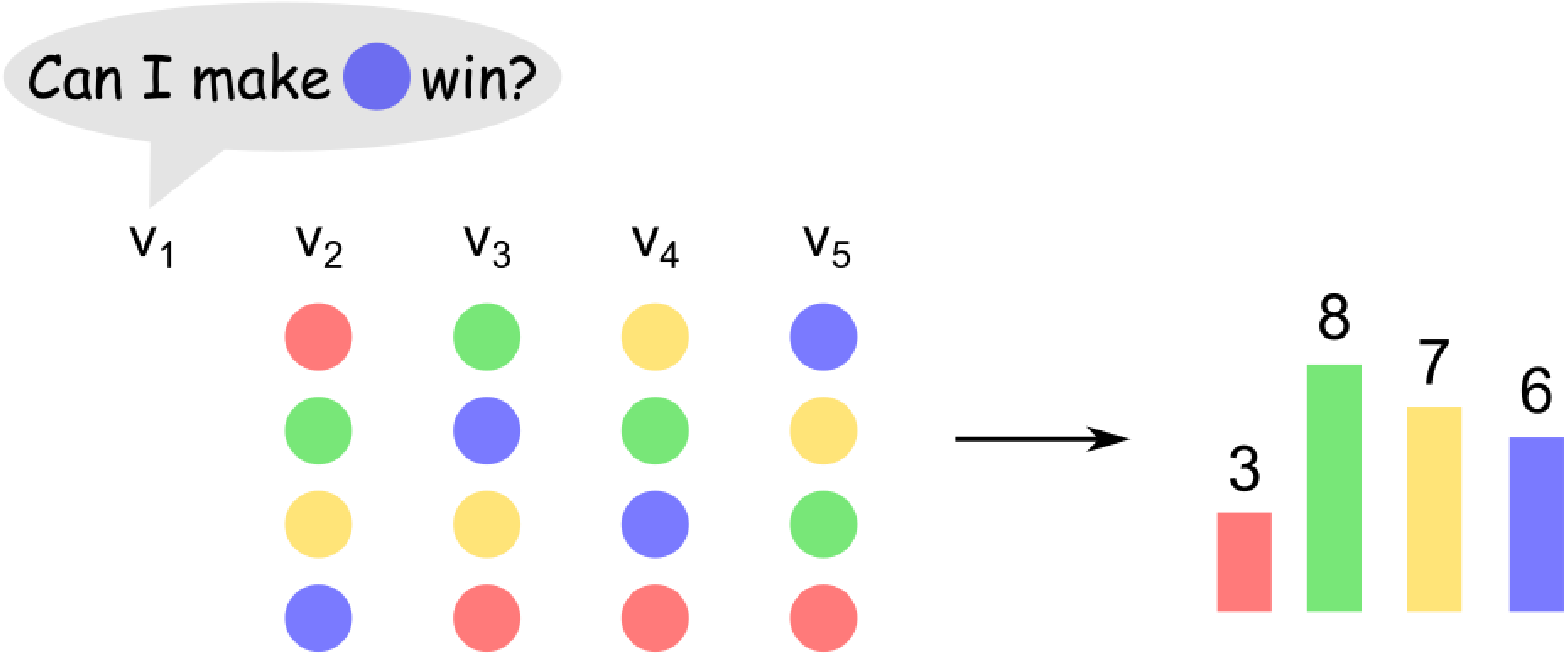
Manipulation under Borda Count



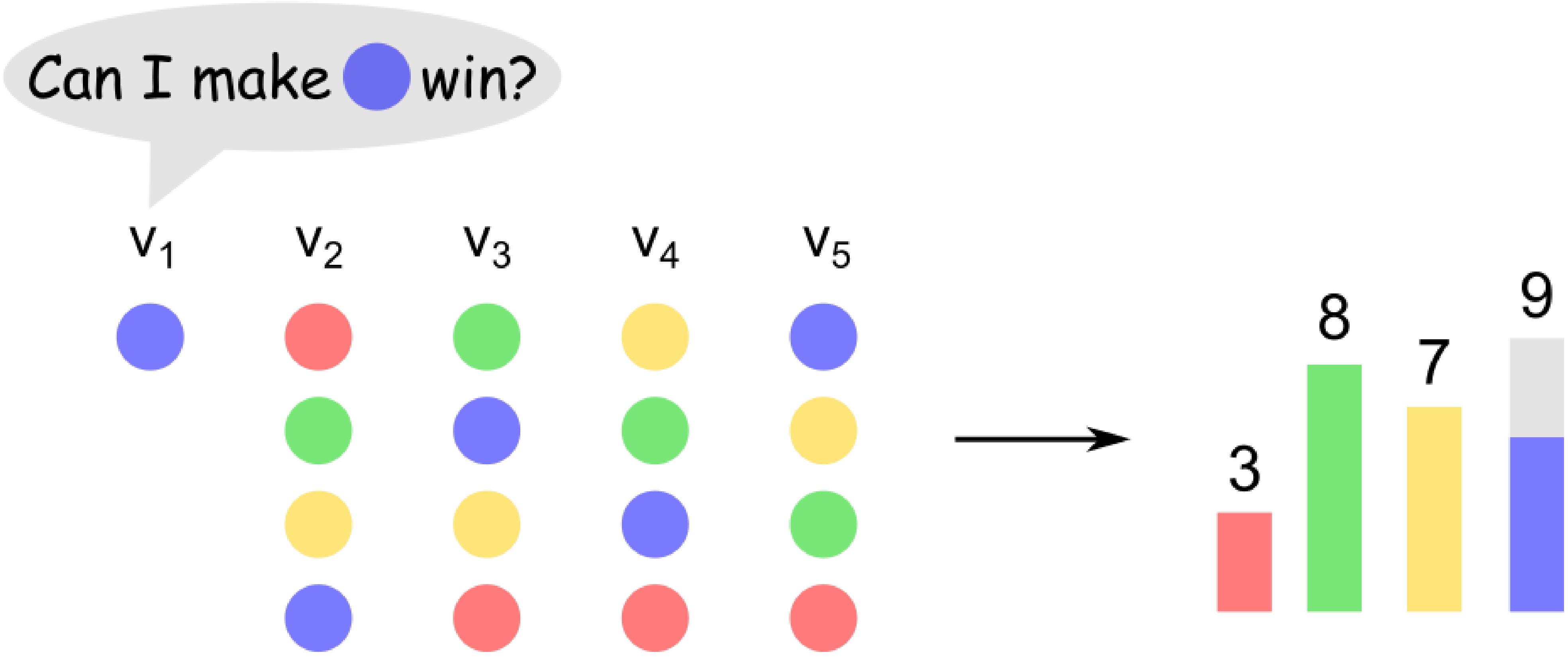
Manipulation under Borda Count



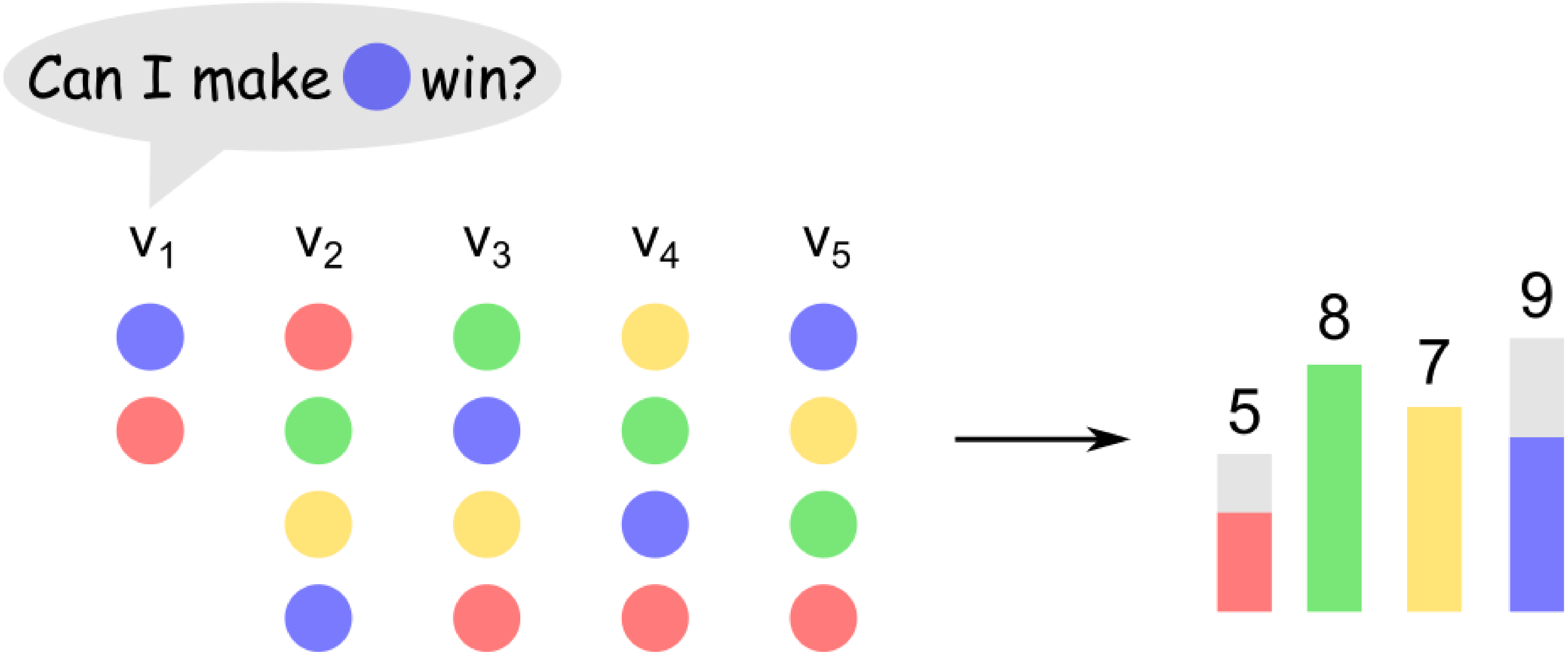
Manipulation under Borda Count



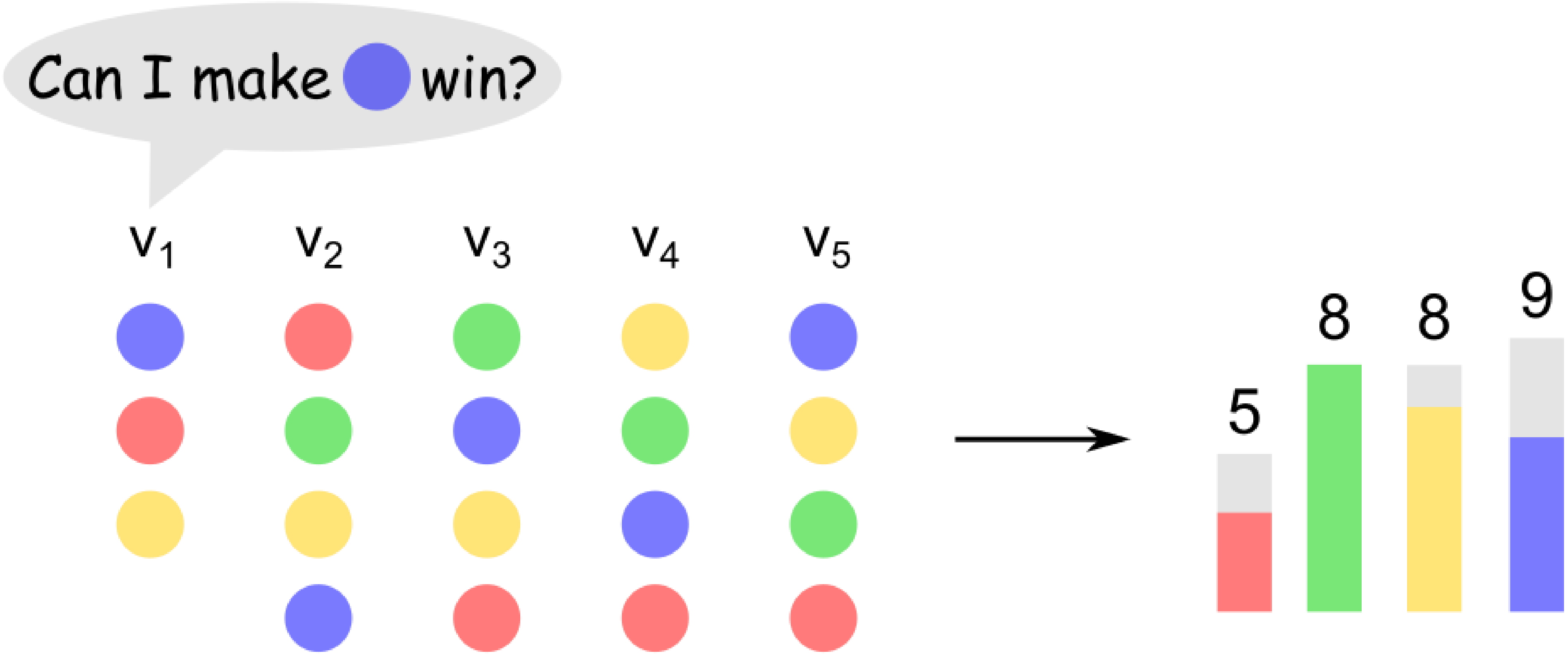
Manipulation under Borda Count



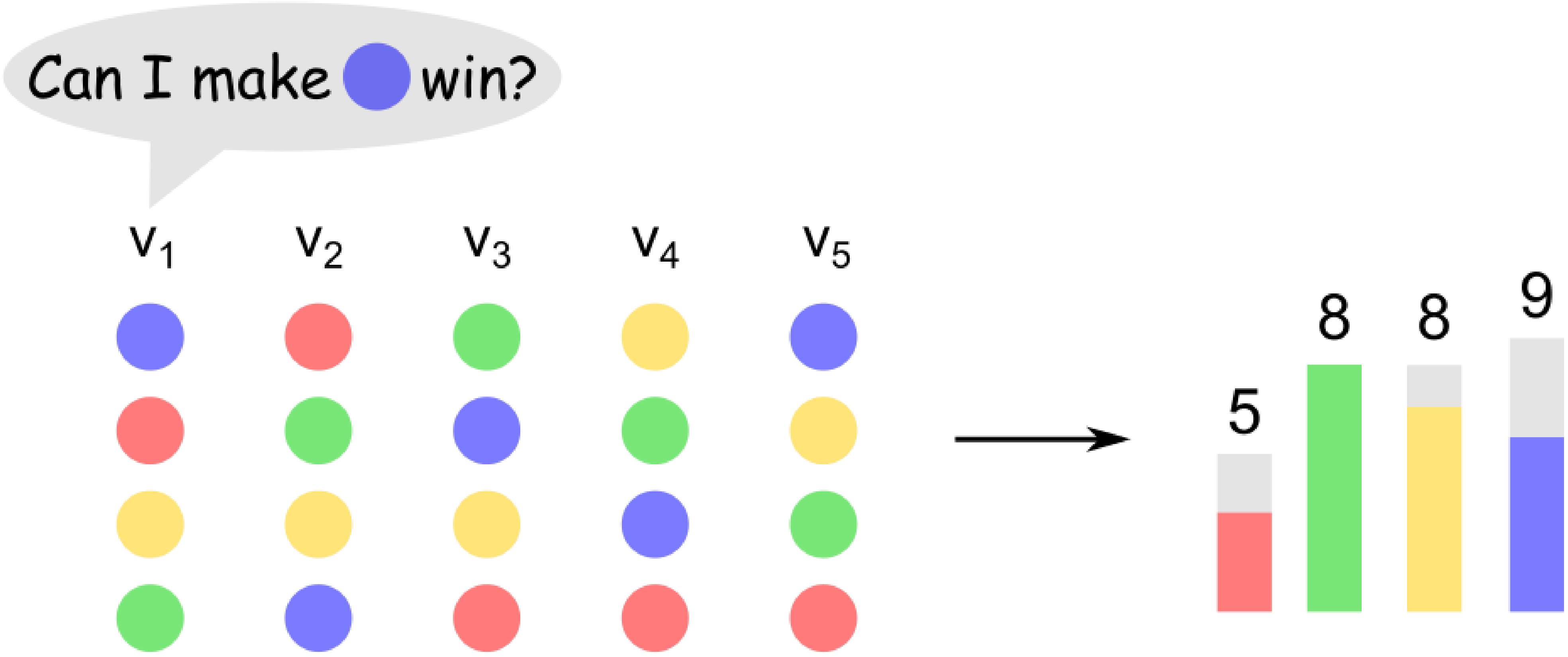
Manipulation under Borda Count



Manipulation under Borda Count



Manipulation under Borda Count



A Greedy Strategy

A Greedy Strategy

- Rank c at the top position in v_1's vote

A Greedy Strategy

- Rank c at the top position in v_1's vote
- While there is an unranked candidate:

A Greedy Strategy

- Rank c at the top position in v_1's vote

- While there is an unranked candidate:

  If a candidate, say x, can be "safely" placed in the next highest position in v_1's list without preventing c from winning, then place x in that position.

A Greedy Strategy

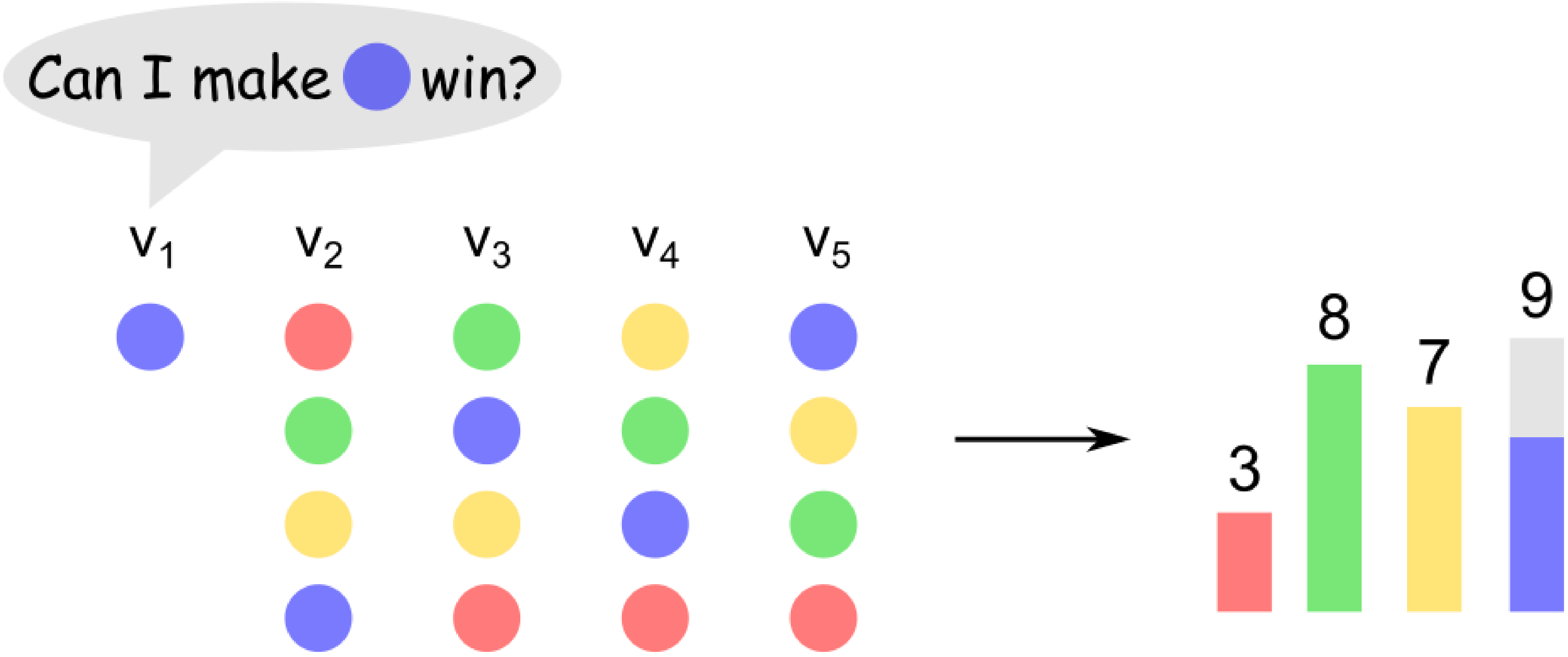
- Rank c at the top position in v_1's vote

- While there is an unranked candidate:

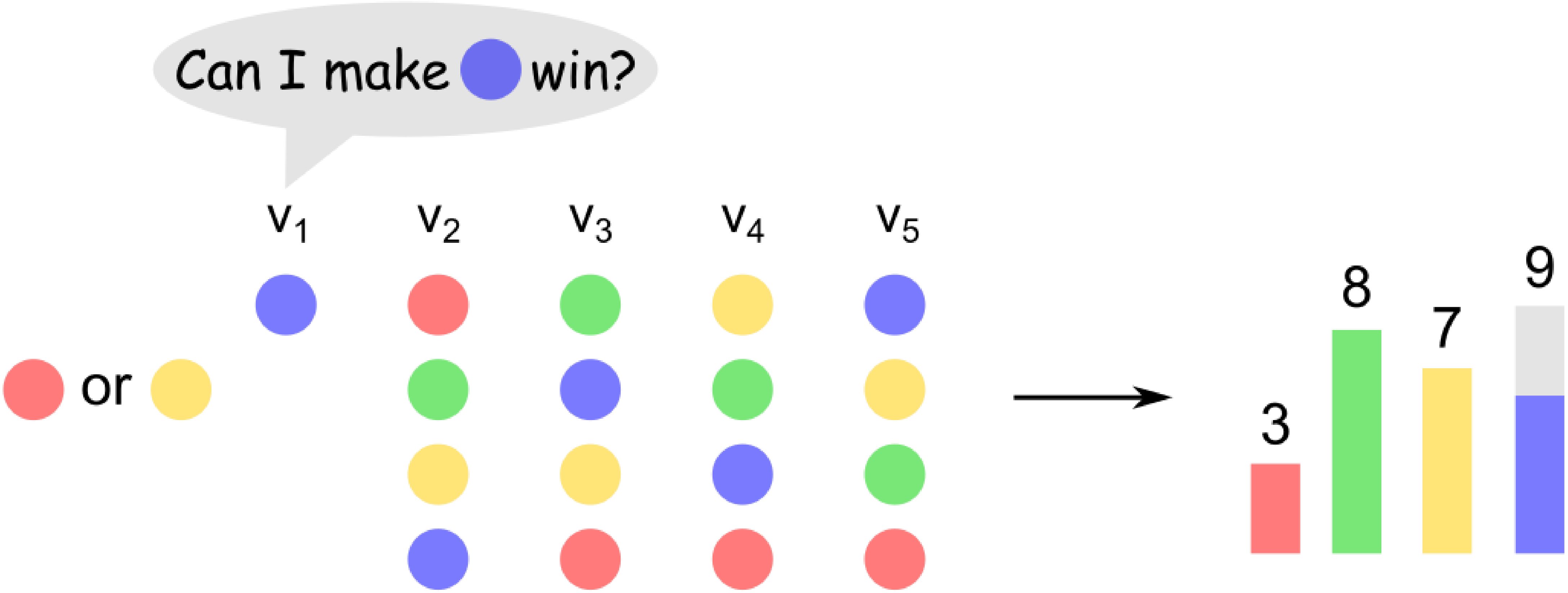
  If a candidate, say x, can be "safely" placed in the next highest position in v_1's list without preventing c from winning, then place x in that position.

- Otherwise, return 'No'.

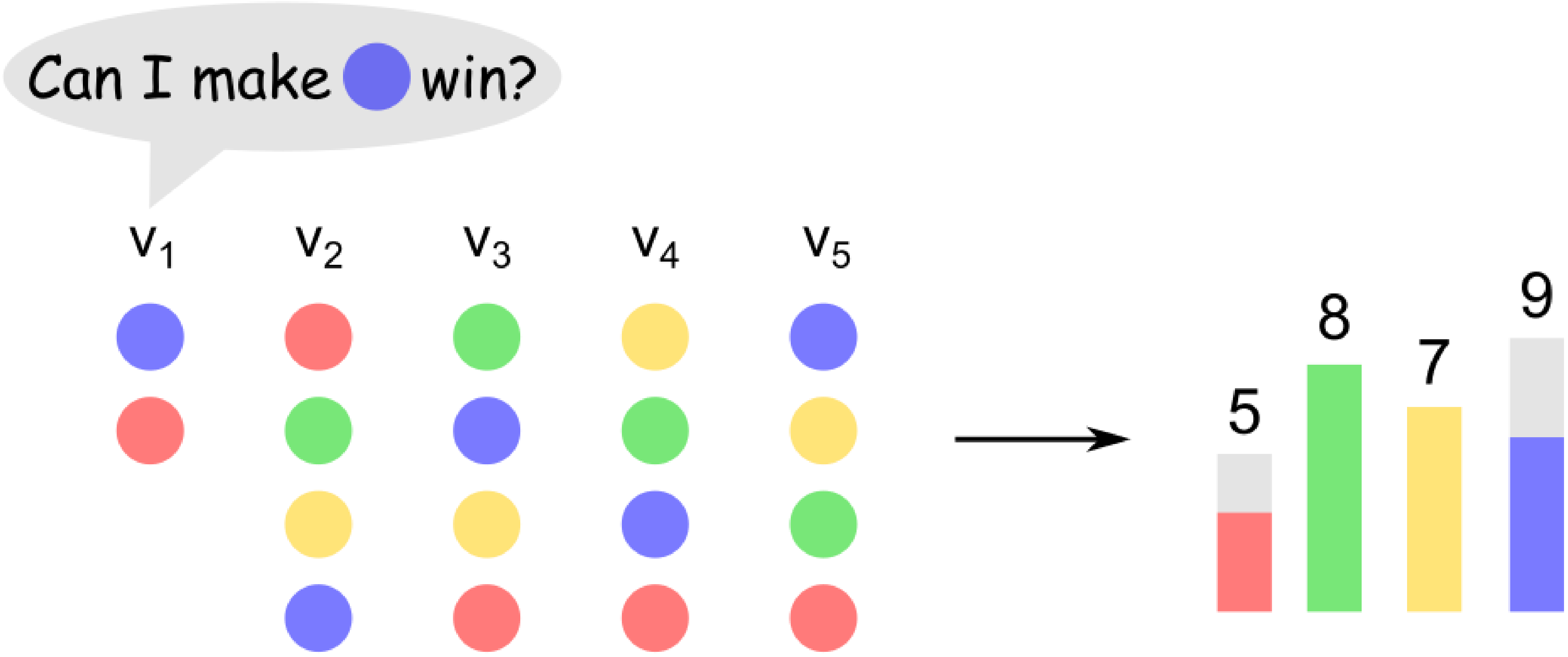
Manipulation under Borda Count



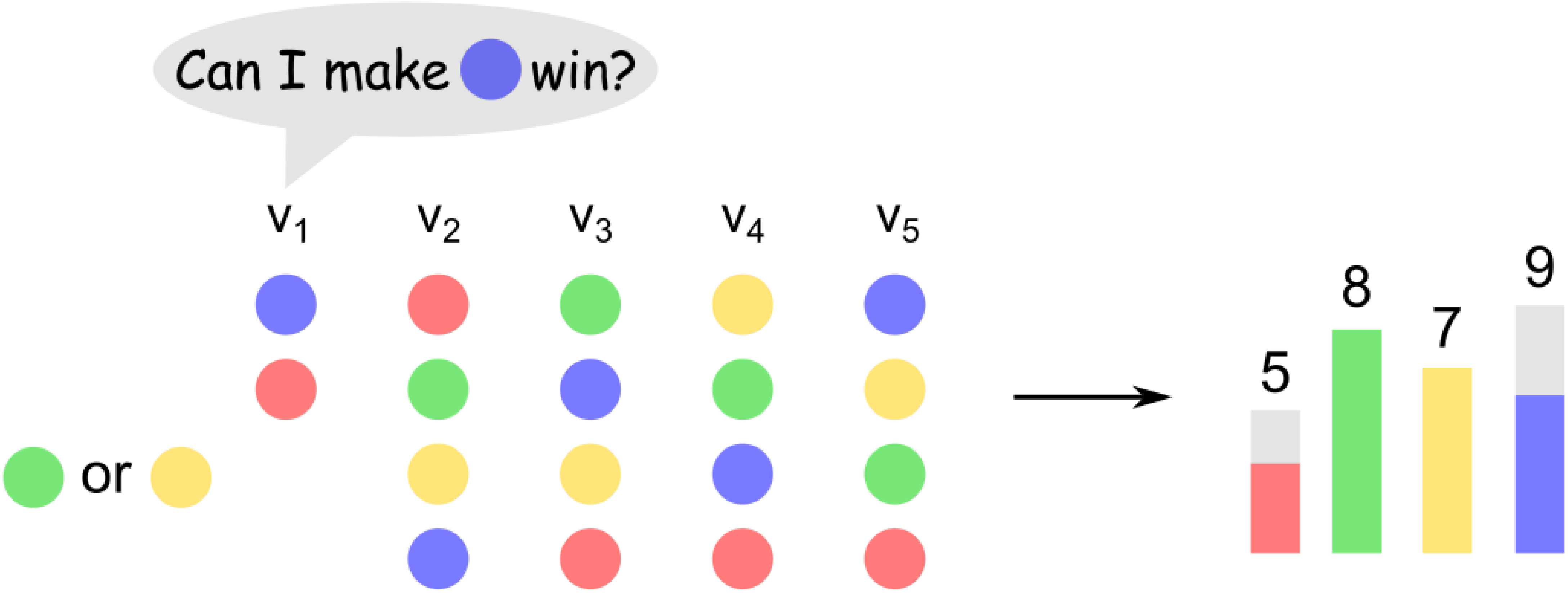
Manipulation under Borda Count



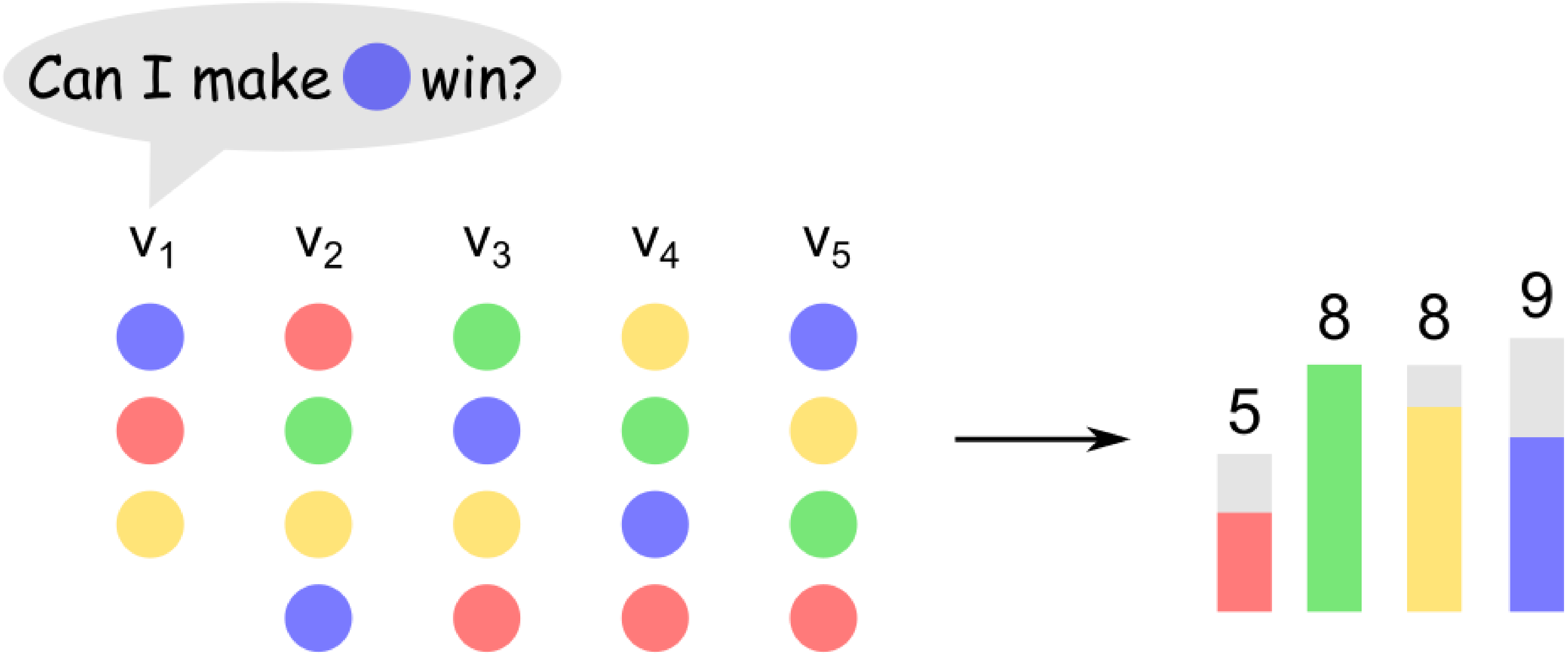
Manipulation under Borda Count



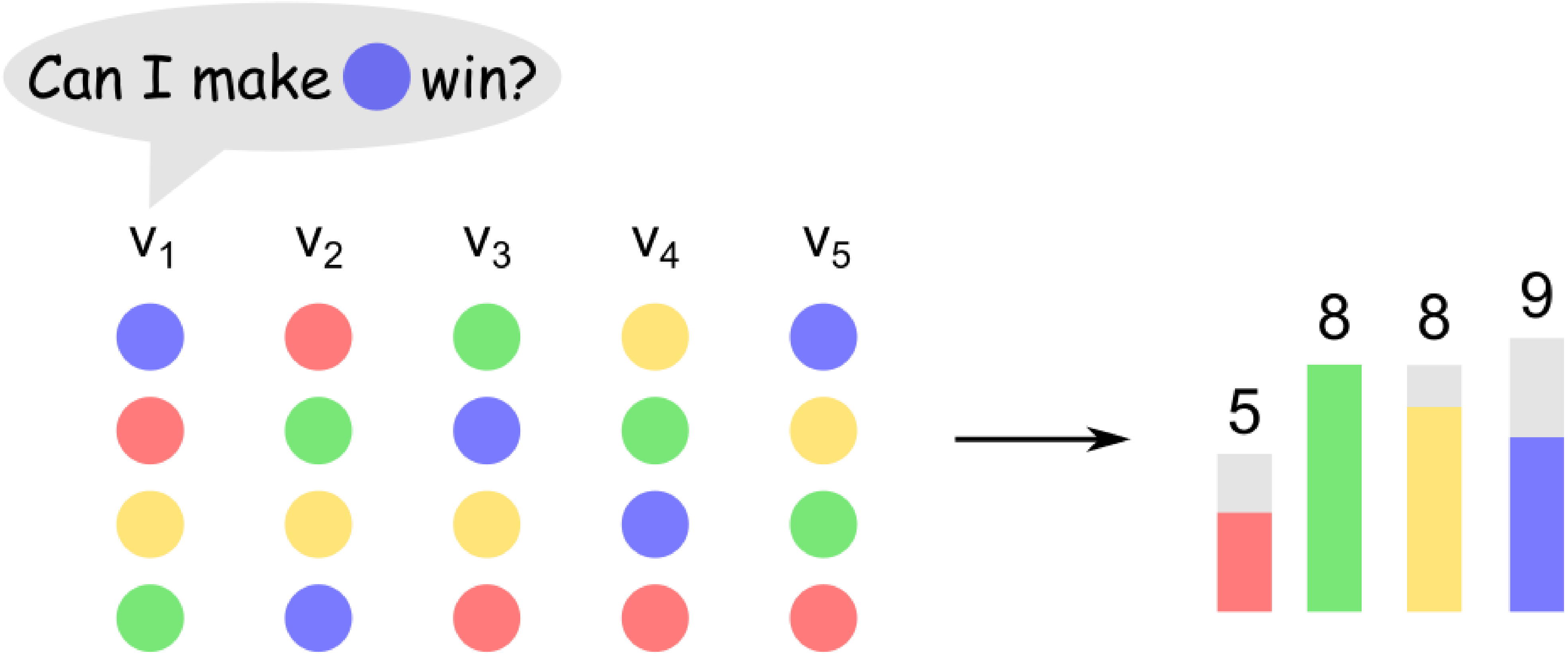
Manipulation under Borda Count



Manipulation under Borda Count

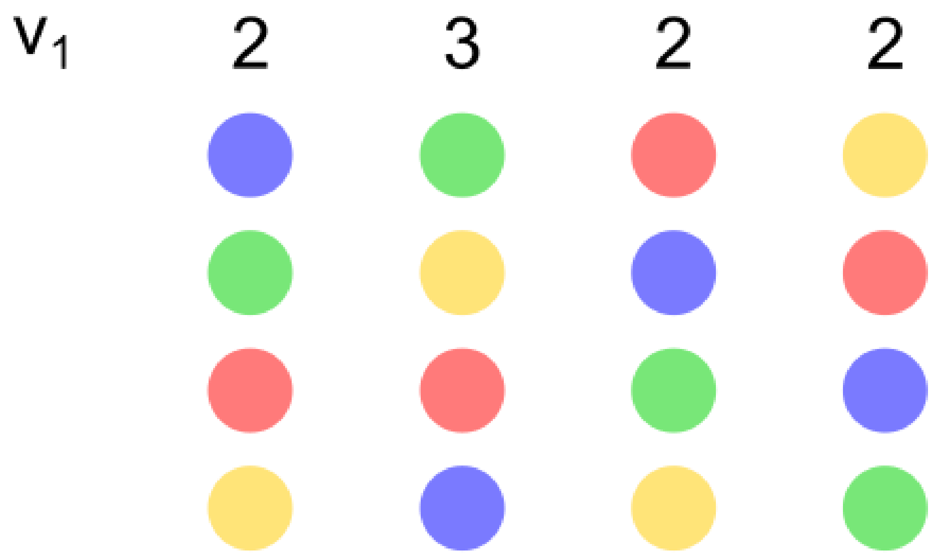


Manipulation under Borda Count

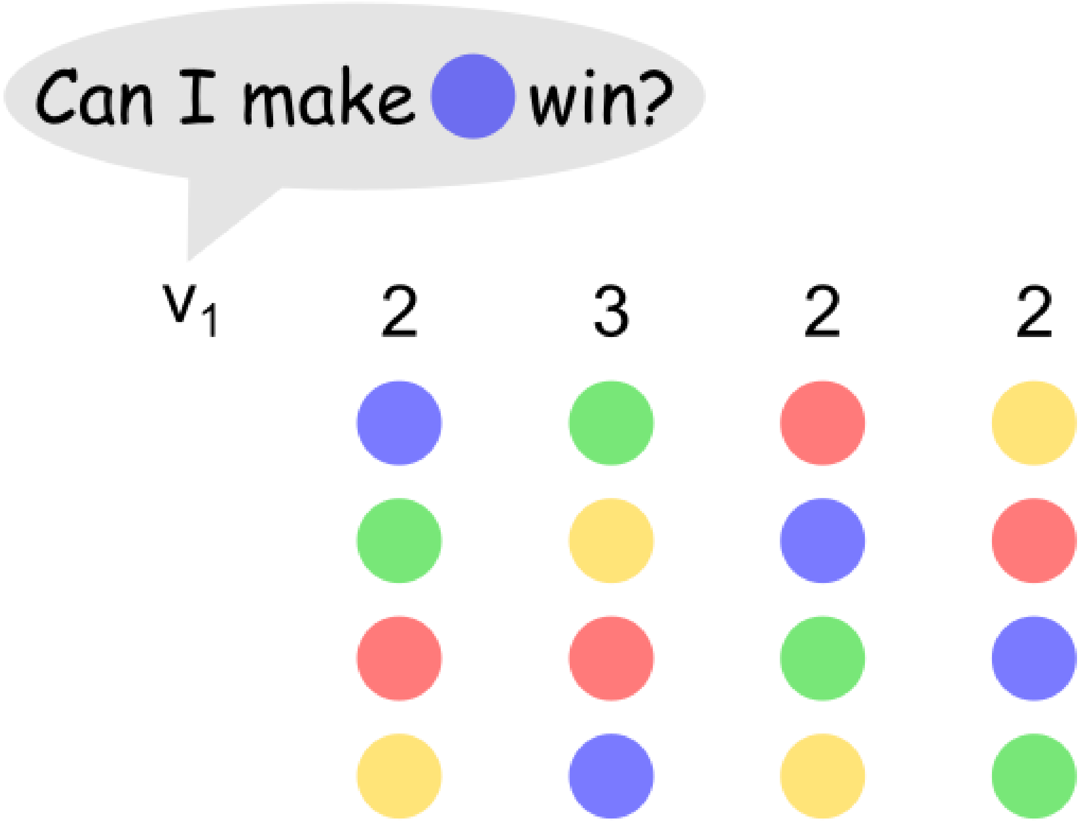


The greedy strategy does not always work.

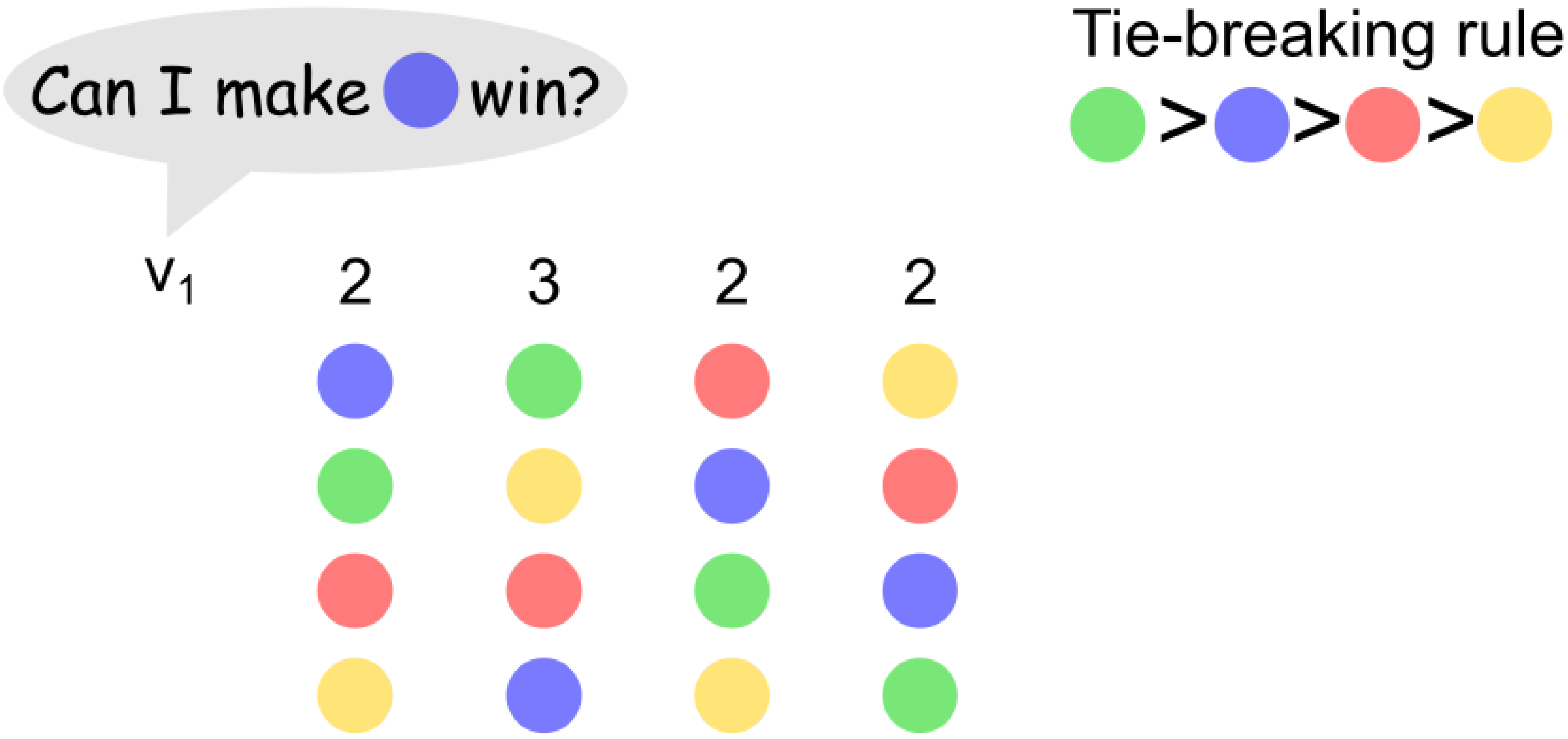
Manipulation under STV



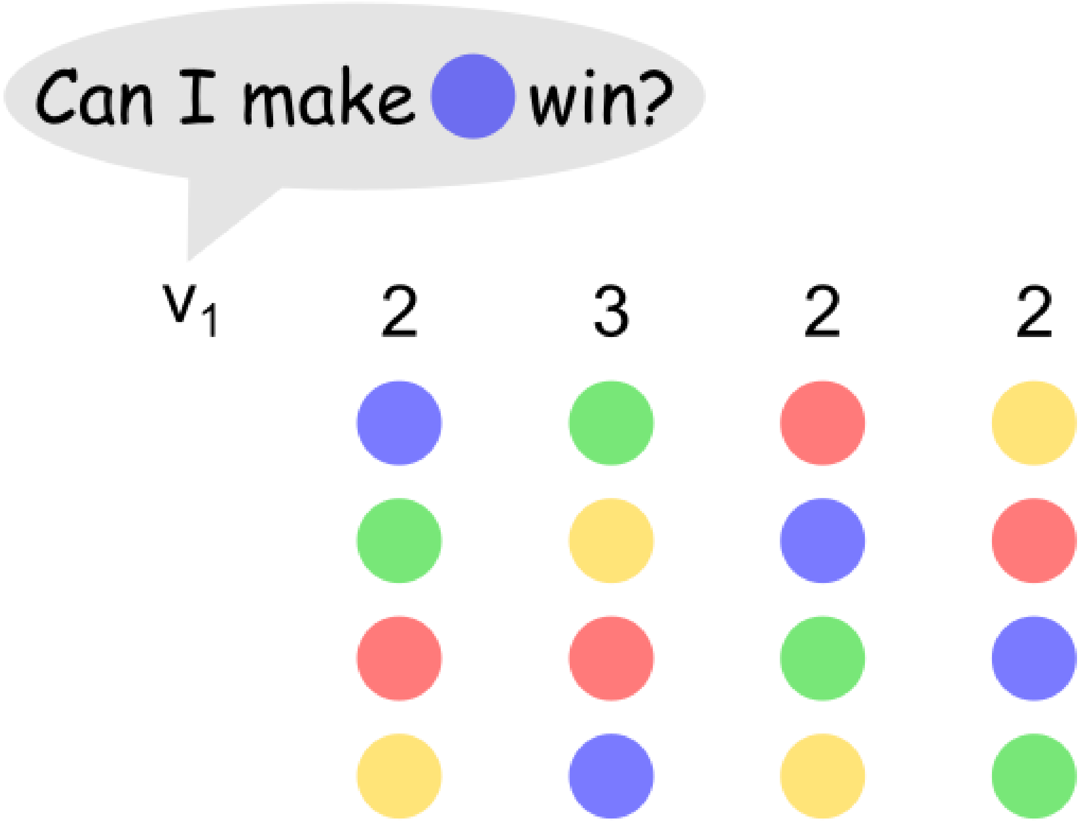
Manipulation under STV



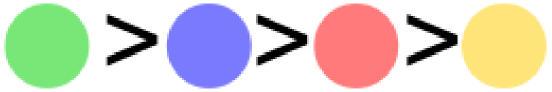
Manipulation under STV



Manipulation under STV

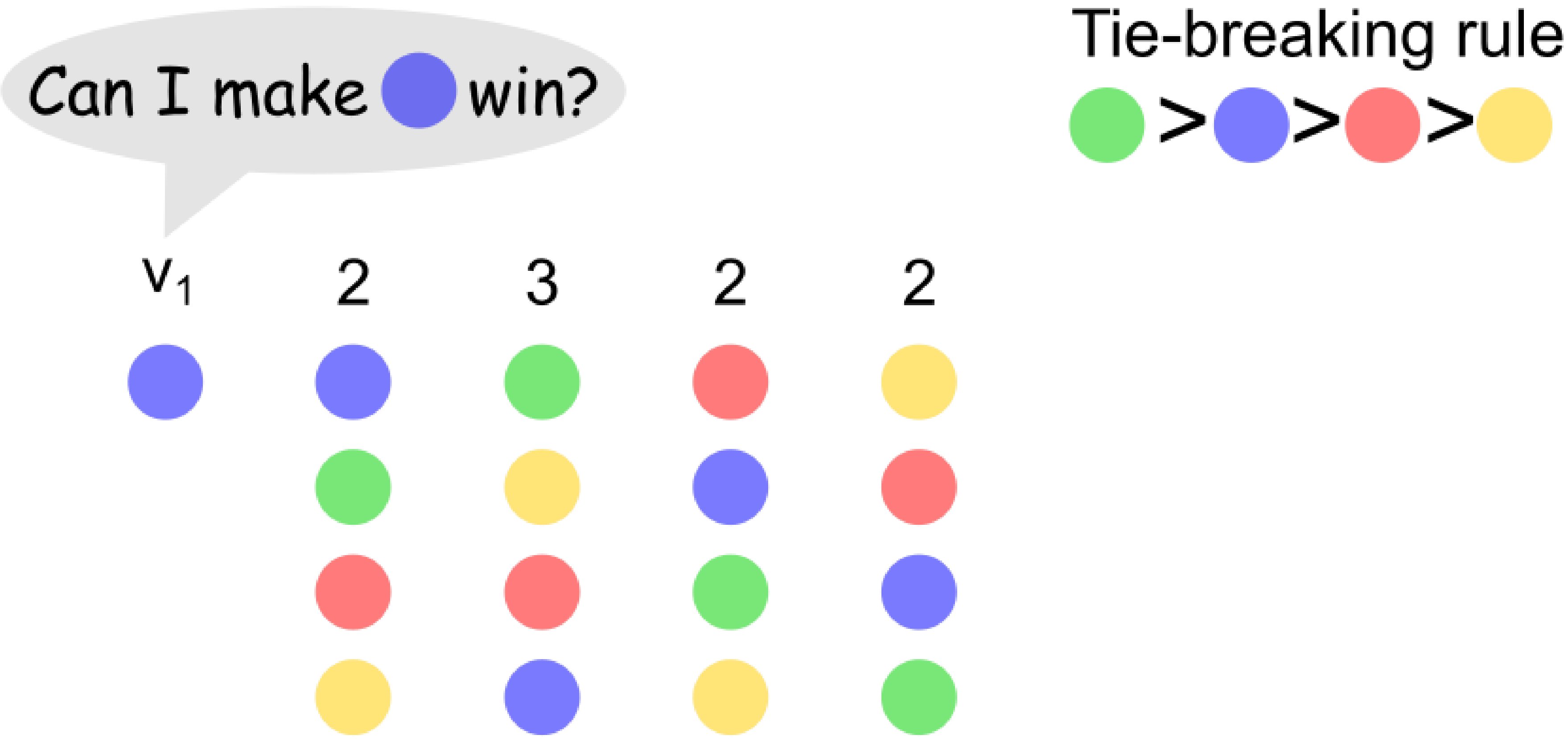


Tie-breaking rule



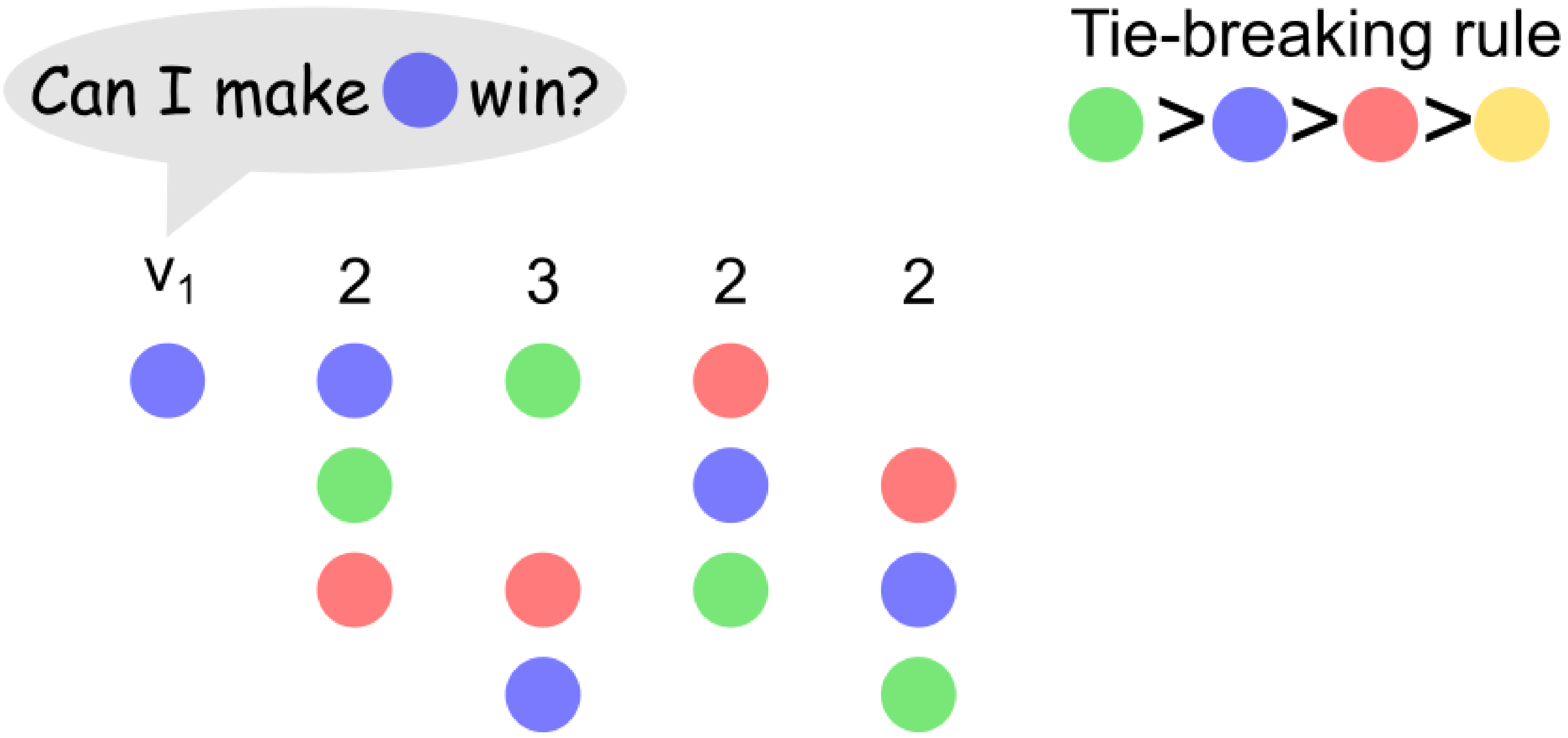
Let's follow the greedy strategy and put at the top.

Manipulation under STV



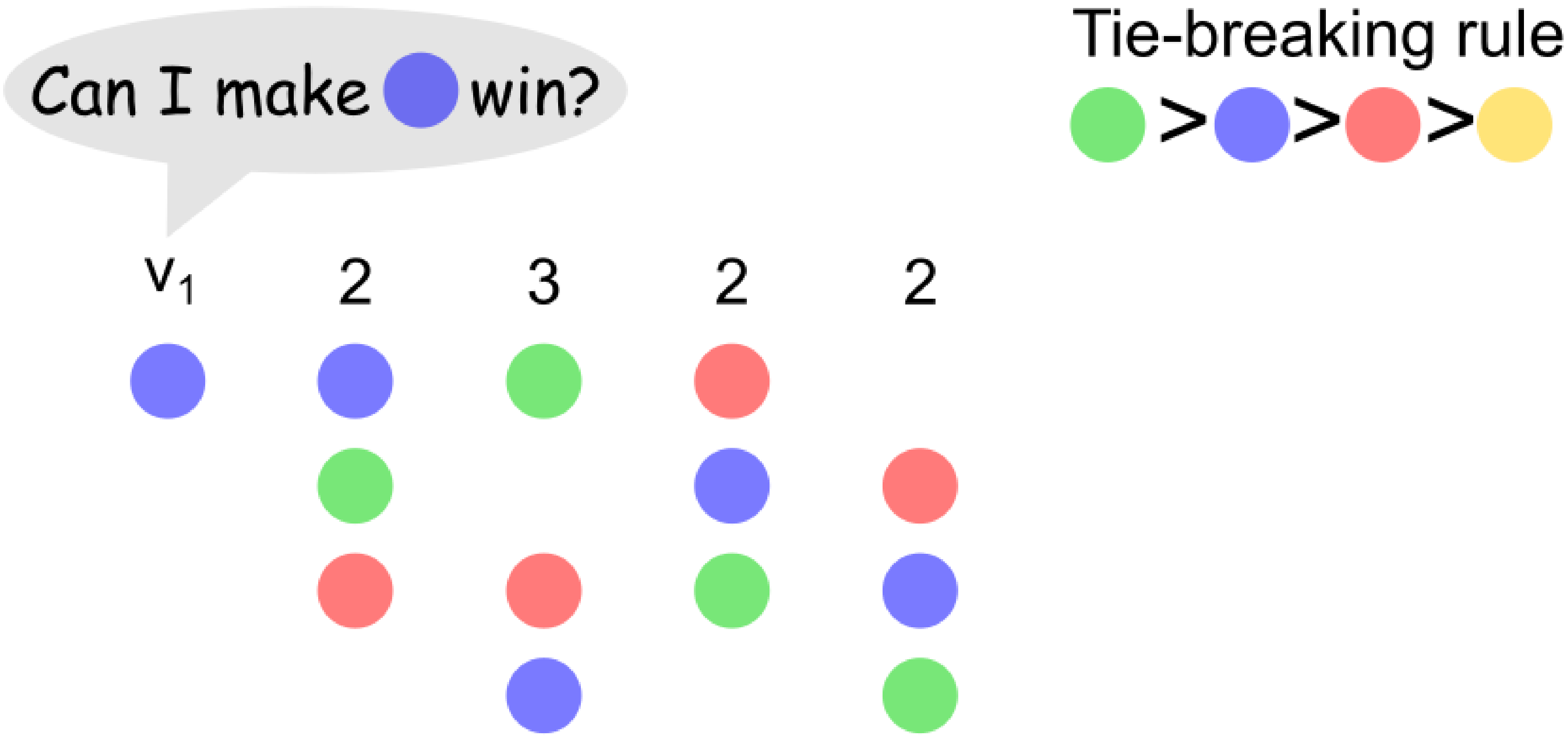
Let's follow the greedy strategy and put at the top.

Manipulation under STV



Let's follow the greedy strategy and put at the top.

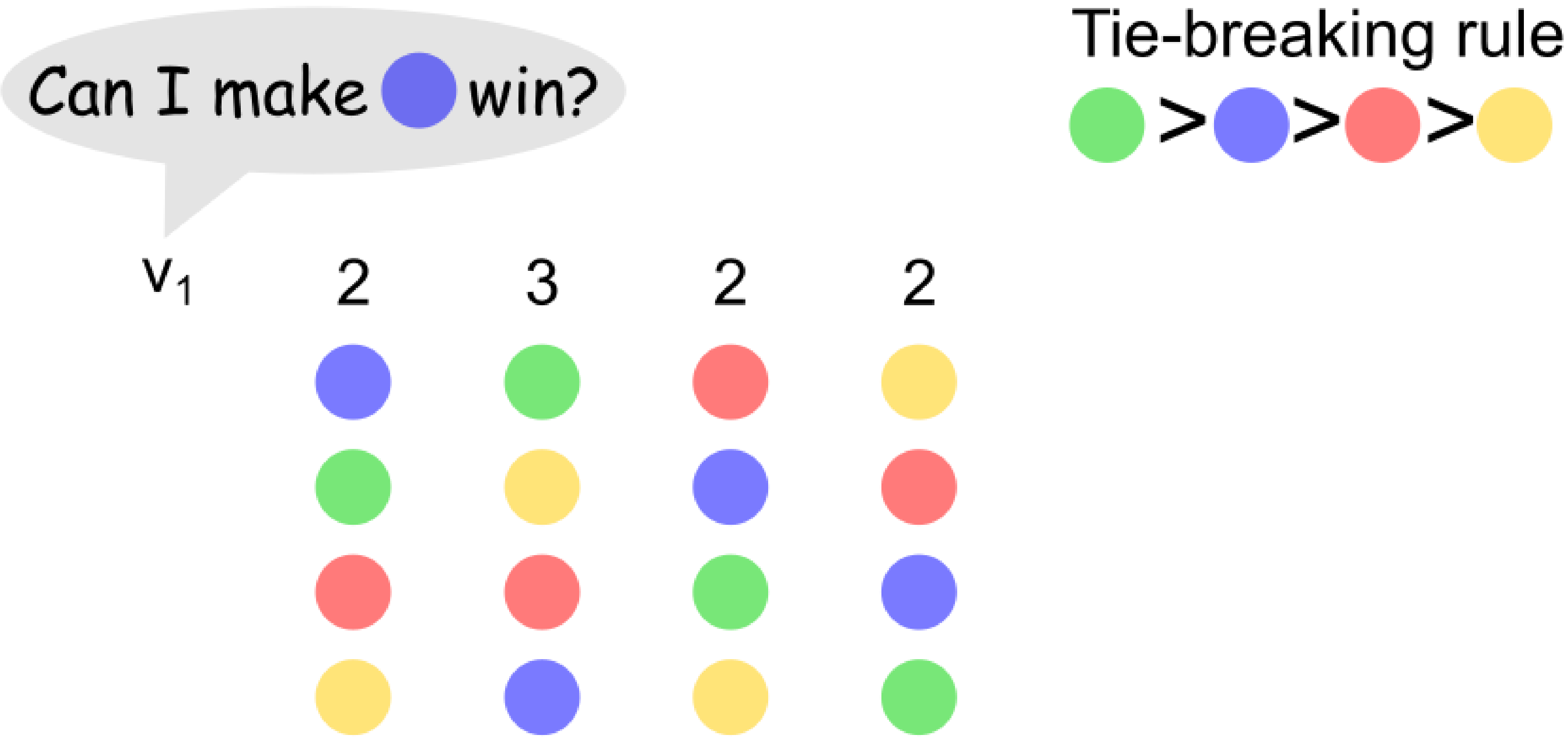
Manipulation under STV



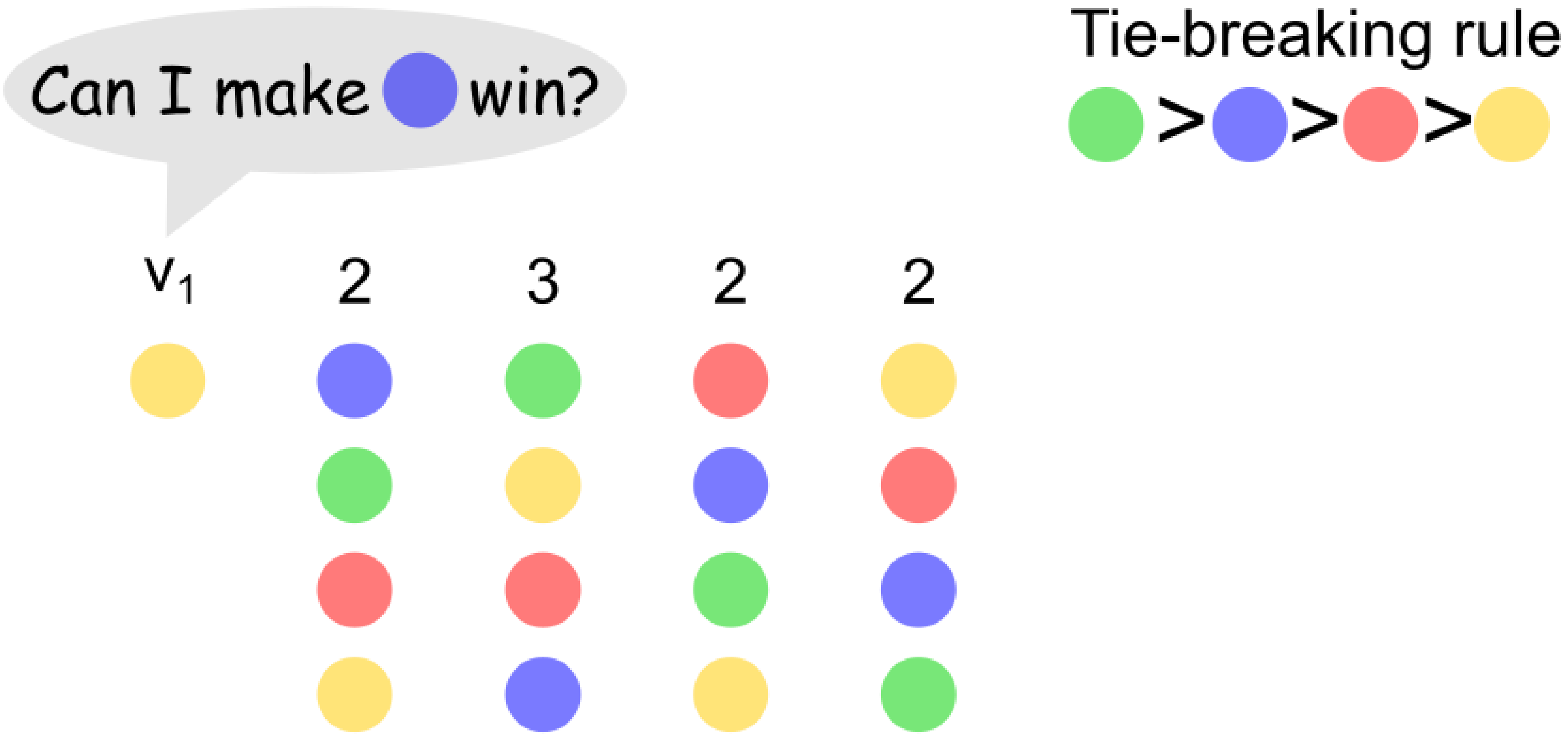
Let's follow the greedy strategy and put at the top.

is eliminated in the next round (due to tie-breaking rule).

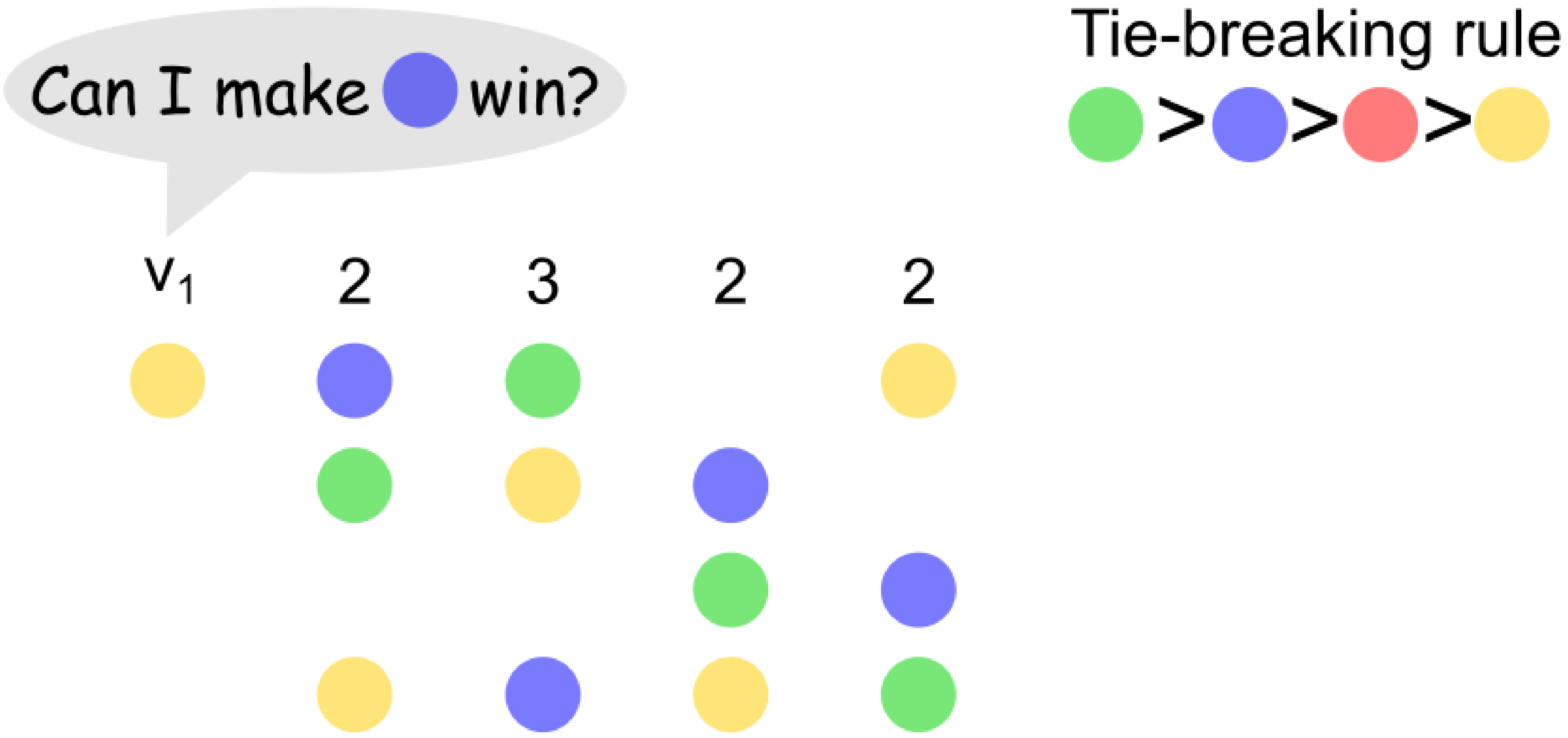
Manipulation under STV



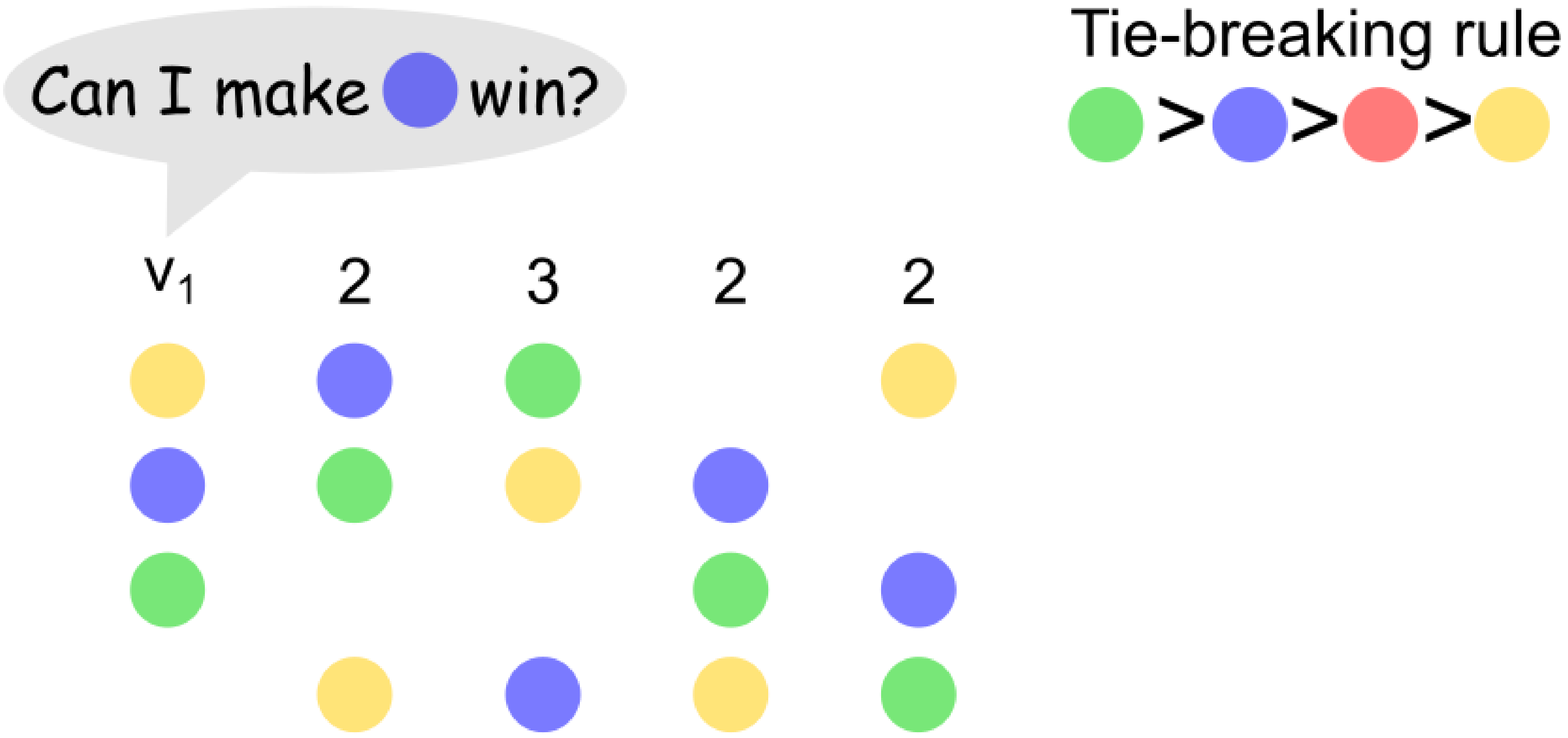
Manipulation under STV



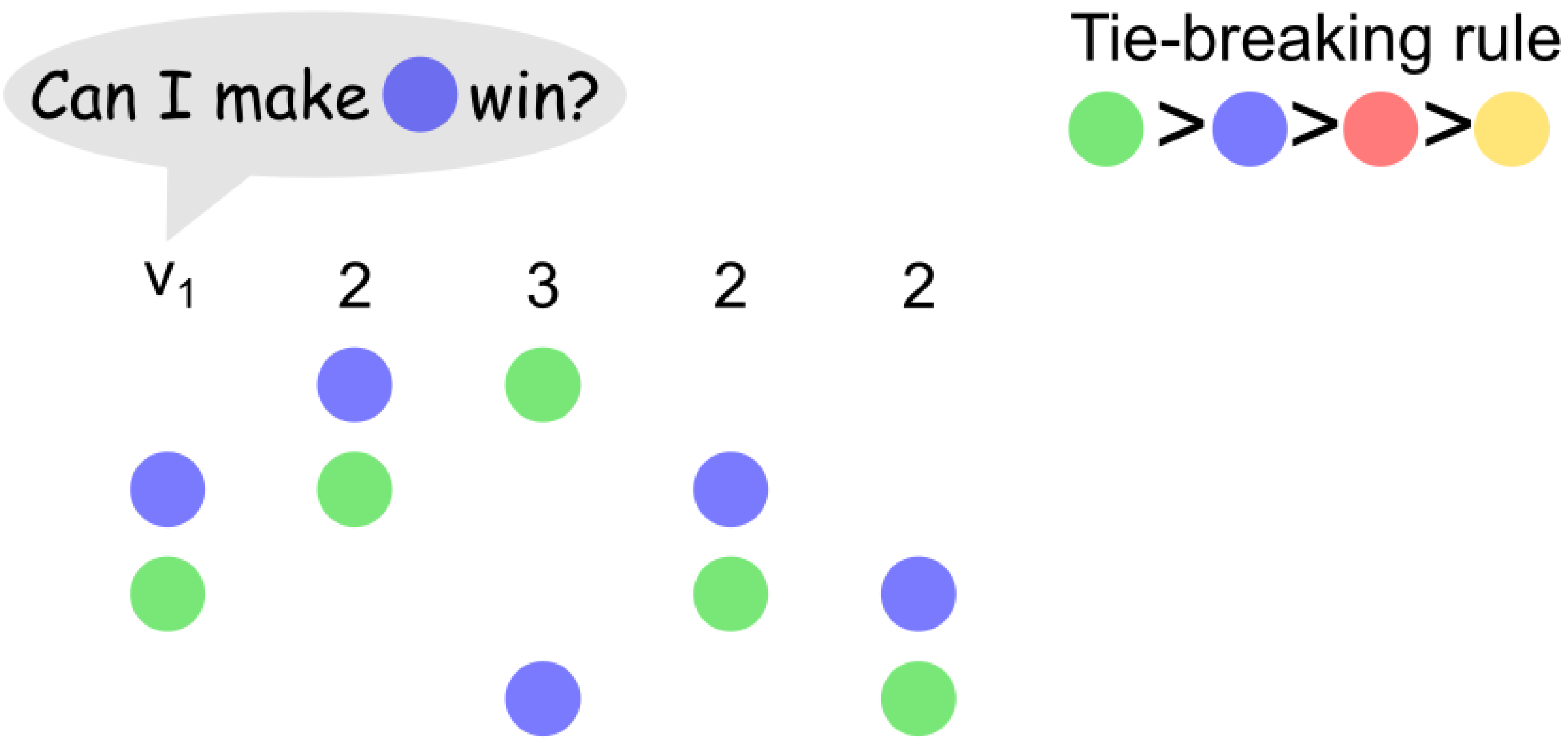
Manipulation under STV

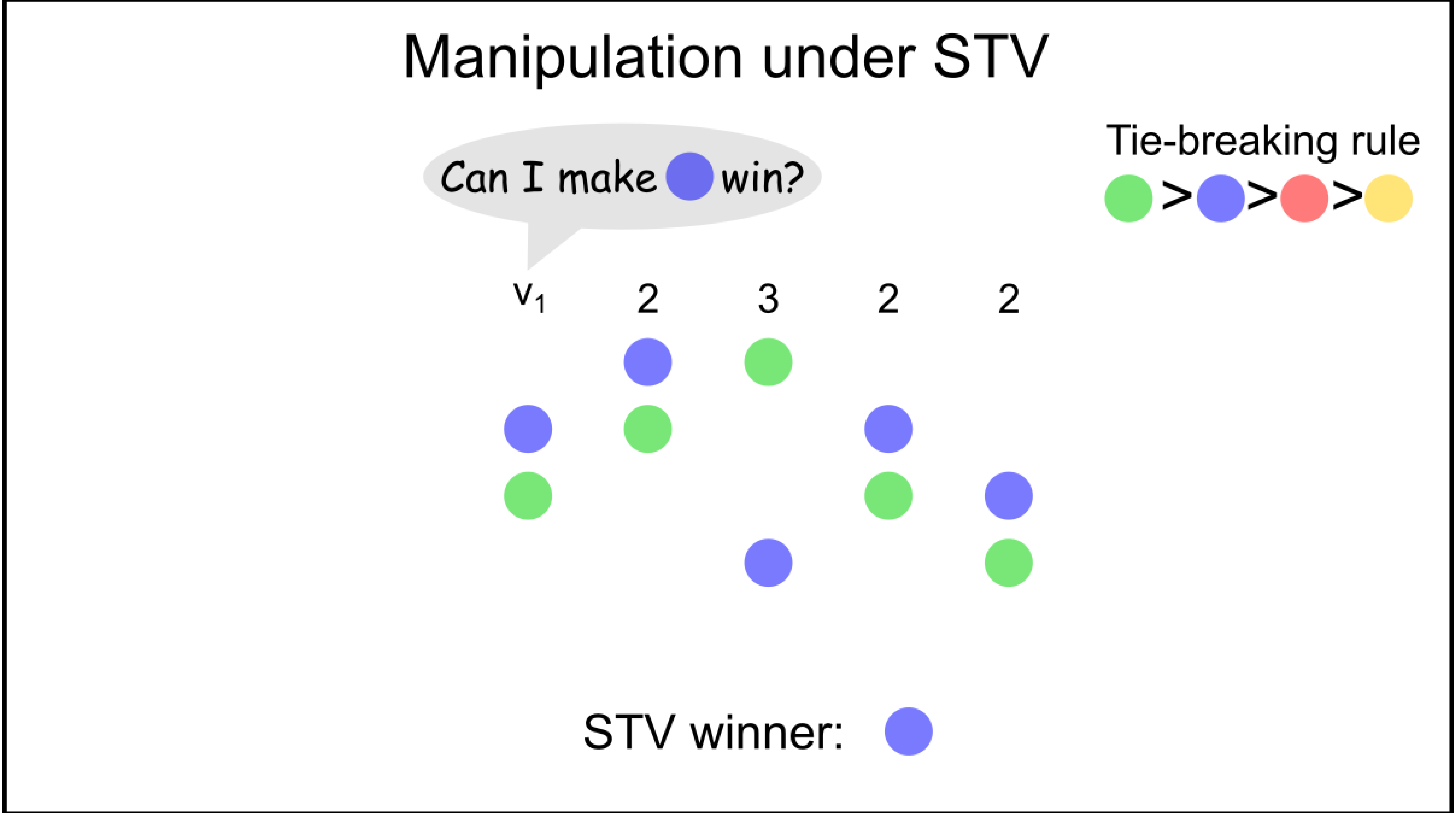


Manipulation under STV



Manipulation under STV




Manipulation under STV
Can I make win?
Tie-breaking rule
> > >
v1
2
3
2
2
STV winner:

So, *when* does the greedy strategy work?

[Bartholdi,Tovey and Trick, SCW 1989]

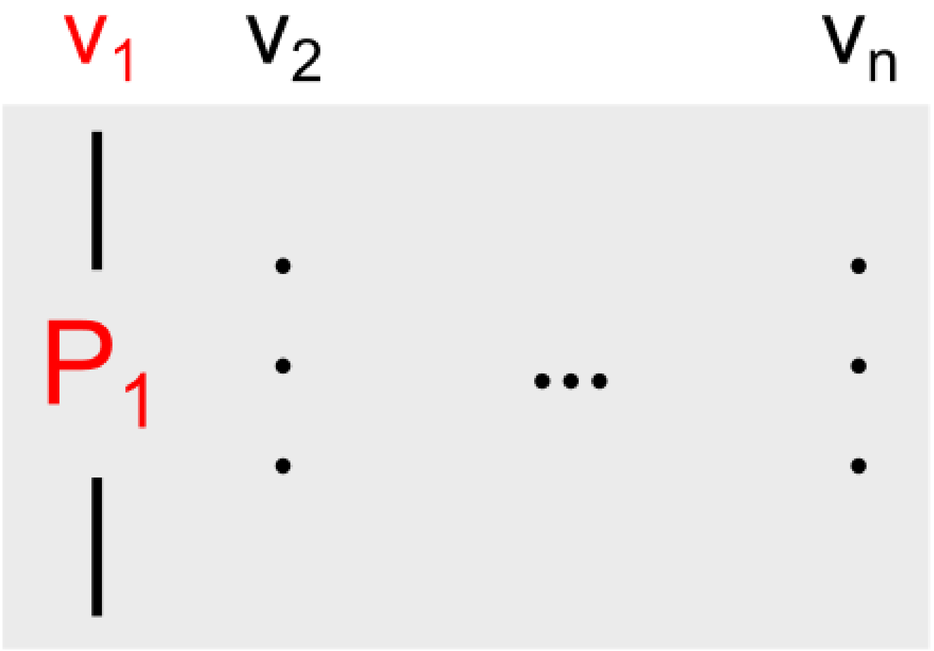
The greedy strategy can correctly solve f-Manipulation in polynomial time for any voting rule f satisfying:

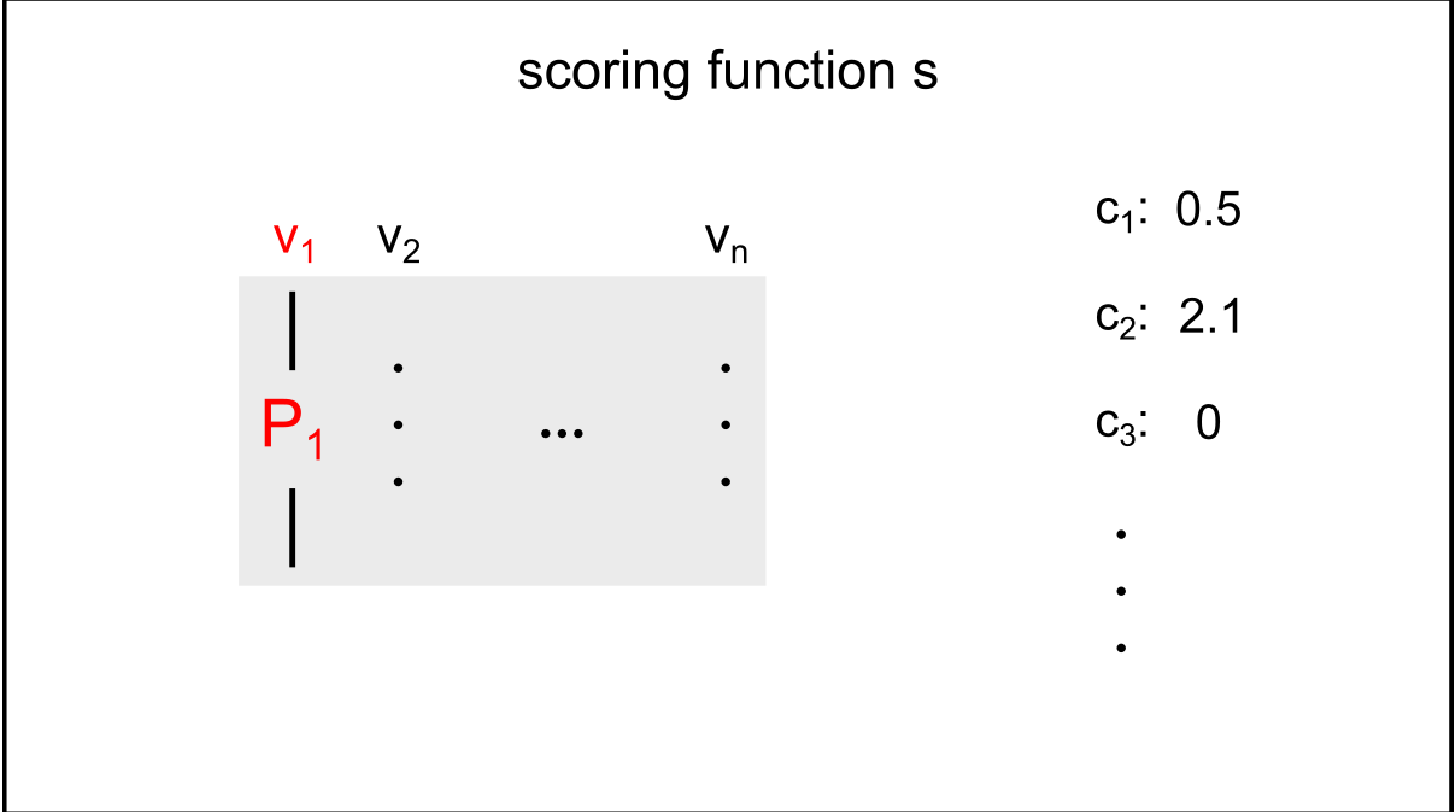
[Bartholdi,Tovey and Trick, SCW 1989]

The greedy strategy can correctly solve f-Manipulation in polynomial time for any voting rule f satisfying:

- Score-based: There exists a *scoring function* s: (P_1,x) → ℝ such that for any vote P_1 of v_1, the f-winner is the candidate maximizing s(P_1,x).

scoring function s




scoring function s
v1
v2
vn
P1
...
c1: 0.5
c2: 2.1
c3: 0

[Bartholdi,Tovey and Trick, SCW 1989]

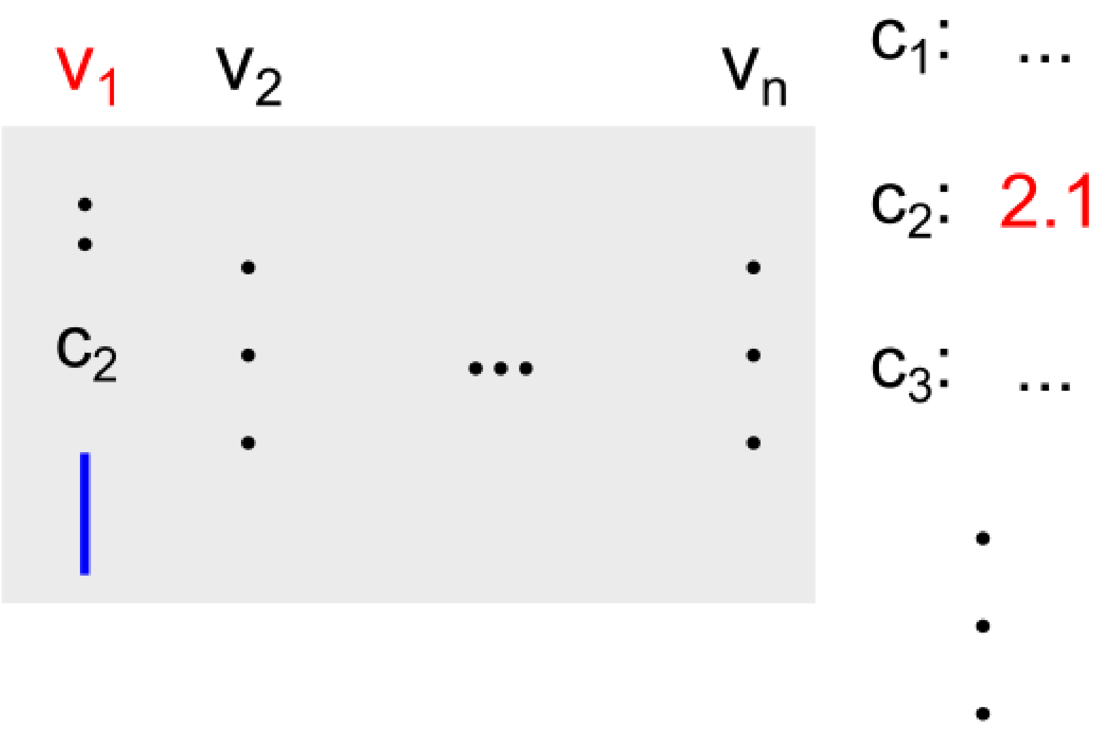
The greedy strategy can correctly solve f-Manipulation in polynomial time for any voting rule f satisfying:

- Score-based: There exists a *scoring function* s: (P_1,x) → ℝ such that for any vote P_1 of v_1, the f-winner is the candidate maximizing s(P_1,x).

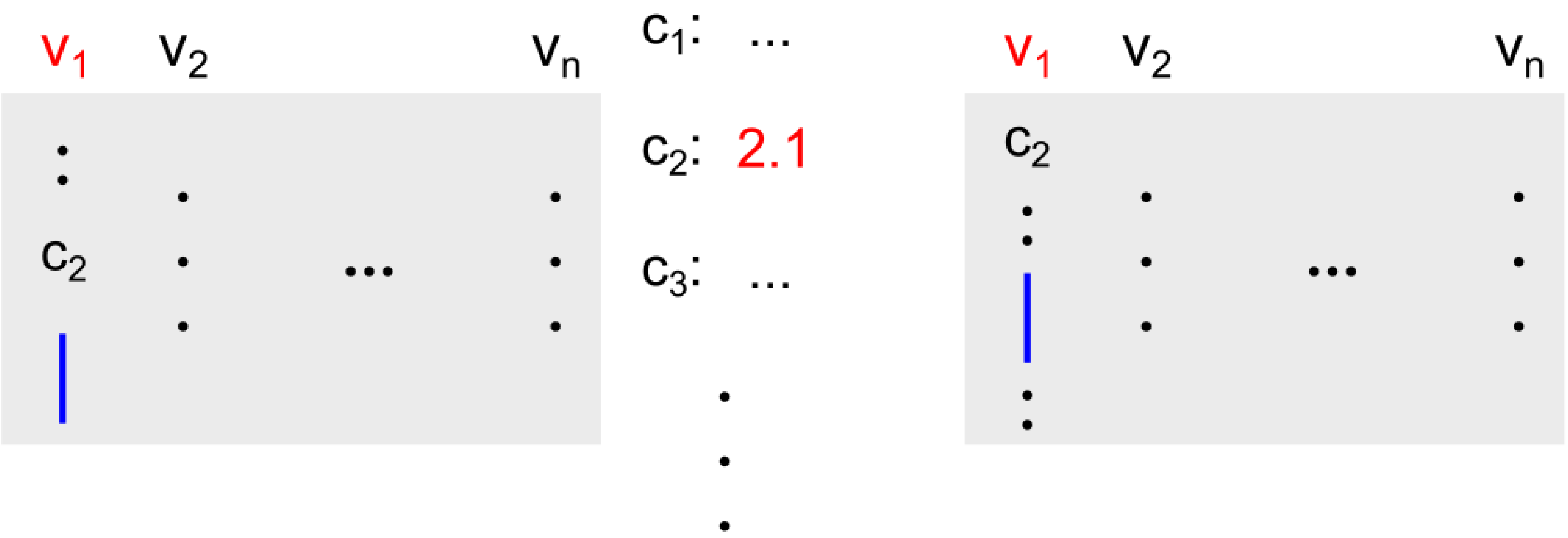
- Monotonicity: Suppose a candidate "x" is preferred over the set of candidates S under P and the set S' under P', and say S ⊆ S'. Then, s(P,x) ≤ s(P',x).

monotone scoring function s

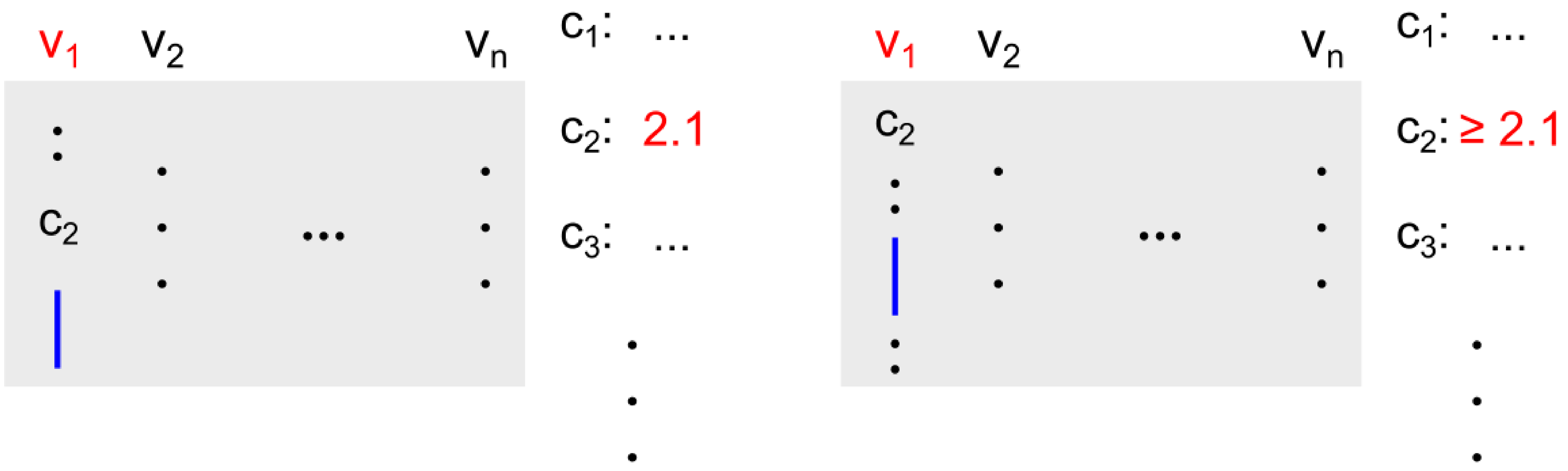
monotone scoring function s



monotone scoring function s



monotone scoring function s



[Bartholdi, Tovey and Trick, SCW 1989]

The greedy strategy can correctly solve f-Manipulation in polynomial time for any voting rule f satisfying:

- Score-based: There exists a *scoring function* s: (P_1,x) → ℝ such that for any vote P_1 of v_1, the f-winner is the candidate maximizing s(P_1,x).
- Monotonicity: Suppose a candidate "x" is preferred over the set of candidates S under P and the set S' under P', and say S ⊆ S'. Then, s(P,x) ≤ s(P',x).
- Efficiency: The voting rule f can be evaluated in polynomial time.

[Bartholdi,Tovey and Trick, SCW 1989]

The greedy strategy can correctly solve f-Manipulation in polynomial time for any voting rule f satisfying:

- **Score-based**: There exists a *scoring function* $s: (P_1,x) \rightarrow \mathbb{R}$ such that for any vote P_1 of v_1, the f-winner is the candidate maximizing $s(P_1,x)$.

- **Monotonicity**: Suppose a candidate "x" is preferred over the set of candidates S under P and the set S' under P', and say $S \subseteq S'$. Then, $s(P,x) \leq s(P',x)$.

- **Efficiency**: The voting rule f can be evaluated in polynomial time.

In particular, for $f \in$ {Plurality, Borda, Copeland}.

Is manipulation *always* easy?

Soc Choice Welfare (1989) 6: 227–241

Social Choice and Welfare

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.

For many voting rules, f-Manipulation is NP-hard.

For many voting rules, f-Manipulation is NP-hard.

[Bartholdi, Tovey, and Trick, SCW 1989]

Copeland with second-order tie-breaking

In case of a tie, winner is the candidate whose defeated competitors have the highest sum of Copeland scores.

For many voting rules, f-Manipulation is NP-hard.

[Bartholdi, Tovey, and Trick, SCW 1989]

Copeland with second-order tie-breaking

In case of a tie, winner is the candidate whose defeated competitors have the highest sum of Copeland scores.

[Bartholdi and Orlin, SCW 1991]

Single Transferable Vote (STV)

For many voting rules, f-Manipulation is NP-hard.

[Bartholdi, Tovey, and Trick, SCW 1989]

Copeland with second-order tie-breaking

In case of a tie, winner is the candidate whose defeated competitors have the highest sum of Copeland scores.

[Bartholdi and Orlin, SCW 1991]

Single Transferable Vote (STV)

[Xia, Zuckerman, Procaccia, Conitzer, Rosenschein, IJCAI 2009]

Ranked Pairs

Consider candidate pairs according to the margin of head-to-head victories, and create a ranking based on it while avoiding cycles.

For many voting rules, f-Manipulation is NP-hard.

NP-hardness is good news!

No general-purpose efficient algorithm that correctly works on all preference profiles (unless P=NP).

For many voting rules, f-Manipulation is NP-hard.

NP-hardness is good news!
No general-purpose efficient algorithm that correctly works on all preference profiles (unless P=NP).

Using worst-case computational hardness as a barrier to manipulation.

For many voting rules, f-Manipulation is NP-hard.

NP-hardness is good news!
No general-purpose efficient algorithm that correctly works on all preference profiles (unless P=NP).

Using worst-case computational hardness as a barrier to manipulation.

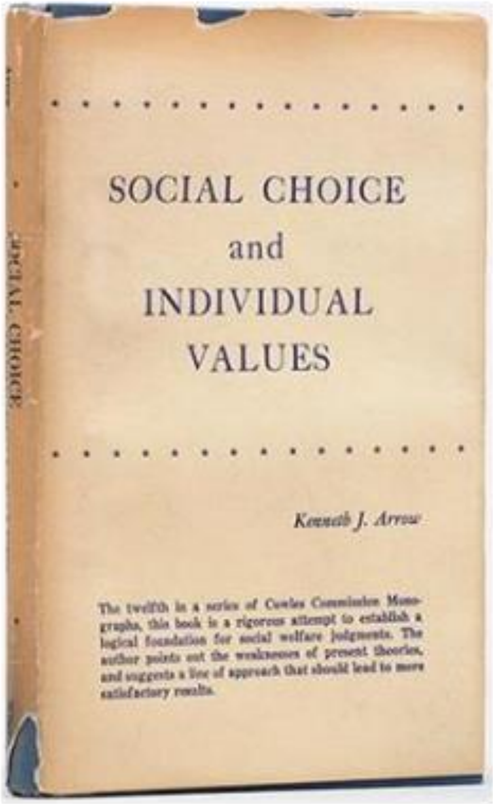
Note: NP-hard *even with* full information.

Remember this?

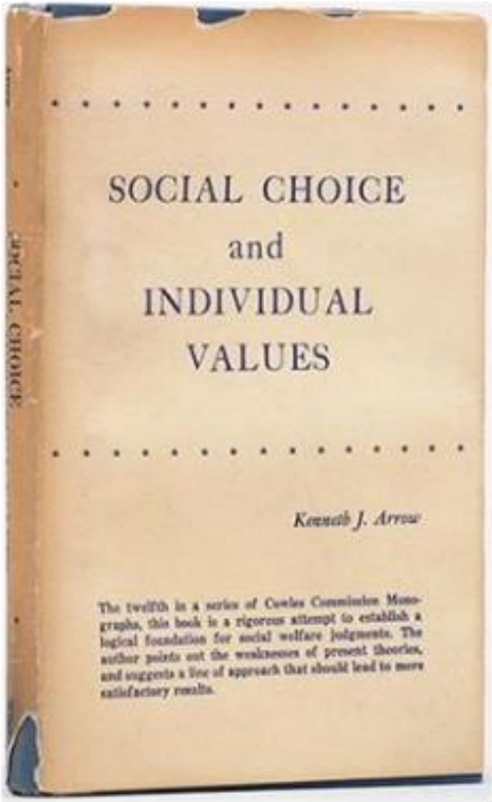
Sort: Criterion / Method	Majority	Maj. loser	Mutual maj.	Condorcet	Cond. loser	Smith/ ISDA	LIIA	IIA	Cloneproof	Monotone	Consistency	Participation	Reversal symmetry	Polytime/ resolvable		Summable	Later-no-		No favorite betrayal	Ballot type	Ranks	
																	Harm	Help			=	>2
Approval	Rated[a]	No	No	No[b][c]	No	No[b]	Yes	Yes[d]	Yes[e]	Yes	Yes	Yes	Yes	O(N)	Yes	O(N)	No	Yes[f]	Yes	Approvals	Yes	No
Borda count	No	Yes	No	No[b]	Yes	No	No	No	Teams	Yes	Yes	Yes	Yes	O(N)	Yes	O(N)	No	Yes	No	Ranking	Yes	Yes
Bucklin	Yes	Yes	Yes	No	No	No	No	No	No	Yes	No	No	No	O(N)	Yes	O(N)	No	Yes	If equal preferences	Ranking	Yes	Yes
Copeland	Yes	Yes	Yes	Yes	Yes	Yes	No	No[b]	Teams, crowds	Yes	No[b]	No[b]	Yes	$O(N^2)$	No	$O(N^2)$	No[b]	No	No[b]	Ranking	Yes	Yes
IRV (AV)	Yes	Yes	Yes	No[b]	Yes	No[b]	No	No	Yes	No	No	No	No	$O(N^2)$	Yes[g]	O(N!)[h]	Yes	Yes	No	Ranking	No	Yes
Kemeny–Young	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No[b]	Spoilers	Yes	No[b][i]	No[b]	Yes	O(N!)	Yes	$O(N^2)$[j]	No[b]	No	No[b]	Ranking	Yes	Yes
Highest median/Majority judgment[k]	Rated[l]	Yes[m]	No[n]	No[b][c]	No	No[b]	Yes	Yes[d]	Yes	Yes	No[o]	No[p]	Depends[q]	O(N)	Yes	O(N)[r]	No[s]	Yes	Yes	Scores[t]	Yes	Yes
Minimax	Yes	No	No	Yes[u]	No	No	No	No[b]	Spoilers	Yes	No[b]	No[b]	No	$O(N^2)$	Yes	$O(N^2)$	No[b][u]	No	No[b]	Ranking	Yes	Yes
Plurality/FPTP	Yes	No	No	No[b]	No	No[b]	No	No	Spoilers	Yes	Yes	Yes	No	O(N)	Yes	O(N)	N/A[v]	N/A[v]	No	Single mark	N/A	No
Score voting	No	No	No	No[b][c]	No	No[b]	Yes	Yes[d]	Yes	Yes	Yes	Yes	Yes	O(N)	Yes	O(N)	No	Yes	Yes	Scores	Yes	Yes
Ranked pairs	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No[b]	Yes	Yes	No[b]	No[p][b]	Yes	$O(N^3)$	Yes	$O(N^2)$	No[b]	No	No[p][b]	Ranking	Yes	Yes
Runoff voting	Yes	Yes	No	No[b]	Yes	No[b]	No	No	Spoilers	No	No	No	No	O(N)[w]	Yes	O(N)[w]	Yes	Yes[x]	No	Single mark	N/A	No[y]
Schulze	Yes	Yes	Yes	Yes	Yes	Yes	No	No[b]	Yes	Yes	No[b]	No[p][b]	Yes	$O(N^3)$	Yes	$O(N^2)$	No[b]	No	No[p][b]	Ranking	Yes	Yes
STAR voting	No[z]	Yes	No[aa]	No[b][c]	Yes	No[b]	No	No	No	Yes	No	No	Depends[ab]	O(N)	Yes	$O(N^2)$	No	No	No[ac]	Scores	Yes	Yes
Sortition, arbitrary winner[ad]	No	No	No	No[b]	No	No[b]	Yes	Yes	No	Yes	Yes	Yes	Yes	O(1)	No	O(1)	Yes	Yes	Yes	None	N/A	N/A
Random ballot[ae]	No	No	No	No[b]	No	No[b]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	O(N)	No	O(N)	Yes	Yes	Yes	Single mark	N/A	No

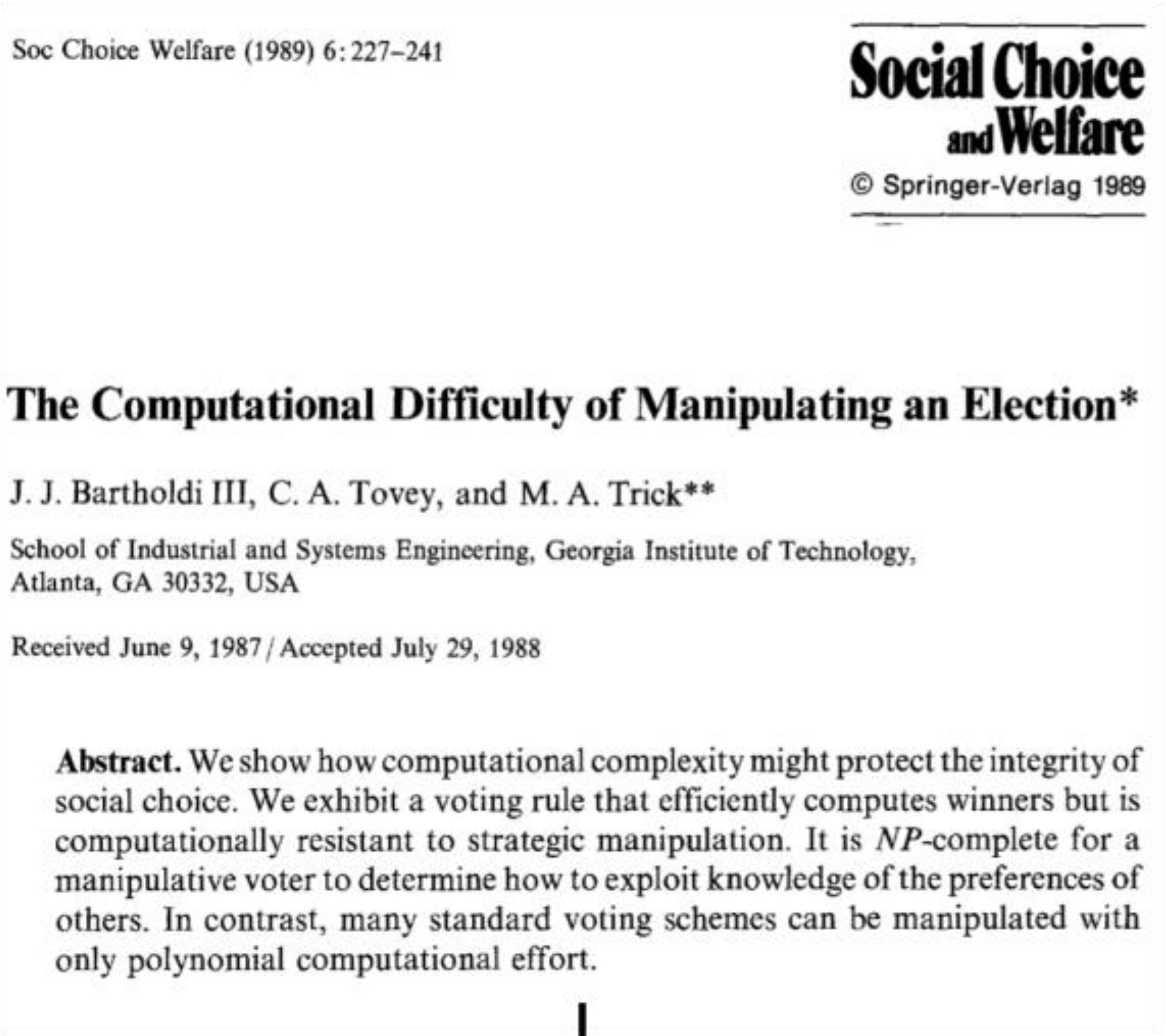
	Single manipulator
Plurality	P [Bartholdi, Tovey and Trick, SCW 1989]
Borda	P [Bartholdi, Tovey and Trick, SCW 1989]
Copeland$^{\alpha}$ (friendly tie-breaking)	P [Bartholdi, Tovey and Trick, SCW 1989]
Ranked pairs	NP-hard [Xia, Zuckerman, Procaccia, Conitzer, and Rosenschein, IJCAI 2009]
Schulze	P [Parkes and Xia, AAAI 2012]

	Single manipulator	Two manipulators
Plurality	P [Bartholdi, Tovey and Trick, SCW 1989]	P
Borda	P [Bartholdi, Tovey and Trick, SCW 1989]	NP-hard [Betzler, Niedermeier and Woeginger, IJCAI 2011; Davies, Katsirelos, Narodytska and Walsh, AAAI 2011]
Copeland$^{\alpha}$ (friendly tie-breaking)	P [Bartholdi, Tovey and Trick, SCW 1989]	NP-hard [Faliszewski, Hemaspaandra and Schnoor, AAMAS 2008]
Ranked pairs	NP-hard [Xia, Zuckerman, Procaccia, Conitzer, and Rosenschein, IJCAI 2009]	NP-hard [Xia, Zuckerman, Procaccia, Conitzer, and Rosenschein, IJCAI 2009]
Schulze	P [Parkes and Xia, AAAI 2012]	P [Gaspers, Kalinowski, Narodytska and Walsh, AAMAS 2013]



Social Choice Theory




Soc Choice Welfare (1989) 6: 227–241

Social Choice and Welfare

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.

Social Choice Theory

Computational Social Choice

References

- “Sports elimination via max flow” with IPL teams: https://www.youtube.com/watch?v=XK6qZjHWo9A

- When it’s easy to recognize the *existence* of a beneficial manipulation but hard to *find* a manipulative vote.

  “Search versus Decision for Election Manipulation Problems”
  Hemaspaandra, Hemaspaandra, and Menton
  https://dl.acm.org/doi/10.1145/3369937