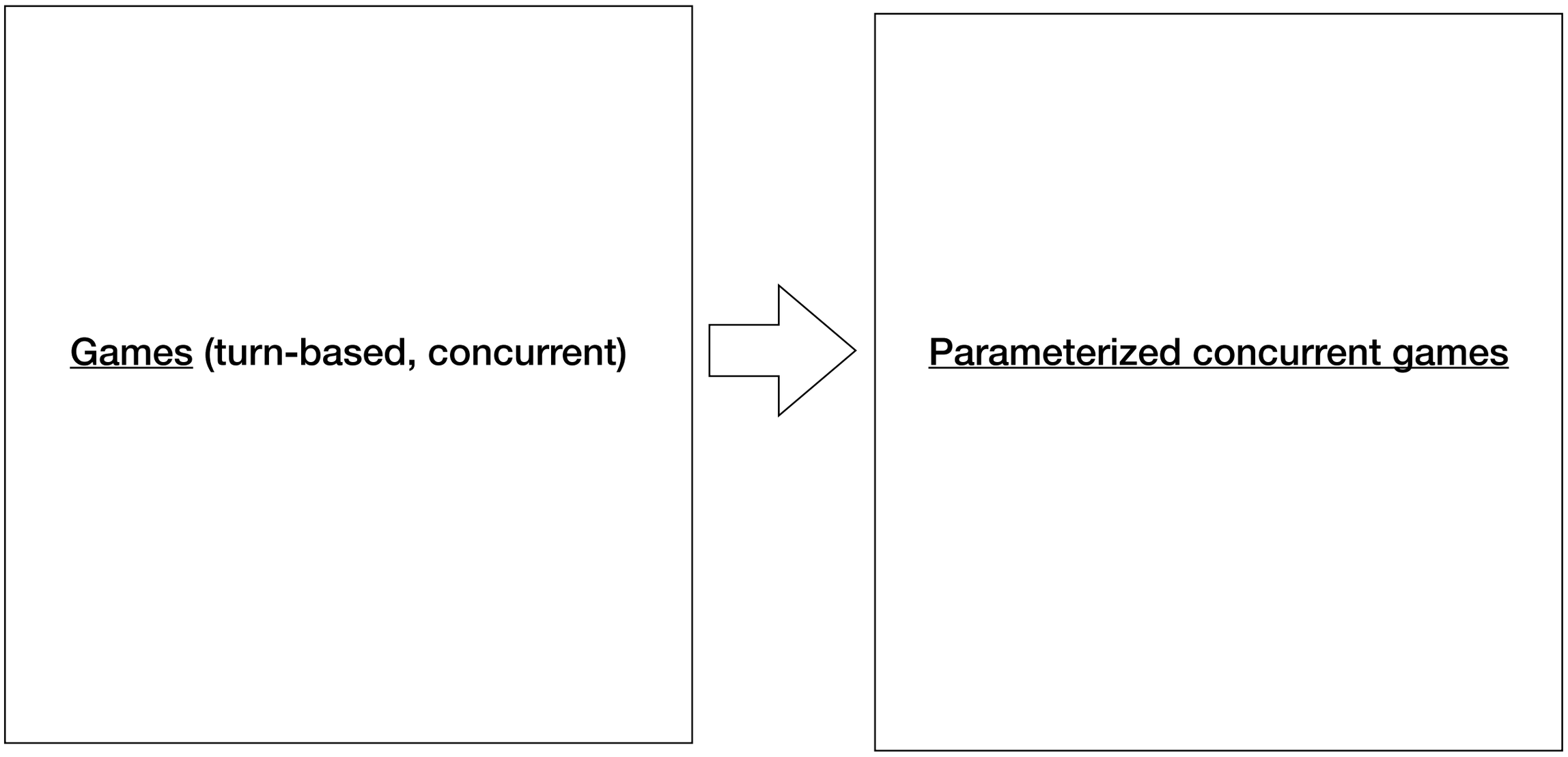
Concurrent Parameterized Games

Nathalie Bertrand, Patricia Bouyer and Anirban Majumdar

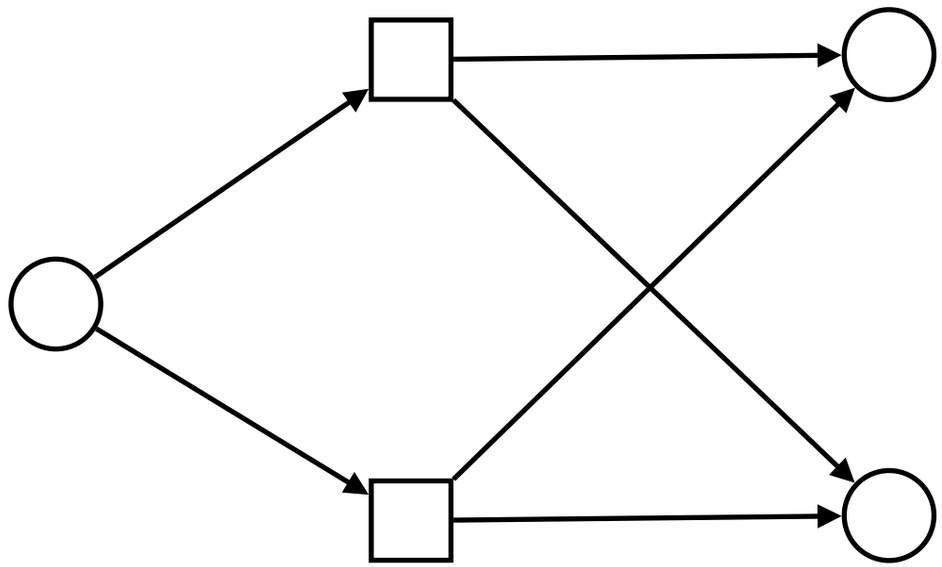
LSV, ENS Paris-Saclay, France
Inria Rennes, France

FSTTCS 2019

Mini-map

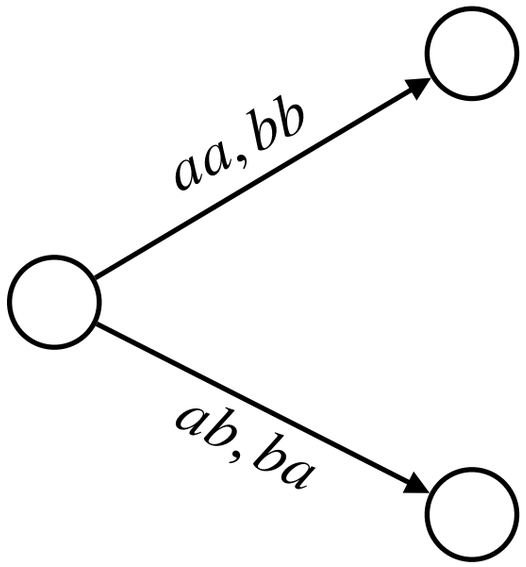

Games (turn-based, concurrent)
Parameterized concurrent games

Turn based 2-player Games



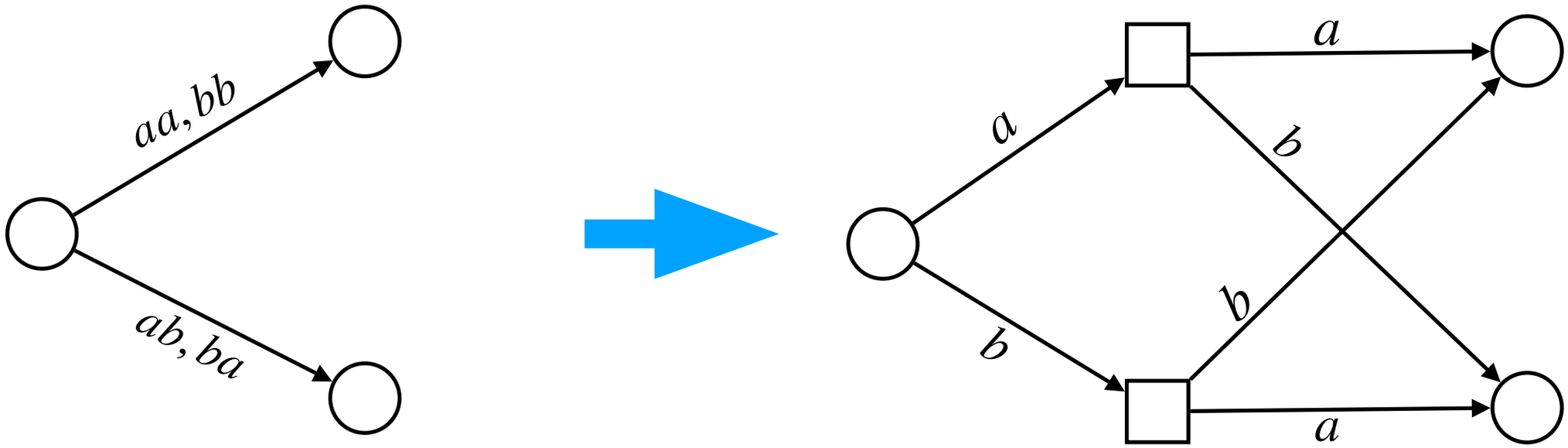
- P_1 (◯) vs P_2 (□)
- Positional (memoryless) strategies for Reachability, Safety…

Concurrent 2-player Games



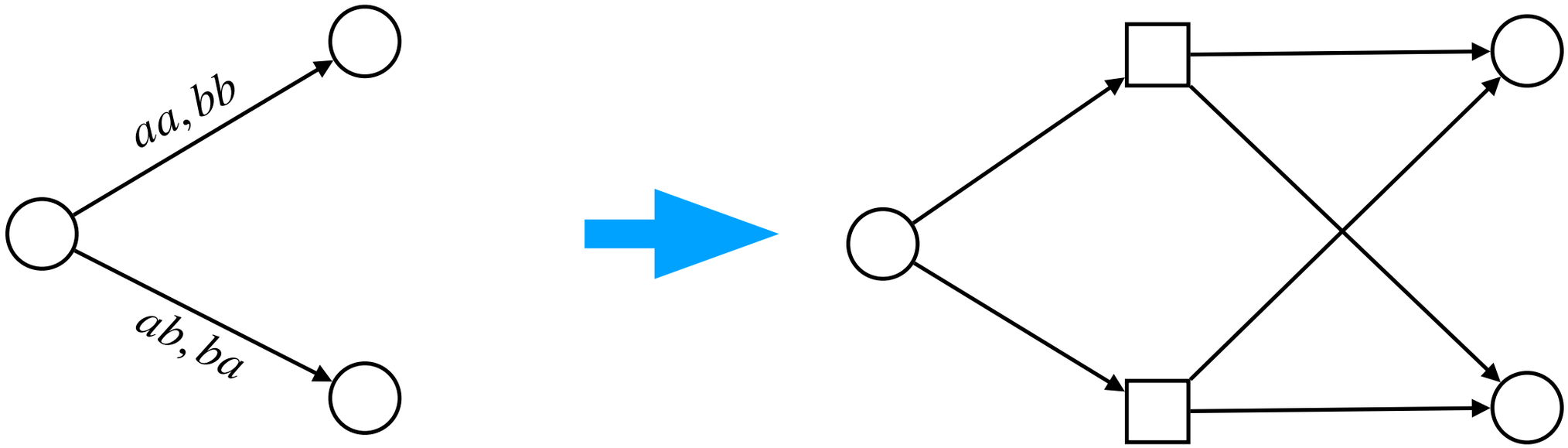
- Actions : $\Sigma = \{a, b\}$
- P_1, P_2 - choose action simultaneously
- Unique successor

Concurrent 2-player Games



- Actions : $\Sigma = \{a, b\}$
- P_1, P_2 - choose action simultaneously
- Unique successor

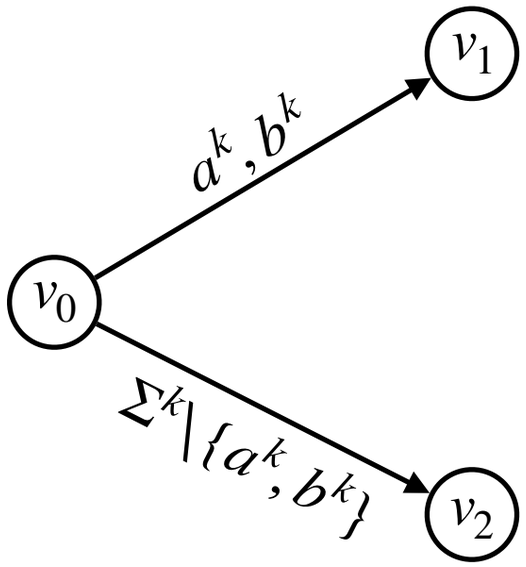
Concurrent 2-player Games



P_1 has winning strategy	⟺	○ has winning strategy

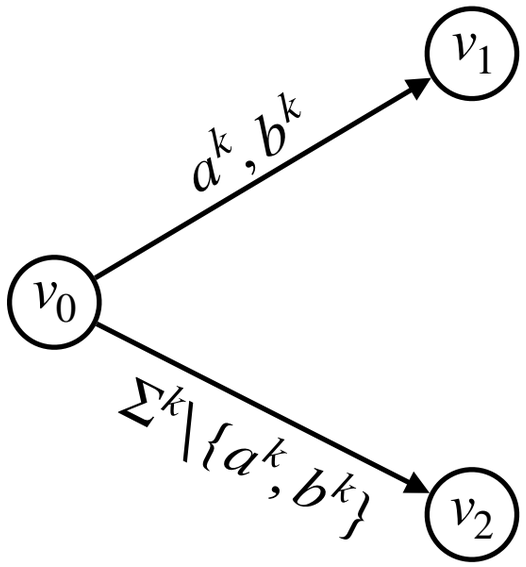
- Actions : $\Sigma = \{a, b\}$
- P_1, P_2 - choose action simultaneously
- Unique successor

Concurrent k-player Reachability games



- Actions : $\Sigma = \{a, b\}$
- $P_1, \ldots, P_k$ - choose action simultaneously

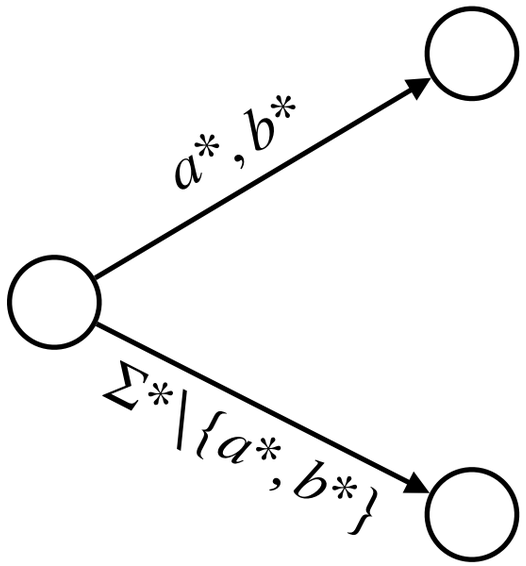
Concurrent k-player Reachability games



- Actions : $\Sigma = \{a, b\}$
- $P_1, \ldots, P_k$ - choose action simultaneously
- Eg: P_1 chooses 'a' ; $P_2, \ldots, P_k$ all choose 'b'

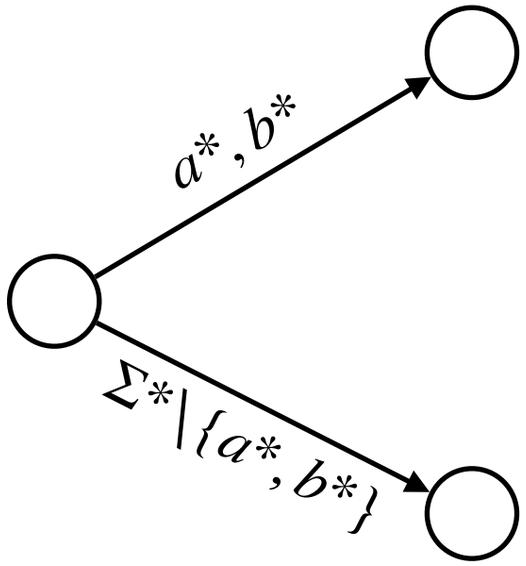
  The game goes from v_0 to v_2

Concurrent parameterized Games

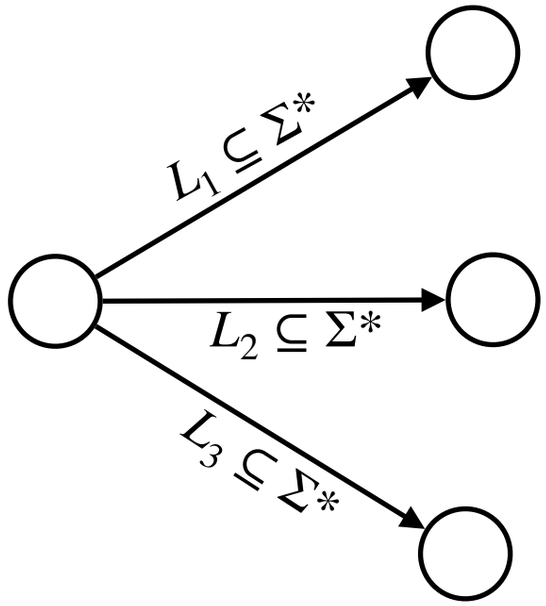


- Actions : $\Sigma = \{a, b\}$
- Fixed but unknown (parameter) number of players
- P_1 vs Rest of world (Env.)

Concurrent parameterized Games

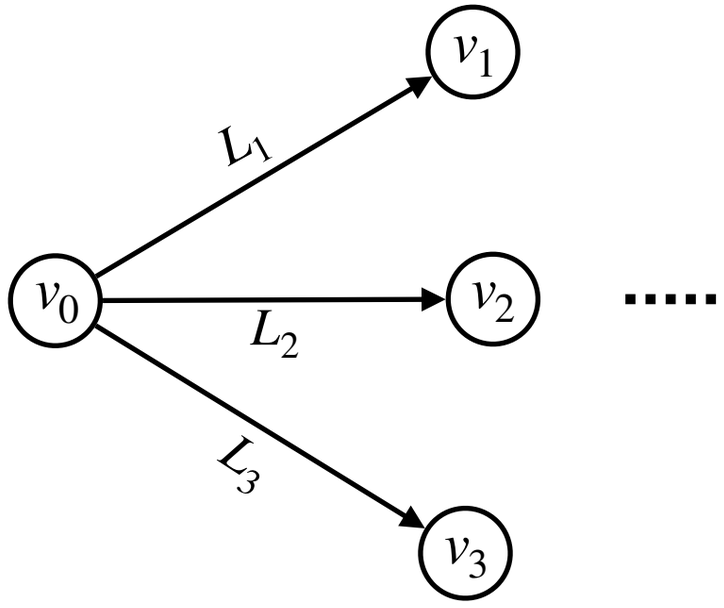


- Actions : $\Sigma = \{a, b\}$
- Fixed but unknown (parameter) number of players
- P_1 vs Rest of world (Env.)

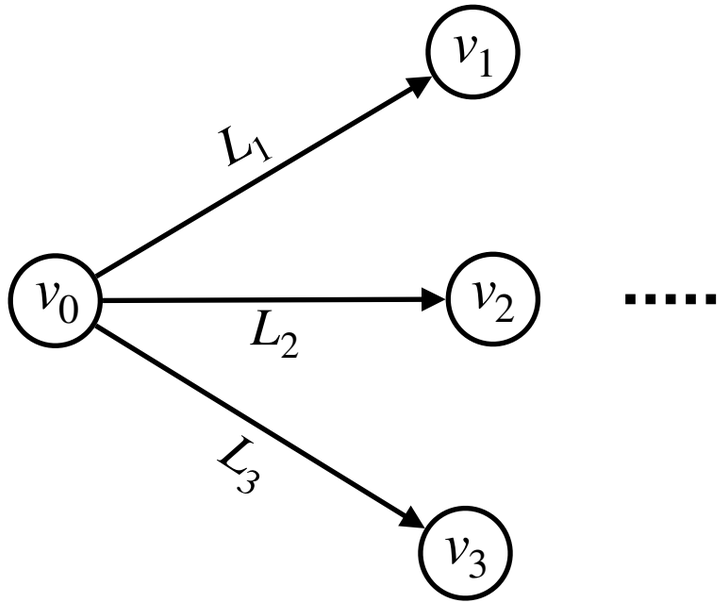


- L_1, L_2, L_3 Regular
- P_1 needs to win against all choices of others

Game explanation

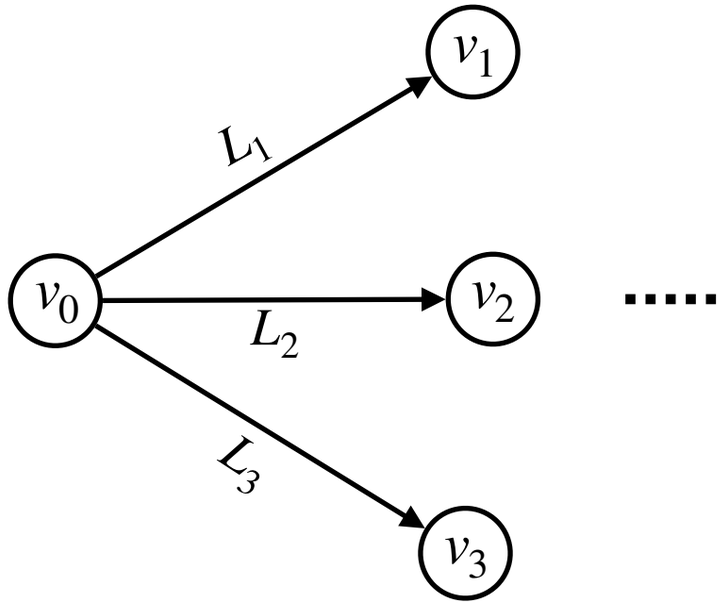


Game explanation



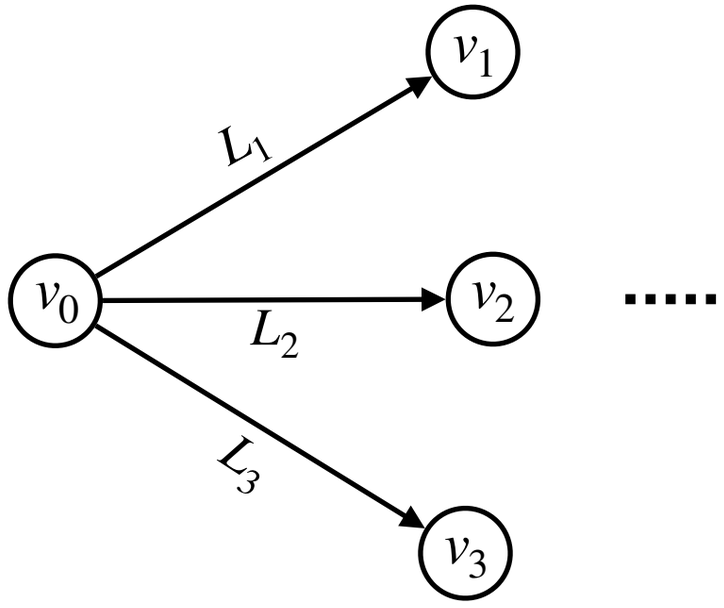
1. Env chooses k: #players (unknown to P_1)

Game explanation



1. Env chooses k: #players (unknown to P_1)
2. P_1 chooses an action 'a_1'
   Others choose actions '$a_2, \ldots, a_k$'

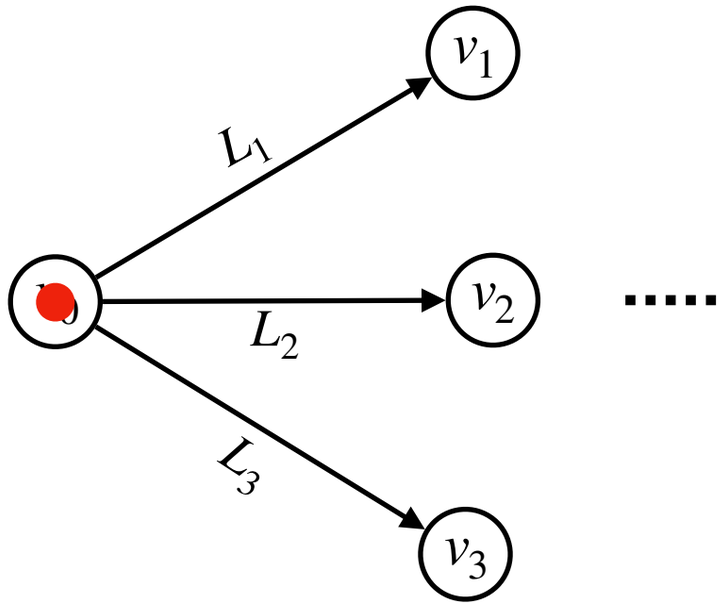
Game explanation



1. Env chooses k: #players (unknown to P_1)
2. P_1 chooses an action 'a_1'
   Others choose actions '$a_2, \ldots, a_k$'
3. This forms a word $w = a_1 a_2 \ldots a_k$

   Env chooses v_i s.t. $w \in L_i$

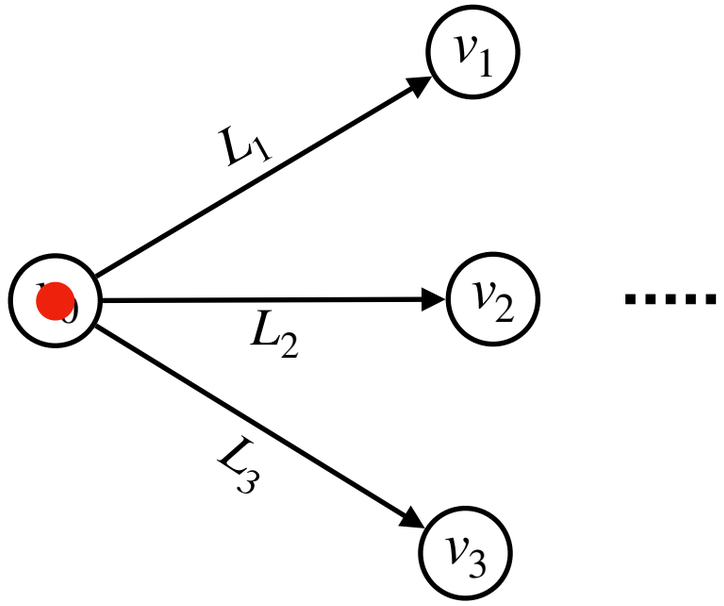
Game explanation



1. Env chooses k: #players (unknown to P_1)
2. P_1 chooses an action 'a_1'
   Others choose actions '$a_2, \ldots, a_k$'
3. This forms a word $w = a_1 a_2 \ldots a_k$

   Env chooses v_i s.t. $w \in L_i$

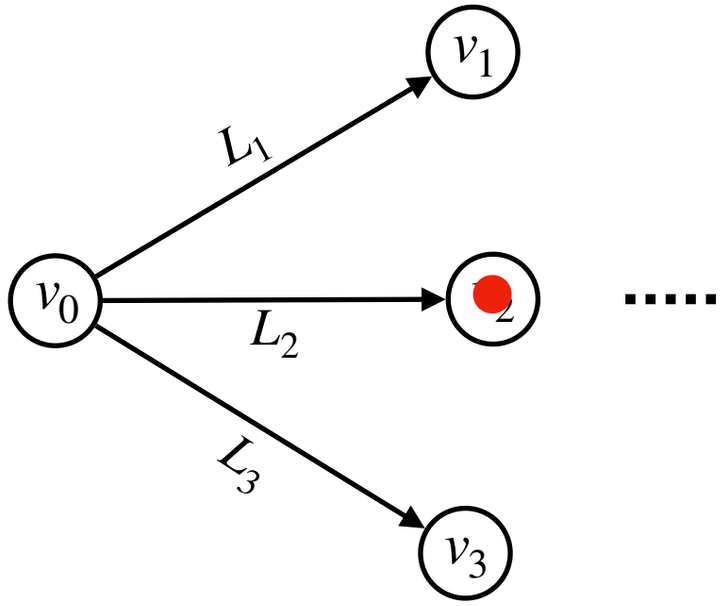
Game explanation



1. Env chooses k: #players (unknown to P_1)
2. P_1 chooses an action 'a_1'
   Others choose actions '$a_2, \ldots, a_k$'
3. This forms a word $w = a_1 a_2 \ldots a_k$

   Env chooses v_i s.t. $w \in L_i$
4. Game proceeds to v_i and goto step 2

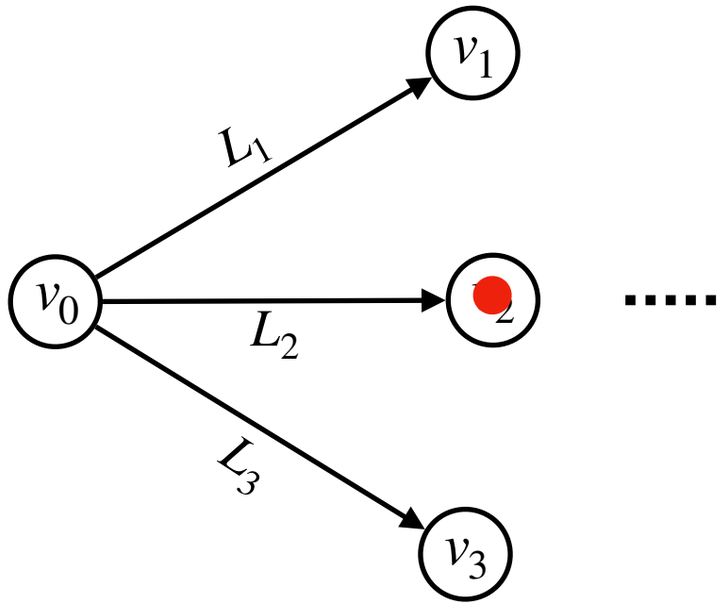
Game explanation



1. Env chooses k: #players (unknown to P_1)
2. P_1 chooses an action 'a_1'
   Others choose actions '$a_2, \ldots, a_k$'
3. This forms a word $w = a_1 a_2 \ldots a_k$

   Env chooses v_i s.t. $w \in L_i$
4. Game proceeds to v_i and goto step 2

Game explanation



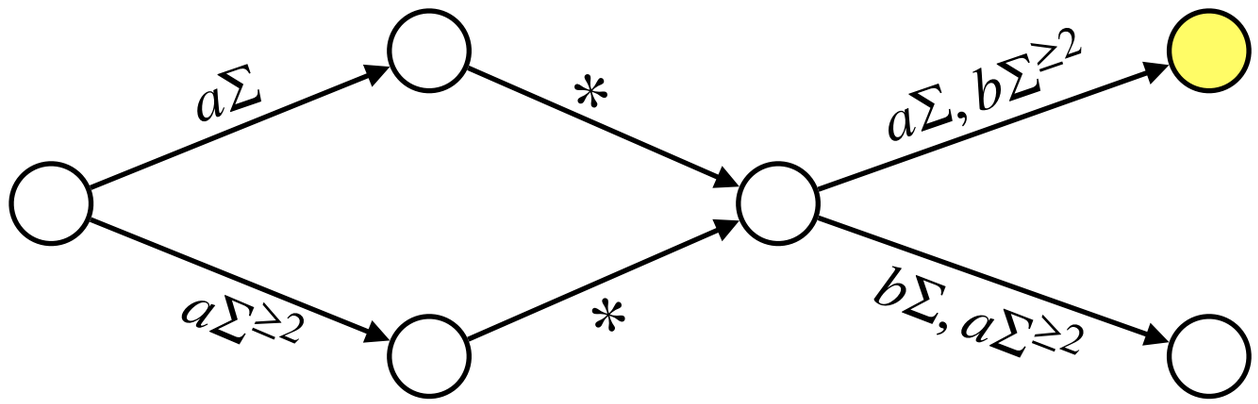
1. Env chooses k: #players (unknown to P_1)
2. P_1 chooses an action 'a_1'
   Others choose actions '$a_2, \ldots, a_k$'
3. This forms a word $w = a_1 a_2 \ldots a_k$

   Env chooses v_i s.t. $w \in L_i$
4. Game proceeds to v_i and goto step 2

P_1 has to win against all choices of others, for all k

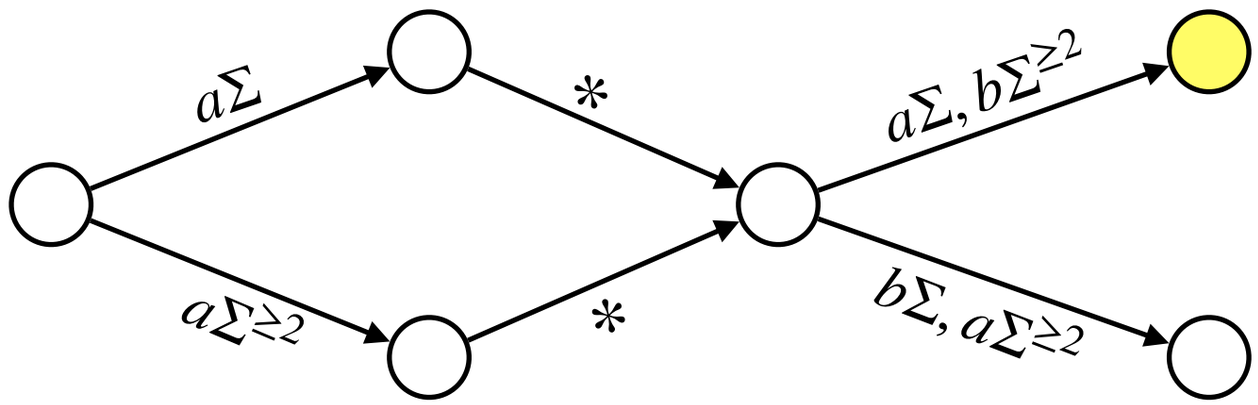
Illustrative example

- Question: Do positional strategies suffice?



Illustrative example

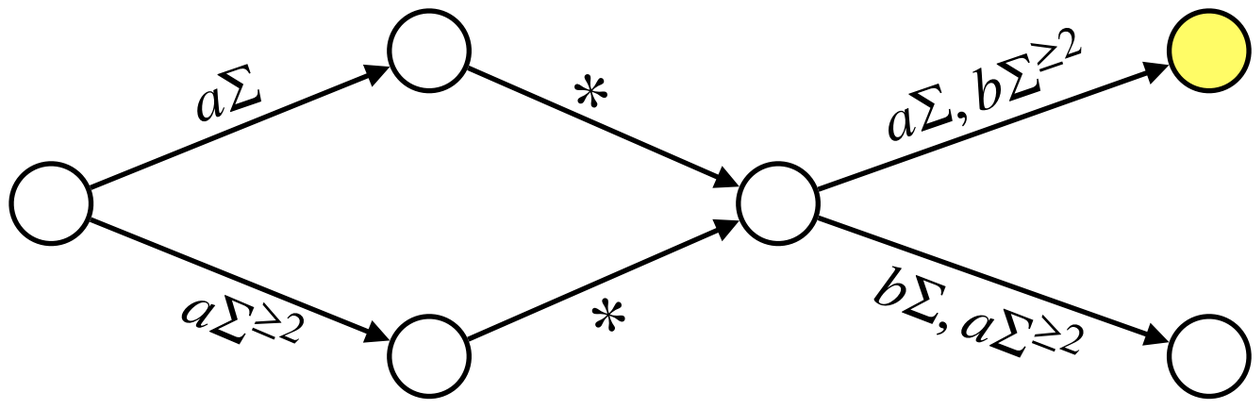
- Question: Do positional strategies suffice?



- Objective of P_1 : Reach
- P_1 has winning strategy

Illustrative example

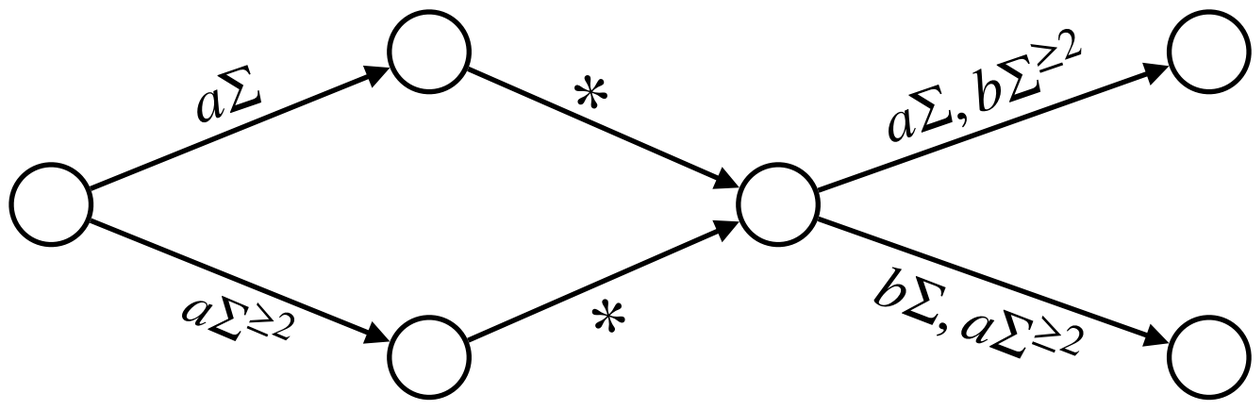
- Question: Do positional strategies suffice?



- Objective of P_1 : Reach
- P_1 has winning strategy
- No positional winning strategy

Problem Simplification

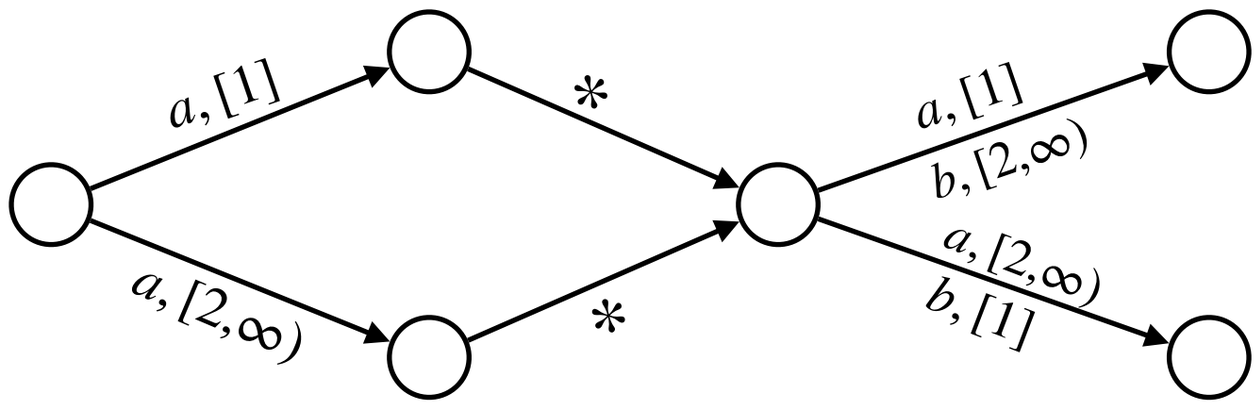
- Goal: Solve Parameterized game for P_1



- Observation: Only number of opponents matter (not their choices)

Problem Simplification

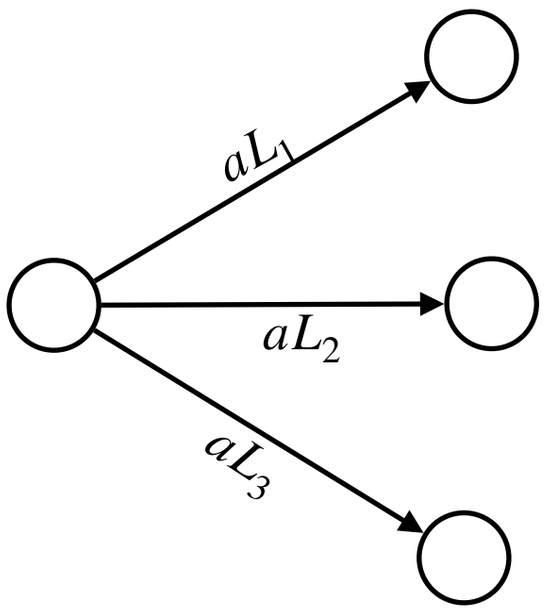
- Goal: Solve Parameterized game for P_1



- Observation: Only number of opponents matter (not their choices)

Problem Simplification

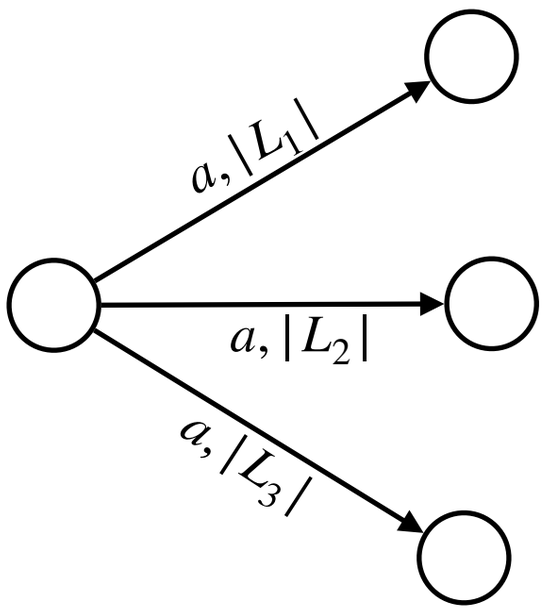
- Goal: Solve Parameterized game for P_1



- Observation: Only number of opponents matter for general case also

Problem Simplification

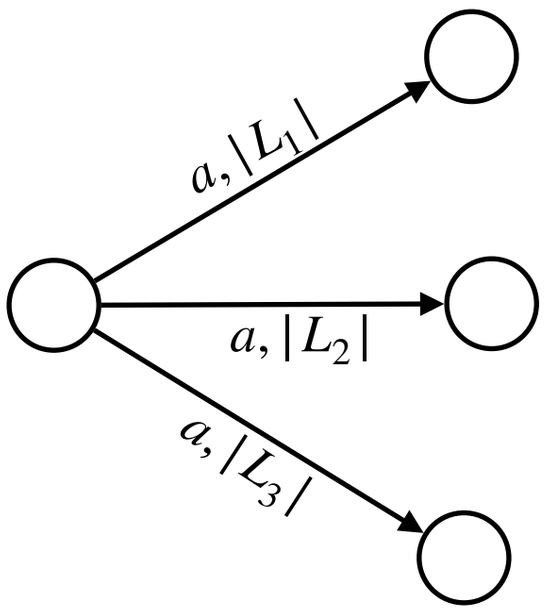
- Goal: Solve Parameterized game for P_1



- Observation: Only number of opponents matter for general case also

Problem Simplification

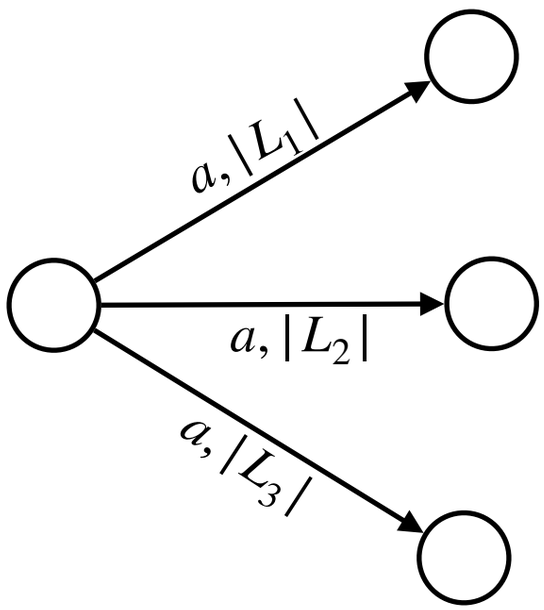
- Goal: Solve Parameterized game for P_1



- Observation: Only number of opponents matter for general case also
- L regular $\Rightarrow$ set of lengths of words $(|L|)$ is semilinear

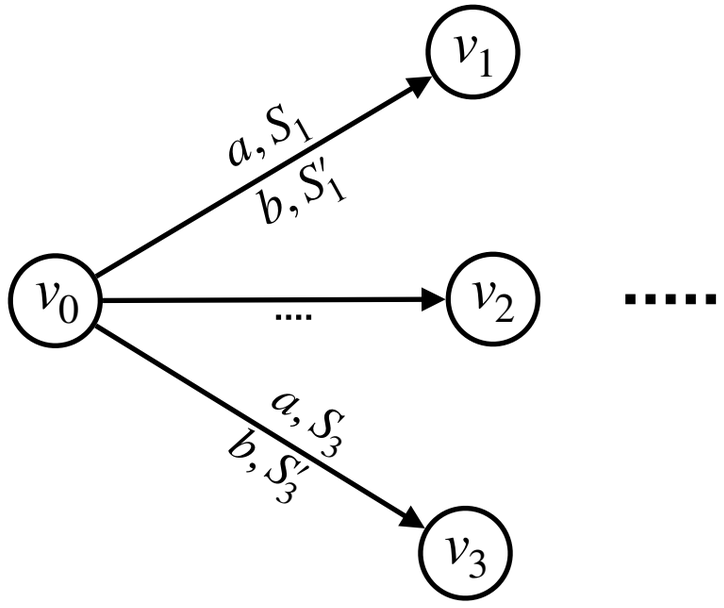
Problem Simplification

- Goal: Solve Parameterized game for P_1

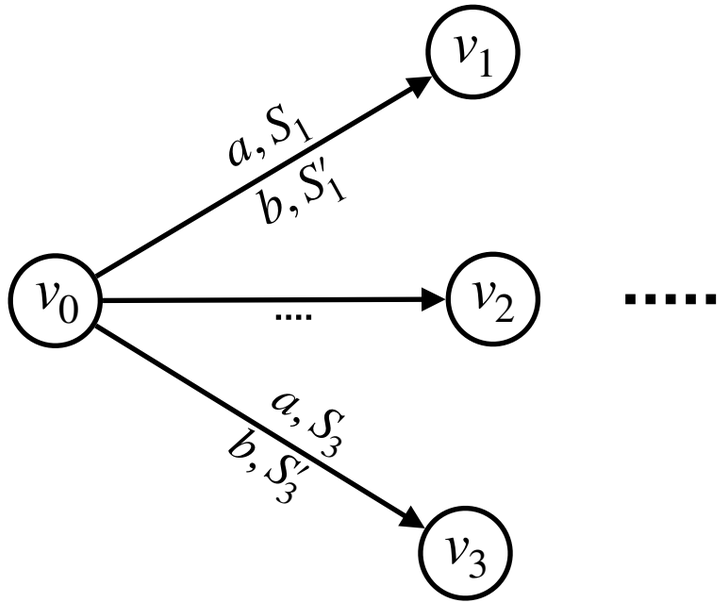


- Observation: Only number of opponents matter for general case also
- L regular $\Rightarrow$ set of lengths of words $(|L|)$ is semilinear
- Different cases on the representation of $|L|$
  - Intervals
  - Unions of intervals
  - Semilinear sets

Game simplification explanation

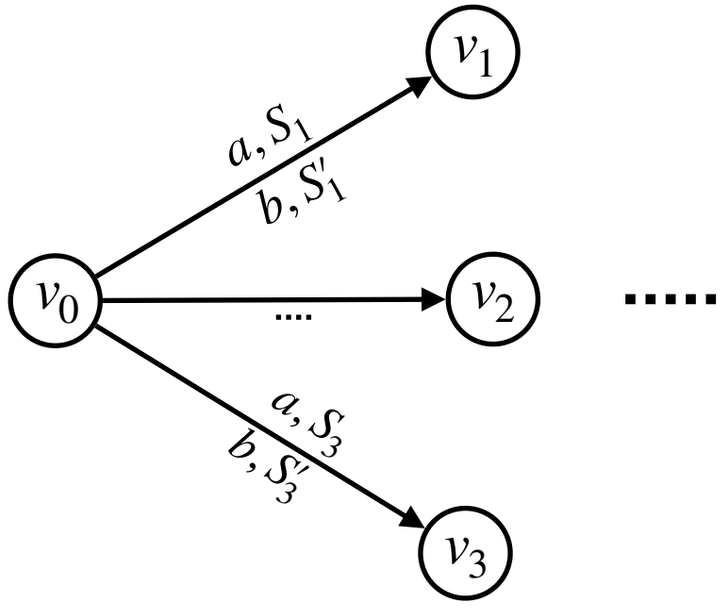


Game simplification explanation



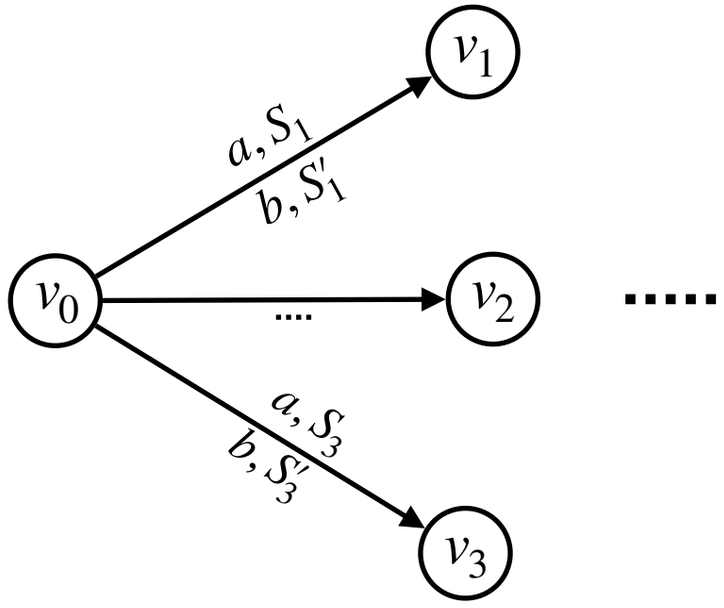
1. Env chooses k: #opponents (unknown to P_1)

Game simplification explanation



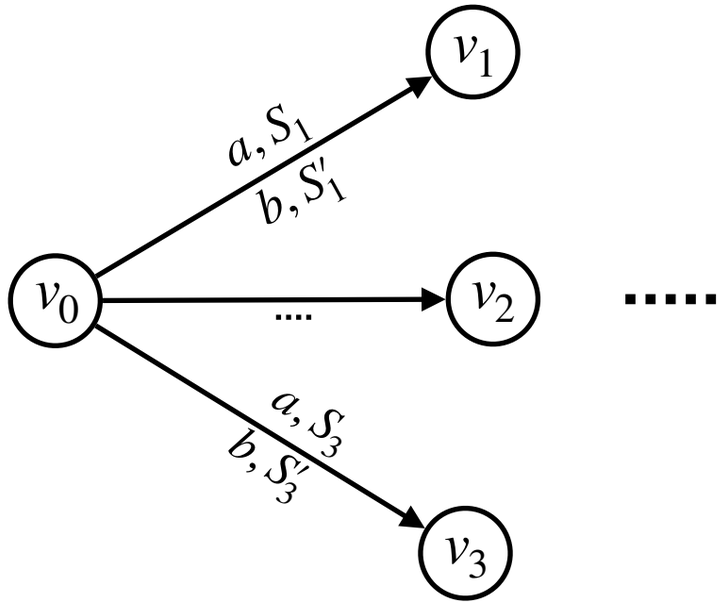
1. Env chooses k: #opponents (unknown to P_1)
2. P_1 chooses an action ‘ a ’ (or ‘ b ’)

Game simplification explanation



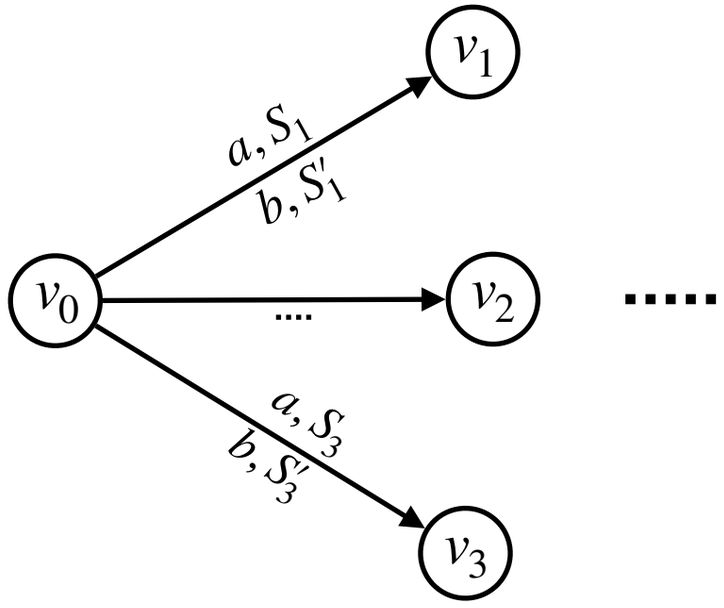
1. Env chooses k: #opponents (unknown to P_1)
2. P_1 chooses an action ‘ a ’ (or ‘ b ’)
3. Env chooses v_i s.t. $k \in S_i$ (resp. $k \in S_i'$)

Game simplification explanation



1. Env chooses k: #opponents (unknown to P_1)
2. P_1 chooses an action ‘ a ’ (or ‘ b ’)
3. Env chooses v_i s.t. $k \in S_i$ (resp. $k \in S'_i$)
4. Game proceeds to v_i and goto step 2

Game simplification explanation

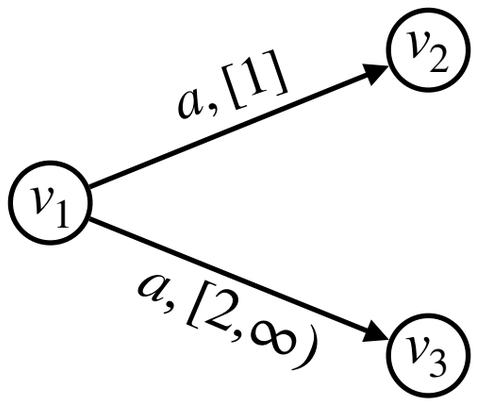


1. Env chooses k: #opponents (unknown to P_1)
2. P_1 chooses an action ‘ a ’ (or ‘ b ’)
3. Env chooses v_i s.t. $k \in S_i$ (resp. $k \in S_i'$)
4. Game proceeds to v_i and goto step 2

P_1 has to win against for all k

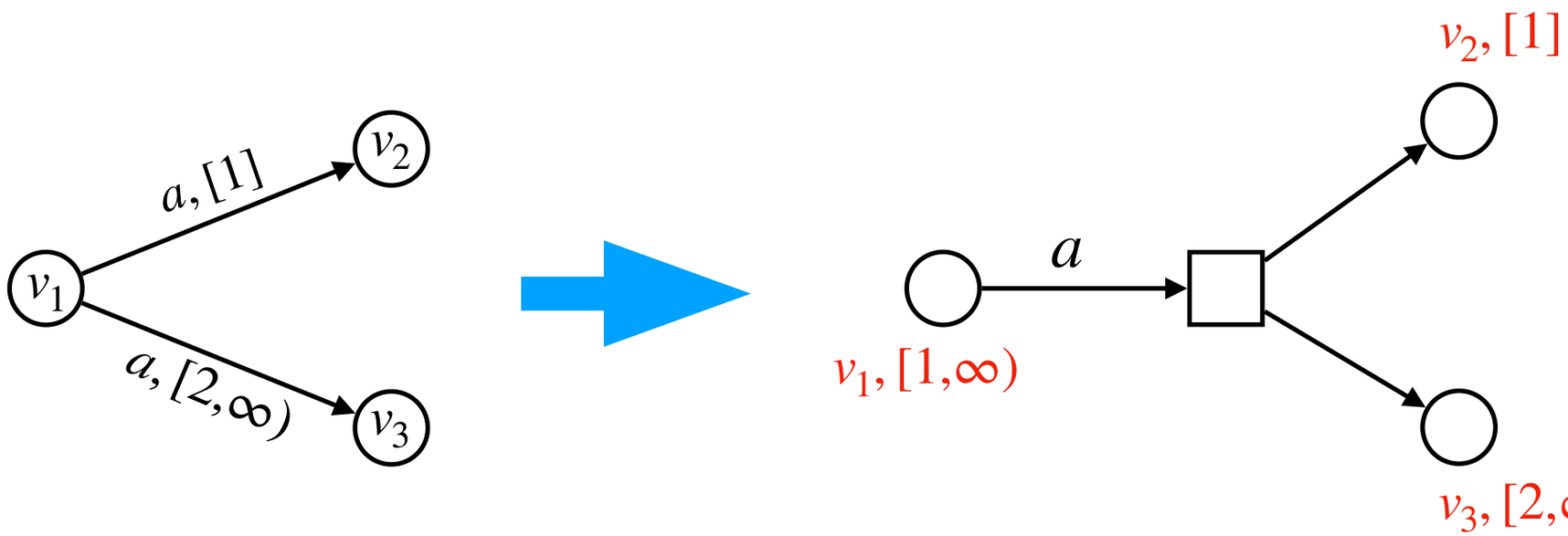
Resolution of the game

- Construct **Knowledge game** ($\mathscr{K}$)



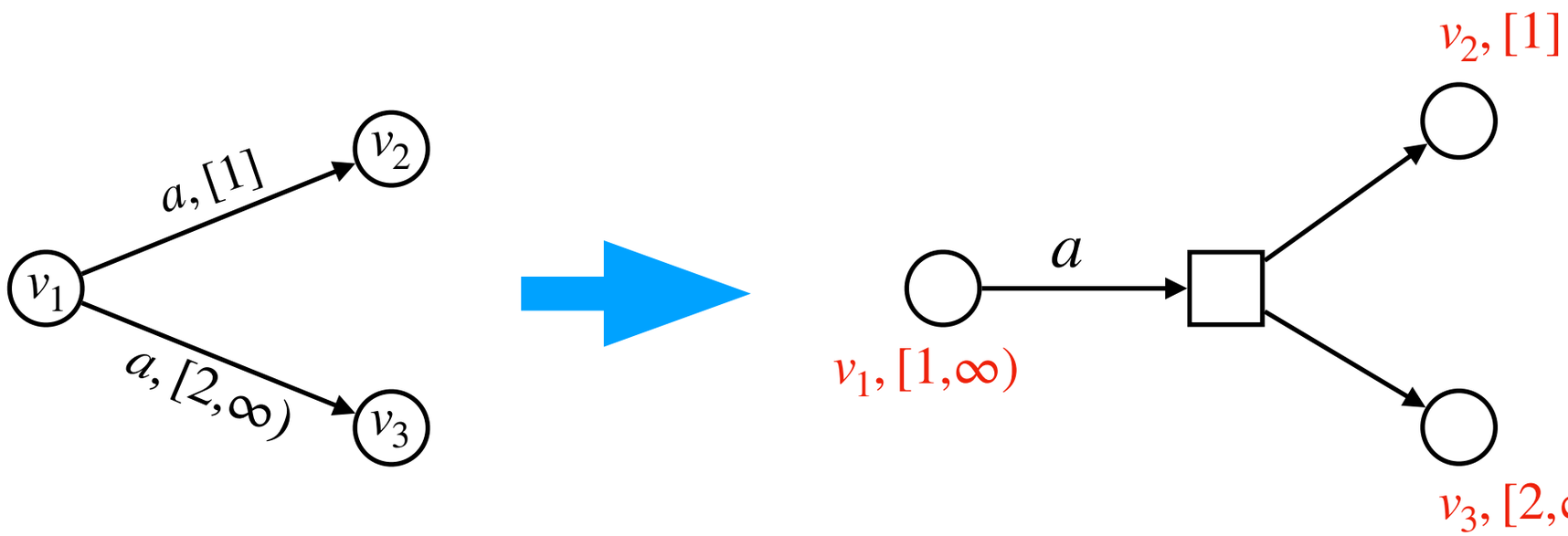
Resolution of the game

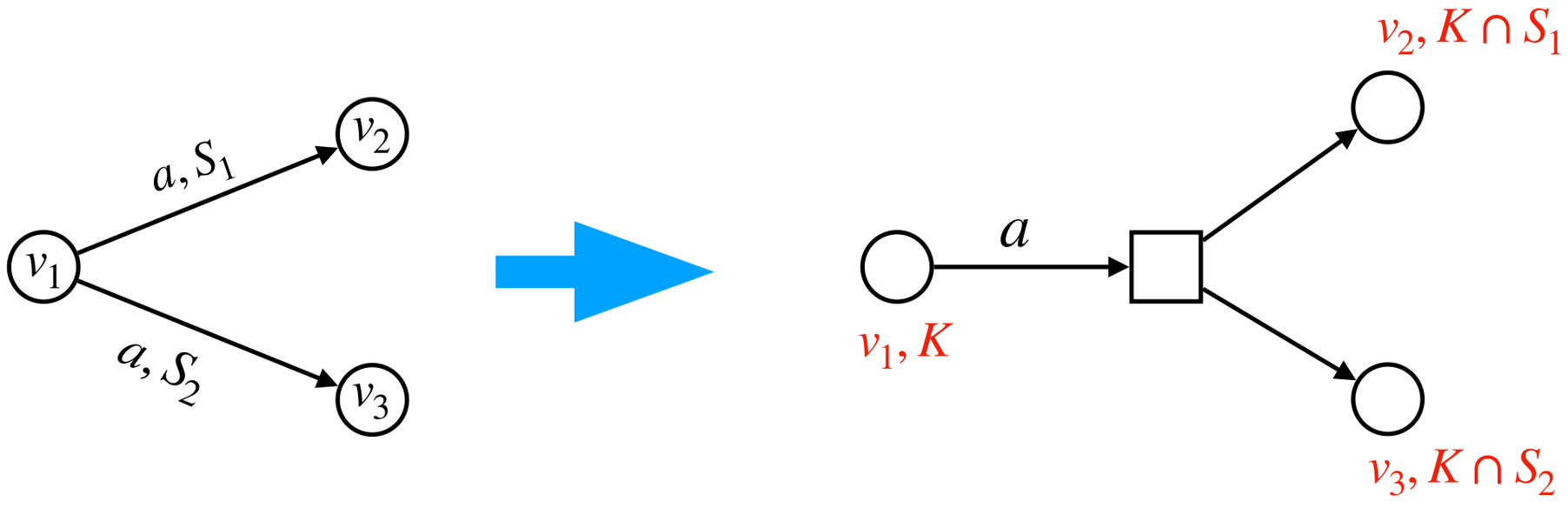
- Construct **Knowledge game** ($\mathscr{K}$)



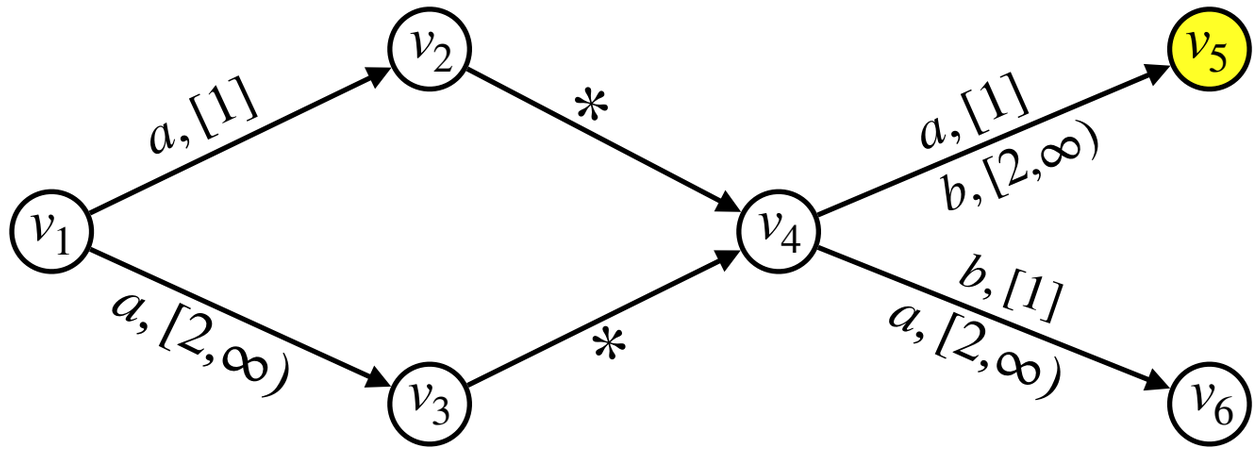
Resolution of the game

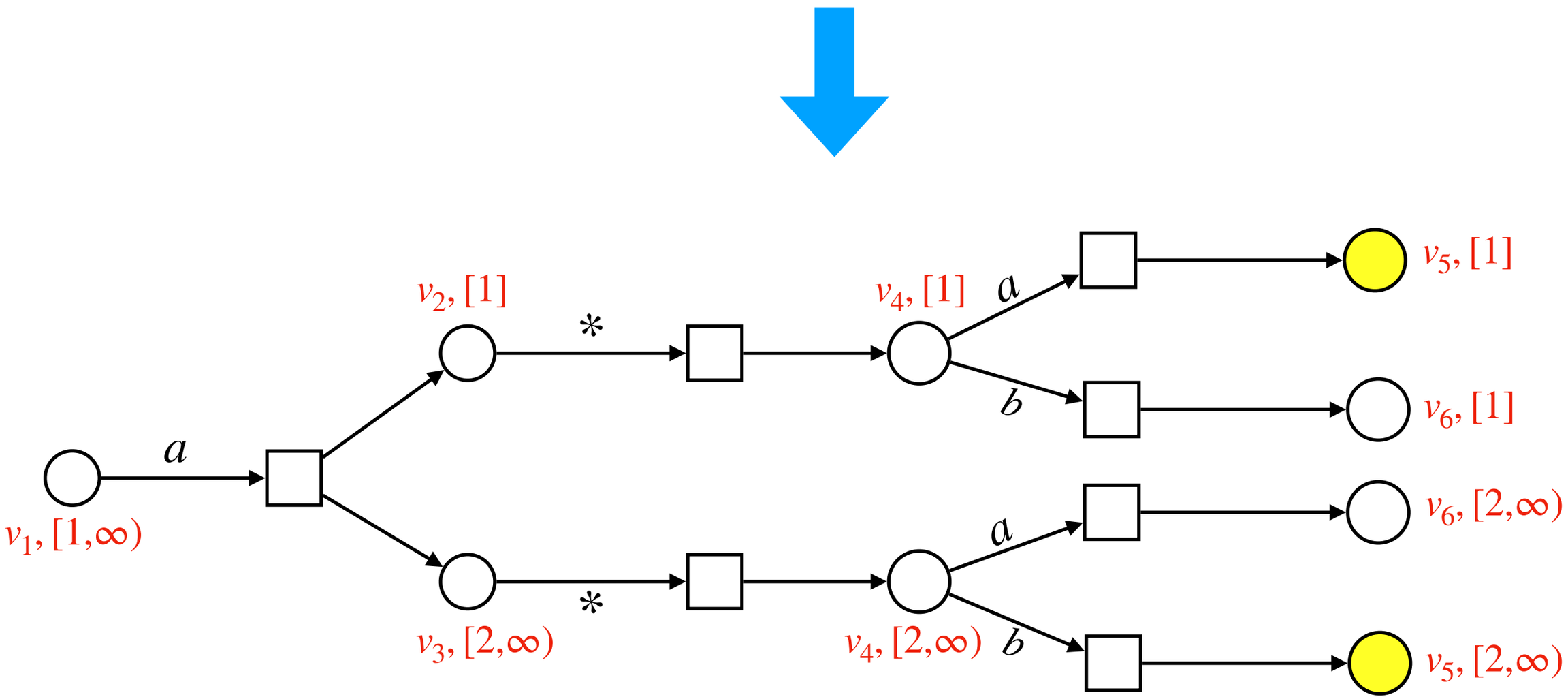
- Construct **Knowledge game** ($\mathscr{K}$)





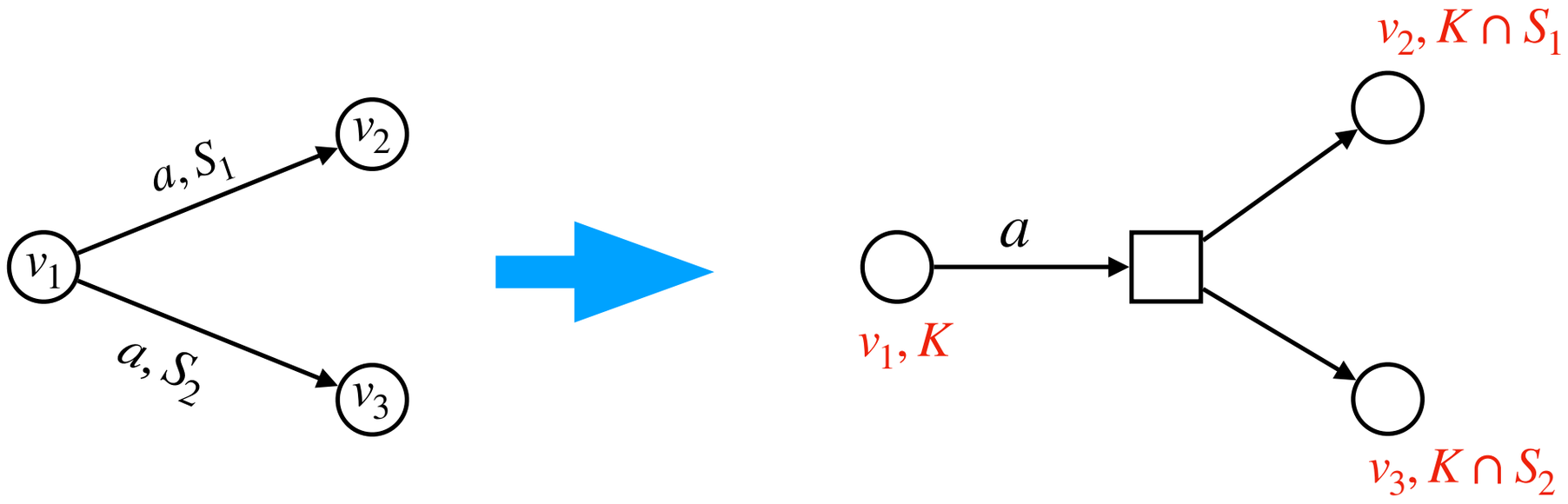
Illustrative example





Resolution of the game

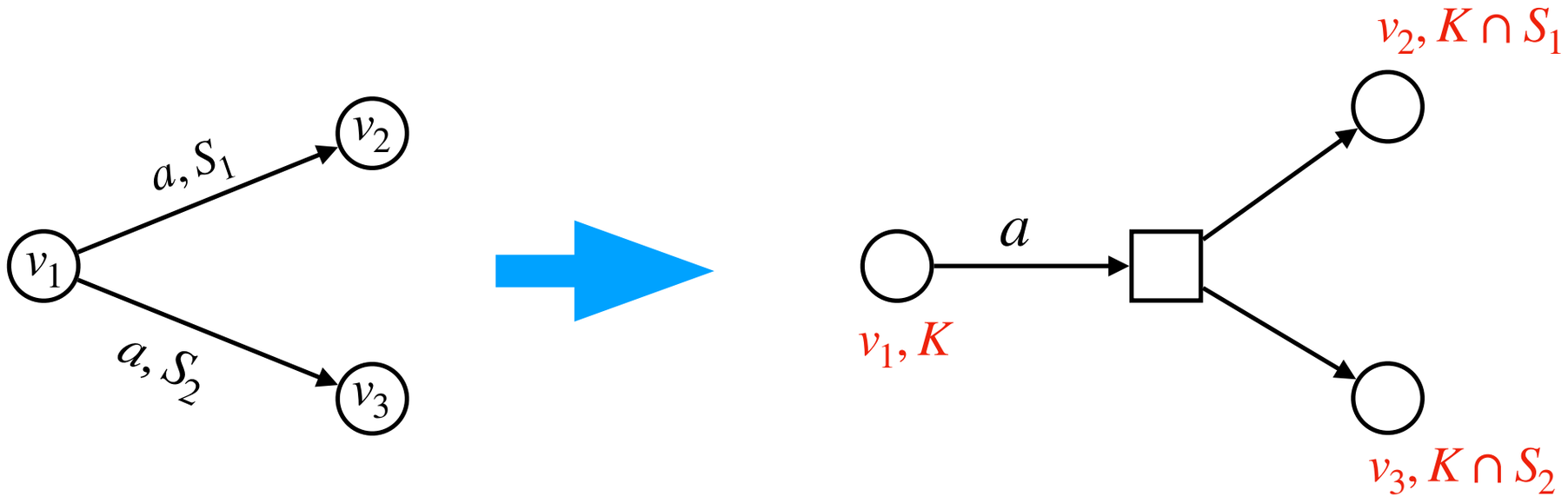
- Construct **Knowledge game** ($\mathscr{K}$)



P_1 has winning strategy	⟺	○ has winning strategy

Resolution of the game

- Construct **Knowledge game** ($\mathscr{K}$)

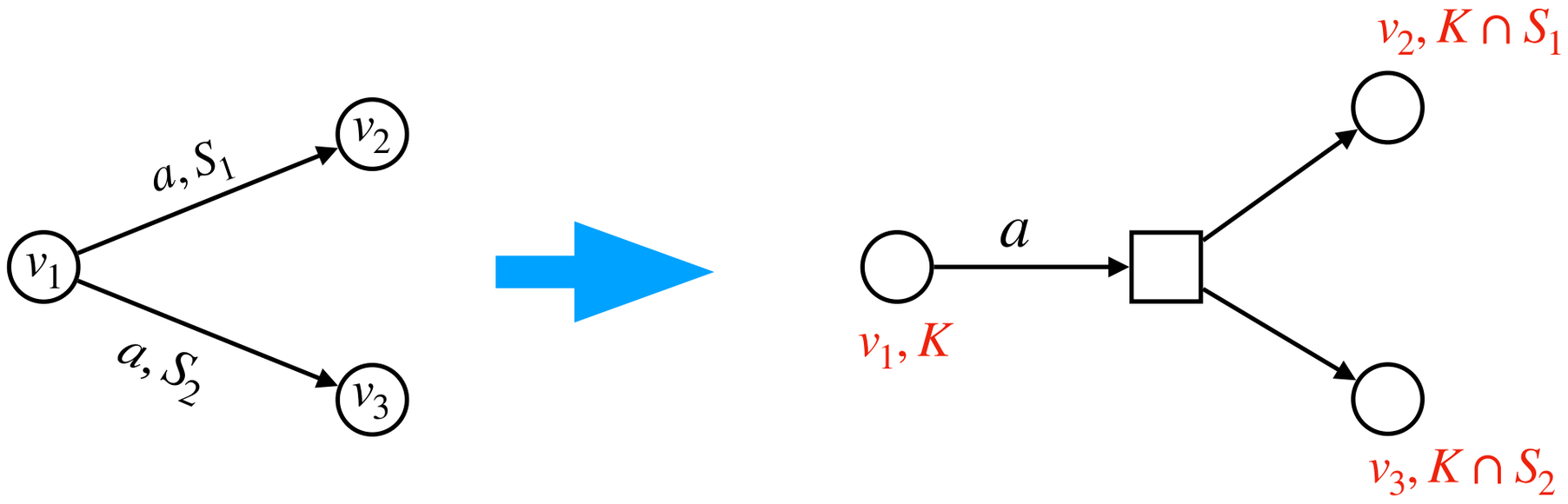


P_1 has winning strategy	⟺	○ has winning strategy

- $\mathscr{K}$ is finite: only intersections
- Solving Parameterized game is <u>decidable</u>

Resolution of the game

- Construct **Knowledge game** ($\mathscr{K}$)

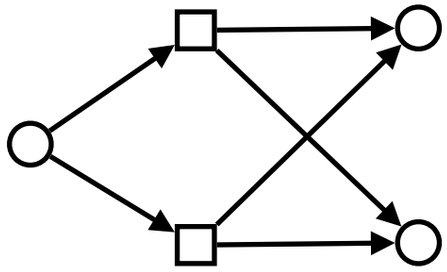


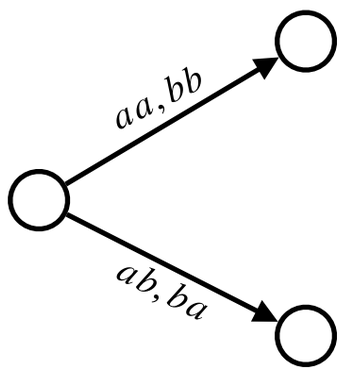
P_1 has winning strategy	⟺	○ has winning strategy

- $\mathscr{K}$ is finite: only intersections
- Solving Parameterized game is decidable
- Complexity of solving parameterized game?

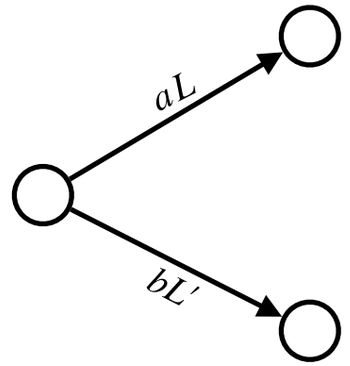
Mini-map

Games (turn-based, concurrent)





Parameterized concurrent games



- strategies need memory
- restrict to number of opponents
- finite knowledge game

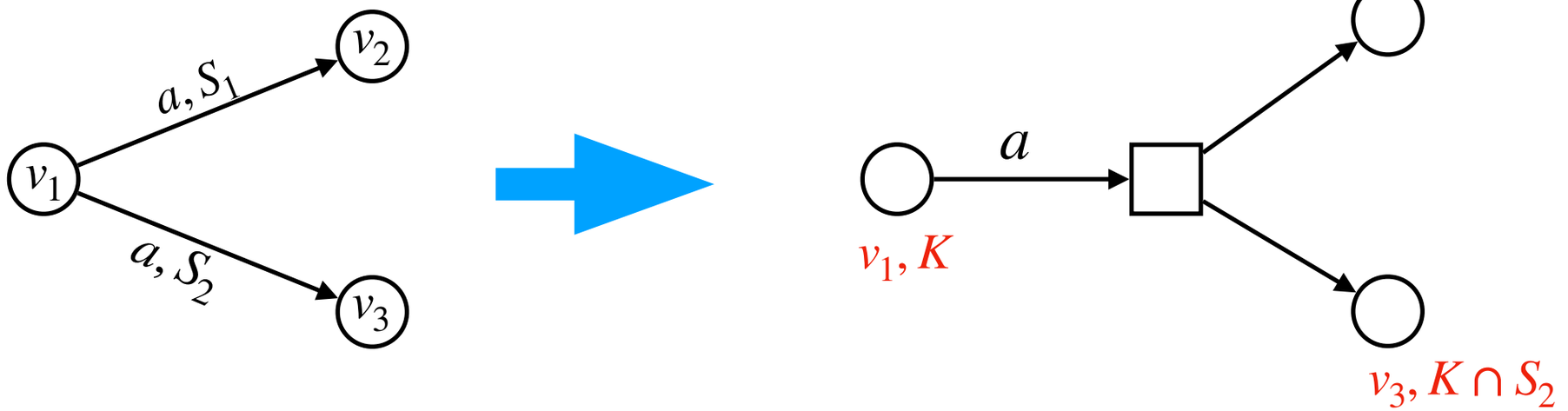
Mini-map: next

- Complexity results:

	Deterministic	Non-deterministic
Intervals	PTIME-complete[1]	
Unions of intervals	NP-complete[1]	PSPACE-complete[1]
Semilinear sets	PSPACE-complete[2]	

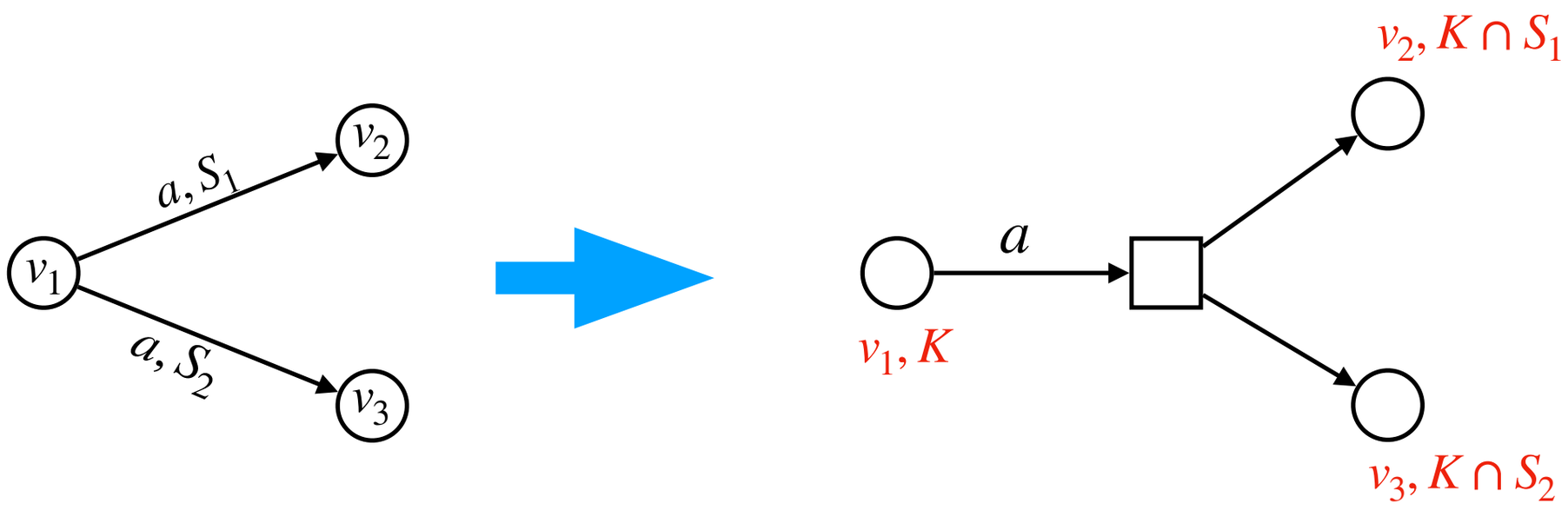
1. in #endpoints
2. in #semilinear sets

Intervals



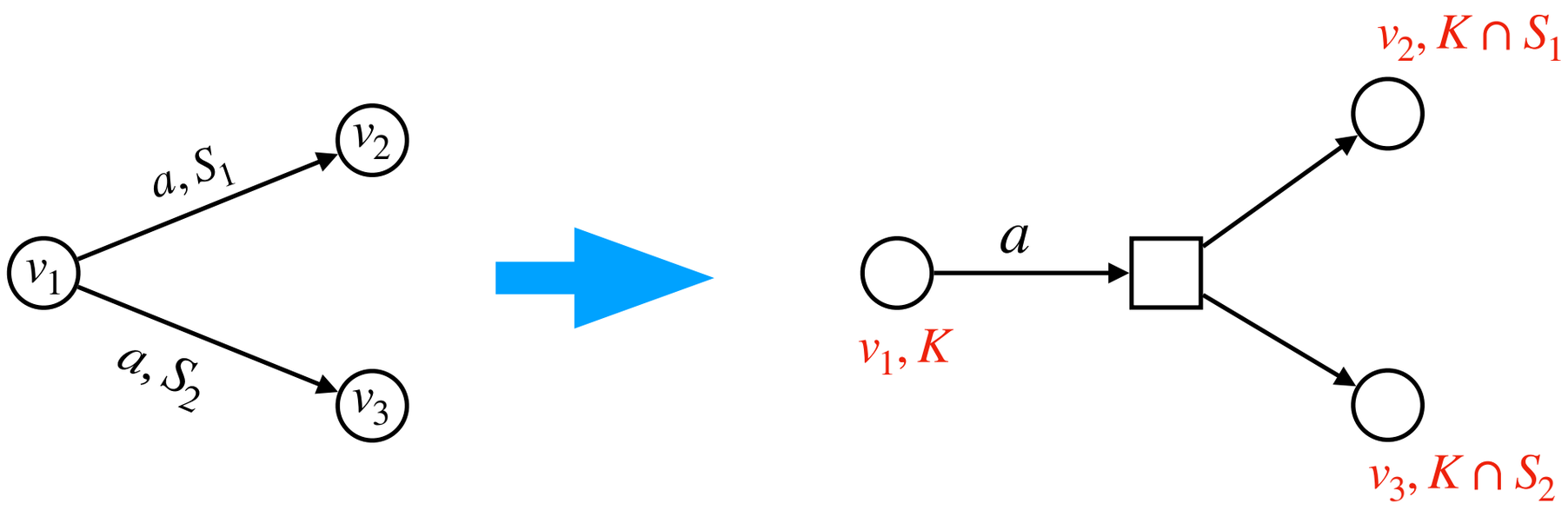
- Size of $\mathscr{K}$: quadratic in #endpoints
- Turn-based game solvable in Polynomial time

Intervals



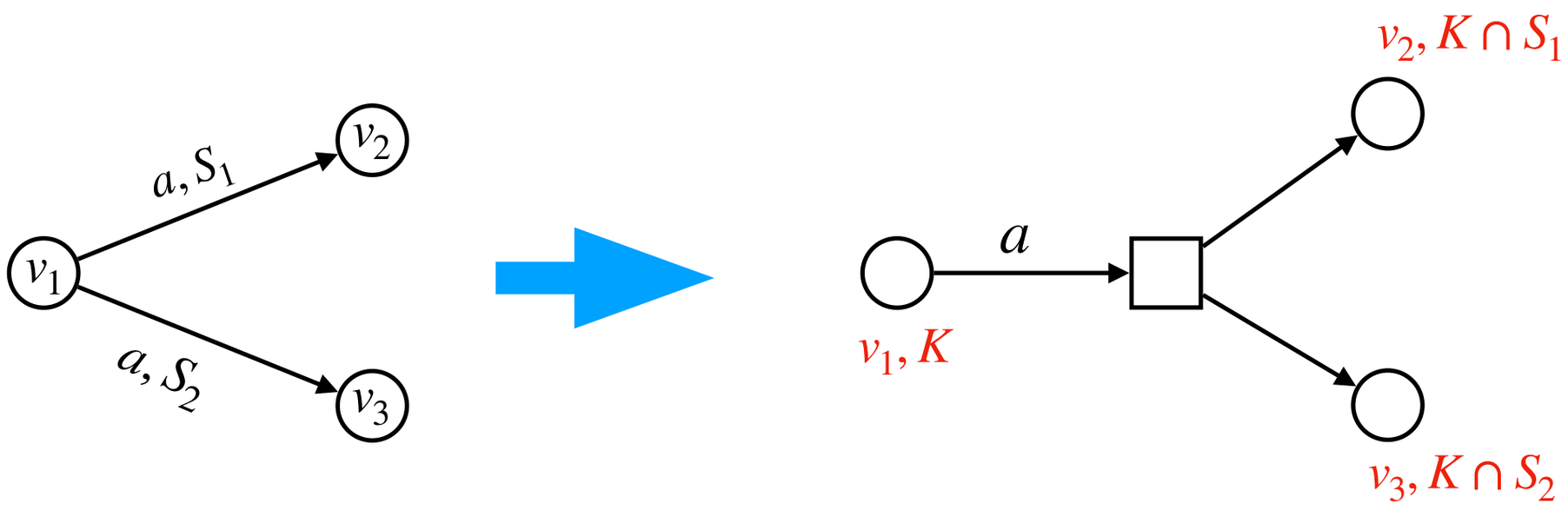
- Size of $\mathscr{K}$: quadratic in #endpoints
- Turn-based game solvable in Polynomial time
- Solving Parameterized game in P

Semilinear sets



- Size of $\mathscr{K}$: exponential in #semilinear predicates

Semilinear sets



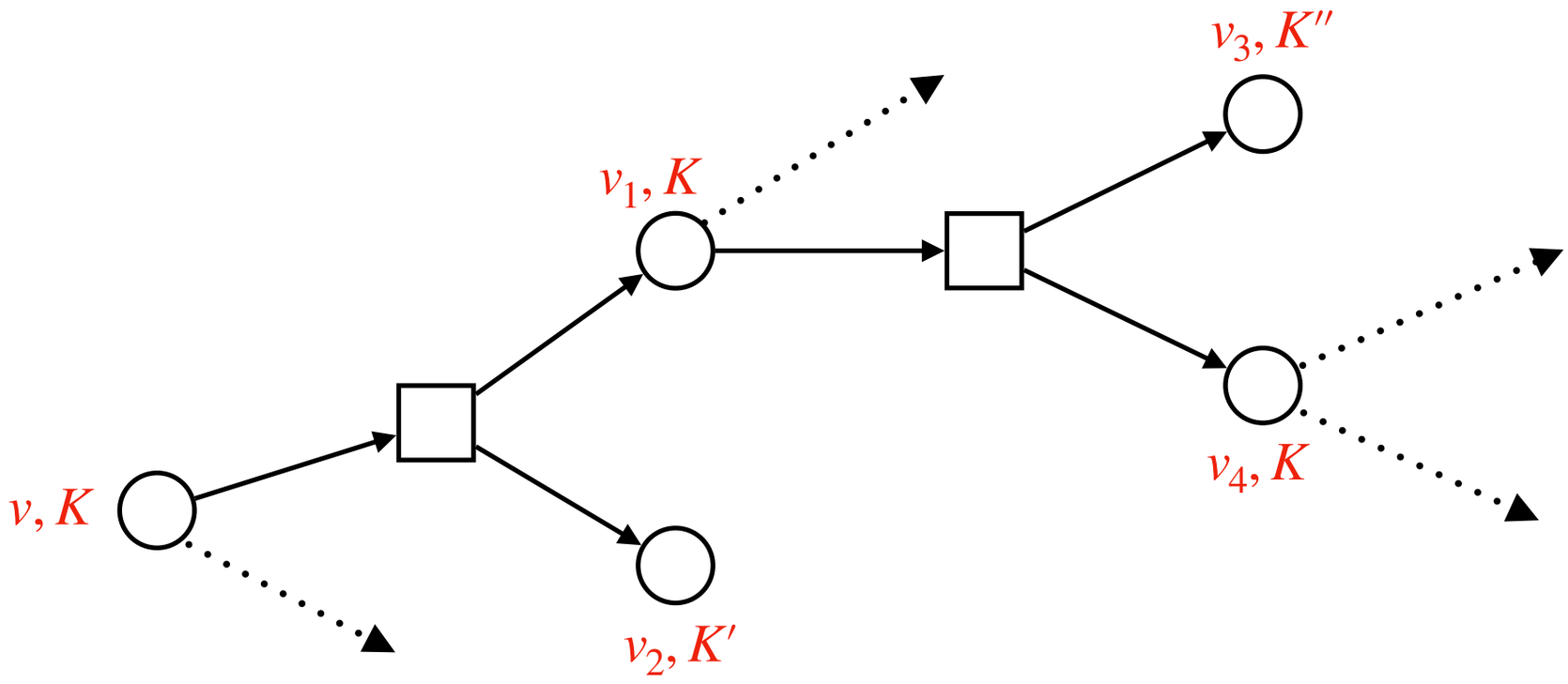
- Size of $\mathscr{K}$: exponential in #semilinear predicates
- Polynomial-space algorithm (next…)

Semilinear Sets - PSPACE upper bound

Step 1. Construct $\mathscr{K}[v, K]$ - restriction of $\mathscr{K}$

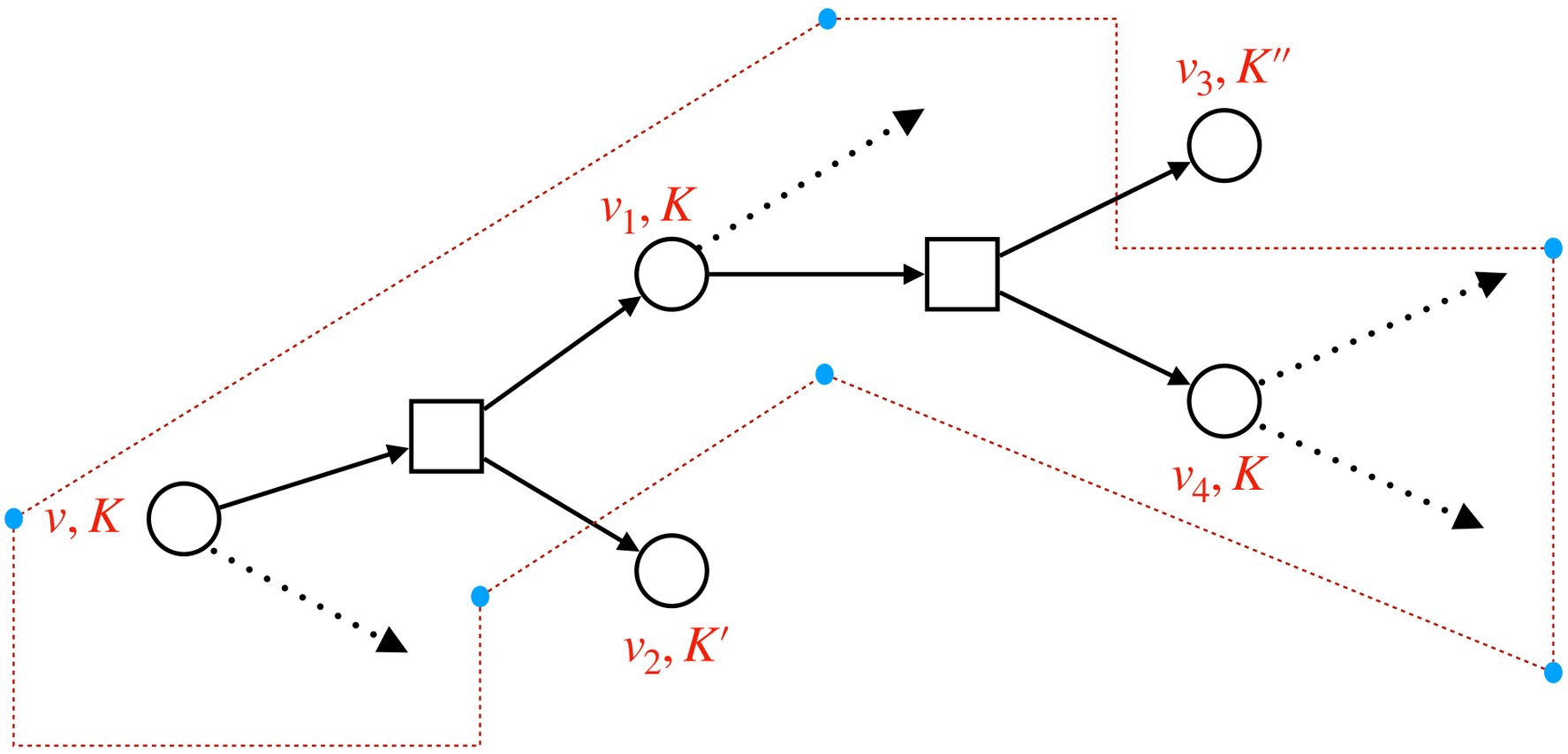
Semilinear Sets - PSPACE upper bound

Step 1. Construct $\mathscr{K}[v, K]$ - restriction of $\mathscr{K}$



Semilinear Sets - PSPACE upper bound

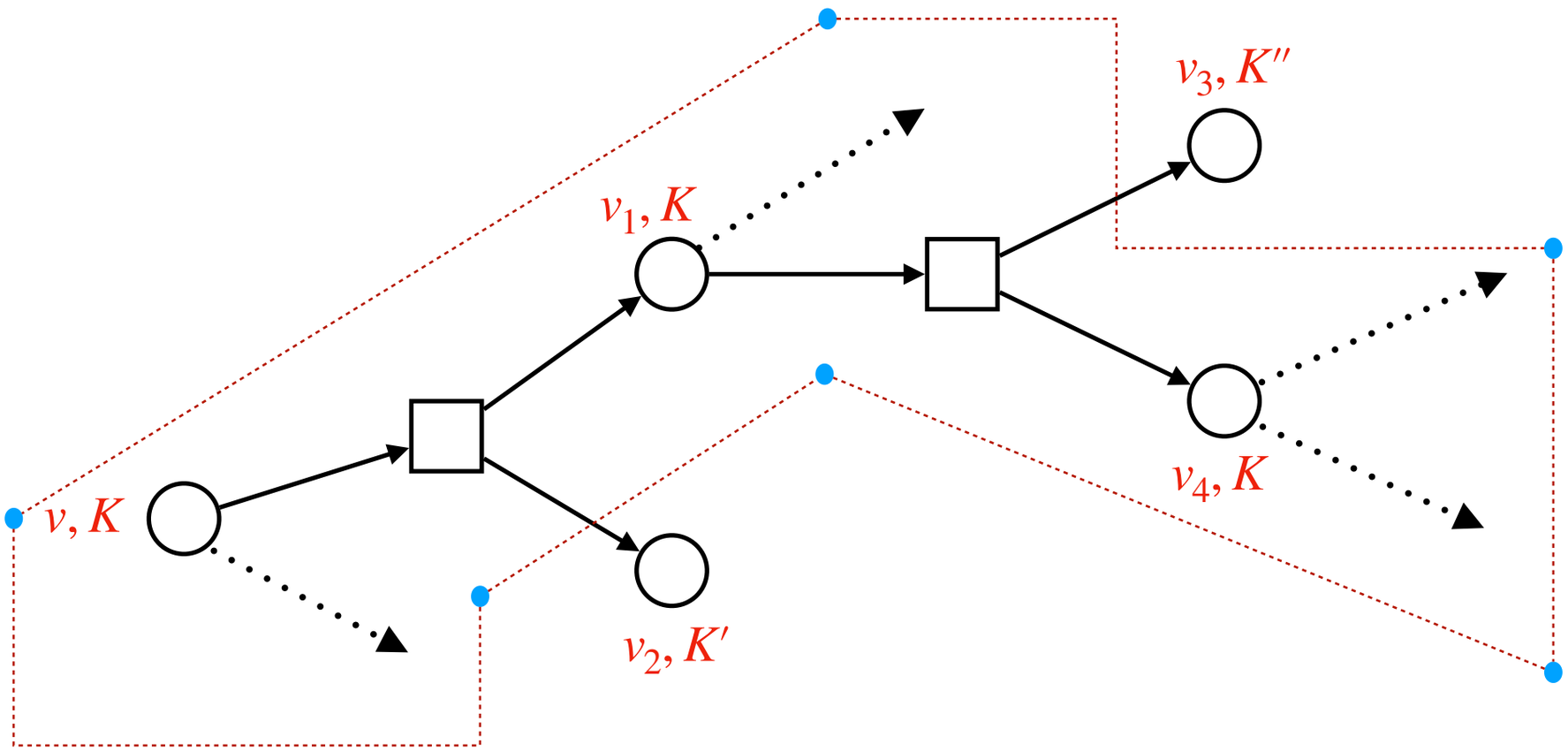
Step 1. Construct $\mathscr{K}[v, K]$ - <u>restriction</u> of $\mathscr{K}$



- Stop at any $K' \subsetneq K$

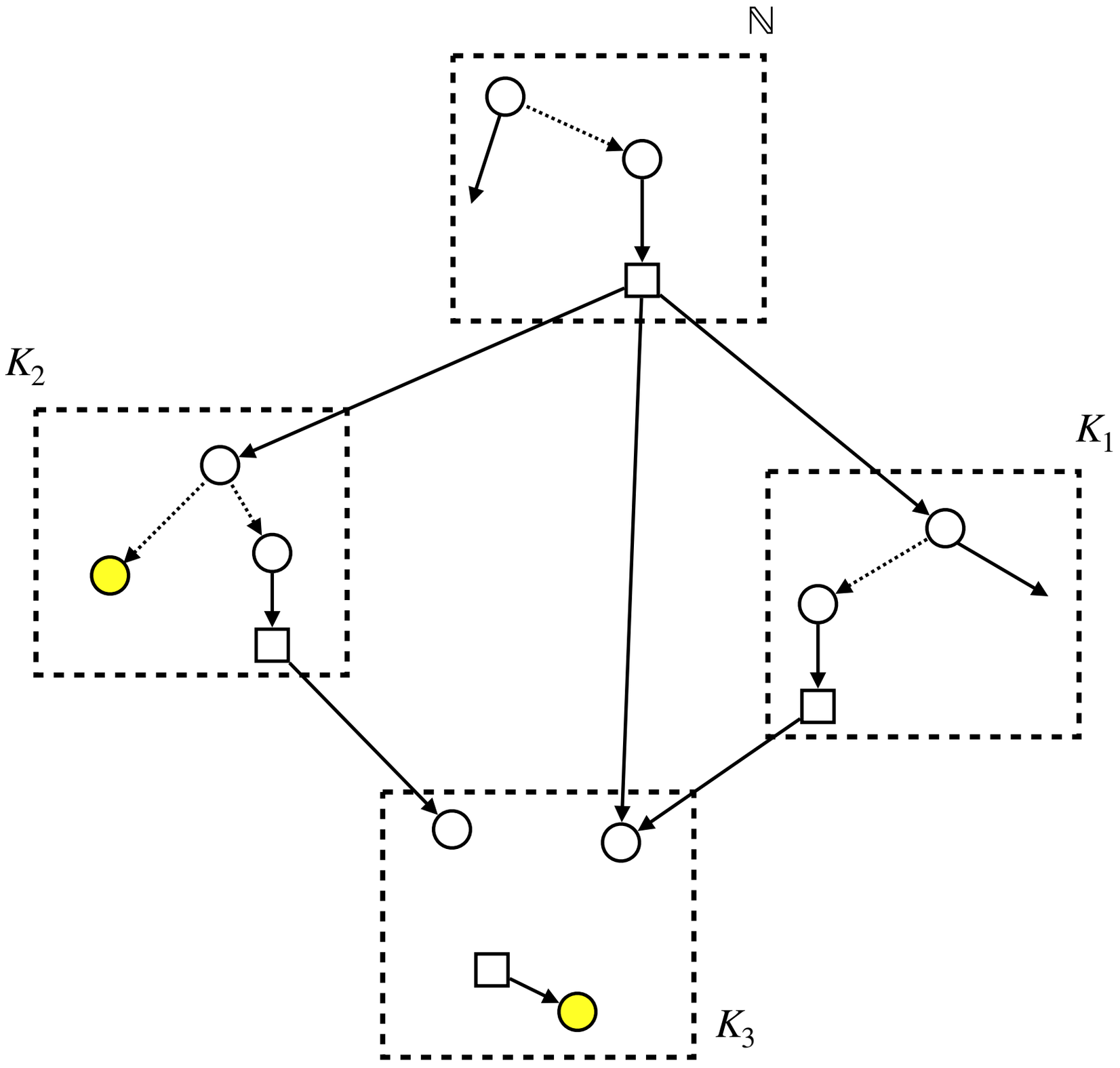
Semilinear Sets - PSPACE upper bound

Step 1. Construct $\mathscr{K}[v, K]$ - <u>restriction</u> of $\mathscr{K}$



- Stop at any $K' \subsetneq K$
- Polynomial size game : solvable in Polynomial time

Semilinear Sets - PSPACE upper bound



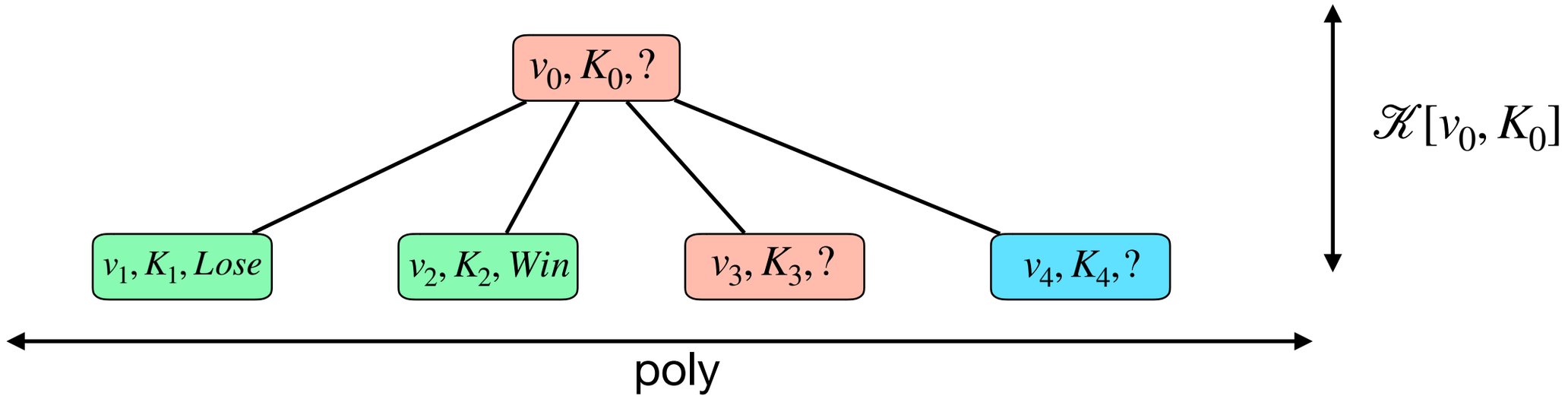
Semilinear Sets - PSPACE upper bound

Step 2. Apply DFS - <u>reuse</u> “space"

$v_0, K_0, ?$

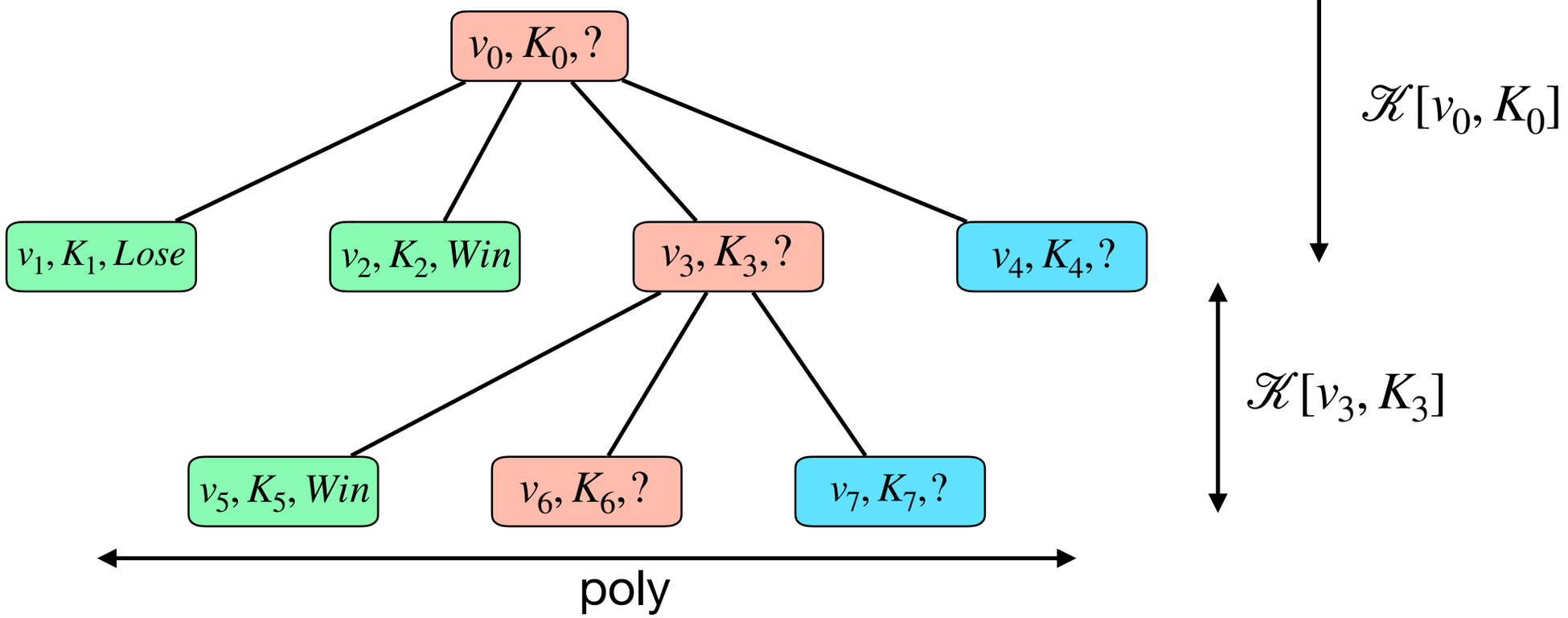
Semilinear Sets - PSPACE upper bound

Step 2. Apply DFS - reuse “space"



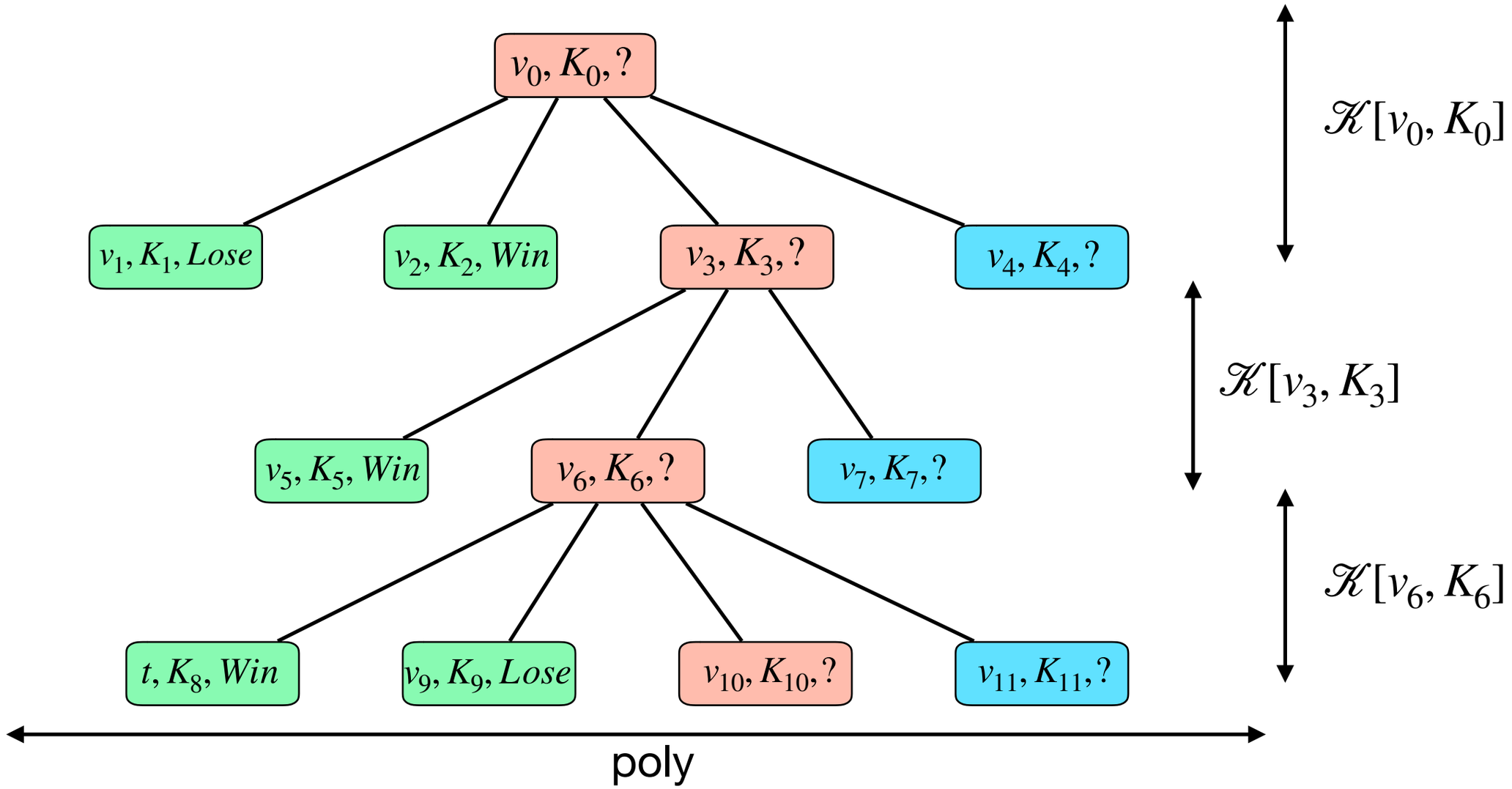
Semilinear Sets - PSPACE upper bound

Step 2. Apply DFS - reuse “space"



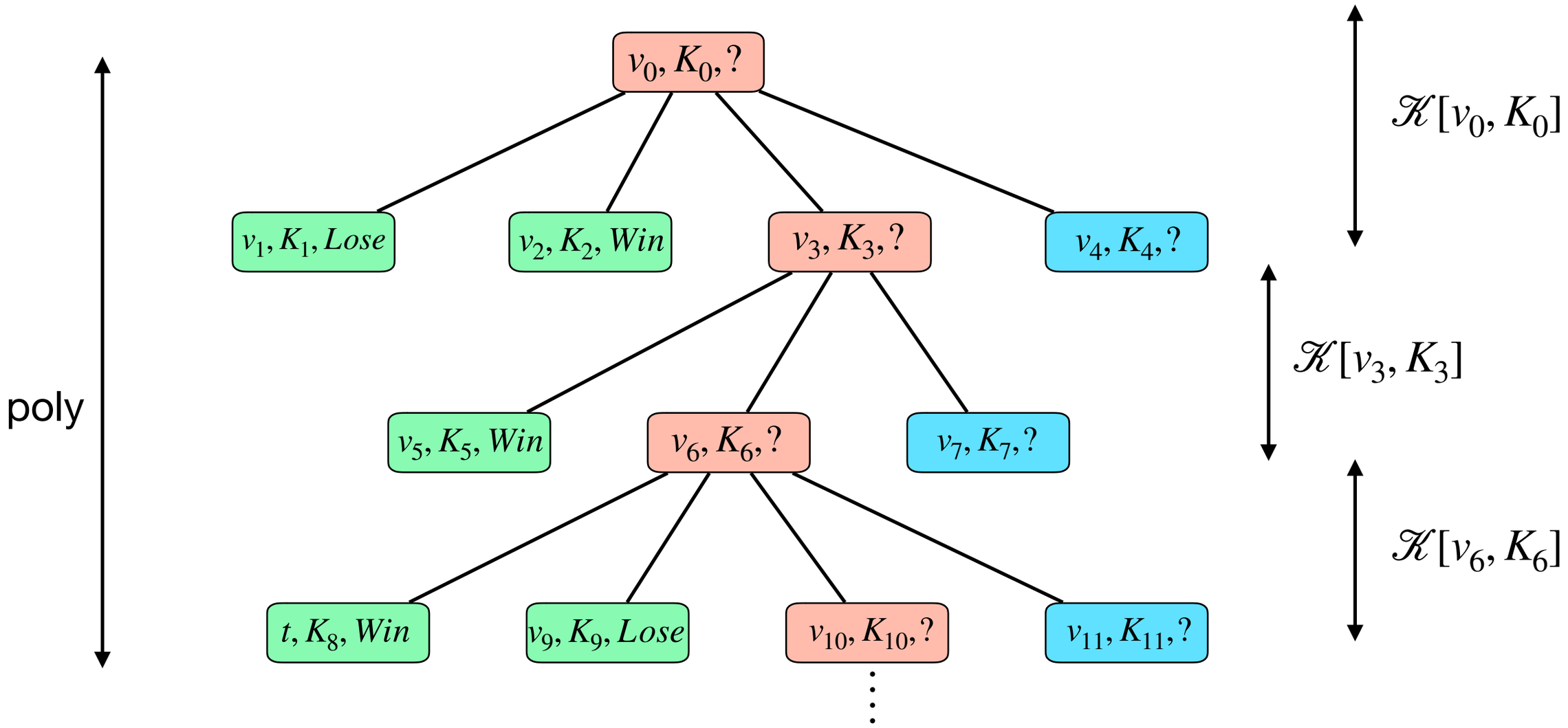
Semilinear Sets - PSPACE upper bound

Step 2. Apply DFS - reuse “space"



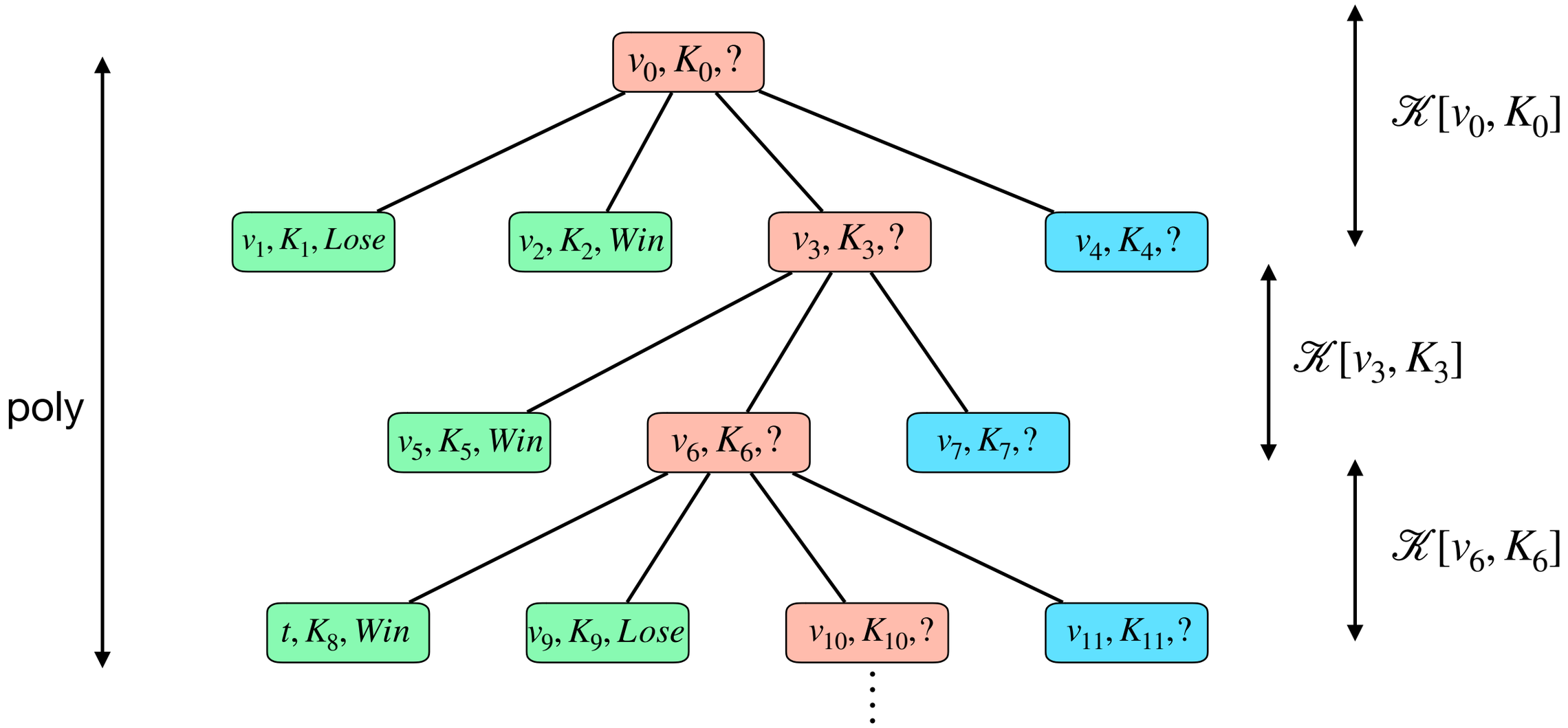
Semilinear Sets - PSPACE upper bound

Step 2. Apply DFS - <u>reuse</u> “space"



Semilinear Sets - PSPACE upper bound

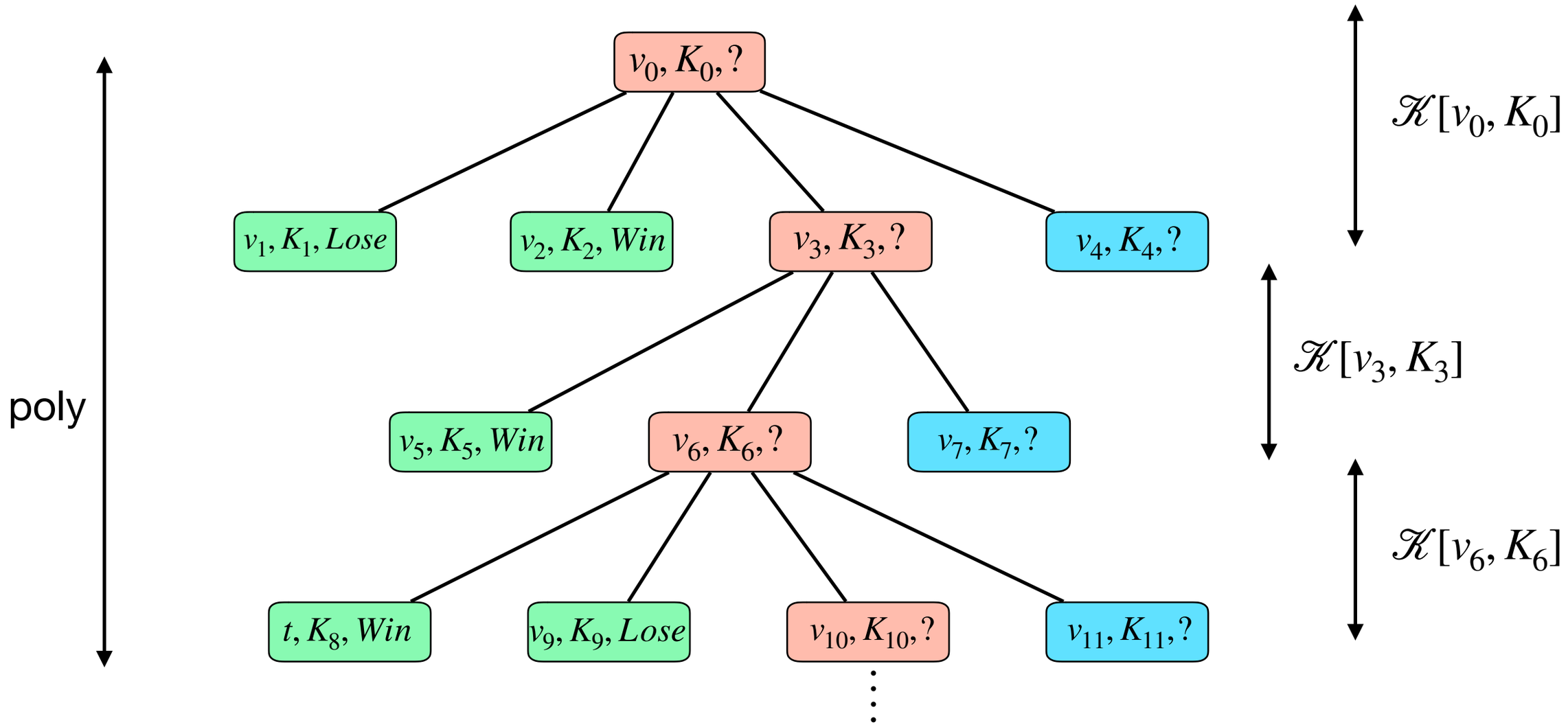
Step 2. Apply DFS - <u>reuse</u> “space"



- tag(*v*,*K*) = Win; if $\begin{cases} \text{either, } v \text{ is target} \\ \text{or, some 'win' is reachable in } \mathscr{K}[v, K] \end{cases}$

Semilinear Sets - PSPACE upper bound

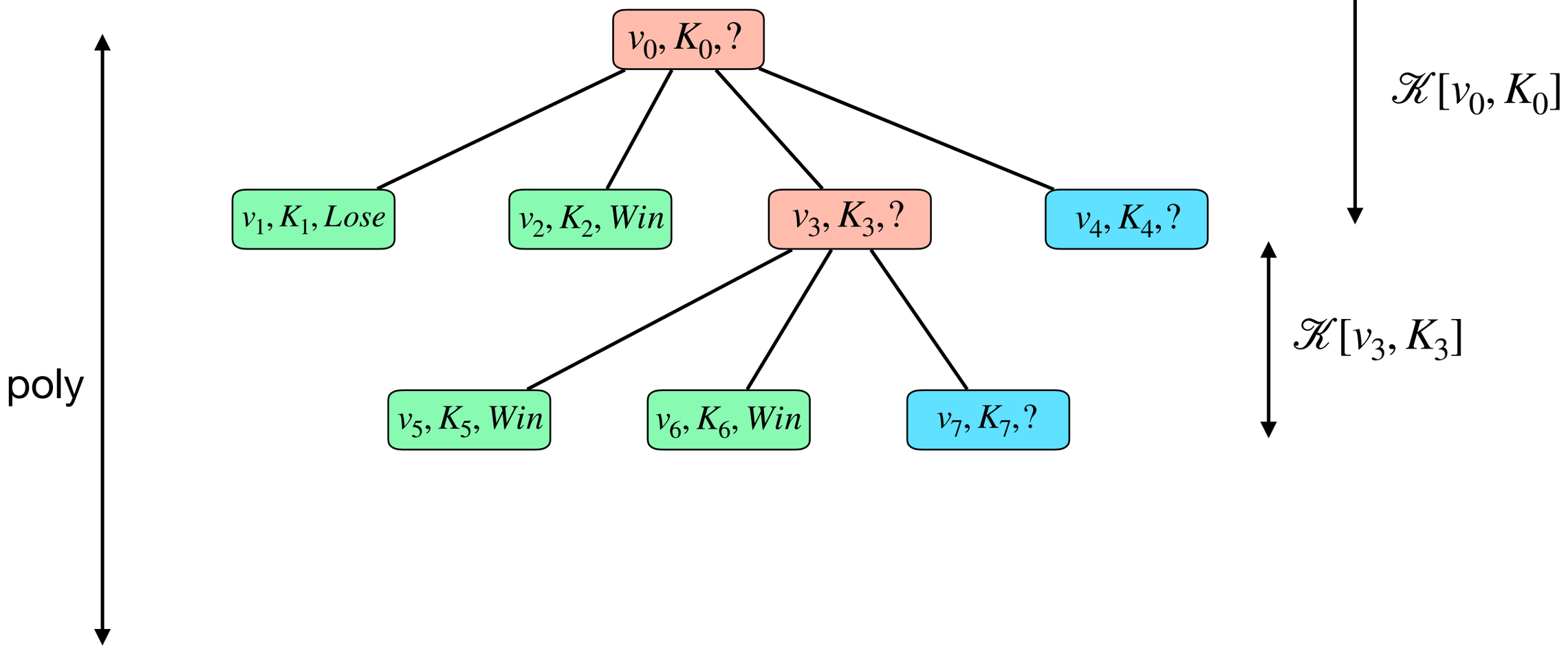
Step 2. Apply DFS - <u>reuse</u> “space"



- tag(v,K) = Win; if $\begin{cases} \text{either, } v \text{ is target} \\ \text{or, some 'win' is reachable in } \mathscr{K}[v, K] \end{cases}$
- tag once computed, the subtree is "forgotten"

Semilinear Sets - PSPACE upper bound

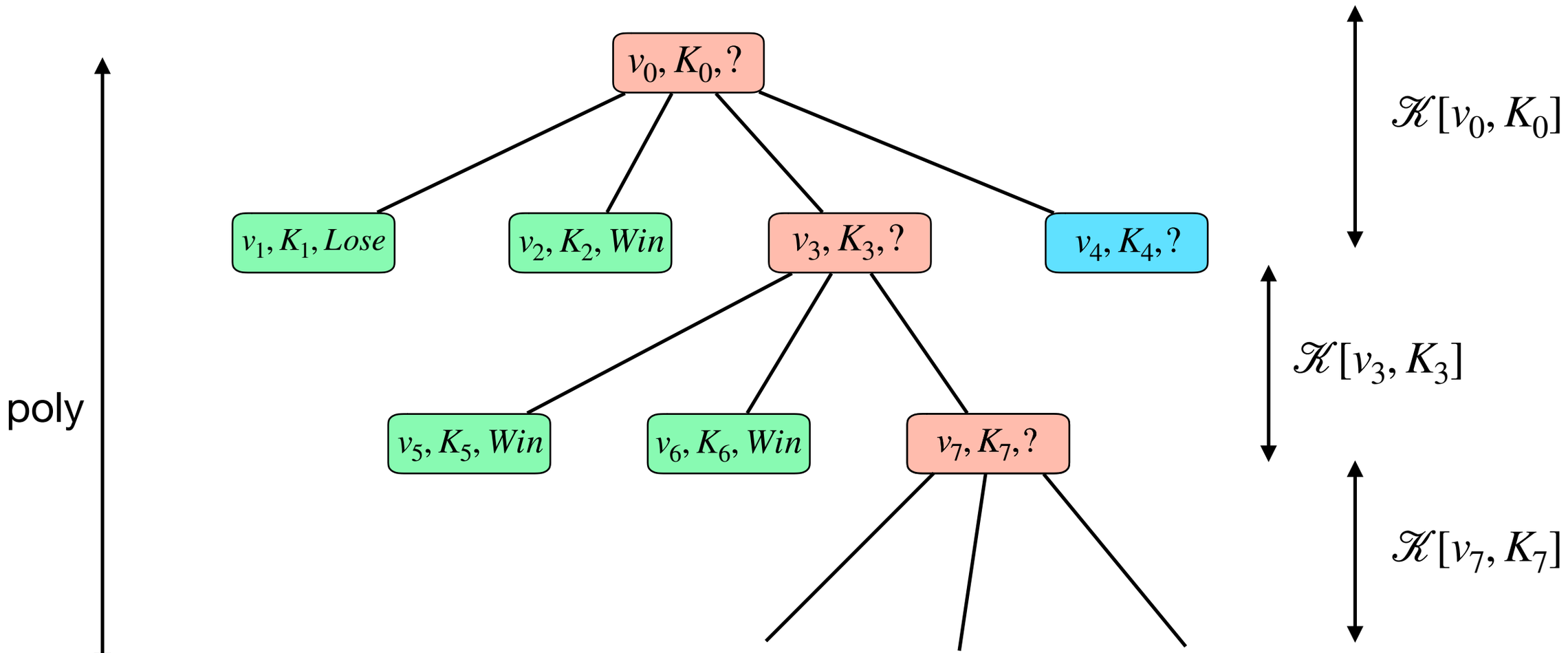
Step 2. Apply DFS - <u>reuse</u> “space"



- tag(v,K) = Win; if $\begin{cases} \text{either, } v \text{ is target} \\ \text{or, some 'win' is reachable in } \mathscr{K}[v, K] \end{cases}$
- tag once computed, the subtree is "forgotten"

Semilinear Sets - PSPACE upper bound

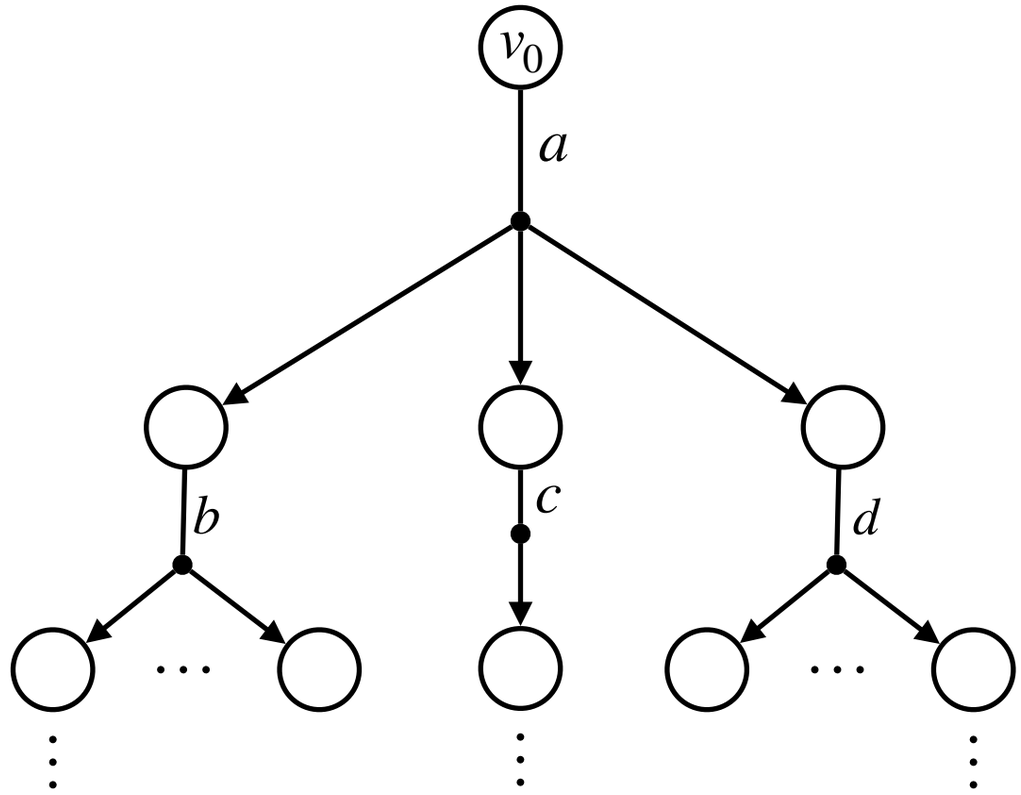
Step 2. Apply DFS - <u>reuse</u> “space"



- tag(v,K) = Win; if $\begin{cases} \text{either, } v \text{ is target} \\ \text{or, some 'win' is reachable in } \mathscr{K}[v, K] \end{cases}$
- tag once computed, the subtree is "forgotten"

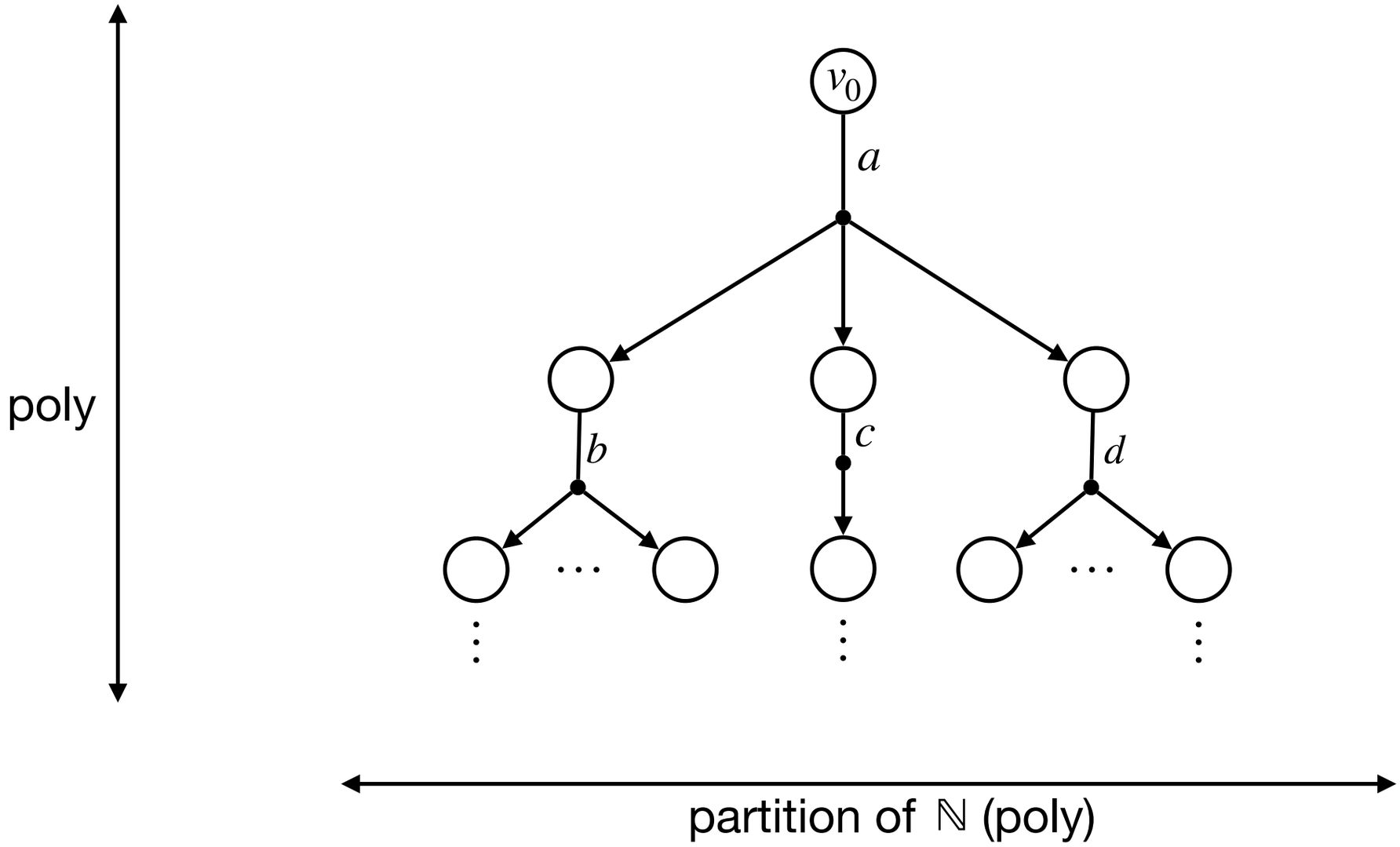
Unions of intervals, Deterministic - NP upper bound

Unions of intervals, Deterministic - NP upper bound



- Non-deterministically guess a strategy

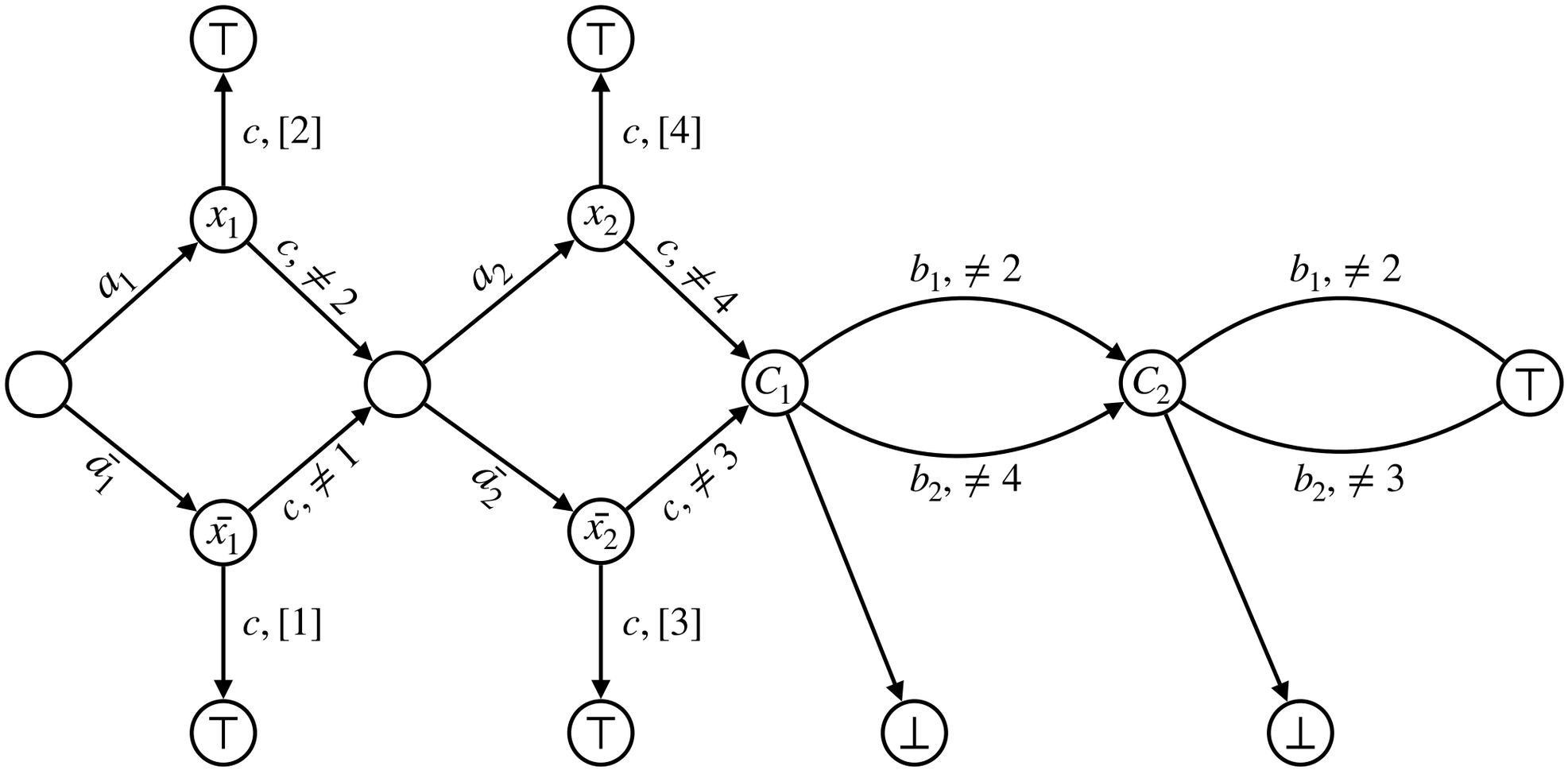
Unions of intervals, Deterministic - NP upper bound



- Non-deterministically guess a strategy
- Size polynomial (in #endpoints)

Unions of intervals, Deterministic - NP-hardness

- Reduce from SAT

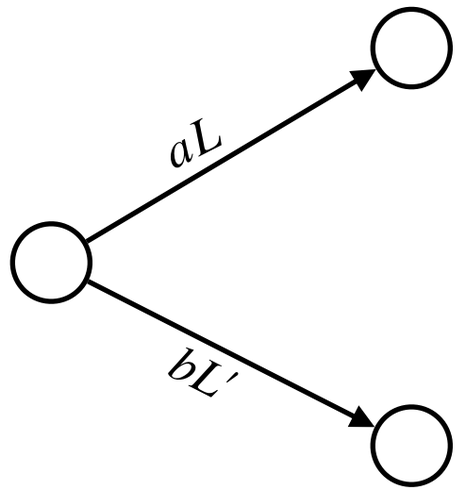


- Eg: $\varphi = (x_1 \vee x_2) \wedge (x_1 \vee \neg x_2)$

- Similar proof for NP-hardness for deterministic arenas

Conclusion

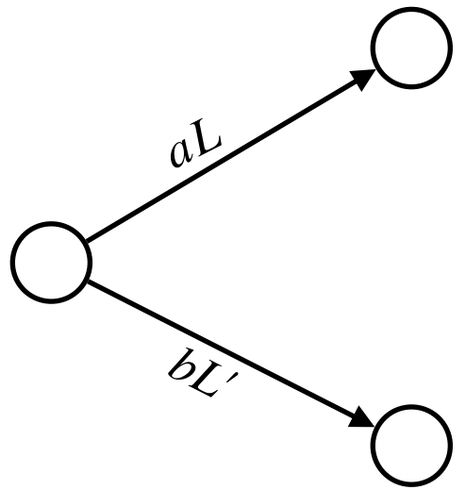
Parameterized Concurrent Games



- Generalisation of 2-player concurrent games
- P_1 against the world
- Strategies need memory
- Knowledge game construction
- PSPACE-completeness in general case
- Better bounds for simpler cases

Conclusion

Parameterized Concurrent Games



- Generalisation of 2-player concurrent games
- P_1 against the world
- Strategies need memory
- Knowledge game construction
- PSPACE-completeness in general case
- Better bounds for simpler cases

Thank You