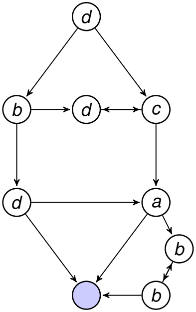
Computing the winning coalitions of a turn-based game with reachability objective for any position

Loïs Vanhée[1] Sophie Pinchinat[2]

[1]ENS Cachan Bretagne, IRISA

[2]IRISA, S4

Formal Setting

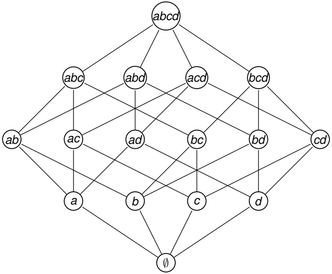


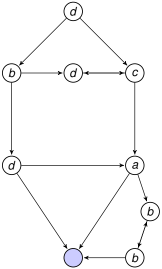
- Turn based game with reachability objective
- $G = (Pos, Agt, Trans, Goal)$
  - Pos: set of positions
  - Agt: set of agents $\{a, b, c, d\}$
  - $Trans : Pos \times Act \to 2^{Pos}$: transition function
  - $Goal \subseteq Pos$: goal positions
  - $owner : Pos \to Agt$: owner of the position
- Coalition $C \subseteq Agt$
- *C is winning from p*: C can enforce reachability of $Goal$ from p

Objectives

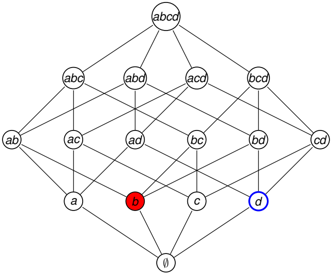
- Given G, link positions and winning coalitions
- Key questions:
  - Given $\mathbf{C} \subseteq 2^{Agt}$, what positions are winning for any $C \in \mathbf{C}$?
    - Coalition-based approach:
      $CBA : 2^{Agt} \to 2^{Pos}$, $CBA(C) = \{p | p \text{ is winning for } C\}$
  - Given $\mathbf{P} \subseteq Pos$, what coalitions are winning from any $p \in \mathbf{P}$?
    - Position-based approach:
      $PBA : Pos \to 2^{2^{Agt}}$, $PBA(p) = \{C | C \text{ is winning from } p\}$
    - Extra feature: get the minimal winning coalitions from p
- Both: compute the set $Win : 2^{Pos \times 2^{Agt}}$
  $Win = \{(p, C) | C \text{ is winning from } p\}$

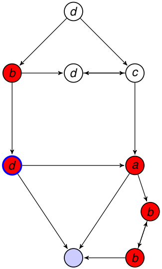
Dual Approaches





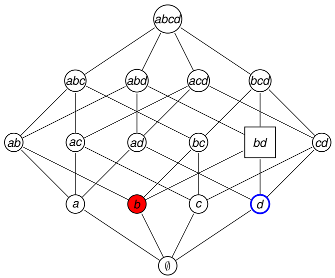
Coalition-Based Approach

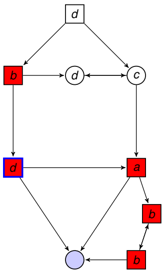




- Compute the winning positions for every coalition C
- $CBA(C) = attr_C$ ($attr$ is the attractor from [GTW02])

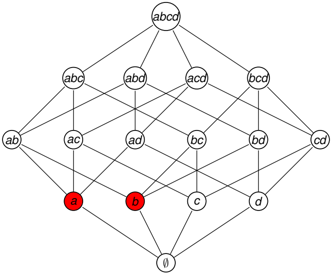
Coalition-Based Approach

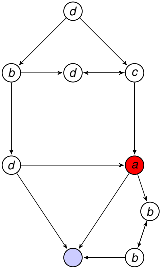




- Coalition-based winning monotonicity (not from scratch): $CBA(C) \cup CBA(C') \subseteq CBA(C \cup C')$

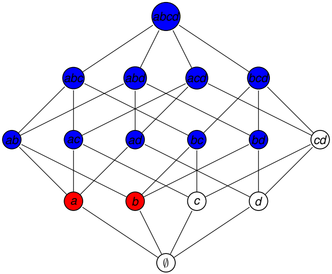
Position-Based Approach

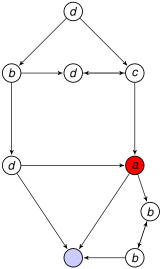




- Compute the winning coalitions for every positions: $PBA(p)$

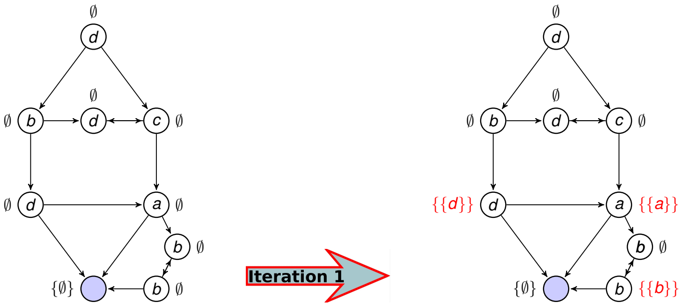
Position-Based Approach





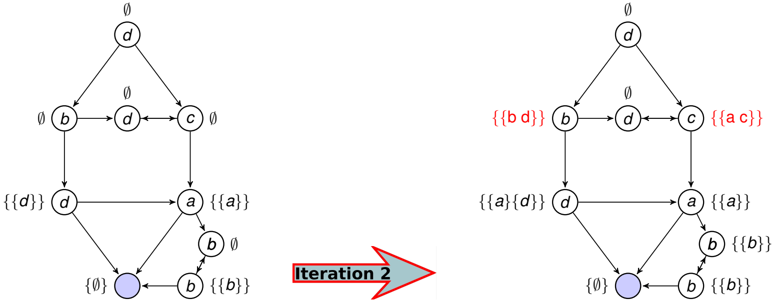
- Position-based winning monotonicity (only upward closure): $C \in PBA(p) \Rightarrow (\forall C' s.t. C \subseteq C' \Rightarrow C' \in PBA(p))$

The Position-Based Algorithm



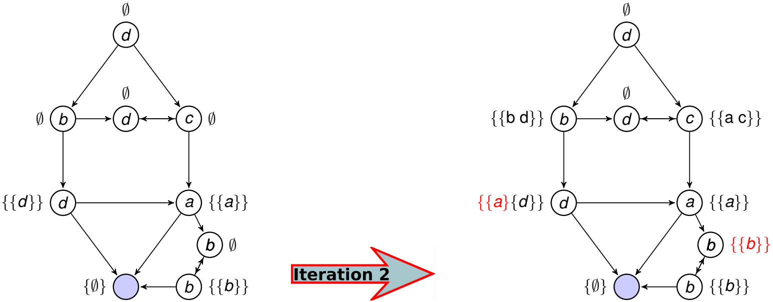
- Iterate until stabilization:
  - Buy the owner: $\{C | owner(p) \in C \wedge \exists p' \in succ(p), C \in Win(p'), \}$
  - Buy every successor: $\{C | \forall p' \in succ(p), C \in Win(p')\}$

The Position-Based Algorithm



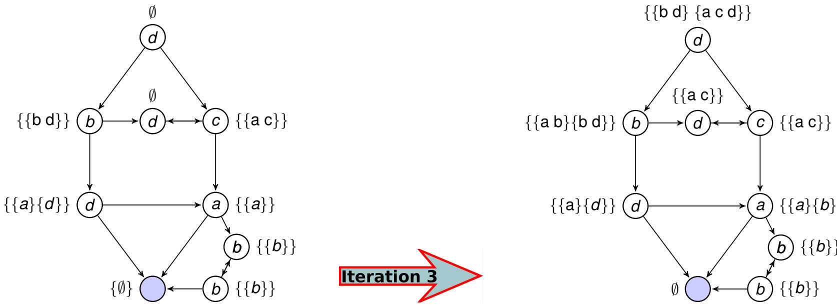
- Iterate until stabilization:
  - Buy the owner: $\{C | owner(p) \in C \wedge \exists p' \in succ(p), C \in Win(p'), \}$
  - Buy every successor: $\{C | \forall p' \in succ(p), C \in Win(p')\}$

The Position-Based Algorithm



- Iterate until stabilization:
  - Buy the owner: $\{C | owner(p) \in C \land \exists p' \in succ(p), C \in Win(p'), \}$
  - Buy every successor: $\{C | \forall p' \in succ(p), C \in Win(p')\}$

The Position-Based Algorithm

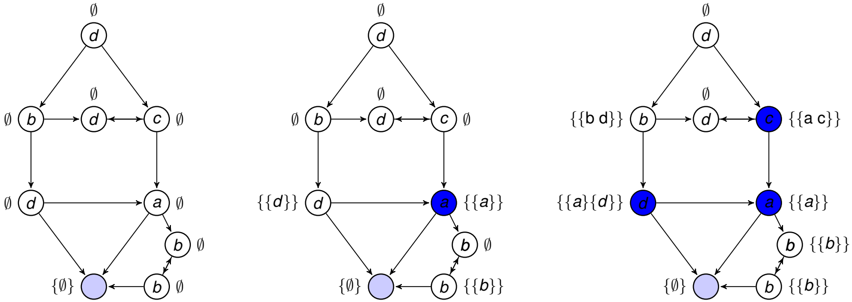


- Iterate until stabilization:
  - Buy the owner: $\{C | owner(p) \in C \wedge \exists p' \in succ(p), C \in Win(p'), \}$
  - Buy every successor: $\{C | \forall p' \in succ(p), C \in Win(p')\}$

Theoretical Results

- Complexity: $O(d * |Trans|(log(|Pos|) + 2^{|Agt|}))$
- Buying every successor is the exponential part: $|Win(p)|$ can increase up to $\Pi_{p' \in succ(p)} |Win(p')|$
- Position-based and coalition-based winning monotonicity are equivalent
- *Duality theorem:*
  - Let $CBA^i(C)$ be the i^{th} attractor on C
  - Let $PBA^i(p)$ be the i^{th} iteration of the position-based algorithm on p
  - $p \in attr^i_C \Leftrightarrow C \in PBA^i(p)$

Example on $\{a, c\}$



Implementation: CBA vs PBA

- CBA:
  - Caching with C-monotonicity
  - With luck the game is (symbollically) explored only once!
  - But coalition enumeration unavoidable
- PBA:
  - Efficient representation with P-monotonicity
  - With luck coalition enumeration is avoided
  - But avoiding position enumeration is harder

Implementation: Symbolic Representation

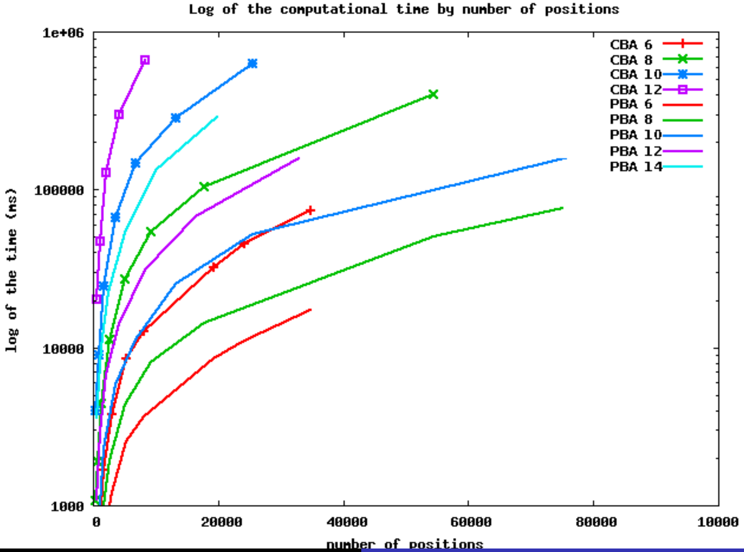
- Associate one variable v_i for every agent a_i
- A coalition: $C \rightarrow (\wedge_{a_i \in C} v_i) \wedge (\wedge_{a_i \notin C} \bar{v}_i)$
- Upward closure from C: $\{C' | C \subseteq C'\} \rightarrow \wedge_{a_i \in C} v_i$

Implementation: Algorithm

- For every position p
  - Buy the owner:
    - $\{C | owner(p) \in C \wedge \exists (p', C) \in Win, p' \in succ(p)\}$
      $F_{buy} = F_p \wedge F_{owner(p)} \wedge (\exists \vec{v_{p_i}}, F_{succ(p)} \wedge F_{Win})$
  - Buy every successor:
    - $\{C | \forall p' \in succ(p), C \in Win(p')\}$
      $F_{\overline{buy}} = F_p \wedge \bigwedge_{p' \in succ(p)} (\exists \vec{v_{p_i}}, F_{p'} \wedge F_{Win})$
  - $F_{Win} \leftarrow F_{Win} \vee F_{buy} \vee F_{\overline{buy}}$
- BDD ordering by position then by (minimal) coalition

Implementation: Querying

- Once *Win* is computed
  - Is C winning from p?
    - Test a valuation: $O(log(|Pos|) * |Agt|)$
  - *Querying:* Check if any C from $\mathbf{C} \subseteq 2^{Agt}$ is winning from any p in $\mathbf{P} \subseteq Pos$:
    - PBA: $O(|\mathbf{P}|(log(|Pos|) + |\mathbf{C}| * |Agt|)$
    - CBA: $O(|\mathbf{C}|(|Agt| + |\mathbf{P}| * log(|Pos|))$
- Upper bounds, BDDs may *hightly* cut down this complexity
- Lazy CBA
- *PBA* contains the set of minimal winning coalitions!
  NP-complete problem [BGMR08]


Log of the computational time by number of positions
1e+06
100000
10000
1000
log of the time (ms)
0
20000
40000
60000
80000
10000
number of positions
CBA 6
CBA 8
CBA 10
CBA 12
PBA 6
PBA 8
PBA 10
PBA 12
PBA 14

Conclusion: Compute $Win \subseteq 2^{Pos \times 2^{Agt}}$ for a turn-based game with a reachability objective

- Two dual approaches: PBA and CBA
  - PBA empirically faster on $|Agt|$ than CBA
  - PBA also computes the minimal winning coalitions
- PBA vs CBA: dependant on the queries

Thomas Brihaye, Mohamed Ghannem, Nicolas Markey, and Lionel Rieg.
Good friends are hard to find!
In *TIME '08: Proceedings of the 2008 15th International Symposium on Temporal Representation and Reasoning*, pages 32–40, Washington, DC, USA, 2008. IEEE Computer Society.

Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors.
Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001], volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.