Backward Model Checking of Uniform Strategies

A Backward-traversal-based Approach for Symbolic Model Checking of Uniform Strategies for Constrained Reachability

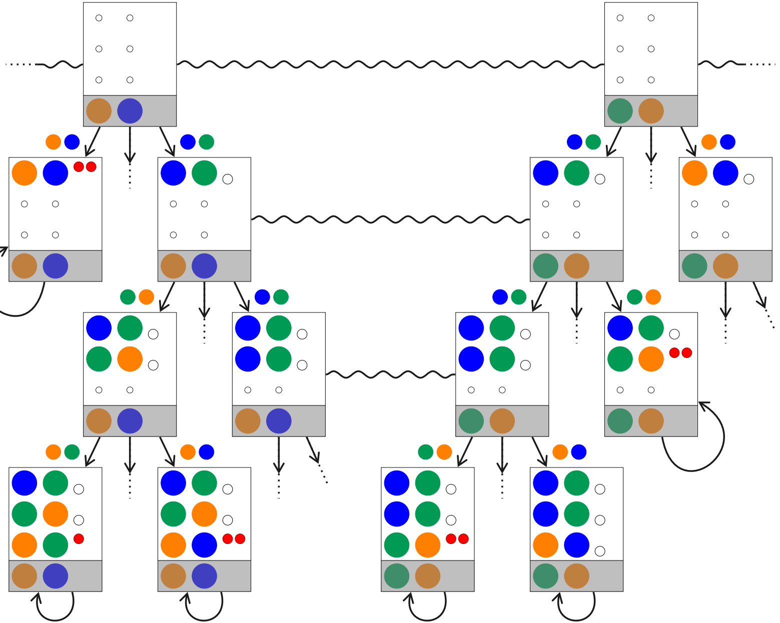
Simon Busard UCLouvain, Belgium
Charles Pecheur UCLouvain, Belgium

8th International Symposium on
Games, Automata, Logics, and Formal Verification.
20–22 September 2017, Rome, Italy

Running example: Mastermind

Simplified Mastermind: 3 colors, 3 turns, 2 pegs (different colors)

Simplified Mastermind FSM (1122 states)



Mastermind

Is there a **strategy**
for the guesser to find the solution?

A **uniform** strategy!

ATL_{ir}, ATL with imperfect information and recall

$ATL_{ir} =$ logic for **uniform** strategies and CTL-**like objectives** in concurrent multi-agent systems

"the guesser has a strategy to eventually win the game"

$\langle\langle guesser \rangle\rangle \mathbf{F}\ win$

"the guesser has a strategy to never play a blue peg until winning"

$\langle\langle guesser \rangle\rangle [\neg blue\ peg\ \mathbf{U}\ win]$

"the guesser has a strategy to never lose"

$\langle\langle guesser \rangle\rangle \mathbf{G}\ \neg lose$

⚠ Model checking ATL formulas is **easy** (PTIME), but model checking ATL_{ir} formulas is Δ_P^2-**complete**!

Existing approaches for ATL_{ir} model checking

The **partial** approach:
S. Busard, C. Pecheur, H. Qu, F. Raimondi (2014)
Improving the Model Checking of Strategies under Partial Observability and Fairness Constraints

The **early** approach:
J. Pilecki, M.A. Bednarczyk, W. Jamroga (2014)
Synthesis and Verification of Uniform Strategies for Multi-agent Systems

The **symbolic** approach:
X. Huang, R. van der Meyden (2014)
Symbolic Model Checking Epistemic Strategy Logic

The partial approach

1. generate all **partial strategies** from states that matter (through a **forward** traversal)

2. check each strategy against the objective

+ early termination

The early approach

- We do not need to get a completely determined adequate partial strategy before checking it
- We can stop if all extensions of the current strategy are winning
- We can stop if **no** general extension is winning

$\Rightarrow$ alternate between extending a partial strategy and checking whether all or no extensions are winning

+ early termination

The symbolic approach

1. Encode the uniform strategies in the **states** of a derived model
2. Perform fixpoint computations on the derived model

$\Rightarrow$ Compute **all** winning strategies at the same time (symbolically)

$\Rightarrow$ explosion of the number of derived states

Pre-filtering surely losing moves

1. It is **easy** to compute the moves belonging to a winning **general** strategy (PTIME)

2. If some move does not belong to a winning general strategy, it does not belong to a winning **uniform** one

$\Rightarrow$ We can **remove the losing moves** before enumerating or encoding the uniform strategies

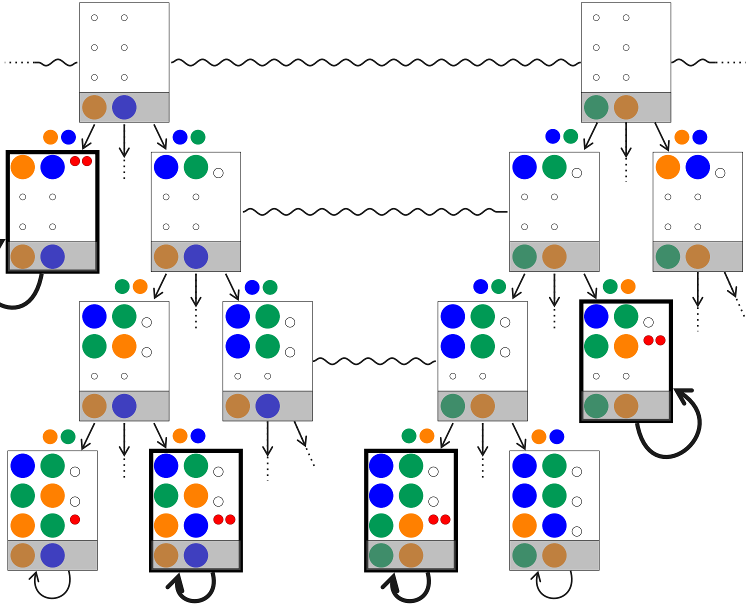
Objective: a new approach

Partial and early approaches:
forward traversal to generate strategies
+ **backward** traversal to check strategies

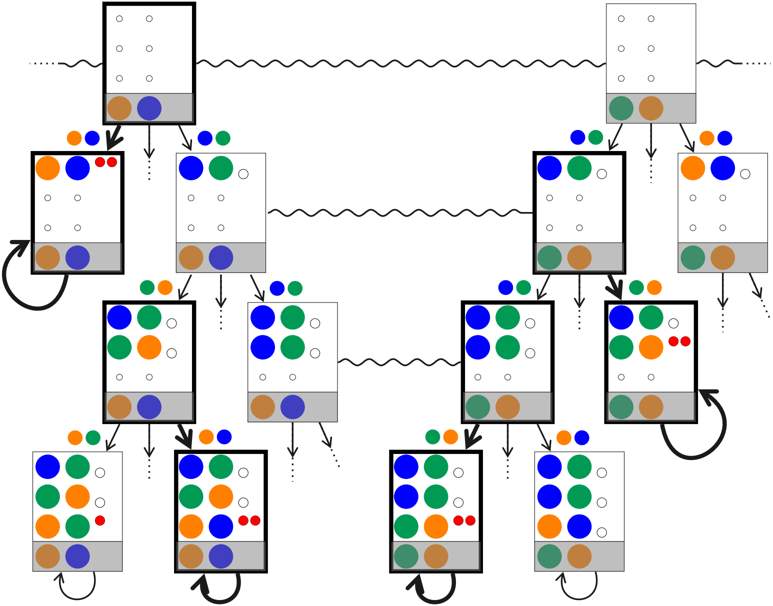
⇒ Design an approach working with a **backward** traversal only
⇒ The backward approach

Outline

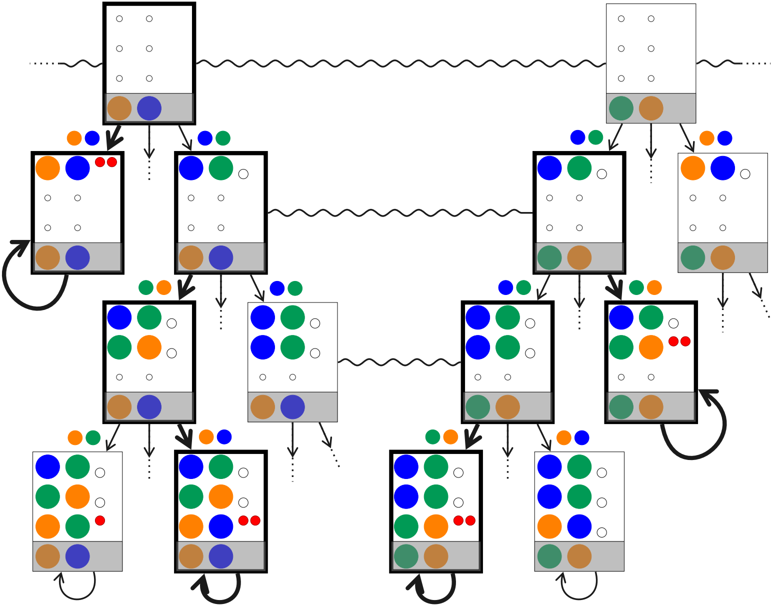
The backward approach



The backward approach



The backward approach



The backward approach

For $\langle\!\langle \Gamma \rangle\!\rangle[P_1 \mathbf{U} P_2]$,

1. start with the moves in states satisfying P_2
2. split them into non-conflicting subsets M_Γ
3. then iterate:
    - compute states for which the current strategy M_Γ is surely winning or surely losing
    - get the **compatible** moves in states satisfying P_1 and reaching states of M_Γ
    - split the newly discovered moves into **non-conflicting subsets** and **extend** M_Γ with them

The backward approach

For $\langle\langle\Gamma\rangle\rangle[P_1 \; \mathbf{U} \; P_2]$,

1. start with the moves in states satisfying P_2
2. split them into non-conflicting subsets M_Γ
3. then iterate:
    - compute states for which the current strategy M_Γ is surely winning or surely losing
    - get the **compatible** moves in states satisfying P_1 and reaching states of M_Γ
    - split the newly discovered moves into **non-conflicting subsets** and **extend** M_Γ with them

Limitations

- Limited to **constrained reachability** objectives:
  $\langle\!\langle\Gamma\rangle\!\rangle\mathbf{X}$, $\langle\!\langle\Gamma\rangle\!\rangle\mathbf{F}$, $\langle\!\langle\Gamma\rangle\!\rangle\mathbf{U}$ operators (and their $[\![\Gamma]\!]$ dual),
  Cannot handle $\langle\!\langle\Gamma\rangle\!\rangle\mathbf{G}$ and $\langle\!\langle\Gamma\rangle\!\rangle\mathbf{W}$ operators

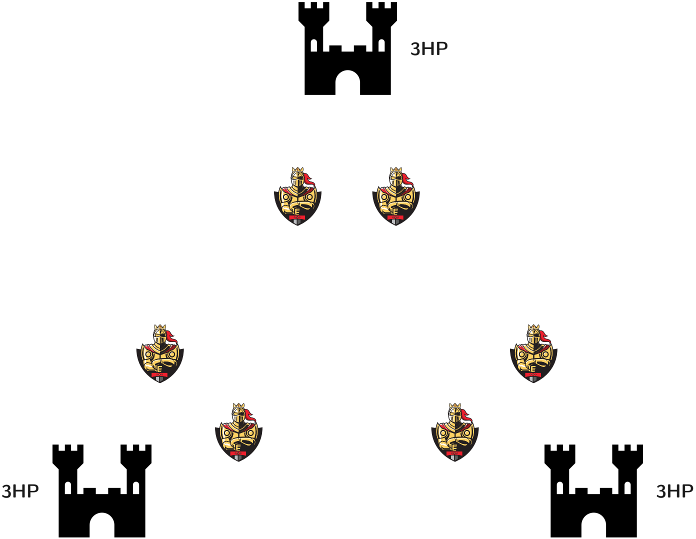
  But can be **mixed** with other approaches to handle other operators

- Split new moves into non-conflicting non-maximal subsets
  $\Rightarrow$ doubly exponential!

  But experiments show competitive results

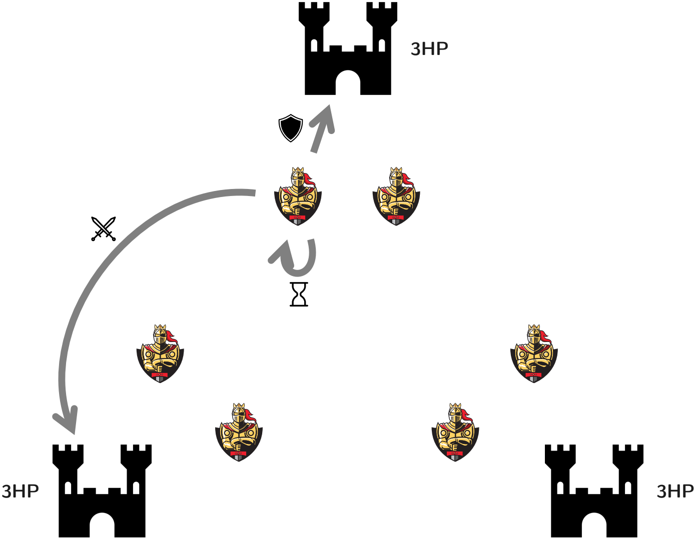
Outline

The three-castle model [1]



[1] J. Pilecki, M.A. Bednarczyk, W. Jamroga (2014).
Synthesis and Verification of Uniform Strategies for Multi-agent Systems

The three-castle model [1]   (±80K strategies per soldier)



[1] J. Pilecki, M.A. Bednarczyk, W. Jamroga (2014).
Synthesis and Verification of Uniform Strategies for Multi-agent Systems

Tested formulas

"the two first castles can defeat the third one"

$$\langle\!\langle Castle_1, Castle_2 \rangle\!\rangle \mathbf{F}\ Castle_3\ defeated$$

Satisfied by all tested instances

"Soldier 1 and Soldier 2 can defeat all castles"

$$\langle\!\langle Soldier_1, Soldier_2 \rangle\!\rangle \mathbf{F}\ all\ defeated$$

Violated by all tested instances

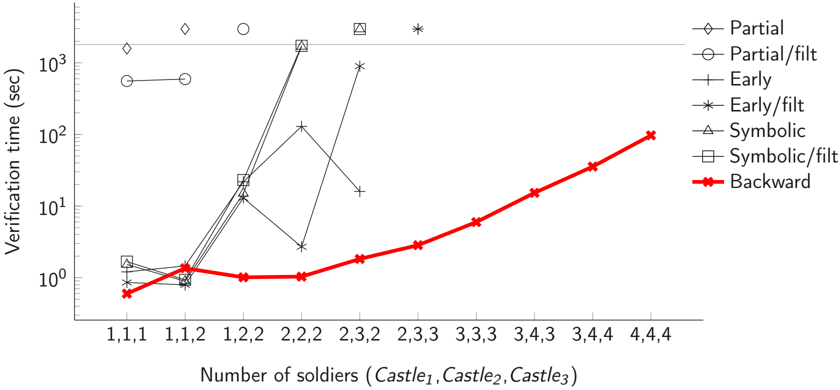
Test setting

- Approaches implemented in the same BDD-based framework (with PyNuSMV, a Python framework based on NuSMV)
- Run on instances of increasing size
- Each test run with a 1800 seconds time limit

Results

"the two first castles can defeat the third one"

$\langle\langle Castle_1, Castle_2 \rangle\rangle \mathbf{F}\ Castle_3\ defeated$



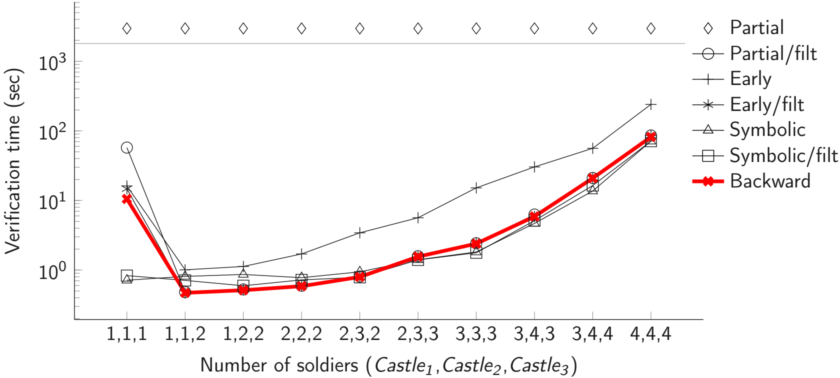
Analysis

- partial approaches have difficulties finding a winning strategy
  pre-filtering helps
- symbolic approaches are better
  pre-filtering does nothing
- early approaches are better
  with irregularities
- backward approach works well
  focus on reaching the target states

Results

"Soldier 1 and Soldier 2 can defeat all castles"

$\langle\langle Soldier_1, Soldier_2 \rangle\rangle \mathbf{F}\ all\ defeated$



Analysis

- partial approach has too many strategies to check
- pre-filtering solves the problem directly
- early approach concludes a bit slower
- backward approach analyses one strategy before concluding

Results summary

The backward approach is similar or better than the other approaches **on the tested model**

because it is **goal-driven** and can **rule losing strategies out**

Outline

Conclusion

The backward approach for finding uniform winning strategies:

- build uniform strategies from the target states
- check that extensions are already winning or surely losing

⚠ Limited to constrained reachability objectives ($\langle\langle\Gamma\rangle\rangle$**U** operator)

Compared with existing approaches **on one model**:

- works better than the other approaches in the first case,
- works similarly to the other approaches in the second

Future work

- Experiment on other case studies
- Experiment with mixed approaches

Thank you!

Questions?