Strategy Repair in Reachability Games.

Nicolas Lecomte
(UMons)

Main reference: [GPP23] Pierre Gaillard, Fabio Patrizi, and Giuseppe Perelli, Strategy Repair in Reachability Games., in proc. of European Conference on Artificial Intelligence (ECAI), 2023.

Table of contents

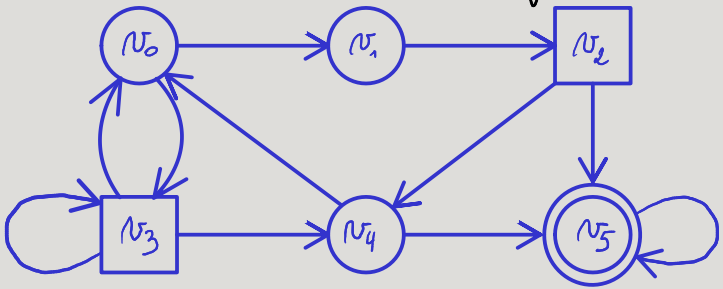
2-player reachability game

A (2-player) arena is a tuple $\mathcal{A} = (V, V_1, V_2, E)$ where

- V is a finite set of vertices
- V_1 and V_2 are sets of vertices controlled by player 1 ($\mathcal{P}_1$) and 2 ($\mathcal{P}_2$) resp. such that $\{V_1, V_2\}$ is a partition of V.
- $E \subseteq V \times V$ is a set of edges.

A 2-player reachability game is a couple $\mathcal{G} = (\mathcal{A}, T)$ where $\mathcal{A}$ is an arena and $T \subseteq V$ is a subset of vertices, called target states.

Example:



2-player reachability game : terminology and notations

- A path is a sequence of vertices $\pi = v_0 v_1 \cdots \in V^\omega$ s.t. $\forall i \in \mathbb{N}, (v_i, v_{i+1}) \in E$.
  ($\pi[0] = v_0$, $\pi[1] = v_1$)
- π is winning for player 1 if $\pi[i] \in G$ for some $i \in \mathbb{N}$. It is winning for player 0, otherwise.
- A (memoryless) strategy for player i is a function $\sigma_i : V_i \to E$ s.t. $\forall v \in V_i, (v, \sigma_i(v)) \in E$.
- A path π is compatible with strategy σ_i if $\forall k \in \mathbb{N}, \pi[k] \in V_i \Rightarrow \sigma(\pi[k]) = \pi[k+1]$.
  ↳ $\mathrm{out}(v, \sigma_1)$ denotes the set of paths that are compatible with σ_1
  and $\mathrm{play}(v, \sigma_1, \sigma_2)$ is the only path starting from v that is compatible with σ_1 and σ_2

- A strategy σ_i is winning for $\mathcal{P}_i$ from v if every path in $\mathrm{out}(v, \sigma_i)$ is winning for $\mathcal{P}_i$
- A vertex v is winning for $\mathcal{P}_i$ if there exists a strategy σ_i that is winning for $\mathcal{P}_i$.
- The set of vertices that are winning for $\mathcal{P}_i$ is denoted by $\mathrm{Win}_i(\mathcal{G})$, and is called the winning region of $\mathcal{P}_i$.
- A strategy σ_i is simply winning if it wins from every vertices in $\mathrm{Win}_i(\mathcal{G})$.

2-player reachability game : Main result + example

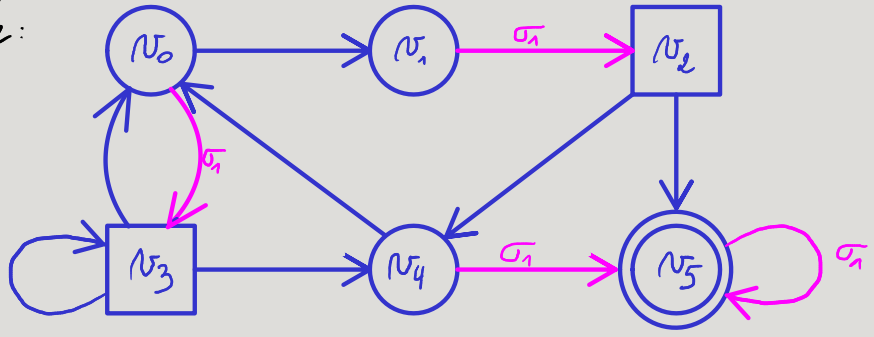
Theorem [Thomas 95]

Reachability games are memoryless determined

ie: every vertex is either winning for $\mathcal{J}_1$ or $\mathcal{J}_2$

and there always exists a memoryless winning strategy

Example :



- $\pi = v_0\, v_3\, v_0 v_3\, v_4\, v_5^*$ is a winning path for $\mathcal{J}_1$
- v_1 is a winning vertex for $\mathcal{J}_1$ with the stategy σ_1.
- $v_3 \in Win_2(\mathcal{G})$

[Thomas 95] Wolfgang Thomas, On the Synthesis of Strategies in Infinite Games, STACS, 1995.

How to compute $Win_1(\mathcal{G})$?

Theorem [Thomas 95].

$Win_1(\mathcal{G})$ is the fixpoint of the following operator:

$$Acces(X) = X \cup \{v \in V_1 \mid \exists v' \in X, (v,v') \in E\} \cup \{v \in V_2 \mid \forall v' \in X, (v,v') \in E\}$$

where X is initialised with $\mathcal{T}$.

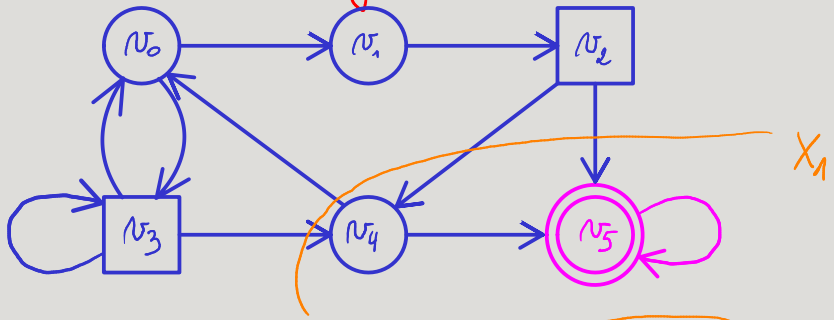
Intuition: At iteration k, the vertices in X are the vertices that can reach the target in at most k steps

⟶ A simply winning strategy can be easily deduced.

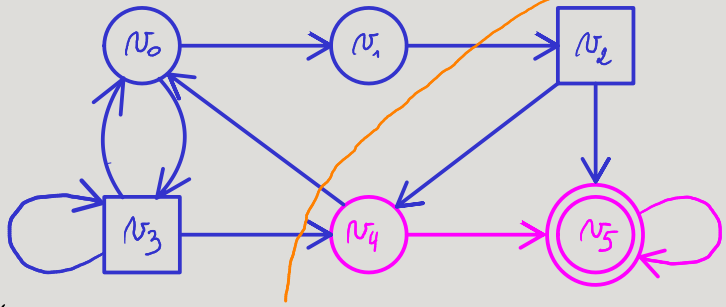
NB: $Win_2(\mathcal{G}) = V \setminus Win_1(\mathcal{G})$

How to compute $Win_1(\mathcal{G})$? An example

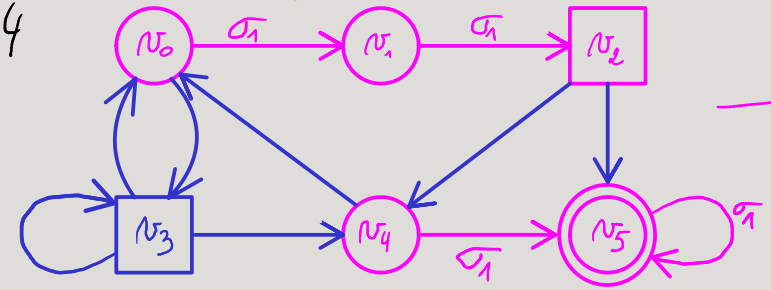
Step 1:



Step 2:



Steps 3-4



$\rightarrow Win_1(\mathcal{G}) = V \setminus \{v_3\}$

σ_1 is a simply winning strategy for P_1.

Strategy repair: Definitions and Problem

Let σ_1, σ_1^* be two $\mathcal{P}_1$ strategies

The distance between σ_1 and σ_1^* is defined as

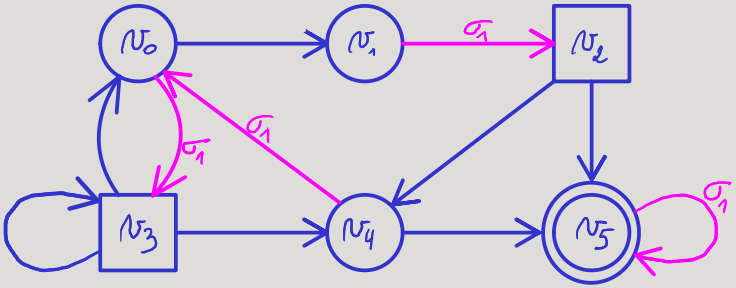
$$d(\sigma_1, \sigma_1^*) := \left| \{ v \in V_1 \mid \sigma_1(v) \neq \sigma_1^*(v) \} \right|$$

Proposition:

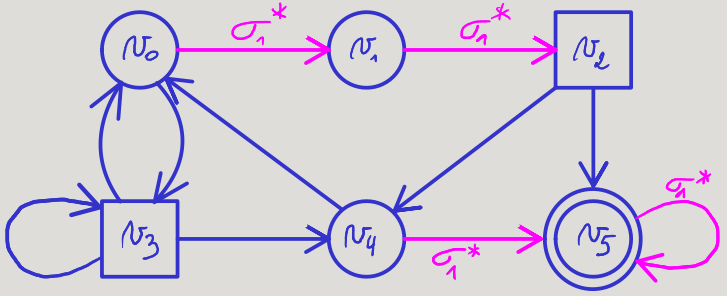
This function is a distance in the space of (memoryless) $\mathcal{P}_1$ strategies.

Example

$$d(\sigma_1, \sigma_1^*) = 2$$



$$Win_1(\mathcal{G}, \sigma_1) = \{v_5\}$$



$$Win_1(\mathcal{G}) = Win(\mathcal{G}, \sigma_1^*) = \{v_0, v_1, v_2, v_4, v_5\}$$

Strategy repair

Def:

For a given reachability game $\mathcal{G}$, a $\mathcal{P}_1$ strategy σ_0 and $k \in \mathbb{N}$, the **strategy repair problem** is to find a simply winning strategy σ_0^* such that $d(\sigma_0, \sigma_0^*) \leq k$

Theorem:

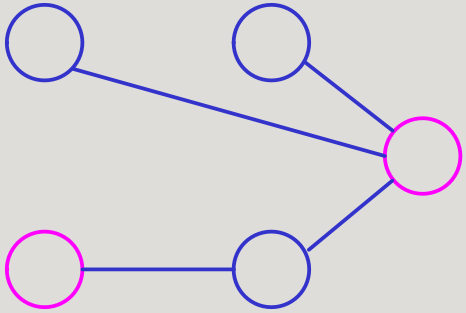
The strategy repair problem is NP-complete.

Vertex cover

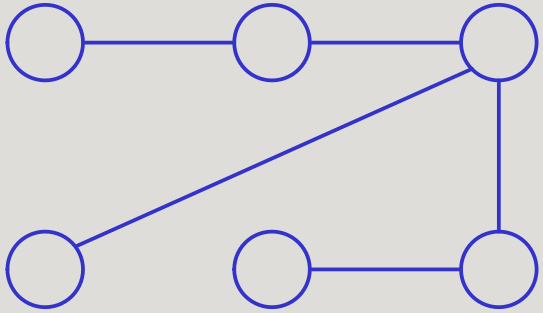
Def:

For a given undirected graph $G = (S, A)$ and $k \in \mathbb{N}$, the vertex cover problem is to find $S' \subseteq S$ such that
- $|S'| \leq k$
- $\forall \{v, v'\} \in A, \quad v \in S' \vee v' \in S'$

Examples:



↳ Yes-instance for $k = 2$



No-instance for $k = 2$.

Proof of the theorem :

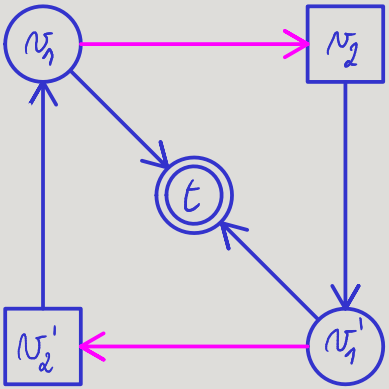
Let $G = (S, A)$ be an undirected graph and $k \in \mathbb{N}$

We construct the following reachability game $\mathcal{G} = (\mathcal{A}, \mathcal{C})$ with $\mathcal{A} = (V, V_0, V_1, E)$ where

- $V = S \times \{1,2\} \cup \{t\}$
- $V_1 = S \times \{1\} \cup \{t\}$
- $V_2 = S \times \{2\}$
- $E = \{ ((v,1),(v,2)) \mid v \in S \} \cup \{ ((v,1), t) \mid v \in S \} \cup \{ ((v,2), (v',1)) \mid \{v, v'\} \in A \}$

We consider the strategy σ_1 such that $\forall v \in S,\ \sigma_1((v,1)) = (v,2)$.

Example for $\{v, v'\} \in A$:



We have that $Win_1(\mathcal{G}) = V$.

Find the minimal changes : an exact algorithm

Def:

Let $X \subseteq V$ be a subset of vertices

$\text{Frontier}_1(X) = \big((V_1 \setminus X) \times X\big) \cap E$ is the set of edges that can be used by $\mathcal{P}_1$ to enter in X.

Input: $\mathcal{G}$ a reachability game, σ_1 a strategy for player 1
Output: Winning strategy for $\mathcal{G}$ minimizing the distance from σ_1

```
Fix(G, σ1) :
1  T' ← Win(G, σ1)   // vertices already winning following σ1
2  if T' = Win1(G) then
3  |  return (σ1, 0)
4  else
5  |  select (v, v') from Frontier(T')
6  |  (σ1', β') ← Fix(G, σ1 [v ↦ (v, v')])
7  |  G' ← G_σ1(v)                    // player 1 must follow σ1 on v
8  |  if v ∈ Win1(G') then
9  |  |  (σ1'', β'') ← Fix(G', σ1)
10 |  |  if β'' < β' + 1 then
11 |  |  |  return (σ1'', β'')
12 |  |  end
13 |  end
14 |  return (σ1', β' + 1)
15 end
```

Base case if σ_1 is already surely winning.

Run the algorithm if we change $\sigma_1(v)$ to (v, v')
↳ This option would require $\beta' + 1$ changes to σ_1.

Run the algorithm if we force $\sigma_1(v)$ to not change.
↳ This option would require β'' changes to σ_1

Comparision between β'' and $\beta' + 1$ to choose the minimal number of changes

Find minimal changes: a greedy approach

Def:

$\text{Repair}_{\sigma_1}(v, v') := \text{Win}_1(\mathcal{G}, \sigma_1[v \mapsto (v, v')]) \setminus \text{Win}_1(\mathcal{G}, \sigma_1)$ is the set of states that are indirectly repaired by changing $\sigma_1(v)$ to (v, v').

→ The greedy algorithm selects the replacement that repairs the largest number of vertices.

Input: $\mathcal{G}$ a reachability game, σ_1 a strategy for player 1

$\text{Fix}(\mathcal{G}, \sigma_1)$:

1 $T' \leftarrow \text{Win}_1(\mathcal{G}, \sigma_1)$ // vertices already winning following σ_1

2 **if** $T' = \text{Win}_1(\mathcal{G})$ **then**

3 | **return** $(\sigma_1, 0)$

4 **end**

5 $F \leftarrow \text{Frontier}_1(T')$

6 $(v, v') \leftarrow \text{argmax}\{|\text{Repair}_{\sigma_1}(v, v')| \ (v, v') \in F\}$ → Selection of the "most promising" repair.

7 $(\sigma_1', \beta') \leftarrow \text{Fix}(\mathcal{G}, \sigma_1[v \mapsto (v, v')])$

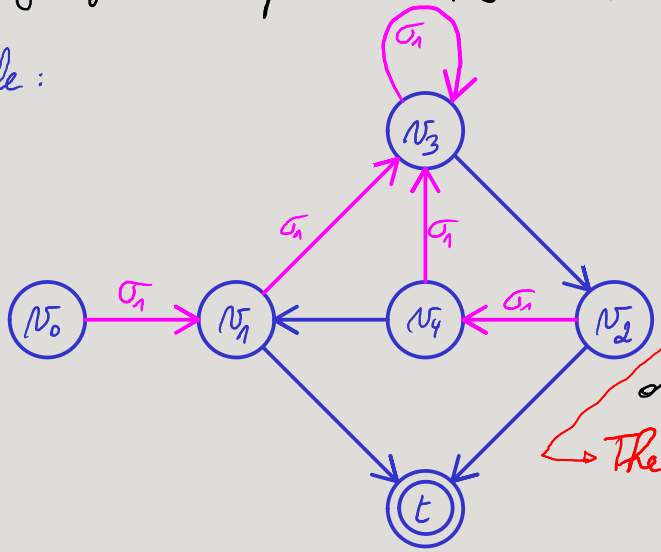
8 **return** $(\sigma_1', \beta' + 1)$

Find minimal changes: a greedy approach

Proposition:

The greedy algorithm outputs a simply winning strategy, but not always with minimal changes.

Example:



$\text{Win}_1(\mathcal{G}, \sigma_1) = \{t\}$

$\text{Frontier}_1(\text{Win}_1(\mathcal{G}, \sigma_1)) = \{v_1, v_2\}$

$\text{Repair}_{\sigma_1}(v_1, t) = \{v_0, v_1\}$

and $\text{Repair}_{\sigma_1}(v_2, t) = \{v_2\}$

→ The greedy algorithm will repair v_1.

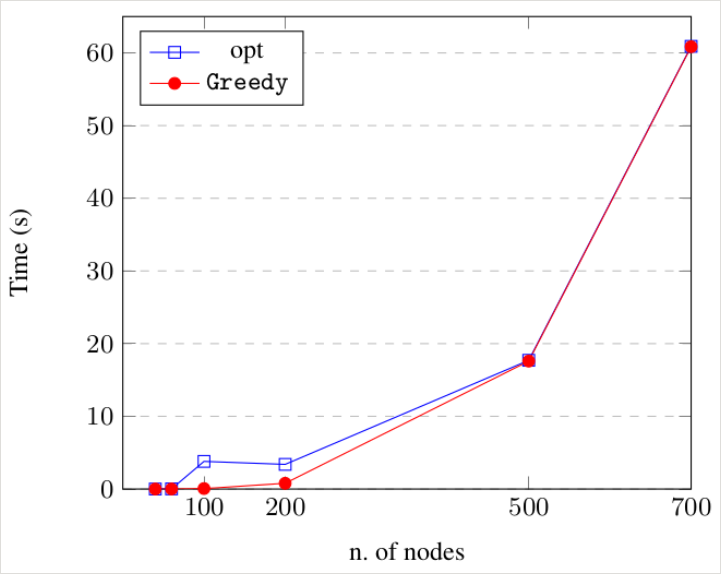
Comparison between the approaches - Random instances

The accuracy of the greedy algorithm on a game is defined as $\frac{d(\sigma_1, \sigma_1^{opt})}{d(\sigma_1, \sigma_1^{greedy})}$

where σ_1 is the initial strategy and σ_1^{opt} and σ_1^{greedy} are the strategies returned by the exact and greedy algorithms respectively.

Results for randomly generated games:

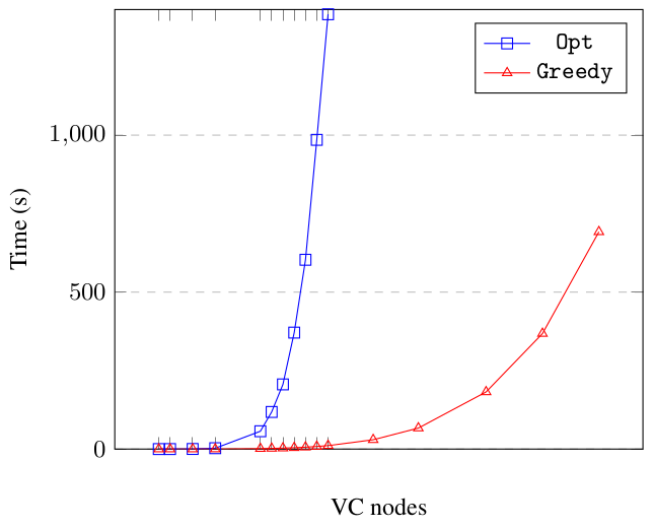
N. nodes	N. experiments	Opt	Greedy	Accuracy
40	5000	0.0012	0.001	0.9994
60	5000	0.02	0.0065	0.9952
100	1000	3.8	0.064	0.9904
200	700	3.39	0.79	0.9994
500	300	17.71	17.6	1
700	150	60.9	60.85	1



Comparison between the approaches - Instances from vertex cover.

Results for instances of Vertex Cover (VC) transformed via the construction from the proof of the NP-completeness theorem (max timer: 30 min)

VC nodes	N. experiments	Opt	Greedy	Accuracy
5	5,000	0.0009	0.0005	0.9815
10	2,000	0.013	0.004	0.889
20	500	0.32	0.041	0.875
30	100	2.8	0.185	0.8835
50	5	56.8	1.44	0.923
55	5	118.2	2.11	0.911
60	5	206	3.09	0.906
65	3	371	4.32	0.914
70	3	603	6	0.9254
75	2	985	8.28	0.937
80	2	1385	11	0.929
100	10	Timeout	29.7	N/A
120	4	Timeout	66.9	N/A
150	4	Timeout	182	N/A
175	3	Timeout	368	N/A
200	3	Timeout	692	N/A



Conclusion :

The presentation in a nutshell :

- Strategy Repair
- NP-completeness of the associated decision problem
- Exact algorithm to find the closest simply winning strategy
- Greedy algorithm with comparison on the performances of these algorithms

Future work mentioned in [GPP23] :

- Obtain approximation guarantees on the accuracy of the greedy approach.
- Extend to other games : parity, Büchi, etc.

Thanks for your attention !

Comparison with the classical fixpoint algorithm

Results about the mean of the distance between the original strategy and the new one.

N. nodes	N. experiments	`Opt` dist	`Greedy` dist	scratch dist
40	5000	0.81	0.811	1.6
60	1000	1.856	1.86	4
100	500	3.5	3.5	10.1
200	100	10.9	10.9	37.2
500	100	55.5	55.5	160
700	5	96	96	257