Augmented Decision Spaces for Stackelberg Security Games: Sparse evolution begets scalability

Adam Żychowski[1], Abhishek Gupta[2], Yew-Soon Ong[3], Jacek Mańdziuk[1,4]

[1]Faculty of Mathematics and Information Science, Warsaw University of Technology, Poland

[2]School of Mechanical Sciences, Indian Institute of Technology Goa, India

[3]College of Computing and Data Science, Nanyang Technological University, Singapore

[4]Faculty of Computer Science, AGH University of Krakow, Poland

Jul 16, 2025

Stackelberg Security Games

- Two asymmetrical players: **Defender and Attacker**
- Each game is composed of $\boldsymbol{m}$ **time steps**
- Each player chooses an action to be performed in each time step
- A player's pure strategy σ_P $(P \in \{D, A\})$ is a sequence of their actions in consecutive time steps: $\sigma_P = (a_1, a_2, \ldots, a_m)$

- **Defender commits strategy first**
- Attacker, **knowing the Defender's strategy**, chooses his strategy

Stackelberg equilibrium

Stackelberg equilibrium: a pair of players' strategies, for which strategy change by any of players leads to his/her result deterioration.

$$(\pi_D^*, R(\pi_D^*)) \in \Pi_D \times \Pi_A$$

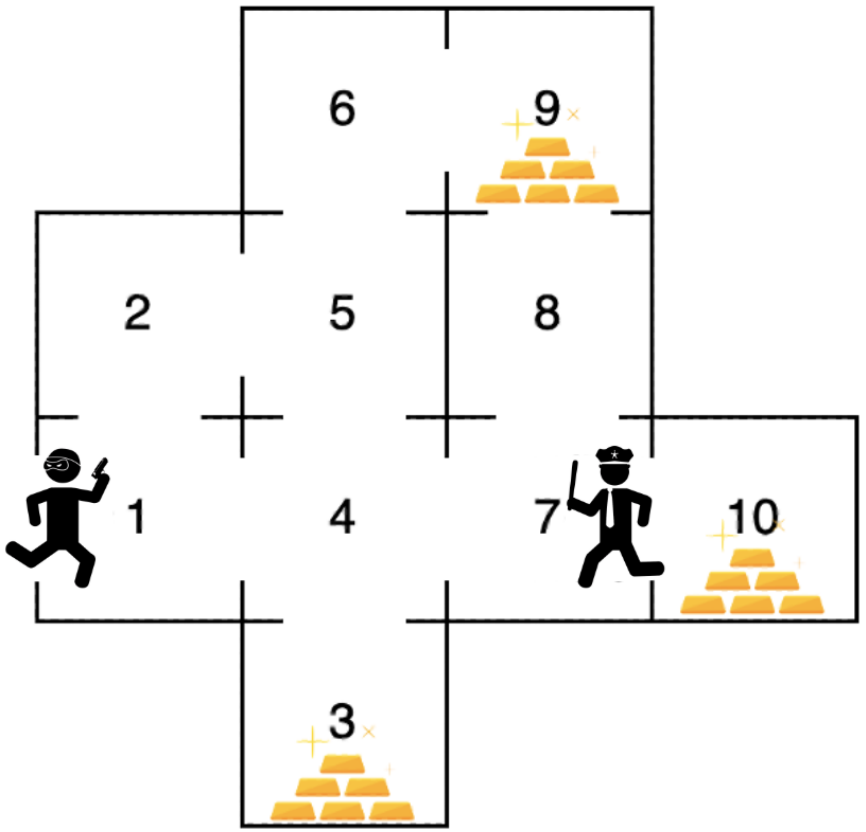
$\pi_D^* = \text{argmax}_{\pi_D \in \Pi_D} U_D(\pi_D, R(\pi_D))$ - Defender's optimal strategy

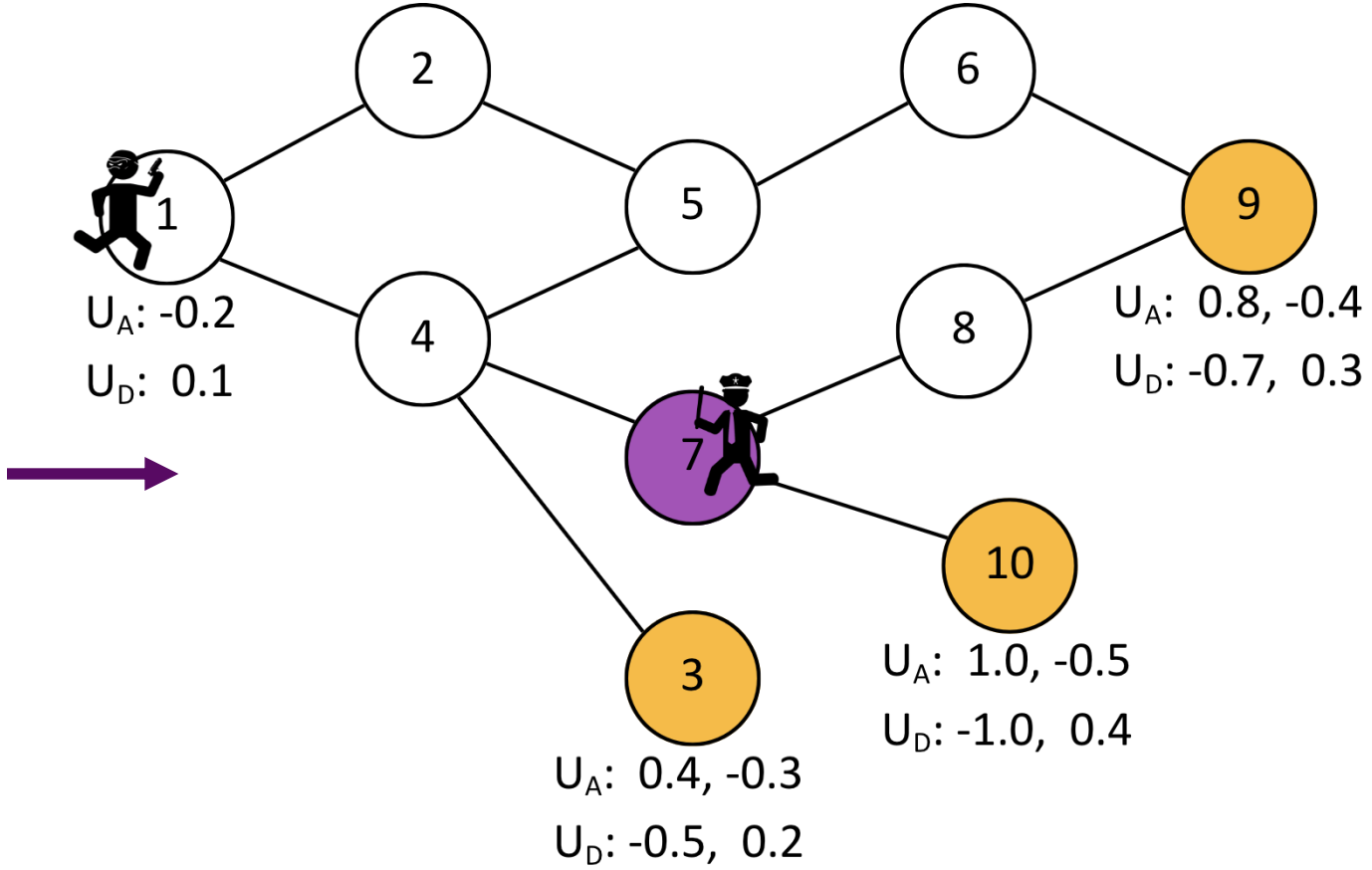
$R(\pi_D) = \text{argmax}_{\pi_A \in \Pi_A} U_A(\pi_D, \pi_A)$ - Attacker's optimal response

$G \in \{D, A\}$ - players (Defender, Attacker)
Π_G - a set of player's G all mixed strategies
U_G - payoff of player G

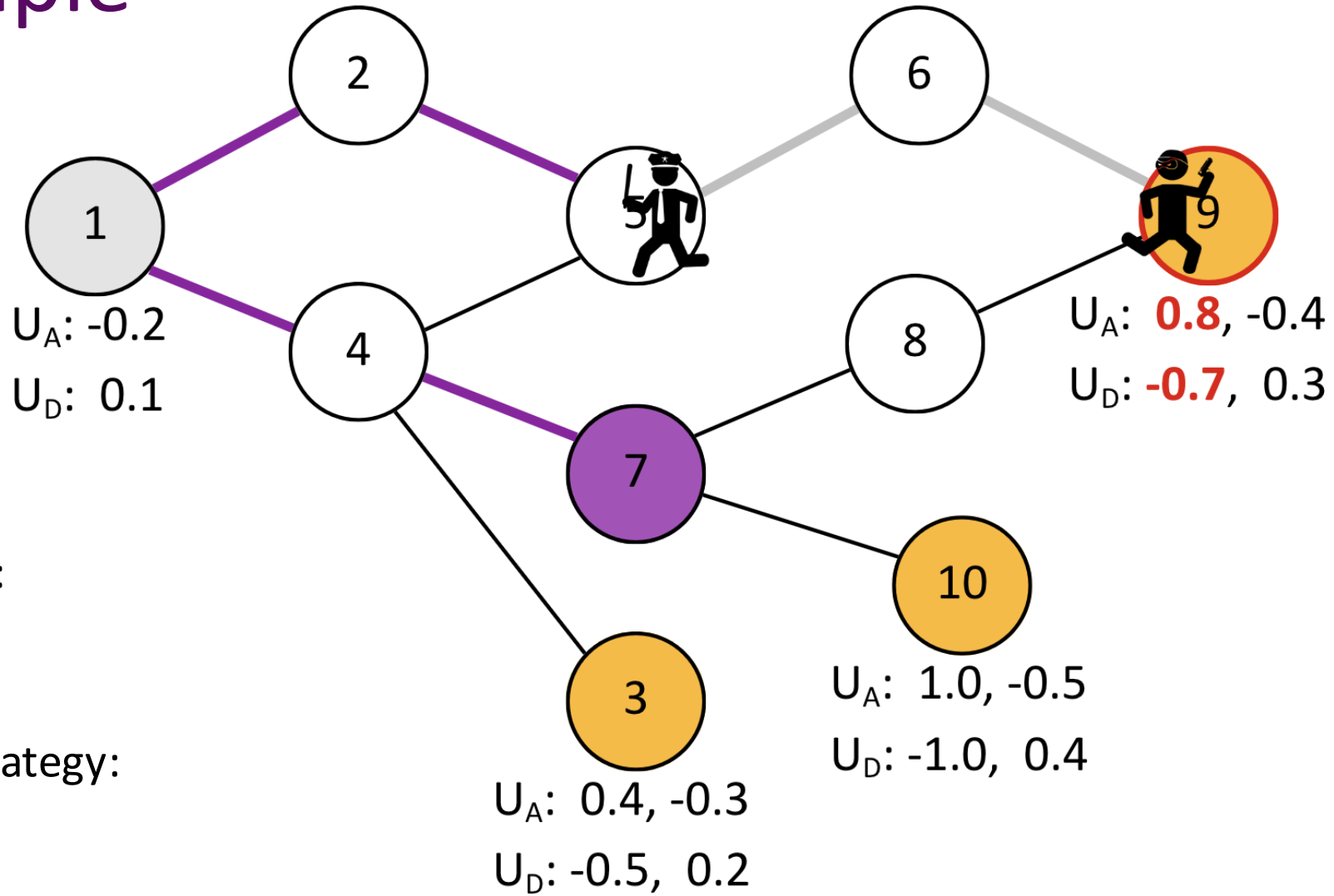
Goal: find optimal Defender's strategy

Example


6
9
2
5
8
1
4
7
10
3


2
6
1
5
9
4
8
7
10
3
U_A: -0.2
U_D: 0.1
U_A: 0.8, -0.4
U_D: -0.7, 0.3
U_A: 1.0, -0.5
U_D: -1.0, 0.4
U_A: 0.4, -0.3
U_D: -0.5, 0.2

Example



Attacker's strategy:

1,2,5,6,9

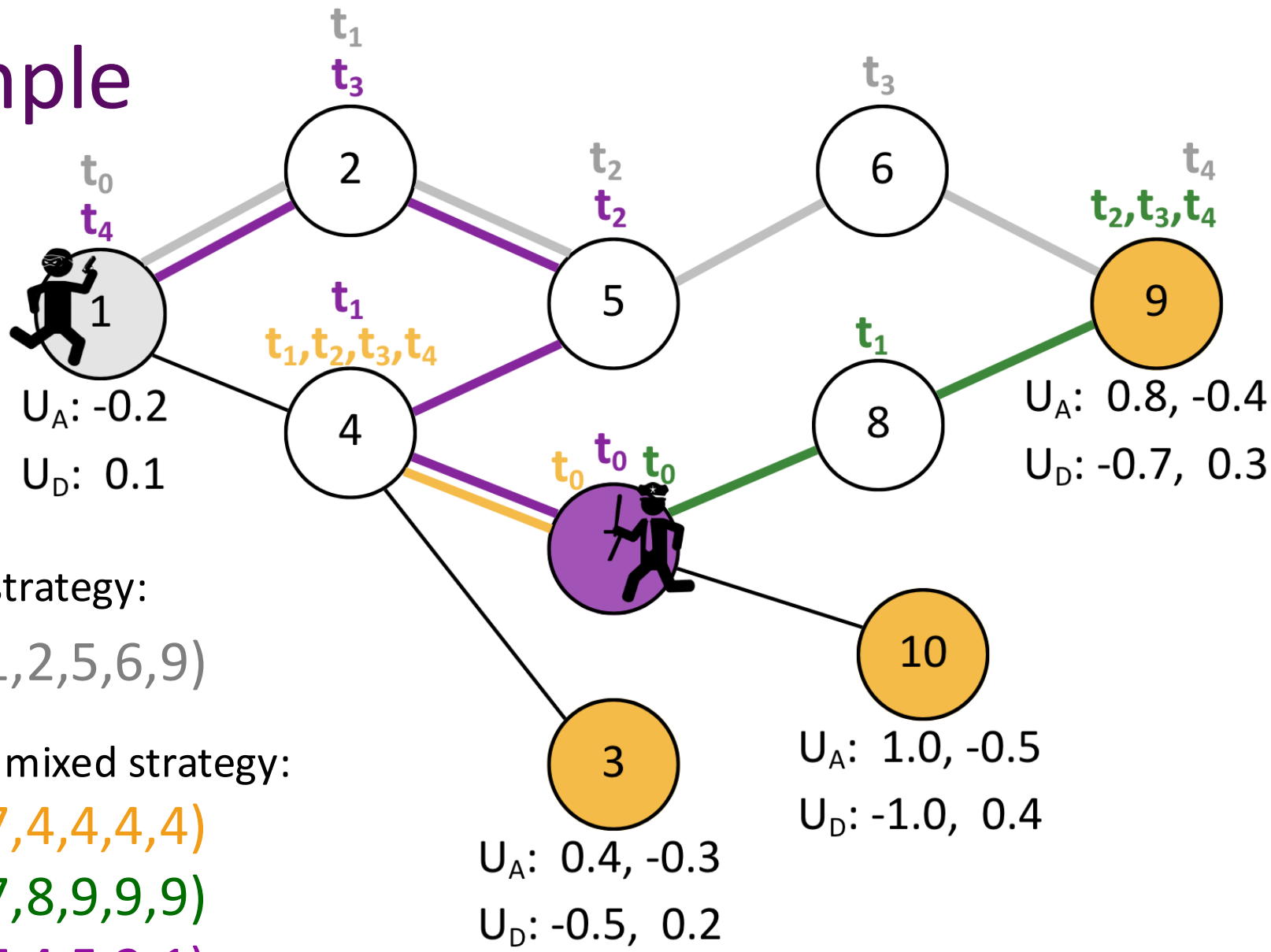
Defender's pure strategy:

7,4,1,2,5

U_A = **0.8**

U_D = **-0.7**

Example



Attacker's strategy:

1.0 (1,2,5,6,9)

Defender's mixed strategy:

0.6 (7,4,4,4,4)

0.1 (7,8,9,9,9)

0.3 (7,4,5,2,1)

$U_A = \mathbf{0.3 \cdot -0.2} + \mathbf{0.1 \cdot -0.4} + \mathbf{0.6 \cdot 0.8} = \mathbf{0.38}$

$U_D = \mathbf{0.3 \cdot 0.1} + \mathbf{0.1 \cdot 0.3} + \mathbf{0.6 \cdot -0.7} = \mathbf{-0.36}$

Real-life applications

Federal Air Marshal Service

US Coast Guard in Boston Harbor

Los Angeles Airport

Poaching in Uganda

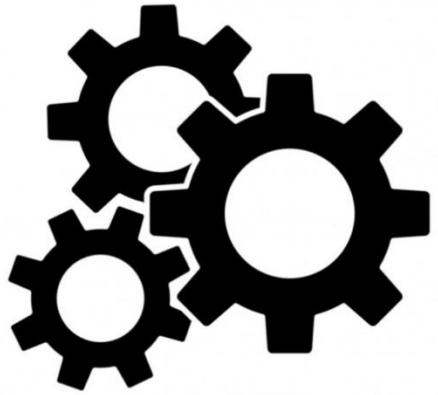
Wildlife and fishery protection

Tickets control in Los Angeles

Green Security Games

Żychowski, A., Mańdziuk, J., Bondi, E., Venugopal, A., Tambe, M., & Ravindran, B. "Evolutionary Approach to Security Games with Signaling." *31st International Joint Conference on Artificial Intelligence* **IJCAI 2022**.

Existing solutions



Mixed Integer Linear Programming

C2016

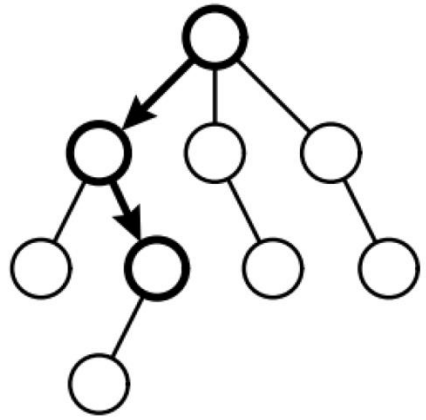
Cermak, J., Bosansky, B., Durkota, K., Lisy, V., Kiekintveld, C. *Using correlated strategies for computing stackelberg equilibria in extensive-form games*. AAAI 2016.

Evolutionary Algorithm

EASG, CoEvoSG

Żychowski A., Mańdziuk J. *Evolution of Strategies in Sequential Security Games*. AAMAS 2021.

Żychowski A., Mańdziuk J. *Coevolution of Players Strategies in Security Games.* Journal of Computational Science 2023



Monte Carlo Tree Search

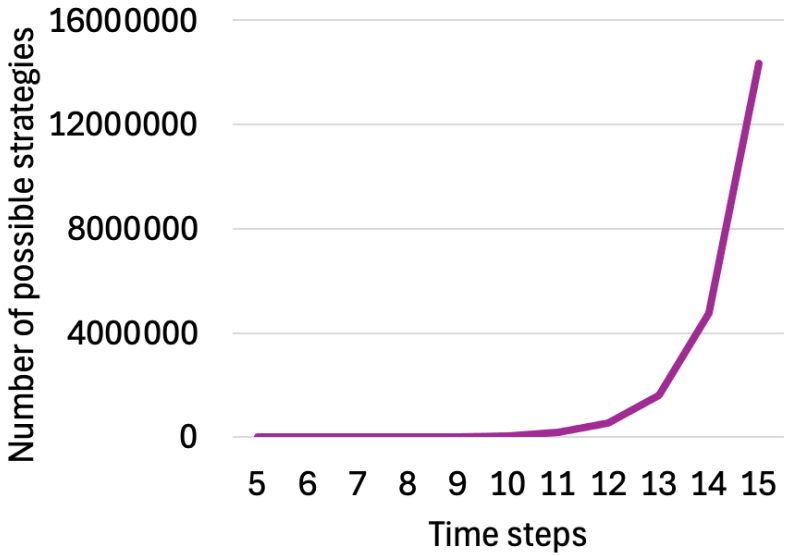
O2UCT

Karwowski J., Mańdziuk J. *Double-oracle sampling method for Stackelberg Equilibrium approximation in general-sum extensive-form games*. AAAI 2020.

Existing methods struggle with **scalability** and **sparsity**

Challenges

- Number of possible pure strategies grows **exponentially**
- NP-hard problem



All possible strategies for the presented example:

(7,4,1,2,1)	(7,4,5,4,5)	(7,4,3,4,4)	(7,4,7,7,10)	(7,4,4,4,1)	(7,8,7,7,4)	(7,8,8,7,10)	(7,10,7,8,8)	(7,7,4,1,1)	(7,7,4,4,4)	(7,7,10,10,7)	(7,7,7,7,7)
(7,4,1,2,5)	(7,4,5,4,3)	(7,4,3,3,4)	(7,4,7,7,7)	(7,4,4,4,5)	(7,8,7,7,8)	(7,8,8,7,7)	(7,10,7,10,7)	(7,7,4,5,2)	(7,7,8,7,4)	(7,7,10,10,10)	
(7,4,1,2,2)	(7,4,5,4,7)	(7,4,3,3,3)	(7,4,4,1,2)	(7,4,4,4,3)	(7,8,7,7,10)	(7,8,8,9,6)	(7,10,7,10,10)	(7,7,4,5,4)	(7,7,8,7,8)	(7,7,7,4,1)	
(7,4,1,4,1)	(7,4,5,4,4)	(7,4,7,4,1)	(7,4,4,1,4)	(7,4,4,4,7)	(7,8,7,7,7)	(7,8,8,9,8)	(7,10,7,7,4)	(7,7,4,5,6)	(7,7,8,7,10)	(7,7,7,4,5)	
(7,4,1,4,5)	(7,4,5,6,5)	(7,4,7,4,5)	(7,4,4,1,1)	(7,4,4,4,4)	(7,8,9,6,5)	(7,8,8,9,9)	(7,10,7,7,8)	(7,7,4,5,5)	(7,7,8,7,7)	(7,7,7,4,3)	
(7,4,1,4,3)	(7,4,5,6,9)	(7,4,7,4,3)	(7,4,4,5,2)	(7,8,7,4,1)	(7,8,9,6,9)	(7,8,8,8,7)	(7,10,7,7,10)	(7,7,4,3,4)	(7,7,8,9,6)	(7,7,7,4,7)	
(7,4,1,4,7)	(7,4,5,6,6)	(7,4,7,4,7)	(7,4,4,5,4)	(7,8,7,4,5)	(7,8,9,6,6)	(7,8,8,8,9)	(7,10,7,7,7)	(7,7,4,3,3)	(7,7,8,9,8)	(7,7,7,4,4)	
(7,4,1,4,4)	(7,4,5,5,2)	(7,4,7,4,4)	(7,4,4,5,6)	(7,8,7,4,3)	(7,8,9,8,7)	(7,8,8,8,8)	(7,10,10,7,4)	(7,7,4,7,4)	(7,7,8,9,9)	(7,7,7,8,7)	
(7,4,1,1,2)	(7,4,5,5,4)	(7,4,7,8,7)	(7,4,4,5,5)	(7,8,7,4,7)	(7,8,9,8,9)	(7,10,7,4,1)	(7,10,10,7,8)	(7,7,4,7,8)	(7,7,8,8,7)	(7,7,7,8,9)	
(7,4,1,1,4)	(7,4,5,5,6)	(7,4,7,8,9)	(7,4,4,3,4)	(7,8,7,4,4)	(7,8,9,8,8)	(7,10,7,4,5)	(7,10,10,7,10)	(7,7,4,7,10)	(7,7,8,8,9)	(7,7,7,8,8)	
(7,4,1,1,1)	(7,4,5,5,5)	(7,4,7,8,8)	(7,4,4,3,3)	(7,8,7,8,7)	(7,8,9,9,6)	(7,10,7,4,3)	(7,10,10,7,7)	(7,7,4,7,7)	(7,7,8,8,8)	(7,7,7,10,7)	
(7,4,5,2,1)	(7,4,3,4,1)	(7,4,7,10,7)	(7,4,4,7,4)	(7,8,7,8,9)	(7,8,9,9,8)	(7,10,7,4,7)	(7,10,10,10,7)	(7,7,4,4,1)	(7,7,10,7,4)	(7,7,7,10,10)	
(7,4,5,2,5)	(7,4,3,4,5)	(7,4,7,10,10)	(7,4,4,7,8)	(7,8,7,8,8)	(7,8,9,9,9)	(7,10,7,4,4)	(7,10,10,10,10)	(7,7,4,4,5)	(7,7,10,7,8)	(7,7,7,7,4)	
(7,4,5,2,2)	(7,4,3,4,3)	(7,4,7,7,4)	(7,4,4,7,10)	(7,8,7,10,7)	(7,8,8,7,4)	(7,10,7,8,7)	(7,7,4,1,2)	(7,7,4,4,3)	(7,7,10,7,10)	(7,7,7,7,8)	
(7,4,5,4,1)	(7,4,3,4,7)	(7,4,7,7,8)	(7,4,4,7,7)	(7,8,7,10,10)	(7,8,8,7,8)	(7,10,7,8,9)	(7,7,4,1,4)	(7,7,4,4,7)	(7,7,10,7,7)	(7,7,7,7,10)	

Motivation

Problem: Returned mixed strategies are highly **fragmented**

↓

- **suboptimal** (in practice optimal mixed strategies usually contains less than 5 pure strategies)
- **difficult to interpret** (less transparent to decision makers, difficult to explain, memorize, coordinate)

↓

Solution: We need *sparse* strategies

Example mixed strategy:

0.0077	(7,4,1,4,3)
0.0094	(7,4,5,6,9)
0.0054	(7,4,7,4,3)
0.0040	(7,4,4,5,2)
0.0241	(7,8,7,4,1)
0.0052	(7,8,9,6,9)
0.0235	(7,8,8,8,7)
0.0158	(7,10,7,7,10)
0.0174	(7,7,4,3,4)
0.0055	(7,7,8,9,6)
0.0095	(7,7,7,4,7)
0.0055	(7,4,5,2,5)
0.0049	(7,4,3,4,5)
0.7981	(7,4,7,10,10)
0.0181	(7,4,4,7,8)
0.0009	(7,8,7,8,8)
0.0364	(7,8,9,9,9)
0.0030	(7,10,7,4,4)
0.0057	(7,10,10,10,10)

Augmented Decision Space Optimization (ADSO)

Evolution Strategy + Augmented Space

ADS splits mixed strategy variables into 2 spaces:

- Binary: **selects** pure strategies (sparsity) - $\hat{\boldsymbol{x}}$
- Real-valued: **tunes** selection probabilities - $\tilde{\boldsymbol{x}}$

Mixed strategy distribution:

$$\boldsymbol{x} = [x_1, x_2, \dots, x_n],\ \sum x_i = 1\ ,\ x_i \geq 0$$

Augmented decision space:

$$\boldsymbol{x}' = (\hat{x}, \tilde{x}) = [\hat{x}_1, \tilde{x}_1, \hat{x}_2\ , \tilde{x}_2, \dots, \hat{x}_n, \tilde{x}_n],\ \ x_i = \hat{x}_i \tilde{x}_i,\quad \hat{x}_i \in \{0,1\}$$

Intuition:

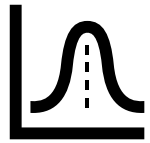
- Rather than waiting for evolution to sparsify the strategies, enforce sparsity from the start
- Direct control of sparsity

Augmented Decision Space Optimization (ADSO)

Probabilistic modelling based evolutionary search:

- sampling binary variables $\hat{x}$ from **Bernoulli distribution** 

$$P(\hat{x}_i) = q_i^{\hat{x}_i} \cdot (1 - q_i)^{1-\hat{x}_i}$$

- sampling real-valued decision variables $\tilde{x}$ from **normal distribution** 

$$p(\tilde{\mathbf{x}}) = \mathcal{N}(\mu_{\tilde{\mathbf{x}}}, \Sigma_{\tilde{\mathbf{x}}})$$

Probabilistic distribution defined in augmented decision space:

$$p(\mathbf{x}') = \mathcal{N}(\mu_{\tilde{\mathbf{x}}}, \Sigma_{\tilde{\mathbf{x}}}) \cdot \prod_{i=1}^{n} q_i^{\hat{x}_i} \cdot (1 - q_i)^{1-\hat{x}_i}$$

Mathematical backbone

Maximizing the expected Defender's payoff:

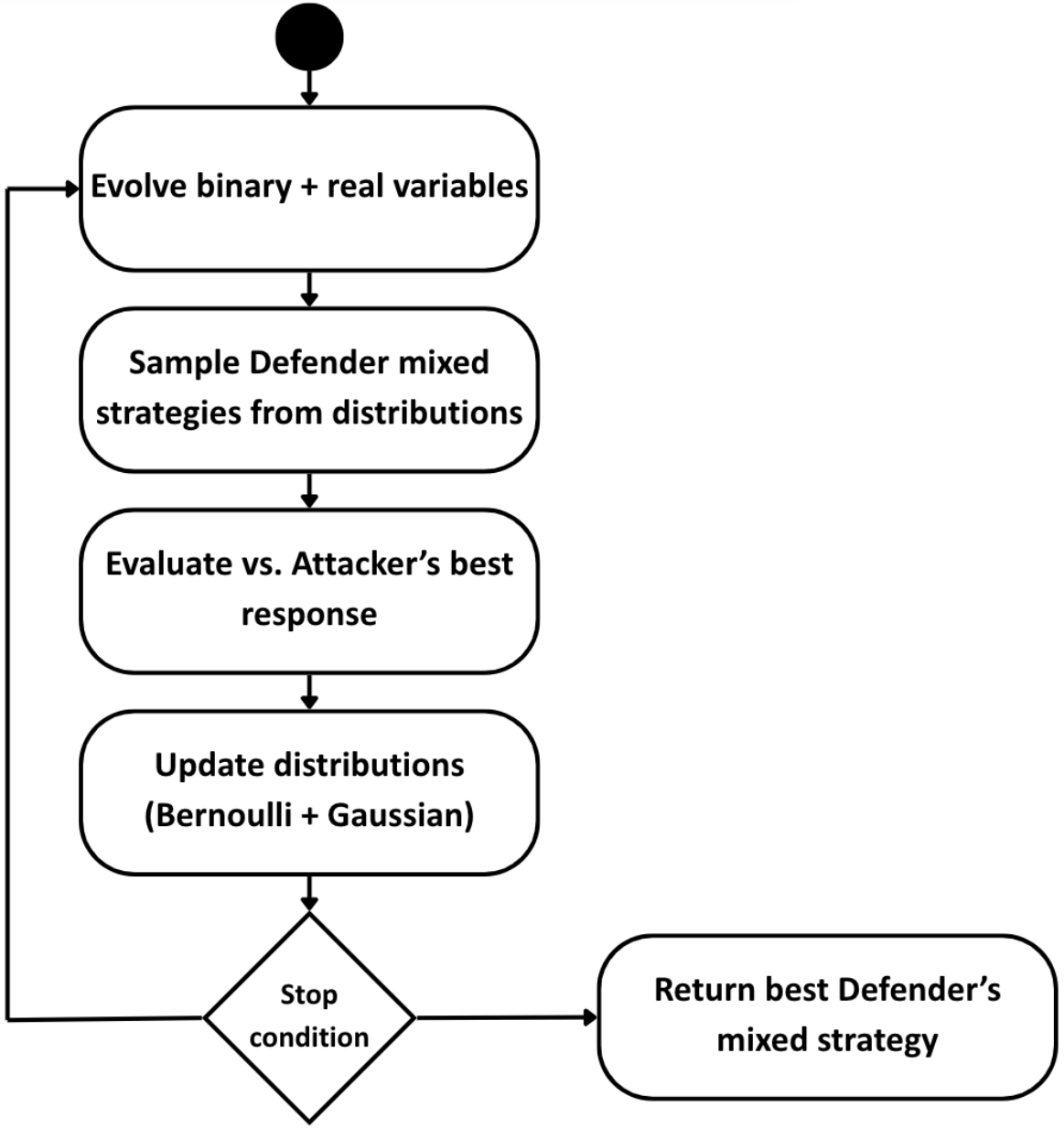
$$\arg \max_{\mu_{\tilde{\mathbf{x}}}, \Sigma_{\tilde{\mathbf{x}}}, q_1, q_2, \ldots, q_n} F = \sum_{\forall \hat{\mathbf{x}}} \left(\int f(\mathbf{x}') \cdot p(\tilde{\mathbf{x}})\, d\tilde{\mathbf{x}} \right) P(\hat{\mathbf{x}})$$

- Built on **Information-Geometric Optimization** (IGO)
- Leverages **Danskin's theorem** for efficiency in zero-sum games
- Simplifies updates for narrow distributions

Distribution updates:

- binary values → $q_i \leftarrow q_i + \frac{\eta}{N} \sum_{j=1}^{N} f_s(\hat{\mathbf{x}}^j, \boldsymbol{\mu}_{\tilde{\mathbf{x}}})(\hat{x}_i^j - q_i)$ (details and derivatives in the paper)
- real values → CMA-ES

ADSO outline



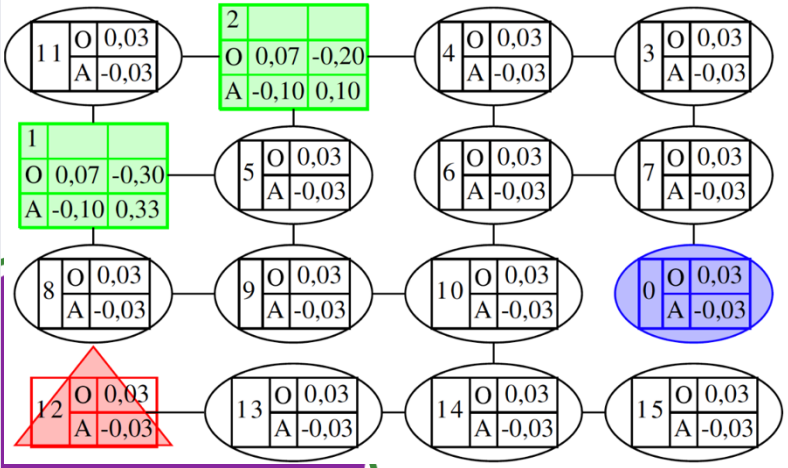
Experimental setup

450 test game instances of 3 types:

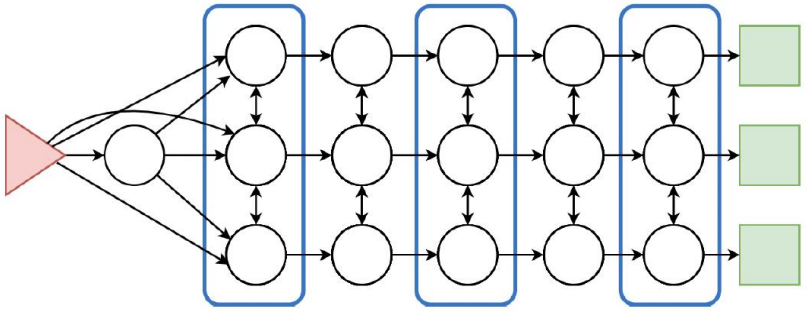
- Warehouse Games (WHG)
- Search Games (SEG)
- FlipIt Games (FIG)

30 independent runs for each game instance

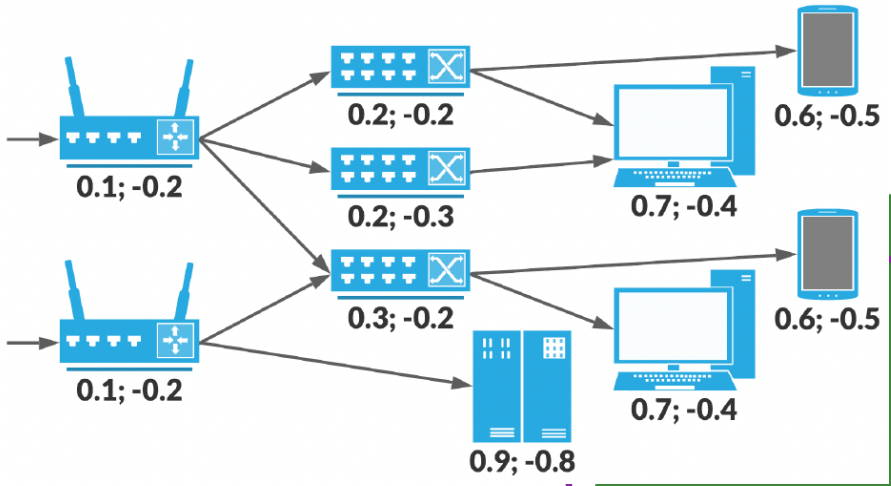
Warehouse Games



Search Games



FlipIt Games



Results

n	C2016	O2UCT	EASG	CoEvoSG	CMA-ES	ADSO
15	**0.052**	0.051	0.051	0.050	0.049	0.051
20	**0.054**	0.053	0.052	0.051	0.050	0.053
25	**0.048**	0.046	0.045	0.044	0.044	0.047
30	-	0.044	0.042	0.040	0.040	**0.045**
40	-	-	0.040	0.038	0.038	**0.041**

m	C2016	O2UCT	EASG	CoEvoSG	CMA-ES	ADSO
3	**0.043**	**0.043**	**0.043**	**0.043**	**0.043**	**0.043**
4	**0.052**	0.050	0.050	0.049	0.048	0.051
5	**0.055**	0.054	0.053	0.052	0.051	0.054
6	**0.058**	0.056	0.054	0.052	0.052	0.055
8	-	**0.053**	0.051	0.049	0.048	0.052
10	-	-	**0.048**	0.046	0.046	**0.048**

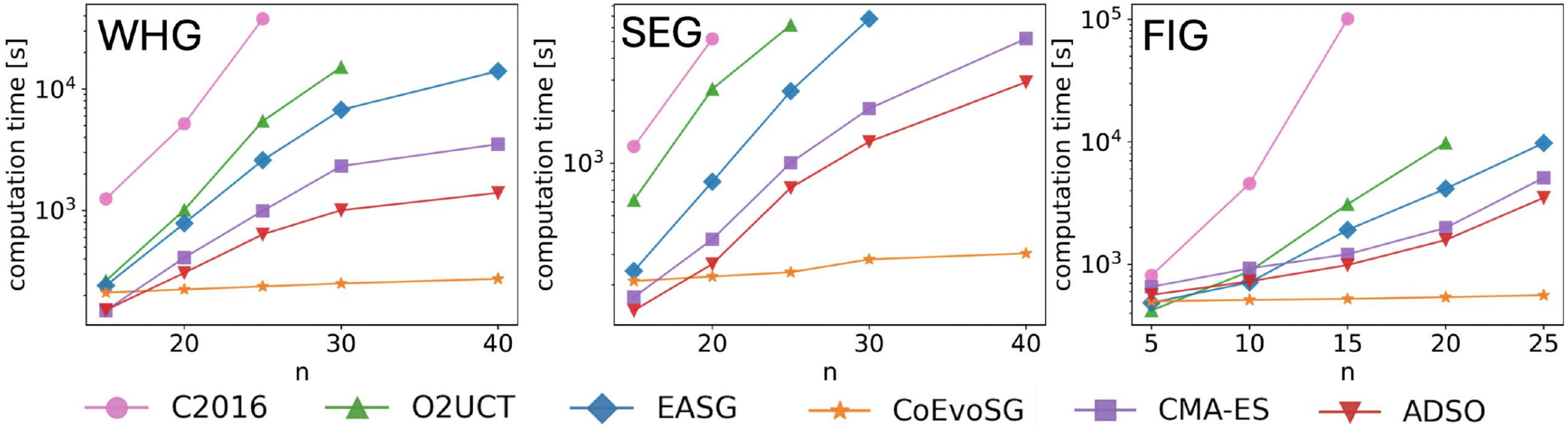
n	C2016	O2UCT	EASG	CoEvoSG	CMA-ES	ADSO
15	**0.122**	0.116	0.115	0.115	0.114	0.119
20	**0.117**	0.112	0.106	0.104	0.104	0.114
25	-	0.123	0.117	0.116	0.115	**0.124**
30	-	-	0.136	0.135	0.134	**0.137**
40	-	-	-	0.152	0.151	**0.154**

m	C2016	O2UCT	EASG	CoEvoSG	CMA-ES	ADSO
3	**0.137**	0.126	0.118	0.118	0.117	0.128
4	**0.124**	0.113	0.110	0.109	0.108	0.117
5	**0.106**	0.093	0.090	0.087	0.087	0.101
6	-	0.129	0.123	0.123	0.123	**0.134**
8	-	-	0.112	0.111	0.110	**0.117**
10	-	-	-	0.144	0.144	**0.149**

Average Defender's payoffs with respect to the number of graph nodes n (top) and time steps m (bottom)

- Matches MILP (optimal) solution in **73.3%** of cases
- Outperforms all heuristics in **74.5%** of games
- Number of returned pure strategies reduced from 10-32 to **6.76**

Results - scalability



- **MILP**: fails on large games
- **CMA-ES, O2UCT, EASG**: dense/fragmented strategies, slower
- **CoEvoSG**: faster evaluation (no searching for Attacker's best response), the worst results
- **ADSO**: sparse strategies, faster evaluation

Summary

- ADSO introduces a novel **dual encoding for mixed strategies**
- Achieves scalable, sparse, and high-quality strategies
- **Beats existing heuristic** and found near the optimal solutions in the majority cases
- Opens doors for broader applications in strategic decision-making
- **General-purpose framework** - not tied to specific game rules

THANK YOU

and consider voting for Best Paper 🙂

Full paper

adam.zychowski@pw.edu.pl

Results

	C2016	O2UCT	EASG	CoEvoSG	CMA-ES	ADSO
WHG	60 (100%)	39 (65.0%)	43 (71.7%)	38 (63.3%)	15 (25.0%)	**49 (81.7%)**
SEG	60 (100%)	36 (60.0%)	26 (43.3%)	17 (28.3%)	12 (20.0%)	**41 (68.3%)**
FIG	30 (100%)	17 (56.7%)	19 (63.3%)	16 (53.3%)	5 (16.7%)	**21 (70.0%)**

The number of games in which each method successfully identified the optimal strategy (achieved a Defender's payoff difference of less than ε = 0.0001 compared to the C2016 solution).

Number of returned pure strategies reduced from 32.73 to **6.76**.

$$\arg \max_{\mu_{\tilde{\mathbf{x}}}, \Sigma_{\tilde{\mathbf{x}}}, q_1, q_2, \ldots, q_n} F = \sum_{\forall \hat{\mathbf{x}}} \left(\int f(\mathbf{x}') \cdot p(\tilde{\mathbf{x}}) \, d\tilde{\mathbf{x}} \right) P(\hat{\mathbf{x}})$$

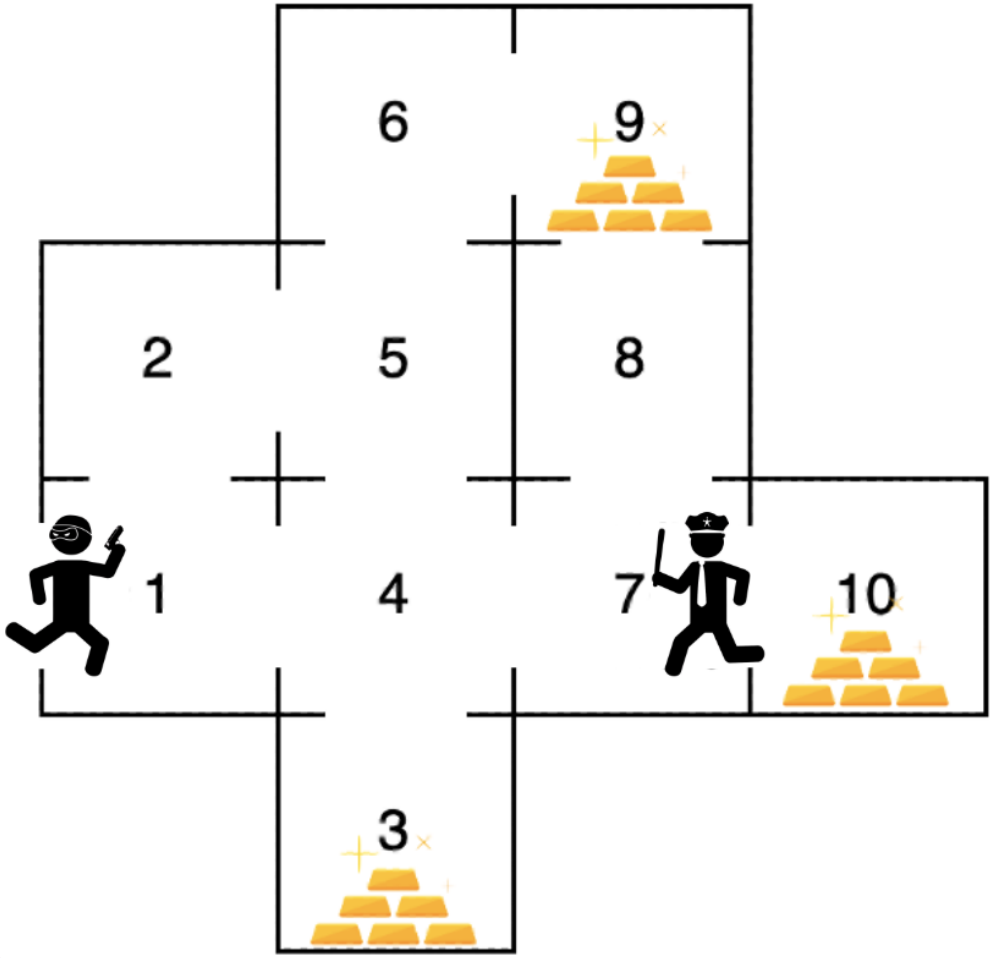
Following the Leibniz integral rule and the log-likelihood trick the gradients of F with respect to the parameters of the binary and the normal distributions are:

$$\nabla_{q_1, \ldots, q_n} F = \sum_{\forall \hat{\mathbf{x}}} \left(\int f(\mathbf{x}') \cdot p(\tilde{\mathbf{x}}) \, d\tilde{\mathbf{x}} \right)$$

$$\nabla_{q_1, \ldots, q_n} \left(\sum_{i=1}^{n} (\hat{x}_i \log(q_i) + (1 - \hat{x}_i) \log(1 - q_i)) \right) P(\hat{\mathbf{x}}) \quad (4)$$

$$\nabla_{\mu_{\tilde{\mathbf{x}}}, \Sigma_{\tilde{\mathbf{x}}}} F = \sum_{\forall \hat{\mathbf{x}}} \left(\int f(\mathbf{x}') \cdot \nabla_{\mu_{\tilde{\mathbf{x}}}, \Sigma_{\tilde{\mathbf{x}}}} (\log(\mathcal{N}(\mu_{\tilde{\mathbf{x}}}, \Sigma_{\tilde{\mathbf{x}}}))) \cdot p(\tilde{\mathbf{x}}) \, d\tilde{\mathbf{x}} \right) P(\hat{\mathbf{x}}). \quad (5)$$

Example

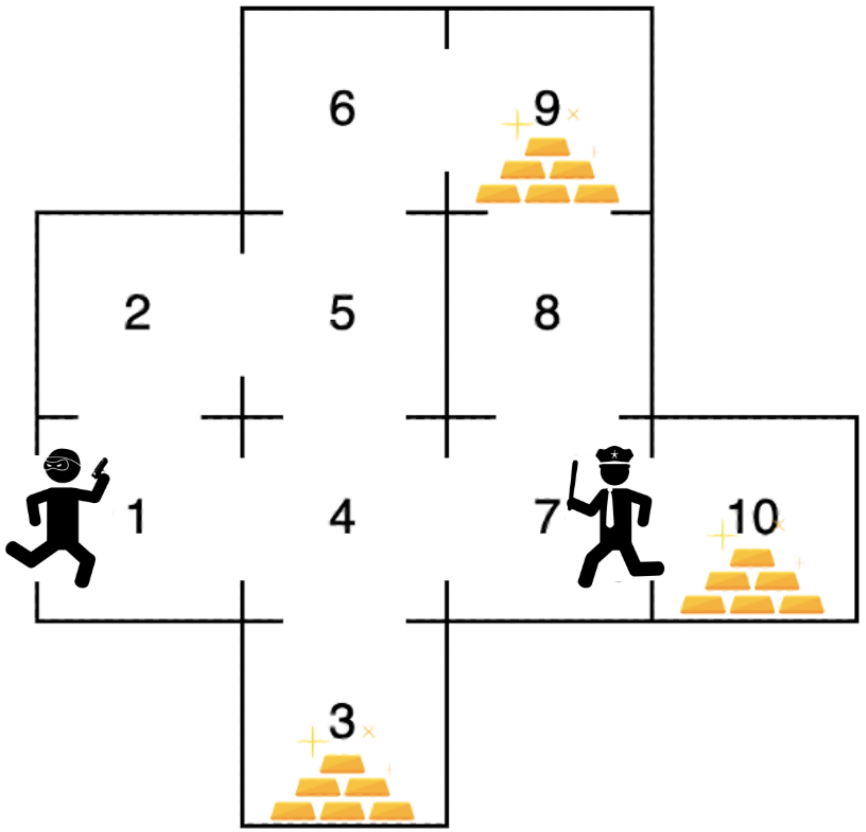


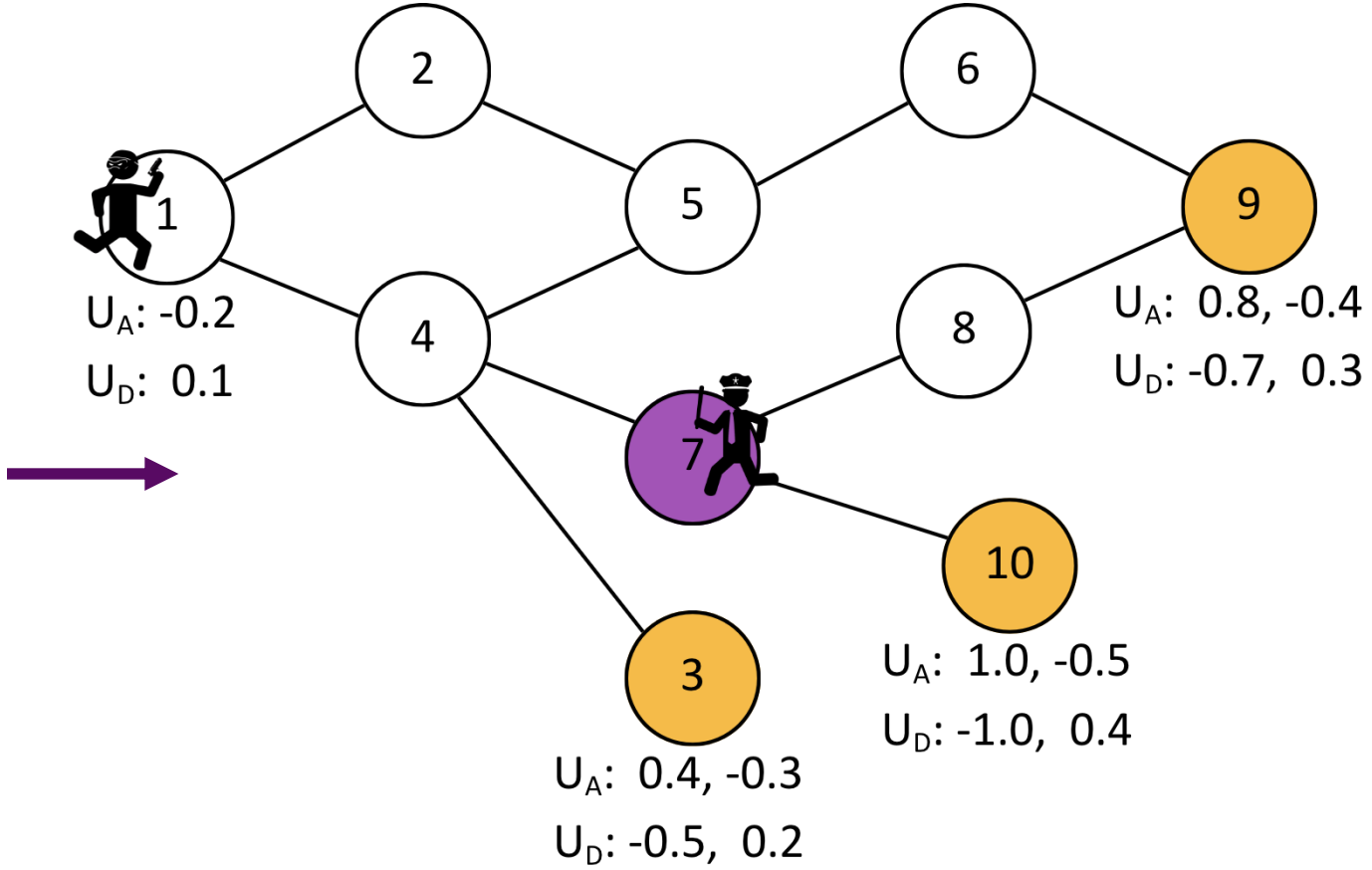
Defender

Attacker

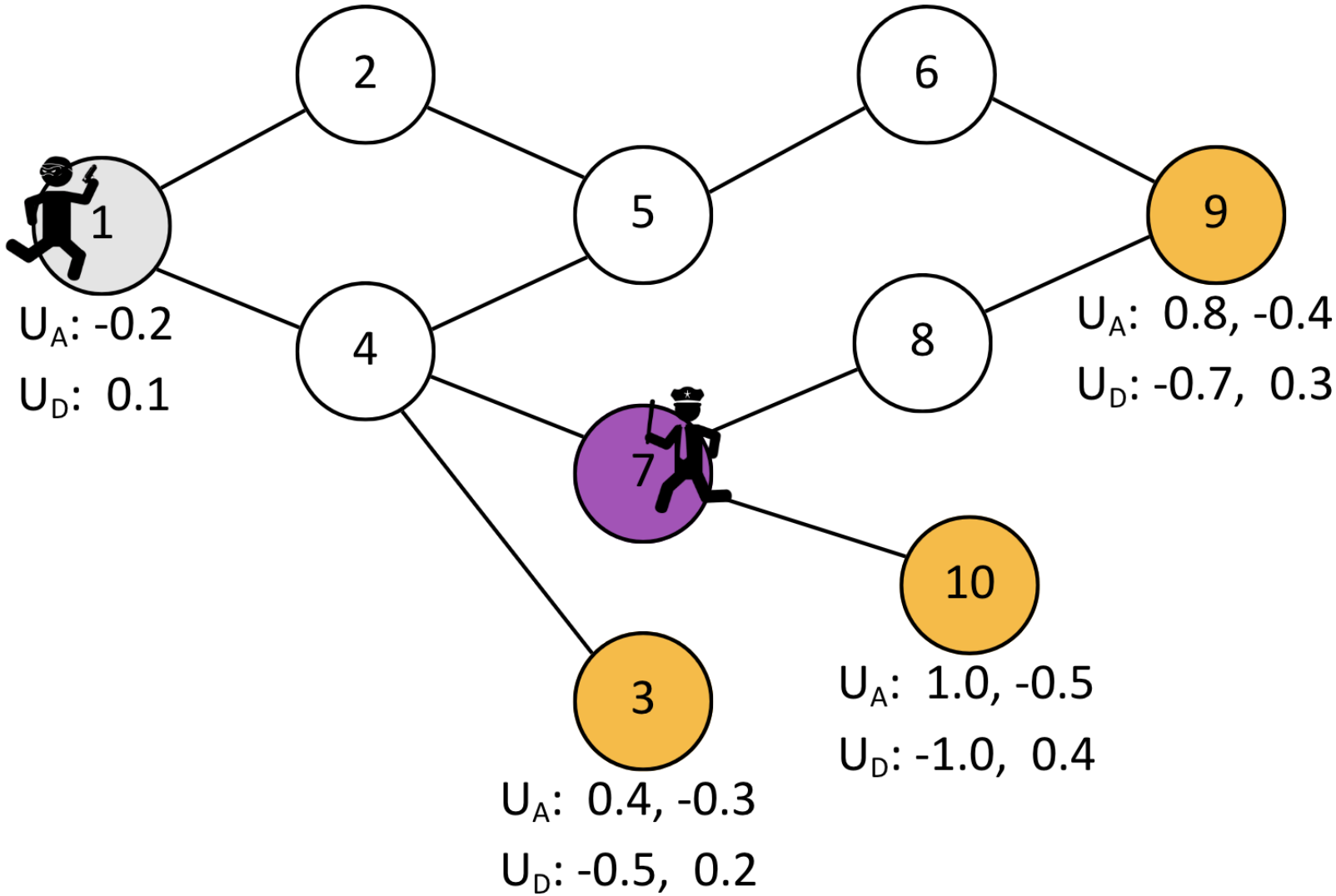
Target

Example


6
9
2
5
8
1
4
7
10
3


2
6
1
5
9
4
8
7
10
3
U_A: -0.2
U_D: 0.1
U_A: 0.8, -0.4
U_D: -0.7, 0.3
U_A: 1.0, -0.5
U_D: -1.0, 0.4
U_A: 0.4, -0.3
U_D: -0.5, 0.2

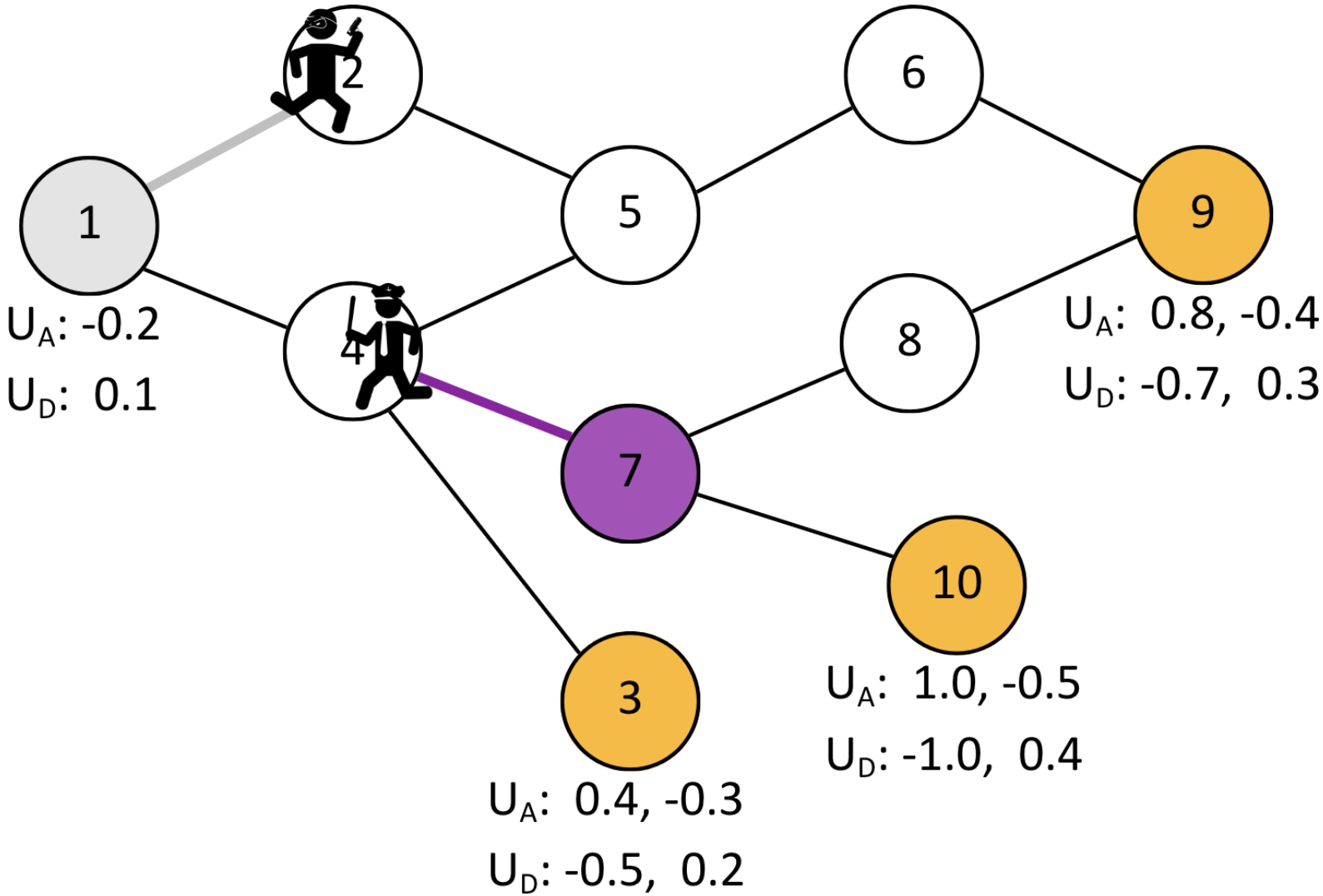
t_0


2
6
1
5
9
U_A: -0.2
U_D: 0.1
4
8
U_A: 0.8, -0.4
U_D: -0.7, 0.3
7
10
3
U_A: 1.0, -0.5
U_D: -1.0, 0.4
U_A: 0.4, -0.3
U_D: -0.5, 0.2

1

7

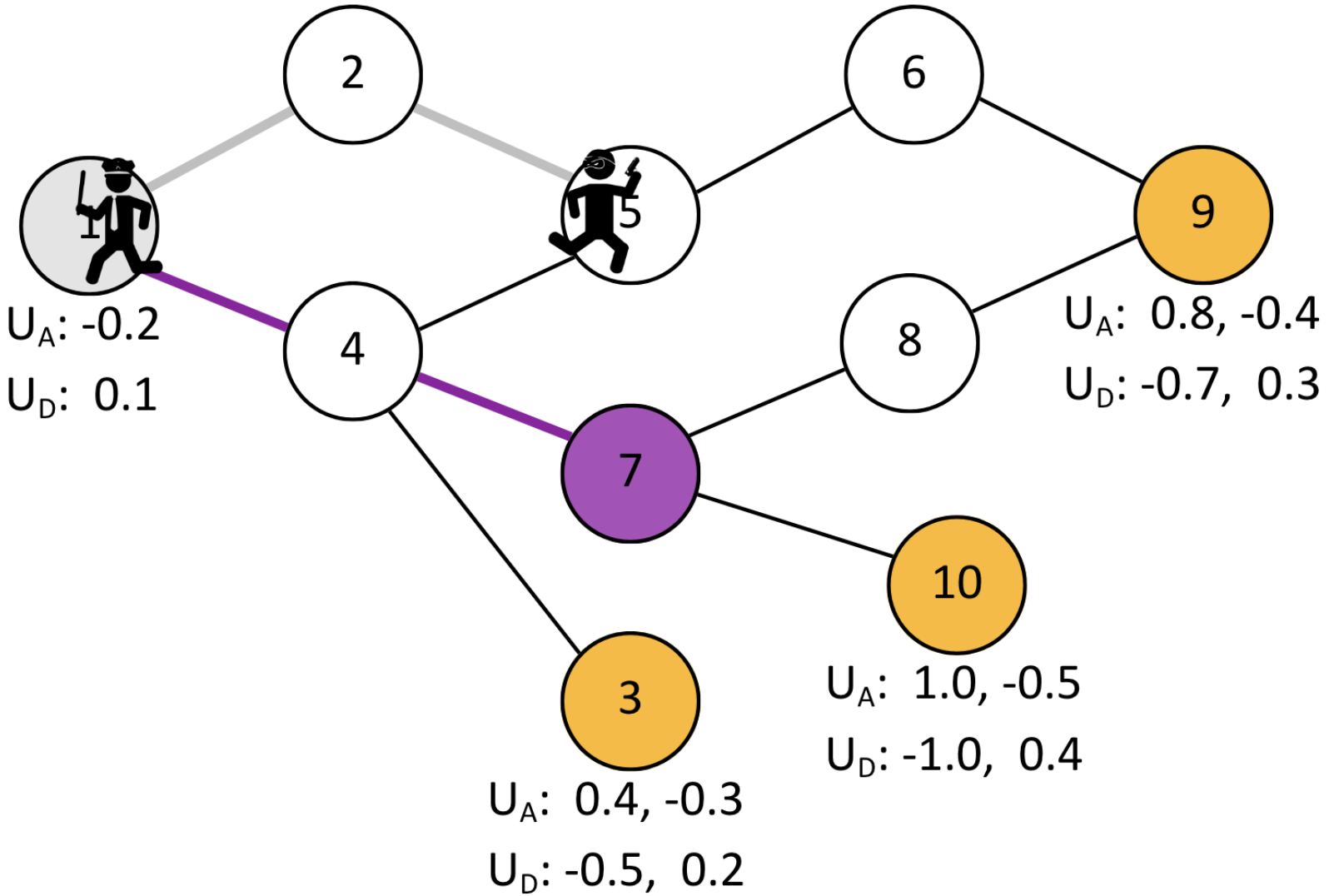
t_1


2
6
1
5
9
4
8
7
10
3
U_A: -0.2
U_D: 0.1
U_A: 0.8, -0.4
U_D: -0.7, 0.3
U_A: 1.0, -0.5
U_D: -1.0, 0.4
U_A: 0.4, -0.3
U_D: -0.5, 0.2

1,2

7,4

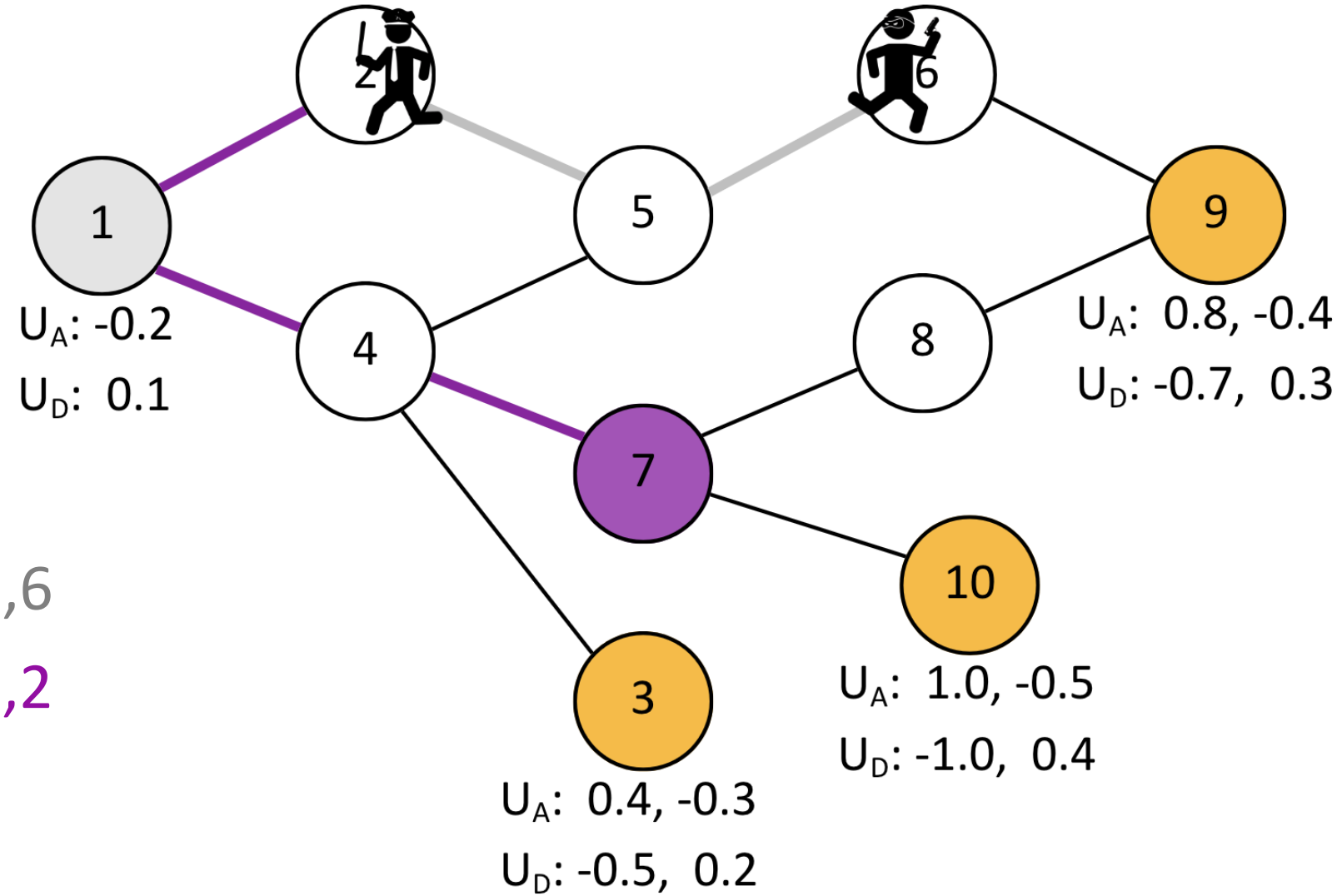
t_2


2
6
1
5
9
U_A: -0.2
U_D: 0.1
4
8
U_A: 0.8, -0.4
U_D: -0.7, 0.3
7
10
3
U_A: 1.0, -0.5
U_D: -1.0, 0.4
U_A: 0.4, -0.3
U_D: -0.5, 0.2

1,2,5

7,4,1

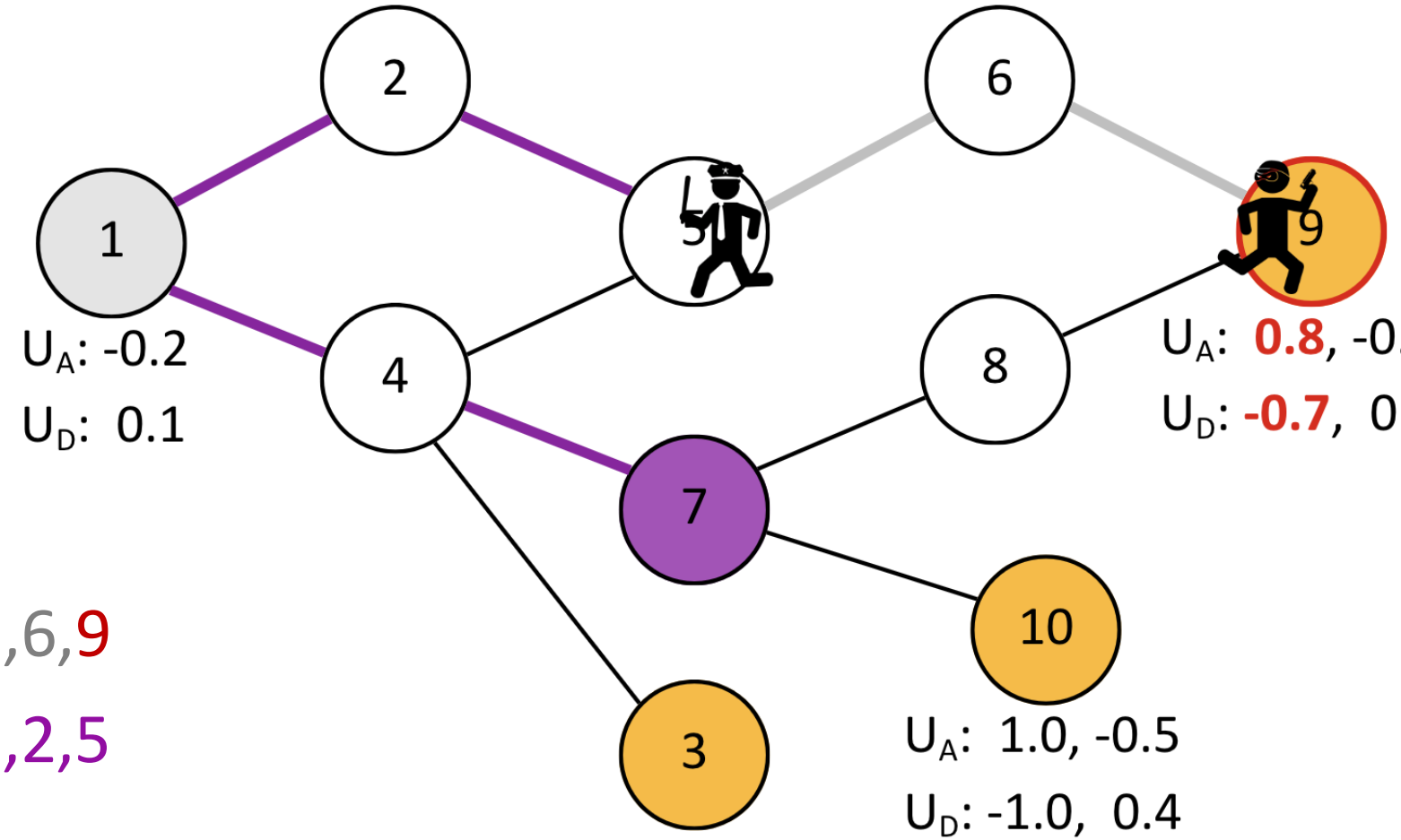
t_3


2
6
1
5
9
4
8
7
10
3
U_A: -0.2
U_D: 0.1
U_A: 0.8, -0.4
U_D: -0.7, 0.3
U_A: 1.0, -0.5
U_D: -1.0, 0.4
U_A: 0.4, -0.3
U_D: -0.5, 0.2

1,2,5,6

7,4,1,2

t_4



U_A: -0.2
U_D: 0.1

U_A: **0.8**, -0.4
U_D: **-0.7**, 0.3

U_A: 1.0, -0.5
U_D: -1.0, 0.4

U_A: 0.4, -0.3
U_D: -0.5, 0.2

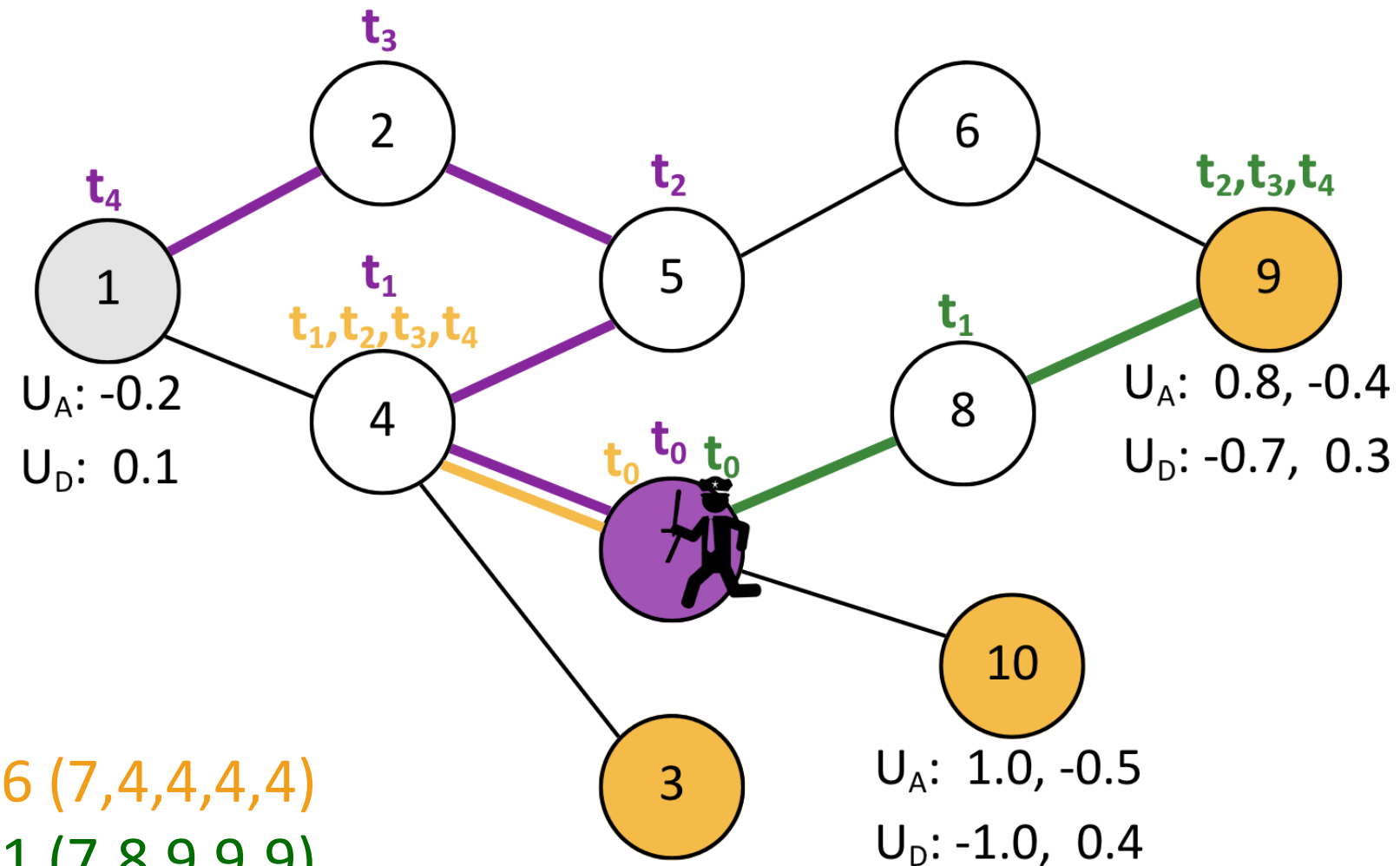
1,2,5,6,9

7,4,1,2,5

U_A = **0.8**

U_D = **-0.7**

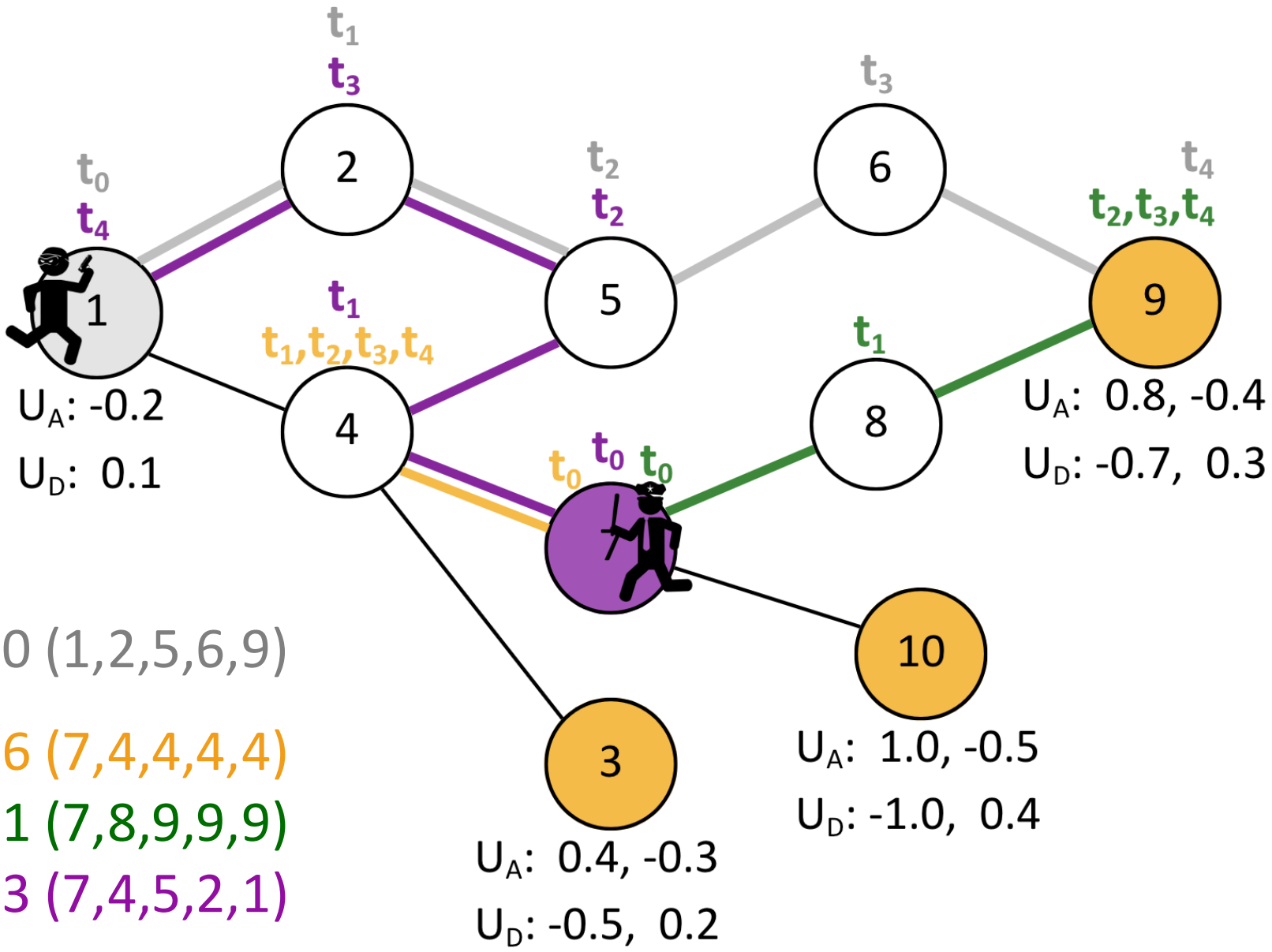
Example - mixed strategy

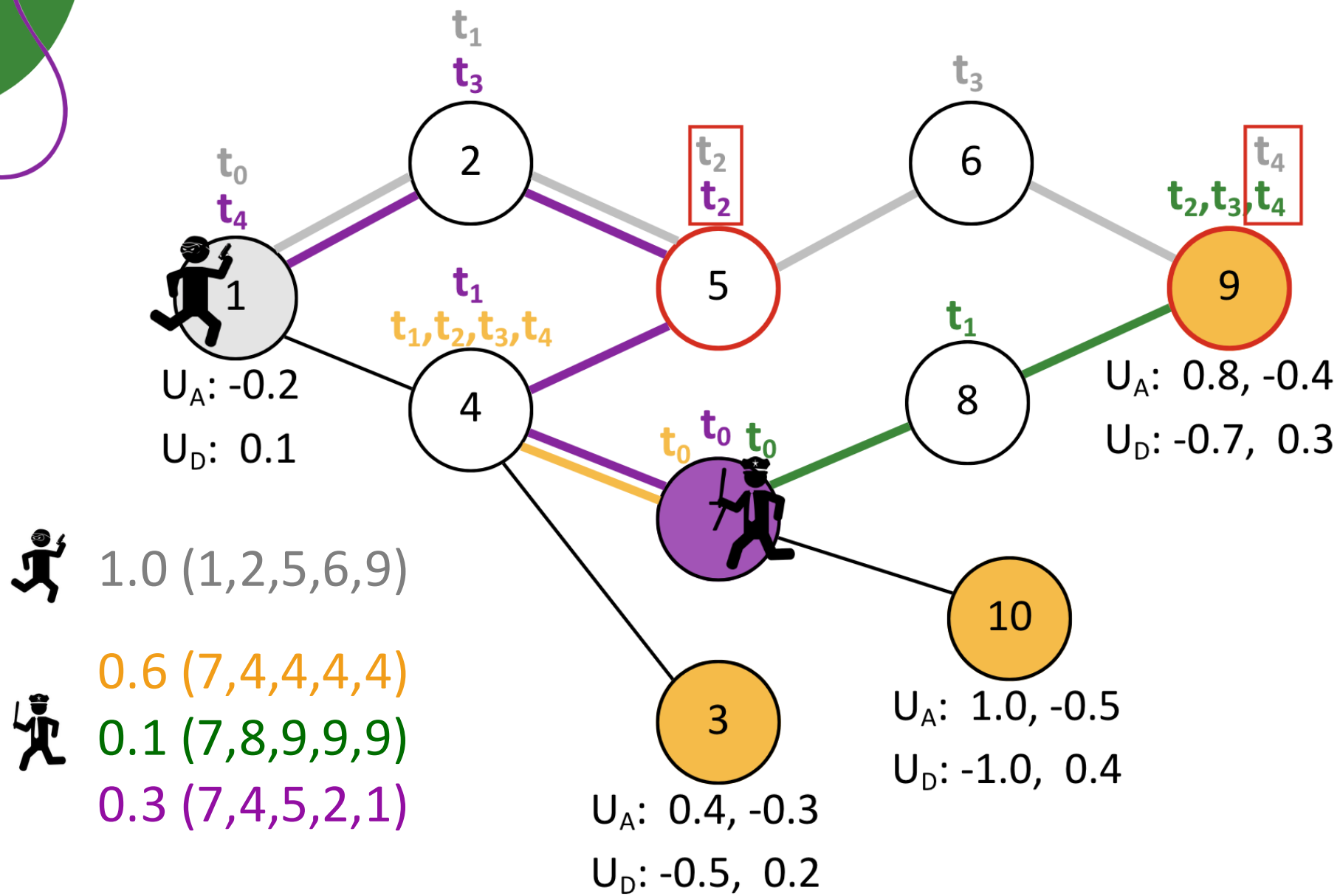


0.6 (7,4,4,4,4)

0.1 (7,8,9,9,9)

0.3 (7,4,5,2,1)


t1
t3
t3
t0
t4
t2
t2
t4
t2,t3,t4
2
6
1
5
9
t1
t1,t2,t3,t4
t1
4
8
UA: -0.2
UD: 0.1
UA: 0.8, -0.4
UD: -0.7, 0.3
t0 t0
t0
10
3
UA: 1.0, -0.5
UD: -1.0, 0.4
UA: 0.4, -0.3
UD: -0.5, 0.2
1.0 (1,2,5,6,9)
0.6 (7,4,4,4,4)
0.1 (7,8,9,9,9)
0.3 (7,4,5,2,1)



1.0 (1,2,5,6,9)

0.6 (7,4,4,4,4)

0.1 (7,8,9,9,9)

0.3 (7,4,5,2,1)

$U_A = 0.3 \cdot -0.2 + 0.1 \cdot -0.4 + 0.6 \cdot 0.8 = \mathbf{0.38}$

$U_D = 0.3 \cdot 0.1 + 0.1 \cdot 0.3 + 0.6 \cdot -0.7 = \mathbf{-0.36}$