Average Payoff Games

Marcin Jurdziński[1] **Ashutosh Trivedi**[1]

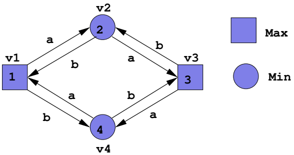
[1]Department of Computer Science
University of Warwick

22nd British Colloquium for Theoretical Computer Science
BCTCS 2006

Average payoff games

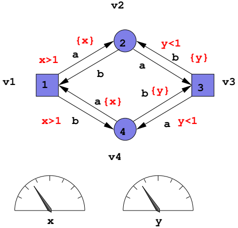
Discrete-time average payoff games

- 2-player perfect information, zero-sum games on weighted finite automata.
- run = $v_1 \xrightarrow{a_1} v_2 \xrightarrow{a_2} v_3 \ldots$
- players optimize
  - *reward per transition* (**mean-payoff games**).



Continuous-time average payoff games

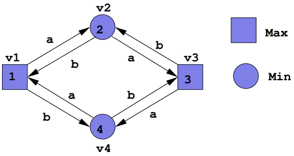
- 2-player perfect information, zero-sum games on priced timed automata
- run = $v_1 \xrightarrow[t_1]{a_1} v_2 \xrightarrow[t_2]{a_2} v_3 \ldots$
- players optimize :
  - *time per transition* (**average time game**)



Average payoff games

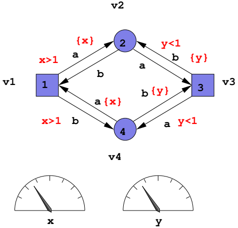
Discrete-time average payoff games

- 2-player perfect information, zero-sum games on weighted finite automata.
- run = $v_1 \xrightarrow{a_1} v_2 \xrightarrow{a_2} v_3 \ldots$
- players optimize
  - *reward per transition* (**mean-payoff games**).



Continuous-time average payoff games

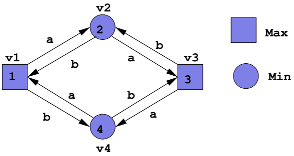
- 2-player perfect information, zero-sum games on priced timed automata
- run = $v_1 \xrightarrow[t_1]{a_1} v_2 \xrightarrow[t_2]{a_2} v_3 \ldots$
- players optimize :
  - *payoff per transition* (**average price game**)



Average payoff games

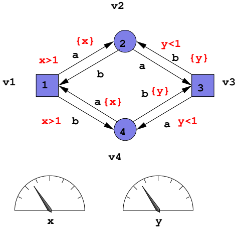
Discrete-time average payoff games

- 2-player perfect information, zero-sum games on weighted finite automata.
- run = $v_1 \xrightarrow{a_1} v_2 \xrightarrow{a_2} v_3 \ldots$
- players optimize
  - *reward per transition* (**mean-payoff games**).



Continuous-time average payoff games

- 2-player perfect information, zero-sum games on priced timed automata
- run = $v_1 \xrightarrow[t_1]{a_1} v_2 \xrightarrow[t_2]{a_2} v_3 \ldots$
- players optimize :
  - *payoff per unit time* (**average price game**)



Average payoff games (discrete-time)

Game arena

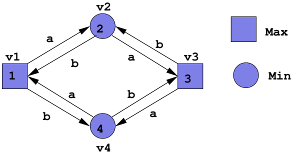
- A finite directed graph $G = \{V, E\}$
- A partition of vertices $V = V_{\text{Max}} \cup V_{\text{Min}}$
- A reward function $r : V \to \mathbb{Z}$.

Optimization goals

- $\pi = v_1 \xrightarrow{a_1} v_2 \xrightarrow{a_2} v_3 \ldots$
- $A(\pi) = \lim_{n\to\infty}(1/n) \cdot \sum_{i=1}^{n} r(v_i)$

Strategies

- **History dependent** $\Sigma_{\text{Max}} = \{V^* \times V_{\text{Max}} \to V\}$
- **Positional** $\Sigma_{\text{Max}} = \{V_{\text{Max}} \to V\}$



Average payoff games

Value of the game

- $A(v, \chi, \mu)$ = limiting average of the game.
- upper-value $A^*(v) = \inf_{\mu \in \Sigma_{\text{Min}}} \sup_{\chi \in \Sigma_{\text{Max}}} A(v, \mu, \chi)$,
- lower value $A_*(v) = \sup_{\chi \in \Sigma_{\text{Max}}} \inf_{\mu \in \Sigma_{\text{Min}}} A(v, \mu, \chi)$.
- (minimax) **value** of the game $A^*(v) = A_*(v) = A(v, \chi^*, \mu^*)$.

Algorithmic problem

- Find the optimal strategy for Max and Min.
- Pseudo-polynomial time algorithm [Zwick and Paterson'95].

Optimality equations

Local witness of global optimality

- **gain**(v) = value of the game.
- **bias**(v) = measure of initial fluctuations.

Optimality equations $\mathcal{OE}$

For every max vertex $v \in V_{\text{Max}}$,

- $g(v) = \max\{g(u) : (v,u) \in E\}$.
- $b(v) = \max\{r(v) - g(v) + b(u) : (u,v) \in E \text{ and } g(v) = g(u)\}$.

Positional optimal strategy

- Solution to $\mathcal{OE}$ gives optimal value of the game.
- Solution to $\mathcal{OE}$ provides *positional optimal strategy*.

Optimality equations

Local witness of global optimality

- **gain**(v) = value of the game.
- **bias**(v) = measure of initial fluctuations.

Optimality equations $\mathscr{O}\mathscr{E}$

For every min vertex $v \in V_{\text{Min}}$,

- $g(v) = \min\{g(u) : (v,u) \in E\}$.
- $b(v) = \min\{r(v) - g(v) + b(u) : (u,v) \in E \text{ and } g(v) = g(u)\}$.

Positional optimal strategy

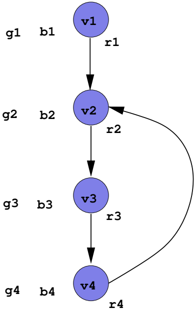
- Solution to $\mathscr{O}\mathscr{E}$ gives optimal value of the game.
- Solution to $\mathscr{O}\mathscr{E}$ provides *positional optimal strategy*.

Optimality equations – Contd.

Optimality equations

If solution for $\mathcal{OE}$ exists, then

- **optimal-cycle**(v) = Cycle reachable from vertex v, when players play optimal strategies.
- **gain**(v) = average of the **optimal-cycle**(v).
- **bias**(v) = optimal distance (fluctuations) from v to an arbitrary vertex in **optimal-cycle**(v).



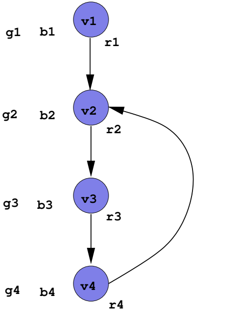

```
g1 = g2 = g3 = g4 = g
b1 = r1 - g + b2

b2 = r2 - g + b3
b3 = r3 - g + b4
b4 = r4 - g + b2
```

Optimality equations – Contd.

Optimality equations

If solution for $\mathcal{OE}$ exists, then

- **optimal-cycle**(v) = Cycle reachable from vertex v, when players play optimal strategies.
- **gain**(v) = average of the **optimal-cycle**(v).
- **bias**(v) = optimal distance (fluctuations) from v to an arbitrary vertex in **optimal-cycle**(v).



g1 = g2 = g3 = g4 = g

b1 = r1 − g + b2

b2 = r2 − g + b3
b3 = r3 − g + b4
b4 = r4 − g + b2

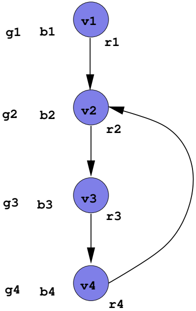
g = (r2 + r3 + r4)/3

Optimality equations – Contd.

Optimality equations

If solution for $\mathcal{OE}$ exists, then

- **optimal-cycle**(v) = Cycle reachable from vertex v, when players play optimal strategies.
- **gain**(v) = average of the **optimal-cycle**(v).
- **bias**(v) = optimal distance (fluctuations) from v to an arbitrary vertex in **optimal-cycle**(v).



g1 = g2 = g3 = g4 = g

b1 = r1 − g + b2

b2 = r2 − g + b3
b3 = r3 − g + b4
b4 = r4 − g + b2

g = (r2 + r3 + r4)/3

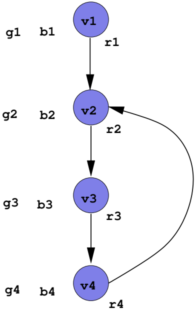
Optimality equations – Contd.

Optimality equations

If solution for $\mathscr{O}\mathscr{E}$ exists, then

- **optimal-cycle**(v) = Cycle reachable from vertex v, when players play optimal strategies.
- **gain**(v) = average of the **optimal-cycle**(v).
- **bias**(v) = optimal distance (fluctuations) from v to an arbitrary vertex in **optimal-cycle**(v).

Does it always exist?



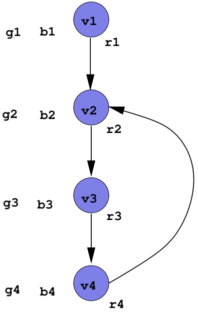
g1 = g2 = g3 = g4 = g

b1 = r1 − g + b2

b2 = r2 − g + b3
b3 = r3 − g + b4
b4 = r4 − g + b2

g = (r2 + r3 + r4)/3

Optimality equations – Contd.



g1 = g2 = g3 = g4 = g

b1 = r1 − g + b2

b2 = r2 − g + b3

b3 = r3 − g + b4

b4 = r4 − g + b2

g = (r2 + r3 + r4)/3

Optimality equations

If solution for $\mathscr{O}\mathscr{E}$ exists, then

- **optimal-cycle**(v) = Cycle reachable from vertex v, when players play optimal strategies.
- **gain**(v) = average of the **optimal-cycle**(v).
- **bias**(v) = optimal distance (fluctuations) from v to an arbitrary vertex in **optimal-cycle**(v).

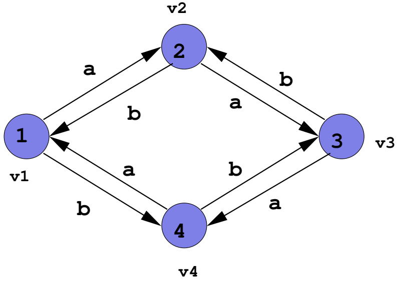
Does it always exist?

Yes!

1-player(Min) strategy improvement algorithm SIA_1

SIA_1

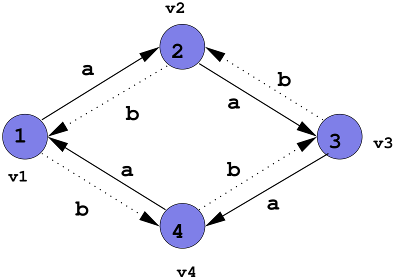
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

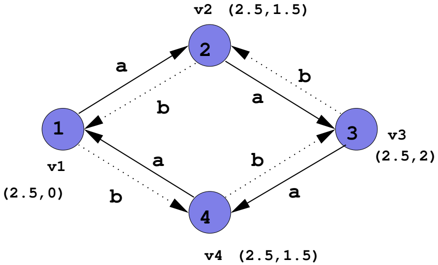
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

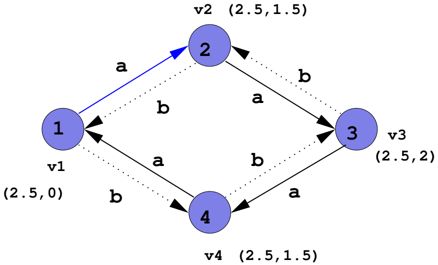
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

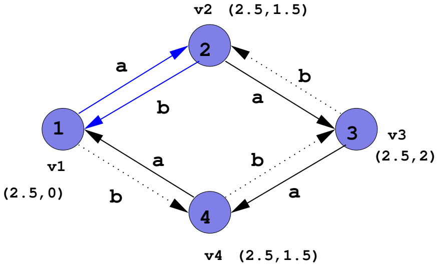
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

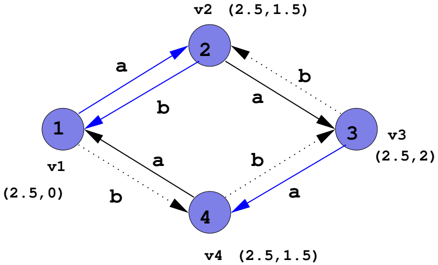
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

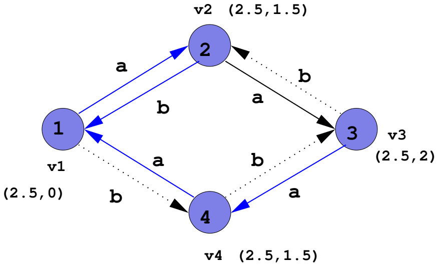
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

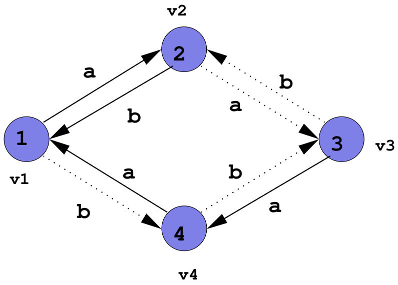
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

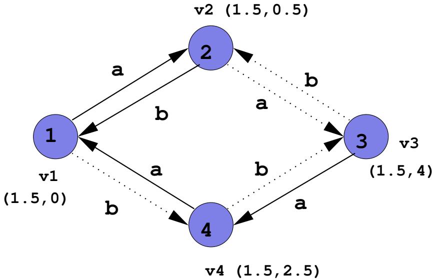
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

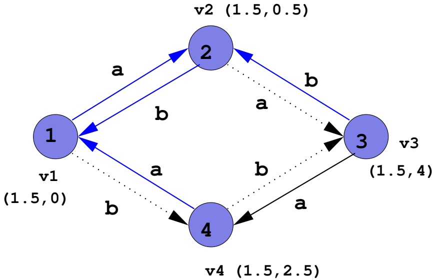
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g,b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

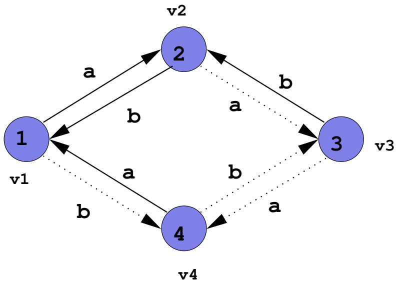
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

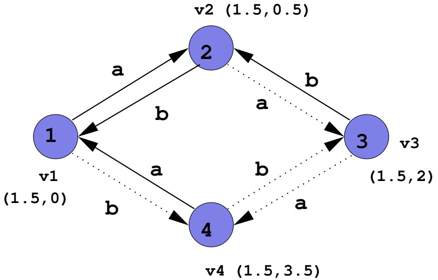
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

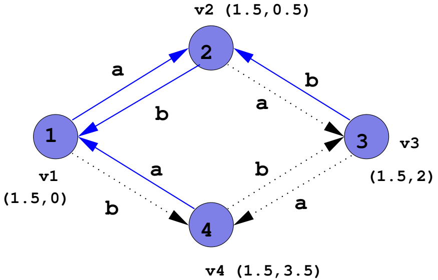
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

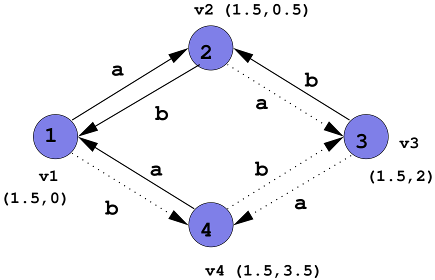
1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



1-player(Min) strategy improvement algorithm SIA_1

SIA_1

1. Fix an arbitrary positional strategy $\mu : V \to V$.
2. Compute gain and bias values for the subgraph restricted to μ.
3. Modify μ to new positional strategy μ', by choosing the successors with minimal (g, b) pair.
4. If $\mu = \mu'$, the stop and report μ as optimal positional strategy. Otherwise $\mu := \mu'$ and goto 2.



2-player strategy improvement algorithm SIA_2

SIA_2

1. Fix an arbitrary positional strategy $\chi : V_{\text{Max}} \to V$ for Max player.
2. Compute the best counter-strategy $\mu : V_{\text{Min}} \to V$ for Min player, against χ.
3. Compute the gain and bias values for the subgraph restricted to μ and χ.
4. Modify χ to new positional strategy χ', by choosing the successor with maximal (g, b) pair.
5. If $\chi = \chi'$, then stop and report μ and χ as the optimal strategies for Min and Max players. Otherwise $\chi := \chi'$ and goto 2.

On the complexity of strategy improvement algorithm

Complexity of strategy improvement algorithm

- Complexity is **not well understood**.
- In practice requires linear number of improvement steps.
- Each improvement step is polynomial in $|V|$.

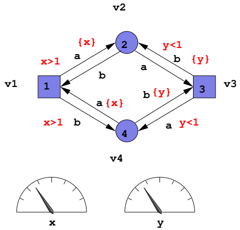
Average payoff games (continuous-time)

Game arena

- A finite directed graph $G = \{V, E\}, V = V_{\text{Max}} \cup V_{\text{Min}}$.
- A finite set of real valued variables – clocks C.
- The graph has **infinitely many configurations** $(V \times (\mathbb{R})^C)$.
- A **price-rate** (price per unit-time step) function $p : V \to \mathbb{Z}$

Optimization goals

- $\pi = v_1 \xrightarrow[t_1]{a_1} v_2 \xrightarrow[t_2]{a_2} v_3 \ldots$
- $A(\pi) = \lim_{n\to\infty} (1/n) \cdot \sum_{i=1}^{n} t_i$
- Average time games



Average time games

Optimality equations

- $g(s) = \max_{a \in A, t \in \mathbb{R}_+} \{g(s') \, : \, s \xrightarrow[t]{a} s'\}$
- $b(s) = \max_{a \in A, t \in \mathbb{R}_+} \{t - g(s) + b(s') \, : \, s \xrightarrow[t]{a} s', g(s) = g(s')\}$

Challenges

- Infinitely many states.
- Infinitely many (positional) strategies.

Solution

- Finitary abstraction (Thin region abstraction)
  - Symbolic representation of strategies.
  - Finite (positional) symbolic strategies.

Conclusion

Results

- Discrete-time Alternative proof of positional determinacy and alternative algorithm.
- Continuous time Elementary proof of determinacy and first algorithm to solve average time games.

Future Work

- Analyse the complexity of strategy improvement algorithm.
- Solving the average price games.
- Implementation of the algorithms into a toolkit for real-time controller synthesis.
- Experimental benchmarking of strategy improvement algorithm against [Zwick and Paterson' 96], [Karp'77], [Bouyer'04 al.] etc.

Appendix

Results on timed game automata

- **Reachability game** (constraint on **time**) on timed game automata [Asarin, E. and Maler, O. 1999]
- **Reachability game** (constraint on **price**) on timed game automata. [Alur et al. 2004],[Bouyer et al. 2004].
- **Average price game** for one player [Bouyer et al. 2004]