Nash equilibrium on weighted games

Kim Larsen and Ghassan Oreiby

22 october 2009

Introduction

- Formal verification methods assume the worst behavior for the adversary;
- Nash equilibrium assumes both player agrees on the strategy.

Introduction

- Formal verification methods assume the worst behavior for the adversary;
- Nash equilibrium assumes both player agrees on the strategy.

We are assuming:

- Pure strategies;
- Perfect information;
- Selfish players.

Weighted games

Definition

a weighted game is a graph (V_1, V_2, E) such that:

- $V = V_1 \cup V_2$ is the set of states for player 1 and player 2 resp.,
- $E \subseteq V \times \mathbb{N}_{>0} \times \mathbb{N}_{>0} \times V$ is the edge relation.

For $e = (q_1, c_1, c_2, q_2) \in E$, we note $Cost_i(e) = c_i$

Objectives

The objective for each player $i \in \{1, 2\}$ is to reach states labeled with $Goal_i$ with minimum cost.

Weighted games

Definition

A strategy for player i is $\lambda_i : E^* \rightarrow E$.

For a run $\rho = e_1 e_2 \dots e_j \dots$ induced by strategies λ_1 and λ_2, the payoff is defined as:

$$payoff_i(\lambda_1, \lambda_2) = \sum_{l \leq k} Cost_i(e_l)$$

with $Goal_i \in Lab(last(e_k))$

Nash equilibrium

Definition

Given a Game G with player 1 and player 2, we say that a game have a Nash equilibrium iff there exists $\lambda_1^\star$ and $\lambda_2^\star$ such that:

- for all λ_1, we have $payoff_1(\lambda_1^\star, \lambda_2^\star) \leq payoff_1(\lambda_1, \lambda_2^\star)$
- for all λ_2, we have $payoff_2(\lambda_1^\star, \lambda_2^\star) \leq payoff_2(\lambda_1^\star, \lambda_2)$

Nash equilibrium

Definition

Given a Game G with player 1 and player 2, we say that a game have a Nash equilibrium iff there exists $\lambda_1^\star$ and $\lambda_2^\star$ such that:

- for all λ_1, we have $\mathit{payoff}_1(\lambda_1^\star, \lambda_2^\star) \leq \mathit{payoff}_1(\lambda_1, \lambda_2^\star)$
- for all λ_2, we have $\mathit{payoff}_2(\lambda_1^\star, \lambda_2^\star) \leq \mathit{payoff}_2(\lambda_1^\star, \lambda_2)$

A Nash equilibrium point (c_1, c_2) is called dominant iff for all Nash equilibrium points (c_1', c_2') we have $c_1 \leq c_1'$ or $c_2 \leq c_2'$.

Nash equilibrium

Definition

Given a Game G with player 1 and player 2, we say that a game have a Nash equilibrium iff there exists $\lambda_1^\star$ and $\lambda_2^\star$ such that:

- for all λ_1, we have $payoff_1(\lambda_1^\star, \lambda_2^\star) \leq payoff_1(\lambda_1, \lambda_2^\star)$
- for all λ_2, we have $payoff_2(\lambda_1^\star, \lambda_2^\star) \leq payoff_2(\lambda_1^\star, \lambda_2)$

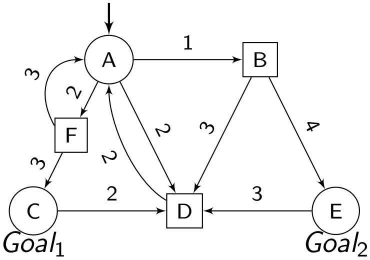
A Nash equilibrium point (c_1, c_2) is called dominant iff for all Nash equilibrium points (c_1', c_2') we have $c_1 \leq c_1'$ or $c_2 \leq c_2'$.

Decision problem

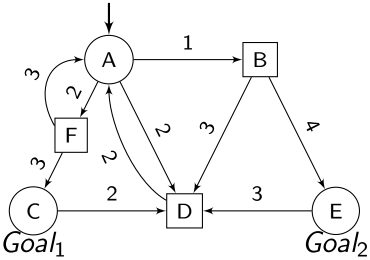
Input: a Weighted game G and a bound $B = (b_1, b_2)$;

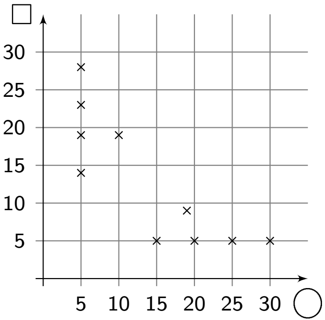
Output: True iff there exists a Nash equilibrium point (c_1, c_2) such that $c_1 \leq b_1$ and $c_2 \leq b_2$.

Example



Example





Hardness

Subset sum problem

Input $S = \{m_1, m_2, \ldots, m_n\}$ and m with $(m_i)_{i \in \{1 \ldots n\}}, m \in \mathbb{N}$;

Output True iff $\exists S' \subseteq S$ s.t. $\sum_{s \in S'} s = m$

Lemma

Deciding the Nash equilibrium problem on weighted games is *NP*-hard.

Hardness

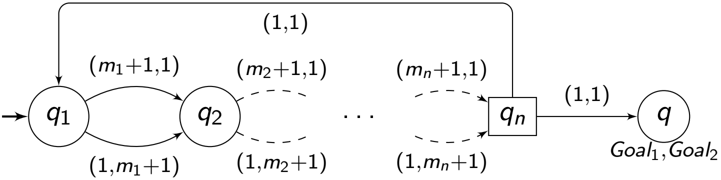
Subset sum problem

Input $S = \{m_1, m_2, \ldots, m_n\}$ and m with $(m_i)_{i\in\{1\ldots n\}}, m \in \mathbb{N}$;

Output True iff $\exists S' \subseteq S$ s.t. $\sum_{s\in S'} s = m$

Lemma

Deciding the Nash equilibrium problem on weighted games is *NP*-hard.



$$B = (m + n + 1, \textstyle\sum_{s\in S} s + n + 1 - m)$$

Algorithm

$Q^0(v) \leftarrow \{(0,0)\}$ if $\{Goal_1, Goal_2\} \subseteq Lab(v)$;
$Q^0(v) \leftarrow \{(0,-)\}$ if $Goal_1 \in Lab(v)$;
$Q^0(v) \leftarrow \{(-,0)\}$ if $Goal_2 \in Lab(v)$;
$Q^0(v) \leftarrow \{\}$ otherwise;
//Main loop
while $Q^i \neq Q^{i-1}$ **do**
  $Q^{i-1} \leftarrow Q^i$;
  foreach $v \in V$ **do**
    foreach $e \in E$ *and* $src(e) = v$ **do**
      foreach $(c_1, c_2) \in Q^{i-1}(dst(e))$ **do**
        if $Cost_1(e) + c_1 \leq Reach_1(E \setminus \{e\}, v)$ *and* $Cost_2(e) + c_2 \leq Reach_2(E \setminus \{e\}, v)$ **then**
          $Q^i(v) \leftarrow Q^i(v) \cup \{(Cost_1(e) + c_1, Cost_2(e) + c_2)\}$

Algorithm

$Q^0(v) \leftarrow \{(0,0)\}$ if $\{Goal_1, Goal_2\} \subseteq Lab(v)$;
$Q^0(v) \leftarrow \{(0,-)\}$ if $Goal_1 \in Lab(v)$;
$Q^0(v) \leftarrow \{(-,0)\}$ if $Goal_2 \in Lab(v)$;
$Q^0(v) \leftarrow \{\}$ otherwise;
//Main loop
while $Q^i \neq Q^{i-1}$ **do**
    $Q^{i-1} \leftarrow Q^i$;
    foreach $v \in V$ **do**
        foreach $e \in E$ *and* $src(e) = v$ **do**
            foreach $(c_1, c_2) \in Q^{i-1}(dst(e))$ **do**
                if $Cost_1(e) + c_1 \leq Reach_1(E \setminus \{e\}, v)$ *and* $Cost_2(e) + c_2 \leq Reach_2(E \setminus \{e\}, v)$ **then**
                    $Q^i(v) \leftarrow Q^i(v) \cup \{(Cost_1(e) + c_1, Cost_2(e) + c_2)\}$

Remark

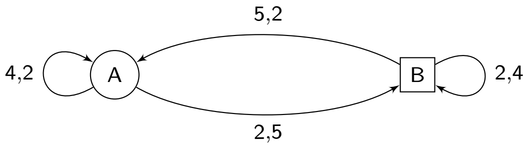
The above algorithm runs in exponential time, but it can be used to extract the Nash equilibrium strategy.

On going work

- Tighter complexity bounds;
- Mean-payoff objectives;

On going work

- Tighter complexity bounds;
- Mean-payoff objectives;
  - Plays are infinite runs, each player is required to minimize the mean-payoff;
  - Infinite memory strategies needed.



Mean payoff game with a NE of $(3, 3)$