CS295 Introduction to Algorithmic Game Theory

Instructor: Ioannis Panageas Scribed by: David Zhu, Dylan Zou

Lecture 6. Potential and Congestion Games

1 Introduction to Potential Games

1.1 Definition of Potential Games

Definition 1.1 (Normal Form Games) *A normal form game is specified by:*

- *Set of n players [n] = {1, 2, ..., n}*
- *For each player $1 \leq i \leq n$ a set of strategies s_i and a utility $u_i : \times_{j=1}^{n} s_j \rightarrow \mathbb{R}$*
- *Set of all strategies $S = s_1 \times s_2 \times ... \times s_n$*

Definition 1.2 ((Exact) Potential Games) *A potential game exists if there is a potential function $\Phi : S \rightarrow \mathbb{R}$ such that for all agents i with strategy s_i,*

$$\Phi(s_i, s_{-i}) - \Phi(s'_i, s_{-i}) = u_i(s_i, s_{-i}) - u_i(s'_i, s_{-i})$$

where s_{-i} denotes the strategies of the other players.

Definition 1.3 (Weighted Potential Games) *A weighted potential game exists if there is a potential function $\Phi : S \rightarrow \mathbb{R}$ such that for all agents i with strategy s_i,*

$$\Phi(s_i, s_{-i}) - \Phi(s'_i, s_{-i}) = w_i \cdot (u_i(s_i, s_{-i}) - u_i(s'_i, s_{-i}))$$

for some $w_i > 0$.

Definition 1.4 (Ordinal Potential Games) *An ordinal potential game exists if there is a potential function $\Phi : S \rightarrow \mathbb{R}$ such that for all agents i with strategy s_i,*

$$\Phi(s_i, s_{-i}) - \Phi(s'_i, s_{-i}) > 0 \textit{ iff } u_i(s_i, s_{-i}) - u_i(s'_i, s_{-i}) > 0$$

Potential and Weighted Potential Games are subsets of Ordinal Potential Games.

Example 1.1 (Finding an Exact Potential Function)

5, 2	-1,-2
-5, -4	1,4

$$\Rightarrow$$

4	0
-6	2

Let A be the matrix of utilities on the left and B be the matrix of the potentials on the right. The potential function is relative to itself, or that we can start the potential from any number. In this example, we start at $B_{(1,2)} = 0$;

$$
\begin{aligned}
B_{(1,2)} &= 0 \\
B_{(1,1)} &= B_{(1,2)} + u_2(1,1) - u_2(1,2) = 4 \\
B_{(2,2)} &= B_{(1,2)} + u_1(2,2) - u_1(1,2) = 2 \\
B_{(2,1)} &= B_{(2,2)} + u_2(2,1) - u_2(2,2) = -6
\end{aligned}
$$

1.2 Properties of Potential Games

In the previous example, note that calculating $B_{(2,1)}$ from $B_{(1,1)}$ yields the same answer; if $A_{(2,1)} = (-5, -3)$ for example, there would be no exact potential function that accurately captures the incentive for both agents.

From this, we see that 1) the potential function is linear-shift-invariant, and 2) many games with pure Nash equilibrium are not necessarily potential games.

In general, capturing the incentive for any individual player to deviate (their change in utility) into a single, global function Φ can be very useful.

Lemma 1.2 *Let G be a Potential Game. G has a pure Nash equilibrium.*

Proof: Let s^* be a pure strategy that maximizes Φ, or that $\Phi(s_i^*, s_{-i}) \geq \Phi(s_i', s_{-i})$ for all s_i'. From the definition of the potential function,

$$\Phi(s_i^*, s_{-i}) - \Phi(s_i', s_{-i}) = u_i(s_i^*, s_{-i}) - u_i(s_i', s_{-i}) \geq 0$$

Since $u_i(s_i^*, s_{-i}) - u_i(s_i', s_{-i}) \geq 0$, s_i^* is a pure Nash equilibrium. ■

1.3 Example Algorithm for Finding the Pure NE in a Potential Game

Algorithm 1 Greedy Algorithm for Calculating the Pure Nash Equilibrium

```
Require: Ordinal Potential Function exists.
  init $s^0$ randomly
  while $\exists i, s_i'$ such that $u_i(s_i') > u_i(s_i)$ do
      $s^{t+1} \leftarrow (s_i', s_{-i}^t)$
      $t \leftarrow t + 1$
  end while
```

Lemma 1.3 *Algorithm 1 results in a pure Nash equilibrium.*

Proof: Represent the set of strategies as a directed graph G with $|S|$ vertices and an edge from vertex $s_1 \to$ vertex s_2 if s_1, s_2 differ in one agent i only with $u_i(s_2) > u_i(s_1)$.

Because $\text{sign}(u_i(s_2) - u_i(s_1)) = \text{sign}(\Phi(s_2) - \Phi(s_1))$, it follows that $\Phi(s_2) > \Phi(s_1)$. This means the potential function strictly increases along the walk; since the graph is finite, G must be acyclic.

Since G is acyclic, G must have a sink (a vertex with outdegree 0). This sink is a pure Nash equilibrium, as there is no deviation in strategy that increases an agent's utility. ■

2 Congestion Games

2.1 Definition of Congestion Game

Congestion games can be used to model various settings such as traffic networks and resource allocation. The defining property of the congestion game is the cost function on each edge (ie. a road in the context of traffic networks) c_e, which increases as the number of players on that edge increases.

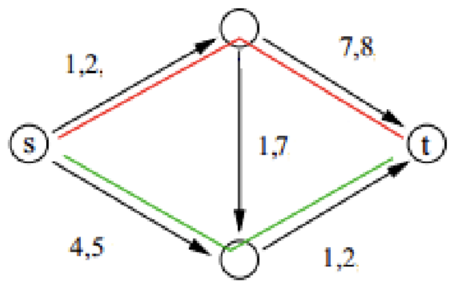
Definition 2.1 (Congestion Games) *Formally, a congestion game is defined by:*

- *Set of n players*
- *Set of edges/facilities/bins E*
- *Set of strategies $S_i \subset 2^E$ for player i*
- *Cost function $c_e : \{1, 2, ..., n\} \to \mathbb{R}$*

Define:

- *Number of players $l_e(s)$ who use edge e*
- *Cost function $c_i(s) = \sum\limits_{e \in E} c_e(l_e)$ of player i*

Example 2.1 (Example of a Congestion Game)



For this game:

$n = \{1, 2\}$ (red, green)
E are the edges of the network.
S_i is all $s - t$ paths.
c_e on edges.

2.2 Congestion Games as Potential Games

Theorem 2.2 (Rosenthal '73) *Congestion Games are Potential Games.*

Proof: Define agent i's strategies s_i and s'_i, and s the set of strategies for all players with $s_i = s_i$ and s' the set of strategies with $s_i = s'_i$.

Let $\Phi(s) = \sum_{e \in E} \sum_{j=1}^{l_e(s)} c_e(j)$, the sum of the total cost for all players.

For an agent i's strategy s_i and s'_i,

$$\Phi(s) = \sum_{e \in s \cap s'} \sum_{l_e(s)} c_e(l) + \sum_{e \in s_i \backslash s'_i} \sum_{l_e(s)} c_e(l) + \sum_{e \in s'_i \backslash s_i} \sum_{l_e(s)} c_e(l) + \sum_{e \in E \backslash (s \cup s')} \sum_{l_e(s)} c_e(l)$$

$$\Phi(s') = \sum_{e \in s \cap s'} \sum_{l_e(s')} c_e(l) + \sum_{e \in s_i \backslash s'_i} \sum_{l_e(s')} c_e(l) + \sum_{e \in s'_i \backslash s_i} \sum_{l_e(s')} c_e(l) + \sum_{e \in E \backslash (s \cup s')} \sum_{l_e(s')} c_e(l)$$

Observe: because only a single agent i's strategy changes, for $e \in s'_i \backslash s_i$, $c_e(l_e(s')) = c_e(l_e(s)) + 1$.
Similarly, for $e \in s_i \backslash s'_i$, $c_e(l_e(s)) = c_e(l_e(s')) + 1$.
Additionally, the first and fourth term in $\Phi(s)$ and $\Phi(s')$ are identical.

Therefore,

$$\begin{aligned}
\Phi(s) - \Phi(s') &= \Big(\sum_{e \in s_i \backslash s'_i} \sum_{l_e(s)} c_e(l) + \sum_{e \in s'_i \backslash s_i} \sum_{l_e(s)} c_e(l) \Big) - \Big(\sum_{e \in s_i \backslash s'_i} \sum_{l_e(s')} c_e(l) + \sum_{e \in s'_i \backslash s_i} \sum_{l_e(s')} c_e(l) \Big) \\
&= \sum_{e \in s_i \backslash s'_i} c_e(l(s)) - \sum_{e \in s'_i \backslash s_i} c_e(l(s'))
\end{aligned}$$

Let the utility $u_i(s_i)$ be the sum of the costs for player i:

$$\begin{aligned}
u_i(s_i) = \sum_{e \in s_i} c_e(l_e(s)) &= \sum_{e \in s_i \cap s'_i} c_e(l_e(s)) + \sum_{e \in s_i \backslash s'_i} c_e(l_e(s)) \\
u_i(s'_i) &= \sum_{e \in s_i \cap s'_i} c_e(l_e(s')) + \sum_{e \in s'_i \backslash s_i} c_e(l_e(s')) \\
\rightarrow u_i(s_i) - u_i(s'_i) &= \sum_{e \in s_i \backslash s'_i} c_e(l_e(s)) - \sum_{e \in s'_i \backslash s_i} c_e(l_e(s')) \\
&= \Phi(s) - \Phi(s')
\end{aligned}$$

■

Remark 2.2 *Monderer and Shapley (1996) showed the converse: potential games are congestion games as well, or that potential games and congestion games are isomorphic.*

Remark 2.3 *Since all congestion games are potential games, any congestion game must have a pure Nash equilibrium. However, the Nash equilibrium does not necessarily minimize the social welfare!*

2.3 Algorithm to Compute NE in Symmetric Network Congestion Games

Definition 2.4 (Min-cost Flow) *Given a graph $G(V, E)$ with source s and sink t, each edge (u, v) in G has a:*

- *Capacity* $c(u, v)$
- *Cost per flow unit* $a(u, v)$
- *Flow* $f(u, v)$

Min-cost flow is defined as the cheapest way of sending some flow d from s to t:

$$
\begin{gathered}
\min \sum_{e:(u,v)} f(u,v) \cdot a(u,v) \\
s.t. \quad f(u,v) \leq c(u,v) \quad \forall (u,v) \in E \\
f(u,v) = -f(v,u) \quad \forall (u,v) \in E \\
\sum_{w} f(u,w) = 0 \quad \forall u \neq s,t \\
\sum_{w} f(s,w) = \sum_{w} f(w,t) = d
\end{gathered}
$$

Elaboration on the constraints:

- $f(u,v) \leq c(u,v)$ *is a capacity constraint that ensures the flow on any edge* (u,v) *does not exceed the capacity of the edge.*
- $f(u,v) = -f(v,u)$ *maintains that any flow for* (u,v) *is the equivalent to the negation of the reversed flow.*
- $\sum_{w} f(u,w) = 0 \quad \forall u \neq s,t$ *ensures the conservation of flow, since any flow that passes into a node should also pass out through it, with the exception of the source and sink nodes.*
- $\sum_{w} f(s,w) = \sum_{w} f(w,t) = d$ *is also flow conservation, guaranteeing that all flow leaving the source ends in sink.*

A linear program using the above constraints can be used to compute the min-cost flow.

Definition 2.5 (Symmetric Network) *A symmetric network is defined as a network where all players have the same set of paths and strategies, and thus have the same source and sink. The symmetric network property is a key assumption for this algorithm.*

Assuming symmetry, Algorithm 2 reduces the congestion game to a min-cost flow problem, which can then be solved using linear programming. First, create a graph with vertices identical to the congestion game graph, but no edges. Then, for each edge $e \leftarrow (u,v)$ in the original graph, add n parallel edges (u,v) to the new graph with capacity 1 and costs $c_e(1), c_e(2), ..., c_e(n)$.

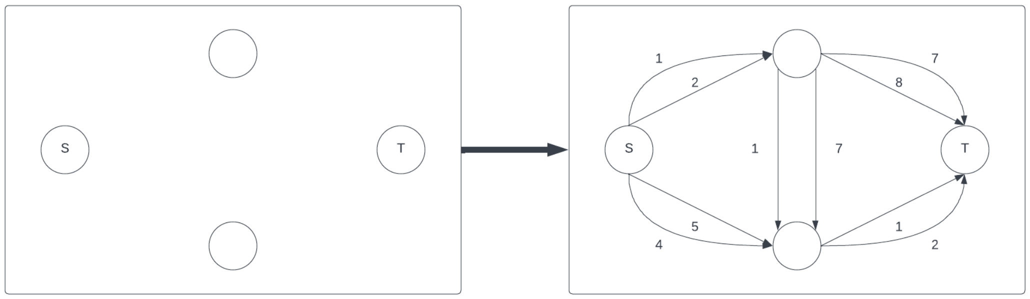
Example 2.3 (Example Reduction of Congestion Game to Min-cost Flow Problem)
The following figure is an example reduction of the congestion game shown in Example 2.1. The capacity between any vertices u and v is the number of edges going from u to v in the diagram. Each edge is labeled with its cost, from which cost per flow unit can be determined.

Algorithm 2 Algorithm to Compute NE in Symmetric Network Congestion Games

Input: Congestion Game Graph $G(V, E)$, Cost function c for each edge

Output: Minimized potential of Congestion Game

```
initialize $G'(V, E')$ with same vertices as $G$        ▷ $E'$ is initially empty
for $e \leftarrow (u, v) \in E$ do
    for each cost $c_e(i)$ in $c_e$ do
        add an edge $(u, v)$ to $E'$ with 1 capacity and cost $c_e(i)$
    end for
end for
Use $G'$ as the graph for the min-cost flow problem described in Definition 2.5.
return linear program solution for min-cost flow
```



References

[1] Marden, R. Jason. https://web.ece.ucsb.edu/∼ jrmarden/ *Lecture 14: Congestion and Potential Games.*

[2] Monderer and Shapley (1996). *Potential Games.* Games and Economic Behavior