STACKELBERG GAMES

Dipayan Mitra

Department of Electrical and Computer Engineering, University of Toronto

Outlines

INTRODUCTION

Two player finite game

Two player finite game

P1 is powerful enough to impose his strategy on the other player

Two player finite game

P1 is powerful enough to impose his strategy on the other player $\longleftarrow$ Leader

Two player finite game

P1 is powerful enough to impose his strategy on the other player $\longleftarrow$ Leader

P2 matches optimally with P1's strategy

Two player finite game

P1 is powerful enough to impose his strategy on the other player $\longleftarrow$ Leader

P2 matches optimally with P1's strategy $\longleftarrow$ Follower

Two player finite game

P1 is powerful enough to impose his strategy on the other player $\longleftarrow$ Leader

P2 matches optimally with P1's strategy $\longleftarrow$ Follower

Such game is called Stackelberg game

Two player finite game

P1 is powerful enough to impose his strategy on the other player $\longleftarrow$ Leader

P2 matches optimally with P1's strategy $\longleftarrow$ Follower

Such game is called Stackelberg game $\longleftarrow$ Proposed by H. von Stackelberg (1934)

Examples

Government setting up policies for spectrum auction

Examples

Government setting up policies for spectrum auction $\longleftarrow$ Leader

Examples

Government setting up policies for spectrum auction $\longleftarrow$ Leader

Participating companies coming up with strategies based on the government's policy

Examples

Government setting up policies for spectrum auction $\longleftarrow$ Leader

Participating companies coming up with strategies based on the government's policy $\longleftarrow$ Follower

Examples

Antivirus company releasing update

Antivirus company releasing update $\longleftarrow$ Leader

Examples

Antivirus company releasing update $\longleftarrow$ Leader

Malicious agents' strategy to exploit the loopholes

Examples

Antivirus company releasing update $\longleftarrow$ Leader

Malicious agents' strategy to exploit the loopholes $\longleftarrow$ Follower

PURE STRATEGY STACKELBERG EQUILIBRIA

Two player finite game

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the NE?

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the NE?

$\implies j^* = 2$ and $k^* = 2$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the NE?

$\implies j^* = 2$ and $k^* = 2$

$\implies$ Associated cost $= (1, 0)$

Two player finite game

Two player finite game

P1 is the leader and P2 is the follower

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P1?

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P1?

$\implies j^* = 1$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P1?

$\implies j^* = 1$

What is the best strategy for P2, knowing $j^* = 1$?

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P1?

$\implies j^* = 1$

What is the best strategy for P2, knowing $j^* = 1$?

$\implies k^* = 1$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P1?

$\implies j^* = 1$

What is the best strategy for P2, knowing $j^* = 1$?

$\implies k^* = 1$

Associated cost $= (0, -1)$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P1?

$\implies j^* = 1$

What is the best strategy for P2, knowing $j^* = 1$?

$\implies k^* = 1$

Associated cost $= (0, -1)$

Note: The cost is lesser than the NE cost (for both the players).

Is it always the case?

Is it always the case?

$\longrightarrow$ No!

Pure Strategy Stackelberg Equilibria

Same cost matrices but P2 as leader and P1 as follower

Pure Strategy Stackelberg Equilibria

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$\implies k^* = 3$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$\implies k^* = 3$

What is the best strategy for P1, knowing $k^* = 3$?

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$\implies k^* = 3$

What is the best strategy for P1, knowing $k^* = 3$?

$\implies j^* = 1$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$\implies k^* = 3$

What is the best strategy for P1, knowing $k^* = 3$?

$\implies j^* = 1$

Associated cost $(-1.5, -\frac{2}{3})$

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$\implies k^* = 3$

What is the best strategy for P1, knowing $k^* = 3$?

$\implies j^* = 1$

Associated cost $(-1.5, -\frac{2}{3})$

Note:

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$\implies k^* = 3$

What is the best strategy for P1, knowing $k^* = 3$?

$\implies j^* = 1$

Associated cost $(-1.5, -\frac{2}{3})$

Note:

$\implies$ The cost is lesser than the NE cost for only P2 (from 0 to $-\frac{2}{3}$)

$$A = \begin{bmatrix} 0 & 2 & 1.5 \\ 1 & 1 & 3 \\ -1 & 2 & 2 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & -\frac{2}{3} \\ 2 & 0 & 1 \\ 0 & 1 & -\frac{1}{2} \end{bmatrix}$$

What is the safest strategy for P2?

$\implies k^* = 3$

What is the best strategy for P1, knowing $k^* = 3$?

$\implies j^* = 1$

Associated cost $(-1.5, -\frac{2}{3})$

Note:

$\implies$ The cost is lesser than the NE cost for only P2 (from 0 to $-\frac{2}{3}$)

$\implies$ Cost increases for P1 (from 1 to 1.5)

Does Stackelberg game setup always reduce the cost for either of the players, when compared with NE?

Does Stackelberg game setup always reduce the cost for either of the players, when compared with NE?

$\implies$ No!

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

What is the NE?

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

What is the NE?

$\Longrightarrow$ NE at $(2, 3)$

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

What is the NE?

$\implies$ NE at $(2, 3)$

$\implies$ Associated cost $(-1, -1)$

Pure Strategy Stackelberg Equilibria

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

Choose P1 as leader

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

Choose P1 as leader

$\implies j^* = 1$

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

Choose P1 as leader

$\implies j^* = 1$

$\implies k^* = 1$ or 2

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

Choose P1 as leader
$\implies j^* = 1$
$\implies k^* = 1$ or 2
$\implies$ Associated cost $= (0, 0)$ or $(1, 0)$

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 2 & 2 & -1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}$$

Choose P1 as leader

$\implies j^* = 1$

$\implies k^* = 1$ or 2

$\implies$ Associated cost $= (0, 0)$ or $(1, 0)$

$\implies$ Cost increases for both P1 and P2

Notations:

$\mathscr{G}(I, \Omega_i, J_i)$, $I = 1, 2$

P1: $u_1 \in \Omega_1$, $|\Omega_1| = m \leftarrow$ Set of indices $M1 = \{1, \ldots, j, \ldots, m\}$

P2: $u_2 \in \Omega_2$, $|\Omega_2| = n \leftarrow$ Set of indices $M2 = \{1, \ldots, k, \ldots, n\}$

P1: Cost function $J_1 : \Omega_1 \times \Omega_2 \to \mathbb{R}$

P2: Cost function $J_2 : \Omega_1 \times \Omega_2 \to \mathbb{R}$

P1 is leader and P2 is the follower

Definition:

In a two-player finite game, with P1 being the leader choosing $u_1 \in \Omega_1$ as his strategy, optimal response of P2 or $BR_2(u_1) \subset \Omega_2$ is defined as,

$$BR_2(u_1) = \{u_2^* \in \Omega_2 : J_2(u_1, u_2^*) \leq J_2(u_1, u_2), \forall u_2 \in \Omega_2\}$$

Stackelberg Game Formulation (in Pure Strategy)

What is the Stackelberg cost for P1?

Definition:
In a two-player finite game, with P1 being the leader, $u_1 \in \Omega_1$ is the Stackelberg equilibrium strategy for P1,

$$J_1^* = \min_{u_1 \in \Omega_1} \max_{u_2 \in BR_2(u_1^*)} J_1(u_1, u_2)$$

Similar formulation can be obtained choosing P2 as leader

Definition:
P1 being the leader with optimal strategy $u_1^* \in \Omega_1$ and P2 with the optimal strategy $u_2^* \in BR_2(u_1^*)$ Stackelburg solution of the game is defined by $u^* = (u_1^*, u_2^*)$.

Definition:
P1 being the leader with optimal strategy $u_1^* \in \Omega_1$ and P2 with the optimal strategy $u_2^* \in BR_2(u_1^*)$ Stackelburg solution of the game is defined by $u^* = (u_1^*, u_2^*)$.

Note:
$\Longrightarrow$ The associated cost for P1: $J_1^* = J_1(u_1^*, u_2^*)$

Definition:
P1 being the leader with optimal strategy $u_1^* \in \Omega_1$ and P2 with the optimal strategy $u_2^* \in BR_2(u_1^*)$ Stackelburg solution of the game is defined by $u^* = (u_1^*, u_2^*)$.

Note:
$\Longrightarrow$ The associated cost for P1: $J_1^* = J_1(u_1^*, u_2^*)$
$\Longrightarrow$ The associated cost for P2: $J_2^* = J_2(u_1^*, u_2^*)$

Definition:
P1 being the leader with optimal strategy $u_1^* \in \Omega_1$ and P2 with the optimal strategy $u_2^* \in BR_2(u_1^*)$ Stackelburg solution of the game is defined by $u^* = (u_1^*, u_2^*)$.

Note:
$\Longrightarrow$ The associated cost for P1: $J_1^* = J_1(u_1^*, u_2^*)$
$\Longrightarrow$ The associated cost for P2: $J_2^* = J_2(u_1^*, u_2^*)$
$\Longrightarrow$ Cost for P1 in NE: J_1^{NE}.

Theorem:
If $BR_2(u_1)$ is a singleton for $u_1 \in \Omega_1$, then,

$$J_1^* \leq J_1^{NE}$$

Theorem:
If $BR_2(u_1)$ is a singleton for $u_1 \in \Omega_1$, then,

$$J_1^* \leq J_1^{NE}$$

Proof. Can be proven by contradiction.

MIXED STRATEGY STACKELBERG EQUILIBRIA

Recall from the pure strategy case

Recall from the pure strategy case

$\Longrightarrow u_1 \in \Omega_1$ and $u_2 \in \Omega_2$ both are finite set. Also $BR_2(u_1) \in \Omega_2$ is also a finite set for all $u_1 \in \Omega_1$

Recall from the pure strategy case

$\Longrightarrow u_1 \in \Omega_1$ and $u_2 \in \Omega_2$ both are finite set. Also $BR_2(u_1) \in \Omega_2$ is also a finite set for all $u_1 \in \Omega_1 \leftarrow$ A Stackelberg equilibria would always exist in pure strategy

Recall from the pure strategy case

$\Longrightarrow u_1 \in \Omega_1$ and $u_2 \in \Omega_2$ both are finite set. Also $BR_2(u_1) \in \Omega_2$ is also a finite set for all $u_1 \in \Omega_1 \leftarrow$ A Stackelberg equilibria would always exist in pure strategy

$\Longrightarrow$ P1 declares his strategy beforehand

Recall from the pure strategy case

$\Longrightarrow u_1 \in \Omega_1$ and $u_2 \in \Omega_2$ both are finite set. Also $BR_2(u_1) \in \Omega_2$ is also a finite set for all $u_1 \in \Omega_1 \leftarrow$ A Stackelberg equilibria would always exist in pure strategy

$\Longrightarrow$ P1 declares his strategy beforehand

Why do we need mixed strategy?

Recall from the pure strategy case

$\Longrightarrow u_1 \in \Omega_1$ and $u_2 \in \Omega_2$ both are finite set. Also $BR_2(u_1) \in \Omega_2$ is also a finite set for all $u_1 \in \Omega_1 \leftarrow$ A Stackelberg equilibria would always exist in pure strategy

$\Longrightarrow$ P1 declares his strategy beforehand

Why do we need mixed strategy?

$\Longrightarrow$ Turns out, P1 can reduce his cost by choosing mixed strategy over pure strategy

$\implies \mathscr{G}(I, \Omega_i, J_i)$, $I = 1, 2$

$\implies$ Set of actions for P1: Ω_1, $|\Omega_1| = m \leftarrow$ Set of indices $M1 = \{1, \ldots, j, \ldots, m\}$.

$\implies$ Set of actions for P2: Ω_2, $|\Omega_2| = n \leftarrow$ Set of indices $M2 = \{1, \ldots, k, \ldots, n\}$.

$\implies$ Probability of P1 selecting j^{th} action: x_j, $\sum_{j=1}^{m} x_j = 1$ and $0 \leq x_j \leq 1$.

$\implies$ Probability of P2 selecting k^{th} action: y_k, $\sum_{k=1}^{n} y_k = 1$ and $0 \leq y_k \leq 1$.

$\implies \mathbf{x} = [x_1, \ldots, x_j, \ldots, x_m]^T$ and $\mathbf{y} = [y_1, \ldots, y_k, \ldots, y_n]^T$

$\implies$ Mixed strategy set for P1: $\Delta_1 = \{\mathbf{x} \in \mathbb{R}^m | 0 \leq x_j \leq 1 \forall j; \sum_{j=1}^{m} x_j = 1\}$.

$\implies$ Mixed strategy set for P2: $\Delta_2 = \{\mathbf{Y} \in \mathbb{R}^n | 0 \leq y_k \leq 1, \forall k; \sum_{k=1}^{n} y_k = 1\}$.

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix}$$

$\Longrightarrow$ Associated cost in NE: $(1, \frac{1}{2})$

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix}$$

$\Longrightarrow$ Associated cost in NE: $(1, \frac{1}{2})$

$\Longrightarrow$ P1 mixes his strategy $\mathbf{x}^* = [\frac{1}{2}, \frac{1}{2}]^T$.

Mixed strategy for P2 would be:

$$\begin{aligned}
\mathbf{y}^* &= \arg\min_{y \in \Delta_2} \mathbf{x}^* B \mathbf{y} && (1)\\
&= \arg\min_{y \in \Delta_2} \left[\frac{1}{2}, \frac{1}{2}\right] \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix} \mathbf{y} && (2)\\
&= \arg\min_{y \in \Delta_2} \left[\frac{3}{4}, \frac{3}{4}\right] \mathbf{y} && (3)\\
& && (4)
\end{aligned}$$

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix}$$

$\Longrightarrow$ Associated cost in NE: $(1, \frac{1}{2})$

$\Longrightarrow$ P1 mixes his strategy $\mathbf{x}^* = [\frac{1}{2}, \frac{1}{2}]^T$.

$\Longrightarrow$ Expected cost in mixed strategy: $(\frac{1}{2}, \frac{3}{4})$

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix}$$

$\Longrightarrow$ Associated cost in NE: $(1, \frac{1}{2})$

$\Longrightarrow$ P1 mixes his strategy $\mathbf{x}^* = [\frac{1}{2}, \frac{1}{2}]^T$.

$\Longrightarrow$ Expected cost in mixed strategy: $(\frac{1}{2}, \frac{3}{4})$

$\Longrightarrow$ Mixed strategy led to reduction in P1's cost

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix}$$

$\Longrightarrow$ Associated cost in NE: $(1, \frac{1}{2})$

$\Longrightarrow$ P1 mixes his strategy $\mathbf{x}^* = [\frac{1}{2}, \frac{1}{2}]^T$.

$\Longrightarrow$ Expected cost in mixed strategy: $(\frac{1}{2}, \frac{3}{4})$

$\Longrightarrow$ Mixed strategy led to reduction in P1's cost

Does a mixed strategy Stackelberg equilibria always exist?

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{bmatrix}$$

$\implies$ Associated cost in NE: $(1, \frac{1}{2})$

$\implies$ P1 mixes his strategy $\mathbf{x}^* = [\frac{1}{2}, \frac{1}{2}]^T$.

$\implies$ Expected cost in mixed strategy: $(\frac{1}{2}, \frac{3}{4})$

$\implies$ Mixed strategy led to reduction in P1's cost

Does a mixed strategy Stackelberg equilibria always exist?

Not really!

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{3} \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{3} \end{bmatrix}$$

$\implies$ There is no Stackelberg equilibrium in mixed strategies.

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{3} \end{bmatrix}$$

$\Longrightarrow$ There is no Stackelberg equilibrium in mixed strategies.

$\Longrightarrow$ There exists sub-optimal mixed strategy.

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{3} \end{bmatrix}$$

$\Longrightarrow$ There is no Stackelberg equilibrium in mixed strategies.

$\Longrightarrow$ There exists sub-optimal mixed strategy.

If Stackelberg mixed strategy exists, how is it related to NE?

Theorem:
For a two player finite game, if the Stackelberg mixed strategy equilibria exists, then the following holds,

$$J_1' \leq J_1^{NE}$$

Takeaways:

$\Longrightarrow$ In a two player finite game, mixed strategy NE would always exist. Whereas, a mixed strategy Stackelberg equilibrium might not exist.

Takeaways:

$\Longrightarrow$ In a two player finite game, mixed strategy NE would always exist. Whereas, a mixed strategy Stackelberg equilibrium might not exist.

$\Longrightarrow$ If the mixed strategy Stackelberg equilibria exist, the player would incur lower cost than that of the NE.

Where can we apply these?

What's now?

Where can we apply these?

$\Longrightarrow$ Federated learning!

FEDERATED LEARNING

Is there a way to train a model without centralizing users' data, i.e. ensuring 'privacy by default'?

Is there a way to train a model without centralizing users' data, i.e. ensuring 'privacy by default'?

Goal

K number of users wish to train a state-of-the-art machine learning model, collectively, without sharing their respective data $\mathcal{D}_i, \forall i \in 1, \ldots, K$; to other users.

Problem Formulation:

Problem Formulation:

1. K number of workers participate in the learning process contributing to a total of n number of data points.

Problem Formulation:

1. K number of workers participate in the learning process contributing to a total of n number of data points.
2. Each worker, k, has a set of data points $\mathbf{S}$ of size $n^{(k)} = |\mathbf{S}|$, i.e. $n = \sum_{k=1}^{K} n^{(k)}$.

Problem Formulation:

1. K number of workers participate in the learning process contributing to a total of n number of data points.

2. Each worker, k, has a set of data points $\mathbf{S}$ of size $n^{(k)} = |\mathbf{S}|$, i.e. $n = \sum_{k=1}^{K} n^{(k)}$.

3. In order to train a machine learning model, with parameters w, on the labled data points $(\mathbf{x}, \mathbf{y})$ for each k, we consider a local objective function of $f^k(w) = \frac{1}{n^{(k)}} \sum_{i \in \mathbf{S}} l(x_i, y_i; w)$.

Problem Formulation:

1. K number of workers participate in the learning process contributing to a total of n number of data points.
2. Each worker, k, has a set of data points $\mathbf{S}$ of size $n^{(k)} = |\mathbf{S}|$, i.e. $n = \sum_{k=1}^{K} n^{(k)}$.
3. In order to train a machine learning model, with parameters w, on the labled data points $(\mathbf{x}, \mathbf{y})$ for each k, we consider a local objective function of $f^k(w) = \frac{1}{n^{(k)}} \sum_{i \in \mathbf{S}} l(x_i, y_i; w)$.
4. In a federated setting, we can write the objective function $f^f(w)$ in the following form,

$$\min_w f^f(w) = \sum_{k=1}^{K} p_k f^k(w) = \mathbb{E}_k[f^k(w)]$$

where $p_k = \frac{n^{(k)}}{n}$, $p_k \geq 0$ & $\sum_k p_k = 1$

How to motivate workers?

$\Longrightarrow$ Workers are using their own CPU power for the shared training.

How to motivate workers?

$\Longrightarrow$ Workers are using their own CPU power for the shared training.

$\Longrightarrow$ How would the server effectively allocate resources to achieve a global accuracy with minimum iterations?

How to motivate workers?

$\Longrightarrow$ Workers are using their own CPU power for the shared training.

$\Longrightarrow$ How would the server effectively allocate resources to achieve a global accuracy with minimum iterations?

$\Longrightarrow$ How to develop a mechanism to motivate workers by optimizing the resource allocation?

How to motivate workers?

$\Longrightarrow$ Workers are using their own CPU power for the shared training.

$\Longrightarrow$ How would the server effectively allocate resources to achieve a global accuracy with minimum iterations?

$\Longrightarrow$ How to develop a mechanism to motivate workers by optimizing the resource allocation?

Stackelberg game formulation to address the challenges?

How to motivate workers?

$\Longrightarrow$ Workers are using their own CPU power for the shared training.

$\Longrightarrow$ How would the server effectively allocate resources to achieve a global accuracy with minimum iterations?

$\Longrightarrow$ How to develop a mechanism to motivate workers by optimizing the resource allocation?

Stackelberg game formulation to address the challenges?

Work by Sarikaya and Erçetin indicates that [1]

[1] Yunus Sarikaya and Özgür Erçetin, "Motivating Workers in Federated Learning: A Stackelberg Game Perspective", *Arxiv*, Aug. 2019. [Online] https://arxiv.org/abs/1908.03092

STACKELBERG GAME FORMULATION IN FEDERATED LEARNING

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.
$\Longrightarrow$ Total time required for completing the t-th update by all K participating workers is $\max_k T_{k,t}$

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.
$\Longrightarrow$ Total time required for completing the t-th update by all K participating workers is $\max_k T_{k,t}$
$\Longrightarrow$ $T_{k,t}$ is exponentially distributed with a mean $\frac{P_k}{c_k}$

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.
$\Longrightarrow$ Total time required for completing the t-th update by all K participating workers is $\max_k T_{k,t}$
$\Longrightarrow$ $T_{k,t}$ is exponentially distributed with a mean $\frac{P_k}{c_k}$
$\Longrightarrow$ c_k denotes the number of CPU cycles required by the worker to perform t-th update

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.
$\Longrightarrow$ Total time required for completing the t-th update by all K participating workers is $\max_k T_{k,t}$
$\Longrightarrow$ $T_{k,t}$ is exponentially distributed with a mean $\frac{P_k}{c_k}$
$\Longrightarrow$ c_k denotes the number of CPU cycles required by the worker to perform t-th update $\longleftarrow$ Worker needs to pay for this

Problem Formulation

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.
$\Longrightarrow$ Total time required for completing the t-th update by all K participating workers is $\max_k T_{k,t}$
$\Longrightarrow$ $T_{k,t}$ is exponentially distributed with a mean $\frac{P_k}{c_k}$
$\Longrightarrow$ c_k denotes the number of CPU cycles required by the worker to perform t-th update $\longleftarrow$ Worker needs to pay for this
$\Longrightarrow$ P_k denote the CPU power that the worker allocates for the update

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.
$\Longrightarrow$ Total time required for completing the t-th update by all K participating workers is $\max_k T_{k,t}$
$\Longrightarrow$ $T_{k,t}$ is exponentially distributed with a mean $\frac{P_k}{c_k}$
$\Longrightarrow$ c_k denotes the number of CPU cycles required by the worker to perform t-th update $\longleftarrow$ Worker needs to pay for this
$\Longrightarrow$ P_k denote the CPU power that the worker allocates for the update
Based on this he would have to negotiate with the server for reward

$\Longrightarrow$ Server (or the leader) sends the initial model update to all K workers
$\Longrightarrow$ Each worker k runs SGD locally on their own dataset and send back the parameter update to the server for global update
$\Longrightarrow$ $T_{k,t}$ is the time taken by k-th worker to perform t-th update.
$\Longrightarrow$ Total time required for completing the t-th update by all K participating workers is $\max_k T_{k,t}$
$\Longrightarrow$ $T_{k,t}$ is exponentially distributed with a mean $\frac{P_k}{c_k}$
$\Longrightarrow$ c_k denotes the number of CPU cycles required by the worker to perform t-th update $\longleftarrow$ Worker needs to pay for this
$\Longrightarrow$ P_k denote the CPU power that the worker allocates for the update
Based on this he would have to negotiate with the server for reward
$\Longrightarrow$ If the server agrees to pay q_k to k-th worker for per unit of CPU power, worker k would get $q_k P_k$ from the server to perform the t-th update.

$\Longrightarrow$ Server's cost function J', it can be defined as follows,

$$J'\left(q_k, P_k\right) = \alpha \mathbb{E}\left[\max_k T_{k,t}\right] + \sum_{k=1}^{K} q_k P_k$$

where α is a constant.

$\Longrightarrow$ Server's cost function J', it can be defined as follows,

$$J'\left(q_k, P_k\right) = \alpha \mathbb{E}\left[\max_k T_{k,t}\right] + \sum_{k=1}^{K} q_k P_k$$

where α is a constant.

$\Longrightarrow$ $\mathbb{E}\left[\max_k T_{k,t}\right]$ from above equation can be represented by,

$$\mathbb{E}\left[\max_k T_{k,t}\right] = \sum_{\mathbf{S} \subseteq \{1,2,\ldots,K\}} (-1)^{|\mathbf{S}|-1} \frac{1}{\sum_{k \in S} \lambda_k}$$

where $\lambda_k = \frac{P_k}{c_k}$

$\Longrightarrow$ Server's cost function J', it can be defined as follows,

$$J'\left(q_k, P_k\right) = \alpha \mathbb{E}\left[\max_k T_{k,t}\right] + \sum_{k=1}^{K} q_k P_k$$

where α is a constant.

$\Longrightarrow$ $\mathbb{E}\left[\max_k T_{k,t}\right]$ from above equation can be represented by,

$$\mathbb{E}\left[\max_k T_{k,t}\right] = \sum_{\mathbf{S} \subseteq \{1,2,\ldots,K\}} (-1)^{|\mathbf{S}|-1} \frac{1}{\sum_{k \in S} \lambda_k}$$

where $\lambda_k = \frac{P_k}{c_k}$

$\Longrightarrow$ Cost function of the worker, J'', can be defined by,

$$J_k''\left(P_k, q_k\right) = q_k P_k - \kappa c_k \left(P_k\right)^2$$

where κ is a chip architecture dependent constant

$\Longrightarrow$ The game formulation for the worker can be as follows:

$$\max_{P_k} J_k''(P_k, q_k) = q_k P_k - \kappa c_k (P_k)^2$$
$$\text{s.t.} P_k \leq P_{\max}$$

where $P_{\max}$ represents the maximum available CPU power.

$\Longrightarrow$ The game formulation for the worker can be as follows:

$$\max_{P_k} J_k''(P_k, q_k) = q_k P_k - \kappa c_k (P_k)^2$$
$$\text{s.t.} P_k \leq P_{\max}$$

where $P_{\max}$ represents the maximum available CPU power.

$\Longrightarrow$ Game formulation for the server would be,

$$\min_{\boldsymbol{q}} J' = \alpha \mathbb{E}\left[\max_k T_{k,t}\right] + \sum_{k=1}^{K} q_k P_k$$
$$\text{s.t.} \sum_{k=1}^{K} q_k P_k \leq B$$

where B is the available budget to the server to pay the workers.

$\Longrightarrow$ Solving for the worker's game,

$$P_k^*(q_k) = \begin{cases} \frac{q_k}{2\kappa c_k} \text{ if } \frac{q_k}{2\kappa c_k} \leq P_{\max} \\ P_{\max} \text{ if } \frac{q_k}{2\kappa c_k} > P_{\max} \end{cases}$$

$\Longrightarrow$ Solving for the worker's game,

$$P_k^*(q_k) = \begin{cases} \frac{q_k}{2\kappa c_k} \text{ if } \frac{q_k}{2\kappa c_k} \leq P_{\max} \\ P_{\max} \text{ if } \frac{q_k}{2\kappa c_k} > P_{\max} \end{cases}$$

$\Longrightarrow$ Optimal solution for the server $q_k^* = \sqrt{\frac{2B\kappa c}{K}}$

RESULTS AND ANALYSIS

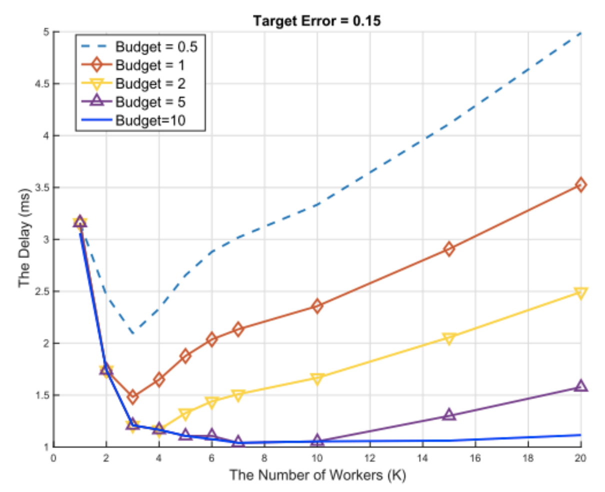

Figure: Analysis of delay with increase in K

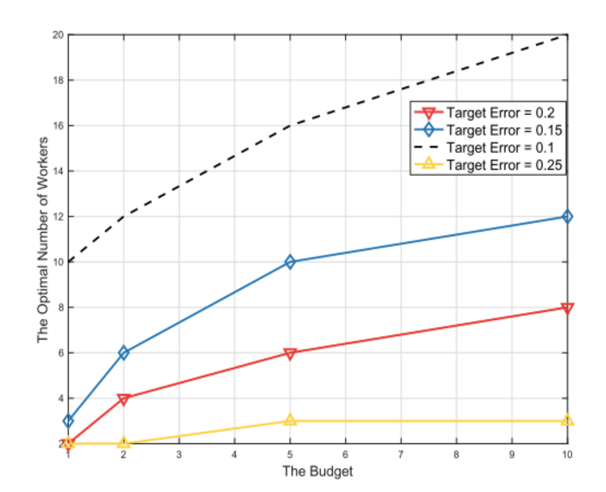

Figure: Analysis of availability of budget and optimality of number of workers

conclusions & future works

$\Longrightarrow$ Laying out the motivations behind studying the Stackelberg game, with relevant examples

$\Longrightarrow$ Laying out the motivations behind studying the Stackelberg game, with relevant examples

$\Longrightarrow$ Understanding of the pure-strategy Stackelberg equilibria and comparison with the pure-strategy NE

$\Longrightarrow$ Laying out the motivations behind studying the Stackelberg game, with relevant examples

$\Longrightarrow$ Understanding of the pure-strategy Stackelberg equilibria and comparison with the pure-strategy NE

$\Longrightarrow$ Understanding of the mixed-strategy Stackelberg equilibria, its existence and comparison with the mixed-strategy NE

Conclusions

$\Longrightarrow$ Laying out the motivations behind studying the Stackelberg game, with relevant examples

$\Longrightarrow$ Understanding of the pure-strategy Stackelberg equilibria and comparison with the pure-strategy NE

$\Longrightarrow$ Understanding of the mixed-strategy Stackelberg equilibria, its existence and comparison with the mixed-strategy NE

$\Longrightarrow$ Defining the problem of federated learning and its connection with the Stackelberg game formulation

$\Longrightarrow$ Laying out the motivations behind studying the Stackelberg game, with relevant examples

$\Longrightarrow$ Understanding of the pure-strategy Stackelberg equilibria and comparison with the pure-strategy NE

$\Longrightarrow$ Understanding of the mixed-strategy Stackelberg equilibria, its existence and comparison with the mixed-strategy NE

$\Longrightarrow$ Defining the problem of federated learning and its connection with the Stackelberg game formulation

$\Longrightarrow$ Understanding the cost function formulation and obtaining the Stackelberg equilibria solution for federated learning.

$\Longrightarrow$ Convergence in federated learning depends on the amount of data one worker pose, as $f^f(w) \propto \frac{n^{(k)}}{n}$

$\Longrightarrow$ Convergence in federated learning depends on the amount of data one worker pose, as $f^f(w) \propto \frac{n^{(k)}}{n} \longleftarrow$ How to use $n^{(k)}$ in the cost function to obtain an optimal resource allocation technique?

$\Longrightarrow$ Convergence in federated learning depends on the amount of data one worker pose, as $f^f(w) \propto \frac{n^{(k)}}{n} \longleftarrow$ How to use $n^{(k)}$ in the cost function to obtain an optimal resource allocation technique?

$\Longrightarrow$ Authors considered the case of all honest workers

Future Works

$\Longrightarrow$ Convergence in federated learning depends on the amount of data one worker pose, as $f^f(w) \propto \frac{n^{(k)}}{n} \longleftarrow$ How to use $n^{(k)}$ in the cost function to obtain an optimal resource allocation technique?

$\Longrightarrow$ Authors considered the case of all honest workers $\longleftarrow$ Can a number of dishonest workers manipulate the server to allocate more resource?

BACKUP SLIDES

Centralized Learning

Figure: Centralized learning

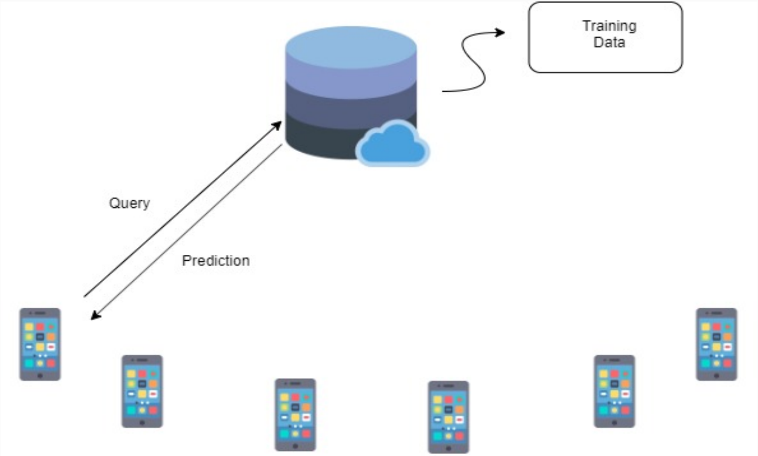

Figure: Centralized learning

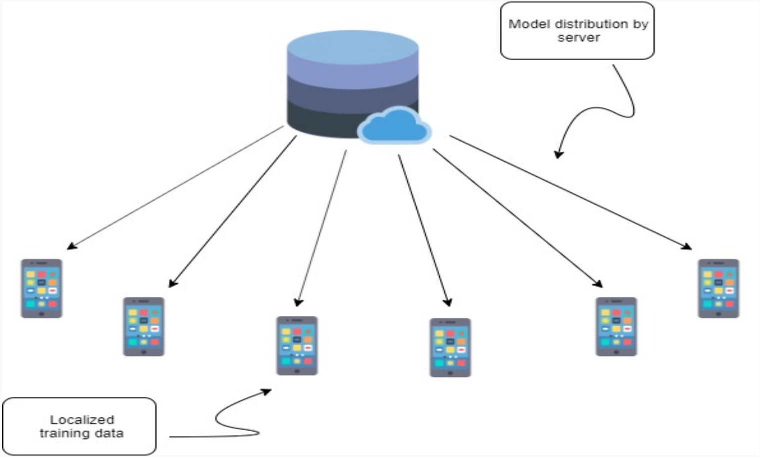

Figure: Federated learning

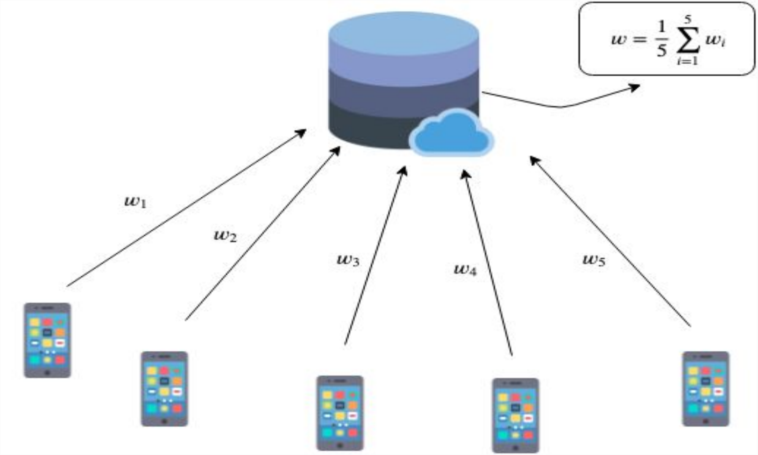

Figure: Federated learning

Federated Averaging ($Fed - Avg$) Algorithm [2]

Until Convergence:

Server:

1. Select K number of users randomly.
2. Send w_t, i.e. parameter update at t^{th} iteration, to all K users.

   User:

   1. Download parameter update w_t from the server.
   2. Run SGD locally, for E epochs, and obtain w_t^k.
   3. Upload $w_t - w_t^k$ to the server.

3. $w_{t+1} = w_t +$ weighted average of the parameter updates by K users.

[2] McMahan *et al.*, 'Communication-efficient Learning of Deep Networks from Decentralized Data', AISTATS, 2017.