Lecture 24: Application of Game Theory to Smart Grid

Distributed Demand Side Management with Energy Storage in Smart Grid

IEEE Transactions on Parallel and Distributed Systems

Hung Khanh Nguyen, Ju Bin Son, Zhu Han

Motivation

Energy generation

Energy distribution

Energy consumption

Fluctuation
in energy generation

Temporal and Spatial
Imbalance between the
energy generation and
consumptions

Fluctuation
in energy consumptions

Energy quality ↓
Grid stability ↓
Grid effectiveness ↓

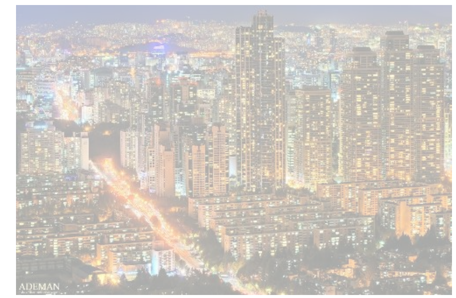
Motivation

Energy generation

- Increase efficiency
- Reduce intermittence

Energy distribution

Energy consumption



- Regulate energy usage
- Change energy price

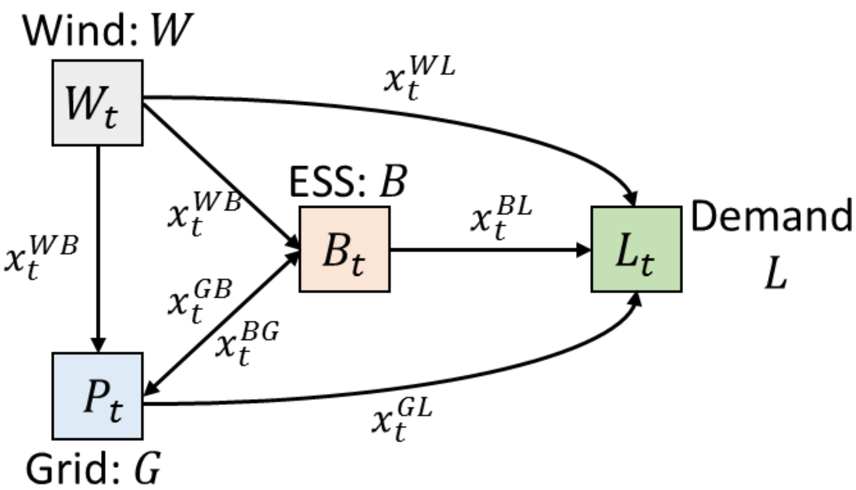
Information on energy generation

Resolve imbalance through Energy Storage System (ESS)

Information on energy consumption

- The ESS charges the excessive energy from the renewable energy facility, and it discharges the energy when the demand of the energy is high.
- Through this mechanism, the ESS can resolve the imbalance between energy generation and usage.
- Consequently, controlling the charging and discharging schedule of the ESS becomes more important

Motivation



- W_t:the amount of wind energy generation at time t
- P_t^B:the buying energy price from the grid at time t
- P_t^S:the selling energy price from the grid at time t
- B_t :the amount of stored energy in the ESS at time t
- L_t:the amount of energy that should be provided at time t

$$\min_{x_1,\ldots,x_T} \sum_{t=1}^{T} \{\hat{P}_t^B(x_t^{GB}+x_t^{GL}) - \hat{P}_t^S(x_t^{WG}+x_t^{BG})\}$$

$$\text{s.t. for } t = 1, \ldots, T$$

$$\hat{L}_t = x_t^{WL} + x_t^{BL} + x_t^{GL}$$

$$B_t = B_{t-1} + x_t^{WB} + x_t^{GB} - x_t^{BL} - x_t^{BG}$$

$$\widehat{W}_t = x_t^{WG} + x_t^{WB} + x_t^{WL}$$

$$0 \le x_t^{WB} + x_t^{GB} \le \bar{C}^B \rho_c$$

$$0 \le x_t^{BG} + x_t^{BL} \le \bar{C}^B \rho_{dc}$$

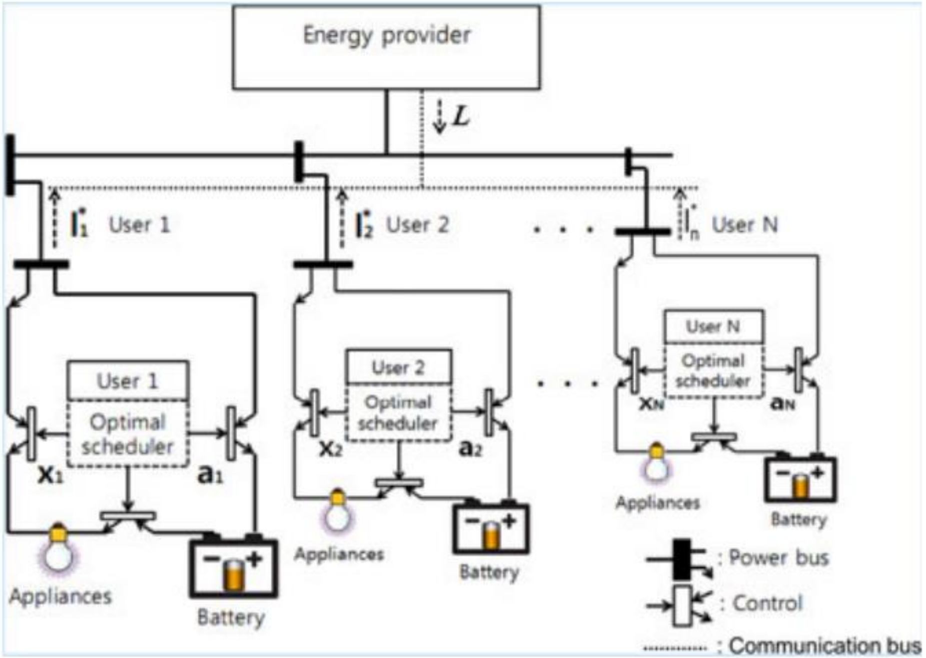
$$\underline{C}^B \le B_t \le \bar{C}^B$$

$$0 \le x_t^{WB}, x_t^{GB}, x_t^{BG}, x_t^{BL}, x_t^{WL}, x_t^{WG}, x_t^{GL}$$

$s_t = (L_t, W_t, P_t^B, P_t^S, B_t)$: state variable representing the current status of the target ESS

$x_t = (x_t^{WG}, x_t^{WB}, x_t^{GB}, x_t^{BG}, x_t^{WL}, x_t^{BL}, x_t^{GL})$: control variables that we adjust

Formulation



- Model composed of one energy provide and N load subscriber (or users)

- Each user is equipped with a battery
  - $\mathcal{N}$ be a set of users, $N = |\mathcal{N}|$

- The time period of analysis is divided into T equal length time slots
  - $\mathcal{T}$ be a set of time slots, $T = |\mathcal{T}|$
  - For example, this division can simply resent $T = 24$ hours of a day,

Formulation

Actions for each user (Appliance with controllable/ shiftable load)

Definitions:

- For each user $i \in \mathcal{N}$, we define the energy consumption vector

$$\boldsymbol{x}_i = \left[x_i^1, \dots, x_i^t, \dots, x_i^T\right]^T$$

  ✓ where x_i^t is the energy needed by user i to supply its appliance at time slot t

- We define $\mathcal{O}_i \in \mathcal{T}$ as the set of operating time slots of user $i \in \mathcal{N}$

Constraints:

- Constraint on minimum and maximum consumption levels for each user in the operating time slot $\mathcal{O}_i$

$$0 \le x_i^t \le x_i^{max}, \qquad \forall t \in \mathcal{O}_i$$

- User can consume energy only for operating time slots

$$x_i^t = 0, \qquad \forall t \in \mathcal{T} \backslash \mathcal{O}_i$$

- The total energy demand D_i^{app} for appliance should be satisfied:

$$\sum_{t \in \mathcal{O}_i} x_i^t = D_i^{app}$$

Formulation

Actions for each user (Battery operation)

Definitions:

- For each user $i \in \mathcal{N}$, we define the energy consumption vector

$$\boldsymbol{a}_i = \left[a_i^1, \dots, a_i^t, \dots, a_i^T\right]^T$$

  - ✓ where a_i^t is the energy charging and discharging of user i for its battery at time slot t
- For each user $i \in \mathcal{N}$, b_i^t is the charge level of battery for user i at time slot t

Constraints:

- a_i^t should satisfy the maximum charging and discharging rate for one time slot

$$-a_i^{max} \leq a_i^t \leq a_i^{max}$$

- The dynamics of battery charging state

$$0 <= b_i^t = b_i^0 + \sum_{h=1}^{t} a_i^t \leq B_i^{cap}, \qquad \forall t \in \mathcal{T}$$

- Amount of energy needed for charging its battery per day

$$\sum_{t=1}^{T} a_i^t = D_i^{bat} = b_i^T - b_i^0$$

Formulation

Actions for each user

Definitions:

- For each day, the total energy demand, E_i the user i purchases from the energy provide to supply for its appliance and battery can be calculated as

$$E_i = D_i^{app} + D_i^{bat}$$

- The actual load demand l_i^t that user i need to buy from the energy provider

$$l_i^t = x_i^t + a_i^t$$

- The actual load demand vector over T time slots of user i is defined as

$$\boldsymbol{l}_i = \left[l_i^t, \ldots, l_i^t, \ldots, l_i^T\right]^T$$

Constraints:

- At each time slot t, user $i's$ battery cannot provide more energy than the amount of energy consumed by its appliance

$$x_i^t + a_i^t \geq 0$$

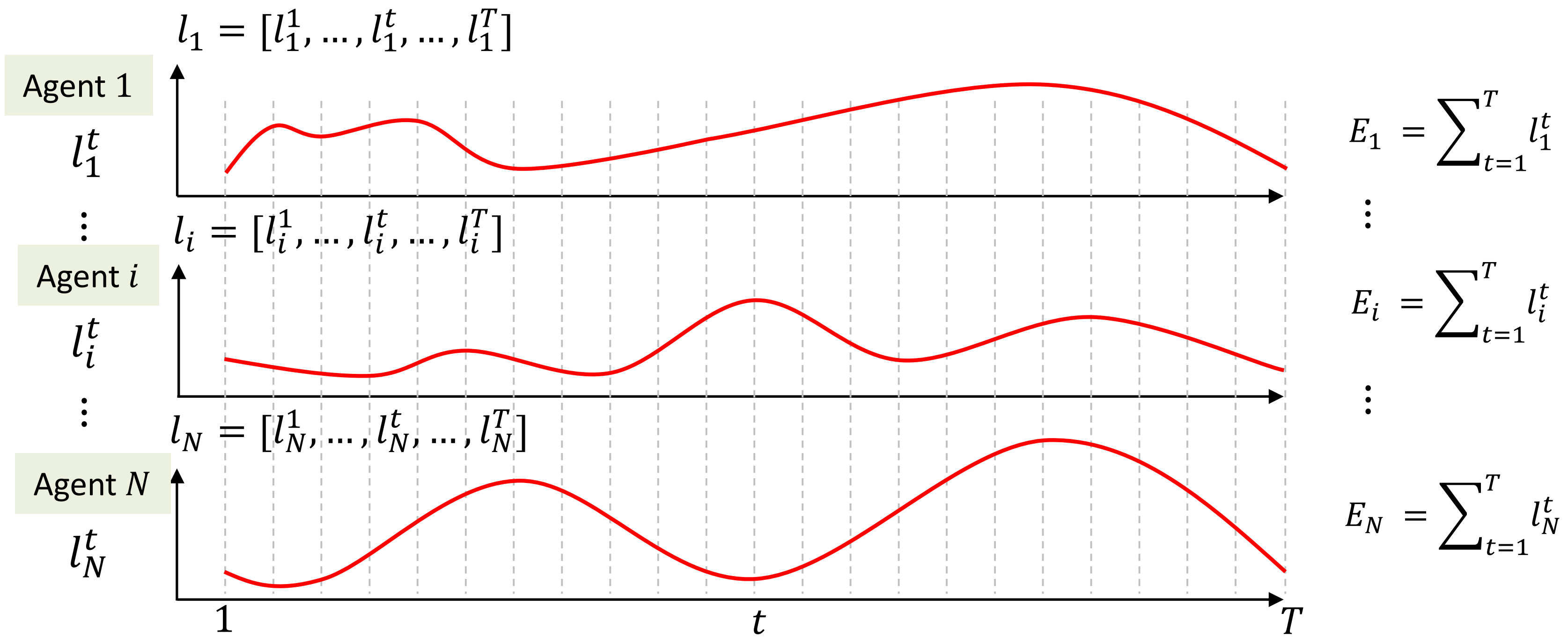
Formulation

Actions for each user

Definitions:

- Then, we can define the set of feasible energy consumption and the energy storage (battery) schedule of user i

$$\mathcal{F}_i = \{\boldsymbol{x}_i, \boldsymbol{a}_i | \text{all the constraints are satisfied}\}$$



Centralized Design

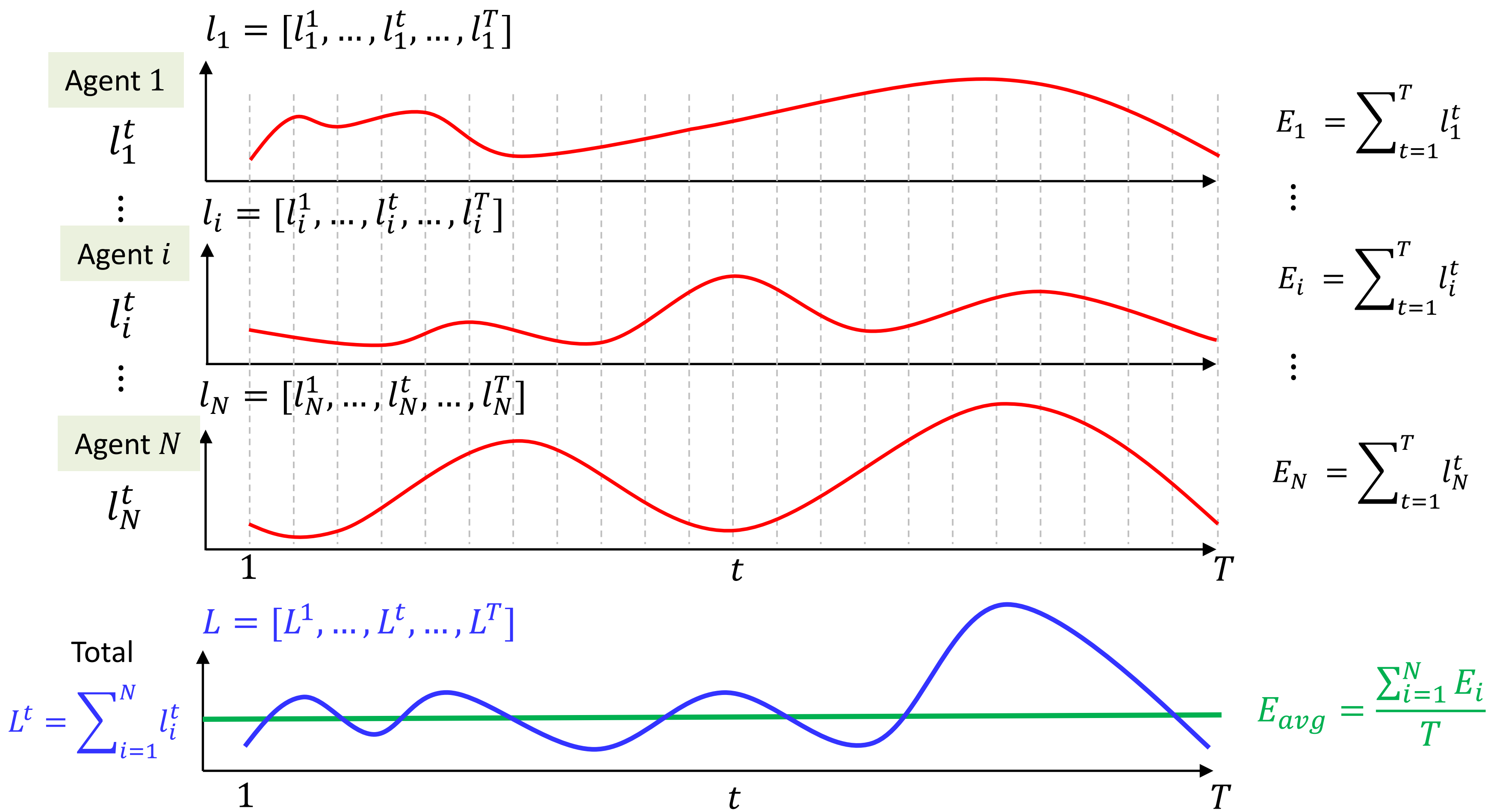
- Given all load demand vectors, $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_N$ of all users, the total load demand of all uses at time slot $t \in \mathcal{T}$ can be calculated as

$$L^t = \sum_{i=1}^{N} l_i^t$$

- The average demand of the system over T time slots can be defined as

$$E_{avg} = \frac{\sum_{i=1}^{N} E_i}{T}$$

Centralized Design



- The square Euclidean distance between the instantons target load demand and average demand can be calculated as

$$\mathrm{SED} = \sum_{t=1}^{T}\left(\sum_{i=1}^{N} l_i^t - E_{avg}\right)^2 = \sum_{t=1}^{T}\left(\sum_{i=1}^{N}\left(x_i^t + a_i^t\right) - E_{avg}\right)^2$$

Centralized Design

- The SED minimization problem can be formulated with respect to variables $\boldsymbol{x}$ and $\boldsymbol{a}$ as

$$\max_{\{\boldsymbol{x}_i,\boldsymbol{a}_i\}\in\mathcal{F}_i,\forall i} \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \left(x_i^t + a_i^t\right) - E_{avg} \right)^2$$

✓ where $\mathcal{F}_i = \{\boldsymbol{x}_i, \boldsymbol{a}_i | \text{all the constraints are satisfied}\}$

Remarks on the problem:

- The objective function is convex (quadratic) and the feasible set $\mathcal{F}_i$ is convex, thus the above optimization problem is convex
  → The optimization problem can be solved using various convex programing techniques
  → Has a unique optimal solution (global maximum)

Decentralized Design

Motivations

- The centralized SED minimization problem is impractical due to the following reasons:
  - ✓ A central controller needs to be implemented as the provider's side to collect all required information for solving the optimization problem, which is difficult due to enormous amount of signaling
  - ✓ By monitoring owners' information, this system infringes the users' privacy
  - ✓ We cannot sure users have willingness to participate in Demand side management (DSM) scheme

- Design a **distributed scheme** in which users independently optimize their energy consumption and storage schedules to achieve the best system performance

Users' independent greedy behavior: Minimize their energy cost	→	Best system performance: Minimum of SED

- Form users' perspective, they only try to schedule their energy consumption in an appropriate ways so that the total amount of money pay for energy provider can be minimized
- Use game theoretic formulation!

Decentralized Design

Energy Cost Sharing Model

- Design a energy cost model for the energy provider to charge for each user energy demand. The cost of generating L_t units of energy is

$$C(L_t) = \delta(L_t)^2$$

  - ✓ δ is a positive coefficient widely used in smart grid literature.

- Then the total energy cost of the system over T time slots can be calculated as

$$C_{total} = \sum_{t=1}^{T} C(L_t)$$

- User i's payment is proportional to the total energy demand of the system

$$C_i = k_i \sum_{t=1}^{T} C(L_t)$$

  - ✓ $k_i = \frac{E_i}{\sum_{j=1}^{N} E_j}$ denote the proportion of user i's energy consumption in the system

Decentralized Design

Utility Function

- Let's denote $\boldsymbol{l}_i = (\boldsymbol{x}_i, \boldsymbol{a}_i)$ the strategy vector for user $i \in \mathcal{N}$
  - $\boldsymbol{x}_i = \left[x_i^1, \dots, x_i^T\right]$ the appliance control
  - $\boldsymbol{a}_i = \left[a_i^1, \dots, a_i^T\right]$ the battery control

- User i's utility function is negative energy payment over T time slots:

$$U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i}) = -C_i = -k_i \sum_{t=1}^{T} C(L_t) = -k_i \sum_{t=1}^{T} \delta\left(l_i^t + l_{-i}^t\right)^2 = -\frac{E_i}{\sum_{j=1}^{N} E_j} \sum_{t=1}^{T} \delta\left(l_i^t + l_{-i}^t\right)^2$$

$$= -\frac{\sum_{t=1}^{T} l_i^t}{\sum_{i=1}^{N} \sum_{t=1}^{T} l_i^t} \sum_{t=1}^{T} \delta\left(l_i^t + l_{-i}^t\right)^2$$

✓ $\boldsymbol{l}_{-i} = [\boldsymbol{l}_1, \dots, \boldsymbol{l}_{i-1}, \boldsymbol{l}_{i+1}, \dots, \boldsymbol{l}_N]$ is the vector of load prolife by all other users except user i

✓ $l_i^t = x_i^t + a_i^t$ is the user i's energy consumption at time t

✓ $l_{-i}^t = \sum_{j \neq i} l_j^t = \sum_{j \neq i}\left(x_i^t + a_i^t\right)$ is the total energy consumption of all the users except i at time t

❖ Note that the utility function for each user is defined over T time slots. That is, we solve very big normal form game, which will be defined at the next slide

Decentralized Design

Game Model

- In a decentralized smart grid system, each user $i \in \mathcal{N}$ minimizes its energy payment independently, which leas to a non-cooperative **Energy Consumption and Storage (NECS) game** $G = \{\mathcal{N}, \{\mathcal{F}_i\}_{i\in\mathcal{N}}, \{U_i\}_{i\in\mathcal{N}}\}$ composed of
    - The players
    - The strategy of each player $i \in \mathcal{N}$, which corresponds to an energy consumption profile, $\boldsymbol{l}_i = (\boldsymbol{x}_i, \boldsymbol{a}_i) \in \mathcal{F}_i$
    - The utility function $U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i})$ for each user $i \in \mathcal{N}$

- The users try to select their energy consumption $\boldsymbol{x}_i$ and battery charging/discharging schedules $\boldsymbol{a}_i$ to minimize their energy payments

- The best strategy for user $i \in \mathcal{N}$ for given other action profiles $\boldsymbol{l}_{-i}$ is then

$$\boldsymbol{l}_i^* = \underset{\boldsymbol{l}_i \in \mathcal{F}_i}{\operatorname{argmax}} \, U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i})$$

Decentralized Design

Nash equilibrium strategy

Definition (Nash equilibrium of the NECS game)

For the NECS game $G = \{\mathcal{N}, \{\mathcal{F}_i\}_{i\in\mathcal{N}}, \{U_i\}_{i\in\mathcal{N}}\}$, a vector of strategies $\boldsymbol{l}^* = (\boldsymbol{l}_i^*, \boldsymbol{l}_{-i}^*)$ constitutes a Nash equilibrium of the NECS game, if an only if, it satisfies the following set of inequalities:

$$U_i(\boldsymbol{l}_i^*, \boldsymbol{l}_{-i}^*) \geq U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i}^*),\ \forall \boldsymbol{l}_i \in \mathcal{F}_i, \forall i \in \mathcal{N}$$

- A Nash equilibrium of the NECS game defines a state in which no user can improve its utility by unilaterally changing the energy consumption schedule, given the equilibrium choices of the other users.

Decentralized Design

Nash equilibrium strategy

Theorem (Existence of Equilibria for infinite Games by Debreu, Glicksberg, Fan)

Consider a strategic game $\{\mathcal{N}, \{S_i\}_{i\in\mathcal{N}}, \{U_i\}_{i\in\mathcal{N}}\}$ such that for each $i \in \mathcal{N}$

- S_i is compact and convex;
- $U_i(s_i, s_{-i})$ is continuous in s_{-i}
- $U_i(s_i, s_{-i})$ is continuous and concave in s_i

Then a unique pure Nash equilibrium exists

Therefore, NECS game has a unique Nash equilibrium

- $\mathcal{F}_i$ is compact and convex;
- $U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i})$ is continuous in $\boldsymbol{l}_{-i}$
- $U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i})$ is continuous and concave in $\boldsymbol{l}_i$

Decentralized Design

Nash equilibrium strategy

Theorem

The unique Nash equilibrium of the NECS game is the global optimal solution of the SED minimization problem

The solution of the SED minimization problem

$$\max_{\{\boldsymbol{x}_i,\boldsymbol{a}_i\}\in\mathcal{F}_i,\forall i} \sum_{t=1}^{T}\left(\sum_{i=1}^{N}\left(x_i^t + a_i^t\right) - E_{avg}\right)^2$$

where $\mathcal{F}_i = \{\boldsymbol{x}_i, \boldsymbol{a}_i | \text{all the constraints are satisfied}\}$

Nash equilibrium of the NECS game

$\boldsymbol{l}^* = (\boldsymbol{l}_i^*, \boldsymbol{l}_{-i}^*)$ satisfying

$$U_i(\boldsymbol{l}_i^*, \boldsymbol{l}_{-i}^*) \geq U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i}^*),\ \forall \boldsymbol{l}_i \in \mathcal{F}_i, \forall i \in \mathcal{N}$$

Proof:

- Let $\{\boldsymbol{x}_1^*, \boldsymbol{a}_1^*\}, \dots, \{\boldsymbol{x}_N^*, \boldsymbol{a}_N^*\}$ be the optimal solution of the SED problem. We define

$$J^* \triangleq \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \left(x_i^{t*} + a_i^{t*} \right) - E_{avg} \right)^2$$
$$= \sum_{t=1}^{T} \left(\sum_{i=1}^{N} l_i^{t*} - E_{avg} \right)^2$$
$$= \sum_{t=1}^{T} \left(l_i^{t*} + l_{-i}^{t*} - E_{avg} \right)^2 \ \text{ with } \ l_{-i}^{t*} \triangleq \sum_{j \neq i}^{N} l_j^{t*}$$

- Since J^* is the optimal value, we have the following inequality for any arbitrary $\{\boldsymbol{x}_i, \boldsymbol{a}_i\}$

$$J^* \leq \sum_{t=1}^{T} \left(\left(x_i^t + a_i^t \right) + \sum_{j \neq i}^{N} \left(x_j^{t*} + a_j^{t*} \right) - E_{avg} \right)^2$$

$$\sum_{t=1}^{T} \left(l_i^{t*} + l_{-i}^{t*} - E_{avg} \right)^2 \leq \sum_{t=1}^{T} \left(l_i^{t} + l_{-i}^{t*} - E_{avg} \right)^2$$

Proof:

$$\sum_{t=1}^{T}\left(l_i^{t*}+l_{-i}^{t*}-E_{avg}\right)^2 \le \sum_{t=1}^{T}\left(l_i^{t}+l_{-i}^{t*}-E_{avg}\right)^2$$

$$\Rightarrow \sum_{t=1}^{T}\left[\left(l_i^{t*}+l_{-i}^{t*}\right)^2-2E_{avg}\left(l_i^{t*}+l_{-i}^{t*}\right)+E_{avg}^2\right] \le \sum_{t=1}^{T}\left[\left(l_i^{t}+l_{-i}^{t*}\right)^2-2E_{avg}\left(l_i^{t}+l_{-i}^{t*}\right)+E_{avg}^2\right]$$

$$\Rightarrow \sum_{t=1}^{T}\left(l_i^{t*}+l_{-i}^{t*}\right)^2 \le \sum_{t=1}^{T}\left(l_i^{t}+l_{-i}^{t*}\right)^2 \qquad \because E_{avg}\sum_{t=1}^{T}\left(l_i^{t}+l_{-i}^{t*}\right)=E_{avg}\sum_{t=1}^{T}L_i^t=E_{avg}\sum_{i=1}^{N}E_i$$

$$\Rightarrow -k_i\sum_{t=1}^{T}\delta\left(l_i^{t*}+l_{-i}^{t*}\right)^2 \ge -k_i\sum_{t=1}^{T}\delta\left(l_i^{t*}+l_{-i}^{t*}\right)^2$$

$$\Rightarrow U_i(\boldsymbol{l}_i^*,\boldsymbol{l}_{-i}^*) \ge U_i(\boldsymbol{l}_i,\boldsymbol{l}_{-i}^*)$$

- The optimal solution of SED problem is a Nash equilibrium for NECS game
- The NECS game has a unique Nash equilibrium
- Therefore, the optimal solution of SED problem is the unique Nash equilibrium of NECS game

The total energy consumption does not change! But only redistributed

$$E=\sum_{t=1}^{T}L^t=\sum_{t=1}^{T}\sum_{i=1}^{N}l_i^t=\sum_{i=1}^{N}\sum_{t=1}^{T}l_i^t=\sum_{i=1}^{N}E_i=E$$

Decentralized Design

Remarks:

- The coincidence of the distributed and centralized solutions is satisfied due to several conditions:
  - The centralized SED problem has a unique optimal solution and NECS game has a unique Nash equilibrium
  - From the quadratic cost sharing function, each user’s best strategy is to schedule its energy consumption in such a way that the load profile of the overall system can be as flat as possible
    - ✓ Therefore, this scheduling mechanism automatically drive the distributed solution to converge to the centralized optimal solution

Algorithm

Game Theory Based Distributed Algorithm

- Consider user $i \in \mathcal{N}$, given fixed $\boldsymbol{l}_{-i}$ and assume that all other users fixed their energy consumption and storage profiles according to $\boldsymbol{l}_{-i}$, then the user i's best response can be determined by solving the local optimization problem:

$$\max_{\boldsymbol{l}_i \in \mathcal{F}_i} U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i})$$

$$\max_{\{\boldsymbol{x}_i, \boldsymbol{a}_i\} \in \mathcal{F}_i} -k_i \sum_{t=1}^{T} \sigma \left(\sum\nolimits_{i=1}^{N} (x_i^t + a_i^t) - E_{avg} \right)^2$$

$$\min_{\{\boldsymbol{x}_i, \boldsymbol{a}_i\} \in \mathcal{F}_i} k_i \sum_{t=1}^{T} \sigma \left(x_i^t + a_i^t + L_{-i}^t - E_{avg} \right)^2$$

Algorithm : Game Theory Based Distributed Algorithm

Algorithm 1. Executed by User i.

Data: Initialize D_i^{app}, b_i^0, b_i^T and B_i^{cap}
while *receive new total load profile signal* $\boldsymbol{L}$ *of the system from the energy provider* **do**
  Calculate $\boldsymbol{L}_{-i}$ by minus its own $\boldsymbol{l}_i$
  Solve (40) to find optimal schedules $\{\boldsymbol{x}_i^*, \boldsymbol{a}_i^*\}$
  if$\{\boldsymbol{x}_i^*, \boldsymbol{a}_i^*\}$ *are changed by comparing with the current values* **then**
    Update new values for $\{\boldsymbol{x}_i^*, \boldsymbol{a}_i^*\}$
    Calculate new value of $\boldsymbol{l}_i^*$ as (11);
    Send the value of $\boldsymbol{l}_i^*$ to the energy provider for calculating the new total load profile $\boldsymbol{L}$
  end
end

- If users update their energy consumption vectors sequentially, i.e., none of two users $i, j \in \mathcal{N}$ update their energy consumption scheduling vectors at the same time, the starting from any random initial condition point, the distributed algorithm will converge to the Nash equilibrium point of NECS game

Algorithm : Game Theory Based Distributed Algorithm

Algorithm 1. Executed by User i.

Data: Initialize D_i^{app}, b_i^0, b_i^T and B_i^{cap}
while *receive new total load profile signal* $\boldsymbol{L}$ *of the system from the energy provider* **do**
  Calculate $\boldsymbol{L}_{-i}$ by minus its own $\boldsymbol{l}_i$
  Solve (40) to find optimal schedules $\{\boldsymbol{x}_i^*, \boldsymbol{a}_i^*\}$
  if $\{\boldsymbol{x}_i^*, \boldsymbol{a}_i^*\}$ *are changed by comparing with the current values* **then**
    Update new values for $\{\boldsymbol{x}_i^*, \boldsymbol{a}_i^*\}$
    Calculate new value of $\boldsymbol{l}_i^*$ as (11);
    Send the value of $\boldsymbol{l}_i^*$ to the energy provider for calculating the new total load profile $\boldsymbol{L}$
  end
end

- **Limitation?**

Algorithm : Proximal Decomposition Algorithm

- Game theory based distributed algorithm is impractical because:
  - ✓ Hard to enforce the sequential updating
  - ✓ Convergence cannot be guaranteed in case of simultaneously updating

- We need an algorithm that
  - ✓ Allows users update their strategies simultaneously
  - ✓ This algorithm is highly tractable and scalable to large numbers of users

Algorithm : Proximal Decomposition Algorithm

- To overcome these issues, the paper propose proximal decomposition algorithm that allows users update their strategies simultaneously

- This algorithm is highly tractable and scalable to large number of users

- At the kth iteration, user $i \in \mathcal{N}$, given fixed $\boldsymbol{l}_{-i}$, solve the following regularized game

$$\max_{\boldsymbol{l}_i \in \mathcal{F}_i} U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i}) - \frac{\tau}{2} \left\| \boldsymbol{l}_i - \boldsymbol{l}_i^{(k)} \right\|^2$$

  ✓ where $\boldsymbol{l}_i^{(k)}$ is the action profile of user i at the kth iteration.

Algorithm : Proximal Decomposition Algorithm

Algorithm 2. Proximal Decomposition Algorithm.

Data: Set $k = 0$ and the initial centroid

$\left(\bar{\boldsymbol{l}}_i^{(0)}\right)_{i=1}^{N} = \mathbf{0}$;

Given δ, any feasible starting point

$\boldsymbol{l}^{(0)} = \left(\boldsymbol{l}_i^{(0)}\right)_{i=1}^{N}$, and $\tau > 0$

while *receive new total load profile signal* $\boldsymbol{L}$ *of the system from the energy provider* **do**

For $i \in \mathcal{N}$, each user computes $\boldsymbol{l}_i^{(k+1)}$ as

$$\boldsymbol{l}_i^{(k+1)} \in \underset{\boldsymbol{l}_i \in \mathcal{F}_i}{\arg\max} \left\{ U_i(\boldsymbol{l}_i, \boldsymbol{l}_{-i}^{(k)}) - \frac{\tau}{2} \|\boldsymbol{l}_i - \bar{\boldsymbol{l}}_i\|^2 \right\} \tag{42}$$

if *the NE has been reached* **then**

Each user $i \in \mathcal{N}$ updates his centroid:

$\bar{\boldsymbol{l}}_i = \boldsymbol{l}_i^{(k+1)}$;

Send the value of $\boldsymbol{l}_i^*$ to the energy provider for calculating the new total load profile $\boldsymbol{L}$;

end

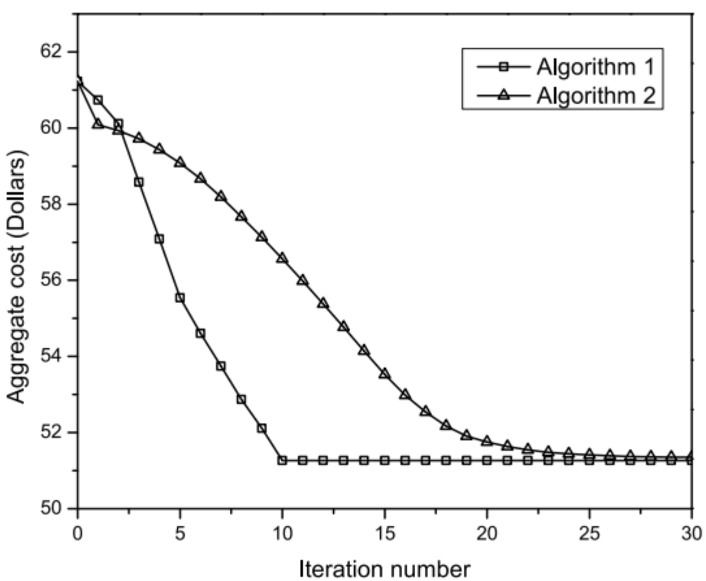
$k \leftarrow k + 1$

end

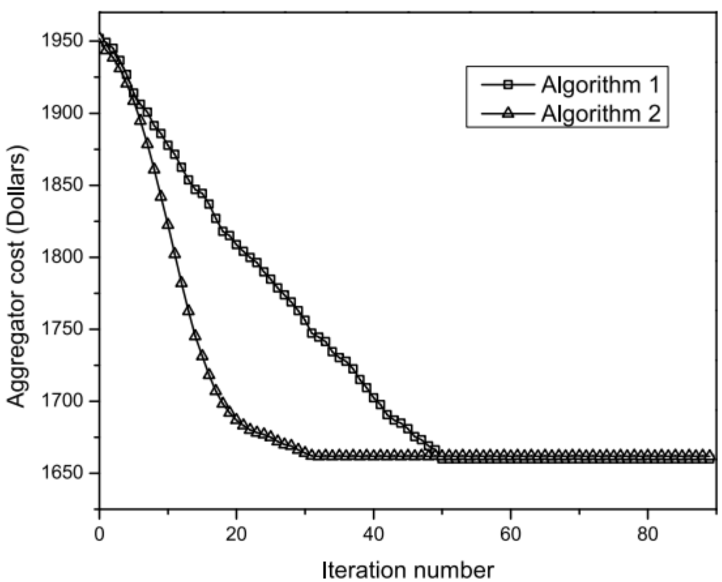
- For sufficient large regularization parameter $\tau > 0$, the game converges to the unique solution of the NECS game that can be computed using a distributed way.

Algorithm : Proximal Decomposition Algorithm

Algorithms performance comparison



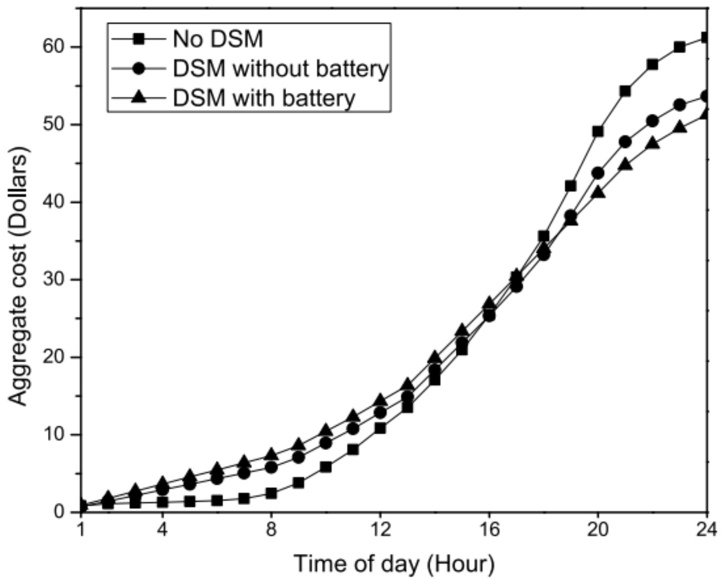
(a) 10 users



(b) 50 users

Algorithm : Proximal Decomposition Algorithm

Energy cost reduction



. 3. Daily aggregate cost across all users.

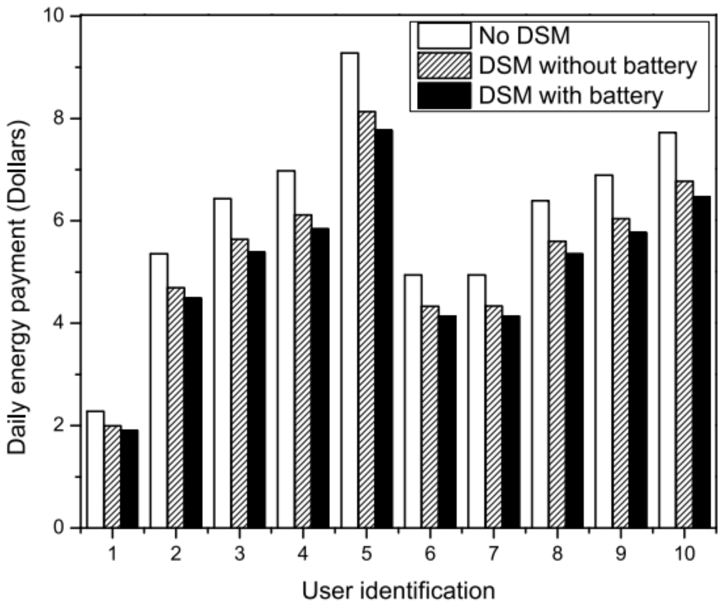


Fig. 5. Daily cost for users.

Algorithm : Proximal Decomposition Algorithm