TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214
DOI: 10.26599/TST.2018.9010039

Distributed Energy Sharing in Energy Internet through Distributed Averaging

Yangyang Ming, Jie Yang, Junwei Cao* and Ziqiang Zhou

Abstract: This paper advances a distributed averaging iteration algorithm for energy sharing in micro grids (called grids or grid sometimes later) of Energy Internet based on common gossip algorithms. This algorithm is completely distributed and only requires communications between neighbors. Using this algorithm, Energy Internet can not only allocate the energy effectively based on grids' load condition, but also schedule the energy transmitted between neighboring grids reasonably. Through theoretical analysis, this paper discusses in which condition, this algorithm can finally reach supply and demand balance. Subsequently, the related simulation verifies the agreed performance of the algorithm under different conditions.

Key words: distributed averaging; energy sharing; gossip algorithm; Energy Internet

1 Introduction

As a modern energy utilizing system, Energy Internet [1] effectively combines the Internet networking technology with highly effective usage of distributed renewable energies through advanced information and communication infrastructure and technologies [2, 3]. As showing basic characters of openness and sharing [4], Energy Internet can realize energy sharing between micro grids which is produced most locally, in order to improve the energy utilizing efficiency. But in Energy Internet, there may lack or not be suitable for a central administration institution sometimes. In order to keep high controlling and managing performance, large scale Energy Internet behaves better by using distributed consensus control strategies. So, energy sharing in distributed means becomes a deserved research topic [5]. In this paper, we realize energy sharing in Energy Internet based on distributed averaging technologies (mainly using gossip algorithms). Through distributed averaging iterations, our designed algorithm can allocate energy in the micro grids according to the energy supply and load demand of everyone.

With the developing of Energy Internet, the concept of Cyber-Physical system is advanced [6], and further develops into Cyber-Physical integration of infrastructure in Energy Internet [7], which means the topo of information network and energy transmission network are isomorphic, and the related facilities of information and energy are co-located and controlled at the same time. Through the integration of infrastructure both for energy and information, this not only increases the energy utilizing efficiency, but also enhances the control ability of Energy Internet, and finally, this becomes the foundation of this algorithm proposed in this paper.

In many reality cases, some nodes need to reach a common state in distributed means, we call this procedure a consensus. The consensus algorithm can be used in a dynamically changing environment [8,9], such as time-varying topology [10, 11], time-varying delays [12], limited bandwidth [13], quantization [10, 14] or non-quantization, etc. It can reach consensus not only in linear [15, 16] motions and sometimes asynchronously [17, 18]. Some algebraic theories can be used in consensus algorithms such as least-mean-square [19]

• Yangyang Ming, Jie Yang, and Junwei Cao are with the Research Institute of Information Technology, Tsinghua University, Beijing 100084, China.
• Ziqiang Zhou is with the Electric Power Research Institute of State Grid Zhejiang Electric Power Company, Hangzhou 310009, China.
∗ Corresponding address: Junwei Cao, FIT 4-415, Tsinghua University. E-mail: jcao@tsinghua.edu.cn
Manuscript received: 2017-06-30; accepted: 2017-11-30

and LaSalle's Invariance Principle [20].

Distributed averaging is a special kind of consensus technologies been used widely such as in flight or vehicle formation [21], fire hazard control, network time synchronization, and so on. It is always executed in distributed means, without any centralized control.

Before distributed averaging, every node will have a different initial value and want to achieve the average value for some reasons. Through proper communication and calculation between neighbors, all connected nodes will reach consensus. And the consensus value will be the same as the average value precisely if there is neither quantization nor limited to integer value.

Gossips are typical distributed averaging algorithms [22]and have many types, such as random gossip [23], broadcast gossip [24] and geography gossip[25], and so on. In random gossip, a node communicated with one randomly selected neighbor and set each other's value to be their average. In geography gossip the neighbor node selected is not limited to one hop, so many hops' averaging is possible. In broadcast gossip, all neighbor nodes of the target node will update their value using the broadcast value they received. Through extending the node's selecting range, geography gossip will have faster convergence rate than random gossip more probably. By broadcast update, broadcast gossip will have fastest convergence value, but this algorithm can not keep the total value (or equally, average value) unchanged, so the consensus state may deviate from the average value.

The gossip algorithm can be used for the computations of some character value, like sums, averages, random samples, quantiles and other aggregation functions [26], which been used in energy sharing is new.

The remainder of this paper is organized as follows: the selection of gossip algorithm is briefly explained in chapter 2; then in chapter 3, the algorithm is described in detail; and related theorems on convergence are proved in chapter4; in subsequent two chapters, simulation and result, and further discussion is given. Finally in chapter 7, the whole conclusion is made.

2 Gossip selection

The basic theory of all nodes' convergence to average value is based on stochastic matrices and doubly-stochastic matrices. Where in stochastic matrices, every element is nonnegative and the sum of every row will equal to 1, that is:

$$\mathbf{A} * \mathbf{1} = \mathbf{1}$$

A is the stochastic matrices, **1** is a *n*1* vector of all ones.

This with connected fixed topology will guarantee the final value being converged. In doubly stochastic matrices, the sum of every row and column will be 1, so it satisfies both

$$\mathbf{A} * \mathbf{1} = \mathbf{1} and \mathbf{1}^T * \mathrm{A} = \mathbf{1}^T$$

$$and \mathbf{A}(i,j) \geq 0 and \mathbf{A}(i,j) \leq 1 for any i, j.$$

This with connected topology and some other proper constraints will not only guarantee convergence, but also ensure converging to the average value. In random and geography gossip, the transfer matrices will be doubly-stochastic. But in broadcast means, its transfer matrix only belongs to stochastic matrix. So the latter algorithm is not considered in this research.

In this paper, we firstly use a simple random gossip algorithm without concern the quantization effect. In this algorithm, one node is randomly selected, and the communication node will be selected as the one which having the maximum different value with the selected node, and then their value is averaged between the two. This algorithm is simple and proved to reach consensus with the average value. It should be mentioned that, although gossip is the basic algorithm used in our energy sharing algorithm, but it's not the research focus of this paper (just as a basic component). So the algorithm with higher performance can be selected later.

3 Algorithm description

3.1 Scene description

Based on the forecast of load and energy changes in Energy Internet, some grids may have spare energy, some others may be in energy shortage. Then the micro grids network can balances the energy demand and supply by using this algorithm. Based on the forecast time scale, the advanced time for calculation can be one day before, one hour before, or even five minutes before. Then before the time reaches, it schedules the energy transmission according to this algorithm. But what should be noted that, so far, it only regulates the total energy transmission in quantities for a period, but real detailed power transmission can also be executed by the same means in the future with the aiding of other proper algorithms, such as the one used in demand response (linear regulating).

Since we are using the theory of Cyber-Physical integration in infrastructure for energy and information, the information handling node and the energy transmission node are located and controlled together (energy router [27]), and the calculated energy sharing result can be directly applied to the energy transmission. This is the foundation of the algorithm proposed below.

3.2 Algorithm description (shown in figure 1)

1. Every node (micro grid) forecasts its energy supply and load demand for the next period. Each node can contain energy producing unit and/or energy store device, the latter one is called a combined node and treated equally with common node. The energy upper limit is the sum of local energy producing capacity and energy store quantities.
2. Set the load of every node as the required energy, and calculating the result of energy redundancy as Equation (1). The elements in Equation (1) are both vectors.

$$redundancy = energy - load \tag{1}$$

3. Set the value of the nodes which has positive energy redundancy to 1, and the negative ones to 0, then execute the distributed averaging algorithm, and get the final value *Tavg*. It will equal to the ratio of positive redundancy node number to the whole node number. (shown in figure 2)
4. Calculate the range of change factor k (used in Equation (7), verified by lemma 3) as

$$0 < k < 2/Tavg \tag{2}$$

5. Using the energy redundancy data, every node changes its value with its neighbors using selected gossip algorithm to reach consensus. In the proceedings, the energy transmitted along the transmission line during every average action is recorded and summed in every line. The related function is:

[flow_m1,temp_load]=distributed_avg(load_energy, link_m);

flow_m1 represents the transferred energy along the transmission line, *temp_load* is the averaged energy redundancy, *load_energy* is the energy redundancy been changed in last iteration, *link_m* is the topo matrix of links connecting the grid nodes.

In this function, the variables are changed as:

```
flow_m(m,j1)=flow_m(m,j1)+(temp_energy(1,m)-
temp_energy(1,j1))/2;
flow_m(j1,m)=flow_m(j1,m)-(temp_energy(1,m)-
temp_energy(1,j1))/2;
temp_energy(1,m)=(temp_energy(1,m)+
temp_energy(1,j1))/2;
temp_energy(1,j1)=temp_energy(1,m);
```

flow_m is the calculated flow in the average proceeding between node *j* to *m*.

temp_energy(1,m) and *temp_energy(1,j1)* is set to the average value between node *j* to *m*.

6. If this is the first iteration, and the calculated distributed averaging result is less than zero (meaning that the total *energy* is less than total *load*), we need to import external energy (mainly imported from backbone grid) to ensure nonnegative energy redundancy. The related initial values are changed accordingly and the algorithm is re-performed.
7. In other situations, we modify the energy and energy redundancy according to the distributed averaging result for energy redundancy and the change factor k. If the average result equals to zero, the supply and demand is balanced, so the iteration is terminated. If average result is positive, we reduce the energy and energy redundancy on selected nodes, otherwise if it is negative, we will increase the energy and energy redundancy on selected nodes. Which should be especially noticed that, the selected nodes are only limited to nodes with *energy>load*, and satisfies some other constraints, which ensures the convergence of the algorithm. The related code is:

```
if temp_load(1,i)>0 && load_energy(1,i)>0 &&
energy_m(1,i)>0
energy1=energy_m(1,i);
energy_m(1,i)=max((energy_m(1,i)-
temp_load(1,i)*k),0);
temp_load(1,i)=temp_load(1,i)-(energy1-
energy_m(1,i));
elseif temp_load(1,i)<0 && load_energy(1,i)>0 &&
energy_m(1,i)<load_m(1,i)
energy1=energy_m(1,i);
energy_m(1,i)=min((energy_m(1,i)+temp_load(1,i)*k),
load_m(1,i));
temp_load(1,i)=temp_load(1,i)+energy_m(1,i)-
energy1;
end
```

In the above code, *temp_load* is the average energy redundancy result, *load_energy* is the initial energy redundancy, *energy_m* is the energy allocation result in last iteration, *load_m* is the initial load in every node.

In the above code, when energy redundancy is larger than zero (with some other proper constraints), if *energy_m* minus *temp_load* is more than zero, then set *energy_m* equals to the minus result, otherwise, set *energy_m* to zero, and minus the changed energy (*energy1-energy_m(1,i)*)from *temp_load*. Similarly, when energy redundancy is less than zero (with some other proper constraints), if *energy_m* adds *temp_load* is no more than *load_m*, then then set *energy_m* equals to the adding result, otherwise, set *energy_m* to *load_m*, and adds the changed energy (*energy_m(1,i)-energy1*)to *temp_load*.

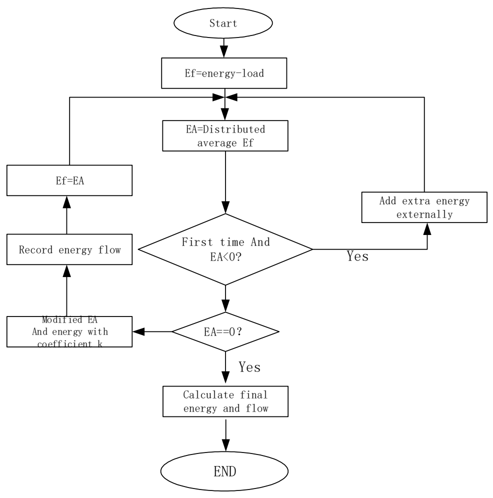
8. Based on modified energy redundancy, the distributed averaging algorithm is executed for next round. The iteration is stopped when distributed averaging result reaches to zero very closely, and the balance is reached.

9. When the algorithm terminates, the recorded flow in every iteration and every transmission line is summed, and the energy supply of every grid is also calculated using below code.

flow_m=flow_m+flow_m1;
energy_m=energy_m-load_energy;

flow_m1 is the changed energy flow in last iteration.

energy_m is the final energy provided by every node.



Start
Ef=energy-load
EA=Distributed average Ef
Ef=EA
Add extra energy externally
Record energy flow
First time And EA<0?
Yes
Modified EA And energy with coefficient k
EA==0?
Yes
Calculate final energy and flow
END

Fig. 1 the proceeding of this algorithm

4 Related theory prove

Some theories on algorithm's convergence characters are listed and proved below.

Lemma 1 If the total energy is more than total load in initial, and $k = 1$, then the algorithm finally converge to zero.

Proof If the total energy is more than the total load, there always exist positive energy redundancy nodes with number n, and $0 < n \leq totalnodenumber$. Then, if $k = 1$ and through step 7, the energy redundancy in every iteration keeps nonnegative (below the energy redundancy result calculated in k_{th} iteration is noted as $result[k]$). And at the same time, we can easily prove:

$result[k] > result[k+1] \geq 0$ if $result[k]! = 0$

So based on the limit theory, the algorithm will finally converge to zero.

■

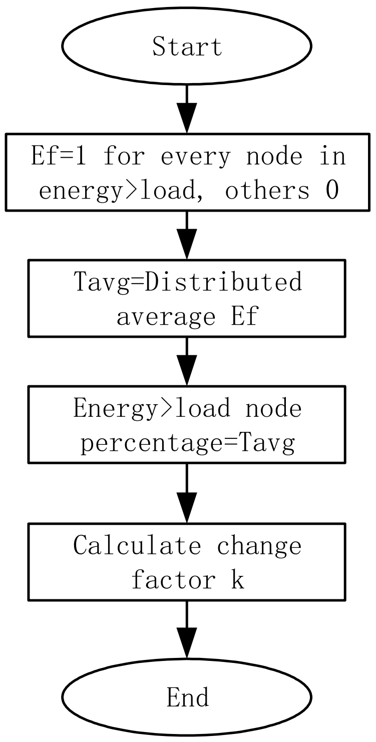
Lemma 2 If the total energy is more than total load in initial, then when $k < 2$, the algorithm finally converge to zero.

Proof We can get that the first calculated energy redundancy $result[1] > 0$ then the proof can be divided into two cases:

Case1: if $result[i] \geq 0$ for all i , like in Lemma 1, we can get $result[k] > result[k+1]$ if $result[k] > 0$, and finally converge to zero.

Case 2: otherwise there are $result[i] < 0$for some i;

As the first calculated energy redundancy $result[1] > 0$, there will always be $result[i+k] \geq 0$



Start
Ef=1 for every node in energy>load, others 0
Tavg=Distributed average Ef
Energy>load node percentage=Tavg
Calculate change factor k
End

Fig. 2 coefficient k calculation

$(k \geq 1)$ after $result[i] < 0$ for any i.

When $k < 2$, we can easily prove that:

1. If $result[i] < 0$ and $result[i+1] < 0$ for any possible iteration, then $|result[i]| > |result[i+1]|$
2. If $result[i] > 0$ and $result[i+1] < 0$ for any possible iteration, then $|result[i]| > |result[i+1]|$
3. If $result[i] < 0$ and $result[i+1] > 0$ for any possible iteration, then $|result[i]| > |result[i+1]|$
4. If $result[i] > 0$ and $result[i+1] > 0$, for any possible iteration, then $|result[i]| > |result[i+1]|$

Based on these foundation, we can get $|result[i]| > |result[i+1]| \geq 0$ for all conditions, so it will converge to zero.

From lemma 2, we can find this algorithm works for any initial conditions with $k < 2$.

■

Lemma 3 If the total energy is more than total load in initial, a possible range of k for which the iteration will converge, is $(0, 2/Tavg)$.

Proof As should be proved, when $|result[i]| > |result[i+1]|$, the calculated energy redundancy will converge to 0. We set

$$|result[i+1]| = |result[i]| + |delta[i+1]|$$

When $result[i] > 0$, then $delta[i+1] < 0$, and if $|result[i]| > 0.5 * |delta[i+1]|$, we can get $|result[i]| > |result[i+1]|$.

Similarly, when $result[i] < 0$ then $delta[i+1] > 0$, and if $|result[i]| > 0.5 * |delta[i+1]|$, we also get $|result[i]| > |result[i+1]|$.

So one sufficient condition for convergence is

$$|result[i]| > 0.5 * |delta[i+1]| \tag{3}$$

Proposed that the initial positive redundancy result node number equals to $Tavg * N$ (N is total node number) Base on expression 3, we have the sufficient condition (summed together)

$$|N * result[i]| > 0.5 * sum|delta[i+1]| \tag{4}$$

because

$$|Tavg * N * K * result[i]| \geq \sum |delta[i+1]| \tag{5}$$

We can set a more strict condition, which is

$$|N * result[i]| > 0.5 * |Tavg * N * K * result[i]| \tag{6}$$

As the number of nodes to be processed (changing redundancy) in every iteration is always no more than $Tavg * N$, so above Equation becomes a sufficient condition for the convergency of the algorithm.

Then we can prove if

$$Tavg > 0 \& K < 2/Tavg \tag{7}$$

the energy redundancy will converge to 0.

As $Tavg$ always $\leq$ 1,so lemma 2 is a special case of lemma 3.

■

5 Simulation and result

5.1 Topology set up

We randomly create a cluster of micro grids in Energy Internet with 7 nodes (micro grids), its simplified topo is shown in figure 3 $(G = (V, E))$. The circle nodes $(\in V)$ represent micro grid, the square nodes $(\in V)$ represent extra energy storage unit (forming the combine node with the circle nodes), the lines $(\in E)$ represent the energy transmission lines between the nodes, and they can be bidirectional in energy transmission. In order to run the energy sharing algorithm, we set up the corresponding connection topo matrix ($link\_m$) for this cluster. The nodes are notated from *1* to *n* respectively, and the size of connection matrix is 7*7.

If there is a transmission line connecting the node *i* and *j* , then set

$$link\_m[i, j] = link\_m[j, i] = 1$$

, otherwise the other elements are set to 0.

The link_m of figure 3 is:

$$\begin{bmatrix} 0,1,1,0,0,0,1 \\ 1,0,0,1,1,1,0 \\ 1,0,0,1,0,0,0 \\ 0,1,1,0,1,1,0 \\ 0,1,0,1,0,0,0 \\ 0,1,0,1,0,0,1 \\ 1,0,0,0,0,1,0 \end{bmatrix}$$

In order to record the energy flow between the connected nodes, we define the positive direction of the flow as from node with small node number to that with big node number, the negative value represents opposite direction.

Before simulation, we need to assign the energy supply and load demand to every node (as in figure 4). When doing this, we can add a priority level to every micro grid. If there are important devices in the micro grid, the priority will be high, and the grid can set some energy redundancy in the initial value set if there are

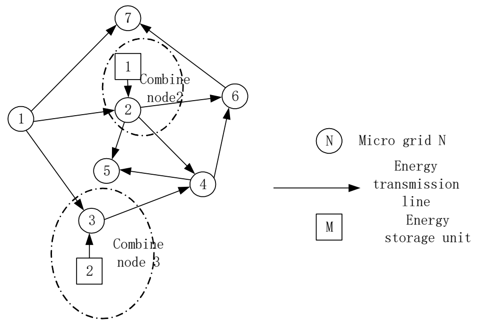


Fig. 3 micro grid topology

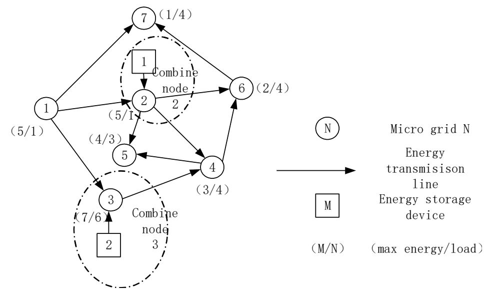


Fig. 4 Initial energy/load set

any energy storage devices.

Otherwise the priority may be low, and without energy redundancy. At the same time, the needs of demand response or demand side management can be considered accordingly by modifying the initial energy and load value.

In order to simplify the algorithm proceeding, we set the initial total energy supply to be more than the initial total load demand, this will not change the spirit of the algorithm.

The initial energy vector is [5,5,7,3,4,2,1], the load vector is [1,1,6,4,3,4,4]. Where the i_{th} value of the two vectors corresponding to the value of node i.

5.2 Gossip algorithm test

We need to test the convergence performance of the gossip algorithm at first. We set the load demand as $rand(1,7) * 100$ and energy supply as $rand(1,7) * 100 + rand(1,7) * k1$, the coefficient $k1$ is set as 0,20,50,100,200 individually. Then we calculate the distributed averaging value of *energy supply minus load demand.*

Then we calculate the averaged abs error in every iteration as:

$$
\begin{aligned}
&n = |initial\_value|_0 \\
&avg\_value = sum(initial\_value)/n \\
&abs\_error = sum(abs(value - avg\_value))/n
\end{aligned}
\tag{8}
$$

Here $|initial\_value|_0$ represents the number of vector $initial\_value$, and vector *value* represents the averaging result of $initial\_value$ in every iteration.

When $abs\_error < k$, the iteration terminates.We can find if k keeps unchanged, although the $k1$ changes largely, the ranges of results are similar, this will ease the simulation. If we set $k = 0.1$, all the simulation examples converge to the average value before about 48_{th} iterations. If we set $k = 0.01$, the iteration round number is about 70_{th}. If we set $k = 0.001$, the value is about 95_{th}. Figure 5-8 show some results of statistic histogram for each iteration turns running 100 times. The horizontal axis represents iteration turns, the vertical axis represents emerged frequency.

5.3 Simulation and result

First, we calculate the $Tavg$ in Step3 using the gossip algorithm(input vector:[1,1,1,0,1,0,0]), we can get the result vector[0.5714, 0.5714, 0.5714, 0.5714, 0.5714, 0.5714, 0.5714] for 100 rounds, which equals to the ratio of initial number of positive energy redundancy nodes to the whole nodes number(4/7) in above topology, which verifying the step3.

Before running the algorithm, we first set the change factor k, as the $Tavg = 4/7$, according to lemma 3, we can get if k less than 3.5, the correct result will be derived. (But which should be noted that it's not a necessary condition) So we set the k vectors as [1,1.2,1.5,1.6,1.8,2,3,3.4] and runs them in turn (k less than 1 is unnecessary, it only reduces the convergence rate).

We run the energy sharing algorithm on the topology designed above, and then on that of deleting one node or deleting one edge. All scenes show the desired results (figure 9 to figure 13).

From above results, we can see that the total energy assigned for every node is the algebraic sum of the self-providing energy and energy transmitted from neighbor nodes (add it), and energy transmitting to neighbor nodes (minus it), which equals to the load of every node.

As can be seen, if the total initial energy is bigger than total initial load, the final energy value is always less than the max energy which can be provided by every

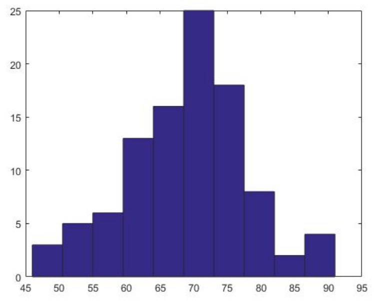


Fig. 5 k=0.001, k1=0

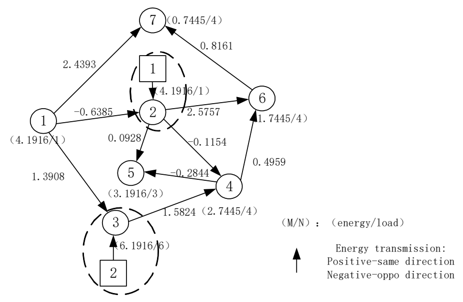


Fig. 9 normal state simulation k=1.2

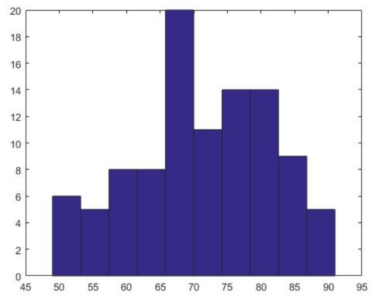


Fig. 6 k=0.001,k1=20

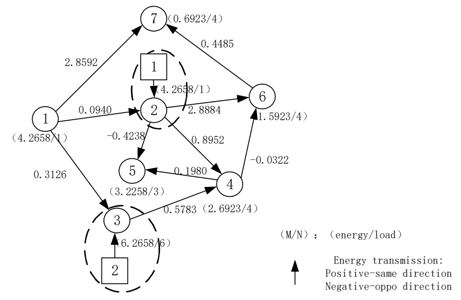


Fig. 10 normal state simulation k=1

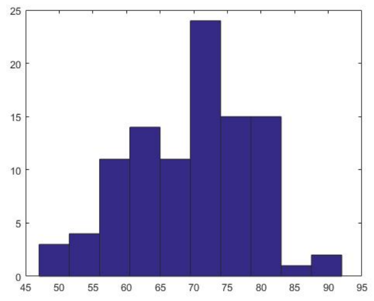


Fig. 7 k=0.001,k1=50

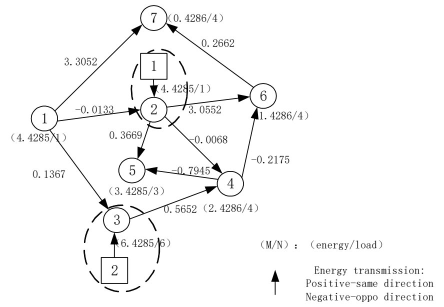


Fig. 11 normal state simulation k=2

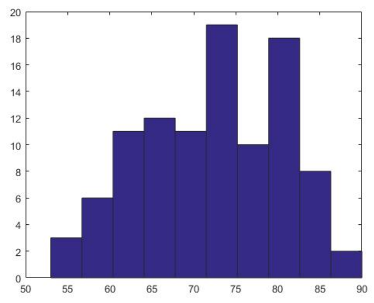


Fig. 8 k=0.001,k1=200

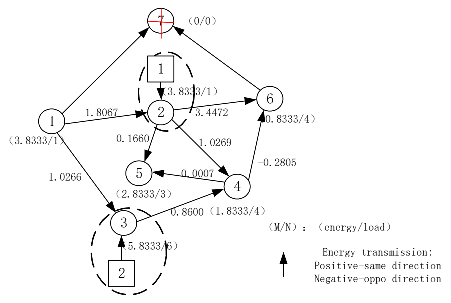


Fig. 12 node failure state simulation k=2

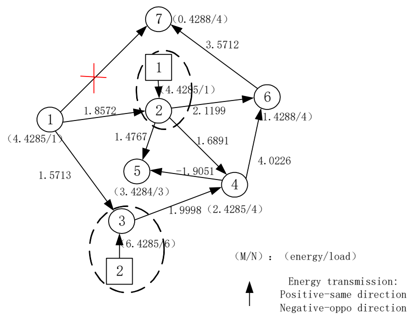


Fig. 13 line failure state simulation k=2

node, which means there are energy spare in every node. This is an especial advantage for energy sharing in micro grids network, as this energy redundancy can be used to face emergency situation, or charge the energy storage unit, which can enhance the steady of the whole network and improve the robustness of the micro grid network.

The simulation also shows the convergence phenomenon for different change factors (k) in figure 14. We can see that when the change factor is small, the energy redundancy monotonically reduced. As the change factor increases, the value of first iteration decreases and there are some fluctuation around the zero axis when $k > 1.6$, which coincide with theory analysis. That is, if $k \geq 1/Tavg$, then there will emerge negative result with high probability. All iterations end before 7 turns. And the instance which have negative results often converges faster than only positive ones, except $k = 1$.

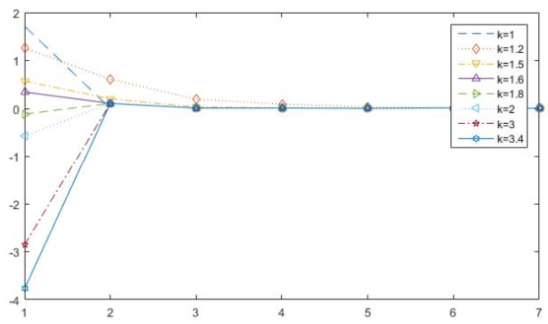


Fig. 14 energy redundancy in every iterations with different k

Though deeply observation and deducing, we can get that if the initial condition (demand and supply) and the change factor in two simulations are both the same, the final energy allocation in every grids will be same too, but the energy transmitted between the grids will be different. At the same time, if both of the initial energy and load equals in a pair of nodes (such as node1 and node2), the final energy load of this pair will be equal, this can be proved by theoretical analysis.

These two cases can be observed for all $k > 1$, the only exception is in $k = 1$, which may be due to the effect of averaging residue of gossip in positive node selection scheme and influences subsequent handling result, but the algorithm still converges and the result is reasonable. So it's easy to analysis the performance of the algorithm in every instance by only one example.

Excluding exception $k = 1$(although it has the same trend listed below), we get the final result in Table 1.

Here the sign (+) means initial supply is more than initial demand in this node, and the sign (-) means the opposite conditions.

From the table combined with the result of Fig.11, we can see that if $k < 1.8$ (energy redundancy is monotone decreasing), with the factor k increasing, the energy provided by nodes with sign (+) monotonically decreases, while the ones with sign (-) monotonically increases (equally speaking, the variance of vector *energy-load* for all nodes decreases with k. Otherwise, if $k >= 1.8$(with energy redundancy fluctuation), however the factor k changes, the energy allocation keeps almost no changed. This result can lead us to properly choose the change factor k to satisfy the extra energy demand.

6 Further discussion

6.1 Executing gossip in parallel

Our used gossip algorithm can be executed in parallel, when two neighbor nodes agree to average, it can be executed in parallel with other nodes' execution. When there are two nodes selecting the same neighbor node, a simple OK or reject would be enough, as here mainly uses wired communication.

6.2 Communication and bandwidth constraint

As the communication media between micro grids mainly use fiber optical, the topo change will be slowly or constant, and the communication power and bandwidth constraint is not as tight as the wireless channel, so no quantization is needed. And the needed characters of topology can be very simple (just ensure properly connected).

6.3 Gossip algorithm termination condition

When execute the gossip algorithm in distribute means, every node should know when to terminate the iteration.

Here, every node can broadcast its value to its one hop neighbors if it changed. When the node found that its value and its neighbors' value keeps no changed for a limited time period, then it can terminate the iteration locally.

6.4 Distributed calculating and central processing

Although the energy sharing algorithm is using distributed computing technologies, but it can also be used in central processing. If there are central processing unit, it can gather every grid's energy and load forecast data and use the iteration algorithm to calculate the energy allocation and get the energy transmitted along every lines. The central processing needs only one iteration through setting total supply equals to total demand.

6.5 State renewing

As the energy sharing is linear, so when the energy and load changes, we can either using enhanced learning or an update calculation to re-calculate the network energy sharing result. And the two means are the same in spirit, just should be noticed is that the enhanced learning should also satisfy the energy constraint.

7 Conclusions

In this paper a completely distributed iteration algorithm is designed for energy sharing in typical Energy Internet situation. Through properly designed algorithm proceeding and change factor selection, the allocated energy and load requirement will be finally balanced as proved, which verifies the performance of the algorithm. The algorithm is tested in different conditions and reasonable results are observed. The future research direction will consider the real constraints on energy transmission (such as capacity constraint of the transmission line) or satisfying some special transmission demands proposed by the owner (such as demand side management). And also, the real performance evaluation between different gossip algorithms will be compared.

Acknowledgment

This work was supported in part by National Natural Science Foundation of China (grant No. 61472200), Beijing Municipal Science & Technology Commission (grant No. Z161100000416004), and the project of Blockchain Application Research on Energy Internet(grant No. 52110417000G).

References

[1] J CAO and M YANG, Energy Internet – towards smart grid 2.0, presented at Fourth International Conference on Networking and Distributed Computing (ICNDC), Los Angeles, CA, USA, 2013.

[2] J CAO, Y WAN and G TU, et al, Information system architecture for smart grids, *Chinese Journal of Computers,* 2013,vol. 36, no. 01, pp. 143-167, 2013.

[3] AQ Huang, ML Crow, GT Heydt, et al, The future renewable electric energy delivery and management (FREEDM) system: the Energy Internet, in *Proceedings of the IEEE,* vol. 99, no. 1, pp. 133-148, 2011.

[4] Z DONG, J ZHAO, F WEN, et al, From smart grid to Energy Internet: basic concept and research framework, *Automation of Electric Power Systems,* vol.38, no. 15, pp. 1-11, 2014.

[5] J Rifkin, *The Third Industrial Revolution*. BeiJing: China Citic Press, 2012.

[6] Y Wan, J Cao, S Zhang, et al, An integrated Cyber-Physical simulation environment for smart grid applications, *Tsinghua Science and Technology,* vol. 19, no. 02, pp. 133-143, 2014.

[7] J CAO, M YANG, D ZHANG, et al, Energy Internet – an infrastructure for Cyber-Energy integration, *Southern Power System Technology,*vol. 8, no.4, pp.1-10, 2014.

[8] A Olshevsky , and J N Tsitsiklis, Convergence speed in distributed consensus and averaging, presented at 45th IEEE Conference on Decision and Control, San Diego, USA, 2006.

[9] M Cao, A S Morse, and B D O Anderson, Reaching a consensus in a dynamically changing environment – convergence rates, measurement delays and asynchronous events, *Siam Journal on Control & Optimization,*vol. 47, no. 2, pp.601-623, 2008.

[10] S Kar, and J M F Moura, Distributed consensus algorithms in sensor networks: quantized data and random link failures, *IEEE Transactions on Signal Processing,*vol. 58, no. 3, pp. 1383-1400, 2007.

Table 1 Energy allocation for different k and nodes

k/node	1(+)	2(+)	3(+)	4(-)	5(+)	6(-)	7(-)
1.2	4.1916	4.1916	6.1916	2.7445	3.1916	1.7445	0.7445
1.5	4.0976	4.0976	6.0976	2.8698	3.0976	1.8698	0.8698
1.6	4.0703	4.0703	6.0703	2.9062	3.0703	1.9062	0.9062
1.8	4.4286	4.4286	6.4286	2.4286	3.4286	1.4286	0.4286
2	4.4285	4.4285	6.4285	2.4286	3.4285	1.4286	0.4286
3	4.4286	4.4286	6.4286	2.4286	3.4286	1.4286	0.4286
3.4	4.4286	4.4286	6.4286	2.4286	3.4286	1.4286	0.4286

[11] Y Hatano, and M Mesbahi, Agreement over random networks, *IEEE Transactions on Automatic Control,* vol. 50, no. 11, pp. 1867-1872, 2005.

[12] F XIAO, and L WANG, Asynchronous consensus in continuous-time Multi-Agent Systems with switching topology and time-varying delays, *IEEE Transactions on Automatic Control,*vol. 53, no. 8, pp. 1804-1816, 2006.

[13] T LI, M FU, L XIE, et al, Distributed consensus with limited communication data rate, *IEEE Transactions on Automatic Control,*vol. 56, no. 2, pp. 279-292, 2011.

[14] T C Aysal, M J Coates, and M G Rabbat, Distributed average consensus with dithered quantization, *IEEE Transactions on Signal Processing,* vol. 56, no. 10, pp. 4905-4918, 2008.

[15] W YU, G CHEN, M CAO, et al, Second-Order consensus for multiagent systems with directed topologies and nonlinear dynamics, *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics,*vol. 40, no. 3, pp. 881-891, 2010.

[16] P LIN, and Y JIA, Consensus of second-order discrete-time Multi-Agent Systems with nonuniform time-delays and dynamically changing topologies, *Automatica,*vol. 45, no. 9, pp. 2154-2158, 2009.

[17] M CAO, A S Morse , and B D O Anderson, Agreeing asynchronously, *IEEE Transaction on Automatic Control,*vol. 53, no. 8, pp. 1826-1838, 2008.

[18] C C Moallemi, and R B Van, Consensus propagation, *IEEE Transactions on Information Theory,*vol. 52, no. 11, pp.1-13, 2006.

[19] L Xiao, S Boyd, and S J Kim, Distributed average consensus with Least-Mean-Square deviation, in *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS)*, Kyoto, Japan, 2006, pp.1-9.

[20] D CHENG, J WANG, and X HU, An extension of LaSalle's invariance principle and its application to multi-agent consensus , *IEEE Transaction on Automatic Control,*vol. 53, no. 7, pp. 1765-1770, 2008.

[21] W Ren, and R W Beard, Distributed consensus in multi-vehicle cooperative control, *Springer London,*vol. 27, no. 2, pp. 71-82, 2008.

[22] A Nedic,A Olshevsky,A Ozdaglar, et al, On distributed averaging algorithms and quantization effects, *IEEE Transactions on Automatic Control,* vol. 54, no. 11, pp. 2506-2517, 2009.

[23] S Boyd, A Ghosh, B Prabhakar, et al, Randomized gossip algorithms, *IEEE Transactions on Information Theory,*vol. 52, no. 6, pp. 2508-2530, 2006.

[24] T C Aysal , M E Yildiz , A D Sarwate , et al, Broadcast gossip algorithms for consensus, *IEEE Transactions on Signal Processing,*vol. 57, no. 7, pp. 2748-2761, 2009.

[25] A D G Dimakis, A D Sarwate, and M J Wainwright, Geographic gossip: efficient averaging for sensor networks, *IEEE Transactions on Signal Processing,*vol. 56, no. 3, pp. 1205-1216, 2007.

[26] D Kempe ,A Dobra, and J Gehrke, Gossip-based computation of aggregate information, presented at 44th Annual IEEE Symposium on Foundations of Computer Science, Cambridge, Massachusettes,2003, pp. 482-491.

[27] Y XU,J ZHANG, W WANG, et al, Energy router: architectures and functionalities toward Energy Internet, presented at The 2nd IEEE International Conference on Smart Grid Communications, Brussels, Belgium, 2011, pp. 31-36.

Yangyang Ming received bachelor's degree in Nanjing university of posts and telecommunications in 2004; received master's degree in Chongqing university of posts and telecommunications in 2007; received PhD degree in Beijing university of posts and telecommunications in 2012; Ming worked as a postdoctor in Tsinghua university till now.

Jie Yang achieved her bachelor's degree at Jinan University in 2007, a master's degree in 2010 and PhD in Beijing Institute of Technology in 2015. Since 2016, she serves in Tsinghua University as a Postdoctoral Researcher.

Junwei Cao achieved his bachelor's degree at Tsinghua University in 1996, master's degree in 1998 and PhD in University of Warwick in 2001. From 2002 to 2006, he worked in the NEC European Laboratory in Germany and the MIT/LIGO laboratory of the United States.Cao has been working at the Tsinghua University since 2006 as Professor and Vice Dean of the Research Institute of Information Technology. He has published more than 200 academic papers and 8 books.