TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214
DOI: 10.26599/TST.2020.9010019
Volume 26,

Improved Approximate Minimum Degree Ordering Method and Its Application for Electrical Power Network Analysis and Computation

Jian Guo, Hong Liang, Songpu Ai, Chao Lu, Haochen Hua, and Junwei Cao*

Abstract: Electrical power network analysis and computation play an important role in the planning and operation of the power grid, and they are modeled mathematically as differential equations and network algebraic equations. The direct method based on Gaussian elimination theory can obtain analytical results. Two factors affect computing efficiency: the number of nonzero element fillings and the length of elimination tree. This article constructs mapping correspondence between eliminated tree nodes and quotient graph nodes through graph and quotient graph theories. The Approximate Minimum Degree (AMD) of quotient graph nodes and the length of the elimination tree nodes are composed to build an Approximate Minimum Degree and Minimum Length (AMDML) model. The quotient graph node with the minimum degree, which is also the minimum length of elimination tree node, is selected as the next ordering vector. Compared with AMD ordering method and other common methods, the proposed method further reduces the length of elimination tree without increasing the number of nonzero fillings; the length was decreased by about 10% compared with the AMD method. A testbed for experiment was built.The efficiency of the proposed method was evaluated based on different sizes of coefficient matrices of power flow cases.

Key words: Approximate Minimum Degree and Minimum Length (AMDML); electrical power network analysis; elimination tree; numerical solution; ordering method

1 Introduction

Nowadays, renewable energy resources, such as wind and solar, account for increasing proportion of electricity supply and they are gradually changing the traditional transmission mode of energy[1], wherein prosumers (self-produced and consumed units) in local microgrids play a significant role. With the deployment of large-capacity energy storage devices and the application of advanced information and communication technologies, a prototype of layered multilevel Energy Internet (EI) has formed[2–4]. The operation of EI will cause more energy flow within source-grid-load-storage, and the operating scenarios in practice are becoming more complicated. This issue requires a fine-grained modeling of various components. On the one hand, planning and operation shall be guided by improving the observability and controllability of the power grid[5]. On the other hand, the current operation status of the power grid needs to be obtained in real time, and weak sections should be identified accurately[6]. All the aforementioned requirements need real-time calculations of various power system online analysis applications such as power flow, transient stability, state estimation, and Dynamic

• Jian Guo, Hong Liang, Songpu Ai, and Junwei Cao are with Beijing National Research Center for Information Science and Technology, Beijing 100084, China. Jian Guo is also with the Department of Electrical Engineering, Tsinghua University, Beijing 100084, China. E-mail: guoj2019@tsinghua.edu.cn; lianghong@mail.tsinghua.edu.cn; aisp@mail.tsinghua.edu.cn; jcao@tsinghua.edu.cn.
• Chao Lu is with the Department of Electrical Engineering, Tsinghua University, Beijing 100084, China. E-mail: luchao@tsinghua.edu.cn.
• Haochen Hua is with the College of Energy and Electrical Engineering, Hohai University, Nanjing 211100, China. E-mail: huahc16@tsinghua.org.cn.
* To whom correspondence should be addressed.
Manuscript received: 2020-02-19; revised: 2020-04-19; accepted: 2020-04-19

Stability Analysis (DSA).

Research achievements have been made in terms of improving the calculation performance of the power system online analysis[7]. One way to improve the calculation performance is to apply computer science and network communication technologies to achieve high-performance calculation. The idea of this method is that the power grid is divided into small blocks on the basis of link relationship, which could run in parallel and interact data with each other on the boundary area. Theoretically, a large number of calculation tasks can be decomposed into small tasks, which can be distributed into several processors and calculated simultaneously. Necessary boundary coordination data should be exchanged synchronously, thereby reducing the computation time under the premise of convergence accuracy. Commonly used methods include specific application-oriented block strategies and synchronous coordination modes, as well as using more advanced computing architectures and optimized storage structure for parallelization.

From the perspective of numerical solution, problems of electrical power network analysis are attributed to solutions of sparse linear equations[8–10]. Two methods, namely, iterative and direct methods, are used to solve linear equations formed from electrical power system analysis, and parallelization is used to speed up computation[11]. The iterative method involves only matrix-vector multiplication computation, which has intrinsic parallelism. However, this method is sensitive to the condition number and the eigenvalue distribution of the coefficient matrix, which limited its applications. The direct method relies on the principle of Gaussian elimination, which forms a triangular factor matrix and obtains results through forward/backward operations. Non-zero fillings will be injected during elimination, which requires additional storage space. The ordering method determines the number of nonzero fillings[12]. Adopting an optimized ordering approach before elimination can reduce nonzero fillings, thereby improving calculation efficiency.

The main contributions of this paper are summarized as follows:

(1) Numerical solutions in power electrical network analysis are discussed, including computing types, process, and methods. Moreover, the role of the elimination tree in numerical calculations and its effect on improving the computing efficiency are extensively studied.

(2) On the basis of graph theory and the modeling of quotient graph theory, an Approximate Minimum Degree Minimum Length (AMDML) computation model is built, and AMDML method is proposed. Time and space complexities of instance are discussed.

(3) Indices of fill ratio, length and average length based on the elimination tree, which represent the computation efficiency of matrix decomposition, are built.

(4) A testbed for electrical power network computation is built and coefficient matrices of power flow cases are used to test the efficacy of our proposed method. Compared with that of Minimum Degree (MD), Minimum Degree Minimum Length (MD-ML), and Approximate Minimum Degree (AMD) ordering methods, the superiority of our method in terms of time is verified.

2 Literature Review

2.1 Numerical methods for electrical power network analysis

Applications for power electrical network analysis can be classified as online and offline[13]. Offline applications are used for system planning and operation, which need a large amount of computation without real-time requirements. Online applications are used for online system operations and economic dispatch, which need to obtain results in a timely manner[14]. Both offline and online applications contain similar functions, including power flow analysis, short circuit calculations, N-1 contingency analysis, Optimal Power Flow (OPF) analysis, and transient stability analysis[15–17]. The offline and online methods differ in terms of their computational efficiency and accuracy. On the basis of the level of details of the underlying power system models, power system computations include steady-state analysis, Dynamic Security Analysis (DSA), and optimizations considering generator/transmission constraints. Figure 1 shows a diagram of power system computational analysis.

From the perspective of numerical computations, the static Security Analysis (SA) and Short Circuit (SC) analysis both involve the knowledge of graph theory and matrix theory for processing topology changes in power networks, because they are based on the state estimation results and power flow calculations. These problems can be viewed as the process of solving sparse linear equations because of the characteristics of physical

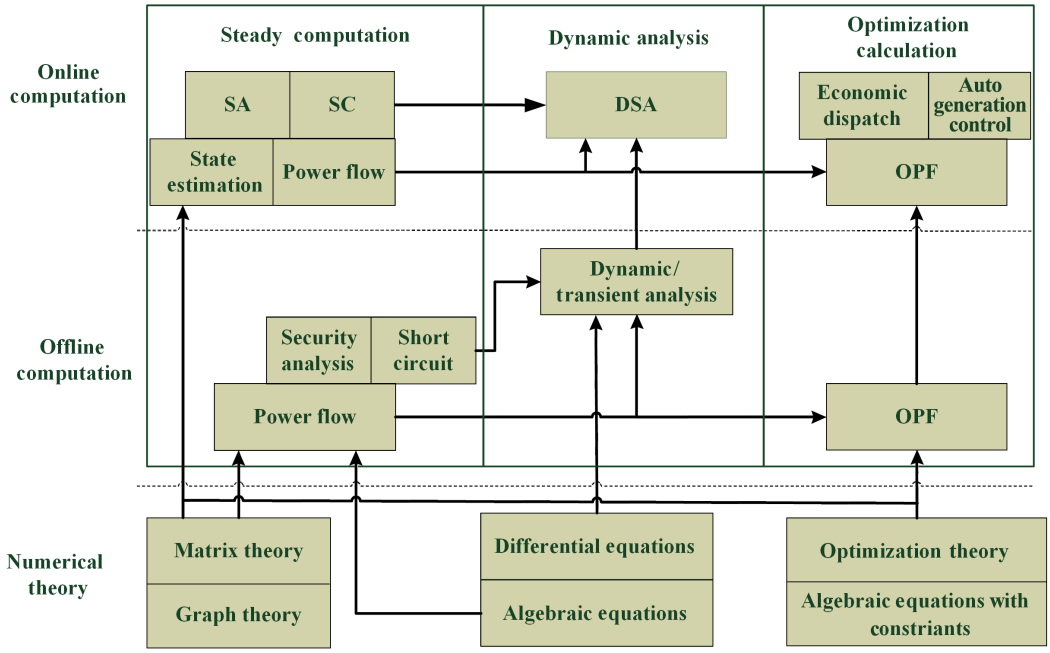

Fig. 1 Diagram of functional maxIterations.

power grids. Transient stability analysis involves the modeling of generators, loads, and power electronics devices. Thus, it can be viewed as the problem of constructing differential equations and solving the linear sparse equations by using iterative approaches. OPF analysis forms constraint vectors/equations by using generator/transmission operating conditions and solves optimization problems, such as economic dispatch and automatic generation control. Table 1 shows different types for power system numerical computations.

On the basis of the above discussions, numerical computations for power electrical networks consist of three modules: the computational algorithm, the data preparation, and the solution of sparse linear equations by using direct methods. Figure 2 shows three modules of power system numerical computations. As for the module of the computational algorithm, static applications, such as OPF, power flow analysis, state estimation, SC analysis, and N-1 security analysis, involve different adjustments after data preparation/preprocessing and sparse linear equation solutions. State estimation, OPF, and power flow analysis need to adjust constraints/initial vectors. N-1 analysis and SC analysis need to adjust parts of the network admittance matrix. Transient stability analysis, involving dynamic models of generators and load, requires dynamic model parameters to be adjusted after each computation step. The data preparation/preprocessing module provides data for the module of a sparse linear function solution and sends the adjusted results back to the module of the computational algorithm for follow-up computations. The data preparation/preprocessing module is mainly responsible for data management and maintenance during the computation process. The module of solving sparse linear equations by using direct methods contains triangular decomposition and forward-backward substitution, and it consumes the most computational time.

Table 1 Numerical computing types of power system.

Application	Type of numerical solutions
N-1 sercurity analysis	Factorization/sparse linear equations
Short circuit	Factorization/sparse linear equations
Transient computation	Differential/sparse linear equations
Optimal power flow	Sparse linear equations with constraints
Power flow	Sparse linear equations
State estimation	Sparse linear equations

2.2 Elimination tree analysis in numerical computation

The techniques for solving large-scale linear equations, especially sparse linear equations, are critical for many scientific and engineering applications[18]. The methods for large-scale sparse linear equations include iterative methods and direct methods for finding solutions. Iterative methods have limited real-world applications because of issues of numerical stability and computational efficiency. Direct methods decompose

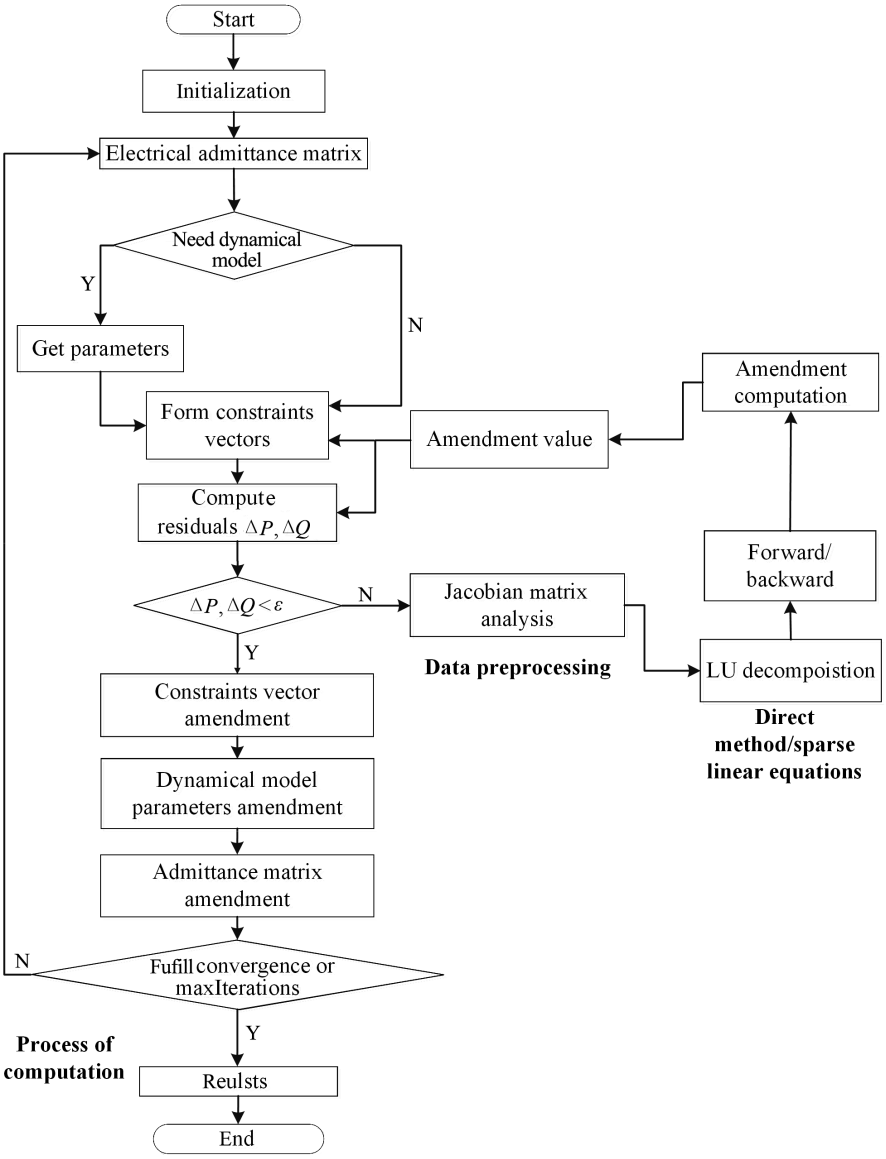

Fig. 2 Numerical solutions for electrical network analysis.

the sparse matrix by using LU, Cholesky, and QR decompositions to form the upper and lower triangular matrices. After decomposition, direct methods then apply forward/backward substitution and obtain numerical and analytical solutions to the linear sparse equations. These direct methods are more popular in real-world applications. Direct methods for solving sparse linear equations are based on the theory of Gaussian elimination. These methods form the upper and lower triangular matrices and obtain exact analytical solutions by using forward-backward substitution. The direct methods involve the following four steps:

(1) The sparse matrix is ordered through row/column interchanges to reduce the number of non-zero matrix blocks and ensure the matrix is diagonal dominant.

(2) Symbolic factorization is performed, the positions for filling in the nonzero matrix elements are confirmed, memories are preallocated, and the data storage structures are created and initialized.

(3) Numerical decomposition is performed to form the upper triangular matrix and lower triangular matrix.

(4) On the basis of the matrices obtained in Step 3, the right-hand-side terms are reordered through Step 1, forward-backward substitution is performed, and solution results are obtained.

In the above steps, Step 2 performs symbolic factorization for the matrix indicating positions of non-zero elements. During right-looking elimination, when the *i*-th row of the matrix eliminates the nonzero element in the *j*-th row of the matrix (i.e., the non-zero element

is located at the k-th column, where $i < j, k > j$), if $\boldsymbol{A}(i,k) \neq 0, \boldsymbol{U}(j,k) \neq 0$. Taking the matrix in Fig. 3a as an example, the matrix in Fig. 3b denotes the newly filled nonzero elements after symbolic factorization. Figure 3c shows the corresponding elimination tree.

The elimination tree can represent the column elimination relationship during the triangular decomposition. It also determines the computation orders for the forward-backward substitution. To improve the computational efficiency of the direct methods, one needs to reduce the number of nonzero element fillings induced by the decomposition on the one hand, and reduce the length of the elimination tree formed by the parameter matrix after the decomposition on the other hand. A common approach for reducing the number of nonzero element fillings is to re-order matrix by the way of basic rank transformation[19]. Then heuristic algorithms with the concept of node degree in graph theory for ordering the minimal degree nodes were also proposed[20]. But minimal degree node ordering problem is demonstrated to be a NP-hard problem[21]. The above methods do not consider the impact of node ordering methods on the length of the elimination trees. Then the length of the elimination tree is defined as an attribute, and the length calculation method is proposed[22]. This method leads to the ordered sequence with a minimal length. However, the number of nonzero element fillings caused by this method is significantly larger than that caused by the MD method. The method has low computational efficiency and practicality. With the corresponding length of the elimination tree considered as a constraint, this paper proposes the AMDML ordering model and the computation algorithm. Numerical results show that compared with the elimination-graph-based MD ordering method and quotient-graph-based AMD ordering method, the proposed method can further reduce the number of nonzero element fillings, decrease the length of the elimination tree, and increase numerical computational efficiency, when it is applied to matrices related to power system computational applications.

3 Definition

Definition 1 Let us define G as reduction graph: $G = (N, E)$, where N represents the set of nodes in graph G and E represents the set of edges in graph G. From this, the subreduction graph of G, $G(C) = (C, E(C))$, where $C \subseteq N$, $E(C) = \{\{u, v\}$ belongs to $E | u, v$ belongs to $C\}$.

Definition 2 Let us define Q as quotient graph,

$$Q = (P, E(P)),$$

$$P = \{\{P_i\} | \bigcup_{i=1}^{q} P_i = N, P_m \cap P_n = \Phi (m \neq n)\},$$

$$E(P) = \{P_m \cap \mathrm{adj}(P_n) \neq \Phi | P_m \neq P_n\} \quad (1)$$

where Q represents a quotient graph, P represents the set of reduction graph nodes within the quotient graph node, $E(P)$ represents the set of edges between nodes in the quotient graph, and $i, m, n \in \mathbf{N}$.

Definition 3 Let us define *X(S)* as the division of N on S,

$$\bar{X}(S) = X(S) \cup \{\{x\} | x \notin S, x \in N\} \quad (2)$$

It is obtained from the cutset of the subset S derived from the reduction graph G,

$$X(S) = \{P \subset S | G(P) \text{ is connected in } G(S), S \subset N\} \quad (3)$$

That is the remaining nodes that belong to the cutset of S and the reduction graph G do not belong to S.

For example, graph G, as shown in Fig. 4a, if $X(S) = \{\{a, b, c\}, \{d, e\}, \{g, h\}\}$, the obtained quotient graph is $Q = G/X(S)$, as illustrated in Fig. 4b.

Factorization of the coefficient matrix on the

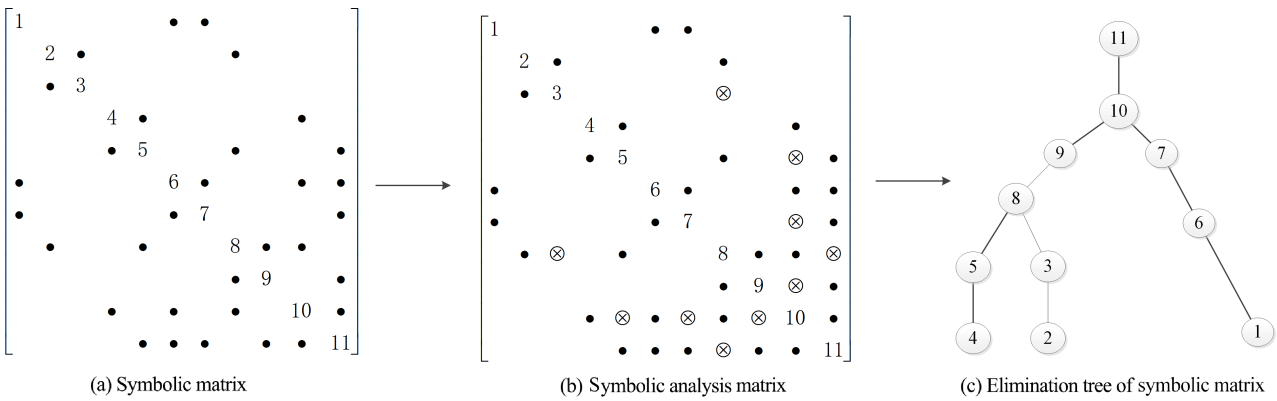

Fig. 3 Elimination tree diagram.

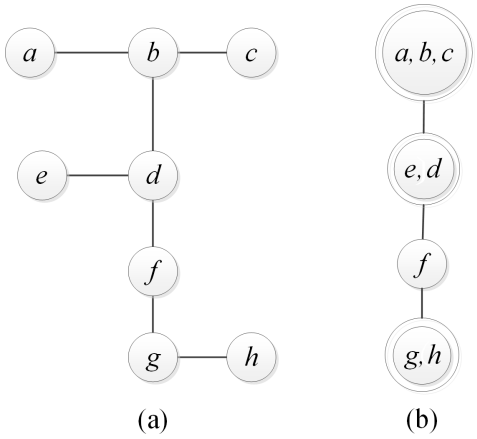

Fig. 4 Example of quotient graph.

quotient graph requires the establishment of a quotient graph model $Q^k = (N^k, P^k, E^k, E(P)^k)$, where k represents the number of reduction steps, $E \subseteq N \times N$, $E(P) \subseteq N \times P$. When $k = 0, Q^0 = G^0, N^0 = N, P^0 = \varnothing, E^0 = E$, and $E(P)^0 = \varnothing$.

From this point of view, a reduction graph can be regarded as a special case of a quotient graph, that is the elements of a quotient graph are the elements of a reduction graph, when the number is 1. The correspondence relationship between a reduction graph and a quotient graph needs to be specified to realize the elimination operation based on the quotient graph. A quotient model $Q^k = (N^k, P^k, E^k, E(P)^k)$ needs to be established to perform the ranking analysis on the quotient graph, which includes both the set of quotient graphs and the set of reduction graphs, where k represents the number of elimination steps, $E \subseteq N \times N, E(P) \subseteq N \times P$. When $k = 0, Q^0 = G^0, N^0 = N, P^0 = \varnothing, E^0 = E, E(P)^0 = \varnothing$.

4 Improved AMD Ordering Method—AMDML

4.1 Model

First, a calculation model AMDML $= (A, M, L, H)$ should be built to use the AMDML method on quotient graph Q, This model is established to represent the change of node connection and node length after each elimination step. The definition of the model is as follows:

$$\begin{aligned} &A_i = \{j\,|(i,j) \in E, i \in N\} \subseteq N,\\ &M_i = \{e|(i,e) \in E(P), i \in N\} \subseteq P,\\ &L_e = \mathrm{adj}_Q(e) = \{i\,|(i,e) \in E(P), e \in P\} \subseteq N,\\ &\mathrm{adj}_Q(i) = (A_i \cup M_i) \subseteq N \cup P \end{aligned} \tag{4}$$

where A_i represents the set of nodes connected to node i in the graph to be reduced, M_i represents the set of nodes connected to node i in quotient graph, L_e is the set of nodes in the graph to be reduced connected to the set of nodes e of quotient graph, and $\mathrm{adj}_Q(i)$ is the set of nodes connected to node i in both quotient graph and the graph to be reduced. The relation between the length of nodes is as follows:

$$\begin{aligned} &H_i = \{H(i)|\mathrm{DFS}(i, \text{root of } N)\},\\ &\mathrm{adj}_G(i) = (A_i \underset{e \in M_i}{\cup} L_e) \setminus \{i\} \end{aligned} \tag{5}$$

where H_i represents the length of the elimination tree of node i, which is traversed by the depth function DFS$(\cdot)$, and $\mathrm{adj}_G(i)$ is the set of nodes connected to node i in the graph to be reduced. Thus, during the elimination process the sets A^k, M^k, and L^k are derived as follows:

$$\begin{aligned} &A_k = (A^{k-1} \setminus (L_s \times L_s)) \cup (N_k \times N_k),\\ &M_k = (M^{k-1} \setminus \underset{e \in M_s}{\cup} e) \cup \{s\},\\ &L_k = (L^{k-1} \setminus \underset{e \in M_s}{\cup} L_e) \cup L_s \end{aligned} \tag{6}$$

After the computation model is constructed, the calculation process of the AMDML method is shown in Fig. 5.

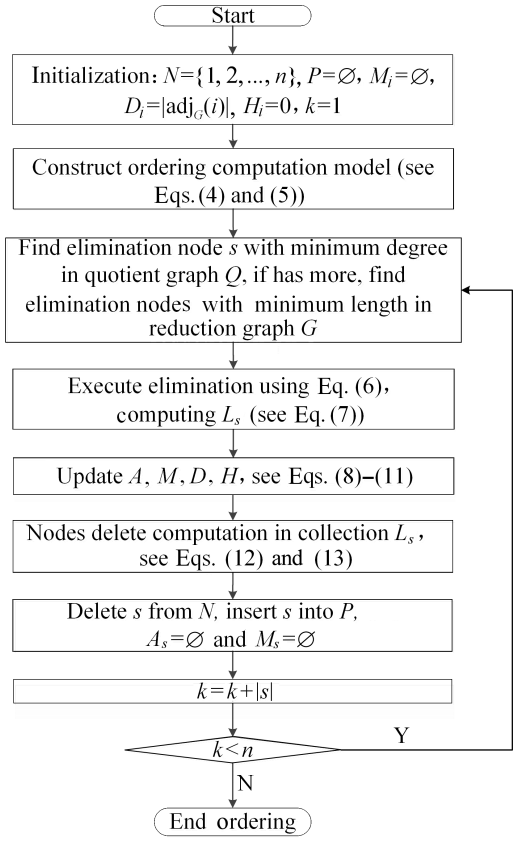

Fig. 5 AMDML ordering process.

$$L_s = (A_s \cup \bigcup_{e \in M_s} L_e) \setminus s \tag{7}$$

$$A_i = (A_i \setminus L_s) \setminus s \tag{8}$$

$$M_i = (M_i \setminus M_s) \cup \{s\} \tag{9}$$

$$D_i = |A_i \setminus i| + |(\bigcup_{e \in M_i} L_e) \setminus i| \tag{10}$$

$$H_i = \mathrm{Max}(H(s) + 1, H(\mathrm{adj}_G(s))) \tag{11}$$

where D and H represent the correction functions of the external value of the quotient graph node after elimination and the length of elimination tree for nodes from the reduction graph, respectively. When more than one nodes satisfy the MD, the nodes with the minimum length are selected in turn as the elimination nodes, which is the selection criteria for generating the ordering vector. When the L_s set is deleted, the feature free relation is introduced, that is, nodes i and j satisfy the following feature free relation in the sketch G:

$$\mathrm{adj}_G(i) \cup \{i\} = \mathrm{adj}_G(j) \cup \{j\} \tag{12}$$

This finding indicates that the degree of nodes i and j are equal and have a direct connection relation. Nodes without a feature relation do not affect the ordering effect on the set. Then, nodes with a feature relation in L_s are treated as follows:

$$\begin{aligned} &i = i \cup j, \\ &d_i = d_i - |j|, \\ &N = N \setminus \{j\}, \\ &A_j = \varnothing, M_j = \varnothing \end{aligned} \tag{13}$$

4.2 Case analysis

Unlike in the AMD method, a new memory space sequence with size n is added to record the change of the elimination tree length of each node during each reduction step in our improved AMDML method. In terms of time complexity, it increases the query time of the minimum length in the list. In the worst case, the time complexity of AMDML only increases $O(n)$ compared with the AMD method. Taking 10×10 matrix as an example (Fig. 6), the reduction process of the quotient graph and reduction graph is shown in Figs. 7 and 8, respectively. The value changes of the external approximation degree and length of each node of the elimination tree in the AMDML ordering process are shown in Table 2.

In the first two steps of reduction, because the selected node is an isolated node in quotient Q, the reduction process affects only the node length, while the node degree does not change. From Q_2 to Q_3, five nodes satisfy the MD of 3; two of them satisfy

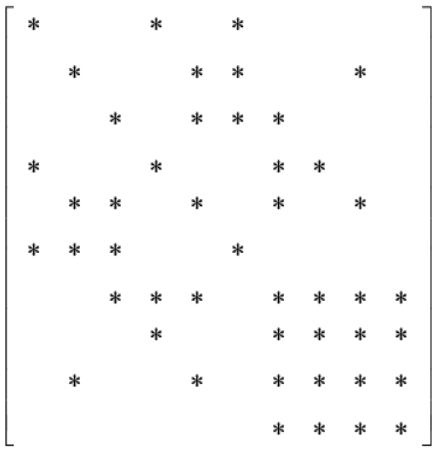

Fig. 6 10×10 symbolic matrix.

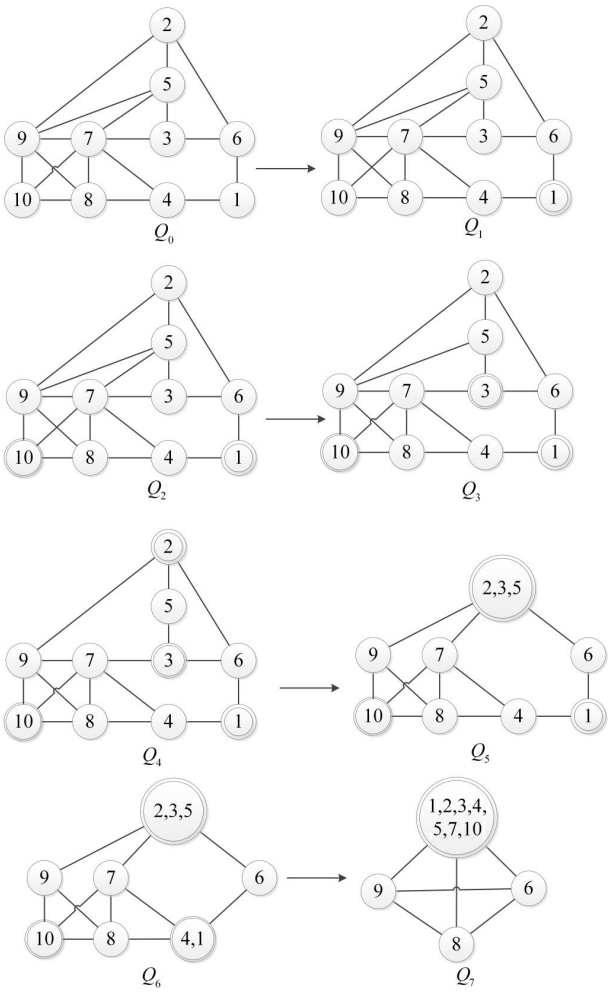

Fig. 7 Quotient graph.

the minimum length, and then Node 3 is selected randomly for reduction. Thus, $A_3 = L_3 = 5, 6, 7$, and edge $(5, 7)$ is taken away from quotient A_5 and A_7 as the redundant edge. Node 5 is selected from Q_4 to Q_5, and $A_5 = M_5 = 2, 3$, $L_2 = 5, 6, 9$, $L_3 = 5, 6, 7$,

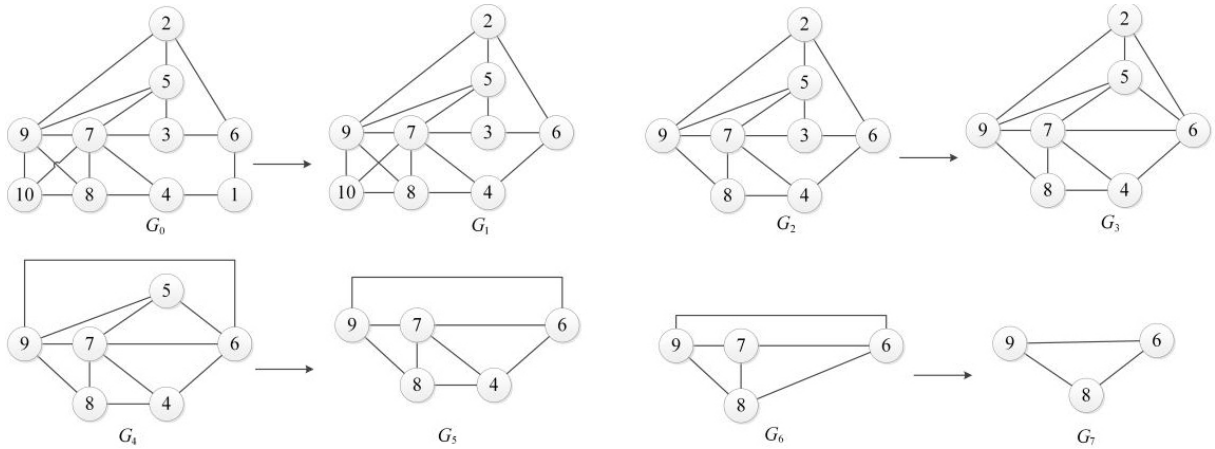

Fig. 8 Reduction graph.

Table 2 Value changes of properties in the AMDML-based elimination process.

Step	Approximate minimum degrees of quotient graph nodes (1–10)	Length of graph nodes (1–10)	Selected node
1	{2,3,3,3,4,3,6,4,5,3}	{0,0,0,0,0,0,0,0,0,0}	1
2	{2,3,3,3,4,3,6,4,5,3}	{0,0,0,1,0,1,0,0,0,0}	10
3	{2,3,3,3,4,3,5,3,4,3}	{0,0,0,1,0,1,1,1,1,0}	3
4	{2,3,3,3,4,4,5,3,4,3}	{0,0,0,1,1,1,1,1,1,0}	2
5	{2,3,3,3,3,4,5,3,4,3}	{0,0,0,1,1,1,1,1,1,0}	5
6	{2,3,3,3,3,3,4,3,3,3}	{0,0,0,1,1,2,2,1,2,0}	4
7	{2,3,3,3,3,3,3,3,3,3}	{0,0,0,1,1,2,2,2,2,0}	7

$L_5 = (A_5 \cup L_2 \cup L_3)$, $L_5 = 6, 7, 9$ is calculated, and the redundant edge $(7, 9)$ is deleted. When the reduction process moves to the seventh iteration, Nodes 6, 8, and 9 satisfy the feature free relation in the quotient graph, that the filling matrices formed by arbitrary ordering combination are the same, and they are added to the end of the ordering vector. Then, the ordering vector [1, 10, 3, 2, 5, 4, 7, 6, 8, 9] is derived.

5 Testbed for Experimental Result Comparison

5.1 Symbolic analysis of coefficient matrix

The construction of the preordering calculation verification procedure in the stage of symbol analysis is illustrated in Fig. 9.

The following definitions are made:

(1) Fill ratio Fr = NonZero (factor matrix) / NonZero (coefficient matrix);

(2) Length L_{en} = MaxLength (Etree (factor matrix)), where Etree $(\cdot)$ is the factor matrix for the elimination tree, which can be obtained through Ref. [19];

(3) Average length AL = Sum(Length (N_i in Etree (factor matrix))) / number of nodes.

Matpower is used on a computer with an Intel Core i5 2.6 GHz processor and 4 GB memory, and the node name, branch number, and dimension of the Jacobian sparse matrix are selected as example nodes, as shown in Table 3.

Dynamic MD method[13], MD-ML method[22], AMD method[23], and the proposed AMDML method are used to perform ordering computation in the symbolic analysis phase, and their coefficient matrices come from power flow cases. Then, the fill ratio Fr, length value L_{en}, and average length value AL are computed based on the elimination tree and symbolic factor matrices. The

Table 3 Information of test cases.

Case name	Number of branches	Jacobian matrix
Case 300	411	530×530
Case 1354	1991	2447×2447
Case 3120	3693	5991×5991
Case 9241	16 049	17 036×17 036
Case 13 659	20 467	23 225×23 225

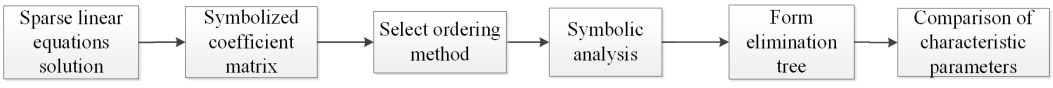

Fig. 9 Comparison of ordering methods in symbolic analysis phase.

times using different ordering methods for symbolic analysis are recorded T as for short. Results are shown in Table 4.

The above calculation results show that both the MD-ML and AMDML methods, which take the length of the eliminating tree as a vector selection constraint, can further reduce the length of the elimination tree and the average length without significantly increasing the filling degree of the factor matrix. Compared with AMD, the AMDML method reduces the length of eliminating tree and the average length by 10% and 5%, respectively. The time for symbolic analysis using the AMDML method is longer than that of the AMD method, because of the node length update computation of the elimination tree (see Eq. (11)). Finally, the calculation speed of back substitution is improved.

The MD-ML method based on the reduction graph obtains the shortest length of elimination tree, because when the length of the elimination tree is used as the node selection attribute in the reduction graph, the MD-ML method determines a single row/column node, while the operation on the quotient graph can determine only the row/column node set. In accordance with the filling degree value of the factor matrix, the filling degree of AMD and AMDML methods based on quotient graph are generally smaller than that of the MD and MD-ML methods on the basis of the reduction graph. The decomposition time of the triangular factor is much longer than that of the back substitution in the serial calculation mode relative to the length of eliminating tree and the average length. Therefore a small filling degree corresponds to a more obvious improvement of calculation efficiency. The time required to complete the above ordering methods satisfies MD-ML > MD > AMDML > AMD. Thus, we can conclude that compared with the common ordering methods, the proposed AMDML method is better in ordering performance evaluation index.

5.2 Time analysis for power flow computation

The computing environment is Win 7 + MATPOWER 5.0[24]. Symbol analysis is performed on Jacobian matrix formed by each node of the above example by nonordering (NaN), MD-ML, AMD, and AMDML methods. Complete power flow calculations were executed subsequently. The calculation times of the Newton method and the P-Q decomposition method are shown in Tables 5 and 6, respectively.

When the node scale is small, preconditioning analysis has little effect on the convergence time of power flow calculation. While the dimension of the sparse matrix is large (the dimension is more than 5000 in this paper), the performance can be improved by ordering preconditioning analysis. A large matrix dimension of matrix corresponds to a more obvious performance improvement. The nonzero element filling degree of AMDML is lower than that of MD-ML, thereby resulting in a more efficient factor decomposition and a shorter computation time.

Compared with the AMD method in symbolic analysis for power flow calculation, the AMDML method can reduce the power flow convergence time further, and improve the calculation efficiency by more than 10%, thereby verifying the feasibility and effectiveness of the proposed method.

Table 5 Comparison of Newton-Raphson power flow computation time.

(ms)

Case name	Iterations	NaN	MD-ML	AMD	AMDML
Case 300	5	13	30	20	12
Case 1354	4	56	45	31	30
Case 3120	6	600	127	107	100
Case 9241	6	7900	454	390	340
Case 13 659	5	19 820	694	430	370

Table 6 Comparison of P-Q power flow computation time.

(ms)

Case name	Iterations	NaN	MD-ML	AMD	AMDML
Case 300	9	4	5	4	3
Case 1354	8	13	10	8	7
Case 3120	13	64	34	26	21
Case 9241	14	530	155	80	70
Case 13 659	14	1050	237	110	92

Table 4 Characteristic parameter of the elimination tree under different ordering methods.

Method	MD				MD-ML				AMD				AMDML			
	L_{en}	AL	Fr	T (ms)	L_{en}	AL	Fr	T (ms)	L_{en}	AL	Fr	T (ms)	L_{en}	AL	Fr	T (ms)
Case 300	54	29.3	1.48	0.14	48	30.4	1.49	0.21	58	31.3	1.49	0.10	54	29.8	1.46	0.1
Case 1354	93	48.7	1.40	1.40	73	44.9	1.41	1.80	83	48.3	1.39	0.90	67	41.6	1.38	1.0
Case 3120	160	100.0	1.72	3.00	110	80.2	1.72	4.00	143	85.8	1.68	1.30	126	85.8	1.69	1.2
Case 9241	300	183.6	1.73	8.00	234	166.1	1.74	10.00	286	197.0	1.68	4.00	260	184.3	1.66	4.3
Case 13 659	279	181.5	1.67	10.00	220	149.3	1.64	13.50	301	202.6	1.64	5.00	286	198.4	1.64	5.2

6 Conclusion

A numerical solution for electrical network analysis involves a continuous solution to sparse linear equations. The structure of coefficient matrix affects the calculation performance. Direct method is used to solve the problem, which will obtain factor matrix after *LU* decomposition. The fill ratio, length, and average length of the elimination tree are characteristic features of computing performance. Ordering method is used to perform elementary row/column transformation in the symbolic analysis stage. This step reduces the property values of attributes, thereby improving the calculation speed.

On the basis of graph analysis and quotient graph theory, this paper built an ordering model, and proposed the AMDML ordering method. Compared with commonly used ordering methods, the method can further reduce the length and average length of the elimination tree without increasing the fill ratio of the factor matrices. Different sizes of matrices of power flow cases verify the efficacy of the proposed method. Further studies can be conducted as follows:

(1) Decomposition of the coefficient matrix is a basic process used in the direct method to solve sparse linear equations. Efficient ordering methods are also applicable to various power electrical network analyses and computing applications, such as state estimation, transient stability, and OPF.

(2) The length of the elimination tree influences the dependencies of computations between nodes, which also affects the solution vector. Multithreaded/multicore architecture-based sparse vector technology can further reduce computing time. Additional attention needs to be paid to elimination tree-based parallelization.

Acknowledgment

This work was supported in part by the National Key Basic Research and Development Program of China (No. 2017YFE0132100), the Tsinghua-Toyota Research Fund (No. 20203910016), and the BNRist Program (No. BNR2020TD01009).

References

[1] J. Cao, K. Meng, J. Wang, M. Yang, Z. Chen, W. Li, and C. Lin, An energy internet and energy routers, *Scientia Sinica Informationis*, vol. 44, no. 6, pp. 714–727, 2014.

[2] K. Wang, J. Yu, Y. Yu, Y. Qian, D. Zeng, S. Guo, Y. Xiang, and J. Wu, A survey on energy internet: Architecture, approach, and emerging technologies, *IEEE Systems Journal*, vol. 12, no. 3, pp. 2403–2416, 2018.

[3] Y. Ming, J. Yang, J. Cao, Z. Zhou, and C. Xing, Distributed energy sharing in energy internet through distributed averaging, *Tsinghua Science and Technology*, vol. 23, no. 3, pp. 233–242, 2018.

[4] G. Kamath, L. Shi, E. Chow, W. Song, and J. Yang, Decentralized multigrid for in-situ big data computing, *Tsinghua Science and Technology*, vol. 20, no. 6, pp. 545–559, 2015.

[5] L. Liu, X. Chen, Z. Lu, L. Wang, and X. Wen, Mobile-edge computing framework with data compression for wireless network in energy internet, *Tsinghua Science and Technology*, vol. 24, no. 3, pp. 271–280, 2019.

[6] L. Tan, S. Kothapalli, and L. Chen, A survey of power and energy efficient techniques for high performance numerical linear algebra operations, *Parallel Computing*, vol. 40, no. 10, pp. 559–573, 2014.

[7] J. Guo, J. Zhou, Q. Li, Y. Chen, Y. Luo, and Y. Lang, Current status of high-performance on-line analysis computation and key technologies for cooperating computation, *Automation of Electric Power Systems*, vol. 42, no. 3, pp. 149–159, 2018.

[8] M. A. Pai and H. Dag, Iterative solver techniques in large scale power system computation, in *Proceedings of the 36th IEEE Conference on Decision and Control*, San Diego, CA, USA, 1997, pp. 3861–3866.

[9] Y. Wan, J. Cao, S. Zhang, G. Tu, C. Lu, X. Xu, and K. Li, An integrated cyber-physical simulation environment for smart grid applications, *Tsinghua Science and Technology*, vol. 19, no. 2, pp. 133–143, 2014.

[10] J. Shu, W. Xue, and W. Zheng, A parallel transient stability simulation for power systems, *IEEE Transactions on Power Systems*, vol. 20, no. 4, pp. 1709–1717, 2005.

[11] S. C. Eisenstat and J. W. H. Liu, Exploiting structural symmetry in unsymmetric sparse symbolic factorization, *SIAM Journal of Matrix Analysis and Applications*, vol. 13, no. 1, pp. 202–211, 1992.

[12] Y. Chen, Z. Huang, Y. Liu, M. J. Rice, and S. Jin, Computational challenges for power system operation, in *Proceedings of 45th Hawaii International Conference on System Science*, Maui, HI, USA, 2012, pp. 2141–2150.

[13] B. Zhang, S. Chen, and Z. Yan, *Advanced Electrical Power Network Analysis*. Beijing, China: Tsinghua Press, 2007.

[14] S. K. Khaitan, A survey of high-performance computing approaches in power systems, in *Proceedings of IEEE Power and Energy Society General Meeting*, Boston, MA, USA, 2016, pp. 1–6.

[15] R. C. Green, L. Wang, and M. Alam, Applications and trends of high performance computing for electric power systems focusing on smart grid, *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 922–931, 2013.

[16] Z. Huang, Y. Chen, and J. Nieplocha, Massive contingency analysis with high performance computing, in *Proc. of IEEE Power and Energy Society General Meeting*, Calgary, Canada, 2009, pp. 1–8.

[17] L. Tan, S. Kothapalli, L. Chen, O. Hussaini, R. Bissiri, and Z. Chen, A survey of power and energy efficient techniques

for high performance numerical linear algebra operations, *IEEE Transactions on Parallel Computing*, vol. 40, no. 10, pp. 559–573, 2014.

[18] Z. Li, V. D. Donde, J. Tournier, and F. Yang, On limitation of traditional multi-core and potential of many-Core processing architectures for sparse linear solvers used in large-scale power system applications, in *Proceedings of IEEE Power and Energy Society General Meeting*, Detroit, MI, USA, 2011, pp. 1–8.

[19] A. Gomez and L. G. Franquelo, Node ordering algorithms for sparse vector method improvement, *IEEE Transactions on Power Systems*, vol. 3, no. 1, pp. 73–79, 1988.

[20] R. Betancourt, An efficient heuristic ordering algorithm for partial matrix refactorization, *IEEE Transactions on Power Systems*, vol. 3, no. 3, pp. 1181–1186, 1988.

[21] J. W. H. Liu, The role of elimination trees in sparse factorization, *SIAM Journal of Matrix Analysis and Applications*, vol. 11, no. 1, pp. 134–172, 1990.

[22] R. Betancourt, An efficient heuristic ordering algorithm for partial matrix refactorization, *IEEE Transactions on Power Systems*, vol. 3, no. 3, pp. 1181–1186, 1988.

[23] P. R. Amestroy, T. A. Davis, and I. S. Duff, Algorithm 837: AMD, an approximate minimum degree ordering algorithm, *ACM Transactions Math Software*, vol. 30, no. 3, pp. 381–388, 2004.

[24] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education, *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 12–19, 2011.

Jian Guo received the BS and MS degrees in computer science from Taiyuan University Of Technology, Taiyuan, China in 2008 and North China Electric Power University, Beijing, China in 2011, respectively, and the PhD degree in electrical engineering from China Electrical Power Research Institute, Beijing, China in 2018. He is currently a postdoctoral researcher with the Department of Electrical Engineering, Tsinghua University, and working as a research assistant in Beijing National Research Center for Information Science and Technology, Beijing, China. His research interests include online analysis, high-performance computation, and their applications in power system and the energy internet.

Hong Liang received the BS degree in computational mathematics from Wuhan University, Wuhan, China in 2013 and the PhD degree in applied mathematics from Tsinghua University, Beijing, China in 2019. She is currently a postdoctoral researcher with the Research Institute of Information Technology, Tsinghua University. Her research interests include control and optimization, machine learning, and their applications in power system and the energy internet.

Songpu Ai received BS degree in mathematics and applied mathematics from Shandong University, Jinan, China in 2012, the MS degree in information and communication technology from University of Agder, Norway in 2015, and the PhD degree in information technology from University of Stavanger, Norway in 2019. He is currently a post-doctoral researcher in the Research Institute of Information Technology, Tsinghua University. His current research interests include blockchain, big data, and artificial intelligence and their applications in power systems, smart grids, and the energy internet.

Haochen Hua received the BS degree in mathematics with finance in 2011 and the PhD degree in mathematical sciences in 2016, both from the University of Liverpool, Liverpool, UK. From 2016 to 2019. He was a postdoctoral fellow in the Research Institute of Information Technology, Tsinghua University. Since 2020, he has been a professor in the College of Energy and Electrical Engineering, Hohai University, Nanjing, China. His current research interests include optimal and robust control theory and its applications in power systems, smart grids, and energy internet.

Chao Lu received the BEng and PhD degrees from Tsinghua University in 1999 and 2005, respectively. He is currently an assistant professor with the Department of Electrical Engineering, Tsinghua University, China. His research interests include power system analysis and control, and intelligent control applications.

Junwei Cao received the PhD degree in computer science from University of Warwick, UK in 2001. He received the MEng and BEng degrees from Tsinghua University in 1998 and 1996, respectively. He is currently a professor and deputy director of Research Institute of Information Technology, Tsinghua University, China. He is also the director of Open Platform and Technology Division, Tsinghua National Laboratory for Information Science and Technology. Before joining Tsinghua in 2006, he was a research scientist of Massachusetts Institute of Technology, USA. Before that he worked as a research staff member of NEC Europe Ltd., Germany. He has published over 130 academic papers and books, which were cited by international researchers for over 3000 times. He is a senior member of the IEEE Computer Society and the member of the ACM and CCF. His research interests include advanced computing technology and applications.