大规模动态经济调度纵横交叉算法

考虑阀点效应的动态经济调度（Dynamic Economic Dispatch，DED）具有非线性、非平滑及非凸等特性。针对大规模、高维度DED优化问题，提出采用一种高效的求解方法——纵横交叉（Crisscross Optimization，CSO）算法。CSO算法采用一种双交叉搜索机制，其中横向交叉引入扩展因子增强全局搜索能力，纵向交叉引入维交叉概念避免维局部最优问题。两种交叉算子交替产生中庸解，通过与父代竞争产生的占优解在种群中相互催化，在避免早熟问题的同时能够迅速收敛到全局最优。将CSO算法应用到四个不同规模动态经济调度模型中，仿真结果表明：与其他复杂的优化算法相比，CSO算法在求解质量和收敛速度方面具有明显的优势。

孟安波　胡函武　卢海明　郭壮志/广东工业大学自动化学院

孟安波/副教授

关键词/Keywords

动态经济调度·
纵横交叉算法·
爬坡率·
阀点效应·

作为电力系统最重要、最复杂的优化问题之一，动态经济调度（Dynamic Economic Dispatch，DED）旨在根据每一个确定时段的预测负荷需求，合理安排各机组的有功出力，在满足发电计划的同时使系统总发电成本最小。与静态经济调度（Static Economic Dispatch，SED）相比，DED除了要满足功率平衡等限制条件外，还需要考虑爬坡率的影响，因而机组组合模型也就更为复杂，进而给求解带来了更大的挑战和困难。

在过去的几十年里，为解决DED优化问题，专家学者们做了大量的工作并提出了一系列的解决方案。传统基于数学分析的方法包括线性规划法[1]、二次规划法[2]和拉格朗日法[3]等，虽然在一定程度上取得了不错的效果，但由于自身的局限性，如线性规划法在求解的过程中将机组模型线性化必然会产生较大的偏差，而二次规划法则要求目标函数必须是连续可导的，因此在优化的过程中很难寻找到全局最优解。为了克服传统优化方法的缺陷，近年来，人们将视线逐渐转移到了各种基于人工智能的启发式算法，并成功地将其运用到DED优化问题中，例如，遗传算法（GA）[4]、粒子群算法（PSO）[4]、模拟退火算法（SA）[5]、人工免疫系统（AIS）[6]、人工蜂群算法（ABC）[4]、进化规划算法（EP）[2]、差分进化算法（DE）[7]、和声搜索算法（HS）[8]、搜寻者优化算法（SOA）[9]和经验竞争算法（ICA）[10]等，这些算法对于动态经济调度这样的多约束、非线性问题表现出了较强的搜索能力，但在求解准确度和收敛速度方面稍显不足。例如，在处理多峰值问题时粒子群算法收敛速度较慢且极易陷入局部最优，而进化规划算法和人工免疫系统则耗时长久。针对单一算法存在的局限，改进算法和混合算法应运而生。改进算法有自适应粒子群算法（APSO）[11]、改进差分进化算法[12]和改进蜂群优化算法（EBSO）[13]等，混合算法则包括免疫遗传算法（HIGA）[14]、混沌差分蜂群优化算法（CDBCO）[15]、混沌差分进化算法（CDE）[16]和人工免疫系统序列二次规划算法（AIS－SQP）[17]等，它们在求解准确度上有了明显的提高。虽然混合算法能够汲取各种算法的优势从而避免早熟现象，却往往牺牲了收敛速度，并且控制参数的设置和算法结构也更为复杂。因此寻求一种全局收

国家自然科学基金项目（51307025），广东省自然科学基金（S2013040013776，S2012040007911），广东省教育厅育苗工程项目（2013LYM_0019）。

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

敛能力强同时收敛速度快的启发式优化算法仍然是一个热门话题。

文中采用CSO算法[18]解决DED问题。文献[18]采用12个经典测试函数来验证该算法的综合性能并将其应用到大规模SED中。和其他启发式算法相比，CSO算法结构简单，控制参数设置少（只需设置一个参数），操作极为方便，最重要的是可以很好地平衡全局收敛能力和收敛速度。为了验证算法的高效性，将其应用到10#机组、30#机组、60#机组和120#机组DED模型中，并与当前最新的研究成果做对比，实验结果证明了算法的优越性。

动态经济调度数学模型

1. 目标函数

DED目标函数是发电机组各段时间发电费用总和，表述为

$$F=\min\sum_{t=1}^{T}\sum_{i=1}^{n}F_{it}(P_{it}) \tag{1}$$

式中，T为计划发电小时数；n为机组台数；F_{it}为机组i在时刻t的费用函数；P_{it}为机组i在时刻t的有功输出；F为总发电成本。考虑阀点效应，机组i在t时刻的燃料费用函数为

$$F_{it}(P_{it})=c_i+b_iP_{it}+a_iP_{it}^2+\left|e_i\sin\left[f_i(P_{i\min}-P_{it})\right]\right| \tag{2}$$

式中，c_i、b_i和a_i为机组i的燃料费用系数；e_i、f_i为机组i考虑阀点效应的常量；$P_{i\min}$为机组i的有功输出下限。

2. 约束条件

1）功率平衡约束

$$\sum_{i=1}^{n}P_{it}-P_{Dt}-P_{\text{loss}}(t)=0 \tag{3}$$

$$P_{\text{loss}}(t)=\sum_{i=1}^{n}\sum_{j=1}^{n}P_{it}\boldsymbol{B}_{ij}P_{jt} \tag{4}$$

式中，P_{Dt}、$P_{\text{loss}}(t)$分别为t时刻的负荷需求和系统网损；$\boldsymbol{B}$为网损系数矩阵。

2）机组有功输出约束

$$P_{i\min}\leqslant P_{it}\leqslant P_{i\max} \tag{5}$$

式中，$P_{i\max}$为机组i有功输出上限。

3）机组爬坡率约束

$$P_{it}-P_{it-1}\leqslant DR_i \tag{6}$$

$$P_{it-1}-P_{it}\leqslant UR_i \tag{7}$$

式中，UR_i、DR_i分别为机组i向上爬坡限度和向下爬坡限度。

同时考虑有功输出约束和爬坡率限制时，式（5）～式（7）可由式（8）代替

$$P_{it}=\begin{cases}P_{it},(P_{i\min}\leqslant P_{it}\leqslant P_{i\max},t=1)\\ \max\{P_{\min},P_{it-1}-DR_i\},(P_{it}\leqslant\max(P_{\min},P_{it-1}-DR_i),t\geqslant 2)\\ \min\{P_{\max},P_{it-1}-UR_i\},(P_{it}\geqslant\min(P_{\max},P_{it-1}-UR_i),t\geqslant 2)\end{cases} \tag{8}$$

纵横交叉算法

该算法采用双重搜索机制，包括横向交叉和纵向交叉两种交叉方式。种群每一次迭代都要经过横向交叉和纵向交叉操作，产生的子代称为中庸解。交叉过后，执行竞争算子，只有比父代适应度更好的子代才能保存下来，称为占优解。

1. 横向交叉

横向交叉的过程中首先将种群中所有粒子进行两两不重复随机配对，然后对配对粒子所有维进行算术交叉，同时在算术交叉的基础上引入扩展因子，加大搜索范围，增强全局搜索能力。父代粒子$X(i)$和$X(j)$进行横向交叉产生子代的公式表述如下

$$MS_{\text{hc}}(i,d)=r_1X(i,d)+(1-r_1)X(j,d)+c_1[X(i,d)-X(j,d)] \tag{9}$$

$$MS_{\text{hc}}(j,d)=r_2X(j,d)+(1-r_2)X(i,d)+c_2[X(j,d)-X(i,d)] \tag{10}$$

式中，c_1、c_2为$[-1,1]$上均匀分布的随机数；r_1、r_2为$[0,1]$上均匀分布的随机数；$X(i,d)$、$X(j,d)$分别为父代粒子i、j的第d维；$MS_{\text{hc}}(i,d)$、$MS_{\text{hc}}(j,d)$为父代粒子i、j产生的子代，称为中庸解。

2. 纵向交叉

与横向交叉不同，纵向交叉是所有粒子在不同维之间进行的交叉操作。纵向交叉引入维交叉概念，从而避免维局部最优问题。假设d_1，d_2为粒子i不同维，那么它们进行纵向交叉后产生的子代为

$$MS_{\text{vc}}(i,d_1)=rX(i,d_1)+(1-r)X(i,d_2)\quad i\in N(1,M)\quad d_1,d_2\in N(1,D) \tag{11}$$

式中，r为$[0,1]$上均匀分布的随机数；$MS_{\text{vc}}(i,d_1)$为

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

粒子i第d_1维和d_2维交叉后产生的子代。

由于不同维元素上下限取值可能不同，因此在进行纵向交叉前需要将粒子各维进行归一化处理。同时为了使早熟收敛的那一维跳出局部最优且不破坏全局最优维的信息，每次纵向交叉都只产生一个粒子。这样既能使粒子摆脱局部最优，又保持了种群的多样性。和横向交叉操作一样，纵向交叉后粒子进入竞争算子，与父代粒子进行比较，适应度更好的得以保存。竞争机制的引入使两种交叉方式完美地结合起来，既提高了收敛准确度，又加快了收敛速度。

纵横交叉算法在DED优化问题中的应用

应用CSO算法解决大规模DED优化问题步骤如下。

（1）初始化种群

在搜索的过程中每个粒子代表一组解，则粒子i的位置可以表示为

$$A(i,j,k)=\mathrm{rand}(P_{\max}-P_{j\min})+P_{j\min} \tag{12}$$

机组每小时的有功输出可以用矩阵$\boldsymbol{P}$为

$$\boldsymbol{P}=\begin{pmatrix} P_{1,1} & \cdots & P_{1,k} & \cdots & P_{1,24} \\ \vdots & & \vdots & & \vdots \\ P_{j,1} & \cdots & P_{j,k} & \cdots & P_{j,24} \\ \vdots & & \vdots & & \vdots \\ P_{N,1} & \cdots & P_{N,k} & \cdots & P_{N,24} \end{pmatrix}$$

式中，$j(j=1,2,\cdots,N)$为第j台机组；$k(k=1,2,\cdots,24)$为第k小时。

（2）适应度评价

根据目标函数计算每个粒子的适应度，找出最小值，记为$fitg_{\mathrm{best}}$。适应度计算公式为

$$fit=\sum_{k}^{T}\sum_{j}^{N}\{c_j+b_jP_{jk}+a_jP_{jk}^2+|e_j\sin[f_j(P_{j\min}-P_{jk})]|\}+\sum_{k}^{T}P_{\mathrm{penish}}|\sum_{j}^{N}P_{jk}-P_{\mathrm{loss}}(k)-P_{Dk}| \tag{13}$$

式中，P_{penish}为惩罚系数，第一项为发电总成本，第二项为惩罚总成本。

（3）横向交叉

将种群粒子两两随机配对，依式（9）、式（10）进行横向交叉，对交叉后的中庸解进行越限处理

$$\begin{aligned} P_{j,k+1}^{\max}&=\min\{P_{j\max},P_{j,k}+UR_j\} \\ P_{j,k+1}^{\min}&=\max\{P_{j\min},P_{j,k}-DR_j\} \end{aligned} \tag{14}$$

（4）更新种群

计算子代每个粒子的适应度，即发电总费用，分别与父代做比较，费用更少的粒子得以保存，作为占优解DS_{hc}。

（5）纵向交叉

由于同一粒子不同维上下限可能不一致，所以需要将更新后的种群进行归一化处理，然后将同一粒子不同维两两随机配对，依式（11）进行纵向交叉，为了减少对已经处于全局最优维的影响同时使陷入早熟收敛的维跳出局部最优，每次交叉只产生一个后代。再将得到的中庸解进行反归一化处理。最后将所得粒子根据式（14）进行越限处理。

（6）更新种群

计算中庸解的适应度，并与上一代作比较，适应度更好的粒子作为占优解进入下次迭代。

（7）输出全局最优解

计算更新后种群粒子的适应度，判断是否达到最大迭代次数，若达到，输出$fitg_{\mathrm{best}}$，否则返回步骤（2）。

实例仿真和结果分析

为了验证算法的高效性和实用性，文中采用4个DED系统对其进行测试。在所有的测试中，阀点效应和爬坡率都在考虑的范围内。总发电时间T选取为1天，分为24个时间段，每个时段为1 h。实验使用Matlab2010a进行仿真，计算机配置为Inter(R) CPU G5400、2.49 GHz、3.40 GB RAM，操作系统为Windows XP Professional。算法参数设定为：种群规模为30个粒子；最大迭代次数为1 000次；横向交叉概率为1；纵向交叉概率为0.6；各测试系统每小时负荷需求、各机组有功输出上下限、网损系数和爬坡率等数据可以参见文献[14]。

1）实例1。10#机组考虑网损DED。

2）实例2。30#机组不考虑网损DED。

3）实例3。60#机组不考虑网损DED。

4）实例4。120#机组不考虑网损DED。

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

1. 10#机组测试系统

10#机组不考虑网损系统是DED经典测试模型。近年来，为了验证算法在实际工程问题中的实用性，众多单一算法和改进算法以及混合算法都采用此模型进行测试。而网损的加入使10#机组模型变得更为复杂，DED的非线性、非平滑特性也由此更为突出，取得全局最优解的难度大大增加。将CSO算法应用于该模型的求解，表1为采用CSO算法取得的总费用以及CPU运行时间和其他算法的比较。

在该测试系统中，采用CSO算法取得的最小费用为1 041 013美元，CPU平均运行时间为1.481 min。从表1可以看出，由其他算法，如单一算法PSO、EP、GA及AIS，ABC和混合算法EP－SQP、CDBCO及HIGA等取得的最小费用均大于采用CSO算法取得的最小费用，大多数CPU运行时间也多于CSO算法。证明了CSO算法具有较好的全局收敛能力和较快的收敛速度。如表2所示，列出了取得最小费用时各机组出力分布。

表1 实例1不同算法结果比较

算法	最小费用/美元	平均费用/美元	最大费用/美元	运行时间/min
EP[2]	1 054 685	1 057 323	NA	47.23
EP－SQP[2]	1 052 668	1 053 771	NA	27.53
GA[4]	1 052 251	1 058 041	1 062 511	3.444
PSO[4]	1 048 410	1 052 092	1 057 170	4.093
AIS[6]	1 045 715	1 047 050	1 048 431	30.973
ECE[19]	1 043 989	1 044 470	NA	0.644
ABC[4]	1 043 381	1 044 963	1 046 805	3.408
CDBCO[15]	1 042 900	1 044 700	NA	1.53
HIGA[14]	1 041 088	1 042 980	1 044 927	3.8
Proposed CSO	1 041 013	1 042 652	1 044 165	1.481

表2 采用CSO算法求解10#机组带网损测试系统最小费用机组出力分布

时间段	机组出力/MW										费用/美元	损失/MW
	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}		
1	226.624	135.000	179.055	60.000	73.000	122.450	129.590	47.000	20.000	55.000	28 480.982	11.719
2	303.248	135.000	178.822	60.000	73.000	122.450	129.590	47.000	20.000	55.000	30 150.698	14.111
3	379.873	135.000	205.027	60.000	122.867	122.450	129.590	47.000	20.000	55.000	33 575.196	18.806
4	379.873	135.000	285.027	60.000	172.733	143.887	129.590	47.000	20.000	55.000	36 936.901	22.110
5	379.873	215.000	304.380	60.000	172.733	122.455	129.590	47.000	20.000	55.000	38 502.689	26.031
6	456.497	222.266	325.306	60.000	222.600	122.450	129.590	47.000	20.000	55.000	41 882.460	32.709
7	456.497	302.266	324.643	60.000	222.600	122.451	129.590	47.000	20.000	55.000	43 729.865	38.047
8	456.497	309.533	305.706	110.000	222.601	160.000	129.590	47.000	20.000	55.000	45 498.054	39.927
9	456.497	389.533	339.851	120.415	222.600	160.000	129.590	77.000	20.000	55.000	49 360.274	46.486
10	456.497	460.000	338.132	170.415	222.600	160.000	129.590	85.312	50.000	55.000	53 336.118	55.546
11	456.497	460.000	330.888	220.415	222.600	160.000	129.590	115.312	52.057	55.000	55 404.106	56.360
12	456.497	460.000	340.000	270.415	235.047	160.000	129.590	120.000	52.057	55.000	57 348.539	58.606
13	456.497	396.799	296.509	300.000	222.600	122.450	129.590	120.000	22.057	55.000	52 577.376	49.503
14	379.873	396.799	300.908	300.000	172.733	122.450	129.590	90.000	20.000	55.000	49 114.041	43.353
15	303.248	316.799	304.837	300.000	172.733	122.450	129.590	85.312	20.000	55.000	45 748.767	33.970
16	226.624	309.533	290.960	250.000	122.867	122.450	129.590	55.312	20.000	55.000	40 749.389	28.336
17	226.624	229.533	309.106	241.246	122.867	122.450	129.590	47.000	20.000	55.000	38 882.951	23.415
18	303.248	309.533	307.406	241.246	122.867	122.461	129.590	47.000	20.000	55.000	42 061.573	30.352
19	379.873	309.533	297.421	241.246	172.733	160.000	129.590	47.000	20.000	55.000	45 508.790	36.396
20	456.497	389.533	323.084	291.246	222.600	160.000	129.590	47.000	50.000	55.000	53 128.575	52.550
21	456.497	396.799	297.400	241.246	172.733	154.928	129.590	47.000	20.000	55.000	48 983.297	47.193
22	379.873	316.799	275.358	191.246	122.867	122.450	129.590	47.000	20.000	55.000	42 396.167	32.183
23	303.248	236.799	195.358	170.751	73.000	122.450	129.590	47.000	20.000	55.000	35 791.648	21.197
24	226.624	222.266	184.279	120.751	73.000	122.450	129.590	47.000	20.000	55.000	31 864.672	16.961
总计											1 041 013.124	835.866

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

2. 30#机组测试系统

该测试系统是将10#机组不考虑网损测试系统中机组和负荷需求同时扩增3倍，那么变量数也从原来的240个增至720个，局部最优点个数急剧上升，对算法的全局搜索能力要求更高。将CSO算法与其他算法应用于该模型的求解，其结果如表3所示。

表3 实例2不同算法结果比较

算法	最小费用/美元	平均费用/美元	最大费用/美元	运行时间/min
EP[2]	3 164 531	3 200 171	NA	NA
DE[7]	3 163 000	3 173 100	NA	0.52
EP－SQP[2]	3 159 204	3 169 093	NA	NA
HS[8]	3 143 254	NA	NA	NA
CE[19]	3 086 110	3 088 870	NA	NA
ECE[19]	3 084 649	3 087 847	NA	1.336
CDE[16]	3 083 930	3 090 542	NA	0.67
CDBCO[15]	3 081 400	3 088 500	NA	1.3
BBPSO[20]	3 062 144	3 067 277	NA	6.38
HHS[8]	3 057 313	NA	NA	23.042
HIGA[14]	3 055 435	3 058 126	3 066 755	NA
Proposed CSO	3 054 967	3 057 939	3 062 612	1.797

由表3可以看出，采用CSO算法求解30#机组DED优化问题时，取得的最小费用、平均费用和最大费用均小于其他算法，即使是改进算法和混合算法也不例外。通常情况下，机组规模越大，寻求全局最优解的难度越高。国内外对此问题的研究主要集中于30#机组及以下规模，至于60#机组及以上规模则较少涉足。为了进一步验证CSO算法在求解大规模DED问题中的优越性，将其应用到60#和120#机组测试系统中，仿真结果如表4所示。

表4 60#机组和120#机组求解结果

算法	最小费用/美元	平均费用/美元	最大费用/美元	运行时间/min
60#机组				
CSO	6 112 329	6 114 694	6 116 984	3.07
120#机组				
CSO	12 213 894	12 221 374	12 230 088	5.75

本文将CSO算法应用到两种模型中，最小费用分别为6 112 329美元和12 213 894美元，比10#机组最小费用的6倍6 246 078美元和12倍12 492 156美元还少，基本呈线性增长。在机组规模变大、变量数增多时，CSO算法仍然表现出较高的求解准确度和较快的收敛速度，进一步证明了CSO算法具有较好的鲁棒性。

从以上实例可以看出，CSO算法能够很好地平衡求解准确度与收敛速度这两方面的能力。且机组规模越大，这种优势越明显。这是由于当机组数增多时，费用函数局部最优点也成倍数增加，大多数单一算法甚至混合算法都无法跳出局部最优，而本文所用算法却能够利用纵向交叉操作使陷入停滞的维跳出局部最优，再通过横向交叉将信息传播至整个种群。这种双重搜索机制使结果呈链式反应在种群中蔓延，大大增加了算法的全局收敛能力和收敛速度。

结束语

文中采用一种全新的群智能随机搜索算法来解决DED优化问题。该算法结构简单，控制参数少。一般情况下横向交叉概率设置为1，纵向交叉概率取值范围为[0.2,0.8]，文中设置为0.6。算法最大创新点在于采用了双重搜索机制，其中纵向交叉操作保持了种群的多样性，解决了其他算法[21,22]在解决大规模非线性优化问题时普遍面临的早熟收敛问题，提高了全局搜索能力。竞争算子的引入，使种群时刻保持最优状态，加快了算法的收敛速度。因此，算法在全局搜索能力和收敛速度方面达到了一个很好的平衡。实验结果证明了算法的有效性。当然，本文尚存在不足之处，在研究的后期，还需要考虑风电的接入[23]以及旋转备用的优化[24]。

参考文献

[1] 吴杰康，郭壮志，秦砺寒，等．基于连续线性规划的梯级水电站优化调度 [J]．电网技术，2009，33 (8)：24-29．

[2] Attaviriyanupap P，Kita H，Tanaka E，et al．A hybrid EP and SQP for dynamic economic dispatch with nonsmooth fuel cost function [J]．IEEE Power Engineering Review，2002，17 (4)：77．

[3] 何小宇，张粒子，谢国辉．改进的拉格朗日松弛

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

法求解机组组合问题 [J]. 电力系统保护与控制，2010 (17)：16-21.

[4] Hemamalini S，Simon S P. Dynamic economic dispatch using artificial bee colony algorithm for units with valve-point effect [J]. European Transactions on Electrical Power，2011，21 (1)：70-81.

[5] Panigrahi C K，Chattopadhyay P K，Chakrabarti R N，et al. Simulated annealing technique for dynamic economic dispatch [J]. Electric Power Components and Systems，2006，34 (5)：577-586.

[6] Hemamalini S，Simon S P. Dynamic economic dispatch using artificial immune system for units with valve-point effect [J]. International Journal of Electrical Power & Energy Systems，2011，33 (4)：868-874.

[7] Balamurugan R，Subramanian S. Differential evolution-based dynamic economic dispatch of generating units with valve-point effects [J]. Electric Power Components and Systems，2008，36 (8)：828-843.

[8] Ravikumar Pandi V，Panigrahi B K. Dynamic economic load dispatch using hybrid swarm intelligence based harmony search algorithm [J]. Expert Systems with Applications，2011，38 (7)：8509-8514.

[9] Sivasubramani S，Swarup K S. Hybrid SOA-SQP algorithm for dynamic economic dispatch with valve-point effects [J]. Energy，2010，35 (12)：5031-5036.

[10] Mohammadi-Ivatloo B，Rabiee A，Soroudi A，et al. Imperialist competitive algorithm for solving non-convex dynamic economic power dispatch [J]. Energy，2012，44 (1)：228-240.

[11] Panigrahi B K，Ravikumar Pandi V，Das S. Adaptive particle swarm optimization approach for static and dynamic economic load dispatch [J]. Energy Conversion and Management，2008，49 (6)：1407-1415.

[12] Yuan X，Wang L，Yuan Y，et al. A modified differential evolution approach for dynamic economic dispatch with valve-point effects [J]. Energy Conversion and Management，2008，49 (12)：3447-3453.

[13] Niknam T，Golestaneh F. Enhanced bee swarm optimization algorithm for dynamic economic dispatch [J]. IEEE Systems Journal，2013，7 (4)：754-762.

[14] Mohammadi-Ivatloo B，Rabiee A，Soroudi A. Nonconvex dynamic economic power dispatch problems solution using hybrid immune-genetic algorithm [J]. IEEE Systems Journal，2013，7 (4)：777-785.

[15] Lu P，Zhou J，Zhang H，et al. Chaotic differential bee colony optimization algorithm for dynamic economic dispatch problem with valve-point effects [J]. International Journal of Electrical Power & Energy Systems，2014，62：130-143.

[16] Lu Y，Zhou J，Qin H，et al. Chaotic differential evolution methods for dynamic economic dispatch with valve-point effects [J]. Engineering Applications of Artificial Intelligence，2011，24 (2)：378-387.

[17] Basu M. Hybrid optimization of artificial immune systems and sequential quadratic programming for dynamic economic dispatch [J]. Electric Power Components and Systems，2009，37 (9)：1036-1045.

[18] Meng A，Chen Y，Yin H，et al. Crisscross optimization algorithm and its application [J]. Knowledge-Based Systems，2014，67：218-229.

[19] Immanuel Selvakumar A. Enhanced cross-entropy method for dynamic economic dispatch with valve-point effects [J]. International Journal of Electrical Power & Energy Systems，2011，33 (3)：783-790.

[20] Zhang Y，Gong D，Geng N，et al. Hybrid bare-bones PSO for dynamic economic dispatch with valve-point effects [J]. Applied Soft Computing，2014，18：248-260.

(下转第49页)

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

2013年7月14日20时06分12秒885毫秒，500 kV蓄增甲线蓄能B厂侧开关A相短路，500 kV电网A相电压降低。从图8的录波数据可知，t_1（0时刻）之前，电网电压为527.6 kV，水乡STATCOM在稳态调压模式下发出8 Mvar容性无功功率。由500 kV专用变压器的高压侧零序电流$3I_0$可知，t_1时刻电网发生接地故障，$t_3=59$ ms继电保护动作切除电网故障，电网电压迅速恢复正常。

电网故障导致水乡站500 kV母线电压A相瞬时跌落至254 kV（正常值的84%），满足了STATCOM进入暂态模式的条件2（$u<282.3$ kV，$t=5$ ms），水乡站STATCOM迅速切换至暂态电压控制模式，并在电网故障后19 ms即达到大容量的无功输出。由图1知，STATCOM是专用变压器的唯一负荷，可以通过监测变压器低压侧电流来观测STATCOM的无功输出。本次STATCOM暂态电压控制期间专用变压器低电流有效值最大为4.65 kA，对应时刻的专用变压器低电压线电压有效值为37.8 kV，可计算最大无功输出为304 Mvar。

根据自动控制原理对响应时间定义，计无功电流达到90%最大值的时间为STATCOM的暂态响应时间。从图8可知，水乡站STATCOM的暂态响应时间约为19 ms，低于设计要求的20 ms。而本次暂态启动期间水乡STATCOM的最大无功输出也达到了设计要求的300 Mvar。

STATCOM能够在电网故障发生后迅速进入暂态电压控制模式，在继电保护动作前，通过大容量无功功率的注入支撑电网电压，有利于提高电网稳定性。

结束语

百兆乏级STATCOM在国内外的工程应用仅有少数例子，本文结合水乡站±200 Mvar STATCOM投运半年的实际参数和运行数据，详细介绍了其电路结构和应用效果，可为后续其他STATCOM装置的系统设计、装置研发和工程实践提供借鉴。

参考文献

[1] 赵晋泉，孙晓明，龚成明，等．含FACTS元件的电力系统电压稳定评估 [J]．电力系统自动化，2011，35 (16)：21-26．

[2] 周建丰，顾亚琴，韦寿祺．SVC与STATCOM的综合比较分析 [J]．电力自动化设备，2007，28 (12)：77-80．

[3] 高凯，黄华，邹彬，等．上海西郊变电站STATCOM的性能测试及试运行情况 [J]．电力设备，2013，8 (7)：27-30．

[4] 谢化安，王永源，万四维，等．东莞变电站链式STATCOM控制保护的RTDS试验研究 [J]．电力系统控制与保护，2013，41 (4)：117-122．

[5] 刘锦宁，李旷，张海涛，等．高压大容量动态无功补偿装置STATCOM起动方式的研究 [J]．电气技术，2012 (2)：29-31．

[6] 许树楷，陈名，傅闯，等．南方电网±200 Mvar静止同步补偿装置系统调试 [J]．南方电网技术，2012，6 (2)：21-25．

（收稿日期：2015－01－20）EA

(上接第23页)

[21] 张建伟，侯朝桢．发电动态经济调度的有效响应能力调整量算法 [J]．电力系统自动化，2004 (22)：27-31．

[22] 韩学山，柳焯．考虑机组爬坡速度和网络安全约束的经济调度解耦算法 [J]．电力系统自动化，2002，26 (13)：32-37．

[23] 王粤，肖进永，栗然．大规模风电场并网的经济调度建模和决策算法 [J]．电气应用，2009，28 (12)：50-54．

[24] 刘清泉，刘文霞，张建华．基于DSM算法的风火联合系统旋转备用优化 [J]．电气应用，2013，32 (5)：76-80．

（收稿日期：2015－02－01）EA

(C)1994-2023 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net