

Development of Multi-Objective Load Shedding Optimization via Back Tracking Search Algorithm with Novel Reactive Power Tracing Index

R. Verayiah
Universiti Tenaga
Nasional
renuga@uniten.edu.my

A. Mohamed
Universiti Kebangsaan
Malaysia
azah@eng.ukm.my

H. Shareef
United Arab Emirates
University
hussain_ln@yahoo.com

I.H.Z. Abidin
Universiti Tenaga
Nasional
izham@uniten.edu.my

Abstract— This paper demonstrates on the development of weak bus identification method using novel reactive power tracing index for determination and selection on appropriate buses for load shedding purpose. Power tracing principles using downstream looking algorithm is utilized for the development of a novel reactive power tracing index known as LQP_LT. The results obtained from the weak bus identification from various power system contingencies are analyzed for determination of the load shed amount. Optimization on the total amount of load shed is customized based on multi-objective functions via backtracking search algorithm. The objective functions are set to minimize the amount of load shedding and to minimize the total LQP_LT indices for all buses. A standard IEEE 57 test bus system is used as a data platform for the power flow simulations carried out in MATLAB environment. The results obtained post- load shedding for the proposed method is found to be robust, efficient and reliable for the implementation and execution of optimized amount and location selection for Under Voltage Load Shedding scheme in large network.

Keywords—reactive power tracing; load shedding; back tracking search

I. INTRODUCTION

Power system stability has gained much attention and has been acknowledged as an important problem for secure system operation [1]. Many major blackouts occurrence due to power system instability have demonstrated the importance of this phenomenon [1-5]. Initially, angle stability has been of primary concern to power utilities. However, in the last two decades power systems have been operating under more stressed conditions than previously due to several factors such as continuing growth in interconnections and integration of emerging new technologies. Limitations to install new generating plants due the environmental pressures and economical reason caused the transmission lines to operate near the loading limits in order to cater for the electricity power consumption in heavy load areas. In addition, the system loading pattern due to deregulation in the electricity market, the growing use of induction machines and large penetration of renewable energy sources in distribution systems have made local coordination control system to appear more complex. All

these stressed conditions in a power system exhibit a new type of unstable behavior which is known as voltage instability.

Voltage instability has been regarded as one of the major cause of power system insecurity. A voltage instability phenomenon takes place when the receiving end voltage decreases well below its normal operating point. Severe voltage instability may lead to voltage collapse which is the process by which the sequence of events accompanying voltage instability leads to a blackout or abnormally low voltages in a significant portion of a power system [6]. Table I shows the documented voltage collapse incidents resulting from voltage instability as a principal cause for system blackouts.

TABLE I. DOCUMENTED VOLTAGE COLLAPSE INCIDENTS [6]

Date	Location	Time Frame	Interrupted Load	Remarks
11 November 2009	Brazil and Paraguay	68s (after initial event)	24731 MW	Voltage collapse in part of the system. Number of people affected: 87 million.
12 July 2004	Southern Greece	30 min	9,000 MW	Number of people affected: 5 million
14 August 2003	United States and Canada	39 min	63,000 MW	Estimated cost: US\$7–10 billion. Number of people affected: 50 million.

In order to prevent power system from wide-spread voltage collapse situation, load shedding is applied as a final safety measure in the mitigation plan. Generally, load shedding can be categorized into two types; under- voltage load shedding (UVLS) and under-frequency load shedding (UFLS). The main goal of UVLS scheme is to shed load in order to restore reactive power relative to demand and to contain the voltage problem within a local area rather than allowing it to spread in geography and magnitude [7].

This paper presents on the development of weak bus identification method using novel reactive power tracing index named as LQP_LT index, for determination and selection on appropriate buses for load shedding purpose. The results obtained from the weak bus identification from various power

system contingencies are analyzed for determination of the load shed amount. Optimization on the total amount of load shed is customized based on multi-objective functions using back tracking search algorithm (BSA). The results obtained for post – load shedding by implementing this new method is very promising and would be suitable for online implementation for under- voltage load shedding scheme (UVLS) for large network.

II. DESCRIPTION OF THE SYSTEM STUDY

The power flow model analysis for the system is performed using MATPOWER in MATLAB. The transmission network studied consists of 57 buses, 80 lines, 7 generators and 40 load buses. The network model of the standard IEEE 57 test bus system [8] is shown in Figure 1. The base case simulations and contingency simulations for power flow are carried out using optimal power flow. The system generates 1278.66 MW and 321.08 MVAR while the load demand is 1250.80MW and 336.40MVAR at base case.

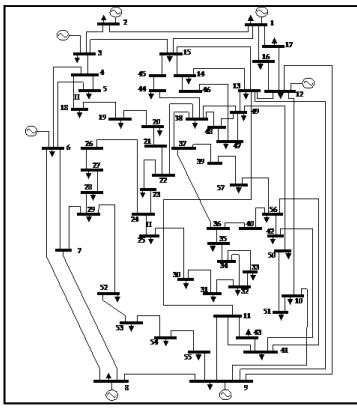


Fig. 1. Single line diagram of IEEE 57 bus system [8].

III. REACTIVE POWER TRACING CONCEPT AND FORMULATION FOR NOVEL LQP_LT INDEX

Load tracing is defined as a task to trace the power contributed by an individual load.

A. Formulation of LQP_LT index

A transmission line can be either absorbing or generating reactive power. The transmission element's contribution to the reactive power flows depends of its π equivalent circuit and the voltage magnitude at its terminals. In order to make the complexity of reactive power tracing accurate, a generalized π equivalent model were developed by considering all possible reactive power flow directions, which are either generations or absorptions, at both terminals of sending and receiving, as well as inside the series impedance of the network elements of the 14 bus system.

Utilizing [9-11], with appropriate modification performed for the purpose of reactive power flow derivation, the flow Q_{lm} on line l-m can be expressed as a summation of load components as in equation (1), where n is the total number of loads in the network.

$$Q_{lm} = Q_{lm}^{L1} + Q_{lm}^{L2} + Q_{lm}^{L3} + \dots + Q_{lm}^{Ln} \quad (1)$$

The component of load defined as Q_{Li} on line l-m is expressed as a fraction x_{lm}^i of load Q_{Li} and written as follows:

$$Q_{lm}^{Li} = x_{lm}^i \cdot Q_{Li} \quad (2)$$

thus,

$$Q_{lm} = \sum_{i=1}^{Ln} x_{lm}^i \cdot Q_{Li} \quad (3)$$

Applying the above concept into LQP_LT of line l-m for summation of individual load components, gives Eq. (4):

$$\begin{aligned} LQP_{LT,lm} = & LQP_{lm}^{L1} + LQP_{lm}^{L2} + LQP_{lm}^{L3} + \\ & \dots + LQP_{lm}^{Ln} \end{aligned} \quad (4)$$

or can also be written as:

$$LQP_{LT,lm} = 4 \left(\frac{X}{V_s^2} \left[\sum_{i=1}^{Ln} Q_{r,lm}^i \right] + \frac{XP_s^2}{V_s^2} \right) \quad (5)$$

The index LQP_LT computed for every line should remain within a value between 0 and 1. Index value '0' indicates very stable while index value '1' indicates unstable. The algorithm is developed in MATLAB environment to automate the reactive tracing in the system study and the power fraction contribution computation for LQP_LT index is formulated accurately for contingency analysis computation.

IV. UNDER VOLTAGE LOAD SHEDDING OPTIMIZATION USING BACK TRACKING SEARCH (BSA)

The BSA algorithm used in this study is a new Evolutionary Algorithm (EA) that is implemented for under voltage load shedding scheme assessment. It is developed by [12] in order to solve numerical optimization problems. The development of BSA is based on random mutation strategy. Figure 2 shows the general pseudo code for BSA processes.

```

Input: ObjFunc, N, D, maxgen, maxeval, fmax, Dc, npop, D
Output: BestChromo, BestChromoIndex, fmax, npop, D
1. Initialize population P, Chromo, fitness, fmax, npop, D
2. For i From 1 to N do
3.   P_i = ObjFunc(Chromo(i)) + fmax; // Initialization of population, P.
4.   Chromo(i) = Chromo(i) + 1; // Initialization of chromosome
5.   fitness(i) = ObjFunc(P_i); // Initial-Fitness values of P
6. end
7. Chromo = Chromo(permute(1:N));
8. BestChromo = Chromo(1);
9. BestChromoIndex = 1;
10. for i = 1:N do
11.   if fitness(i) < fmax then
12.     BestChromo = P_i;
13.     BestChromoIndex = i;
14.   end
15. end
16. Chromo = Chromo(permute(1:N));
17. for i = 1:N do
18.   Chromo(i) = Chromo(i) + 1;
19. end
20. Chromo = Chromo(permute(1:N));
21. for i = 1:N do
22.   Chromo(i) = Chromo(i) + 1;
23. end
24. Chromo = Chromo(permute(1:N));
25. for i = 1:N do
26.   Chromo(i) = Chromo(i) + 1;
27. end
28. Chromo = Chromo(permute(1:N));
29. for i = 1:N do
30.   Chromo(i) = Chromo(i) + 1;
31. end
32. Chromo = Chromo(permute(1:N));
33. for i = 1:N do
34.   Chromo(i) = Chromo(i) + 1;
35. end
36. Chromo = Chromo(permute(1:N));
37. for i = 1:N do
38.   Chromo(i) = Chromo(i) + 1;
39. end
40. Chromo = Chromo(permute(1:N));
41. for i = 1:N do
42.   Chromo(i) = Chromo(i) + 1;
43. end
44. Chromo = Chromo(permute(1:N));
45. for i = 1:N do
46.   Chromo(i) = Chromo(i) + 1;
47. end
48. Chromo = Chromo(permute(1:N));
49. for i = 1:N do
50.   Chromo(i) = Chromo(i) + 1;
51. end
52. Chromo = Chromo(permute(1:N));
53. for i = 1:N do
54.   Chromo(i) = Chromo(i) + 1;
55. end
56. Chromo = Chromo(permute(1:N));
57. for i = 1:N do
58.   Chromo(i) = Chromo(i) + 1;
59. end
60. Chromo = Chromo(permute(1:N));
61. for i = 1:N do
62.   Chromo(i) = Chromo(i) + 1;
63. end
64. Chromo = Chromo(permute(1:N));
65. for i = 1:N do
66.   Chromo(i) = Chromo(i) + 1;
67. end
68. Chromo = Chromo(permute(1:N));
69. for i = 1:N do
70.   Chromo(i) = Chromo(i) + 1;
71. end
72. Chromo = Chromo(permute(1:N));
73. for i = 1:N do
74.   Chromo(i) = Chromo(i) + 1;
75. end
76. Chromo = Chromo(permute(1:N));
77. for i = 1:N do
78.   Chromo(i) = Chromo(i) + 1;
79. end
80. Chromo = Chromo(permute(1:N));
81. for i = 1:N do
82.   Chromo(i) = Chromo(i) + 1;
83. end
84. Chromo = Chromo(permute(1:N));
85. for i = 1:N do
86.   Chromo(i) = Chromo(i) + 1;
87. end
88. Chromo = Chromo(permute(1:N));
89. for i = 1:N do
90.   Chromo(i) = Chromo(i) + 1;
91. end
92. Chromo = Chromo(permute(1:N));
93. for i = 1:N do
94.   Chromo(i) = Chromo(i) + 1;
95. end
96. Chromo = Chromo(permute(1:N));
97. for i = 1:N do
98.   Chromo(i) = Chromo(i) + 1;
99. end
100. Chromo = Chromo(permute(1:N));
101. for i = 1:N do
102.   Chromo(i) = Chromo(i) + 1;
103. end
104. Chromo = Chromo(permute(1:N));
105. for i = 1:N do
106.   Chromo(i) = Chromo(i) + 1;
107. end
108. Chromo = Chromo(permute(1:N));
109. for i = 1:N do
110.   Chromo(i) = Chromo(i) + 1;
111. end
112. Chromo = Chromo(permute(1:N));
113. for i = 1:N do
114.   Chromo(i) = Chromo(i) + 1;
115. end
116. Chromo = Chromo(permute(1:N));
117. for i = 1:N do
118.   Chromo(i) = Chromo(i) + 1;
119. end
120. Chromo = Chromo(permute(1:N));
121. for i = 1:N do
122.   Chromo(i) = Chromo(i) + 1;
123. end
124. Chromo = Chromo(permute(1:N));
125. for i = 1:N do
126.   Chromo(i) = Chromo(i) + 1;
127. end
128. Chromo = Chromo(permute(1:N));
129. for i = 1:N do
130.   Chromo(i) = Chromo(i) + 1;
131. end
132. Chromo = Chromo(permute(1:N));
133. for i = 1:N do
134.   Chromo(i) = Chromo(i) + 1;
135. end
136. Chromo = Chromo(permute(1:N));
137. for i = 1:N do
138.   Chromo(i) = Chromo(i) + 1;
139. end
140. Chromo = Chromo(permute(1:N));
141. for i = 1:N do
142.   Chromo(i) = Chromo(i) + 1;
143. end
144. Chromo = Chromo(permute(1:N));
145. for i = 1:N do
146.   Chromo(i) = Chromo(i) + 1;
147. end
148. Chromo = Chromo(permute(1:N));
149. for i = 1:N do
150.   Chromo(i) = Chromo(i) + 1;
151. end
152. Chromo = Chromo(permute(1:N));
153. for i = 1:N do
154.   Chromo(i) = Chromo(i) + 1;
155. end
156. Chromo = Chromo(permute(1:N));
157. for i = 1:N do
158.   Chromo(i) = Chromo(i) + 1;
159. end
160. Chromo = Chromo(permute(1:N));
161. for i = 1:N do
162.   Chromo(i) = Chromo(i) + 1;
163. end
164. Chromo = Chromo(permute(1:N));
165. for i = 1:N do
166.   Chromo(i) = Chromo(i) + 1;
167. end
168. Chromo = Chromo(permute(1:N));
169. for i = 1:N do
170.   Chromo(i) = Chromo(i) + 1;
171. end
172. Chromo = Chromo(permute(1:N));
173. for i = 1:N do
174.   Chromo(i) = Chromo(i) + 1;
175. end
176. Chromo = Chromo(permute(1:N));
177. for i = 1:N do
178.   Chromo(i) = Chromo(i) + 1;
179. end
180. Chromo = Chromo(permute(1:N));
181. for i = 1:N do
182.   Chromo(i) = Chromo(i) + 1;
183. end
184. Chromo = Chromo(permute(1:N));
185. for i = 1:N do
186.   Chromo(i) = Chromo(i) + 1;
187. end
188. Chromo = Chromo(permute(1:N));
189. for i = 1:N do
190.   Chromo(i) = Chromo(i) + 1;
191. end
192. Chromo = Chromo(permute(1:N));
193. for i = 1:N do
194.   Chromo(i) = Chromo(i) + 1;
195. end
196. Chromo = Chromo(permute(1:N));
197. for i = 1:N do
198.   Chromo(i) = Chromo(i) + 1;
199. end
200. Chromo = Chromo(permute(1:N));
201. for i = 1:N do
202.   Chromo(i) = Chromo(i) + 1;
203. end
204. Chromo = Chromo(permute(1:N));
205. for i = 1:N do
206.   Chromo(i) = Chromo(i) + 1;
207. end
208. Chromo = Chromo(permute(1:N));
209. for i = 1:N do
210.   Chromo(i) = Chromo(i) + 1;
211. end
212. Chromo = Chromo(permute(1:N));
213. for i = 1:N do
214.   Chromo(i) = Chromo(i) + 1;
215. end
216. Chromo = Chromo(permute(1:N));
217. for i = 1:N do
218.   Chromo(i) = Chromo(i) + 1;
219. end
220. Chromo = Chromo(permute(1:N));
221. for i = 1:N do
222.   Chromo(i) = Chromo(i) + 1;
223. end
224. Chromo = Chromo(permute(1:N));
225. for i = 1:N do
226.   Chromo(i) = Chromo(i) + 1;
227. end
228. Chromo = Chromo(permute(1:N));
229. for i = 1:N do
230.   Chromo(i) = Chromo(i) + 1;
231. end
232. Chromo = Chromo(permute(1:N));
233. for i = 1:N do
234.   Chromo(i) = Chromo(i) + 1;
235. end
236. Chromo = Chromo(permute(1:N));
237. for i = 1:N do
238.   Chromo(i) = Chromo(i) + 1;
239. end
240. Chromo = Chromo(permute(1:N));
241. for i = 1:N do
242.   Chromo(i) = Chromo(i) + 1;
243. end
244. Chromo = Chromo(permute(1:N));
245. for i = 1:N do
246.   Chromo(i) = Chromo(i) + 1;
247. end
248. Chromo = Chromo(permute(1:N));
249. for i = 1:N do
250.   Chromo(i) = Chromo(i) + 1;
251. end
252. Chromo = Chromo(permute(1:N));
253. for i = 1:N do
254.   Chromo(i) = Chromo(i) + 1;
255. end
256. Chromo = Chromo(permute(1:N));
257. for i = 1:N do
258.   Chromo(i) = Chromo(i) + 1;
259. end
260. Chromo = Chromo(permute(1:N));
261. for i = 1:N do
262.   Chromo(i) = Chromo(i) + 1;
263. end
264. Chromo = Chromo(permute(1:N));
265. for i = 1:N do
266.   Chromo(i) = Chromo(i) + 1;
267. end
268. Chromo = Chromo(permute(1:N));
269. for i = 1:N do
270.   Chromo(i) = Chromo(i) + 1;
271. end
272. Chromo = Chromo(permute(1:N));
273. for i = 1:N do
274.   Chromo(i) = Chromo(i) + 1;
275. end
276. Chromo = Chromo(permute(1:N));
277. for i = 1:N do
278.   Chromo(i) = Chromo(i) + 1;
279. end
280. Chromo = Chromo(permute(1:N));
281. for i = 1:N do
282.   Chromo(i) = Chromo(i) + 1;
283. end
284. Chromo = Chromo(permute(1:N));
285. for i = 1:N do
286.   Chromo(i) = Chromo(i) + 1;
287. end
288. Chromo = Chromo(permute(1:N));
289. for i = 1:N do
290.   Chromo(i) = Chromo(i) + 1;
291. end
292. Chromo = Chromo(permute(1:N));
293. for i = 1:N do
294.   Chromo(i) = Chromo(i) + 1;
295. end
296. Chromo = Chromo(permute(1:N));
297. for i = 1:N do
298.   Chromo(i) = Chromo(i) + 1;
299. end
299. Chromo = Chromo(permute(1:N));
300. Chromo = Chromo(permute(1:N));
301. Chromo = Chromo(permute(1:N));
302. Chromo = Chromo(permute(1:N));
303. Chromo = Chromo(permute(1:N));
304. Chromo = Chromo(permute(1:N));
305. Chromo = Chromo(permute(1:N));
306. Chromo = Chromo(permute(1:N));
307. Chromo = Chromo(permute(1:N));
308. Chromo = Chromo(permute(1:N));
309. Chromo = Chromo(permute(1:N));
310. Chromo = Chromo(permute(1:N));
311. Chromo = Chromo(permute(1:N));
312. Chromo = Chromo(permute(1:N));
313. Chromo = Chromo(permute(1:N));
314. Chromo = Chromo(permute(1:N));
315. Chromo = Chromo(permute(1:N));
316. Chromo = Chromo(permute(1:N));
317. Chromo = Chromo(permute(1:N));
318. Chromo = Chromo(permute(1:N));
319. Chromo = Chromo(permute(1:N));
320. Chromo = Chromo(permute(1:N));
321. Chromo = Chromo(permute(1:N));
322. Chromo = Chromo(permute(1:N));
323. Chromo = Chromo(permute(1:N));
324. Chromo = Chromo(permute(1:N));
325. Chromo = Chromo(permute(1:N));
326. Chromo = Chromo(permute(1:N));
327. Chromo = Chromo(permute(1:N));
328. Chromo = Chromo(permute(1:N));
329. Chromo = Chromo(permute(1:N));
330. Chromo = Chromo(permute(1:N));
331. Chromo = Chromo(permute(1:N));
332. Chromo = Chromo(permute(1:N));
333. Chromo = Chromo(permute(1:N));
334. Chromo = Chromo(permute(1:N));
335. Chromo = Chromo(permute(1:N));
336. Chromo = Chromo(permute(1:N));
337. Chromo = Chromo(permute(1:N));
338. Chromo = Chromo(permute(1:N));
339. Chromo = Chromo(permute(1:N));
340. Chromo = Chromo(permute(1:N));
341. Chromo = Chromo(permute(1:N));
342. Chromo = Chromo(permute(1:N));
343. Chromo = Chromo(permute(1:N));
344. Chromo = Chromo(permute(1:N));
345. Chromo = Chromo(permute(1:N));
346. Chromo = Chromo(permute(1:N));
347. Chromo = Chromo(permute(1:N));
348. Chromo = Chromo(permute(1:N));
349. Chromo = Chromo(permute(1:N));
350. Chromo = Chromo(permute(1:N));
351. Chromo = Chromo(permute(1:N));
352. Chromo = Chromo(permute(1:N));
353. Chromo = Chromo(permute(1:N));
354. Chromo = Chromo(permute(1:N));
355. Chromo = Chromo(permute(1:N));
356. Chromo = Chromo(permute(1:N));
357. Chromo = Chromo(permute(1:N));
358. Chromo = Chromo(permute(1:N));
359. Chromo = Chromo(permute(1:N));
360. Chromo = Chromo(permute(1:N));
361. Chromo = Chromo(permute(1:N));
362. Chromo = Chromo(permute(1:N));
363. Chromo = Chromo(permute(1:N));
364. Chromo = Chromo(permute(1:N));
365. Chromo = Chromo(permute(1:N));
366. Chromo = Chromo(permute(1:N));
367. Chromo = Chromo(permute(1:N));
368. Chromo = Chromo(permute(1:N));
369. Chromo = Chromo(permute(1:N));
370. Chromo = Chromo(permute(1:N));
371. Chromo = Chromo(permute(1:N));
372. Chromo = Chromo(permute(1:N));
373. Chromo = Chromo(permute(1:N));
374. Chromo = Chromo(permute(1:N));
375. Chromo = Chromo(permute(1:N));
376. Chromo = Chromo(permute(1:N));
377. Chromo = Chromo(permute(1:N));
378. Chromo = Chromo(permute(1:N));
379. Chromo = Chromo(permute(1:N));
380. Chromo = Chromo(permute(1:N));
381. Chromo = Chromo(permute(1:N));
382. Chromo = Chromo(permute(1:N));
383. Chromo = Chromo(permute(1:N));
384. Chromo = Chromo(permute(1:N));
385. Chromo = Chromo(permute(1:N));
386. Chromo = Chromo(permute(1:N));
387. Chromo = Chromo(permute(1:N));
388. Chromo = Chromo(permute(1:N));
389. Chromo = Chromo(permute(1:N));
390. Chromo = Chromo(permute(1:N));
391. Chromo = Chromo(permute(1:N));
392. Chromo = Chromo(permute(1:N));
393. Chromo = Chromo(permute(1:N));
394. Chromo = Chromo(permute(1:N));
395. Chromo = Chromo(permute(1:N));
396. Chromo = Chromo(permute(1:N));
397. Chromo = Chromo(permute(1:N));
398. Chromo = Chromo(permute(1:N));
399. Chromo = Chromo(permute(1:N));
400. Chromo = Chromo(permute(1:N));
401. Chromo = Chromo(permute(1:N));
402. Chromo = Chromo(permute(1:N));
403. Chromo = Chromo(permute(1:N));
404. Chromo = Chromo(permute(1:N));
405. Chromo = Chromo(permute(1:N));
406. Chromo = Chromo(permute(1:N));
407. Chromo = Chromo(permute(1:N));
408. Chromo = Chromo(permute(1:N));
409. Chromo = Chromo(permute(1:N));
410. Chromo = Chromo(permute(1:N));
411. Chromo = Chromo(permute(1:N));
412. Chromo = Chromo(permute(1:N));
413. Chromo = Chromo(permute(1:N));
414. Chromo = Chromo(permute(1:N));
415. Chromo = Chromo(permute(1:N));
416. Chromo = Chromo(permute(1:N));
417. Chromo = Chromo(permute(1:N));
418. Chromo = Chromo(permute(1:N));
419. Chromo = Chromo(permute(1:N));
420. Chromo = Chromo(permute(1:N));
421. Chromo = Chromo(permute(1:N));
422. Chromo = Chromo(permute(1:N));
423. Chromo = Chromo(permute(1:N));
424. Chromo = Chromo(permute(1:N));
425. Chromo = Chromo(permute(1:N));
426. Chromo = Chromo(permute(1:N));
427. Chromo = Chromo(permute(1:N));
428. Chromo = Chromo(permute(1:N));
429. Chromo = Chromo(permute(1:N));
430. Chromo = Chromo(permute(1:N));
431. Chromo = Chromo(permute(1:N));
432. Chromo = Chromo(permute(1:N));
433. Chromo = Chromo(permute(1:N));
434. Chromo = Chromo(permute(1:N));
435. Chromo = Chromo(permute(1:N));
436. Chromo = Chromo(permute(1:N));
437. Chromo = Chromo(permute(1:N));
438. Chromo = Chromo(permute(1:N));
439. Chromo = Chromo(permute(1:N));
440. Chromo = Chromo(permute(1:N));
441. Chromo = Chromo(permute(1:N));
442. Chromo = Chromo(permute(1:N));
443. Chromo = Chromo(permute(1:N));
444. Chromo = Chromo(permute(1:N));
445. Chromo = Chromo(permute(1:N));
446. Chromo = Chromo(permute(1:N));
447. Chromo = Chromo(permute(1:N));
448. Chromo = Chromo(permute(1:N));
449. Chromo = Chromo(permute(1:N));
450. Chromo = Chromo(permute(1:N));
451. Chromo = Chromo(permute(1:N));
452. Chromo = Chromo(permute(1:N));
453. Chromo = Chromo(permute(1:N));
454. Chromo = Chromo(permute(1:N));
455. Chromo = Chromo(permute(1:N));
456. Chromo = Chromo(permute(1:N));
457. Chromo = Chromo(permute(1:N));
458. Chromo = Chromo(permute(1:N));
459. Chromo = Chromo(permute(1:N));
460. Chromo = Chromo(permute(1:N));
461. Chromo = Chromo(permute(1:N));
462. Chromo = Chromo(permute(1:N));
463. Chromo = Chromo(permute(1:N));
464. Chromo = Chromo(permute(1:N));
465. Chromo = Chromo(permute(1:N));
466. Chromo = Chromo(permute(1:N));
467. Chromo = Chromo(permute(1:N));
468. Chromo = Chromo(permute(1:N));
469. Chromo = Chromo(permute(1:N));
470. Chromo = Chromo(permute(1:N));
471. Chromo = Chromo(permute(1:N));
472. Chromo = Chromo(permute(1:N));
473. Chromo = Chromo(permute(1:N));
474. Chromo = Chromo(permute(1:N));
475. Chromo = Chromo(permute(1:N));
476. Chromo = Chromo(permute(1:N));
477. Chromo = Chromo(permute(1:N));
478. Chromo = Chromo(permute(1:N));
479. Chromo = Chromo(permute(1:N));
480. Chromo = Chromo(permute(1:N));
481. Chromo = Chromo(permute(1:N));
482. Chromo = Chromo(permute(1:N));
483. Chromo = Chromo(permute(1:N));
484. Chromo = Chromo(permute(1:N));
485. Chromo = Chromo(permute(1:N));
486. Chromo = Chromo(permute(1:N));
487. Chromo = Chromo(permute(1:N));
488. Chromo = Chromo(permute(1:N));
489. Chromo = Chromo(permute(1:N));
490. Chromo = Chromo(permute(1:N));
491. Chromo = Chromo(permute(1:N));
492. Chromo = Chromo(permute(1:N));
493. Chromo = Chromo(permute(1:N));
494. Chromo = Chromo(permute(1:N));
495. Chromo = Chromo(permute(1:N));
496. Chromo = Chromo(permute(1:N));
497. Chromo = Chromo(permute(1:N));
498. Chromo = Chromo(permute(1:N));
499. Chromo = Chromo(permute(1:N));
500. Chromo = Chromo(permute(1:N));
501. Chromo = Chromo(permute(1:N));
502. Chromo = Chromo(permute(1:N));
503. Chromo = Chromo(permute(1:N));
504. Chromo = Chromo(permute(1:N));
505. Chromo = Chromo(permute(1:N));
506. Chromo = Chromo(permute(1:N));
507. Chromo = Chromo(permute(1:N));
508. Chromo = Chromo(permute(1:N));
509. Chromo = Chromo(permute(1:N));
510. Chromo = Chromo(permute(1:N));
511. Chromo = Chromo(permute(1:N));
512. Chromo = Chromo(permute(1:N));
513. Chromo = Chromo(permute(1:N));
514. Chromo = Chromo(permute(1:N));
515. Chromo = Chromo(permute(1:N));
516. Chromo = Chromo(permute(1:N));
517. Chromo = Chromo(permute(1:N));
518. Chromo = Chromo(permute(1:N));
519. Chromo = Chromo(permute(1:N));
520. Chromo = Chromo(permute(1:N));
521. Chromo = Chromo(permute(1:N));
522. Chromo = Chromo(permute(1:N));
523. Chromo = Chromo(permute(1:N));
524. Chromo = Chromo(permute(1:N));
525. Chromo = Chromo(permute(1:N));
526. Chromo = Chromo(permute(1:N));
527. Chromo = Chromo(permute(1:N));
528. Chromo = Chromo(permute(1:N));
529. Chromo = Chromo(permute(1:N));
530. Chromo = Chromo(permute(1:N));
531. Chromo = Chromo(permute(1:N));
532. Chromo = Chromo(permute(1:N));
533. Chromo = Chromo(permute(1:N));
534. Chromo = Chromo(permute(1:N));
535. Chromo = Chromo(permute(1:N));
536. Chromo = Chromo(permute(1:N));
537. Chromo = Chromo(permute(1:N));
538. Chromo = Chromo(permute(1:N));
539. Chromo = Chromo(permute(1:N));
540. Chromo = Chromo(permute(1:N));
541. Chromo = Chromo(permute(1:N));
542. Chromo = Chromo(permute(1:N));
543. Chromo = Chromo(permute(1:N));
544. Chromo = Chromo(permute(1:N));
545. Chromo = Chromo(permute(1:N));
546. Chromo = Chromo(permute(1:N));
547. Chromo = Chromo(permute(1:N));
548. Chromo = Chromo(permute(1:N));
549. Chromo = Chromo(permute(1:N));
550. Chromo = Chromo(permute(1:N));
551. Chromo = Chromo(permute(1:N));
552. Chromo = Chromo(permute(1:N));
553. Chromo = Chromo(permute(1:N));
554. Chromo = Chromo(permute(1:N));
555. Chromo = Chromo(permute(1:N));
556. Chromo = Chromo(permute(1:N));
557. Chromo = Chromo(permute(1:N));
558. Chromo = Chromo(permute(1:N));
559. Chromo = Chromo(permute(1:N));
560. Chromo = Chromo(permute(1:N));
561. Chromo = Chromo(permute(1:N));
562. Chromo = Chromo(permute(1:N));
563. Chromo = Chromo(permute(1:N));
564. Chromo = Chromo(permute(1:N));
565. Chromo = Chromo(permute(1:N));
566. Chromo = Chromo(permute(1:N));
567. Chromo = Chromo(permute(1:N));
568. Chromo = Chromo(permute(1:N));
569. Chromo = Chromo(permute(1:N));
570. Chromo = Chromo(permute(1:N));
571. Chromo = Chromo(permute(1:N));
572. Chromo = Chromo(permute(1:N));
573. Chromo = Chromo(permute(1:N));
574. Chromo = Chromo(permute(1:N));
575. Chromo = Chromo(permute(1:N));
576. Chromo = Chromo(permute(1:N));
577. Chromo = Chromo(permute(1:N));
578. Chromo = Chromo(permute(1:N));
579. Chromo = Chromo(permute(1:N));
580. Chromo = Chromo(permute(1:N));
581. Chromo = Chromo(permute(1:N));
582. Chromo = Chromo(permute(1:N));
583. Chromo = Chromo(permute(1:N));
584. Chromo = Chromo(permute(1:N));
585. Chromo = Chromo(permute(1:N));
586. Chromo = Chromo(permute(1:N));
587. Chromo = Chromo(permute(1:N));
588. Chromo = Chromo(permute(1:N));
589. Chromo = Chromo(permute(1:N));
590. Chromo = Chromo(permute(1:N));
591. Chromo = Chromo(permute(1:N));
592. Chromo = Chromo(permute(1:N));
593. Chromo = Chromo(permute(1:N));
594. Chromo = Chromo(permute(1:N));
595. Chromo = Chromo(permute(1:N));
596. Chromo = Chromo(permute(1:N));
597. Chromo = Chromo(permute(1:N));
598. Chromo = Chromo(permute(1:N));
599. Chromo = Chromo(permute(1:N));
600. Chromo = Chromo(permute(1:N));
601. Chromo = Chromo(permute(1:N));
602. Chromo = Chromo(permute(1:N));
603. Chromo = Chromo(permute(1:N));
604. Chromo = Chromo(permute(1:N));
605. Chromo = Chromo(permute(1:N));
606. Chromo = Chromo(permute(1:N));
607. Chromo = Chromo(permute(1:N));
608. Chromo = Chromo(permute(1:N));
609. Chromo = Chromo(permute(1:N));
610. Chromo = Chromo(permute(1:N));
611. Chromo = Chromo(permute(1:N));
612. Chromo = Chromo(permute(1:N));
613. Chromo = Chromo(permute(1:N));
614. Chromo = Chromo(permute(1:N));
615. Chromo = Chromo(permute(1:N));
616. Chromo = Chromo(permute(1:N));
617. Chromo = Chromo(permute(1:N));
618. Chromo = Chromo(permute(1:N));
619. Chromo = Chromo(permute(1:N));
620. Chromo = Chromo(permute(1:N));
621. Chromo = Chromo(permute(1:N));
622. Chromo = Chromo(permute(1:N));
623. Chromo = Chromo(permute(1:N));
624. Chromo = Chromo(permute(1:N));
625. Chromo = Chromo(permute(1:N));
626. Chromo = Chromo(permute(1:N));
627. Chromo = Chromo(permute(1:N));
628. Chromo = Chromo(permute(1:N));
629. Chromo = Chromo(permute(1:N));
630. Chromo = Chromo(permute(1:N));
631. Chromo = Chromo(permute(1:N));
632. Chromo = Chromo(permute(1:N));
633. Chromo = Chromo(permute(1:N));
634. Chromo = Chromo(permute(1:N));
635. Chromo = Chromo(permute(1:N));
636. Chromo = Chromo(permute(1:N));
637. Chromo = Chromo(permute(1:N));
638. Chromo = Chromo(permute(1:N));
639. Chromo = Chromo(permute(1:N));
640. Chromo = Chromo(permute(1:N));
641. Chromo = Chromo(permute(1:N));
642. Chromo = Chromo(permute(1:N));
643. Chromo = Chromo(permute(1:N));
644. Chrom
```

MATLAB, the following multi-objectives function shown in Eq. 6 is formed in order to solve the under voltage load shedding problem.

$$f = \min(\sum LQP_{LT}, \sum_{i=1}^n (P_{shed_i} + Q_{shed_i})) \quad (6)$$

The objective function above is programmed as bi-level programming whereby the first problem are constrained to be the optimal solution of the lower-level problem. To obtain a feasible solution for the bi-level optimization function defined, the following constraints were established.

$$P_{Gi}^0 - P_{Di}^0 + \Delta P_{Di} = \sum_{j=1}^N |V_i| |V_j| |Y_{ij}| \cos(\delta_{ij} + \delta_j - \delta_i) \quad (7)$$

$$Q_{Gi}^0 - Q_{Di}^0 + \Delta Q_{Di} = \sum_{j=1}^N |V_i| |V_j| |Y_{ij}| \cos(\delta_{ij} + \delta_j - \delta_i) \quad (8)$$

$$\frac{\Delta P_{Di}}{\Delta P_{Di}^0} = \frac{\Delta Q_{Di}}{\Delta Q_{Di}^0} \quad \text{fixed power factor} \quad (9)$$

$$V_{i,lower} \leq V_i \leq V_{i,upper} \quad (10)$$

Parameters P_{Di} and Q_{Di} terms refer to the real and reactive load demands on bus I, while P_{Gi} refers to the real power from a generator bus i. Index “0” refers to parameters at initial stage. The control variables that leads to an optimal solution are ΔP_{Di} and ΔQ_{Di} . The operational constraint which is the system voltage is defined as:

$$0.85 \leq V_i \leq 1.06 \quad (11)$$

The system is considered stable if the all the bus voltages lies within 0.85 p.u. to 1.06 p.u.

With the above formulas, the BSA algorithm is programmed to deliver precise, robust and simplified optimization solution.

V. CASE STUDY AND SIMULATION RESULTS

The methodology proposed is simulated using two different case studies. The first case is for heavily loading conditions while the second case is for heavily loaded with N-1 contingency conditions.

A. Heavily Loaded Conditions

The base case loading is increased with constant power factor for all load buses. The generations maintains at base values so that demand exceeded generations and voltage collapse is encountered. At loading factor of 2.00, the system is at verge of instability and lowest voltage is recorded at bus 30, 31, 32 and 33 with values of 0.719 p.u., 0.650 p.u., 0.703 p.u. and 0.697 p.u. Further increase on the load buses with total loading factor of 2.188 has stressed the system beyond its

convergence limit. At this point, the system is facing voltage collapse. Figure 3 and Figure 4 shows the voltage profile at verge of instability and after voltage collapse. The maximum real and reactive powers that need to be shed are computed as 179.393MW and 88.527 MVAR. The reactive power tracing

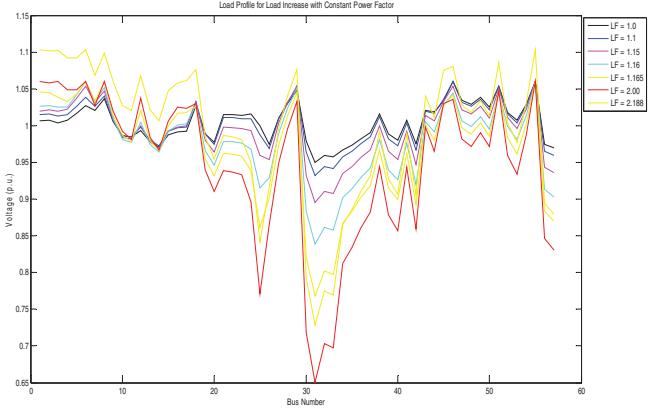


Fig. 3. Voltage profiles at loading factor of 2.00

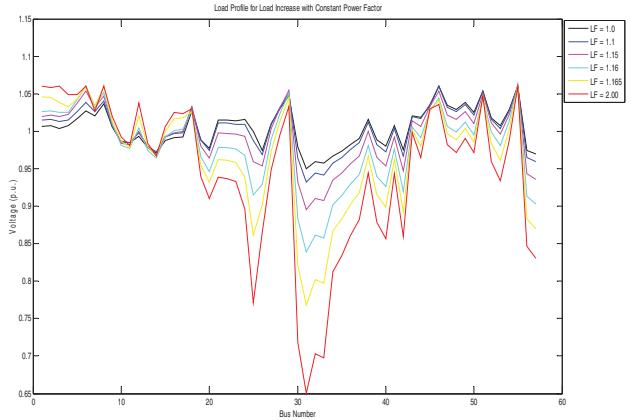


Fig. 4. Voltage profiles at loading factor of 2.188

index is computed from the power flow analysis for maximum loading factor before system collapse. The total LQP_LT for the system at this point is 78.96. Table II shows the priority ranking results based on maximum LQP_LT computed at each load buses. These are the buses selected for under voltage load shedding implementation. Figure 5 shows the LQP_LT index computed at each line due to loads in the system. Load amounts on the selected load buses are shed and optimized using the proposed BSA algorithm. The system attains all the objectives set at 50th iterations. The voltages at all buses increased to a minimum value of 0.85 and maximum value of 1.06. The total LQP_LT computed after load shedding is 57.129 and the minimum amount of load shed is 68.228MW and 71.816MVAR. Figure 6 shows the LQP_LT index computed at each line due loads after load shedding is performed. The index values at all lines has reduced to a much lower value comparatively. At this point, the system convergence is also achieved.

TABLE II. PRIORITY RANKING RESULTS FOR WEAKEST LOAD BUSES

Rank	Load Buses	LQP_LT
1	33	0.40281932
2	25	0.37748132
3	57	0.30618249
4	31	0.30027011
5	30	0.29396103
6	18	0.28268256
7	32	0.27620438
8	42	0.22566778
9	56	0.21328637
10	49	0.19594214
11	41	0.18659278

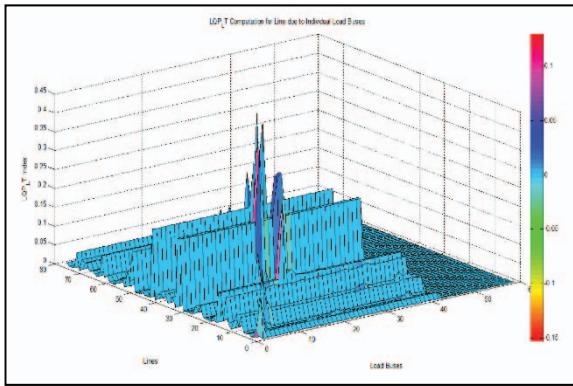


Fig. 5. LQP_LT index computed at each line

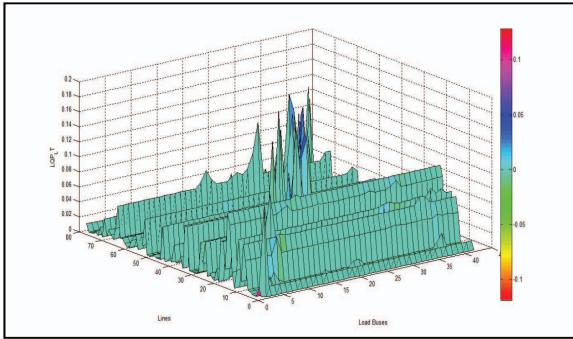


Fig. 6. LQP_LT index computed at each lines after load shedding optimization using BSA

Table III shows the minimized LQP_LT index ranked from highest to lowest for the part of the load buses. Figure 7 shows the voltage profile improvement recorded at every iteration until system convergence is achieved. The voltages are seen to have increased and stayed within the desired stable region at the end of the load shedding optimization process.

B. Heavily Loaded with N-1 Contingency Conditions

The optimal load flow simulation for the single line contingency, N-1, is performed in the same method carried out

as the first case. Line connection from bus 22 to bus 23 is set to be offline.

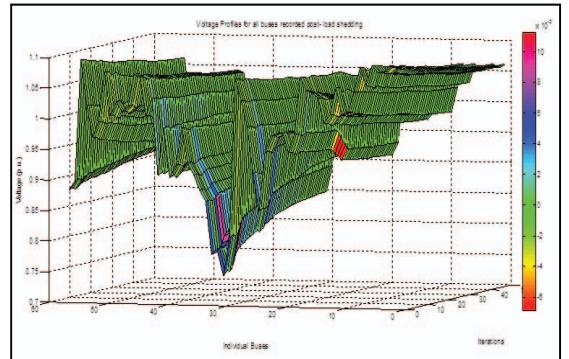


Fig. 7. Voltage profiles recorded at every iterations when performing load shedding optimization using BSA algorithm

TABLE III. PRIORITY RANKING RESULTS FOR WEAKEST LOAD BUSES POST LOAD-SHEDDING

Rank	Load Buses	LQP_LT
1	19	0.19567890
2	18	0.188701689
3	17	0.16071992
4	25	0.158208105
5	33	0.156603837
6	49	0.147148026
7	31	0.143793466
8	57	0.132941795
9	56	0.122751281
10	50	0.12261890
11	42	0.122522827

C. Heavily Loaded with N-1 Contingency Conditions

The optimal load flow simulation for the single line contingency, N-1, is performed in the same method carried out as the first case. Line connection from bus 22 to bus 23 is set to be offline. The load increments were performed in stages with constant power factor, and it was found that the system can have a maximum loading factor of 1.7266 before the system achieve divergence state. The systems become voltage collapse when the loading factor is increased to 1.7439. Figure 8 shows the voltage profile at voltage collapse. The maximum real and reactive powers that need to be shed are computed as 21.60MW and 5.81 MVAR. The total LQP_LT for the system at verge of instability is found to be 67.354.

Figure 9 shows the LQP_LT computed at every line due to heavy loads and line outage. Table IV shows the priority ranking results based on maximum LQP_LT computed at each load buses. The top five load buses are selected for under voltage load shedding implementation. Load amount on the selected load buses are shed and optimized using the proposed BSA algorithm. The system attains all the objectives set at 17th iterations. The voltages at all buses increased to a minimum value of 0.85 and maximum value of 1.06. The total LQP_LT

computed after load shedding is 53.4220 and the minimum amount of load shed is 9.944MW and 3.053MVAR. Figure 10 shows the LQP_LT index computed at each line due loads after load shedding is performed. The index values at all lines has reduced to a much lower value comparatively. At this point, the system convergence is also achieved.

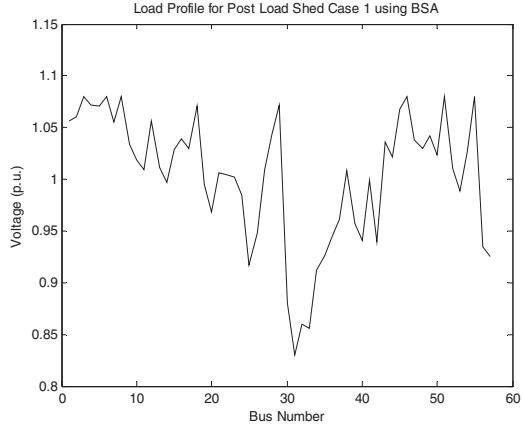


Fig. 8. Voltage profiles for N-1 contingency at Voltage Collpase

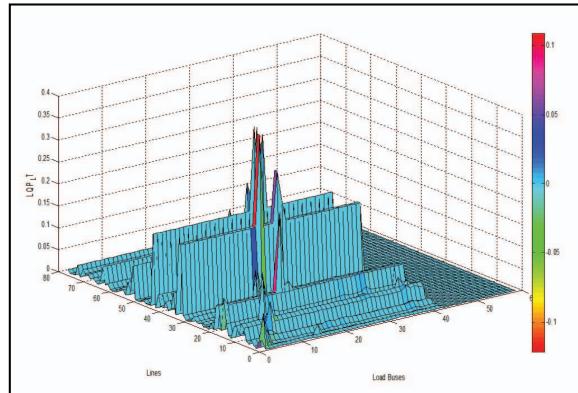


Fig. 9. LQP_LT index computed at each line at verge of instability

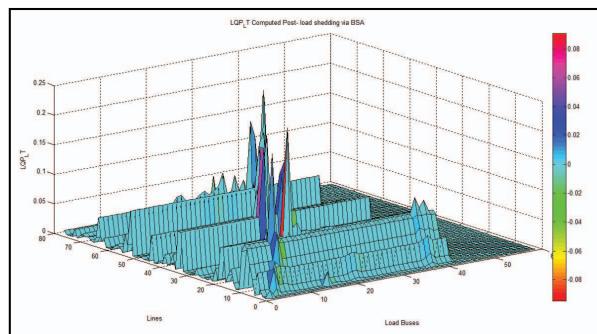


Fig. 10. LQP_LT index computed at each lines after load shedding

Table V shows the minimized LQP_LT index ranked from highest to lowest for part of the load buses. Figure 11 shows the voltage profile improvement recorded at every iteration

until system convergence is achieved. The voltages are seen to have increased and stayed within the desired stable region at the end of the load shedding optimization process.

TABLE IV.

PRIORITY RANKING RESULTS FOR WEAKEST LOAD BUSES IN N-1 CONTINGENCY

Rank	Load Bus	LQP_LT
1	25	0.37432002
2	31	0.35677248
3	33	0.334199388
4	30	0.283343264
5	57	0.243293417
6	18	0.241823738
7	32	0.236724676
8	56	0.200286334
9	42	0.176282688

TABLE V.

PRIORITY RANKING RESULTS POST LOAD SHEDDING IN N-1 CONTINGENCY POST LOAD- SHEDDING

LOAD BUSES	LQP_LT MAXIMUM
18	0.245061111
9	0.224280125
57	0.189122591
25	0.182136199
56	0.181216469
33	0.177398107
42	0.165236686
49	0.161157683
31	0.149543534
41	0.148841083

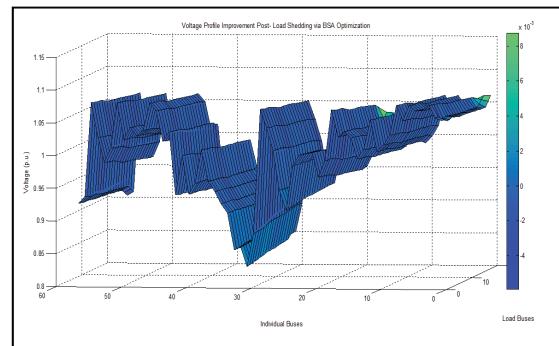


Fig. 11. Voltage profiles recorded at every iterations when performing load shedding optimization usng BSA algorithm

VI. CONCLUSIONS

To summarize, a new approach of multi-level optimization for under voltage load shedding has been recommended. The method implements line stability factor, LQP_LT which has the ability to trace the stressed lines contributed by individual

load in a system. Enabling the priority ranking list based on the traced LQP_LT, system operator can perform an accurate selection of critical load bus prior to performing any corrective action against voltage instability condition. The optimization results obtained on the amount of load to shed via BSA algorithm simulated on a large network has provided the best global minimum solution to recover the system from stressed conditions and improve the overall system voltage stability. The BSA algorithm is found to be effective and robust in providing the optimum solution. Finally, this study has established the capability and implementation of power tracing techniques with BSA optimization solution in a large power system network for under voltage load shedding scheme.

ACKNOWLEDGMENT

The authors gratefully acknowledge Universiti Tenaga Nasional for the financial support via research project UNIIG 2016 and Universiti Kebangsaan Malaysia for the software support.

REFERENCES

- [1] Andersson G., Donalek P., Farmer R., Hatziaargyriou N., Kamwa, I., Kundur P., Martins N., Paserba J., Pourbeik P., Sanchez-Gasca J., Schulz R., Stankovic A., Taylor C. & Vittal V., "Causes of the 2003 Major Grid Blackouts in North America and Europe, and Recommended Means to Improve System Dynamic Performance.", Power Systems IEEE Transactions, vol 20, 2005, 1922-1928.
- [2] Force T., Abraham S., Dhaliwal H., Efford R.J, Keen L. J., McLellan A., Manley J., Vollman K., Diaz N. J. & Ridge, T., "Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations", US-Canada Power System Outage Task Force. 2004.
- [3] Larsson S. & Danell A., 'The Black-out in Southern Sweden and Eastern Denmark', IEEE PES Power Systems Conference and Exposition, 2006, pp. 309- 313.
- [4] Berizzi A., "The Italian 2003 Blackout", IEEE Power Engineering Society General Meeting, 2004, pp. 1673-1679.
- [5] Corsi S. & Sabelli C., " General Blackout in Italy Sunday September 28", Power Engineering Society General Meeting (IEEE), pp. 1691-1702.
- [6] Glavic M., Novosel D., Heredia E., Kosterev D., Salazar A., Habibi Ashrafi, A. & Donnelly M., "See it fast to keep calm: Real-Time Voltage Control under Stressed Conditions", IEEE Power Energy Mag 10(4), 2012, pp. 43-55.
- [7] Mozina C., "Undervoltage Load Shedding", Power systems Conference on Advanced Metering, Protection, Control, Communication, and Distributed Resources, 2007, pp. 39-54.
- [8] Illinois Center for a Smarter Electric Grid (ICSEG). <http://ieseg.iti.illinois.edu/ieee-57-bus-system/>.
- [9] R. Verayiah, and I.Z. Abidin, "A Study on static voltage collapse proximity indicators ", in Proceedings of IEEE 2nd International Power and Energy Conference, pp. 531-536, 2008.
- [10] Abhyankar A.R., Soman S.A., Khaparde S.A., "Optimization approach to real power tracing: an application to transmission fixed cost allocation", IEEE Trans. Power Syst., 2006, 21, pp. 1350–1361.
- [11] A. Mohamed, G.B. Jasmon, S. Yusoff, "A Static Voltage Collapse Indicator using Line Stability Factors," Journal of Industrial Technology, Vol. 7, N1, pp. 73-85, 1989.
- [12] P. Civicioglu, "Backtracking search optimization algorithm for numerical optimization problems", Appl Math Comput, 219 (15) (Apr. 2013), pp. 8121–8144.
- [13] Amraee T, Ranjbar AM, Feuillet R., "Adaptive under voltage load shedding scheme using model predictive control", Electr Power System Res, 2011, vol 81, pp. 507-13.
- [14] R.Verayiah, A. Mohamed, H.Shareef, "Modified Novel Line Stability Factor Index with Reactive Power Tracing for Identification of Vulnerable Buses in Power System", Applied Mechanics and Materials, Vol. 785 (2015), pp.398-402.
- [15] I. Musirin and T. K. A. Rahman, "Novel fast voltage stability index (FVSI) for voltage stability analysis in power transmission system," Proceedings Student Conference on Research and Development, Shah Alam, Malaysia, 2002, pp. 265-268.
- [16] Hamid, Z., Musirin, I., Othman, M.M., Rahim, M.N.A., "Bus priority ranking via stability index tracing and Evolutionary Programming", (2012) Journal of Theoretical and Applied Information Technology, vol 36 (1), pp. 48-59.
- [17] Hamid, Z.A., Musirin, I., "Optimal Fuzzy Inference System incorporated with stability index tracing: An application for effective load shedding," (2014) Expert Systems with Applications, 41 (4 PART 1), pp. 1095-1103.
- [18] Bialek J, "Tracing the flow of Electricity. Generation, Transmission and Distribution", IEE Proceedings, 1996, vol 143, pp.313–320.