Discussions on computational time for solving the framework

-- Controlled Evolution-Based Day-Ahead Robust Dispatch Considering Frequency Security with Frequency Regulation Loads and Curtailable Loads

The proposed dispatch algorithm is shown in Fig. 2, which mainly consists of the day-ahead decision layer and the evolution layer.

1. Computational time analysis for the day-ahead decision layer

The optimization model in the day-ahead decision layer is detailed in equation (35). The key factor threating the model's computational time is the non-convex maximum frequency deviation constraint (20):

$$\frac{(\Delta P \ - \sum_{b\in\Omega_{FQR}} p_{b,t})}{B^{base}} \leq (1 - \frac{f_{min}}{f_0}) g(H, R, F_H, T_R)$$

In the existing scheduling methods, there are two main ideas to handle the constraint. The one is to ignore some key parameters of the non-convex function $g(H, R, F_H, T_R)$ and add it into the model, but it is difficult to ensure frequency safety. Another idea is to introduce binary auxiliary variables to perform piecewise linear approximation on the $g(H, R, F_H, T_R)$. This results in the addition of a large number of both additional binary auxiliary variables and hyperplanes in the day-ahead optimization model, significantly increasing the computational time.

In this paper, the day-ahead decision model is shown in (35). Without ignoring the key parameters of the function $g(H, R, F_H, T_R)$, the maximum frequency deviation constraint (20) is transformed by convex relaxation, and the constraint (20) is transformed as the form of (35c), as shown in Section III. In the case studies of the modified IEEE-14 bus system, this method generates 4 hyperplanes L_h $(h = 1, \dots, 4)$, no additional binary auxiliary variables are required, and the frequency security can be well guaranteed.

Therefore, our day-ahead optimization model (35) avoids introducing a large number of both binary auxiliary variables and hyperplanes for considering the maximum frequency deviation constraint, **significantly reducing the complexity of the day-ahead decision model**. The case studies show well effect of the proposed method in ensuring frequency security.

2. Computational time analysis for the evolution layer

The optimization model of the evolution layer is shown in equation (36). It is a linear programming model and is easy to solve. For the evolution of adverse scenarios $\tilde{p}_{r,i} \in \Omega_{\mathrm{W}}$, parallel computing can be used to solve the model for each $\tilde{p}_{r,i}$, significantly reducing the computing time.

3. Computational time for the case of the modified IEEE 14-bus system

For the improved IEEE14-bus system with a high proportion of renewable energy shown in Section VI, this case compute ends after 2 iterations according to the algorithm shown in Fig. 2.

The algorithm is coded in MATLAB with YALMIP interface and solved by GUROBI 12.0 solver. Programming environment is Core i7-12700H @ 2.70GHz with RAM 16 GB.

In the first iteration, the day-ahead decision layer takes 8.36 seconds, and the intraday evolution layer takes 11.64 seconds; in the second iteration, the day-ahead decision layer takes 10.67 seconds, and the intraday evolution layer takes 10.28 seconds. The example shows that the proposed method has a short computation time and can be applied to the day-ahead stage.