"Application of Red Fox Algorithm in Unit Commitment"

Minor Project Report

Submitted in Partial Fulfillment of the Requirements for the Degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRICAL ENGINEERING

By

Priyanshu Sharma: 20BEE094

Vinamra Gupta: 20BEE129

Department of Electrical Engineering
Institute of Technology
NIRMA UNIVERSITY
Ahmedabad 382 481

December 2023

CERTIFICATE

This is to certify that the Minor Project Report entitled "____" submitted by Mr./Ms. ____ (Roll No. ______________) towards the partial fulfillment of the requirements for the award of degree in Bachelor of Technology in the field of Electrical Engineering of Nirma University is the record of work carried out by him/her under our supervision and guidance. The work submitted has in our opinion reached a level required for being accepted for examination. The results embodied in this minor project work to the best of our knowledge have not been submitted to any other University or Institution for the award of any degree or diploma.

Date:

Institute – Guide

Name of Guide
Department of Electrical Engineering
Institute of Technology
Nirma University
Ahmedabad

Head of Department
Department of Electrical Engineering
Institute of Technology
Nirma University
Ahmedabad

Director
Institute of Technology
Nirma University
Ahmedabad

ACKNOWLEDGEMENT

I must acknowledge the strength, energy and patience that almighty **GOD** bestowed upon me to start & accomplish this work with the support of all concerned, a few of them I am trying to name hereunder.

I would like to express my deep sense of gratitude to *all faculties of Electrical Engineering Department* for their valuable guidance and motivation.

I would like to express my sincere respect and profound gratitude to **Prof. (Dr.) S. C. Vora**, *Professor & Head of Electrical Engineering Department* for supporting and providing the opportunity for the summer internship.

I would also like to thank all my friends who have helped me directly or indirectly for the completion of my summer internship.

No words are adequate to express my indebtedness to my parents and for their blessings and good wishes. To them I bow in the deepest reverence.

- PRIYANSHU SHARMA (20BEE094)
- VINAMRA GUPTA (20BEE129)

Abstract

Unit commitment problem is a critical task in power system operation, which aims to determine the on/off status of generating units to meet the forecasted load while minimizing the operation the operating cost. Traditional optimization algorithms for unit commitment, such as genetic algorithms and particle swarm optimization, have been shown to be effective in finding high-quality solutions. However, these algorithms can be computationally expensive and may not be able to escape local optima.

The red fox in the snow (FOX) algorithm is a novel metaheuristic optimization algorithm inspired by the hunting behaviour of red foxes. The FOX algorithm has been shown to be effective in solving a variety of optimization problems, including the unit commitment problem.

In this paper, the application of the Redfox Algorithm (FOX) is presented to solve the unit commitment problem.

Keywords

FOX Optimizer, Unit commitment, Redfox Algorithm, Optimizing power system operation, Generation scheduling, Mixed integer linear programming.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

UCP	: Unit Commitment
SR	: Spinning Reserve
MILP	: Mixed Integer Linear Programming
GA	: Genetic Algorithm
PSO	: Particle Swarm Optimization
MW	: Mega Watts
SUH	: Start Up Hot Cost
SUC	: Start Up Cold Cost
EPL	: Extended Priority List
LR	: Lagrange Relaxation
SA	: Simulated Annealing
EP	: Evolutionary Programming
DE	: Differential Evolution
BFA	: Bacterial Foraging Algorithm
ICA	: Imperialist Competitive Algorithm
HSA	: Harmony Search Algorithm
BRABC	: Biogeography-Based Optimization with Roulette Wheel Selection
LRGA	: Lagrange Relaxation Genetic Algorithm
EGA	: Enhanced Genetic Algorithm
MPSO-EO	: Modified particle swarm optimization (MPSO) along with equilibrium optimizer (EO)

LIST OF NOMENCLATURE

P_{gi}	: real power output of generator *i*
P_D	: power demand
P_{gimin}	: minimum power output of generated by unit *i*
P_{gimax}	: maximum power output of generated by unit *i*
U_i	: ON/OFF status
ST_i	: startup cost of generator *i*
HSC_i	: the unit's hot startup cost
CSC_i	: the unit's cold startup cost
DT_i	: the unit's down time
MDT_i	: the unit's minimum down time
CSH_i	: the cold start hour
$Dist\_S\_T_{it}$	: distance sound travels
Sp_S	: speed of sound in the air
$Time\_S\_T_{it}$	: sound travels time
$Dist\_Fox\_Prey_{it}$	: distance of the fox from the prey
$Jump_{it}$	: jump height

TABLE OF CONTENTS

1. Introduction to Unit Commitment

The unit commitment problem (UCP) is a complex optimization problem in electrical power system operation. It involves deciding which generating units to turn on and off, and at what power output, to meet the forecasted demand for electricity at each time interval, while minimizing the total operating cost. The UCP is classified as a short-term problem, as it is usually considered for a period of 24 hours.

The complexity of the Unit Commitment Problem (UCP) is heightened due to its incorporation of both binary and continuous variables. Binary variables delineate the operational status (on/off) of power-generating units, whereas continuous variables signify the power output of these units. Moreover, the UCP necessitates the consideration of an extensive array of constraints, encompassing unit-specific limitations (e.g., minimum up and down times for power-generating units) and system-wide constraints (e.g., transmission line capacities) [1].

At its core, the unit commitment problem seeks to strike a balance between two key objectives: ensuring power supply reliability and minimizing operating costs. Power supply reliability involves meeting the forecasted demand and maintaining system frequency within acceptable limits, while minimizing operating costs entails reducing expenses associated with fuel consumption, start-up and shutdown costs, maintenance, and emissions. These conflicting objectives require sophisticated optimization techniques to find the optimal commitment schedule that satisfies both requirements. [2]

It involves the optimal scheduling of power generation units within an electricity grid over a specific time horizon to meet forecasted electricity demand while minimizing operational costs. This section outlines the mathematical formulation of the Unit Commitment problem and the key variables, constraints, and objectives that characterize this complex optimization challenge.

1.1 NEED FOR UNIT COMMITMENT

Unit commitment is a critical process for the efficient and reliable operation of power systems. It involves making decisions about which generating units should be online and at what power levels, to meet forecasted electricity demand while minimizing operational costs.[2] [3]

There are several reasons why unit commitment is so important:

1. **To ensure adequate supply**: By committing enough units to supply the load, the power system can prevent potential shortages and ensure grid reliability even during peak demand periods.
2. **To reduce costs**: Unit commitment can help to minimize operational costs by making strategic decisions about which units to operate. By optimizing these choices, the system can reduce fuel consumption, start-up and shut-down costs, and overall operational expenses.

3. **To maximize efficiency**: By committing the most economical units, the power system can operate closer to the optimal efficiency points of those units. This results in reduced fuel consumption for a given power output, contributing to cost savings and environmental benefits.

1.2 VARIABLES

1. **Binary Variables:** For each time period and each generation unit, a binary variable is used to represent the status of the unit. If x is equal to 1, the unit is online (generating power); if is equal to 0, the unit is offline.
2. **Continuous Variables:** For each time period and each generation unit, a continuous variable is used to represent the power output of the unit when it is online. This variable is subject to the operational constraints of the unit.

1.3 CONSTRAINT

1. **Demand Constraint**: The sum of the power output of all online units each time must meet the forecasted electricity demand.[2][7][8]

   $\sum_{i=0}^{n} P_{g,i} = P_D$, $i = 0,1, 2, \ldots\ldots n,$ where n = no of generator unit
2. **Ramp Rate Constraint**: The change in power output from one time period to the next for each unit must not exceed the unit's ramp rate, which ensures the stability of the power system [2][7].

$$PT_i(t) - PT_i(t-1) \leq UR_i$$
$$PTi(t-1) - PTi(t) \leq DR_i$$

   where, UR_i and DR_i are the ramp up and ramp down rates of unit *i* respectively.

3. **Minimum Up and Down Time Constraints**: These constraints ensure that a unit remains online for a minimum number of consecutive time periods after starting up and offline for a minimum number of consecutive time periods after shutting down [2][7][8].

$$U_i^t = \begin{cases} 0 \rightarrow 1, \text{ if } T_{i,off}^{t-1} \geq T_{i,down} \\ 1 \rightarrow 0, if\ T_{i,on}^{t-1} \geq T_{i,up} \\ 0\ or\ 1,\ otherwise \end{cases}$$

   where $T_{i,up}$, is the minimum-up time of ith unit in hours, $T_{i,on}^{t-1}$ is the continuously ON time of ith unit till hour (t-1) in hours.
4. **Power Output Constraints:** The power output of each unit must lie within its specified minimum and maximum limits, which are often determined by its technical characteristics. [2][7]

$$P_{gimin} \leq P_{gi} \leq P_{gimax}$$

5. **Spinning Constraint:** The concept of spinning reserve involves the power capacity available from all synchronized units in the system, minus the current loads and losses. [2][8] The spinning reserve requirement (SR) is denoted as:

$$\sum_{i=0}^{N}(Pmaxi - Pgi)Ui \geq \mathrm{SR}$$

6. **Fuel Constraints:** The fuel constraints account for the finite availability of fuel resources or the predetermined amount of fuel to be burned for power generation. [2]

7. **Start Up Hot/Cold Constraint:** A plant that has been recently shut down tends to start up faster and more efficiently compared to one that has been cooled down. This disparity in cost is incorporated into the cost function, assuming a step-function model. If a plant is turned off within a specific timeframe, it incurs only the cold start cost; otherwise, it involves the hot start cost, as outlined below: [1][2][7]

$$\mathrm{ST}_i = \mathrm{HSC}_{i,}\ \ if\ \ DT_i < MDT_i + CSH_i$$

Or

$$\mathrm{ST}_i = \mathrm{CSC}_{i,}\ \ if\ \ DT_i > MDT_i + CSH_i$$

1.4 DIFFERENT OPTIMIZATION METHOD FOR UC PROBLEM

The electric power sector has employed various strategies to address unit commitment complexities over the years. Given the problem's intricacy, diverse solution approaches are imperative. Efficiency gains in computation time and fuel expenses distinguish certain methods. In restructured markets, slight alterations in total costs can significantly impact annual fuel expenses. A review of the unit commitment literature highlights the development of multiple methodologies aimed at resolving this issue. They include:

1.4.1 DYNAMIC PROGRAMMING

The Dynamic Programming method is a way to solve problems by moving step by step to find an answer. It's an old but widely used method for the Unit Commitment problem because it can adapt to different-sized problems and match specific utility needs [7]. However, as the problem gets bigger, it becomes much harder to solve, needing a lot more computing power. This can make finding solutions difficult. Also, it's not very good at handling certain constraints related to how quickly plants can start up or how long they need to stay running.

1.4.2 MIXED INTEGER LINEAR PROGRAMMING

The utilization of the Mixed Integer Linear Programming (MILP) method is widespread in thermal unit commitment tasks. It employs binary variables (0 or 1) to indicate unit start-up, shutdown, and operational states. This technique transforms the quadratic production cost into a linear form and models start-up expenses as a step-like function. Despite guaranteeing an optimal solution within a finite series of actions, MILP faces difficulties with escalating unit numbers, resulting in heightened memory demands and notable computational delays [7].

1.4.3 LAGRANGE RELAXATION METHOD

This technique employs Lagrange multipliers to loosen the constraints. The method formulates unit commitment as a cost function, incorporating both single unit considerations and coupling constraints. By incorporating Lagrange multipliers into the coupling constraints and cost, a quicker solution is achieved. However, as the number of units increases, issues concerning solution feasibility and quality arise. Its application in production UC programs is relatively more recent compared to dynamic programming. Nevertheless, a drawback is observed: unit commitments derived from an LR dual solution, even one that is nearly optimal, often demonstrate instances of over-commitment.

1.4.4 GENETIC ALGORITHM

The initial component comprises a population matrix delineating generator quantities and scheduling times, with generators designated as 0 or 1. The most proficient members transmit information to the next generation. Through a random selection of crossover points, genetic exchange occurs between parents, generating two new members for subsequent generations. Exceptional performers receive substantial rewards. The advantages of employing a genetic algorithm (GA) encompass its robust optimization methodology, straightforward implementation, and ability to generate multiple UC schedules. GAs function by applying the principle of "survival of the fittest" within a population of potential solutions, iteratively refining approximations towards a solution. Additionally, GAs have found application in solving the Unit Commitment Problem (UCP) in hydro-thermal power systems.

1.4.5 PARTICLE SWARM OPTIMIZATION

The Particle Swarm Optimization (PSO) is a population-based algorithm mimicking social behaviors like birds flocking. It begins with a group of potential solutions called particles that move through a search space at certain speeds. These particles are influenced by the information they gather while exploring. PSO has few adjustable settings and a simple evolutionary process, yet it can offer high-quality solutions for complex power system problems. For instance, in the unit commitment of thermal units in power systems, PSO aims to minimize total operating costs by finding the best combinations of units that meet constraints and achieve the lowest cost.

2. Evolution of Red Fox Algorithm

Due to the challenges of dimensionality and computational limitations in prior approaches, an innovative solution for addressing unit commitment issues has emerged [2]. Known as the Red Fox Optimization, this method was pioneered by Hardi M. Mohammed and Tarik A. Rashid. Drawing inspiration from the hunting behaviors of foxes in their natural environment, this algorithm amalgamates the searching and hunting techniques employed by the red fox [5].

2.1 Fox life and Behaviour in snow

Foxes thrive in environments with limited species diversity and low productivity. They come in various colors, with white and red being the most prevalent. Among the common types are the arctic fox and red fox. The red fox is highly widespread and has successfully made homes in urban regions across Australia, the USA, Canada, Europe, and Japan. [5]. Foxes adapt to snowy environments by using their thick fur as insulation against the cold. They have keen senses, especially hearing, to locate prey like rodents under the snow. Their paws act like built-in snowshoes, allowing them to walk on snow without sinking. Foxes often hunt in winter, taking advantage of the cover provided by the snow. Their behavior involves pouncing into the snow to catch prey or scavenging for food buried beneath. Additionally, foxes may change color to white during winter, a process called "winter morph," helping them blend in with the snowy landscape for better camouflage and hunting success. The red fox exemplifies a species that consumes both mammals and plants, residing for an average span of 2 to 5 years while weighing anywhere from 5 to 25 pounds.

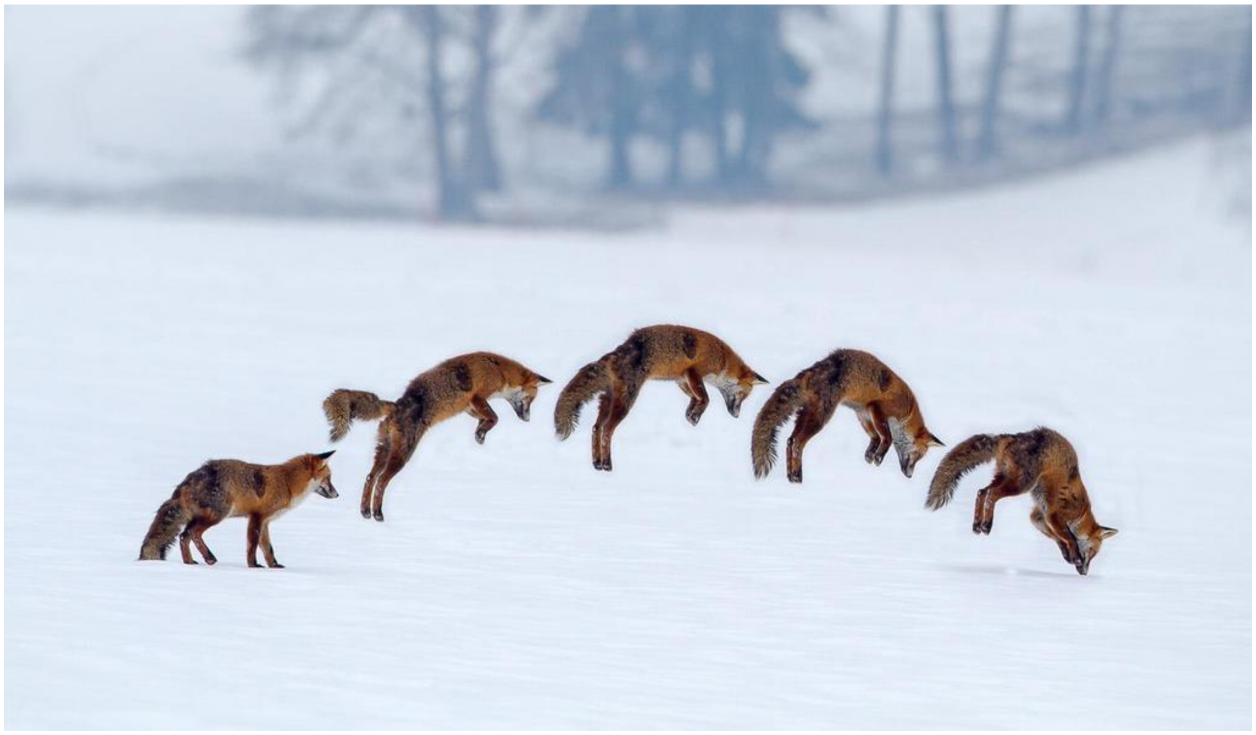

Figure 1 Red fox hunting behaviour

The FOX algorithm emulates a red fox's hunting behaviour in snowy conditions. Key steps are:

1. Snow obscures prey visibility; the fox searches randomly.
2. Using ultrasound, the fox detects prey but takes time to approach.
3. By listening and gauging time differences, the fox estimates distance.
4. Afterward, it calculates the jump needed to catch prey.

5. Random walking optimizes time and position.

Initially, the fox explores randomly to find prey by listening for ultrasound signals. This exploratory behaviour inspires the FOX algorithm. Upon hearing prey, the fox enters an exploitation phase. Calculating the distance travelled by sound, it decides to jump towards the prey. Studies suggest the fox prefers a northeast jump for an 82% success rate, but a jump in the opposite direction yields an 18% chance of catching the prey [5].

2.2 Fox Optimizer Algorithm

Initially, FOX establishes the population known as the X matrix, where each X denotes a specific position occupied by red foxes. The fitness of every search agent (each row within the X matrix) is evaluated using standard benchmark functions in each iteration. The process involves comparing the fitness value of each agent to others, consistently monitoring the best fitness and its corresponding position across iterations to identify the BestFitness and BestX.

To maintain a harmony between the exploration and exploitation phases, a condition integrating a random variable is implemented. This variable aims to equitably distribute iterations between both phases, approximately allocating 50% for exploration and 50% for exploitation. This equilibrium assists in preventing the algorithm from becoming trapped in local optima.

The algorithm's effectiveness lies in adjusting the search performance based on BestX. A variable 'a' decreases after each iteration, aiding the search agents in tracking prey more effectively. Additionally, the fitness value influences agents to avoid stagnation; if the position remains unchanged, the exploration phase is curtailed to activate other phases. The subsequent sections elaborate on exploitation and exploration strategies.

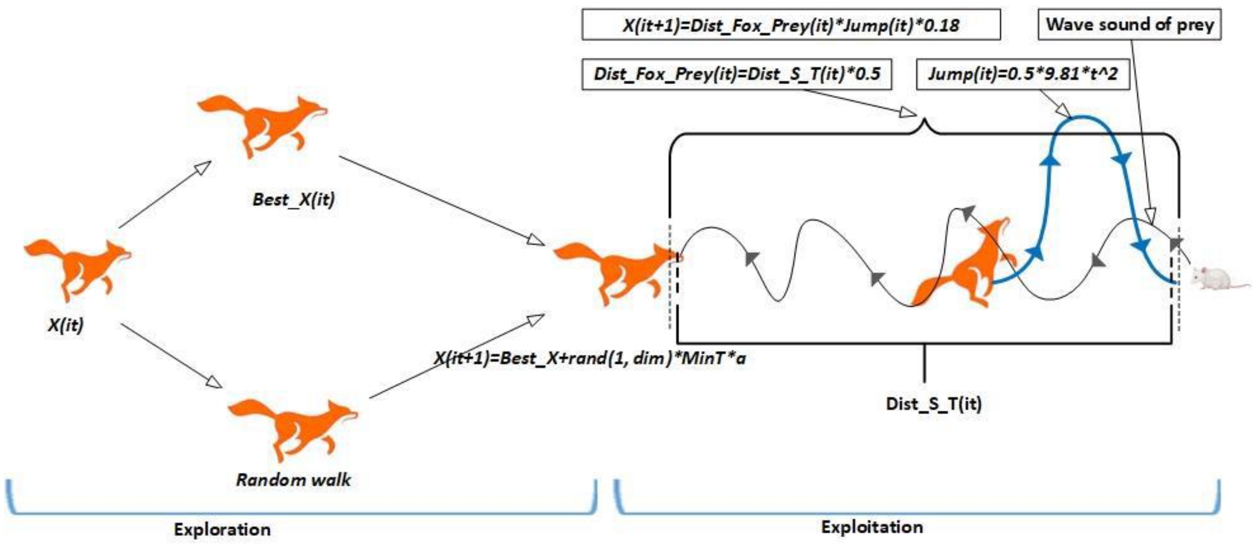
X(it+1)=Dist_Fox_Prey(it)*Jump(it)*0.18
Wave sound of prey
Dist_Fox_Prey(it)=Dist_S_T(it)*0.5
Jump(it)=0.5*9.81*t^2
Best_X(it)
X(it)
X(it+1)=Best_X+rand(1, dim)*MinT*a
Dist_S_T(it)
Random walk
Exploration
Exploitation

Figure 2 Red Fox Behavior for Hunting: Exploration and Exploitation

2.2.1 Exploitation

The exploitation phase in the red fox algorithm involves a decision criterion hinged upon the probability of capturing prey, as detailed in Section 2.2. The random variable 'p' ranges between 0 and 1. If 'p' exceeds 0.18, it prompts the red fox to seek a new position. This quest necessitates calculating the distance sound travels ('Dist_ST'), equation (1), the fox's distance from the prey ('Dist_Fox_Prey'), and a jumping parameter ('Jump_{it}'). At the start, the sound travel time ('Time_ST') is determined using a random number between 0 and 1. This time calculation is based on the speed of sound (denoted as 'Sp_S' and equals 343 m/s in air). 'Time_ST' is a randomly generated value within the range [0, 1]. Another equation (equation 2) calculates 'Sp_S' considering the time taken for sound to travel between the fox and its prey. This equation employs the currently identified best position and divides the sound travel time between the fox and its prey to determine 'Sp_S'.

Subsequently, the distance sound travels, calculated by equation (1), aids in determining 'Dist_Fox_Prey,', equation (3), halving 'Dist_ST.' To find the object's distance from the sensor in physics, the sound's distance is divided by 2 since the object's distance is half that of the sound wave.
Regarding the red fox's quest to catch the prey, it computes the jumping height ('Jump_{it}'). The equation for 'Jump_{it}' is equation (4).

In this code snippet, '9.81' represents the gravitational acceleration, while 't' denotes the squared average time taken for sound to travel during the upward and downward phases of a jump. The 'tt' variable is derived by dividing the total 'Time_ST' by the dimensions. The average of 'tt' is obtained by dividing it by 2, which is then equated to 't'. Both the gravitational force and the average time are halved to account for the two separate time periods in the jump - the ascent and descent phases.

Two equations, (5) and (6), determine the red fox's new location, each executed based on a 'p' condition. 'Equation (5)' multiplies 'Dist_Fox_Prey' and 'Jump_{it}' by 'c1' if 'p' exceeds 0.18. Conversely, if 'p' is less than or equal to 0.18, 'Equation (6)' is multiplied by 'c2' for finding the new position. 'c1' and 'c2' range from 0 to 0.18 and 0.19 to 1, respectively, signifying the northeast and opposite northeast jumps by the red fox. Hence, 'c1' facilitates exploiting a new position, directing the fox towards global optima, while 'c2' entails a lower chance of capturing prey.

2.2.2 Exploration

During the phase of random exploration, the fox navigates the search area based on its most successful known position. Unlike the jumping technique, this phase involves random movements aimed at exploring potential locations of prey. To ensure controlled yet random navigation toward the best position, the fox relies on two critical variables: MinT, representing the minimum time, and 'a'. Equations (7) and (8) depict the calculation of MinT and 'a', respectively. MinT is determined by computing the minimum time average, obtained by dividing the sum of $Time\_S\_Tit\ (i, :)$ by the problem's dimension.
The variable 'a' is defined by the equation $a = 2 * (it - (1 / Maxit))$. In this equation, $Maxit$ represents the maximum iterations. The calculated MinT and 'a' significantly influence the search phase, guiding the fox closer to the best possible solution. The use of $rand(1, dimension)$ ensures stochastic movement for the fox to explore the prey area randomly. Moreover, these variables play

a crucial role in maintaining a balance between exploration and exploitation phases, thereby enhancing the FOX algorithm's search capacity.
Equation (9) demonstrates how the fox explores the search space to find a new position, $X(it+1)$, by combining the best solution found so far, $BestXit$, with the random variable $rand(1, dimension)$, MinT, and 'a'. These equations, as part of the FOX algorithm, can be adjusted to existing algorithms, potentially improving their performance or serving as a foundation for creating new metaheuristic algorithms.
Both phases' equations require minimal adjustments when incorporated into existing algorithms, particularly for solving problems in multidimensional spaces using the FOX method. Instead of fundamental alterations, adaptation is key to effectively applying these equations to specific problems.

Table 1: Exploitation and Exploration Equations

Equation	**Representation**
$Dist\_S\_T_{it} = Sp\_S * Time\_S\_T_{it}$	1
$Sp\_S = BestPosition_{it} / Time\_S\_T_{it}$	2
$Dist\_Fox\_Prey_{it} = Dist\_S\_T_{it} * 0.5$	3
$Jump_{it} = 0.5 * 9.81 * t\text{^}2$	4
$X_{(it+1)} = Dist\_Fox\_Prey_{it} * Jump_{it} * c1$	5
$X_{(it+1)} = Dist\_Fox\_Prey_{it} * Jump_{it} * c2$	6
$tt = sum\ (TimeST\ _{it}\ (i, :))/dimension, \quad MinT = Min(tt)$	7
$a = 2 * (it - (1/Max_{it}))$	8
$X_{(it+1)} = BestX_{it} * rand(1, dimension) * MinT * a$	9

The flowchart and Algorithm of the FOX optimizer can be seen in the Figure 3 and Algorithm 1 respectively.

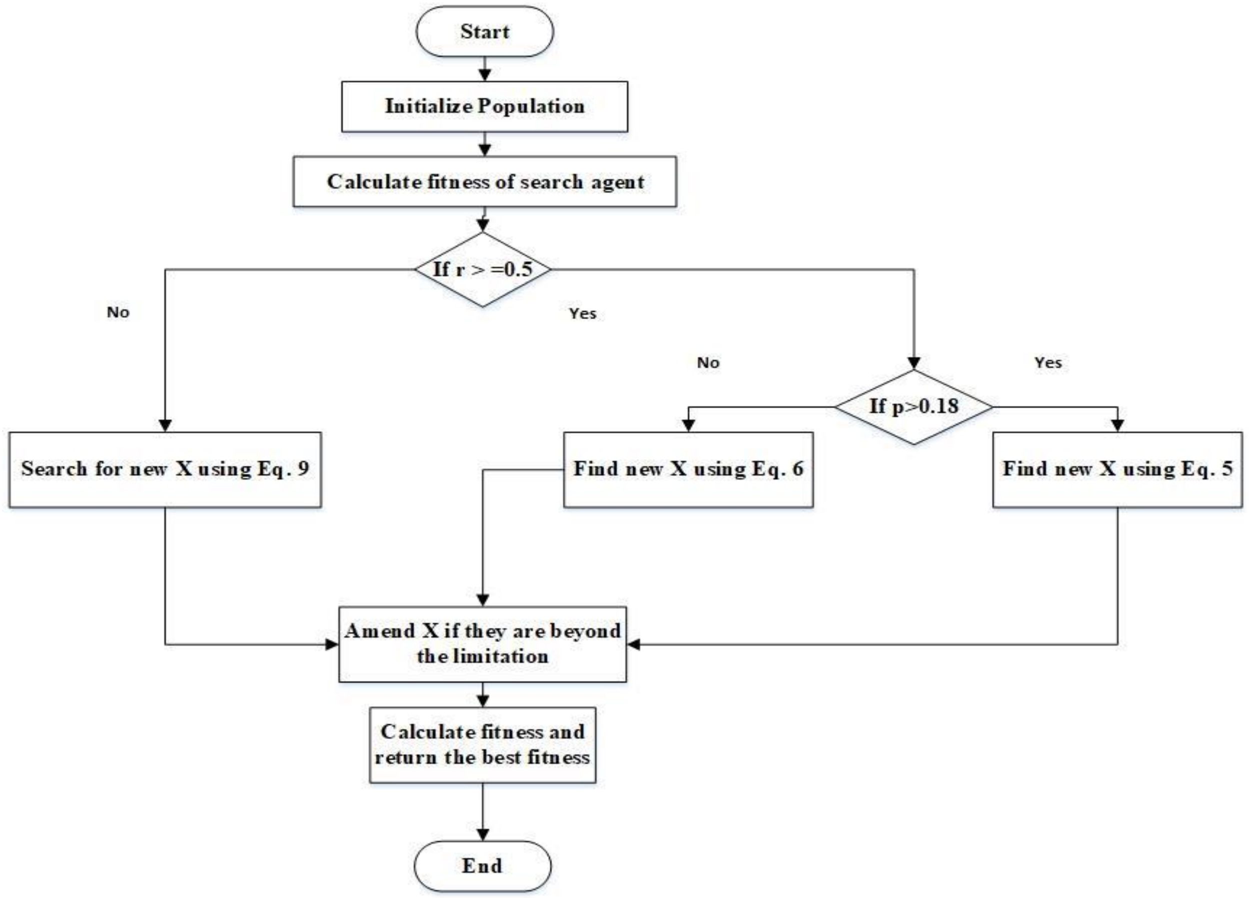
Start
Initialize Population
Calculate fitness of search agent
If r > =0.5
No
Yes
No
Yes
If p>0.18
Search for new X using Eq. 9
Find new X using Eq. 6
Find new X using Eq. 5
Amend X if they are beyond the limitation
Calculate fitness and return the best fitness
End

Figure 3: FOX Algorithm Flowchart

Algorithm 1: RED FOX Optimizer

```
1. Initialize the population of red foxes $\boldsymbol{X}$ i (i=1, 2, ….., n).
2. While it is less than Maxit:
   a. Initialize variables: $\boldsymbol{Dist\_S\_T}$, $\boldsymbol{Sp\_S}$, $\boldsymbol{Time\_S\_T}$, $\boldsymbol{BestX}$, $\boldsymbol{Dist\_Fox\_Prey}$, $\boldsymbol{Jump}$,
   $\boldsymbol{MinT}$, a, and BestFitness.
   b. Calculate the fitness of each search agent.
   c. Select the best position (BestX) and best fitness (BestFitness) among the fox population in
   each iteration.
   d. If the current fitness is greater than the previous fitness:
      • Update BestFitness with the current fitness.
      • Update BestX with the current position (X(i, :)). e. If a random value (r) is greater than
        or equal to 0.5:
      • If a probability value (p) is greater than 0.18:
          • Initialize time randomly.
          • Calculate Distance_Sound_travels using Equation (1).
          • Calculate Sp_S from Equation (2).
          • Calculate distance from fox to prey using Equation (3).
          • Calculate average time (Tt).
          • Set T as half of Tt.
          • Calculate jump using Equation (4).
          • Calculate the next position ($\boldsymbol{X}(\boldsymbol{it}+\boldsymbol{1})$) using Equation (5).
      • Else, if p is less than or equal to 0.18:
```

- Perform the same calculations as above but use Equation (6) to find $\boldsymbol{X(it+1)}$. f. If the random value (r) is less than 0.5:
- Calculate MinT using Equation (7).
- Explore $\boldsymbol{X(it+1)}$ using Equation (9). g. Check and adjust the position if it goes beyond the limits. h. Evaluate search agents based on their fitness. i. Update BestX. j. Increment the iteration count (it) by 1.

3. End of the while loop.
4. Return BestX and BestFitness.

The FOX algorithm begins by initializing the population of red foxes randomly. In the initial iteration, it verifies whether each fox's position falls within the boundary defined by the benchmark function. The fitness values of the foxes are computed based on this, and the BestFitness and BestX values are selected from these evaluations. The exploitation phase commences if a randomly generated number, 'r,' is greater than or equal to 0.5. During this phase, a condition involving 'p' decides whether the red fox adopts a new position using specific equations. If 'r' is less than 0.5, activating the else condition, the exploration phase triggers, determining a new position based on the best position, a random number, and variables MinT and 'a'. Initially, the algorithm returns the BestFitness and then repeats these steps iteratively to identify the optimal fitness and position across subsequent iterations.
Regarding computational complexity, FOX demonstrates a time complexity of O(SearchAgents * D * it) per iteration, where SearchAgents represents the population size, D indicates the dimensionality of the problem, and 'it' stands for the number of iterations. Consequently, FOX can be characterized by an O(n^2) time complexity. The space complexity of FOX is influenced by the vectors and matrices within Algorithm 1, resulting in an O(n^2) space complexity for each iteration [5].

3. Problem Formulation

The red fox algorithm described earlier is employed to solve the unit commitment problem involving 10 generator units, considering different parameters, costs, and constraints associated with power generation units. Simulations are conducted using test systems that have been modified based on the references provided [16]. Detailed data can be found in Table 2. It is assumed that the spinning reserve amounts to 10% of the load demand [1]. Additionally, the fuel cost function for each generator is approximated in quadratic form [1]. Specific load demands are outlined in Table 3.

Table 2

UNIT DATA FOR THE 10- UNIT SYSTEM

Generator	Pmax (MW)	Pmin (MW)	a $/h	b $/MWh	c $/MWh^2	Start-up HOT cost ($)	Start-up COLD cost ($)	Cooling time unit	Initial ON unit	Initial OFF unit	Min. Down Time (MIN OFF)	Min. UP Time (MIN ON)	Initial condition
1	455	150	1000	16.19	0.00048	4500	9000	5	8	0	8	8	1

2	455	150	970	17.26	0.00031	5000	10000	5	8	0	8	8	1
3	130	20	700	16.6	0.002	550	1100	4	0	-5	5	5	0
4	130	20	680	16.5	0.00211	560	1120	4	0	-5	5	5	0
5	162	25	450	19.7	0.00398	900	1800	4	0	-6	6	6	0
6	80	20	370	22.26	0.00712	170	340	2	0	-3	3	3	0
7	85	25	480	27.74	0.00079	260	520	2	0	-3	3	3	0
8	55	10	660	25.92	0.00413	30	60	0.5	0	-1	1	1	0
9	55	10	665	27.27	0.00222	30	60	0.5	0	-1	1	1	0
10	55	10	670	27.79	0.00173	30	60	0.5	0	-1	1	1	0

Experiments are conducted employing test systems modified from [16].

Table 3

LOAD DEMAND FOR 24-HOUR

Hour	Load (MW)	Hour	Load (MW)	Hour	Load (MW)	Hour	Load (MW)
1	700	7	1150	13	1400	19	1200
2	750	8	1200	14	1300	20	1400
3	850	9	1300	15	1200	21	1300
4	950	10	1400	16	1050	22	1100
5	1000	11	1450	17	1000	23	900
6	1100	12	1500	18	1100	24	800

It starts by defining input parameters such as maximal and minimal power outputs (Pmax and Pmin), associated costs (a, b, c, SUH, SUC), initial conditions (InitialTON, InitialTOFF), cooling time (Tcold), and demands (DEMAND). It proceeds to initialize and organize data structures, then runs the FOX algorithm to solve the UC problem iteratively.

The core implementation iterates through SA (Search Agents) red foxes, evaluating their fitness using specific functions. The algorithm explores and exploits potential solutions, updating the red fox positions and calculating fitness at each iteration. The best solution and its fitness across all iterations are tracked and updated.

The mathematical expression representing the problem seeking minimization [8] is:

$$TC == \sum_{t=1}^{T} \sum_{i=1}^{N} \left[F_i\left(P_i^t\right) + SU_{i,t}(1 - U_i^{t-1}\right] * U_i^t$$

This code optimizes the generation schedule for power units, considering various constraints and costs, and aims to minimize total operational costs while fulfilling demand. The iterative process refines the solution by updating the population and evaluating fitness iteratively until the defined maximum iteration limit is reached. Ultimately, it produces an optimized power generation schedule that minimizes costs while meeting demand constraints.

4. Result

The application of the FOX algorithm showcased both exploration and exploitation techniques while addressing the unit commitment problem involving 10 generator units over a 24-hour timeframe. We obtained the Total Cost is $ 562888.167490. Table 2 shows results display the status of ten generator units (G1 to G10) over a 24-hour period, detailing their operational states in each hour. Each generator's power output is shown, indicating whether they were turned on (producing power) or off (not producing power) during the respective hours.

Table 4 The optimal solution derived from the Fox optimizer

Generator Units										
Hours	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
01	455	245	0	0	0	0	0	0	0	0
02	455	295	0	0	0	0	0	0	0	0
03	455	385	0	0	0	0	0	0	10	0
04	455	455	0	0	40	0	0	0	0	0
05	455	455	0	0	80	0	0	0	10	0
06	455	360	130	130	25	0	0	0	0	0
07	455	410	130	130	25	0	0	0	0	0
08	455	455	130	130	30	0	0	0	0	0
09	455	455	130	130	85	20	25	0	0	0
10	455	455	130	130	162	33	25	10	0	0
11	455	455	130	130	162	73	25	10	10	0
12	455	455	130	130	162	80	25	43	10	10
13	455	455	130	130	162	33	25	0	10	0
14	455	455	130	130	100	20	0	10	0	0
15	455	455	130	130	30	0	0	0	0	0
16	455	310	130	130	25	0	0	0	0	0
17	455	240	130	130	25	0	0	10	0	10
18	455	350	130	130	25	0	0	10	0	0
19	455	455	130	130	30	0	0	0	0	0
20	455	455	130	130	162	33	25	0	0	10
21	455	455	130	130	85	20	25	0	0	0
22	455	455	0	0	145	20	25	0	0	0
23	455	420	0	0	25	0	0	0	0	0
24	455	345	0	0	0	0	0	0	0	0

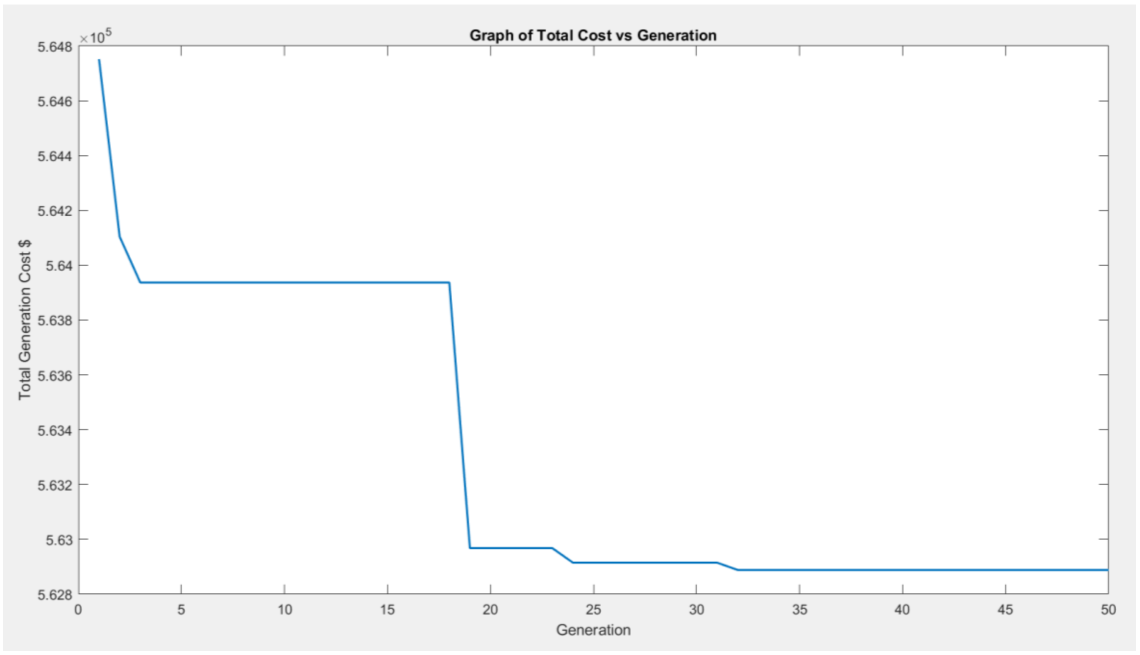

Figure 4 Graph of Total Cost vs Iteration

As result, Figure 6 illustrates a promising trend towards cost reduction across multiple iterations. The consistent decrease in total cost indicates that the optimization algorithm applied over the iterations is progressively improving, refining the solution to minimize the total cost. The cost appears to converge to around $562,888, suggesting that the optimization process has potentially reached a stable solution by the end of the 50 iterations.

Table 5: Comparing the Effectiveness of the Red Fox Technique Against Alternative Approaches in a 10-Unit System [8][15]

Method	**TOTAL PRODUCTION COSTS ($)**
EPL	563,977
LR	565,673.13
SA	565,828
EP	564,551
IBPSO	563,977
DE	563,938
BFA	564,842
ICA	563,938
HSA	565,827
BRABC	563,937.72
LRGA	564,800
GA	564,217.08
EGA	563,937.57
MPSO-EO	564,795.331
REDFOX	**562,888.16**

Among various optimization methods, the REDFOX algorithm yielded one of the lowest total production costs, standing at $562,888.16, showcasing its efficiency and competitiveness in minimizing overall expenses. We have taken data in Table 5 from reference no. 8 and 15.

5. Conclusion

In this study, the application of the Redfox Algorithm (FOX) in tackling the unit commitment problem (UCP) was explored. The traditional optimization methods like genetic algorithms and particle swarm optimization, while effective, often encounter challenges related to computational intensity and susceptibility to getting stuck in local optima. In response to these constraints, the FOX algorithm presents an innovative method for tackling optimization problems, drawing inspiration from the hunting strategies observed in red foxes. It employs a random walk strategy rooted in the exploration phase's optimal position, utilizes a distance assessment technique to gauge the fox and prey separation, and incorporates a leaping action to capture the prey during the exploitation phase.

The application of the Red Fox Algorithm proved successful in resolving the UCP for 10-unit test systems within 24-hour scheduling time horizons. The strategic integration of problem-specific operators into the basic red fox algorithm notably amplified exploration and exploitation within the search space. This amalgamation effectively mitigated the risk of entrapment in local optimal solutions, ensuring more resilient and effective outcomes.

The findings from this research highlight how the FOX algorithm effectively handles the UCP and deals with the real-time constraints of power system operations. This suggests it could be a valuable tool for making better decisions about unit commitments in power systems.

Reference

[1] P. Sriyanyong and Y. H. Song, “Unit commitment using particle swarm optimization combined with lagrange relaxation,” *2005 IEEE Power Eng. Soc. Gen. Meet.*, vol. 3, no. 6, pp. 2752–2759, 2005, doi: 10.1109/pes.2005.1489390.

[2] D. Ananthan, “Unit Commitment Solution Using Particle Swarm Optimisation (PSO),” *IOSR J. Eng.*, vol. 4, no. 3, pp. 01–09, 2014, doi: 10.9790/3021-04310109.

[3] P. V. R. Krishna, G. P. Rao, and D. S. Sao, “Particle Swarm Optimization Technique to Solve Unit Commitment Problem,” *Int. J. Adv. Comput. Res.*, vol. 2, no. 7, pp. 100–105, 2012.

[4] Y. R. Guo and A. O. Function, “Improved PSO Approach for The Solution of Unit Commitment Problem,” no. AIIE, pp. 510–513, 2015.

[5] H. M. Mohammed, “FOX : A Fox-inspired Optimization Algorithm,” 2022.

[6] D. Połap and M. Woźniak, “Red fox optimization algorithm,” vol. 166, no. September 2020, 2021.

[7] B. Ax, “Mixed Integer Linear Programming in Unit Commitment”.

[8] P. K. Singhal and R. Naresh, “Solution of Unit Commitment Problem Using Enhanced Genetic

Algorithm," 2014.

[9] J. M. Arroyo and A. J. Conejo, "Unit Commitment Problem," IEEE Trans. Power Syst., vol. 17, no. 4, pp. 1216–1224, 2002.

[10] U. Ojha, J. Kasera, and M. T. Scholar, "IMPLEMENTATION OF GENETIC ALGORITHM FOR OPTIMAL," vol. 5, no. 11, pp. 4617–4623, 2018.

[11] J. L. Tonry and C. W. Stubbs, "Solving the unit commitment problem in large systems using hybrid PSO algorithms Solving the unit commitment problem in large systems using hybrid PSO algorithms," 2021, doi: 10.1088/1757-899X/1105/1/012007.

[12] A. Jaszcz, D. Połap, and R. Damaˇ, "Lung X-Ray Image Segmentation Using Heuristic Red Fox Optimization Algorithm," vol. 2022, 2022.

[13] T. Logenthiran and D. Srinivasan, "Particle Swarm Optimization for unit commitment problem," 2010 IEEE 11th Int. Conf. Probabilistic Methods Appl. to Power Syst. PMAPS 2010, pp. 642–647, 2010, doi: 10.1109/PMAPS.2010.5528899.

[14] O. O. Akinola, A. E. Ezugwu, J. O. Agushaka, R. A. Zitar, and L. Abualigah, Multiclass feature selection with metaheuristic optimization algorithms: a review, vol. 34, no. 22. Springer London, 2022. doi: 10.1007/s00521-022-07705-4.

[15] A. Sayed et al., "A hybrid optimization algorithm for solving of the unit commitment problem considering uncertainty of the load demand," Energies, vol. 14, no. 23, 2021, doi: 10.3390/en14238014.

[16] A.J. Wood, B.F. Wollenberg, Power Generation Operation and Control, 2nd ed., New York: John Wiley & Sons, Inc., 1996.