# *Original Research Article*

## Comparison and Optimization of Energy Efficient Algorithm for Component base Distributed Computing in 5G Networks

**Abstract:** In the present scenario, the wired network or wireless networks is an application across the world. So the wire networks used in various software industries, educational institutions, and various enterprises used such as distributed data centers. The data transmission works like a flow of electricity in a linear way. So in this process during the data transmission exhibits from one stage to another the energy consumption in carbon footprint (i.e. $CO_2$). The data transmission is two types of methods (1) Communication-Based (2) Component-Based. Here this paper concludes the compared study of component-based energy consumption using the Bellman-Ford Algorithm and Dijkstra's Algorithm. The results and finding measurement as discussed in this paper.

**Keywords:** *Energy Optimization, Bellman-Ford Algorithm, Distributed Computing Networks, Bellman-Ford Time Complexity, Energy Consumption, Dijkstra's Algorithm Time Complexity.*

## 1. Introduction

A distributed system is a collection of independent computers that appears to its users as a single connected system. Distributed computing network deals with determining how packets will be routed (flow) from source to destination. It can be of three types: Static: Routes are based on static tables that are "wired into" the network and are rarely changed. Dynamic: All packets of one application can follow different routes depending on the topology of the network, the shortest path, and the current network load. Semi-Dynamic: A route is selected from the start of each communication and then all the packets of the application follow the same route from the source.

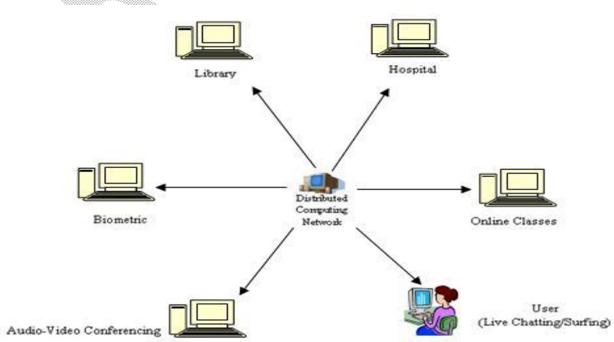



Figure 1. Distributed Computing Network

Power-aware and energy-aware systems are those where power or energy consumption is a principal design consideration. For example, power-aware systems may utilize techniques to change the system's behavior based on the amount of power being consumed. Energy-aware systems may modify the operation of the system based on the amount of energy in components.
Distributed computing depended on components based. There are various techniques to find energy. The routing algorithm is based on a single source.

Distributed computing is a much broader technology that has been around for more than three decades now. Simply stated, distributed computing is computing over distributed autonomous computers that communicate only over a network (Figure 1). Distributed computing systems are usually treated differently from parallel computing systems or shared-memory systems, where multiple computers share a common memory pool that is used for communication between the processors. Distributed memory systems use multiple computers to solve a common problem, with computation distributed among the connected computers (nodes) and using message-passing to communicate between the nodes. For example, grid computing, studied in the previous section, is a form of distributed computing where the nodes may belong to different administrative domains. Another example is the network-based storage virtualization solution described in an earlier section the use of distributed computing between data with metadata servers.

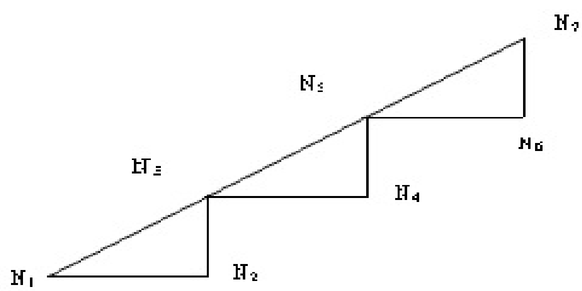


**Figure.2** Step Network Topology

## 2. Literature Review

Biswas et al. [1] found “the energy consumption of the well-known Chandy/Misra/Bryant and YAWNS synchronization algorithms. Distributed simulation algorithms require a significant amount. They described the Energy consumption of synchronization algorithms in distributed simulations”. Jafari et al. [2] proposed “the method of OPTICS density-based clustering in wireless sensor networks. It is an efficient algorithm that can create a cluster with different regions. They suggested OPTICS density-based clustering algorithm has better performance, compared to similar algorithms such as LEACH, SEP, EEHC, and DBSCAN and could increase network lifetime and the number of packets sent by reducing energy consumption”. Kumar et al. **[3]** described “Extended Lifetime with Minimum Energy Consumption (ELMEC) scheme is proposed for reducing the energy consumption of cooperative communication in WSN and to increase the network lifetime. The transmission distance, number of cooperating nodes of the Virtual Multiple-Input-Multiple-Output (V-MIMO), and the modulation order are jointly optimized by the proposed scheme to augment the network lifetime and consume minimum end-to-end energy. The total energy consumption per bit and network lifetime of the proposed scheme is compared with the traditional minimum energy consumption scheme. of additional energy compared to sequential execution. Further, different synchronization algorithms”. Tabatabaei [4] “ the Cuckoo Optimization Algorithm was used to select the shortest route between the relay groups. The results show that the process of selecting a stable route using the Cuckoo Optimization Algorithm and the TOPSIS algorithm has a valuable impact on the performance of networks, and the proposed algorithm shows better performance compared to the methods introduced in Tabatabaei and Behravesh with regard to the throughput and the end-to-end delay”.  Anand Chatterjee et al [5] “This paper proposes new routing-chains protocol for energy-efficient networks in Gaussian and uniformly distributed sensor networks. The Gaussian distributed sensor network is given the name of the protocol is Gaussian routing-chains protocol for energy-efficient networks (GREEN). We have also presented a comparative analysis between the proposed green and green with existing protocol EECF". Lin et al. [6] have proposed "a method for minimizing the energy for NP-complete problem solution by Dijkstra's algorithm and Yen's k-shortest paths algorithm. They have evaluated in Abilene network (eg. Real and synthetic traffic matrices)". Andrews et al. [7] have proposed the model for routing and scheduling for energy saving in power mode scheduling. Anbazhagan et al. [8] have described "the power management techniques for IEEE 802.16m network using power saving in heterogeneous traffic. They have proposed an algorithm combined cyclic binary exponent (CCBE) and combined truncated binary exponent (CTBE)". Lewis et al. [9] have proposed to develop a system-wide energy consumption model for servers, in hardware performance and experimental results. Bilal et al. [10] have proposed energy efficiency which depends on (i) DCN architecture (eg. Electrical optical and hybrid) (ii) network traffic management (iii) network-aware resource allocation and energy efficiency. Niewiadomska et al.[11] have described two-level control frameworks for reducing the power consumption in computer networks, (i) Local control mechanism for the network device, (ii) network-wide control framework technique for reducing power consumption. Galinina et al. [12] have given the optimization of optimal power control schemes for reducing energy in 4G networks. Fang et al. [13] have formulated a stochastic optimization problem and designed the control algorithm and evaluated the energy performance on the throughput of the data in networks. Alzamil et al.[14] have proposed a profiling system architecture used for energy consumption, in cloud computing. Bianzino et al. [15] have investigated full system-based architecture in computer networks for energy-efficient wired networks. Niewiadomska-Szynkiewicz et al. [16] have described "a Control system for reducing energy consumption in a backbone computer network". Luca, Chiaraviglio., et al.[17**]** have evaluated a "particular algorithm or procedure to offer energy saving capabilities in networks, but rather we formulate a theoretical model based on random graph theory that allows estimating the potential gains achievable by adopting sleep modes in networks where energy proportional devices are deployed".

## 3. Methodology Used

The Bellman–Ford algorithm is an algorithm that computes the shortest paths from a single source vertex to all of the other vertices in a weighted graph. It is slower as comparatively Dijkstra's algorithm for the same problem and its solution, but more versatile, as it is capable of handling graphs in which some of the edge weights are negative numbers.

Bellman-Ford algorithm works by over computation the length of the path from the starting vertex to all other vertices.

A very important application of Bellman-Ford is to check if there is a negative cycle in the graph; the Time Complexity of the Bellman-Ford algorithm is relatively high.

The time complexity of the bellman algorithm is O (V.E) in case E=V2, O (V3)

### 3.1 Bellman-Ford Algorithm

Bellman-Ford Algorithm has many types of variants but the most common one is to find the shortest paths from the source vertex to all other vertices in a graph.

The Time Complexity of the Bellman-Ford Algorithm is O ($V^2$) but with a min-priority queue, the complexity is O ((V+E) logV) with the use of Fibonacci series and the application of the algorithm in majors like Digital Mapping Services, Google Maps, Designate Servers, and IP Routing.

---

**Algorithm 1:** Bellman-Ford Algorithm

---

**Input Data:** A directed graph G∑ (V, E), Where V=v1, v2, v3……$v_n$ and E=e1, e2, e3………$e_n$ the starting vertex S, and the weight W of each edge

**Result:** the shortest path from S to all other vertices in G

```
BELLMAN-FORD (G,w, s)
 INITIALIZE-SINGLE-SOURCE (G, s)
   for i ← 1 to |V[G]| − 1
      do for each edge (u, v) ∈ E[G]
         do RELAX(u, v,w)
            for each edge (u, v) ∈ E[G]
         do if d[v] > d[u] + w(u, v)
      then return FALSE
  return TRUE
```

The Bellman-Ford algorithm defines in time *O(V E)*, since the initialization state in line 1 takes *Θ(V)* time, so each of the |*V*|−1 passes over the edges in lines 2–4 takes *Θ(E)* time, and the for loop of lines 5–7 takes revised *O(E)* time.

1. Set all vertices distances = infinity except for the source vertex, set the source distance =0
2. Push the source vertex in a min-priority queue in the form (distance, vertex), as the comparison in the min-priority queue will be according to vertices distances.
3. Pop the vertex with the minimum distance from the priority queue (at first the popped vertex = source).
4. Update the distances of the connected vertices to the popped vertex in case of "current vertex distance + edge weight < next vertex distance", then push the vertex with the new distance to the priority queue.
5. If the popped vertex is visited before, just continue without using it.
6. Apply the same algorithm again until the priority queue is empty.

### 3.2 Dijkstra's Algorithm

Dijkstra's Algorithm

The following algorithm [18] for finding single-source shortest paths in a weighted graph like directed or

Undirected with no negative-weight edges. Dijkstra's algorithm solves the single-source shortest-paths problem on a weighted, directed graph $G = (V, E)$ for the case in which all edge weights are nonnegative. Let $w(u, v) \geq 0$ for each edge $(u, v) \in E$. As we shall see, with a good implementation, the running time of Dijkstra's algorithm is lower than that of the Bellman-Ford algorithm.

Dijkstra's algorithm maintains a set *S* of vertices whose final shortest-path weights from the source *s* have already been determined. The algorithm repeatedly selects the vertex $u \in V - S$ with the minimum shortest-path estimate, adds *u* to *S*, and relaxes all edges leaving *u*. In the following implementation, we use a min-priority queue *Q* of vertices, keyed by their *d* values.

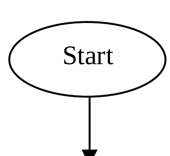
```
Dijkstra(G,w, s)
    Initialize-Single-Source (G, s)
    S ← ∅
    Q ← V[G]
    while Q _= ∅
        do u ← EXTRACT-MIN(Q)
    S ← S ∪ {u}
        for each vertex v ∈ Adj[u]
    do RELAX(u, v,w)
```

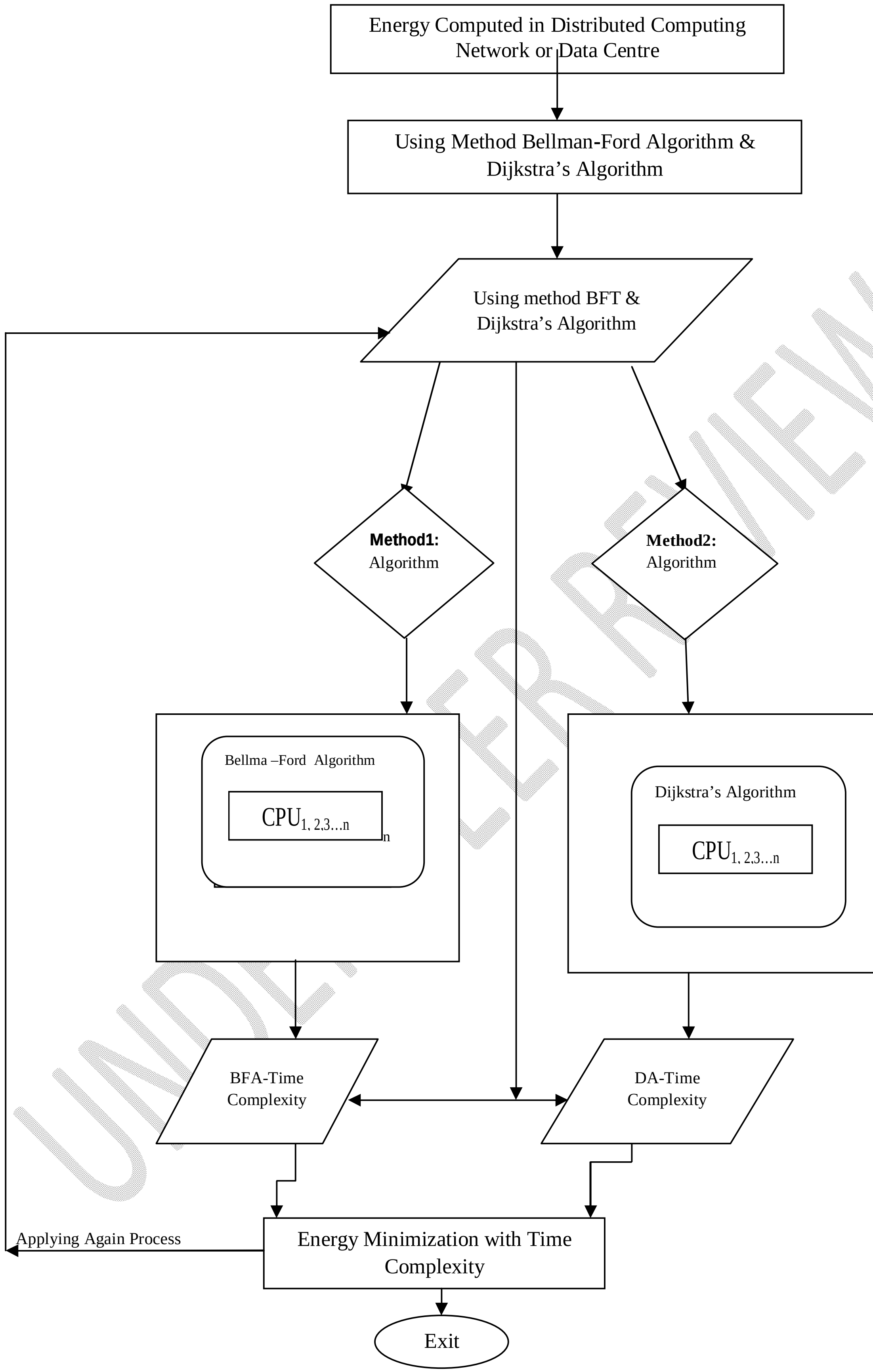
This algorithm always selects the lightest or closest vertex in $V - S$ to add to set $S$, we say that it uses a greedy strategy.

In Line 1 performs the usual initialization of $d$ and $\pi$ values, and line 2 initializes the set $S$ to the empty set. The algorithm maintains the invariant that $Q = V - S$ at the start of each iteration of the **while** loop of lines 4–8. Line 3 initializes the min-priority queue $Q$ to contain all the vertices in $V$; since $S = \emptyset$ at that time, the invariant is true after line 3. Each time through the **while** loop of lines 4–8, a vertex $u$ is extracted from $Q = V - S$ and added to set $S$, thereby maintaining the invariant. (The first time through this loop, $u = s$.) Vertex $u$, therefore, has the smallest shortest-path

Estimate of any vertex in $V - S$. Then, lines 7–8 relax each edge $(u, v)$ leaving $u$, thus updating the estimate $d[v]$ and the predecessor $\pi[v]$ if the shortest path to $v$ can be improved by going through $u$. Observe that vertices are never inserted into $Q$ after line 3 and that each vertex is extracted from $Q$ and added to $S$ exactly once so that the **while** loop of lines 4–8 iterates exactly $|V|$ times. Because Dijkstra's algorithm always chooses the "lightest" or "closest" vertex in $V - S$ to add to set $S$, we say that it uses a greedy strategy.








**Flow chart1**: Explained Flowchart of System Model Principle

**Table.1** Energy Computed By Various Components at Various Stages

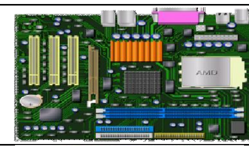
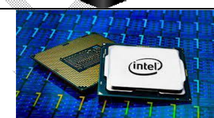
| Sr No. | Device Name | Power Cons(Min) in Watts | Power Cons(Max) in Watts | Components-wise energy consumption (in Watts) | Device Components |
|---|---|---|---|---|---|
| 1 | Mother Board (regular) | 25 | 40 | $P_{MAX}=V\times I=40\times 5=200$ watt |  |
| | High end motherboard | 45 | 80 | $P_{MAX}=V\times I=80\times 5=400$ watt | |
| 2. | DDR1 RAM(2.5 Volts) | 4 | 5.5 | $P_{MAX}=V\times I=3\times 5.5=16.5$ |  |
| | DDR2 RAM(1.8 Volts) | 3 | 4.5 | $P_{MAX}=V\times I=3\times 4.5=13.5$ | |
| | DDR3 RAM(1.5 Volts) | 2 | 3 | $P_{MAX}=V\times I=3\times 3=9$ watt | |
| 3. | Solid State Device(SSD) | 0.6 | 2.8 | 18 watt |  |
| 4. | 2.5" HDD | 0.7 | 3 | $P_{MAX}=V\times I=3\times 1.78A=5.34$ Watt |  |
| | 3.5" HDD | 6.5 | 9 | $P_{MAX}=V\times I=9\times 1.78A=16.02$Watt | |
| 5. | Intel Top End CPU (Core i7-E) | 30 | 55 | 130 to 150 W $P=CV^2f=140$Watt |  |
| 6. | 80 mm Case Fan | 0.6 | 5 W | $P=0.3\times 5=1.5$Watt 0.6 to 1.8 W |  |

**Table2.** Time Computed in Various Stages in BFT and DTC

| Time Parameter | $T_0$ | $T_{Sender}$ | $T_{Dest}$ | $T_{Worst}$ | Bellman-Ford Time Complexity | Dijkstras Time Complexity |
|---|---|---|---|---|---|---|
| **Standby Mode** | 0 | No | No | No | 20 Sec | 30 Sec |
| **Active Mode** | 0 | 0.5 | 0.7 | 1.0 | 40 Sec | 30 Sec |
| **Idle Mode** | 0 | 0.4 | 0.6 | 1.0 | 27Sec | 25Sec |

**Table3.** Energy Computed In Various Components Used Dynamic Voltage Frequency Scaling (DVFS) For Processor

| Node / Energy | N1 | | N2 | | N3 | | N4 | | N5 | | N6 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $E_{MB}$ | 25 | 40 | 25 | 40 | 25 | 40 | 26 | 40 | 27 | 40 | 25 | 40 |
| $E_{RAM}$ | 4 | 5.5 | 4 | 5.5 | 4 | 5.5 | 4 | 5.5 | 4 | 5.5 | 4 | 5.5 |
| $E_{HDD}$ | 6.5 | 9 | 5.5 | 9 | 6.5 | 9 | 6.5 | 9 | 6.5 | 9 | 6.5 | 9 |
| $E_{PROC}$ | 30 | 55 | 30 | 55 | 30 | 55 | 30 | 55 | 30 | 55 | 30 | 55 |
| $E_{FAN}$ | 0.6 | 5 | 0.6 | 5 | 0.6 | 5 | 0.6 | 5 | 0.6 | 5 | 0.6 | 5 |
| $E_{TotaL}$ | $SYS_{N1}$=118 | | $SYS_{N2}$=104 | | $SYS_{N3}$=102 | | $SYS_{N4}$=105 | | $SYS_{N5}$=182 | | $SYS_{N6}$=135 | |
| $E_{TotaL}$=746 Watt | | | | | | | | | | | | |

**Table 4.** Node wise energy computed in Idle, Standby and Active mode

| Processor/Energy | Energy$_{idle}$ | Energy$_{Active}$ | Energy$_{Standby}$ | Energy$_{Total}$ |
|---|---|---|---|---|
| **SYS$_{N1}$** | 29 | 68 | 24 | 118 |
| **SYS$_{N2}$** | 32 | 24 | 22 | 104 |
| **SYS$_{N3}$** | 30 | 87 | 29 | 102 |
| **SYS$_{N4}$** | 27 | 28 | 24 | 105 |
| **SYS$_{N5}$** | 26 | 54 | 21 | 182 |
| **SYS$_{N6}$** | 38 | 57 | 30 | 135 |
| | **E$_{TotaL}$=182** | **E$_{TotaL}$=318** | **E$_{TotaL}$=150** | **E$_{TotaL}$=746 watt** |

Here CPU has no power, RAM maintains power everything else is in standby mode, and the screen and display are turned off of the computer. In idle mode, a process is any running process in the program. The program uses computer resources but is not actively being utilized. In active mode, the client starts listening on a random node for incoming data connections from the server (i.e. client sends the file transfer protocol (FTP) command node to inform the server on which node is listening)

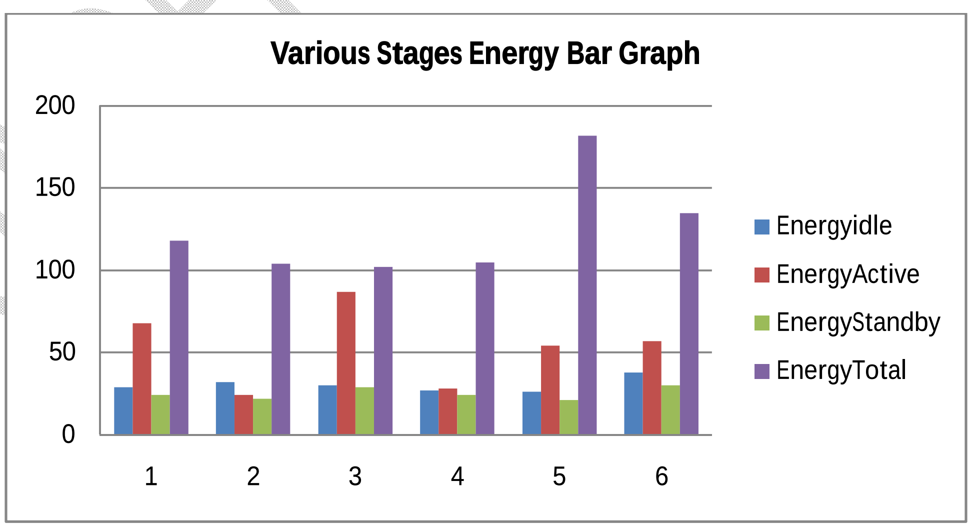
$$E=P/T \quad (1)$$

Here, E is the energy to be calculated and P is the power given in table 4 and time is calculated as given in table 5 in active mode.

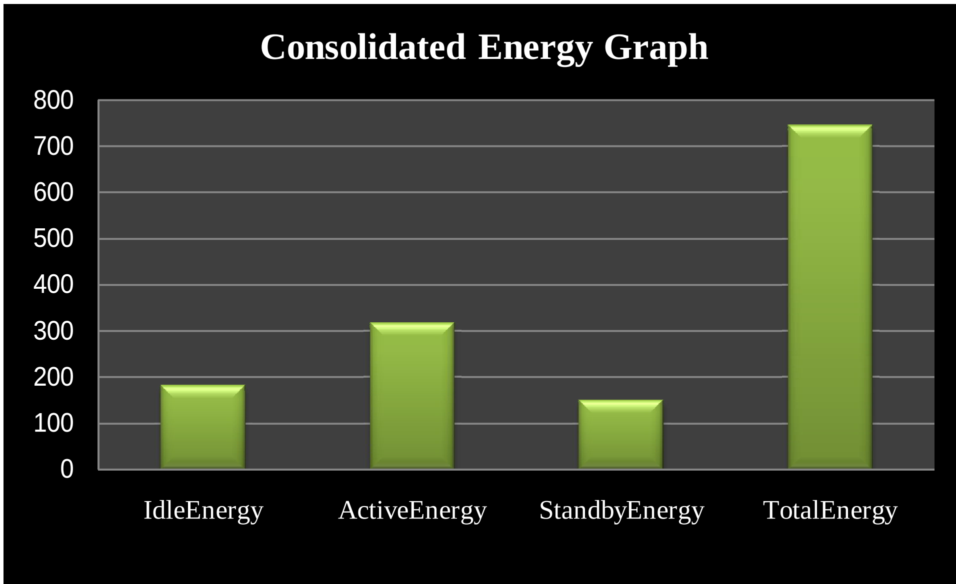
**Table5.** Energy Computed of Bellman-Ford Algorithm and Dijkstra's Algorithm for Single Source

| **Methods/Algorithm/ Energy Consumption** | **Standby Mode** | **Active Mode** | **Ideal Mode** | **Total Energy Consumption Component-Based** |
|---|---|---|---|---|
| BFA $_{\text{Time Complexity}}$ | 08Sec | 9.5Sec | 8.5Sec | 101.88 (in active Mode) |
| DA $_{\text{Time Complexity}}$ | 07Sec | 9 Sec | 8.5Sec | 96.22watt(in active Mode) |

Graph between Bellman-Ford Algorithm time complexity, Dijkstra's Algorithm time complexity and Fuzzy Logic Methods time function.




**Graph1.** Various Stages Energy Bar Graph




**Graph2.** Consolidated Energy Graph

## 4. Results & Discussion

In this paper, the results represent the minimum energy consumption used by the single source routing algorithm bellman algorithm and Dijkstra's algorithm. Here are three types of energy calculated firstly we have computed various types of energy computed in components using the formula in table1. Secondly, we have computed node-wise energy calculated in minimum and maximum power from Node1 to Node 6 belonging to various components. Now finally node wise energy is computed in various stages like idle mode, active mode, and sleep mode, and found the total energy as finally. Now as the result, we have found the time complexity in the Bellman-Ford and Dijkstra's algorithm. Here the time computed in Dijkstra's algorithm is best for a single source algorithm instead of the Bellman-ford algorithm for sending data to the destination in the networks. We take minimum time and energy to find the formula as given above in equation 1. It is compared concluded results. In the future the algorithm used in fit for best routing algorithm in any network topology. The time complexity of packets flow from sender to destination found by this paper and concluded by the result of the component base is best by to find the energy optimize and consumption is low for the sender to destination. The best way to find the shortest path to find and latency will be low in the results for the same. So the futuristic algorithm for energy optimization in components wise Dijkstra's algorithm is very useful for distributed computing and data centers.

## 5. Conclusion.

In this paper, we have concluded the implementation of various single-source optimization algorithms. Firstly, based on the architecture of the distributed network, the main reasons for the energy consumption of the network devices component are summarized. Next, the main algorithms at each level of the component are used to study the energy consumption of the devices. The performance of the bellman ford algorithm and Dijkstra's algorithm strategy is calculated and evaluated for several performances such as convergence of components, idle transmission, the standby energy consumption of network components, and sleep mode energy balance. The performance curve of the algorithm illustrates the simulation results. At the same time, the application of these algorithms in the distributed network can effectively reduce the communication energy consumption of the components devices and effectively extend the network lifetime.

## References


**[1] Biswas, Aradhya., and Fujimoto, Richard.,** *"Energy Consumption Of Synchronization Algorithms in Distributed Simulations",* Journal of Simulation, www.palgrave.com/journals, pp. 01-16, 2016.

**[2] Jafari, Hasan., Nazari, Mousa., and Shamshirband, Shahaboddin.,** *"Optimization Of Energy Consumption in Wireless Sensor Networks Using Density-Based Clustering Algorithm",* International Journal of Computers and Applications, https://doi.org/10.1080/ 1206212X.2018.1497117, ISSN: 1206-212X (Print) 1925-7074, http://www.tandfonline.com /loi/tjca20, pp. 01-11, 2018.

**[3] Kumar, K. Senthil., & Amutha, R.,** *"Lifetime Maximizations with Guaranteed Minimum End-To-End Energy Consumption Scheme for Virtual Multiple-Input-Multiple-Output In Wireless Sensor Networks",* Australian Journal of Electrical and Electronics Engineering, ISSN: 1448- 837X (Print) 2205-362X, https://www.tandfonline.com/loi/tele20, pp.01-13, 2019.

**[4] Tabatabaei, Shayesteh.,** *"A New Routing Protocol for Energy Optimization in Mobile Ad Hoc Networks Using the Cuckoo Optimization and the TOPSIS Multi-Criteria Algorithm",* Cybernetics and Systems an International Journal, DOI: 10.1080/01969722, 2021.1899597, pp. 01-22, 2021.

**[5] Ritika., Anand Chatterjee & Kumar, Vinay.,** *"Energy-Efficient Routing Protocol Via Chain Formation In Gaussian Distributed Wireless Sensor Networks",* International Journal of Electronics Letters, http://dx.doi.org/10.1080/21681724.2017.1279223, pp. 01-15, 2017.

**[6] Lin, Gongqi., Soh, Sieteng., Chin, Kwan-Wu., Lazarescu, Mihai.,** *"Efficient Heuristics For Energy-Aware Routing In Networks with Bundled Link".* Computer Networks" pp. 1774–1788, 2013.

**[7] Andrews,Matthew., Fernandez Anta, Antonio., Zhang, Lisa., et al.,** " *Routing and Scheduling for Energy and Delay Minimization in the Power down Model".* pp. 226-237, 2013.

**[8] Anbazhagan, Rajesh., and Rangaswamy, Nakkeeran.,** *"Investigations on Enhanced Power Saving Mechanism for IEEE 802.16m Network with Heterogeneous Traffic".* Journal of Network and Computer Applications, 2014.

**[9] Lewis, Adam., Ghosh, Soumik., and Tzeng, N.-F.,** *"Run-Time energy Consumption Estimation Based on Workload in Server Systems*". University of Louisiana, Lafayette, Louisiana 70504.

**[10] Bilal, Kashif., Khalid, Osman., Alvarez,Enrique., Hameed,Abdul., et, al.,** *"A Taxonomy and Survey On Green Data Center Networks".* Future Generation Computer Systems, pp. 189–208, 2014.

**[11] Niewiadomska-Szynkiewicz, Ewa., Sikora,Andrzej., Arabas, Piotr., et al.** *"Dynamic Power Management In Energy-Aware Computer Networks And Data Intensive Computing System".* Future Generation Computer Systems pp. 284–296, 2014.

**[12] Galinina, Olga., Andreev, Sergey., Turlikov, Andrey., Koucheryavy, Yevgeni.,** *" Optimizing Energy Efficiency of a Multi-Radio Mobile Device in Heterogeneous Beyond-4G Networks*". Performance Evaluation, pp. 18– 41, 2014.

**[13] Fang, Weiwei., Li, Yangchun., Zhang, Huijing., Xiong, Naixue., et al.** *"On The Throughput-*

*Energy Trade off For Data Transmission between Cloud and Mobile Devices".* Information Sciences, pp 79–93, 2014.

**[14] Alzamil, Ibrahim., Djemame, Karim., et al.,** *"Energy-Aware Profiling for Cloud Computing Environments",* Electronic Notes in Theoretical Computer Science pp. 91–108, 2015.

**[15] Prem Bianzino, Aruna., Claude, Chaudet., Rossi, Dario., et al.,** *" A Step Towards Energy-Efficient Wired Networks",* Green Networking, 2013.

**[16] Niewiadomska-Szynkiewicz, Ewa., Sikora, Andrzej., Arabas, Piotr., et al.** *"Control System for Reducing Energy Consumption in Backbone Computer Network".* Concurrency Computation, pp. 1738–1754, 2013.

**[17] Luca, Chiaraviglio., et al**. *"Modeling Sleep Mode Gains in Energy-Aware Networks", Computer* Networks 57, Pp. 3051–3066, http://dx.doi.org/10.1016/j.comnet.2013.07.011, 2013.

**[18]** https://courses.cs.washington.edu/courses/cse373/17su/lectures/Lecture%2016%20-20Dijkstra's%20 Algorithm.pdf.

**[19] Corman, Thomas H., et al. "***Introduction to Algorithms Second Edition"* The MIT

Press Cambridge, Massachusetts London, England, ISBN 0-07-013151-1 (McGraw-Hill), 2002.

**[20] Tabatabaei, Shayesteh**., *"A New Routing Protocol for Energy Optimization in Mobile Ad Hoc Networks Using the Cuckoo Optimization and the TOPSIS Multi-Criteria Algorithm",* Cybernetics and Systems An International Journal, DOI: 10.1080/01969722.2021.1899597, www.tandfonline.com / action/ journalInformation?journalCode=ucbs20,2021.

**[21] Shaun , Duerr ., Ababei ,Cristinel., & Ionel, Dan M.,** *"A Case for Using Distributed Energy Storage for Load Balancing and Power Loss Minimization in Distribution Networks"* Electric Power Components and Systems,ISSN:1532-5008print/1532-5016 online https://doi.org/10.1080/15325008.2020.1825556 www.tandfonline.com /, 2020.

**[22] Islam, Kamrul., and Akl, Selim G.,** *"A Local Algorithm To Compute Multiple Connected Dominating sets in wireless sensor networks"* International Journal of Parallel, Emergent and Distributed Systems, Vol. 26, No. 5, 369–380, www.tandfonline.com , October 2011.

**[23] Zhao ,Jiewen., & Yuan, Jiaojie.,** *"An Intelligent Model To Reduce The Energy Consumption of Sensor Network Nodes"* International Journal Of Computers and Applications https://doi.org/10.1080/1206212X.2019.1707436, ISSN: 1206-212X (Print) 1925-7074, www.tandfonline.com, 2019