**Analysis of the algorithms:**

Many of the asymptotically fastest algorithms in max flow problems are push relabel algorithms. Fastest implementations of max flow problems usually involve push relabel algorithms. While for maximum flow problems using ford Fulkerson method and max flow min cut theorem the running time of the problem depends on the selection of augmenting path. If the selection is done poorly the algorithm might not even terminate. The time complexity mentioned here for the ford Fulkerson algorithm is when the edges contain integer values. If the values are real valued, the algorithm never terminates. Hence comes the Edmonds Karp algorithm that finds the augmenting path by using breadth first search algorithm terminating in polynomial time. For the ford Fulkerson algorithm the run time is O(EF) if edge capacities are integers, where E is the number of edges and F is the increment occurring for each addition of the path. The idea behind the algorithm is to find augmenting paths and add it to the flow until no more augmenting paths can be added. That is, when the flow is maximized. When all the edges are added, it is the worst case time complexity. The lesser the number of edges, with few augmenting paths, the better the time taken. For the Edmonds Karp algorithm, the path chosen is done using BFS in which case it is optimal, making finding of augmenting paths easier.

The time complexity also depends on various factors like

1. the form of representation of the graph problem( such as adjacency matrix or adjacency list) depending on whether the graph is dense or sparse. Usually directed graphs are used in the max flow problems.
2. The data given as the input which are the number of edges in the graph
3. The time taken to find an augmenting path

The time complexity of push relabel operation is given by O(V2E) which is asymptotically more efficient than Edmonds-Karp algorithm. Depending upon the specific requirements even lower time complexity can be achieved. The variant based on highest label node selection is O(V2) and a sub cubic time complexity can be achieved using dynamic trees though it is not widely used. The reason attributed to the lower time complexity of this algorithm is because it does not view the entire graph. It views only its neighbors making the time consumed less supported by the push and relabel operation upon the node for each of its neighbors.

Figure comparison of time for ford Fulkerson and push relabel

This is a graph obtained calculating the time of ford Fulkerson and push relabel algorithms. The graph is calculated for same data input that is, for the same graphs. The y-axis denotes the time taken and x-axis, the constant value of the graph. Since the values of the graph cannot be plotted no value or rather a constant value is stated in the plot. Series 2 is obtained for the execution of Ford Fulkerson algorithm which is clearly taking more time than the push relabel algorithm which is shown in series 1. The second column in the table indicates that the operation is performed on same data. Since Edmonds Karp is simply the modification of Ford Fulkerson, a separate analysis for the algorithm is not made. A huge difference in the execution times can be noticed as the network of the graph varies. For large graphs, it can be seen that utilization of push relabel method the effect can be significant.

From this analysis, it can be concluded that the push relabel algorithms are more efficient than most other flow maximizing algorithms including the ones mentioned here.