Overview of DenStream Algorithm:

1. The DenStream algorithm is a method for dynamic data stream clustering. It operates with the following initial parameters:
    a. Epsilon: This parameter determines whether a point belongs to a micro-cluster or a cluster (joining micro-clusters).
    b. Lambda (Decaying Factor): It reduces the importance of previous points in micro-clusters. A higher value gives less weight to older points.
    c. Beta (0 < beta <= 1): A scaling factor.
    d. Mu: The minimum number of points required in a cluster (epsilon Neighborhood), specified as an integer.
2. Initial State with 100 Points:
    a. Initially, there are no potential or outlier micro-clusters.
    b. As each point arrives, it is inserted and checked for merging with existing clusters.
    c. If no cluster exists, a new micro-cluster (potentially an outlier cluster) is created.
    d. If beta * mu is less than the micro-cluster weight, it becomes the actual cluster. Otherwise, it resides in the outlier buffer as a potential cluster.
    e. Periodically, we evaluate potential clusters to determine if they will never become actual clusters, using the decay factor (lambda). If they are deemed unlikely to become actual clusters, they are deleted to free up memory.
    f. Once a potential cluster transitions to an actual cluster, it remains in memory indefinitely.
    g. Ultimately, we can apply the DBSCAN algorithm to cluster the micro-clusters.

3. Conclusion:
    a. The DenStream algorithm proves beneficial for continuous clustering in time-series data. Its adaptability to evolving data streams makes it a valuable tool for dynamic clustering applications.
4. Here is the algorithm from the paper itself:

Algorithm 2 DenStream $(DS, \epsilon, \beta, \mu, \lambda)$

```
T_p = ⌈(1/λ) log(βμ/(βμ−1))⌉;
Get the next point p at current time t from data stream DS;
Merging(p);
if (t mod T_p)=0 then
    for each p-micro-cluster c_p do
        if w_p(the weight of c_p)< βμ then
            Delete c_p;
        end if
    end for
    for each o-micro-cluster c_o do
        ξ = (2^(−λ(t−t_o+T_p)) − 1) / (2^(−λT_p) − 1);
        if w_o(the weight of c_o)< ξ then
            Delete c_o;
        end if
    end for
end if
if a clustering request arrives then
    Generating clusters;
end if
```

Algorithm 1 Merging (p)

Try to merge p into its nearest p-micro-cluster c_p;
if r_p (the new radius of c_p) $\leq \epsilon$ **then**
    Merge p into c_p;
else
    Try to merge p into its nearest o-micro-cluster c_o;
    if r_o (the new radius of c_o) $\leq \epsilon$ **then**
        Merge p into c_o;
        if w (the new weight of c_o) $> \beta\mu$ **then**
            Remove c_o from outlier-buffer and create a new p-micro-cluster by c_o;
        end if
    else
        Create a new o-micro-cluster by p and insert it into the outlier-buffer;
    end if
end if

References:

Cao, Feng, et al. "Density-based clustering over an evolving data stream with noise." *Proceedings of the 2006 SIAM international conference on data mining*. Society for industrial and applied mathematics, 2006.