```
Input: $\mathbf{W} = \{\frac{1}{n}\}$; $\eta = \lambda$;$L$ initial cluster centroids.
Output: A distance matrix $D_{ij}[\mathbf{W}]$ between all object pairs.
while W not stable do
    foreach Attributes k = 1 to n do
        foreach Objects i = 1 to N do
            foreach Objects j = 1 to N do
                s[k] += $\frac{1}{N^2}|x_{ik} - x_{jk}|$;
            end
        end
        foreach Objects i = 1 to N do
            foreach Objects j = 1 to N do
                d[i][j][k] = $\frac{|x_{ik}-x_{jk}|}{s[k]}$;
                D[i][j] += w[k][i] * exp($-\frac{d[i][j][k]}{\eta}$);
            end
        end
    end
    foreach Objects i = 1 to N do
        foreach Objects j = 1 to N do
            D[i][j] = -$\eta$ * log(D[i][j]);
        end
    end
    foreach Objects i = 1 to N do
        foreach Objects j = 1 to N do
            D[i][j] = max(D[i][j],D[j][i]);
        end
    end
    foreach Objects i = 1 to N do
        foreach Objects j = 1 to N do
            Find K=$\sqrt{N}$ nearest neighbors of Object i;
        end
    end
    foreach Attributes k = 1 to n do
        foreach Objects i = 1 to N do
            foreach Objects j = 1 to K do
                S[k][i] += $\frac{1}{K}$ * d[i][j][k];
            end
            w[k][i] = exp(-$\frac{S[k][i]}{\lambda}$);
        end
    end
    foreach Objects i = 1 to N do
        foreach Attributes k = 1 to n do
            Sum[i] += w[k][i];
        end
        foreach Attributes k = 1 to n do
            w[k][i] /= Sum[i];
        end
    end
end
```

Algorithm 1: Sequential COSA2 algorithm

```
Input: $\mathbf{W} = \{\frac{1}{n}\}$; $\eta = \lambda$; $\overline{N} = \frac{N}{p_x}$, $\overline{n} = \frac{n}{p_z}$; $p = p_x \cdot p_y \cdot p_z$.
Output: A distance matrix $D_{ij}[\mathbf{W}]$ between all object pairs.
while $\boldsymbol{W}$ not stable do
    while $nIter > 0$ do
        foreach Attributes $k$ = 1 to $\overline{n}$ do
            foreach Objects $i$ = 1 to $\overline{N}$ do
                foreach Objects $j$ = 1 to $\overline{N}$ do
                    local_s[k] += $\frac{1}{N^2}|x_{ik} - x_{jk}|$;// $s_k = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}|x_{ik} - x_{jk}|$
                end
            end
        end
        MPI_Allreduce(local_s, s, $\overline{n}$, MPI_FLOAT, MPI_SUM, grid.slice_comm_along_z);
        foreach Attributes $k$ = 1 to $\overline{n}$ do
            foreach Objects $i$ = 1 to $\overline{N}$ do
                foreach Objects $j$ = 1 to $\overline{N}$ do
                    d[i][j][k] = $\frac{|x_{ik} - x_{jk}|}{s[k]}$;// $d_{ijk} = \frac{|x_{ik} - x_{jk}|}{s_k}$
                    // $D_{ij}^{(\eta)}[\mathbf{w}] = -\eta \cdot \log \sum_{k=1}^{n} w_{ki} \cdot e^{-\frac{d_{ijk}}{\eta}}$
                    local_D[i][j] += local_w[k][i] * exp($-\frac{d[i][j][k]}{\eta}$);
                end
            end
        end
        MPI_Allreduce(local_D, D, $\overline{N} \cdot \overline{N}$, MPI_FLOAT, MPI_SUM, grid.depth_comm);
        foreach Objects $i$ = 1 to $\overline{N}$ do
            foreach Objects $j$ = 1 to $\overline{N}$ do
                D[i][j] = -$\eta$ * log(D[i][j]);
            end
        end
        //$D_{ij}^{1}[\mathbf{W}] = \max(D_{ij}^{(\eta)}[\mathbf{w}_{c(i|\mathbf{W})}], D_{ij}^{(\eta)}[\mathbf{w}_{c(j|\mathbf{W})}])$ (30)
        Combining observation pair weights (max): D[i][j] = max(D[i][j],D[j][i]);
        foreach Objects $i$ = 1 to $\overline{N}$ do
            foreach Objects $j$ = 1 to $\overline{N}$ do
                Find KNN of Object i → local_KNN[i][1...K]; // KNN($i$) = $\{j | D_{ij} \leq d_{i(K)}\}$
            end
        end
        MPI_Allgather(local_KNN, $\overline{N} \cdot K$, MPI_FLOAT,
                    global_KNN, $p_x \cdot \overline{N} \cdot K$, MPI_FLOAT, grid.row_comm);
        foreach $i$ = 1 to $\overline{N}$ do
            Find KNN[i][1...K] from global_KNN[i][1...$p_x \cdot K$];
        end
        foreach Attributes $k$ = 1 to $\overline{n}$ do
            foreach Objects $i$ = 1 to $\overline{N}$ do
                foreach Objects $j$ = 1 to $K$ do
                    local_S[k][i] += $\frac{1}{K}$ * d[i][j][k];// $S_{ki} = \frac{1}{K}\sum_{j \in KNN(i)} d_{ijk}$
                end
            end
        end
        MPI_Allreduce(local_S, S, $\overline{n} \cdot \overline{N}$, MPI_FLOAT, MPI_SUM, grid.row_comm);
        foreach Attributes $k$ = 1 to $\overline{n}$ do
            foreach Objects $i$ = 1 to $\overline{N}$ do
                local_w[k][i] = exp(-$\frac{S[k][i]}{\lambda}$);// $w_{ki} = \frac{e^{-\frac{S_{ki}}{\lambda}}}{\sum_{k'=1}^{n} e^{-\frac{S_{ki}}{\lambda}}}$
            end
        end
        Normalize local_w → w.
    end
    $\eta = \eta + \alpha \cdot \lambda$
end
```

Algorithm 2: Parallel COSA2 algorithm