Appendix B: Algorithm Pseudo Code

This appendix contains the pseudo code for the proposed optimal de-identification algorithm. The auxiliary functions below are used in the algorithm.

Function	Description
InfoLoss(*node*)	Computes the information loss for a particular node in the lattice. The particular information loss metric maybe Prec, DM*, non-uniform entropy, or any other metric that satisfies the monotonicity property.
IsKAnonymous(*node*)	Determines whether the node is k-anonymous. This is the most time consuming function in the algorithm.
IsTaggedKAnonymous(*node*)	Determines whether a particular node has already been tagged as k-anonymous. Returns True or False.
IsTaggedNotKAnonymous(*node*)	Determines whether a particular node has already been tagged as not k-anonymous. Returns True or False.
TagKAnonymous(*node*)	Tag a particular *node* as k-anonymous. This will also tag as k-anonymous all other higher nodes in the lattice along the same generalization strategies that pass through the *node*.
TagNotKAnonymous(*node*)	Tag a particular node as not k-anonymous. This will also tag as not k-anonymous all other lower nodes in the lattice along the same generalization strategies that pass through the *node*.
Node(*lattice*,*height*,*index*)	This function is used to navigate a lattice. It returns the node at *index* in a particular *height*. The *index* values start from the left.
Lattice(*bottom-node*, *top-node*)	Creates a lattice with a particular node at the bottom and another at the top.
Height(*lattice*, *node*)	This function returns the height of a particular node in the particular (sub-) lattice.
Width(*lattice*, *height*)	Returns the number of nodes at a particular height in the lattice. This is used mainly to traverse a level in a lattice.
CleanUp(*node*)	Removes all nodes in the solutions set that are generalizations of *node* (i.e., on the same generalization strategies).

The algorithm below is started by creating the top and bottom nodes of the main lattice.

KMin Algorithm

```
// takes the root and sink nodes of a lattice as input

Kmin(Bnode,Tnode)
{
    L=Lattice(Bnode,Tnode)
    $H_H$=Height(L,Tnode)
    If $H_H$ > 1 then
        h= $\left\lfloor {}^{H_H}\!/_{2} \right\rfloor$
        For p=1 to Width(L, h)
            N = Node(L,h,p)
            If IsTaggedKAnonymous(N) == True then
                Kmin(Bnode, N)
            Else if IsTaggedNotKAnonymous(N) == True then
                Kmin(N,Tnode)
            Else if IsKAnonymous(N) == True then
                TagKAnonymous(N)
                KMin(Bnode,N)
            Else
                TagNotKAnonymous(N)
                KMin(N,Tnode)
            End If
        End For
    Else
        // this is a special case of a two node lattice
        if IsTaggedNotKAnonymous(Bnode) == True then
            N = Tnode
        Else if IsKAnonymous(Bnode) == True then
            TagKAnonymous(Bnode)
            N = Bnode
        Else
            TagNotKAnonymous(Bnode)
            N = Tnode
        End If

        S= S + N
        CleanUp(N)
    End if
}
```

Main

```
Main
    S=$\{\ \}$
    KMin(Bottom-Node, Top-Node)
    Optimal=$\min_{x \in S}\left(InfoLoss\left(x\right)\right)$
End Main
```