Homology Search Algorithm

Input: *vertex* set $V_{experimental}$, consists of experimental compounds

Output: List *L*, consists of tuples and each tuple (*v*, *u*, *r*) represents that *compounds v* and *u* have relationship *r*

function($V_{experimental}$)
    L ← empty *tuple* list
    for each *v, u* in $V_{experimental}$ **do**
        if diff($Mass_v$, $Mass_u$) ≈ the mass of specific nucleobase or methylation **then**
            $Mass_{specific}$ ← the specific mass
            push (*v*, *u*, $Mass_{specific}$) into *L*
        end **if**
    end **for**

    return *L*
end **function**

Acid Labile Algorithm

Input: *vertex* set V_{noad}, consists of experimental compounds before acid digest
vertex set V_{ad}, consists of experimental compounds after acid digest

Output: List *L*, consists of tuples and each tuple (*v*, *u*, *r*) represents that *compound v* changes to *u* after acid treatment, and their relationship is *r*

function($V_{noad,}$ V_{ad})
    L ← empty *tuple* list
    for each *v in* V_{noad} and *u* in V_{ad} **do**
        if diff($Mass_v$, $Mass_u$) ≈ any acid labile mass difference **then**
            $Mass_{acid\_labile}$ ← the acid labile mass difference
            push (*v*, *u*, $Mass_{acid\_labile}$) into *L*
        end **if**
    end **for**

    return *L*
end **function**

MassSum Algorithm

Input: *vertex set* $V_{experimental}$, consists of experimental compounds
$Mass_{intact}$, an intact mass

Output: vertex *set V*, consists of compounds related to the given intact mass

function ($V_{experimental}$, $Mass_{intact}$)
    $Mass_{H2O}$ ← initialize with mass value of H2O
    V ← initialize with empty *set*
    for each *v, u* in $V_{experimental}$ **do**
        if sum($Mass_v$, $Mass_u$) ≈ $Mass_{intact}$ + $Mass_{H2O}$ **then**
            push (*v*, *u*) into *V*
        end **if**
    end **for**

    return *L*
end **function**

GapFill Algorithm

Input: *vertex* set $V_{experimental}$, consists of experimental compounds masses tuple ($Mass_{left}$, $Mass_{right}$), the masses of both ends

Output: *vertex* set *V*, consists of compounds chosen from the gap

function($V_{experimental}$, $Mass_{left}$, $Mass_{right}$)
    V ← empty *vertex* set
    for each *v* in $V_{experimental}$ **do**
        $Link_{left}$ ← **diff**($Mass_v$, $Mass_{left}$) ≈ **sum**(Masses of specific nucleobases)
        $Link_{right}$ ← **diff**($Mass_v$, $Mass_{right}$) ≈ **sum**(Masses of specific nucleobases)
        If $Link_{left}$ and $Link_{right}$ then
            push *v* into *V*
        end **if**
    end **for**

    G ← *Graph* with each *node* represents a *vertex* of *V*
    for each *v*, *u* in G **do**
        if diff($Mass_v$, $Mass_u$) ≈ **sum**(Masses of specific nucleobases) **then**
            add *edge* (*v*, *u*) into *G*
        end **if**
    end **for**

    while *True* **do**
        Num_{max} ← the max *edges* count for any *nodes* of *graph G*
        $Node_v$ ← the *node* has the least edges in *graph G*
        Num_{min} ← the *edges* count of *node* $Node_v$
        if Num_{min} >= Num_{max} **then**
            break
        else then
            remove $Node_v$ and its edges from *G*
        end **if**
    end **while**

    V ← vertices for all the remain nodes of *G*
    return *V*
end **function**

Ladder Complemention Algorithm

Input: List L_{in}, consists of a tuple (V, $Mass_{intact}$, d) list, V is a *vertex* set contains a set of compounds, or ladder. $Mass_{intact}$ represents the intact mass relative to V, and d is the ladder direction, etc. from 5' to 3' direction, only 3|5 is allowed

Output: List L_{out}, consists of a tuple (V, S) list, V represents a mass ladder, and S is the nucleotide sequence of V

function(L_{in})
    L_{out} ← empty *tuple* list, and allocate space for at least 76 positions
    for each (V, $Mass_{intact}$, d) in L_{in} **do**
        L_{mass} ← the mass value list for vertices of V
        if d equals 3 **then**:
            $Mass_{H2O}$ ← the mass value of H2O
            for each mass m in L_{mass} do
                $m \leftarrow Mass_{intact} + Mass_{H2O} - m$
            end **for**
        end **if**

        for each mass m in L_{mass} **do**
            $p \leftarrow$ **int**(m / *320.0*)
            push m into L_{out} at the end of position p
        end **for**

    for each item pairs (p_v, p_u) at adjacent positions in L_{out} **do**
        calculate their mass differences and get the relative nucleotide bases
    end **for**

    for each mass ladder in L_{out} **do**
        V ← the mass ladder
        S ← the nucleotide sequence
        push (V, S) into L_{out}
    end **for**

    return L_{out}
end **function**