Weights Problem

Assume we have N items such that item i weights w(i) and that a carry weight, max, is given. Select an optimal set of items such that the total weigth of the items will be as close to (and less than) the carry weight, max. This is sometimes referred to as the 'knapsack' problem.

The solution of this problem can be applied to many areas, for example, suppose I want to tape a selection from 20 songs to fit on a 30 minute side of a tape such that the total time of the selection will be a close to 30 minutes as possible.

A possible solution can be adapted from the program for generating sets. We generate all the subsets and keep track of the optimal set so far, and when all the subsets have been generated we output the optimal set. This solution calculates the weight of all subsets even though many will be deemed impossible to be included in a solution. For example, if the optimal weight so far is say 60 and we find that adding all the remaining items won't make the current set heavier than this, then we should ignore the current set and all super-sets of it.

Suppose we have the following 14 items with weights

1	2	3	4	5	6	7	8	9	10	11	12	13	14
180	227	126	163	144	186	152	191	150	180	182	166	191	208

Assume max carry weight is 1800 then an optimal selection (10 items) could be

{1, 2, 3, 4, 7, 8, 10, 11, 13, 14}

Total weight is: 1800

or if the max carry weight was 1790 then the selection (also 10 items) could be

{1, 2, 3, 4, 6, 7, 8, 12, 13, 14}

Total weight is: 1790

The total weight may not equal the carry weight. For example, if the carry weight was 13 and the weights were 6, 4 & 8 then the optimal weight would be 12.

The routine Add_El

The procedrue call

add_el (i, sum, achwgt)

adds an item, i**, to the current set whose current weight (before i is added) is** sum**. The weight achievable by the current set is** achwgt**. Suppose the weights were 6, 4 & 8. Assume we are adding item 2 under the condition that item 1 is to be excluded then the achievable weight is 12, as item 1 is not counted. The achievable weight is also governed by the carrry weight, max. If max=13 and items 1 & 2 are in the current set, then adding item 3 (weight 8) is too much and so the achievable weight is current achievable weight minus**

weight of item 3, achwgt - 8.

In the routine that follows,the attribute, s**, indicates the current set. The attribute,** opset**, is the current optimal set with optimal weight,** opt**.**

```
Add_El(i,sum,AchWgt : INTEGER) is
        local
                newsum, new_ach : INTEGER
        do
                newsum := sum + wgts.item(i)
                if newsum <= max then
                        -- including item i
                        s.put(true,i) -- add i to s
                        if i < s.count then
                                Add_El(i+1,newsum,AchWgt)
                        elseif AchWgt > opt then
                                opt := AchWgt
                                opset.copy(s)
                        end
                        s.put(false,i) -- remove i
                end
                -- excluding item i
                new_ach := AchWgt - wgts.item(i)
                if opt < new_ach then
                        if i < s.count then
                                Add_El(i+1,sum,new_ach)
                        else
                                opt := new_ach
                                opset.copy(s)
                        end
                end
        end -- Ad_El
```

```
class
        WEIGHTS
creation
        make
feature
        s,opset : ARRAY[BOOLEAN] -- Integer Set, s@i iff i ε s
        wgts : ARRAY[INTEGER]
        opt, max : INTEGER

        Add_El(i,sum,AchWgt : INTEGER) is
                local
                        newsum, new_ach : INTEGER
                do
                        newsum := sum + wgts.item(i)
                        if newsum <= max then
                                -- include item i
                                s.put(true,i) -- add i to s
                                if i < s.count then
                                        Add_El(i+1,newsum,AchWgt)
                                elseif AchWgt > opt then
                                        opt := AchWgt
                                        opset.copy(s)
                                end
                                s.put(false,i) -- remove i
                        end
                        -- excluding i
                        new_ach := AchWgt - wgts.item(i)
                        if opt < new_ach then
                                if i < s.count then
                                        Add_El(i+1,sum,new_ach)
                                else
                                        opt := new_ach
                                        opset.copy(s)
                                end
                        end
                end -- Ad_El
```

```
make is
        local
                N,k, totw : INTEGER
        do
                io.put_string("%N Enter number of items: ")
                io.read_integer
                N := io.last_integer
                !!s.make(1,N)
                !!opset.make(1,N)
                !!wgts.make(1,N)
                opt := 0 -- by default it is 0
                read_file("wgts.txt",N)
                io.new_line
                from
                        totw := 0
                        k := 1
                until
                        k > N
                loop
                        totw := totw + wgts.item(k)
                        io.put_integer(wgts.item(k))
                        io.putchar(' ')
                        k := k+1
                end
                io.put_string("%N Enter Max carry weight: ")
                io.read_integer
                max := io.last_integer
                Add_El(1,0,totw)
                print_set(opset,opt)
        end -- make
```

```
        read_file(flname:STRING; size:INTEGER) is
                local
                        in_file : PLAIN_TEXT_FILE
                        w,i : INTEGER
                do
                        from
                                !!in_file.make_open_read(flname)
                                i := 1
                        until
                                i > size
                        loop
                                in_file.read_integer
                                w := in_file.last_integer
                                wgts.put(w,i)
                                i := i+1
                        end
                        in_file.close
                end -- read_file

        Print_Set( s1:ARRAY[BOOLEAN]; wgt:INTEGER) is
                local
                        i,k,tot : INTEGER
                do
                from
                        tot := 0
                        io.putchar('{')
                        i := 1
                        k := 0
                until
                        i > s1.count
                loop
                        if s1.item(i) then
                                if k = 0 then
                                        io.put_integer(i)
                                else
                                  io.putchar(',')
                                  io.put_integer(i)
                                end
                                  k := k+1
                        end
                        i := i+1
                end
                io.putchar('}')
                io.put_string("%N Total weight is: ")
                io.put_integer(wgt)
        end -- Print_Set

end -- Class WEIGHTS
```