Ying Lu June 23 (3:30pm)

derived from the previous scheme a depth-first algorithm in which: (a) selection of the branching variable x_j is the same as in Kolesar; (b) the search continues from the node associated with the insertion of item j (condition $x_j = 1$), i.e. following a greedy strategy.

Other algorithms have been derived from the Greenberg–Hegerich approach (Barr and Ross (1975), Laurière (1978)) and from different techniques (Lageweg and Lenstra (1972), Guignard and Spielberg (1972), Fayard and Plateau (1975), Veliev and Mamedov (1981)). The Horowitz–Sahni one is, however, the most effective, structured and easy to implement, and has constituted the basis for several improvements.

2.5.1 The Horowitz–Sahni algorithm

$\frac{P_1}{W_1} \leq \frac{P_2}{W_2} \leq \frac{P_3}{W_3} \cdots \leq \frac{P_n}{W_n}$ (P_j = processor power consumption; W_j = processor capacity)

Assume that the items are sorted as in ~~(2.7)~~. A *forward move* consists of inserting the largest possible set of new consecutive items into the current solution. A *backtracking move* consists of removing the last inserted item from the current solution. Whenever a forward move is exhausted, the upper bound U_1 corresponding to the current solution is computed and compared with the best solution so far, in order to check whether further forward moves could lead to a better one: if so, a new forward move is performed, otherwise a backtracking follows. When the last item has been considered, the current solution is complete and possible updating of the best solution so far occurs. The algorithm stops when no further backtracking can be performed.

In the following description of the algorithm we use the notations

$(\hat{x}_j)$ = current solution;

$\hat{z}$ = current solution value $\left(= \sum_{j=1}^{n} p_j \hat{x}_j\right)$;

$\hat{c}$ = current residual capacity $\left(= c - \sum_{j=1}^{n} w_j \hat{x}_j\right)$; $\cancel{\left(\sum_{j=1}^{n} W_j \hat{x}_j = U_{tot}\right)}$

(x_j) = best solution so far;

z = value of the best solution so far $\left(= \sum_{j=1}^{n} p_j x_j\right)$.

Assumption : $\sum_{j=1}^{n} W_j \geq U_{tot}$

procedure HS:
input: $n, c, (p_j), (w_j)$;
output: $z, (x_j)$;
begin
1. [initialize]
$z := 0$;
$\hat{z} := 0$;

Ying Lu June 23 (3:30pm)

$\hat{c} := \cancel{c}$; u_{tot}
$p_{n+1} := \cancel{0}$; $+\infty$
$w_{n+1} := +\infty$;
$j := 1$; $\hat{i} = -1$;

2. [compute upper bound U_1]
   $\boxed{\text{find } r = \min\{i : \sum_{k=j}^{i} w_k > \hat{c}\};}$

   sum = 0;
   for k = j to n+1 do
   { sum += w_k;
   if sum > $\hat{c}$
   { r = k;
   break } }

   ~~$u := \sum_{k=j}^{r-1} p_k + \lfloor(\hat{c} - \sum_{k=j}^{r-1} w_k)p_r/w_r\rfloor$;~~
   ~~**if** $z \geq \hat{z} + u$ **then go to** 5;~~

   m = −1; minP = +∞;
   for k = j−1 to 1 do
   if ($\hat{x}_k$ == 0 && P_k < minP)
   then { m = k;
   minP = P_k; }

   if m == −1
   then $u = \sum_{k=j}^{r-1} P_k + (\hat{c} - \sum_{k=j}^{r-1} W_k)\, P_r/W_r$;
   else if $W_m \geq \hat{c} - \sum_{k=j}^{r-1} W_k$
   then $u = \sum_{k=j}^{r-1} P_k + P_m$
   else $u = \sum_{k=j}^{r-1} P_k + P_m + (\hat{c} - \sum_{k=j}^{r-1} W_k - W_m)\, P_r/W_r$;
   if z == 0 then go to 3
   if z ≤ $\hat{z}$ + u then go to 5

3. [perform a forward step]
   while $w_j \leq \hat{c}$ **do**
   begin
   $\hat{c} := \hat{c} - w_j$;
   $\hat{z} := \hat{z} + p_j$;
   $\hat{x}_j := 1$;
   $j := j + 1$
   end;
   if $j \leq n$ **then**
   begin
   $\hat{x}_j := 0$;
   $j := j + 1$
   end;
   if $j < n$ **then go to** 2;
   if $j = n$ **then go to** 3; 3.5

3.5 if $\hat{c}$ == 0 then { $\hat{i}$ = −1; go to 4 }
   for k = 1 to n do
   if ($\hat{x}_k$ == 0 && $W_k \leq \hat{c}$)
   then go to 5 //if we reach this point, then solution so far is infeasible!
   minP = +∞;
   $\hat{i}$ = −1
   for k = 1 to n do
   { $\hat{y}_k$ = 0;
   if ($\hat{x}_k$ == 0 && P_k < minP)
   { minP = P_k; h = k } }
   $\hat{y}_h$ = 1; $\hat{i} = \hat{z} + P_h$

4. [update the best solution so far]
   if ~~$\hat{z} > z$~~ **then** z == 0 or max($\hat{z}$, $\hat{i}$) < z
   begin
   $z := \cancel{\hat{z}}$; max($\hat{z}$, $\hat{i}$);
   for $k := 1$ **to** n **do** $x_k := \hat{x}_k$
   end;
   for k = 1 to n do
   if ($\hat{y}_k$ == 1)
   x_k = 1
   $j := n$;
   if $\hat{x}_n = 1$ **then**
   begin
   $\hat{c} := \hat{c} + w_n$;
   $\hat{z} := \hat{z} - p_n$;
   $\hat{x}_n := 0$
   end;

5. [backtrack]
   $\boxed{\begin{array}{l}\text{find } i = \max\{k < j : \hat{x}_k = 1\};\\ \textbf{if } \text{no such } i \textbf{ then return};\end{array}}$

   $\hat{i}$ = −1
   for k = j−1 to 1 do
   if $\hat{x}_k$ == 1
   then { $\hat{i}$ = k; break }
   if $\hat{i}$ == −1 then
   return;

   $\hat{c} := \hat{c} + w_i$;
   $\hat{z} := \hat{z} - p_i$;
   $\hat{x}_i := 0$;
   $j := i + 1$;
   go to 2
end.

$U_{tot} = 5$

$n = 4$

$\frac{p_i}{w_i}$: $\frac{7}{2}$ $\frac{8}{2}$ $\frac{18}{4}$ $\frac{10}{2}$

i = 1 2 3 4

$r = 5$

$m = 1$

$U = \sum_{k=4}^{4} P_k + P_m + (\hat{c} - W_4 - W_1) * \frac{P_r}{W_r}$

$= 10 + 7 + (+\infty) \frac{P_r}{W_r}$

$= +\infty$