المحاضرة الحادية عشر

ب) طريقة الطماع (Greedy method)

تستخدم هذه الطريقة غالبا لحل مسائل الامثلية (optimization problem). لهذه المسائل تعطى مجموعة قيود(constraints) ودالة هدف (objective function). مجموعة الحلول التي تحقق القيود تسمى حلولا ممكنة(feasible solution) والحل الممكن الذي يعطي افضل دالة هدف يسمى الحل الامثل (optimal solution).

ملاحظة / اغلب المسائل التي تحلها طريقة الطماع تتكون من n من المداخل.

الفكرة/ يبنى الحل الامثل في طريقة الطماع على مراحل. في كل مرحلة يتخذ افضل قرار تبعا لمقياس امثلية مناسب (مقياس الطماع). وحيث ان القرار المتخذ لن يتغير لاحقا لذا يجب ان يضمن امكانية الحل.

يوجد نموذجان لطريقة الطماع :-
١) نموذج المجموعة الجزئية(Subset paradigm):- في هذا النموذج يتم انتقاء مجموعة جزئية مثلى من المدخلات تبعا لمقياس امثلية معين ويجب ان تحقق هذه المجموعة قيود المسالة. التجريد الضبطي التالي يوضح عمل هذا النموذج:

```
SolType Greedy(Type a[ ], int n)
// a[1:n] contains the n inputs.
{
   SolType solution=Empty; // initialize the solution.
   for(int i=1; i<=n; i++){
      Type x=Select(a);
      if feasible(solution, x)
        solution=Union(solution, x);
   }
    return solution;
}
```

٢) نموذج الترتيب(Ordering paradigm): في هذا النموذج يتم اتخاذ القرارات باعتبار المدخلات بترتيب معين. كل قرار يتم اتخاذه باستخدام مقياس للامثلية والذي يمكن حسابه من خلال القرارات المتخذه مسبقا.

امثلة على طريقة الطماع:-

١- مسألة الجراب(Knapsack problem)

المعطيات:
- n من الكيانات.
- جراب سعته C.
- للكيان i الوزن w_i $(1 \le i \le n)$.
- اضافة الكسر x_i $(0 \le x_i \le 1)$ يحقق فائدة $p_i x_i$.

المطلوب:
ملء الجراب بحيث تعظم الفائدة الحاصلة.

- دالة الهدف

$$\text{Maximize} \sum_{i=1}^{n} p_i x_i$$

- القيود

$$\sum_{i=1}^{n} w_i x_i \le C$$

مثال/

$C=20\ ,\ n=3\ ,\ P[1:3]=(25, 24, 15)\ ,\ w[1:3]=(18, 15, 10)$

توجد ثلاث حلول ممكنة باستعمال طريقة الطماع وكما مبين في الجدول التالي:

(x_1, x_2, x_3)	$\sum w_i x_i$	$\sum p_i x_i$	المقياس
(1, 2/15, 0)	20	28.2	الفائدة الاعلى اولا
(0, 2/3, 1)	20	31	الوزن الاقل اولا
(0, 1, 1/2)	20	31.5	الفائدة الاكبر لكل وحدة وزن (p_i/w_i) اولا

```
Algorithm GreedyKnapsack(P, W, C, n):
Input: Positive integers n and C; arrays w[1:n] and
       p[1:n], and each containing positive real sorted
       in nonincreasing order according to the values of
       p[i]/w[i].
Output: an array x[1:n] where 0≤xi≤1; an real maxprofit
        that is the maximum profit
1.  for i ← 1  to  n
2.     x[i]← 0.0
3.  endfor
4.  U ← C
5.  maxprofit← 0.0
6.  for i ← 1  to  n
7.     if w[i]>U then  goto  step 13
8.       x[i]← 1.0
9.       U ← U-w[i]
10.      maxprofit ← maxprofit + p[i]
11.    endif
12. endfor
13. if i ≤ n then
14.   x[i] ← U/w[i]
15.   maxprofit← maxprofit + p[i]*x[i]
16. endif
17. return array x and  maxprofit
```

تتطلب هذه الخوارزمية بغض النظر عن وقت ترتيب الكيانات ابتدائيا O(n) من الوقت فقط. حيث تعطي هذه الخوارزمية حلا امثل دائما. لمسألة (0/1 kanpsack) تحذف تعليمة if الاخيرة وفي هذه الحالة لن يكون هناك ضمان للحصول على حل امثل.