IEDA 5230 Homework #4

No submission required.

**Dynamic Programming**

Try to solve the following problems under common constraints

x1 + x2 + … + xn = a, and x1, x2, …, xn in {0,1,…,a}

1. Max f1(x1, x2) + f2(x2, x3) + f3(x3, x4) + …. + fn-1(xn-1, xn)
2. Max f1(x1, x2) + f2(x2, x3) + f3(x3, x4) + …. + fn-1(xn-1, xn) + fn(xn, x1)

Solution 1)

Let Fk(xk, ak) be the maximum value of fk(xk, xk+1) + fk+1(xk+1, xk+2) + …. + fn-1(xn-1, xn) under the condition that xk + xk+1 + … + xn = ak.

For k = n - 1, Fn-1(xn-1, an-1) = fn-1(xn-1, an-1 - xn-1), which can be obtained directly.

For k = 1,…, n-2, there is a DP recursion

Fk(xk, ak) = max {fk(xk, xk+1) + Fk+1(xk+1, ak - xk) | xk+1 in {0, 1,…, ak-xk} }.

The optimal solution will be given by max{F1(x1, a) | x1 in {0,1,…,a} }.

Solution 2)

Let Fk(xk, ak; x1) be the max value of fk(xk, xk+1) + fk+1(xk+1, xk+2) + …. + fn-1(xn-1, xn) + fn(xn, x1) under the condition that i) xk + xk+1 + … + xn = ak and ii) x1 is fixed.

For k = n - 1, Fn-1(xn-1, an-1; x1) = fn-1(xn-1, an-1 - xn-1) + fn(an-1 - xn-1, x1).

For k = 1,…, n-2, there is a DP recursion

Fk(xk, ak; x1) = max {fk(xk, xk+1) + Fk+1(xk+1, ak - xk; x1) | xk+1 in {0, 1,…, ak-xk} }.

The optimal solution will be given by max{F1(x1, a; x1) | x1 in {0,1,…,a} }.

**Wagner and Whitin Model of production planning**. Questions 2 to 4 should use dynamic programming.

1. Formulate an integer linear program for solving the problem.
2. There is an extra requirement in which the number of productions is at most M (M < T).
3. There is a requirement in which every 5 days there is at least one production and every 8 days there are at least **three** productions.
4. There are two modes to choose for each production, with the fixed production cost k1, k2, and unit production cost, c1, c2 respectively.

**Preliminary**

In the basic Wagner and Whitin model, the T-periods problem is partitioned into some subproblems where i) there is a production at the first period in each subproblem, and ii) there is no inventory carryover between subproblems. Then the problem becomes how to find the best combination of these subproblems. We can do this because of certain optimality properties.

For questions 2) to 4), it can be verified that these optimality properties still hold, hence the structure of subproblems also holds. However, there are extra constraints on the feasible combination of subproblems, which needs more parameters to define the state.

For notational simplicity, we use C(*t*,*s*) to represent the total production and inventory cost for a subproblem from period t to period s-1, i.e.,

C(*t*,*s*)= *K* + *c*(*d*t+ *d*t+1+…+*ds*-1)+ *h*(*d*t+1+2*d*t+2+…+(*s*-1-*t*)*ds*-1).

Note that C(*t*,*s*) is slightly different from the C*t*,*s*in lecture notes.

**Solution 1)**

There are different ways to formulate. Below is one from the viewpoint of network flow.

Define binary variable xt,s=1 if periods t and s are two adjacent production periods. Then we have the following formulation.

Subject to

**Solution 2)**

Let F(t, m) be the minimum total cost from period t to T given that a) there is a production at period t, and b) the total number of productions from t to T is m. Considering the possibility of the next production at period *s*, we can have the dynamic programming recursion as

.

The DP has initial condition F(T+1,m)=0 for m=0,1,…,M. The optimal solution is given by F(1,M).

**Solution 3)**

Let F(t, s) be the minimum total cost from period t to T given that a) there is one production at period t, and the next production at period s. Note that in F(t,s), (t,s) should satisfy s=t+1,…,t+5 under the 5-periods requirement. Consider r, the next possible production after period s. Due to the 8-periods requirement from period t, r cannot be greater than t+7; due to the 5-period requirement, r cannot be greater than s+5. So the possible range of r is {s+1, s+2,…, min(s+5,t+7)}.

Then we have the dynamic programming recursion as

The DP has initial condition F(t,T+1)=C(t,T+1). The optimal solution is given by min{ F(1,s)| s=2,…6}.

**Solution 4)**

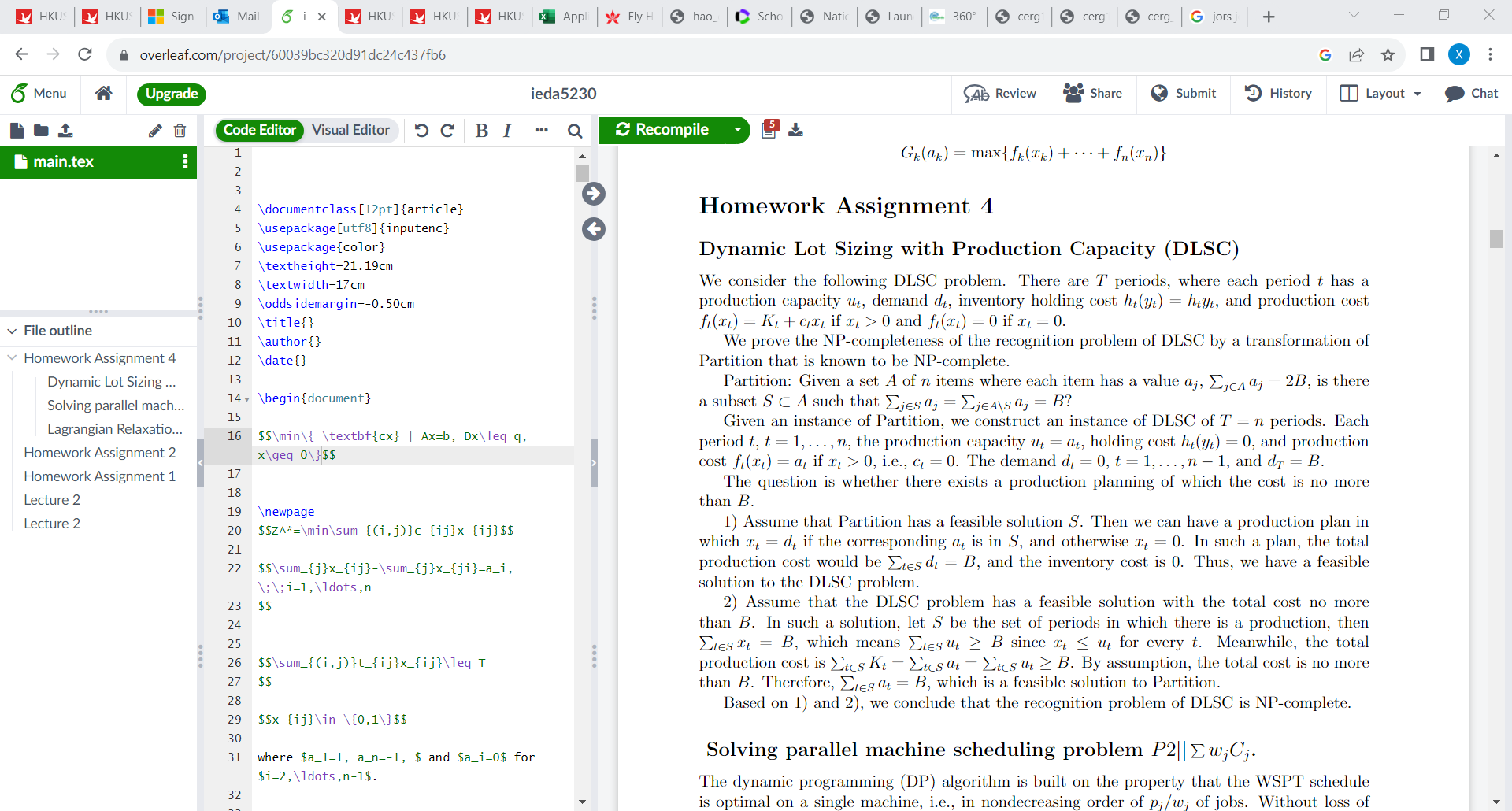
In this question, we only need to modify the calculation of each subproblem (t, s). Specifically, we only need to compare the production cost and select the lower one. The inventory cost is independent of that.

C(*t*,*s*)= min {*k1*+ *c1*(*d*t+ *d*t+1+…+*ds*-1), *k2* + *c2*(*d*t+ *d*t+1+…+*ds*-1)}+ *h*(*d*t+1+2*d*t+2+…+(*s*-1-*t*)*ds*-1).

**Remark:**

3) is similar to the fish catching example in the lecture notes. In both problems, the state is defined by two parameters because these two parameters jointly determine the feasible decisions and the consequence of each decision.

**NP-completeness of Dynamic Lot Sizing with capacity**

****

**NLP**. Solve the following NLP by K-T condition. Note that *a* is a given parameter. Discuss how the optimal solution may change with *a*.

Min*x,y,z* *x*2 + 2*y*2 + 3*z*2

Subject to *x* + *y* + *z* = 6

*x* + 2*y* + 3*z* ≤ *a*

**Solution**

Lagrangian function

L = x2+2y2+3z2 + s(x+y+z-6) + t(x+2y+3z-a)

KKT condition (we use Lx) to denote the partial derivative of L

Lx = 2x+s+t = 0

Ly = 4y+s+2t = 0

Lz = 6z+s+3t = 0

Ls = x+y+z-6 = 0

Lt = (x+2y+3z-a) ≤ 0 (x+2y+3z-a) t = 0

t >= 0

There are two cases to discuss.

Case 1: x+2y+3z<a and t=0

Solving the above equations gives

z=36/11, y=18/11, z=12/11, s=-72/11. This holds for a > 108/11

Case 2: x+2y+3z=a and 0≤t

X=9-7a/12 y=a/6 z=-3 +5a/12 t=18-11a/6 s=-108+3a

This holds for a ≤ 108/11

**Convexity.**

Consider a function g(x1,x2,…,xn), xj≥0, j=1,…,,n, which is the sum of the largest two values in (x1,x2,…,xn), for example, g(1,4,6,3)=4+6, g(4,6,6,2,5)=6+6.

Prove or disprove that g(x1,x2,…,xn) is convex. To prove, you may use some of the properties of convexity preservation (introduced in the lectures), or prove it directly based on definition. To disprove, present a counter example.

**Solution**

The function g(x1,x2,…,xn) is convex.

Proof: Define function fjk(x1,x2,…,xn)=xj+xk. Then g(x1,x2,…,xn) = max{ fjk(x1,x2,…,xn) | j<k}. Note that each fjk(x1,x2,…,xn) is a linear function and hence convex. So g(x1,x2,…,xn), as the maximum of convex functions, is convex.

For example, f1,2(1,4,6,3) =1+4 = 5, f1,4(1,4,6,3) =1+3 = 4.

Then g(1,4,6,3) = f2,3(1,4,6,3)=10