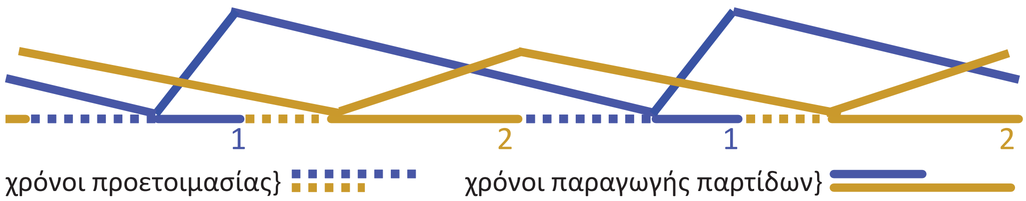
Μεγέθη παρτίδων παραγωγής (economic lot sizing)

Θεωρούμε συστήματα που παράγουν πολλούς τύπους προϊόντων. Εδώ θα κοιτάξουμε μία μηχανή που παράγει N προϊόντα, i = 1, …, N.

- Κάθε προϊόν παράγεται κατά παρτίδες, χωριστά από άλλους τύπους.
- Πριν ξεκινήσει ένα προϊόν, η μηχανή χρειάζεται χρόνο προετοιμασίας.
- Υπάρχει επίσης και κόστος προετοιμασίας. Επομένως …
- … **δεν συμφέρει να κάνουμε συχνά αλλαγές** γιατί θα χαθεί όλος ο χρόνος σε προετοιμασίες και θα αυξηθεί το κόστος παραγωγής.
- Μετά την παραγωγή τους, τα προϊόντα αποθηκεύονται και πωλούνται όταν υπάρχει ζήτηση. Υπάρχει κόστος αποθέματος. Επομένως …
- **… συμφέρει να κάνουμε συχνά αλλαγές** για να παράγουμε λίγο και συχνά κάθε προϊόν ώστε να μην μένει πολύ χρόνο στην αποθήκη.
- Έχουμε **αντικρουόμενα** κριτήρια. Θα εύρουμε τη λύση με min κόστος.



προϊόντα 1 και 2

Μεγέθη παρτίδων παραγωγής (συνέχεια)

Υποθέστε ότι οι ρυθμοί ζήτησης D_i και παραγωγής P_i είναι σταθεροί για κάθε i = 1, …, N και δεν επιτρέπονται ελλείψεις. Δίνονται

C_i: κόστος παραγωγής ανά μονάδα προϊόντος i

h_i: κόστος αποθέματος ανά μονάδα προϊόντος και ανά μονάδα χρόνου

A_i, S_i: κόστος και χρόνος προετοιμασίας πριν την παραγωγή παρτίδας i

Όταν οι παράμετροι αυτές δεν αλλάζουν με τον χρόνο, το καλύτερο πρόγραμμα παραγωγής είναι **περιοδικό**. Παραδείγματα με N = 4:

Κυκλικό πρόγραμμα: … (1, 3, 4, 2), (1, 3, 4, 2), (1, 3, 4, 2) …

Μη κυκλικό … (1, 2, 3, 1, 2, 4), (1, 2, 3, 1, 2, 4), (1, 2, 3, 1, 2, 4), …

Θα περιοριστούμε σε **κυκλικά** προγράμματα όπου, σε μία περίοδο-κύκλο, παράγεται **μόνο μία παρτίδα** από κάθε προϊόν **και με την ίδια σειρά**.

Μεγέθη παρτίδων για κυκλικό πρόγραμμα

Έστω T η διάρκεια ενός κύκλου, Q_i το μέγεθος παρτίδας και T_i ο χρόνος παραγωγής της. Ζητείται το T με το ελάχιστο κόστος. Ο χρόνος T **πρέπει να είναι αρκετός** για προετοιμασίες, παραγωγές και, αν συμφέρει, για **προσωρινή παύση λειτουργίας** για κάποιο διάστημα $T_0 \geq 0$. **Περιορισμός:**

(Π1): $T = S_1+T_1+\ldots+S_N+T_N+T_0$ και $T_0 \geq 0$ **ή** **(Π1):** $T - (T_1+\ldots+T_N) \geq (S_1+\ldots+S_N)$

Μεγέθη παρτίδων παραγωγής (συνέχεια)

Η παραγωγή $Q_i = T_i P_i$ στο υποδιάστημα T_i ισούται με τη ζήτηση TD_i σε όλο το T. Δηλαδή $Q_i = T_i P_i = TD_i$, κι έτσι από το **T προκύπτουν όλα τα T_i, Q_i**.

Το $\rho_i = D_i/P_i = T_i/T$ είναι το κλάσμα του χρόνου T για παραγωγή του i. Για κάθε i ισχύει το Μοντέλο II (b_i=0, $P_i<\infty$). Μέσο κόστος ανά μονάδα χρόνου:

$$K(T) = \sum_{i=1}^{N}\left[\frac{A_i D_i}{Q_i} + C_i D_i + h_i \frac{Q_i}{2}(1-\rho_i)\right] \overset{Q_i = TD_i}{=} \sum_{i=1}^{N}\left[\frac{A_i}{T} + C_i D_i + h_i \frac{TD_i}{2}(1-\rho_i)\right]$$

Το πρόβλημα γράφεται: **min K(T), υπό τον περιορισμό ανισότητας** (Π1).

Στο προηγούμενο μάθημα είδαμε μία διαδικασία επίλυσης:

Επιλύουμε το πρόβλημα χωρίς την ανισότητα.

Αφού δεν έχουμε άλλους περιορισμούς και η K(T) είναι κυρτή, η K(T) έχει ολικό ελάχιστο στο $T = T^*$:

$$\frac{\partial K}{\partial T} = 0 \Rightarrow T^* = \sqrt{2\sum_{i=1}^{N} A_i \Big/ \sum_{i=1}^{N} h_i D_i (1-\rho_i)}$$

Ελέγχουμε αν η T^* ικανοποιεί τον (Π1): $T^* - (T_1+\ldots+T_N) \geq (S_1+\ldots+S_N)$

$\Rightarrow T^* - T^*(\rho_1+\ldots+\rho_N) \geq (S_1+\ldots+S_N)$. Το δεξί μέλος είναι θετικό. Πρέπει:

(i) $\boldsymbol{\rho} = \rho_1+\ldots+\rho_N < 1$ (ρ = κλάσμα του χρόνου που η μηχανή δουλεύει)

(ii) $T^* \geq (S_1+\ldots+S_N)/(1-\rho) = \mathbf{T_{min}}$.

Αν ισχύουν, τότε η βέλτιστη λύση είναι $T_{opt} = T^*$ με $T_0 = (T^*-T_{min})(1-\rho)$.

Αν ισχύει η (i) αλλά όχι η (ii), τότε $T = T_{min}$ και $T_0 = 0$.

Αν δεν ισχύει η (i) είναι αδύνατο να ικανοποιηθεί όλη ζήτηση.

Παράδειγμα (διαφορετικό από των σημειώσεων)

Προϊόν i	D_i	P_i	A_i	S_i	C_i	h_i=0.2C_i
A	3000	10000	50	0.001	10	?
B	3000	5000	70	0.002	15	?
C	5000	∞	120	0.005	5	?
Αθροίσματα			**240**	**0.008**		

Προϊόν i	ρ_i=D_i/P_i	h_i=0.2C_i	$h_i D_i(1-\rho_i)$
A	**0.3**	**2**	**4200**
B	**0.6**	**3**	**3600**
C	**0**	**1**	**5000**
Αθροίσματα	**ρ = 0.9**		**12800**

$$T^* = \sqrt{2\sum_{i=1}^{N} A_i \Big/ \sum_{i=1}^{N} h_i D_i (1-\rho_i)} = \sqrt{480/12800} = 0.19365$$

$T_{min} = (S_1+\ldots+S_N)/(1-\rho) = 0.008/0.1 = 0.08$. Ικανοποιείται ο (Π1).

Συνεπώς, δεχόμαστε την $T^* = 0.19365$. Από αυτήν υπολογίζουμε $Q_i = T^* D_i$ και $T_i = T^*\rho_i$. Συμφέρει χρόνος αδράνειας $T_0 = (T^*-T_{min})(1-\rho) = 0.011365$. Αφού ο ρυθμός P_C είναι άπειρος, έχουμε ρ_C=0 και T_C=0.

Αν ο χρόνος είναι σε μήνες, $T^* \approx 0.2(30) = 6$ ημέρες και $T_0 \approx 0.34$ ημέρες ή 8 ώρες.

Ελαχιστοποίηση χρόνου προετοιμασιών με Δυναμικό Προγραμματισμό

Έστω ότι, αν μετά το k ακολουθεί το i, η προετοιμασία εξαρτάται από k και i: $S_i = d_{ki}$. *Παραγωγή λιπαντικών:* $d_{γράσο \to λεπτό\ λάδι} >> d_{απλό\ λάδι \to λεπτό\ λάδι}$

Ερώτημα: Ποιο **κυκλικό πρόγραμμα** με N εργασίες έχει **min** $S_1+\ldots+S_N$;
Πρόβλημα περιπλανώμενου πωλητή: Η συντομότερη **κυκλική διαδρομή** μέσω πόλεων 1,…,N.

Δυναμικός προγραμματισμός (ΔΠ, dynamic programming)

Προβλήματα ελαχιστοποίησης με **μία** μεταβλητή λύνονται πιο **εύκολα**.
Η αρχή του **ΔΠ** εφαρμόζεται στην ελαχιστοποίηση **αθροιστικών** συναρτήσεων, όπως η $K(x_0,\ldots,x_N) = f_0(x_0)+\ldots+f_N(x_N)$, ... ή και πιο σύνθετων: $K(x_0,\ldots,x_N) = f_0(Y_0,x_0)+f_1(Y_1,x_1)+\ldots+ f_N(Y_N,x_N)$, για Y_0 γνωστό και $Y_{n+1}= g(Y_n,x_n)$.

Ελάχιστο κόστος με f_n ως f_N ξεκινώντας από Y_n=y: $\mathbf{C_n(y) = \min (f_n+\ldots+f_N)}$.

Εξισώσεις ΔΠ
n = N Για κάθε πιθανή $y=Y_N$, βρες το x_N ώστε $C_N(y) = \min f_N(y,x_N)$.
n < N Για κάθε πιθανή y, βρες x_n ώστε $C_n(y) = \min [f_n(y,x_n)+C_{n+1}(g(y,x_n))]$.
n = 0 Μόνο για Y_0, βρες x_0 ώστε $C_0(Y_0) = \min [f_0(Y_0,x_0) + C_1(g(Y_0,x_0))]$.

Ελαχιστοποίηση χρόνου προετοιμασιών με ΔΠ (συνέχεια)

Ζητείται ένα κυκλικό πρόγραμμα παραγωγής που **ξεκινά** χωρίς προετοιμασία με το προϊόν N, παράγει τα **υπόλοιπα προϊόντα με τη βέλτιστη σειρά** (αυτήν ζητούμε!) και προετοιμάζεται για το N.
Σχηματικά: N → σύνολο {1, 2, …, N–1} με βέλτιστη σειρά → N(*)

Ορίζουμε C(i, S) = (άγνωστο) ελάχιστο κόστος για N → {σύνολο S} → i

Αλγόριθμος ΔΠ
S = ∅: $C(i,\varnothing) = d_{Ni}$ για i = 1,2,…,N–1
S με στοιχείο j: $C(i,\{j\}) = C(j,\varnothing) + d_{ji}$ για κάθε j = 1,2,…,N–1 και κάθε i ≠ j.

κάθε συνδυασμός S με n προϊόντα : $C(i,S) = \min_{\substack{\text{j: κάθε δυνατή τελευταία} \\ \text{εργασία του συνόλου S}}} \left[C(j,S-\{j\}) + d_{ji}\right]$

Τελικό S = {1,2,…,N–1}: $C(N,S) = \min_j [C(j,S-\{j\}) + d_{ji}]$ για κάθε j = 1,2,…,N–1.

Ο συντομότερος χρόνος προετοιμασιών είναι ο τελευταίος, C(N,S).
Σε κάθε πρόβλημα min C(i,S) σημειώνουμε το j→i που συμβαίνει το min.
Ξεκινώντας από το τέλος προς την αρχή βρίσκουμε το βέλτιστο πρόγραμμα.

(*) Στις σημειώσεις παρουσιάζεται το ισοδύναμο πρόβλημα: 1 → {2, …, N} → 1

Παράδειγμα (διαφορετικό από των σημειώσεων)

d_{ij}	**1**	**2**	**3**	**4**
1	-	3	0	5
2	6	-	7	8
3	0	12	-	8
4	3	4	1	-

0 ενδιάμεσες εργασίες:

$C(1,\varnothing) = d_{41} = 3 \Rightarrow 4\rightarrow1$

$C(2,\varnothing) = d_{42} = 4 \Rightarrow 4\rightarrow2$

$C(3,\varnothing) = d_{43} = 1 \Rightarrow 4\rightarrow3$

1 ενδιάμεση εργασία:

$C(2,\{1\}) = C(1,\varnothing)+d_{12} = 3+3 = 6 \Rightarrow 1\rightarrow2$

$C(3,\{1\}) = C(1,\varnothing)+d_{13} = 3+0 = 3 \Rightarrow 1\rightarrow3$

$C(1,\{2\}) = C(2,\varnothing)+d_{21} = 4+6 = 10 \Rightarrow 2\rightarrow1$

$C(3,\{2\}) = C(2,\varnothing)+d_{23} = 4+7 = 11 \Rightarrow 2\rightarrow3$

$C(1,\{3\}) = C(3,\varnothing)+ d_{31} = 1+0 = 1 \Rightarrow 3\rightarrow1$

$C(2,\{3\}) = C(3,\varnothing)+ d_{32} = 1+12 = 13 \Rightarrow 3\rightarrow2$

2 ενδιάμεσες εργασίες:

$C(3,\{1,2\}) = \min[C(2,\{1\})+d_{23}, C(1,\{2\})+d_{13}] = \min[6+7, 10+0] = 10 \Rightarrow 1\rightarrow3$

$C(2,\{1, 3\}) = \min[C(3,\{1\})+d_{32}, C(1,\{3\})+d_{12}] = \min[3+12, 1+3] = 4 \Rightarrow 1\rightarrow2$

$C(1,\{2, 3\}) = \min[C(3,\{2\})+d_{31}, C(2,\{3\})+d_{24}] = \min[11+0, 13+8] = 11 \Rightarrow 3\rightarrow1$

3 ενδιάμεσες εργασίες:

$C(4,\{1, 2, 3\}) = \min[C(3,\{1, 2\})+d_{34}, C(2,\{1, 3\})+d_{24}, C(1,\{2, 3\})+d_{14}]$

$= \min[10+8, 4+8, 11+5] = \mathbf{12} \Rightarrow 2\rightarrow4$. Μικρότερος χρόνος: **12**

Βέλτιστες αλλαγές: $2\rightarrow4, 1\rightarrow2, 3\rightarrow1, 4\rightarrow3$. Πρόγραμμα $4\rightarrow3\rightarrow1\rightarrow2\rightarrow4$.