ΚΕΦΑΛΑΙΟ 4: ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Εισαγωγή

Ένα σύστημα παραγωγής χρησιμοποιεί **πόρους** για να εκτελέσει **εργασίες** μέσω των οποίων παράγει προϊόντα ή υπηρεσίες.

Εργασία: δραστηριότητα που **δεσμεύει πόρους** για κάποιον **χρόνο**.

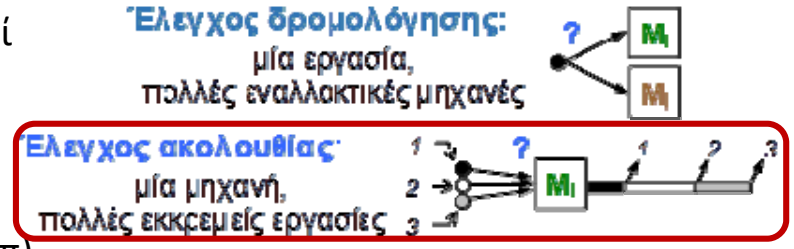
Παραδείγματα εργασιών: σχεδίαση νέου προϊόντος, κατεργασίες κομματιών, πακετάρισμα, βαφή, συντήρηση εξοπλισμού - επισκευή βλαβών, προετοιμασία των μηχανών, εργασίες καθαριότητας, ...

Πόρος: i) απαραίτητος για να εκτελεσθεί μία εργασία (ή ομάδα εργασιών)
ii) κατά τον **χρόνο** χρήσης, γίνεται **μόνο** η συγκεκριμένη εργασία

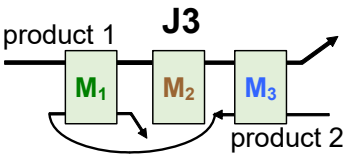
Κατηγορίες πόρων: ●εξαντλούμενοι (πρώτη ύλη, συστατικό προϊόντος)
●σταθεροί (μηχανές, εργαζόμενοι)

Προγραμματισμός παραγωγής: **Πότε** πρέπει να ξεκινήσει η χρήση **ποιων πόρων** για **πόσο χρόνο** και για **ποια εργασία**,

- ούτως ώστε να ελαχιστοποιηθεί το **κόστος** παραγωγής
- και να τηρηθούν τυχόν **περιορισμοί** (π.χ. αναγκαίοι χρόνοι παραγωγής, να παραδοθούν οι παραγγελίες κλπ)

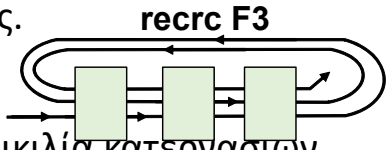


Γεωμετρίες συστημάτων παραγωγής



Κατάστημα εργασιών Jm (jobshop): Διαφορετικά προϊόντα, διαφορετικά στάδια παραγωγής (φασεολόγια).

Κατάστημα ροής, γραμμή παραγωγής Fm (flowshop, production line): Προϊόντα έχουν ίδια στάδια παραγωγής.



●Μπορεί να γίνονται πολλές επισκέψεις σε κάποιες μηχανές (πχ. γραμμή με επανεισόδους=recrc Fm)
●Ευελιξία-παράλληλες μηχανές: Pm ●Μία μηχανή με ποικιλία κατεργασιών.

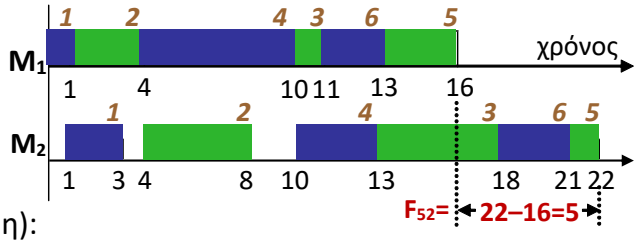
Προβλήματα α|β|Z. **α**=σύστημα (1,Fm,...) | **β**=εργασίες (r_i:χρόνοι άφιξης >0, prmp:επιτρέπονται διακοπές εργασιών,...) | **Z**=κόστος που θέλουμε min

Προσομοίωση α|β|Z. **Δεδομένα:** n εργασίες, r_i=χρόνος άφιξης εργασίας i, p_{ij}=διάρκεια της i στη M_j. **Σειρά εκτέλεσης i. Υπολογισμοί:** ●C_{ij}=**Χρόνος περάτωσης** της i στη M_j ●**Συνολικός χρόνος περάτωσης** C_i ●**Χρόνος ροής** $F_{ij}=C_{ij}$ –πότε έφθασε=p_{ij}+αναμονή ●**Μέσος χρόνος ροής** στη $M_j=(F_{1j}+...+F_{nj})/n$

Το διάγραμμα Gantt. Έστω το **F2||C_{max}**, όπου $r_i=0$ και C_{max}=χρόνος τελευταίας i.

Πρόγ/μα i	*1*	*2*	*4*	*3*	*6*	*5*
M_1: p_{i1}	1	3	6	1	2	3
M_2: p_{i2}	2	4	3	5	3	1

$C_{max}=C_{52}=22$



Μέσοι χρόνοι ροής:

Στη M_1: $\bar{F}_1=(1+4+10+11+13+16)/6=9.16$

Στη M_2 (μαζί με αναμονή, αν η M_2 δεν είναι ελεύθερη):

$\bar{F}_2=[(3-1)+...+(18-11)+(21-13)+(22-16)]/6=5$ **Στο σύστημα**: $\bar{F}=(3+8+...+22)/6=14.17$

Προβλήματα $1||\Sigma F_i$, $1||\Sigma w_i F_i$ και $1|r_i,prmp|\Sigma F_i$

Μία μηχανή πρόκειται να εκτελέσει n εργασίες, με διάρκειες p_i, i=1,…,n. Θα εύρουμε προγράμματα εκτέλεσης που ελαχιστοποιούν κάποιες συναρτήσεις κόστους που συνδέονται με τους **χρόνους ροής**.

$1||\Sigma F_i$: Εδώ $r_i=0$ και $F_i=C_i-r_i=C_i$. Έστω το πρόγραμμα (1,2,...,n). Τότε

$F_1=p_1$
$F_2=p_1+p_2$
…
$F_n=p_1+p_2+\ldots+p_{n-1}+p_n$

$$\sum_{i=1}^{n} F_i = np_1+(n-1)p_2+\ldots+2p_{n-1}+1p_n$$

***Θεώρημα**: Για δύο ακολουθίες μη αρνητικών αριθμών $\alpha_i\geq 0$ και $\beta_i\geq 0$, i=1,…,n, το **εσωτερικό γινόμενο** $\alpha_1\beta_1+\ldots+\alpha_n\beta_n$ **γίνεται ελάχιστο** αν οποιαδήποτε από τις δύο διαταχθεί κατά φθίνουσα τάξη ↓ και η άλλη κατ' αύξουσα ↑ (γίνεται μέγιστο όταν οι ακολουθίες έχουν όμοια διάταξη).*

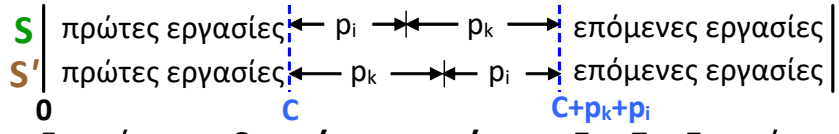
Στο ΣF_i η ακολουθία n,n–1,...,1 είναι φθίνουσα. Άρα το ελάχιστο ΣF_i (και ο μέσος χρόνος ροής $\Sigma F_i/n$) επιτυγχάνεται με τον **κανόνα συντομότερης διάρκειας** ή SPT (shortest processing time): $\mathbf{p_1 \leq p_2 \ldots \leq p_{n-1} \leq p_n}$

Παράδειγμα: Έχετε καθυστερήσει την παράδοση τριών μελετών. Για να τις τελειώσετε θέλετε 13, 6, και 9 ημέρες. Κάθε ημέρα καθυστέρησης κοστίζει 100 ευρώ ανά μελέτη. Με ποια σειρά πρέπει να τις κάνετε;

Λύση *(α) Διερεύνηση:* Για σειρά (1,2,3), κόστος= $100p_1+100(p_1+p_2)+100(p_1+p_2+p_3)=100(3p_1+2p_2+1p_3)$. Άρα πρόκειται για **ίδιο** πρόβλημα με το $1||\Sigma F_i$.
(β) Εφαρμογή: **SPT:6≤9≤13** και, με την αρχική αρίθμηση, **2-3-1**.

$1||\Sigma w_i F_i$: Τώρα ο χρόνος ροής F_i έχει κόστος $w_i\geq 0$ ανά μονάδα χρόνου.

Η μέθοδος απόδειξης με αντιμετάθεση (interchange argument):
Επιλέγουμε **αυθαίρετα** ένα πρόγραμμα S και από αυτό φτιάχνουμε ένα όμοιο S′, εκτός από δύο διαδοχικές εργασίες του S, έστω (i, k), οι οποίες στο πρόγραμμα **S′ αντιμετατίθενται** σε (k, i). Έστω C η στιγμή όπου ξεκινά η i στο S και η k στο S′:



(A) Ανάλυση: **Αυθαίρετα** ζητούμε **το S να έχει μικρότερο** $F_w=\Sigma w_i F_i$ από το S′ και βρίσκουμε την συνθήκη που πρέπει να ισχύει. Πρέπει

$F_w(S)$=(πρώτες εργασίες)+ $w_i(C+p_i)+w_k(C+p_i+p_k)$+(τελευταίες εργασίες)
$\leq F_w(S')$=(πρώτες εργασίες)+ $w_k(C+p_k)+w_i(C+p_k+p_i)$+(τελευταίες εργασίες)

Με απλοποιήσεις προκύπτει $w_kp_i \leq w_ip_k \Leftrightarrow p_i/w_i \leq p_k/w_k$ (1)

(B) Σύνθεση: Το συμπέρασμα **(A)** διατυπώνεται **και ως εξής**: "Όποιο πρόγραμμα **S′** έχει δύο διαδοχικές εργασίες που παραβιάζουν την (1), μπορεί να βελτιωθεί". Συνεπώς, στο βέλτιστο πρόγραμμα η (1) ισχύει παντού και γίνεται ο **κανόνας συντομότερης διάρκειας με βάρη** (WSPT):

$$(1,2,\ldots,n) \Leftrightarrow \frac{p_1}{w_1} \leq \frac{p_2}{w_2} \leq \ldots \leq \frac{p_n}{w_n}$$

Παράδειγμα: Στις μελέτες με χρόνους 13, 6 και 9, αν το αντίστοιχο κόστος υπερημερίας είναι 260, 100 και 180, τότε p_i/w_i: 13/260=0.05, 6/100=0.06 και 9/180=0.05. Η βέλτιστη σειρά είναι 1-3-2 **ή** 3-1-2 (έχουν ίδιο F_w).

Αφαιρούνται από τις σημειώσεις Φεβ. 2020 και την ύλη το πρόβλημα **1|r_i|ΣF_i** μαζί με το αντίστοιχο λυμένο παράδειγμα εκεί. **Αντικαθίστανται από τα παρακάτω:**

1|r_i,prmpt|ΣF_i: Κάποιες εργασίες δεν είναι διαθέσιμες στην αρχή αλλά φθάνουν σε χρόνους r_i γνωστούς ή άγνωστους. Έχουμε $F_i=C_i-r_i$. **Επίσης:**

α) Όταν φθάνει μία εργασία, **επιτρέπεται** η μηχανή **να διακόψει** την εργασία που εκτελούσε και να ξεκινήσει αυτήν που μόλις έφθασε.

β) Η διακοπείσα εργασία θα συνεχισθεί αργότερα από το σημείο που διακόπηκε και ο χρόνος μέχρι να ολοκληρωθεί ισούται με τον **υπόλοιπο χρόνο τη στιγμή της διακοπής** (preempt-resume).

Βέλτιστη πολιτική: Όταν **φθάνει** νέα εργασία ή **ολοκληρώνεται** κάποια και εκκρεμούν άλλες, τότε επιλέγεται η εργασία με τον **συντομότερο υπόλοιπο χρόνο παραγωγής** (shortest remaining processing time, **SRPT**).

Παράδειγμα: Στα αριστερά είναι τα δεδομένα και κάτω τα βήματα επίλυσης.

Δεδομένα για 4 εργασίες

Κάθε στήλη *t=0,1,... αντιστοιχεί στο διάστημα [t,t+1). Η στήλη 0 είναι το διάστημα από 0 ως λίγο πριν το 1.* ***Κάθε κελί*** *αντιστοιχεί σε μία εργασία i και ένα [t,t+1).*

1. Ξεκινάμε *χρωματίζοντας απλώς τα κελιά που γίνονται αφίξεις (γαλάζιο).*

2. *Κάθε στιγμή t=0,1,...* ***βάζουμε τα ακόλουθα σημάδια στα κελιά της στήλης t****:*

α) *Αν η εργασία i δεν έχει έλθει ή αν η i έχει τελειώσει, το κελί της είναι κενό. Αλλιώς αν είναι παρούσα,* ***σημειώνουμε στο κελί τον υπόλοιπο χρόνο της****.*

β) Επιλέγουμε *με SRPT ποια εργασία εκτελείται στο [t,t+1) και βάζουμε ένα ↔.*

i	r_i	p_i	t:	0	1	2	3	4	5	6	7	8	9	10	11	12
1	1	2			2 ↔	1 ↔	**Τέλος**									
2	6	3								3	3 ↔	2	2 ↔	1 ↔	**Τέλος**	
3	0	5		5 ↔	4	4	4 ↔	3 ↔	2 ↔	1 ↔	**Τέλος**					
4	8	1										1 ↔	**Τέλος**			

Τελικά: $\sum F_i = (3-1)+(11-6)+(7-0)+(9-8) = 15$ και $\overline{F} = 15/4 = 3.75$