$\alpha \mid \beta \mid \gamma$ Notation

In der Literatur hat sich die 3-Parameter-Notation $\alpha \mid \beta \mid \gamma$ für deterministische Scheduling-Probleme durchgesetzt, wobei

α die **Maschinenumgebung** beschreibt,

β die **Jobcharakteristika** sowie weitere Nebenbedingungen angibt und

γ die **Zielfunktion** (Optimierungskriterium) charakterisiert.

Maschinenumgebung

Der Parameter α setzt sich aus dem Tupel $\alpha_1\alpha_2$ zusammen, wobei gilt:

$\alpha_1 \in \{\circ, P, Q, R, O, J, F\}$ ($\circ$ steht für 'keine Bedingung')

Einmaschinenprobleme

- $\alpha_1 = \circ$: Jeder Auftrag besteht aus *einer* Operation, wobei nur *eine* Maschine zur Verfügung steht, damit gilt $p_{i1} = p_i$.

Parallelmaschinenprobleme

Jeder Auftrag besteht aus *einer* Operation, die auf einer *beliebigen* Maschine durchgeführt werden kann, insbesondere:

- $\alpha_1 = P$: Die Bearbeitung erfolgt auf genau einer von m **identischen parallelen Maschinen**, d. h. $p_{ij} = p_i$.
- $\alpha_1 = Q$: m **uniforme parallele Maschinen** stehen zur Auswahl, jede Maschine M_j hat die *Geschwindigkeit* s_j, es ergibt sich: $p_{ij} = p_i/s_j$.
- $\alpha_1 = R$: m **heterogene parallele Maschinen** stehen zur Bearbeitung zur Verfügung. Die *Geschwindigkeit* s_{ij} zur Abarbeitung der Operation (i, j) ist sowohl job- als auch maschinenabhängig. Es gilt: $p_{ij} = p_i/s_{ij}$.

Shop-Probleme

$\alpha_1 \in \{O, F, J\}$: Jeder Auftrag wird auf jeder Maschine höchstens einmal bearbeitet.

- $\alpha_1 = O$: **Open-Shop Problem**: die technologischen und organisatorischen Reihenfolgen sind beliebig wählbar.
- $\alpha_1 = J$: **Job-Shop Problem**: die technologischen Reihenfolgen sind fest vorgegeben, die organisatorischen Reihenfolgen sind zu bestimmen.
- $\alpha_1 = F$: **Flow-Shop Problem**: die technologischen Reihenfolgen sind fest vorgegeben und für alle Aufträge identisch: $M_1 \rightarrow M_2 \rightarrow \ldots \rightarrow M_m$, die organisatorischen Reihenfolgen sind zu bestimmen.

Für Job-Shop Probleme wird oft zugelassen, dass ein Auftrag mehrfach auf einer Maschine zu bearbeiten ist (**recirculation**).

$\alpha_2 \in \{\circ, c\}$, $(\alpha_1 = \circ \Longleftrightarrow \alpha_2 = 1)$:

- $\alpha_2 = c$: Die Anzahl der Maschinen m ist **konstant** $(m = c)$.
- $\alpha_2 = \circ$: Die Anzahl der Maschinen ist **variabel** und Bestandteil der Problemgröße.

Jobcharakteristika und Nebenbedingungen

β enthält keinen, einen oder mehrere Parameter der Menge $\{\beta_1, \ldots \beta_5\}$.

$\beta_1 \in \{\circ, pmtn\}$:

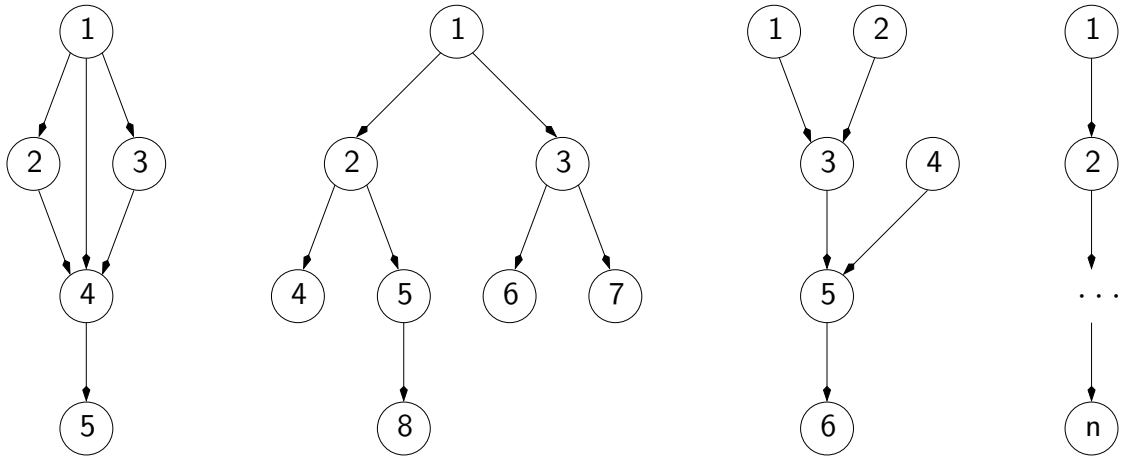
- $\beta_1 = pmtn$: **Preemption** (Unterbrechung) ist erlaubt, d. h. die Bearbeitung eines Auftrags auf einer Maschine darf jederzeit unterbrochen und zu einem späteren Zeitpunkt fortgesetzt werden.
- $\beta_1 = \circ$: Preemption ist nicht erlaubt.

$\beta_2 \in \{\circ, prec, tree, outtree, intree, chain\}$:

Zwischen den Aufträgen gibt es **Vorrangbedingungen**: $i \rightarrow k$ heißt, dass Auftrag A_i vollständig bearbeitet sein muss, bevor Auftrag A_k starten kann. Die Vorrangbedingungen sind durch einen zyklenfreien Digraphen gegeben.

Vorrangbedingungen

- $\beta_2 = prec$: Die Vorrangbedingungen sind beliebig.
- $\beta_2 = outtree$: Jeder Auftrag A_i hat höchstens einen Vorgänger.
- $\beta_2 = intree$: Jeder Auftrag A_i hat höchstens einen Nachfolger.
- $\beta_2 = tree$: Es gilt entweder $intree$ oder $outtree$.
- $\beta_2 = chain$: Die Vorrangbedingungen formen einen Weg (oder mehrere Wege).
- $\beta_2 = \circ$: Es existieren keine Vorrangbedingungen.



Release Dates

$\beta_3 \in \{r_i \geq 0, \circ\}$:

- $\beta_3 = r_i \geq 0$: Jeder Auftrag A_i hat einen vorgegebenen **Bereitstellungstermin** $r_i \geq 0$.
- $\beta_3 = \circ$: Es gilt $r_i = 0$ für alle $i \in I$. Alle Aufträge stehen zum Zeitpunkt 0 zur Verfügung.

Due Dates

$\beta_4 \in \{d_i, \circ\}$:

- $\beta_4 = d_i$: Jeder Auftrag A_i darf nicht später als zum vorgegebenen **Fälligkeitstermin** $d_i \geq 0$ beendet sein.
- $\beta_4 = \circ$: Es gibt keine vorgegebenen Fälligkeitstermine.

Bearbeitungszeiten

$\beta_5 \in \{p_{ij} = 1, p_{ij} = p, \underline{p} \leq p_{ij} \leq \overline{p}, \circ\}$:

- $\beta_5 = p_{ij} = 1$ für $\alpha_1 \in \{O, F, J\}$: Es gilt $p_{ij} = 1$ für alle $(i, j) \in SIJ$

  (analog: $\beta_5 = p_i = 1$ für $\alpha_1 \in \{\circ, P, Q\}$: Jeder Auftrag A_i hat die Bearbeitungszeit $p_i = 1$).

- $\beta_5 = p_{ij} = p$: Die Bearbeitungszeiten aller Operationen $(i, j) \in I \times J$ sind **konstant**.

- $\beta_5 = \underline{p} \leq p_{ij} \leq \overline{p}$: Es sind nur **ganzzahlige** Werte aus dem Interval $[\underline{p}, \overline{p}]$ für alle $(i, j) \in I \times J$ zugelassen.

- $\beta_5 = \circ$: $p_{ij} \geq 0$ und **ganzzahlig** für alle $(i, j) \in I \times J$.

Mögliche weitere Nebenbedingungen (1)

- *non-delay*: Es sind nur **non-delay Schedules** erlaubt.
- *no-wait*: **Wartezeiten** der Aufträge zwischen zwei aufeinanderfolgenden Operationen sind verboten.
- *no-idle*: **Stillstandszeiten** der Maschinen sind verboten (d. h. wenn eine Maschine startet, muss sie die Aufträge hintereinander bearbeiten).
- *recrc* (**recirculation**): Die **mehrmalige** Bearbeitung eines Auftrags A_i auf derselben Maschine M_j ist möglich.
- *prmu* (**permutation**): Es werden nur **Permutationspläne** zugelassen (d.h. auf allen Maschinen ist die gleiche organisatorische Reihenfolge der Aufträge zu wählen).

Mögliche weitere Nebenbedingungen (2)

- $n = k$ – die Anzahl der Aufträge ist **konstant** und gleich k
- $n_i \leq k$ – die Anzahl der Operationen je Auftrag A_i ist **höchstens gleich** k
- ST_{si} – es treten **reihenfolgeunabhängige Setup-Zeiten** auf (können sowohl bei Batching-Problemen als auch bei Problemen ohne Batching zu berücksichtigen sein)
- ST_{sd} – es treten **reihenfolgeabhängige Setup-Zeiten** auf (können sowohl bei Batching-Problemen als auch bei Problemen ohne Batching zu berücksichtigen sein)
- res – es gelten **Ressourcenbeschränkungen** (z.B. Verbrauch an Energie, Bedarf an Arbeitskräften usw.; es existieren verschiedene Typen)

Zielfunktion (Optimierungskriterium)

γ charakterisiert eine Zielfunktion (Optimierungskriterium) aus $\{f_{max}, \sum f_i\}$.

Für einen gegebenen Schedule lässt sich für jeden Auftrag A_i angeben:

eine **Fertigstellungszeit:**	C_i	(completion time)
eine **Terminabweichung:**	$L_i = C_i - d_i$	(lateness)
eine **Verspätung:**	$T_i = \max\{0, C_i - d_i\}$	(tardiness)
ob A_i verspätet ist:	$U_i = \begin{cases} 0, & \text{wenn } C_i \leq d_i \\ 1, & \text{sonst} \end{cases}$	(unit penalty)

$$f_{max} \in \{C_{max}, L_{max}\}$$

Gesamtbearbeitungszeit oder Makespan: $C_{max} = \max_{i \in I}\{C_i\} \rightarrow \min!$

maximale Terminabweichung: $L_{max} = \max_{i \in I}\{L_i\} \rightarrow \min!$

Summenzielfunktionen

$$\sum f_i \in \left\{ \sum C_i,\ \sum T_i,\ \sum U_i,\ \sum w_i C_i,\ \sum w_i T_i,\ \sum w_i U_i \right\},$$

wobei $w_i \in \mathbb{N}$ ein **Gewicht** für den Auftrag A_i ist.

- $\sum C_i = \sum_{i \in I} C_i \to \min!$
- $\sum w_i T_i = \sum_{i \in I} w_i T_i \to \min!$
- $\sum w_i U_i = \sum_{i \in I} w_i U_i \to \min!$

. . .

Eigenschaften der Zielfunktionen

- $\sum w_i C_i$ und $\sum w_i L_i$ unterscheiden sich nur durch die Konstante $\sum w_i d_i$.
- Jeder Schedule, der optimal für L_{max} ist, ist auch gleichzeitig optimal für T_{max} und U_{max} (*Hinweis:* die Umkehrung gilt nicht!).
- Alle hier dargestellten Optimierungskriterien

$$F(C_1, \ldots, C_n) \rightarrow \min!$$

sind **regulär**, d. h. F ist nichtfallend in den Terminen $C_1, \ldots, C_n$:

$$\text{mit } C_i \leq C_i^* \text{ für alle } i \in I \text{ folgt } F(C_1, \ldots, C_n) \leq F(C_1^*, \ldots, C_n^*).$$

- Beispiel für ein nichtreguläres Kriterium:

$$\sum \mid C_i - d_i \mid = \sum_{i \in I} \mid C_i - d_i \mid \rightarrow \min!$$

Weitere Bedingungen

Darüber hinaus gibt es in der Literatur eine Vielzahl weiterer Bedingungen. Beispielsweise ist ein **flexibles** (oder **hybrides**) **Flow-Shop Problem** (FFS) eine Kombination von einem Flow-Shop und einem Parallel-Shop Problem:

$$\left\{ \begin{array}{c} M_1^1 \\ M_2^1 \\ \vdots \\ M_{i_1}^1 \end{array} \right\} \longrightarrow \left\{ \begin{array}{c} M_1^2 \\ M_2^2 \\ \vdots \\ M_{i_2}^2 \end{array} \right\} \longrightarrow \quad \cdots \quad \longrightarrow \left\{ \begin{array}{c} M_1^s \\ M_2^s \\ \vdots \\ M_{i_s}^s \end{array} \right\}$$

Unter einem **Mixed-Shop Problem** versteht man eine Kombination von einem Job-Shop und einem Open-Shop Problem.

Weiter unterscheidet man u.a. **ressourcenbeschränkte**, **stochastische**, **periodische** oder **zyklische Scheduling-Probleme**, **Batching-Probleme** oder **Probleme mit Intervallbearbeitungszeiten** (Probleme mit Uncertainty).