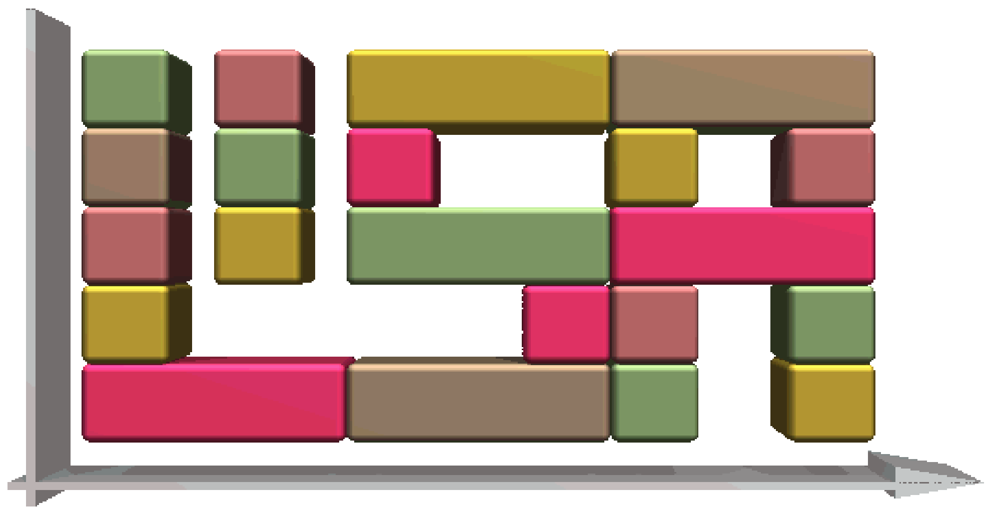
Scheduling-Theorie

Mathematische Modelle und Methoden für deterministische Scheduling-Probleme



LiSA - A Library of Scheduling Algorithms

Scheduling-Probleme in Fertigung und Service

(a) Fertigung

- Projektplanung und -scheduling (z.B. CPM und PERT)
- **Maschinenbelegung (Job Scheduling)**
- Fließbandabstimmung
- Losgrößenplanung (z. B. Economic Lot Scheduling Problem)
- Planung und Scheduling in Supply Chains

(b) Service

- Intervall-Scheduling, Reservierungen und Timetabling (z.B. Stundenplanung an Schulen)
- Scheduling und Timetabling in Sport und Unterhaltung (z.B. Planung von Sportwettkämpfen)
- Planung, Scheduling und Timetabling im Transport (z.B. Tanker-Scheduling, Fahrplangestaltung bei der Eisenbahn)
- Personaleinsatzplanung (Workforce Scheduling)

Scheduling-Probleme

deterministische Scheduling-Probleme:

alle Daten zu den Aufträgen sind deterministisch fixiert

(a) **offline Scheduling:**

alle Daten zu den Aufträgen sind *a priori* bekannt

(b) *online Scheduling:*

die Daten zu den Aufträgen sind *nicht a priori* bekannt (Daten eines Auftrags werden erst zum Zeitpunkt seiner Ankunft bekannt)

stochastische Scheduling-Probleme:

Zufallsvariable treten auf (z.B. Bearbeitungszeiten)

Scheduling-Probleme: Beispiele

Beispiel 1. Ein kleines Unternehmen produziert Papiertüten für Zement, Hundefutter u. ä.. Das Basismaterial besteht aus Papierrollen. Diese werden in drei Abteilungen bearbeitet, die folgende Arbeitsgänge ausführen:

1. Bedrucken des Papiers, 2. Leimen der Seitenränder, 3. Zunähen des Bodens.

In jeder Abteilung gibt es eine Anzahl von Maschinen, die sich z.B. in der Arbeitsgeschwindigkeit, der Anzahl der Farben beim Drucken oder auch durch die Größe des Papiers, das sie bearbeiten können, unterscheiden. Jede Zuordnung der eingehenden Aufträge zu den Maschinen erzeugt eine bestimmte Menge Tüten in einer bestimmten Zeit. Die Bearbeitungszeit der Operationen sei dabei proportional zu der Menge der erzeugten Tüten.

Wird ein Auftrag verspätet fertig, so entstehen Strafkosten, die von der Wichtigkeit des Auftrags und der Größe der Verspätung abhängen. Die Maschinen haben reihenfolgeabhängige Rüstzeiten (Setup-Zeiten), d. h. die Umrüstzeit einer Maschine vor Bearbeitung eines Auftrags hängt von diesem Auftrag und dem zuvor auf dieser Maschine bearbeiteten Auftrag ab. Unter diesen Bedingungen ist die Produktion so zu steuern, dass die Summe der Strafkosten minimiert wird.

Scheduling-Probleme: Beispiele

Beispiel 2. Eine Funktion eines Multiprozessorcomputersystems besteht darin, die Zeit für die zu bearbeitenden Programme für den Zentralprozessor ‘einzuteilen’. Die exakte Zeit für ein Programm ist gewöhnlich nicht bekannt. Es kann jedoch die Verteilung der zufälligen Bearbeitungszeiten einschließlich Erwartungswert und Varianz bekannt sein.

Zusätzlich hat jedes Programm einen Prioritätsfaktor (ein Gewicht), das vom Nutzer oder Operator vorgegeben wird.

Ziel ist es, eine Reihenfolge der Bearbeitung der Programme zu finden, so dass die mittlere Summe der gewichteten Bearbeitungsendzeitpunkte minimiert wird. Hierbei kann es erlaubt sein, länger laufende Programme in mehrere Teile zu splitten (Preemption). Damit wird verhindert, dass relativ kurze Programme zu lange im System sind. Man kann sich leicht überlegen, dass bei Hinzunahme der Bedingung ‘Unterbrechung erlaubt’ der optimale Zielfunktionswert kleiner werden kann.

Scheduling-Probleme: Beispiele

Beispiel 3. Für die Vorlesung Scheduling-Theorie gibt es drei Bücher, die sechs besonders interessierte Studenten als Prüfungsvorbereitung durcharbeiten wollen. Jeder Student kann gleichzeitig höchstens ein Buch lesen und auch jedes Buch kann zu einem festen Zeitpunkt nur von höchstens einem Student gelesen werden.

Unter der Annahme, dass jeder Student für jedes Buch die gleiche Zeit zum Lesen hat, ist ein Plan gesucht, der es allen Studenten in möglichst kurzer Zeit ermöglicht, sich vorzubereiten. In diesem Beispiel entspricht das Buch einer Maschine und der Student dem Auftrag (ist auch umgekehrt möglich!).

Beispiel 4. Johnsonsche Regel (1954) für das Problem $F2 \mid\mid C_{max}$:

n Aufträge A_i sollen auf zwei Maschinen bearbeitet werden, wobei jeder Auftrag zuerst auf der Maschine M_1 und dann auf der Maschine M_2 zu bearbeiten ist. Die Bearbeitungszeiten t_{ij} für Auftrag A_i auf Maschine M_j sind fest vorgegeben. Gesucht ist ein Bearbeitungsplan mit minimaler Gesamtbearbeitungszeit aller Aufträge auf den beiden Maschinen.

Algorithmus 1: Johnsonsche Regel

Exaktes Verfahren für Problem $F2 \mid\mid C_{max}$

Johnsonsche Regel

Eingabe: n, Bearbeitungszeitmatrix $P = (p_{ij})$ vom Format $n \times 2$;
Ausgabe: Optimale organisatorische Reihenfolge auf den Maschinen M_1 und M_2.
BEGIN
Ordne zuerst alle Aufträge A_i mit $p_{i1} \leq p_{i2}$ nach nichtfallenden Werten
der Bearbeitungszeiten auf der ersten Maschine an;
Ordne dann die verbleibenden Aufträge A_i nach nichtwachsenden Werten
der Bearbeitungszeiten auf der zweiten Maschine an;
END.

Bezeichnungen und Definitionen

Eine Menge von **Aufträgen (Jobs)** A_i, $i \in I = \{1, ..., n\}$, ist auf auf einer Menge von **Maschinen** $M_j, j \in J = \{1, ..., m\}$, zu bearbeiten.

Die **Bearbeitungszeit** von Auftrag A_i auf Maschine M_j sei $p_{ij} \geq 0$, also fixiert und gegeben.

Matrix der Bearbeitungszeiten: $P = (p_{ij})_{n \times m}$

Die Menge der **Operationen** ist durch $SIJ = \{(i, j) \in I \times J \mid p_{ij} > 0\}$ definiert, wobei:
n_i: Anzahl der Operationen für Auftrag A_i und
v_j: Anzahl der Operationen auf Maschine M_j.

Häufig wird vorausgesetzt, dass jeder Auftrag zu einem festen Zeitpunkt auf höchstens einer Maschine bearbeitet wird und jede Maschine zu einem festen Zeitpunkt höchstens einen Auftrag bearbeitet.

Zunächst darf die Bearbeitung der Operationen nicht unterbrochen werden.

Scheduling-Problem

Machine Orders (MO):
Unter der **technologischen Reihenfolge** des Auftrags A_i versteht man die Reihenfolge der Maschinen, auf denen er bearbeitet wird: $M^i_{j_1} \to M^i_{j_2} \to \ldots \to M^i_{jn_i}$, $i \in I$.

Job Orders (JO):
Unter der **organisatorischen Reihenfolge** auf der Maschine M_j versteht man die Reihenfolge der Aufträge auf dieser Maschine: $A^j_{i_1} \to A^j_{i_2} \to \ldots \to A^j_{iv_j}$, $j \in J$.

Ein **Plan (Sequence)** ist eine zulässige Kombination aus MO und JO.

Ein **Schedule** ist ein Zeitplan zu einem Plan, er wird beschrieben durch die Start- oder Fertigstellungszeiten aller Operationen.

Scheduling-Problem: Gesucht ist ein Schedule zu einer zulässigen Kombination von technologischen und organisatorischen Reihenfolgen unter den verschiedensten zusätzlichen Nebenbedingungen, so dass eine gegebene Zielfunktion ihren optimalen Wert annimmt.

Die Graphen *G(MO)*, *G(JO)* und *G(MO,JO)*

Es werden folgende Digraphen auf der Menge der Operationen als Knoten definiert:

$G(MO) = (V, A(MO))$ enthält die gerichteten Kanten aus den technologischen Reihenfolgen:
$((i, j), (k, l)) \in A \leftrightarrow (i = k \wedge$ Auftrag A_i geht nach Bearbeitung auf M_j direkt auf M_l über)
(**horizontale Kanten**)

$G(JO) = (V, A(JO))$ enthält die gerichteten Kanten aus den organisatorischen Reihenfolgen:
$((i, j), (k, l)) \in A \leftrightarrow (j = l \wedge$auf Maschine M_j wird nach Auftrag A_i direkt A_k bearbeitet.)
(**vertikale Kanten**)

$G(MO, JO) = (V, A(MO, JO)) = G(MO) \cup G(JO)$ enthält die gerichteten Kanten aus den technologischen und organisatorischen Reihenfolgen:

Eine Kombination aus MO und JO ist **zulässig**, wenn der zugehörende Digraph $G(MO, JO)$ zyklenfrei ist. Ein solcher Digraph wird als **Plan- oder Sequencegraph** bezeichnet.

Rangmatrizen der drei Graphen

Die azyklischen Digraphen $G(MO)$, $G(JO)$ und $G(MO, JO)$ werden durch die Matrizen MO, JO und LR beschrieben, die die Ränge der Knoten (i, j) in den jeweiligen Graphen angeben. Existiert die Operation (i, j) nicht, wird an der Position (i, j) in der jeweiligen Matrix ein Punkt gesetzt.

Der **Rang eines Knotens** (i, j) bezeichnet die Anzahl der Knoten auf einem längsten Weg zu (i, j) in dem gegebenen Digraphen, wobei der Knoten (i, j) mitgezählt wird.

Beispiel 5. Seien $n = 3,\ m = 4,\ SIJ = I \times J \setminus \{(13)\}$ sowie

technologische Reihenfolgen:

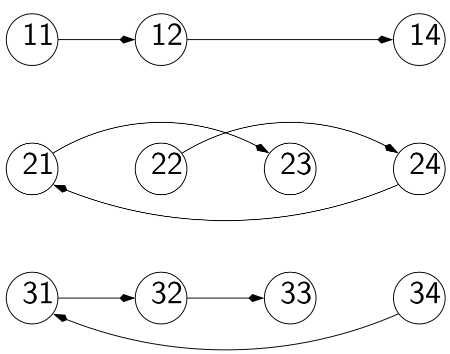
$A_1 : M_1 \to M_2 \to M_4$
$A_2 : M_2 \to M_4 \to M_1 \to M_3$
$A_3 : M_4 \to M_1 \to M_2 \to M_3$

organisatorische Reihenfolgen:

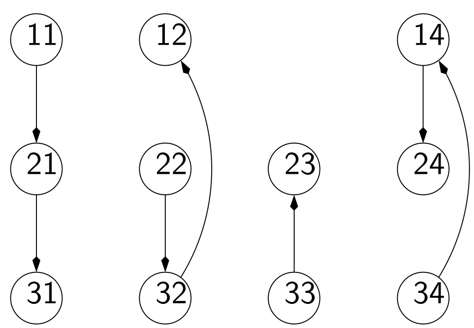
$M_1 : A_1 \to A_2 \to A_3$
$M_2 : A_2 \to A_3 \to A_1$
$M_3 : A_3 \to A_2$
$M_4 : A_3 \to A_1 \to A_2$

Beispiel für die Graphen *G(MO)*, *G(JO)* und *G(MO,JO)*

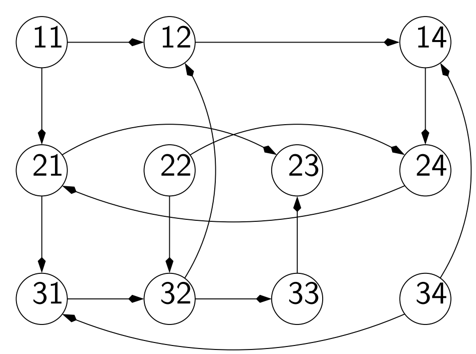
$G(MO)$:



$G(JO)$:



$G(MO, JO)$:



$$MO = \begin{pmatrix} 1 & 2 & . & 3 \\ 3 & 1 & 4 & 2 \\ 2 & 3 & 4 & 1 \end{pmatrix} \qquad JO = \begin{pmatrix} 1 & 3 & . & 2 \\ 2 & 1 & 2 & 3 \\ 3 & 2 & 1 & 1 \end{pmatrix}$$

Ist diese Kombination zulässig?

Schedules

Jeder Knoten (i, j) des Plangraphen $G(MO, JO)$ wird mit der Bearbeitungszeit p_{ij} gewichtet.

Ein **Schedule** S ist ein Zeitplan zu einem Plan LR, der durch die Matrix aller Start- oder Fertigstellungszeiten der Operationen beschrieben wird.

Bemerkung: Zu einem Schedule S gehört **genau** ein Plan LR.
Die Menge aller Schedules MS zu einem festen Plan hat überabzählbar viele Elemente.

Bemerkung: Die Relation R mit

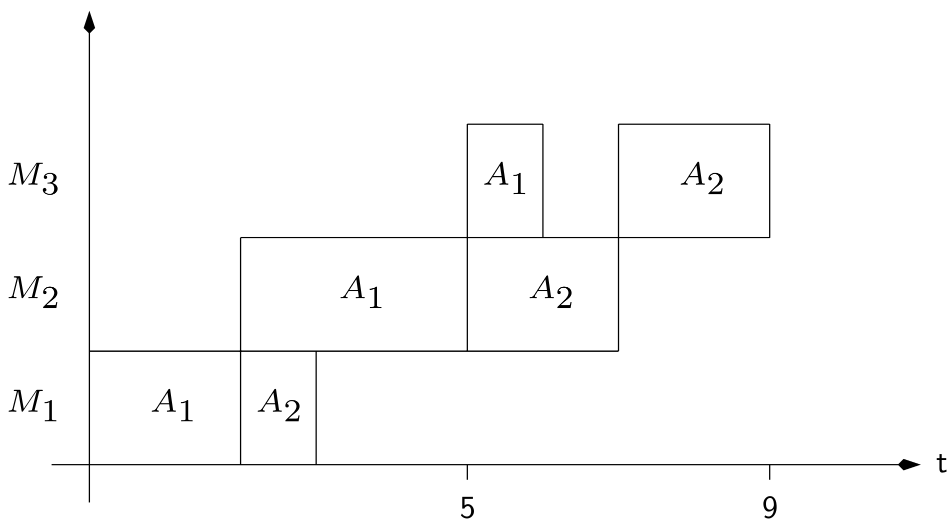
$$\forall\ S_1, S_2 \in MS : \ \ S_1 \, R \, S_2 \ \leftrightarrow LR(S_1) = LR(S_2)$$

ist eine **Äquivalenzrelation** auf MS.

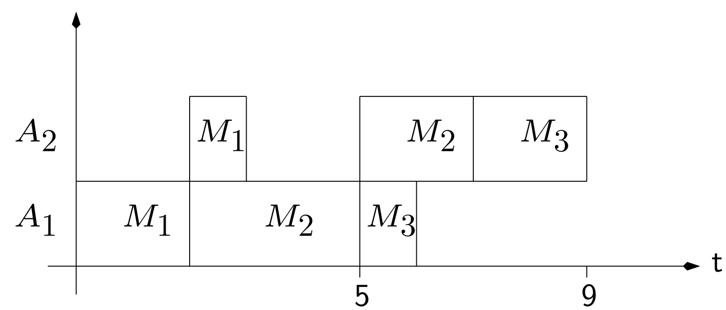
Als Äquivalenzklassenvertreter werden **semiaktive Schedules** gewählt: Jede Operation wird unter Einhaltung des zugrunde liegenden Plans so früh wie möglich bearbeitet (**linksbündig**). Die Fertigstellungszeiten aller Operationen $(i, j) \in SIJ$ eines semiaktiven Schedules werden in der Matrix $C = (c_{ij})$ **(completion times)** zusammengefasst.

Ganttdiagramme

Beispiel 6. Gegeben sind die Bearbeitungszeitmatrix $P = \begin{pmatrix} 2 & 3 & 1 \\ 1 & 2 & 2 \end{pmatrix}$ und die Ganttdiagramme eines semiaktiven Schedules:



Maschinenorientiertes Ganttdiagramm



Auftragsorientiertes Ganttdiagramm

Die Gesamtbearbeitungszeit beträgt 9 Zeiteinheiten.

Klassen von Schedules

- Ein Schedule heißt **semiaktiv**, wenn keine Operation ohne Änderung des Plans früher fertig werden kann.
- Ein Schedule heißt **aktiv**, wenn keine Operation früher bearbeiten werden kann, ohne dass sich die Abarbeitung einer anderen Operation verzögert.
- Ein Schedule heißt **non-delay**, wenn keine Maschine still steht, sofern zu diesem Zeitpunkt ein Auftrag zur Bearbeitung auf der Maschine anliegt.

Zusammenhang zwischen den Schedules: