

## Modell für Scheduling

# Benötigte Daten

## Beschreibung

Die Daten sind in 2 Teile aufgeteilt:

- System-Info, beinhaltet Informationen über die Umgebung in der produziert wird
- Orders, beinhaltet Kundenaufträge/Bestellungen, die in dem beschriebenen System umgesetzt werden sollen

## System-Info

Beinhaltet Informationen über die Produktions-Umgebung

**Tasks** Alle Arbeitsschritte / Aktionen, die in der Produktions-Umgebung durchgeführt werden können. (Wichtig: Tasks die z.Bsp. bei Task A als „follow up task“ angegeben werden sollen KEINEN Verweis auf Task A in den „preceding tasks“ haben).

- id – eindeutige ID des Tasks (kann fortlaufende Nummer sein, nur für interne Identifizierung)
- name – Name des Tasks (für einfacher lesbare Ergebnisse)
- resources – (liste) benötigt Ressourcen, um den Task durchzuführen (id + anzahl)
- result\_resources – (liste) alle Ressourcen (ID + Anzahl (Losgröße)) die durch den Task entstehen
- preceding\_tasks – (liste) Tasks, die gemeinsam mit diesem Task durchgeführt werden müssen (vorangestellte Tasks, müssen daher nicht extra als Arbeitsschritt übermittelt werden)
- follow\_up\_tasks – (liste) Tasks, die nach dem Task durchgeführt werden müssen (wie preceding tasks)
- independent – boolean, um festzustellen, ob der Task im Schedule verwendet werden soll (e.g. false für Tasks die immer nur als vorangestellter Task vorkommen und allein keinen Sinn ergeben)
- prepare\_time – Zeit die zum Einrichten der Maschine/Arbeitsstation für diesen Task benötigt wird

- `unprepare_time` – Zeit, die nach der Verwendung der Maschine/Arbeitsstation für diesen Task benötigt wird

**Recipes** Die Rezepte für die verschiedenen Ressourcen, die hergestellt werden sollen (Produkte oder auch Zwischenprodukte)

- `id` – eindeutige ID des Rezepts (kann auch fortlaufende Nummer sein, nur für interne Identifizierung)
- `name` – Name des Rezepts (für einfacher lesbare Ergebnisse)
- `tasks` – (liste) IDs der Tasks, die durchgeführt werden müssen, um das Rezept umzusetzen (Ergebnis und Ergebnismenge ergeben sich aus dem letzten Task)

**Resources** Alle Ressourcen, die in der Produktions-Umgebung verwendet oder hergestellt werden können

- `id` – eindeutige ID der Ressource (kann auch fortlaufende Nummer sein, nur für interne Identifizierung)
- `name` – Name der Ressource (für einfacher lesbare Ergebnisse)
- `stock` – Anzahl der Ressource im Lagerbestand
- `price` – Preis pro Einheit, um die Ressource einzukaufen (0 wenn die Ressource nicht gekauft werden kann)
- `renewable` – boolean, um festzustellen, ob die Ressource verbraucht wird oder nicht (könnte z.Bsp. auch Ressource Mitarbeiter sein, der wieder verfügbar ist, wenn ein Task beendet ist)
- `recipes` – (liste) IDs der Rezepte, durch die diese Ressource hergestellt werden kann (falls vorhanden, sonst leere Liste)

**Workstations** Alle Maschinen / Arbeitsstationen, die in der Produktions-Umgebung vorhanden sind

- `id` – eindeutige ID der Maschine/Arbeitsstation (kann auch fortlaufende Nummer sein, nur für interne Identifizierung)
- `name` – Name der Maschine/Arbeitsstation (für einfacher lesbare Ergebnisse)

- `basic_resources` – (liste) beinhaltet die IDs der benötigten Ressourcen + die Anzahl für jede Ressource, die (unabhängig der bevorstehenden Tasks) benötigt werden um die Maschine/Arbeitsstation zu betreiben (z.Bsp. Mitarbeiter)
- `tasks` – (liste) IDs aller Tasks, die auf dieser Maschine/Arbeitsstation durchgeführt werden können + die Dauer des Tasks auf dieser Maschine/Arbeitsstation

## Orders

Repräsentiert Kundenaufträge/Bestellungen, die mithilfe der Produktions-Umgebung umgesetzt werden soll.

- `id` – eindeutige ID der Bestellung (kann auch fortlaufende Nummer sein, nur für interne Identifizierung)
- `arrival_time` – Zeitpunkt, zu dem die Bestellung im System eingegangen ist
- `delivery_time` – Zeitpunkt, zu dem die Bestellung fertig sein soll
- `latest_acceptable_time` – Spätester Zeitpunkt, zu dem die Bestellung fertig sein soll, falls der gewünschte Lieferzeitpunkt nicht eingehalten kann
- `resources` – (liste) IDs + Anzahl der bestellten Ressourcen, die hergestellt werden sollen, + der Preis, den der Kunde für diese Ressource bezahlen wird
- `penalty` – falls die Bestellung nicht durchgeführt wird
- `tardiness_fee` – falls die Bestellung nach dem gewünschten Termin, aber vor dem letzten möglichen Termin durchgeführt wird
- `divisible` – boolean, gibt an ob auch nur Teile der Bestellung bearbeitet werden können, oder ob alle bestellten Ressourcen geliefert werden müssen (oder gar keine)
- `customer_id` – ID des Kunden, von dem der Auftrag stammt

# Input / Output

Genaue Struktur der Datenübertragung mit Beispieldaten unter "JSON-Beispiel"(section )

Input:

Orders:

*OrderA*

*Task<sub>1</sub> – Task<sub>n</sub>*

*price<sub>1</sub> – price<sub>n</sub>*

*penalty*

*tardiness\_fee*

*delivery\_date*

*latest\_date*

Mapping von Tasks zu Jobs Bsp.:

(Zuteilung zu exakten Job, Bsp. Task: Material schneiden -> Job: Material schneiden für Bestellung A)

*TaskA.1*

*TaskID*

*JobID*

Intermediate:

$j_1 - (w, s)$

$j_2 - (w, s)$

$j_2 - (w, s)$

.

.

.

$j_n - (w, s)$

Output:

$$\left[ \begin{array}{ccccc} (j_{11}, s_{11}) & (j_{12}, s_{12}) & (j_{13}, s_{13}) & \dots & (j_{1n}, s_{1n}) \\ (j_{21}, s_{21}) & (j_{22}, s_{22}) & (j_{23}, s_{23}) & \dots & (j_{2n}, s_{2n}) \\ \dots & \dots & \dots & \dots & \dots \\ (j_{m1}, s_{m1}) & (j_{m2}, s_{m2}) & (j_{m3}, s_{m3}) & \dots & (j_{mn}, s_{mn}) \end{array} \right] \text{ Dauer?}$$



Figure 0.1: Ablauf

## Modell

### Variablen Definition

$O$  - Set of all Orders

$o$  - Specific Order,  $o \in O$

$W$  - Set of all Workstations

$T$  - Set of all Tasks

$T_o$  - Set of all Tasks needed for Order  $o$ ,  $T_o \subset T$   
 $x$  - Specific Task,  $x \in T$   
 $T_{xp}$  - Set of all Tasks preceding Task  $x$ ,  $T_{xp} \subset T$   
 $T_{xf}$  - Set of all Tasks following Task  $x$ ,  $T_{xf} \subset T$   
 $J$  - Set of all Jobs  
 $J_w$  - Set of all Jobs assigned to Workstation  $w$ ,  $w \in W$   
 $j_x$  - Job linked to Task  $j$ ,  $x \in T$ ,  $j \in J$   
 $x_j$  - Task linked to Job  $j$ ,  $x \in T$ ,  $j \in J$   
 $W_j$  - Set of Workstations eligible for Job  $j$ ,  $W_j \subset W$   
 $R$  - Set of all Resources  
 $w$  - Specific Workstation  $w \in W$   
 $R_j$  - Set of Resources needed for Job  $j$ ,  $R_j \subset R$   
 $r$  - Specific Resource,  $r \in R$   
 $d_{xw}$  - Duration of Task  $x$  on Workstation  $w$   
 $j_w$  - Selected Workstation for a Job  $j$ ,  $w \in W_j$   
 $j_s$  - Start time slot of Job  $j$   
 $j_{dt}$  - Deliver time slot of Job  $j$   
 $j_{lt}$  - Latest allowed time slot of Job  $j$   
 $i_{rt}$  - Amount of Resource  $r$  in the Inventory at time slot  $t$   
 $l_j$  - Binary Variable, 1 if Job  $j$  is late (Tardiness Fee applies)  
 $l_o$  - Binary Variable, 1 if Order  $o$  is late (Tardiness Fee applies)  
 $u_o$  - Binary Variable, 1 if Order  $o$  can only be fulfilled partially  
 $p_o$  - Price the customer pays for Order  $o$   
 $f_o$  - Price reduction for tardy orders  
 $O_c$  - Set of all Orders which could be completed in the derived schedule,  
 $O_c \subset O$

## Objective Function + Nebenbedingungen

### Objective Functions

Equation 1 Minimize tardy jobs

Equation 2 Maximize earning

Equation 3 Minimize deviation from the expected delivery date

Equation 4 Maximize Set Size of scheduled jobs

Equation 5 Maximize Set Size of scheduled jobs with least tardy jobs

Equation 6 Maximize Set Size of scheduled jobs with least deviations

$$\begin{aligned}
j &:= J_i \\
\text{minimize } & \sum_{i=0}^J l_j
\end{aligned} \tag{1}$$

$$\begin{aligned}
o &:= O_{c,i} \\
\text{maximize } & \sum_{i=0}^{O_c} p_o - (l_o * f_o)
\end{aligned} \tag{2}$$

$$\begin{aligned}
j &:= J_i \\
y &:= x_j \\
\text{minimize } & \sum_{i=0}^J |(j_s + d_{y,w}) - j_{dt}|
\end{aligned} \tag{3}$$

$$\text{maximize } |O_c| \tag{4}$$

$$\begin{aligned}
j &:= J_i \\
\text{maximize } & |O_c| - \sum_{i=0}^J l_j
\end{aligned} \tag{5}$$

$$\begin{aligned}
j &:= J_i \\
y &:= x_j \\
x &:= \sum_{i=0}^J (|(j_s + d_{y,w}) - j_{dt}| > 0) \\
\text{maximize } & |O_c| - x
\end{aligned} \tag{6}$$

### Nebenbedingungen

Equation 7 to make sure jobs finish before last possible time slot

Equation 8 to make sure jobs have sufficient resources to start

Equation 9 checks if tardiness fee applies



Equation 10 jobs can't start before preceding tasks finish  
Equation 11 follow up jobs can't start before job finished  
Equation 12 only one active job at one time slot for each Workstation  
Equation 13 starting times need to be after timeslot 0

$$j_s \leq j_{lt} - d_{j,w} \quad (7)$$

$$\sum_{m=0}^{R_j} (i_{r_m,t} - r_m \leq 0) < 1 \quad (8)$$

$$l_j := j_{dt} \leq j_s \leq j_{lt} \quad (9)$$

$$\begin{aligned} & y \in T_{j_{xp}} \\ \sum_{m=1}^y ((j_{y_m,s} + d_{y_m,w}) - (j_{y_{m-1}} + d_{y_{m-1},w}) < 0) < 1 \end{aligned} \quad (10)$$

$$\begin{aligned} & y \in T_{j_{xf}} \\ \sum_{m=1}^y ((j_{y_{m-1},s} + d_{y_{m-1},w}) < j_{y_m,s}) < 1 \end{aligned} \quad (11)$$

$$\begin{aligned} & \forall w \in W \\ & i = J_{wm} \\ & j = J_{wn} \\ & x = x_i \\ & x' = x_j \end{aligned} \quad (12)$$

$$\sum_{m=0}^{J_w} \left( \sum_{n=0}^{J_w-1} ((1 - (j_s + d_{x',w} < i_s)) + (1 - (j_s > i_s + d_{x,w}))) \right) < 1$$

$$\begin{aligned} & j = J_m \\ \sum_{m=0}^J (j_s < 0) < 1 \end{aligned} \quad (13)$$

## JSON-Beispiel

```
1 {
2   "system-info":
3   {
4     "tasks":
5     [
6       {
7         "id": 0,
8         "name": "example_task",
9         "resources":
10        [
11          {
12            "id": 4,
13            "amount": 20
14          },
15          {
16            "id": 6,
17            "amount": 2
18          }
19        ],
20        "result_resources":
21        [
22          {
23            "resource_id": 1,
24            "amount": 100
25          },
26          {
27            "resource_id": 2,
28            "amount": 10
29          }
30        ],
31        "preceding_tasks":
32        [
33          1, 2, 3
34        ],
35        "follow_up_tasks":
36        [ ],
37        "independent": true,
38        "prepare_time": 10,
```

```

39         "unprepare_time": 5
40     },
41     {
42         "id": 1,
43         "name": "example_task 2",
44         "resources":
45         [
46             {
47                 "id": 7,
48                 "amount": 4
49             }
50         ],
51         "result_resources":
52         [
53             {
54                 "resource_id": 3,
55                 "amount": 4
56             }
57         ],
58         "preceding_tasks":
59         [ ],
60         "follow_up_tasks":
61         [
62             4, 5
63         ],
64         "independent": false,
65         "prepare_time": 10,
66         "unprepare_time": 5
67     }
68 ],
69 "recipes":
70 [
71     {
72         "id": 0,
73         "name": "example_recipe",
74         "tasks":
75         [
76             0, 4
77         ]
78     },

```

```

79         {
80             "id": 1,
81             "name": "example_recipe 2",
82             "tasks":
83             [
84                 5, 6
85             ]
86         }
87     ],
88     "resources":
89     [
90         {
91             "id": 0,
92             "name": "example_resource",
93             "stock": 500,
94             "price": 10,
95             "renewable": false,
96             "recipes":
97             [
98                 0, 1
99             ]
100         },
101         {
102             "id": 1,
103             "name": "example_resource 2",
104             "stock": 300,
105             "price": 15,
106             "renewable": true,
107             "recipes":
108             [ ]
109         }
110     ],
111     "workstations":
112     [
113         {
114             "id": 0,
115             "name": "example_machine",
116             "basic_resources":
117             [
118                 {

```

```

119         "id": 10,
120         "amount": 5
121     },
122     {
123         "id": 12,
124         "amount": 3
125     }
126 ],
127 "tasks":
128 [
129     {
130         "task_id": 0,
131         "duration": 10
132     },
133     {
134         "task_id": 3,
135         "duration": 5
136     },
137     {
138         "task_id": 4,
139         "duration": 20
140     }
141 ]
142 }
143 ]
144 },
145 "orders":
146 [
147     {
148         "id": 0,
149         "arrival_time": "07.02.2022 00:00:00",
150         "delivery_time": "20.02.2022 14:30:00"
151     },
152     {
153         "latest_acceptable_time": "26.02.2022
154             00:00:00",
155         "resources":
156 [
157     {
158         "id": 1,
159         "amount": 20,

```

```

157         "price": 700
158     },
159     {
160         "id": 3,
161         "amount": 10,
162         "price": 300
163     }
164 ],
165 "penalty": 100,
166 "tardiness_fee": 10,
167 "divisible": true,
168 "customer_id": 1
169 },
170 {
171     "id": 0,
172     "arrival_time": "09.02.2022 13:00:00",
173     "delivery_time": "25.02.2022 15:00:00"
174     ,
175     "latest_acceptable_time": "26.02.2022
176         00:00:00",
177     "resources":
178     [
179         {
180             "id": 0,
181             "amount": 15,
182             "price": 500
183         },
184         {
185             "id": 1,
186             "amount": 30,
187             "price": 1000
188         },
189         {
190             "id": 2,
191             "amount": 10,
192             "price": 200
193         }
194     ],
195     "penalty": 100,
196     "tardiness_fee": 10,

```

```
195         "divisible": false,  
196         "customer_id": 0  
197     }  
198 ]  
199 }
```