Strojniški vestnik - Journal of Mechanical Engineering 53(2007)12, 844-857
UDK - UDC 658.51
Pregledni znanstveni članek - Review scientific paper (1.02)

Razvoj naprednih metod za vodenje proizvodnih postopkov

The Development of Advanced Methods for Scheduling Production Processes

Tadej Tasič - Borut Buchmeister - Bojan Ačko
(Fakulteta za strojništvo, Univerza v Mariboru)

Načrtovanje rokov in zmogljivosti spada v področje vodenja proizvodnje. Terminiranje pomeni razporejanje omejenih virov opravilom v določenem časovnem obdobju. Torej gre za postopek odločanja s ciljem optimizacije enega ali več meril. Splošno obsega grobo (dolgoročno), srednjeročno in podrobno (kratkoročno) načrtovanje. Zadnjemu dajemo zaradi neposredne povezave z opravilno učinkovitostjo največji pomen. V prispevku so predstavljeni razviti postopki terminiranja: prednostna pravila, omejeno iskanje v snopu in hevristično preiskovanje glede na omejitve. Vse je podkrepljeno s prikazom na primerih.

(Ključne besede: načrtovanje proizvodnih postopkov, deterministično razvrščanje, terminiranje, prednostna pravila)

The planning of dates and capacities belongs to the area of production management. Scheduling represents the allocation of scarce sources to tasks in a definite period of time. So it is about a process of deciding, with the goal of optimization, about one or more criteria. In general it includes rough (long-term), medium-term and detailed (short-term) planning. The last of these has the greatest importance because of a direct connection with operational effectiveness. Some developed scheduling procedures are presented in the paper, i.e., the priority (dispatching) rules, the filtered-beam search, and the constraint-guided heuristic search. All this is substantiated with described examples.

(Keywords: production process planning, deterministic sequencing, scheduling, priority rules)

0 UVOD

"Terminiranje pomeni načrtovanje postopkov dela z dodeljevanjem virov dela opravilom in določitvijo njihovih začetnih časov. Različne komponente problemov terminiranja so opravila, omejitve, viri in optimizacijska merila. Opravila morajo biti programirana tako, da zadovoljijo določene zahteve. Seveda pa v praksi navadno upoštevamo več meril." [1].

V zadnjih desetletjih se je dosti naredilo na teoretičnih raziskavah s področja determinističnega razvrščanja. Število in raznolikost različnih modelov sta presenetljivi. Ko rešujemo probleme razvrščanja, menimo, da je število delovnih mest in strojev končno – določeno. Tako število delovnih nalogov označimo z n, število strojev pa z m. Običajno dodamo indeks (števec), in sicer j dodelimo delovnim mestom, i pa strojem.

0 INTRODUCTION

"Scheduling is to forecast the processing of work by assigning resources to tasks and fixing their start times. The different components of a scheduling problem are the tasks, the potential constraints, the resources and the objective function. The tasks must be programmed to optimise a specific objective. Often it will be more realistic in practice to consider several criteria." [1].

In the past few decades a considerable amount of theoretical research has been done in the field of deterministic scheduling. The number and variety of different models is astounding. In all the scheduling problems considered, the number of jobs and the number of machines are assumed to be finite. The number of jobs is denoted by n and the number of machines by m. Usually, the subscript j refers to a job and the subscript i to a machine.

Optimizacijska merila lahko razdelimo na dva tipa: na povezana s časom dokončanja naloge in povezana s stroški. V načrtovanju rokov in zmogljivosti uporabljamo za reševanje problemov različne metode, ki se razlikujejo glede na zapletenost problema. Za preproste primere uporabljamo hevristična prednostna pravila (npr. pravilo najkrajšega časa obdelave, pravilo najzgodnejšega dobavnega roka, pravilo najzgodnejšega prihoda itn.). Kadar imamo na voljo več strojev, je nujna razširitev z dodatnim pravilom dodelitve nalog strojem. V primeru enakih strojev običajno uporabljamo pravilo prvega razpoložljivega stroja, ta dodeli naročilo prvemu stroju, ki je na voljo. Če stroji niso enaki, se najbolj uporablja pravilo najhitrejšega stroja, ki dodeli naročilo najhitrejšemu med razpoložljivimi stroji.

Za načrtovanje projektov uporabljamo Ganttove diagrame ali gantograme. Pogosta metoda za pomoč pri izvajanju med seboj logično povezanih dejavnosti z namenom, da bi dosegli vnaprej postavljen cilj, je tudi tako imenovano mrežno načrtovanje (mrežni diagrami).

Za bolj zapletene probleme v sodobni proizvodnji pa moramo posegati po matematično-analitičnih metodah reševanja problemov. Preprost primer je Johnsonov algoritem [2], prikazan na sliki 1, kjer so:

J – nalog, ki izpolnjuje zahteve,
p – skupni čas opravila,
C_{max} – skupni čas izvedbe vseh nalogov,
prmu – vrstni red izvajanja nalogov.

Druga možnost je uporaba računalniških simulacij, ki dinamično prikazujejo potek nekega postopka in omogočajo ob spremembah

Optimization criteria can be divided into two types: criteria that are connected with the time of the completion of a job, and criteria connected with costs. When planning dates and capacities, we usually use different methods for solving problems. They differ in terms of the complexity of a problem (in simple cases we use heuristic dispatching rules – for example the rule of the shortest processing time, the earliest due date rule, the first-come first-served rule, etc.). When we can use several machines, an additional rule is a necessary solution for the allocation of tasks to the machines. In the case when machines are identical we normally use the rule of the first available machine, which assigns an order to the first available machine. If the machines are not identical, the rule of the fastest machine is the most appropriate, and this assigns an order to the fastest among the available machines.

When planning projects we help ourselves with Gantt diagrams or Gantt charts. A frequent method for help, using performance of activities that are mutually connected with the intention of reaching a goal that is set in advance, is also network modelling (network diagrams).

For more complicated problems in modern production we have to implement mathematical-analytical methods for solving problems. A simple example is Johnson's algorithm, which is shown in Figure 1 [2], where:

J – a set of jobs, that satisfies a condition,
p – processing time,
C_{max} – completion time of all jobs,
prmu – indicates that the operations occur on the machines in the same order.

The second possibility is the use of computer simulations that dynamically show the course of a

```
/*T je niz nalog, ki jih moramo razporediti*/     /*T is the set of jobs to schedule*/
Naj bo / Let:          $U = \{J_i \in T \mid p_{i,1} < p_{i,2}\}$;
Naj bo / Let:          $V = \{J_i \in T \mid p_{i,1} \geq p_{i,2}\}$;
Razporedi U naraščajoče po vrednostih $p_{i,1}$;     Sort $U$ by increasing values of $p_{i,1}$;
Razporedi V padajoče po vrednostih $p_{i,2}$;        Sort $V$ by decreasing values of $p_{i,2}$;
                       $S = U // V$;
                       $C^{*}_{\max} = C_{\max}(S)$;
Izpiši / Print:        $S,\ C^{*}_{\max}$;
```

Sl. 1. *Johnsonov algoritem za $F2|prmu|C_{maks}$ problem*
Fig. 1. *Johnson's algorithm for the $F2|prmu|C_{max}$ problem*

parametrov in razmer pri delu vnaprejšnje določevanje stanj načrtovanega postopka. Eno prvih preglednic terminiranja zasledimo v Conway, Maxwell in Miller [3]. V znanstveni literaturi zasledimo mnogo raziskav s teh področij, nekatere novejše prispevke so objavili avtorji: Lawler idr. [4], Brah in Wheeler [5], Polajnar idr. [6], Brucker [7], Buchmeister idr. [8], Gomes in Meile [9], Blažewicz idr. [10], Garcia-Sabater [11], Nyman in Levitt [12], Leung [13], Palmer [14], Koo in Jang [15].

1 NAČIN OZNAČEVANJA PROBLEMOV TERMINIRANJA ($\alpha \mid \beta \mid \gamma$)

Vsak problem terminiranja opišemo s trojčkom $\alpha \mid \beta \mid \gamma$. Polje α opisuje stroje in vsebuje en zapis. Polje β podaja podrobnosti o lastnostih postopkov in omejitvah. Lahko je prazno, z enim ali več zapisi. Polje γ vsebuje cilj optimizacije (funkcijo, katero minimiziramo) in ima običajno en zapis [16].

Mogoči zapisi v polju α:

1 – en stroj,
$\boldsymbol{P_m}$ – m enakih vzporednih strojev,
$\boldsymbol{Q_m}$ – m različnih vzporednih strojev,
$\boldsymbol{F_m}$ – m v vrsto postavljenih strojev,
$\boldsymbol{J_m}$ – m različnih strojev (vsak nalog ima svoje zaporedje opravil) itn.

Nekaj mogočih zapisov v polju β:

$\boldsymbol{r_j}$ – čas prihoda naloga (sprostitev),
$\boldsymbol{s_{jk}}$ – od zaporedja nalogov odvisni pripravljalni in končni časi,
prec – omejitev glede predhodno izvedenih opravil (strogo zaporedje),
prmu – omejitev glede enakega vrstnega reda nalogov po strojih (ni prehitevanja),
block – zastoji v sistemu (npr. zaradi polnih zalogovnikov),
brkdwn – izpadi (nedelovanje) strojev itn.

Nekaj mogočih zapisov v polju γ:

$\boldsymbol{C_{max}}$ – skupni časi izvedbe vseh nalogov,
$\boldsymbol{L_{max}}$ – največje odstopanje roka dobave
$\sum w_j T_j$ – skupna utežena kasnitev nalogov itn.

2 PREDNOSTNA PRAVILA

S prednostnim pravilom priredimo vsakemu nalogu, ki je v čakalni vrsti pred določenim delovnim mestom, prednostno število (skalarno vrednost). To število potem določa položaj naloga v čakalni vrsti v primerjavi z drugimi čakajočimi

process, and make possible a forecast of the planning-process conditions when it comes to changes of parameters out of the conditions of the work. One of the first studies about scheduling was published by Conway, Maxwell and Miller [3]. In the scientific literature we can find many researches, some of the newest in this area are in the publications of Lawler et al. [4], Brah and Wheeler [5], Polajnar et al. [6], Brucker [7], Buchmeister et al. [8], Gomes and Meile [9], Blažewicz et al. [10], Garcia-Sabater [11], Nyman and Levitt [12], Leung [13], Palmer [14], Koo and Jang [15].

1 THE NOTATION OF SCHEDULING PROBLEMS ($\alpha \mid \beta \mid \gamma$)

A scheduling problem is described by a triplet, $\alpha \mid \beta \mid \gamma$. The α field describes the machine environment and contains a single entry. The β field provides details of the processing characteristics and constraints. This field may contain no entries, a single entry, or multiple entries. The γ field contains the objective to be optimized and usually contains a single entry.

Possible entries in the α field:

1 – single machine,
$\boldsymbol{P_m}$ – m identical parallel machines,
$\boldsymbol{Q_m}$ – m different parallel machines,
$\boldsymbol{F_m}$ – flow shop with m machines,
$\boldsymbol{J_m}$ – job shop with m machines (each job has a unique routing), etc.

Some possible entries in the β field:

$\boldsymbol{r_j}$ – release date,
$\boldsymbol{s_{jk}}$ – sequence-dependent setup times,
prec – precedence constraints,
block – blocking (full buffers between operations),
brkdwn – breakdowns of machines, etc.

Some possible entries in the γ field:

$\boldsymbol{C_{max}}$ – makespan (completion time of all jobs),
$\boldsymbol{L_{max}}$ – maximum lateness,
$\sum w_j T_j$ – total weighted tardiness.

2 THE PRIORITY RULES

With a priority rule we can arrange a priority value (a scalar value) to each job (an order) that is in the waiting queue in front of a defined working place (a machine). This value then defines the position of a job in the waiting queue in comparison nalogi. Nalog z največjo prednostjo se prvi izvede. Po dogovoru je to največje oziroma najmanjše prednostno število ([6], [17] in [18]).

Prednostna pravila so lahko statična ali dinamična, lokalna ali celotna, temeljna ali sestavljena in druga. Uporaba sestavljenih, seštevno ali množilno vezanih pravil omogoča sočasno kompromisno izpolnjevanje več proizvodnih ciljev. Raziskave pa kažejo, da stopnja učinkovitosti pri posameznih ciljih ni večja kakor pri uporabi temeljnih prednostnih pravil [6].

Prednostna pravila se delijo na:

- pravila, ki vključujejo čase opravil,
- pravila, ki vključujejo dobavni rok,
- pravila, ki ne vključujejo ne časov opravil, ne dobavnih rokov in na
- kombinirana pravila.

to other waiting jobs. After an agreement this is the largest, or the smallest, priority value ([6], [17] and [18]).

A priority rule can be static or dynamic, local or global, basic or complex, etc. The use of complex, additive or multiplicative tied rules makes it possible that more of the production goals are simultaneously improved. Research shows us that the degree of effectiveness in single goals is not higher than the one, which is achieved with basic priority rules [6].

The priority rules are divided into:

- rules that include operation times,
- rules that include the due date,
- rules that include neither operation times nor due dates,
- complex (composite) rules.

2.1 Primeri prednostnih pravil

2.1 Examples of priority rules

SPT:

SPT (Shortest Processing Time):

$$\min \sum_{j \in O_i} p_{ij} \qquad (1),$$

kjer so: i – števec nalogov,
j – števec opravil,
O – zaporedje vseh opravil naloga,
p – skupni čas opravil.

Prednost ima nalog z najmanjšim skupnim časom vseh opravil(najmanjši obseg dela).

where: i – index of jobs (counter),
j – index of operations (counter),
O – sequence of all operations of jobs,
p – processing time.

The job with the shortest processing times of all operations (the smallest extent of work) has the priority.

EDD:

EDD (Earliest Due Date):

$$\min d_i \qquad (2),$$

kjer je: d – dobavni rok naloga.

Prednost ima nalog z najzgodnejšim dobavnim rokom.

where: d – due date.

The job with the earliest due date has the priority.

LRPT:

LRPT (Longest Remaining Processing Time):

$$\max \sum_{j \in P_i} p_{ij} \qquad (3),$$

kjer je: P – zaporedje preostalih opravil naloga.

Prednost ima nalog z največjim skupnim časom preostalih opravil.

where: P – sequence of remaining operations of a job.

The job with the longest remaining processing time has the priority.

LS:

LS (Least Slack):

$$\min\left(d_i - t - \sum_{j \in P_i} p_{ij} \right) \qquad (4),$$

kjer je: t – trenutni čas.

Prednost ima nalog z najmanjšo časovno rezervo med dobavnim rokom in trenutnim

where: t – actual (current) time.

The job with the least slack between due date and actual date, considering the processing time of datumom z upoštevanjem skupnega časa preostalih opravil. Pri nalogih z enakim dobavnim rokom deluje kot LRPT; pri nalogih z enakim skupnim časom preostalih opravil pa kot EDD.

LSS:

the remaining operations, has the priority. When all the jobs have the same due date it works as LRPT; when jobs have the same processing time of remaining operations it works as EDD.

LSS (Least Static Slack):

$$\min\left(d_i - r_i - \sum_{j\in O_i} p_{ij}\right) \qquad (5),$$

kjer je: r – čas prihoda naloga.

Prednost ima nalog z najmanjšo časovno zalogo (zračnostjo), pri čemer se upoštevajo dobavni rok, čas prihoda naloga in skupni čas vseh opravil. Izračunana vrednost se s časom ne spreminja.

WSPT:

where: r – arrival time of a job.

The job with the least slack, where the due date, the arrival time of the job and the processing time of all the operations should be considered, has the priority. The calculated value does not change with time.

WSPT (Weighted Shortest Processing Time):

$$\min \sum_{j\in O_i} w_{ij} \cdot p_{ij} \qquad (6),$$

kjer je: w – utež naloga.

Prednost ima nalog z najmanjšim uteženim skupnim časom vseh opravil.

ATC:

Pravilo ATC je sestavljeno prednostno pravilo, ki kombinira pravilo WSPT (najmanjši uteženi čas obdelave) in pravilo LS (najmanjša zračnost). S pravilom ATC razvrstimo vsakič po en nalog. Ko je stroj prost, izračunamo vrednost funkcije $I_j(t)$ za vse čakajoče naloge. Nalog z največjo vrednostjo izberemo za obdelavo na stroju. Vrednost funkcije $I_j(t)$ za čakajoče naloge se s časom spreminja.

where: w – weight of a job.

The job with the shortest weighted processing time for all the operations has the priority.

ATC (Apparent Tardiness Cost):

The ATC rule is a complex priority rule that combines the WSPT rule (Weighted Shortest Processing Time) and the LS rule (Least Slack). The ATC rule always selects one job at a time. Every time a machine becomes free, we calculate the value of the function $I_j(t)$ for all the waiting jobs. The job with the highest function value is chosen for processing on the machine. The value of the function $I_j(t)$ for all the waiting jobs is time dependent.

$$I_j(t) = \frac{w_j}{p_j} \cdot e^{-\frac{\max(d_j - p_j - t, 0)}{k \cdot \overline{p}}} \qquad (7),$$

kjer sta: k – izkustveno določen parameter pogleda naprej,
$\overline{p}$ – povprečni čas obdelave preostalih delovnih nalogov.

Če je k zelo velik, se pravilo ATC zoži v pravilo WSPT, če pa je zelo majhen, pa v pravilo LS, če ni zakasnelih delovnih nalogov, oziroma v pravilo WSPT za zakasnele naloge.

2.2 Uporaba zahtevnega prednostnega pravila

Podrobneje obravnavamo pravilo stroškov kasnitve z upoštevanimi pripravljalno-zaključnimi časi (ATCS). Pravilo je narejeno za problem $1|s_{jk}|\sum w_j T_j$. Cilj je minimizirati vsoto uteženih kasnitev, s tem da so pripravljalni in končni časi odvisni od zaporedja delovnih nalogov. Posledično

where: k – empirically defined parameter (look ahead),
$\overline{p}$ – mean processing time of the remaining jobs.

If k is very large, the ATC rule narrows into the WSPT rule, and if it is very small it narrows into the LS rule, if there are no overdue jobs it narrows into the WSPT rule for the overdue jobs.

2.2 Application of a complex priority rule

We observe the rule of tardiness costs with considered setup times (ATCS). The apparent tardiness cost with setups (ATCS) rule is designed for the $1|s_{jk}|\sum w_j T_j$ problem. The objective is to minimize the sum of the weighted tardinesses, but now the jobs are subject to sequence-dependent setup je prednost vsakega delovnega naloga j odvisna od delovnega naloga, ki je bil pravkar končan na stroju. Pravilo ACTS je kombinirano; WSPT + LS + LSS na podlagi ene vrednosti (skalar). Izračun vrednosti delovnega naloga j v času t, ko je bil delovni nalog l končan na stroju, prikazuje naslednja enačba:

times. This implies that the priority of any job j depends on the job just completed on the machine just freed. The ACTS rule combines the WSPT rule, the LS rule, and the LSS rule in a single ranking index. The rule calculates the index of job j at time t when job l has completed its processing on the machine:

$$I_j(t,l)=\frac{w_j}{p_j}\cdot e^{-\frac{\max(d_j-p_j-t,0)}{k_1\bar{p}}}\cdot e^{-\frac{s_{lj}}{k_2\bar{s}}} \quad (8),$$

kjer so: $\bar{s}$ – povprečna vrednost pripravljalnih in končnih časov preostalih delovnih nalogov, ki čakajo na razvrščanje,
k_1 – parameter, odvisen od dobavnih rokov,
k_2 – parameter, odvisen od pripravljalnih in končnih časov.

Parametra k_1 in k_2 sta funkciji treh faktorjev:
- faktorja ozkosti dobavnih rokov:

where: $\bar{s}$ – the mean of the setup times of the remaining jobs,
k_1 – the due-date-related scaling parameter,
k_2 – the setup-time-related scaling parameter.

The parameters k_1 and k_2 are functions of three factors:
- the due-date tightness factor:

$$\tau=1-\frac{\bar{d}}{\hat{C}_{max}} \quad (9),$$

- faktorja razpona dobavnih rokov:

- the due-date range factor:

$$R=\frac{d_{max}-d_{min}}{\hat{C}_{max}} \quad (10),$$

- faktorja obsega pripravljalnih in končnih časov:

- the setup-time severity factor:

$$\eta=\frac{\bar{s}}{\bar{p}} \quad (11).$$

Tudi skupni čas izvedbe nalogov na enem stroju je odvisen od razvrstitve zaradi pripravljalnih in končnih časov. Preprosto ocenitev proizvodnega časa vseh n nalogov na enem stroju prikazuje naslednja enačba:

Even with a single machine the makespan is now schedule dependent because of the setup times. A simple estimate for the makespan on a single machine is the following:

$$\hat{C}_{max}=\sum_{j=1}^{n}p_j+n\cdot\bar{s} \quad (12).$$

Ta ocenitev bo najverjetneje precenila skupni proizvodni čas zaradi tega, ker bo končno razvrščanje izkoristilo pripravljalne in končne čase, da bodo nižji od povprečnih. Definiciji τ in R morata upoštevati omenjeno ocenitev skupnega proizvodnega časa. Poizkusne študije pravila ATCS predlagajo izbor parametrov k_1 in k_2 takole:

This estimate will most likely overestimate the makespan because the final schedule will take advantage of the setup times, which are lower than average. The definitions of τ and R have to be modified by replacing the makespan with its estimate. An experimental study of the ATCS rule has suggested the selection of parameters k_1 and k_2:

$$k_1=4,5+R,\ R<0,5$$
$$k_1=6-2\cdot R,\ R\geq 0,5 \quad (13)$$

$$k_2=\frac{\tau}{2\sqrt{\eta}} \quad (14).$$

2.2.1 Primer razvrščanja s pravilom ATCS

Za primer vzemimo $1|s_{jk}|\sum w_jT_j$ za štiri delovne naloge, katerih normativni časi, roki dobave in uteži so prikazani v preglednici 1.

2.2.1 Example of scheduling with the ACTS rule

Consider an instance of $1|s_{jk}|\sum w_jT_j$ with the four jobs in Table 1 (the processing times, due dates and weights are presented).

Preglednica 1. *Štirje delovni nalogi (osnovni podatki)*
Table 1. *Four jobs (basic data)*

Nalog Job	1	2	3	4
p_j	13	9	13	10
d_j	12	37	21	22
w_j	2	4	2	5

Preglednica 2. *Pripravljalni in končni časi prvega naloga v zaporedju*
Table 2. *Setup times for the first job in the sequence*

Nalog Job	1	2	3	4
s_{0j}	1	1	3	4

Preglednica 3. *Od zaporedja odvisni pripravljalni in končni časi*
Table 3. *The sequence-dependent setup times*

Nalog Job	1	2	3	4
s_{1j}	-	4	1	3
s_{2j}	0	-	1	0
s_{3j}	1	2	-	3
s_{4j}	4	3	1	-

Pripravljalni in končni časi s_{0j} za prvi delovni nalog v zaporedju so predstavljeni v preglednici 2.

Od zaporedja odvisni pripravljalni in končni časi nalogov, ki sledijo prvemu, so prikazani v tretji preglednici.

Za uporabo pravila ATCS moramo določiti povprečni normativni čas $\overline{p}$ in povprečni pripravljalni in zaključni čas $\overline{s}$. Povprečni normativni čas je 11,25, povprečni pripravljalni in končni čas pa 2. Ocena izdelovalnega časa je podana z enačbo:

The setup times s_{0j} of the first job in the sequence are presented in Table 2.

The sequence-dependent setup times of the jobs following the first job are shown in Table 3.

To use the ATCS rule the mean processing time $\overline{p}$ and the mean setup time $\overline{s}$ have to be determined. The mean processing time is 11.25 and the mean setup time is 2. The estimate for the makespan is in the following equation:

$$\hat{C}_{max} = \sum_{j=1}^{n} p_j + n\cdot\overline{s} = 45 + 4\cdot 2 = 53 \qquad (15),$$

kjer so:
- faktor razpona dobavnih rokov $R = 25/53 \approx 0,47$,
- faktor ozkosti dobavnih rokov $\tau = 1 - 23/53 \approx 0,57$,
- faktor obsega pripravljalno-zaključnih časov pa je $\eta = 2/11,25 \approx 0,18$.

Z uporabo enačb (13) in (14) določimo parameter $k_1 = 5$ in parameter $k_2 = 0,7$ (zaokroženo). Da določimo, kateri delovni nalog bo prvi, moramo izračunati $I_j(0, 0)$ za vse $j = 1, \ldots, 4$.

where:
- the due-date range factor $R = 25/53 \approx 0.47$,
- the due-date tightness coefficient $\tau = 1 - 23/53 \approx 0.57$,
- the setup-time severity coefficient is $\eta = 2/11.25 \approx 0.18$.

Using the Equations (13) and (14), the parameter k_1 is 5 and the parameter k_2 is 0.7 (rounded). To determine which job goes first, $I_j(0, 0)$ has to be calculated for $j = 1, \ldots, 4$.

$$I_1(0,0) = \frac{2}{13}\cdot e^{-\frac{\max(12-13,0)}{56,25}}\cdot e^{-\frac{1}{1,4}} \approx 0,15\cdot 1\cdot 0,51 \approx 0,075$$

$$I_2(0,0) = \frac{4}{9}\cdot e^{-\frac{\max(37-9,0)}{56,25}}\cdot e^{-\frac{1}{1,4}} \approx 0,44\cdot 0,61\cdot 0,51 \approx 0,137$$

$$I_3(0,0)=\frac{2}{13}\cdot e^{-\frac{\max(21-13,0)}{56,25}}\cdot e^{-\frac{3}{1,4}}\approx 0,15\cdot 0,87\cdot 0,103=0,013$$

$$I_4(0,0)=\frac{5}{10}\cdot e^{-\frac{\max(22-10,0)}{56,25}}\cdot e^{-\frac{4}{1,4}}\approx 0,50\cdot 0,81\cdot 0,057\approx 0,023$$

Iz zgornjih enačb je razvidno, da ima delovni nalog 2 največjo prednost. Ker je njegov pripravljalni in končni čas 1, je njegov čas dokončanja 10 (1 do 9). V drugem ponavljanju morajo biti izračunani $I_1(10, 2)$, $I_3(10, 2)$ in $I_4(10, 2)$. Da poenostavimo izračun, lahko vrednosti $k_1 \cdot \bar{p}$ in $k_2 \cdot \bar{s}$ pustimo enake (vmes je nalog 2 že izločen). Z nadaljnjo uporabo pravila ATCS je končni rezultat zaporedje nalogov (2,4,3,1) in vsota uteženih kasnitev enaka 98. Preračun vseh možnosti kaže, da je to zaporedje optimalno. Opazimo lahko, da je vselej izbran delovni nalog z najmanjšim pripravljalnim in končnim časom glede na predhodni nalog.

3 OMEJENO ISKANJE V SNOPU

Ta metoda temelji na zamisli "razvejaj in omeji". Preštevne metode "razvejaj in omeji" so trenutno najbolj razširjene za iskanje optimalnih rešitev za probleme "NP-hard" pri razvrščanju. Glavna pomanjkljivost teh metod je v tem, da po navadi porabijo ogromno časa, saj je lahko število obravnavanih vozlišč zelo veliko.

Za primer vzemimo stroj z n čakajočimi delovnimi nalogi. Predpostavimo, da so bili za vsako vozlišče na ravni k izbrani delovni nalogi za prvih k leg. Na ravni 0 je eno vozlišče z n vejami do n vozlišč na ravni 1. Vsako vozlišče na ravni 1 je razvejano v n-1 vozlišč na ravni 2, kar pomeni skupno $n(n$-1) vozlišč na ravni 2. Na ravni k je $n!/(n$-$k)!$ vozlišč. Na najnižji ravni, to je na n, je število vozlišč $n!$. Metoda "razvejaj in omeji" skuša izločiti vozlišča z določanjem spodnje meje doseganja cilja (ciljne funkcije) za vse delne razvrstitve, ki rastejo iz danega vozlišča. V primeru, da je spodnja meja višja od vrednosti ciljne funkcije za že znano razvrstitev, je moč vozlišče izločiti in se njegov "potomec" ne upošteva. Če dosežemo predhodno dokaj dobro razvrstitev nalogov z uporabo določene hevristične metode, preden se odločimo za uporabo metode "razvejaj in omeji", je mogoče tako izločiti mnogo vozlišč. Kljub postopku izločanja pa ponavadi ostane vseeno preveč vozlišč za preračun. Prednost "razvejaj in omeji" je v tem, da smo po preračunu po vseh vozliščih prepričani, da je rešitev optimalna.

From the calculation we can see that job 2 has the highest priority. Because its setup time is 1, its completion time is 10 (1 to 9). At the second iteration, $I_1(10, 2)$, $I_3(10, 2)$ and $I_4(10, 2)$ have to be calculated. To simplify the computations the values of $k_1 \cdot \bar{p}$ and $k_2 \cdot \bar{s}$ can be kept the same (job 2 has been already completed). Continuing the application of the ATCS rule results in the sequence (2,4,3,1) with the sum of the weighted tardinesses equal to 98. Complete enumeration shows that this sequence is optimal. Note that this sequence always selects, whenever the machine is freed, one of the jobs with the smallest setup time.

3 THE FILTERED-BEAM SEARCH

This method is based on the ideas of branch and bound. Enumerative branch-and-bound methods are currently the most widely used methods for obtaining optimal solutions to "NP-hard" scheduling problems. The main disadvantage of branch and bound is that it is usually extremely time consuming, because the number of nodes one must consider is very large.

Consider, for example, a single machine problem with n jobs. Assume that for each node at level k, jobs have been selected for the first k positions. There is a single node at level 0, with n branches emanating from it to n nodes at level 1. Each node at level 1 branches out into n-1 nodes at level 2, resulting in a total of $n(n$-1) nodes at level 2. At level k, there are $n!/(n$-$k)!$ nodes. At the bottom level, level n, there are $n!$ nodes. The branch-and-bound method attempts to eliminate a node by determining a lower bound on the objective for all the partial schedules that sprout out of that node. If the lower bound is higher than the value of the objective under a known schedule, then the node may be eliminated and its offspring disregarded. If one could obtain a reasonably good schedule through some clever heuristic before going through the branch-and-bound procedure, then it might be possible to eliminate many nodes. Even after these eliminations there are usually still too many nodes to be evaluated. The main advantage of branch and bound is that, after evaluating all the nodes, the final solution is known with certainty to be optimal.

Omejeno iskanje v snopu pomeni prilagoditev metode "razvejaj in omeji", saj ne upoštevamo vseh vozlišč po ravneh. Za razvejanje se upoštevajo samo najbolj obetavna vozlišča na ravni *k*, preostala na tem ravni trajno zbrišemo. Število vozlišč, ki jih obdržimo, pomeni širino snopa izbora. Ključna sestavina metode je način vrednotenja obetavnosti posameznih vozlišč. Če izločanju posvetimo malo časa, je metoda hitra, lahko pa nas stane dobrega rezultata, saj lahko izločimo obetavna vozlišča. Na drugi strani pa preveč previdno izločanje vozlišč pomeni veliko časovno potratnost postopka. V ta namen se uporablja filter. Za vsa vozlišča na ravni *k* naredimo grobo napoved. Na njenih predvidevanjih izberemo vozlišča za nadaljnji preračun, preostala pa zbrišemo. Število izbranih vozlišč za preračun označuje širino filtra. Na podlagi preračuna izbranih vozlišč izluščimo skupino vozlišč (toliko kolikor znaša širina snopa, nikakor pa ne več ko širina filtra), iz katerih bomo izvedli nadaljnje razvejanje.

3.1 Primer

Za primer vzemimo $1||\sum w_j T_j$, torej z enakim merilom kakor v prejšnjem primeru (preglednica 4).

Ker je število delovnih nalogov majhno, naredimo samo en tip napovedi za vozlišča na katerikoli ravni. Uporabili nismo nobenega sistema filtriranja. Za širino snopa izberemo 2, kar pomeni, da na vsaki ravni obdržimo dve vozlišči. Napoved na vozlišču naredimo z razvrščanjem delovnih nalogov s pravilom ATC (ni razlikovanja pripravljalnih in končnih časov). S faktorjem razpona dobavnih rokov $R = 11/37$ in s faktorjem ozkosti dobavnih rokov $\tau \approx 32/37$ (glej enačbi (9) in (10)), izberemo vrednost parametra pogleda naprej $k = 5$.

The filtered-beam search is an adaptation of branch and bound in which not all nodes at any given level are evaluated. Only the most promising nodes at level *k* are selected as nodes to branch from. The remaining nodes at that level are discarded permanently. The number of nodes retained is called the beam width of the search. The evaluation process that determines which nodes are the promising ones is a crucial component of this method. Evaluating each node carefully, to obtain an estimate for the potential of its offspring, is time consuming. There is a trade-off here: a crude prediction is quick but may lead to good solutions being discarded, whereas a more thorough evaluation may be prohibitively time consuming. Here is where the filter comes in. For all the nodes generated at level *k*, a crude prediction is made. Based on the outcome of these crude predictions, a number of nodes are selected for a thorough evaluation, and the remaining nodes are discarded permanently. The number of nodes selected for a thorough evaluation is referred to as the filter width. Based on the outcome of a careful evaluation of all the nodes that pass the filter, a subset of these nodes (the number being equal to the beam width, which, therefore, cannot be greater than the filter width) is selected, from which further branches will be generated.

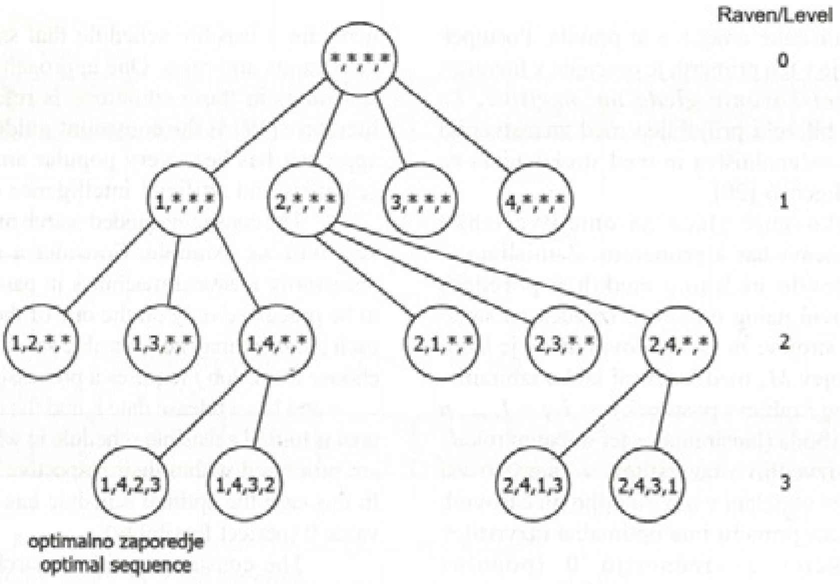
3.1 Example

Consider the instance of $1||\sum w_j T_j$, therefore with the same objective as in the previous example (Table 4).

Because the number of jobs is rather small, only one type of prediction is made for the nodes at any particular level. No filtering mechanism is used. The beam width is chosen to be 2, which implies that at each level only two nodes are retained. The prediction at a node is made by scheduling the unscheduled jobs according to the ATC rule. With the due-date range factor $R = 11/37$ and the due-date tightness factor $\tau \approx 32/37$ (Equations (9) and (10)), the look-ahead parameter *k* is chosen to be 5.

Preglednica 4. *Podatki za štiri naloge*
Table 4. *Data for four jobs*

Nalog Job	1	2	3	4
p_j	10	10	13	4
d_j	4	2	1	12
w_j	14	12	1	12



Sl. 2. *Omejeno iskanje v snopu za* $1\|\sum w_jT_j$

Fig. 2. Beam search applied to $1\|\sum w_jT_j$

Oblikujemo drevo "razvejaj in omeji", s predpostavko, da je zaporedje razvito, z začetkom v trenutku $t = 0$. Tako so na j-ti ravni drevesa delovni nalogi dani v j-to lego. Na 1. ravni drevesa imamo štiri vozlišča: (1,*,*,*), (2,*,*,*), (3,*,*,*) in (4,*,*,*), kar prikazuje slika 2. Z uporabo pravila ATC za preostale tri delovne naloge na vsakem od štirih vozlišč izražajo naslednja štiri zaporedja: (1,4,2,3), (2,4,1,3), (3,4,1,2) in (4,1,3,2) z vrednostmi ciljne funkcije 408, 436, 814 in 440. Ker je širina snopa 2, obdržimo le prvi dve vozlišči.

Vsako izmed teh dveh vozlišč se razveja v nadaljnja tri vozlišča na ravni 2. Vozlišče (1,*,*,*) se razveja v (1,2,*,*), (1,3,*,*) in (1,4,*,*), vozlišče (2,*,*,*) pa v (2,1,*,*), (2,3,*,*) in (2,4,*,*). Z uporabo pravila ATC za preostala delovna naloga v vsakem od šestih vozlišč na drugi ravni dobimo boljše rezultate v vozliščih (1,4,*,*) in (2,4,*,*), ki ju v nadaljevanju obdržimo, ostala štiri pa opustimo.

Dve vozlišči na ravni 2 se razvejata v štiri vozlišča na ravni 3 (zadnja stopnja), (1,4,3,2), (1,4,2,3), (2,4,1,3) in (2,4,3,1). Od teh štirih zaporedij je najboljše zaporedje (1,4,2,3) s skupno uteženo kasnitvijo 408. To zaporedje je optimalno.

A branch-and-bound tree is constructed with the assumption that the sequence is developed, starting from $t = 0$. So, at the j-th level of the tree the jobs are put into the j-th position. At level 1 of the tree there are four nodes: (1,*,*,*), (2,*,*,*), (3,*,*,*) and (4,*,*,*), see Figure 2. Using the ATC rule for the remaining jobs at each one of four nodes results in four sequences: (1,4,2,3), (2,4,1,3), (3,4,1,2) and (4,1,2,3) with objective values of 408, 436, 814 and 440. Because the beam width is 2, only the first two nodes are retained.

Each of these two nodes leads to three nodes at level 2. Node (1, *, *, *) leads to nodes (1,2,*,*), (1,3,*,*) and (1,4,*,*), and node (2,*,*,*) leads to nodes (2,1,*,*), (2,3,*,*) and (2,4,*,*). Applying the ATC rule to the remaining two jobs in each one of the six nodes at level 2 results in nodes (1,4,*,*) and (2,4,*,*) being retained and the remaining four being discarded.

Two nodes at level 2 lead to four nodes at level 3 (the last level), (1,4,3,2), (1,4,2,3), (2,4,1,3) and (2,4,3,1). Of these four sequences, sequence (1,4,2,3) is the best with a total weighted tardiness equal to 408. This sequence is optimal.

4 HEVRISTIČNO PREISKOVANJE GLEDE NA OMEJITVE

V mnogih dejanskih primerih ni pravega cilja. Zahtevamo le izvedljivo razvrstitev, ki

4 THE CONSTRAINT-GUIDED HEURISTIC SEARCH

In many real-world situations there is not really an objective. Rather, it is only required to izpolnjuje določene omejitve in pravila. Postopek za razvrščanje v teh primerih je omenjen v literaturi [19] kot *preiskovanje glede na omejitve*. Ta postopek je bil zelo priljubljen med znanstveniki na področju računalništva in med strokovnjaki za umetno inteligenco [20].

Preiskovanje glede na omejitve lahko najbolje opišemo kar s primerom. Zamislimo si določeno število ne nujno enakih vzporednih strojev. Delovni nalog mora biti izveden na samo enem izmed strojev; za vsak delovni nalog je izbor mogočih strojev M_j, med katerimi lahko izbiramo. Delovni nalog *j* zahteva postopek $p_j = 1, j = 1, ..., n$ in ima čas prihoda (lansiranja) r_j ter dobavni rok d_j. Cilj je najti izvedljivo razvrstitev, v kateri so vsi delovni nalogi obdelani v okviru njihovih časovnih okvirov. V tem primeru ima optimalna razvrstitev ciljno funkcijo z vrednostjo 0 (popolna izvedljivost).

Preiskovanje glede na omejitve se lahko izvaja na podlagi naslednjih pravil: vselej razvrstimo en delovni nalog. Ko je delovni nalog razvrščen, je dodeljen določenemu stroju, ki je v tem času prost; izbrana pravila določajo, kdaj je določen delovni nalog dan v obdelavo na nekem stroju. Delovni nalogi so lahko razporejeni glede na kritičnost ali prilagodljivost; tisti z najmanjšo prilagodljivostjo je najbolj kritičen in se izvaja prvi. Poznamo več postopkov merjenja prilagodljivosti nalogov (po časovni zračnosti, po številu mogočih strojev itn.). Tudi stroje izbiramo po določenih pravilih (npr. po prilagodljivosti, merjenje po številu nalogov, ki jih stroj lahko obdela; najprej izbiramo tiste z najmanjšo prilagodljivostjo) [21].

Pomembno načelo preiskovanja glede na omejitve je tako imenovana razširitev omejitev. Dodelitev določenega delovnega naloga k danemu časovnemu koraku na stroju ima vpliv na dodeljevanje preostalih delovnih nalogov na tem stroju in na drugih strojih. Ti vplivi lahko kažejo na kršitev omejitev in povzročijo, da ostane del preiskovanega prostora neupoštevan.

4.1 Primer

Za primer vzemimo $P3|r_j, p_j = 1, M_j|\sum U_j$, $U_j = 1$, če nalog kasni, drugače je $U_j = 0$. Imamo tri stroje in devet delovnih nalogov. Časi obdelave, časi prihoda nalogov in dobavni roki so predstavljeni v preglednici 5. Neskončni čas (∞) pomeni, da naloga na danem stroju ni mogoče

generate a feasible schedule that satisfies various constraints and rules. One approach for generating schedules in these situations is referred to in the literature [19] as the constraint-guided search. This approach has been very popular among computer scientists and artificial intelligence experts [20].

The constraint-guided search may be described best with an example. Consider a number of not necessarily identical machines in parallel. A job has to be processed only on the one of the machines; for each job there may be a feasible set of machines M_j to choose from. Job *j* requires a processing $p_j = 1, j = 1, ..., n$ and has a release date r_j and the due date d_j. The goal is to find a feasible schedule in which all the jobs are processed within their respective time windows. In this case the optimal schedule has an objective of value 0 (perfect feasibility).

The constraint-guided search may operate according to the following rules. Jobs are scheduled one at a time. When a job is scheduled, it is assigned to a specific time slot on a specific machine, which is still free. During each iteration an unassigned job is selected according to a set of job rules that have been arranged in some priority. The job rules specify whether the particular job actually can be processed on a given machine. The jobs can be ordered according to their criticality or flexibility; the job with the least flexibility is the most critical and has the highest priority. The flexibility of a job can be measured in several ways (flexibility in time – slack time, flexibility with regard to the number of appropriate machines, etc.). The machines can also be ordered in such a way that the machine with the least flexibility has the highest priority (the flexibility of a machine, measured by the number of jobs that can be processed on the machine) [21].

An important concept in constraint-guided search is constraint propagation. The assignment of a particular job to a given time slot on a given machine has implications with regard to the assignment of other jobs on the given machine and on other machines. These implications may point to the violation of hard constraints and may indicate that the associated part of the search space can be disregarded.

4.1 Example

Consider the problem $P3|r_j, p_j = 1, M_j|\sum U_j$, $U_j = 1$, if $C_j > d_j$, otherwise $U_j = 0$. We have three machines and nine jobs. The processing times, release dates, and due dates of the job are presented in Table 5. If the processing time of a job on a machine is infinity (∞), then the job cannot be processed on that

Preglednica 5. *Podatki za devet delovnih nalogov*
Table 5. *Data for nine jobs*

Nalog Job	1	2	3	4	5	6	7	8	9
p_{1j}	∞	1	1	∞	1	∞	∞	1	1
p_{2j}	1	1	∞	1	1	1	∞	1	1
p_{3j}	1	1	∞	1	1	∞	1	∞	1
r_j	1	1	0	0	0	0	1	1	2
d_j	3	2	1	2	1	1	3	3	3

Preglednica 6. *Faktor prilagodljivosti devetih delovnih nalogov*
Table 6. *Flexibility factor of nine jobs*

Nalog Job	1	2	3	4	5	6	7	8	9
Φ_j	4	3	1	4	3	1	2	4	3

Preglednica 7. *Faktor prilagodljivosti časovnega koraka nalogov*
Table 7. *Flexibility factor of a job timeslot*

Stroj Machine	1	1	1	2	2	2	3	3	3
časovni korak timeslot	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)	(3,1)	(3,2)	(3,3)
$\Phi_{(i,l)}$	2	2	2	3	4	3	2	4	3

Preglednica 8. *Dodelitev nalogov časovnim korakom*
Table 8. *Assignment of jobs to timeslots*

Stroj Machine	1	1	1	2	2	2	3	3	3
časovni korak timeslot	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)	(3,1)	(3,2)	(3,3)
$\Phi_{(i,l)}$	2	2	2	3	4	3	2	4	3

obdelati. Cilj je najti izvedljivo razvrstitev, kjer bodo vsi nalogi pravočasno končani ($\sum U_j = 0$).

Imamo devet korakov, po tri na vsakem stroju. Vsak nalog ima svoj časovni okvir, kar pomeni omejitve. Za vsak delovni nalog lahko izračunamo faktor prilagodljivosti Φ_j. V našem primeru je izračunan kot število časovnih korakov, katerim je lahko delovni nalog dodeljen na različnih strojih. Predstavljen je v preglednici 6.

Namesto prilagodljivosti stroja definiramo prilagodljivost časovnega koraka na stroju, katero predstavlja število delovnih nalogov, ki so lahko obdelani v danem koraku. Faktor prilagodljivosti $\Phi_{(i,l)}$ koraka (i, l) predstavimo v spodnji preglednici. Korak (i, l) je l-ti korak na i-tem stroju.

Zaporedje nalogov po povečani prilagodljivosti je: 3, 6, 7, 2, 5, 9, 1, 8, 4. Prednosti časovnih korakov dajejo (možen) vrstni red (1,3), (3,1), (1,1), (1,2), (2,1), (2,3), (3,3), (3,2), (2,2).

machine. The goal is to find a feasible schedule with all the jobs completed on time ($\sum U_j = 0$).

There are nine timeslots and three on each machine. Each job has its own time window, which represents a set of constraints. For each job a flexibility factor Φ_j can be calculated. In this example: Φ_j is the number of timeslots to which a job may be assigned on the various machines. The flexibility factor Φ_j of job j is presented in Table 6.

Instead of the flexibility of a machine, the flexibility of a timeslot on a machine is determined. It is defined as the number of jobs that can be processed during the timeslot. The flexibility factor $\Phi_{(i,l)}$ of the job timeslot (i, l) is presented in Table 7.

The sequence of jobs in increasing flexibility results in: 3, 6, 7, 2, 5, 9, 1, 8, 4. The priorities of the timeslots may result in the sequence (1,3), (3,1), (1,1), (1,2), (2,1), (2,3), (3,3), (3,2), (2,2).

Delovni nalog 3 je izbran najprej. V koraku (1,3) ga ni mogočo obdelati (prepozno), zato preverjamo naprej. V (3,1) tudi ne gre (3. stroj), tako je dodeljen koraku (1,1). Delovni nalog 6 je ob upoštevanju omejitev dodeljen koraku (2,1). Končni rezultati so prikazani v preglednici 8. Vsem omejitvam je zadoščeno.

Ko je neki delovni nalog razvrščen, se lahko faktor prilagodljivosti preostalim delovnim nalogom in časovnim korakom spremeni. Zato je mogoče uporabiti nove prednosti glede na nove faktorje prilagodljivosti. V podanem primeru tega nismo uporabili.

Preiskovanje glede na omejitve ne da vedno izvedljive rešitve že po prvem koraku. Lahko se zgodi, da pri zadnjem delovnem nalogu ni mogoča nobena izvedljiva dodelitev. V tem primeru se mora metoda nasloniti na kasnejši postopek, ki lahko najde izvedljivo rešitev z izmenjavami parov nalogov.

Job 3 is selected first. It is checked to determine whether it is allowed to be processed in the timeslot (1, 3). It is not (too late). The next timeslot is tried (3,1) – not on the 3rd machine, and so on, until a timeslot is found during which it is allowed to be processed. Job 3 is then assigned to slot (1, 1). Job 6 is considered in the same manner and assigned to slot (2, 1). Continuing in this manner results in the following assignment – see Table 8. All the constraints are fulfilled.

After a job has been assigned, the flexibility factors of the remaining jobs to be assigned and the remaining timeslots available may change. It would have been possible to reorder the remaining jobs, as well as the remaining timeslots, based on the new flexibility factors. In the example above this was not done.

The constraint-guided search does not always yield a feasible solution after the first pass. It may happen that when the last job has to be assigned, no feasible assignment is possible. In this case, the method has to rely on a post-processing procedure, which, through pairwise interchanges, attempts to construct a feasible solution.

5 SKLEP

Problemi terminiranja spadajo na področje optimizacije. Ko se posvetimo takšnemu problemu, moramo vedno iskati njegovo zahtevnost, ker le-ta določa naravo algoritma, katerega naj bi uporabili pri reševanju. Za zahtevna razvrščanja v praksi po navadi narava problema zahteva neko svojo rešitev, zato lahko ustvarimo postopke, ki so kombinacija več predstavljenih načinov in tehnik v tem prispevku. Novejši postopki pridružujejo še simulacijsko tehniko. Ker potrebne računske zmogljivosti rastejo vsaj eksponentno z velikostjo problema terminiranja, smo prisiljeni v podoptimalno reševanje, kar pomeni: v "sprejemljivo kratkem" času dobiti "dovolj dobro" rešitev (blizu optimalni). V svetu je bilo v zadnjem desetletju razvitih (v industriji in v znanosti) na stotine sistemov za terminiranje, a nerešeni splošni problemi ostajajo. Prispevek poudarja možnosti za uporabo naprednih metod terminiranja.

Sistemi terminiranja bodo v prihodnosti temeljili na: simulaciji, umetni inteligenci, grafičnih predstavitvah, poenostavljanju, lastni organizaciji, mehkih podatkih in mehki logiki.

5 CONCLUSION

Scheduling problems belong to the field of optimisation. When we focus on such a problem, we must always determine its complexity, which defines the nature of the algorithm that is applicable for finding a solution. For many hard-scheduling problems with specific solutions one may design procedures that combine elements of several presented techniques in the paper. Newer approaches also associate a simulation technique. Because of the at least exponential growth of the computational capabilities with the size of a scheduling problem, we are forced to use the sub-optimal solving: in an acceptable short time we must get a sufficient (near-optimal) solution. Over the past decade, hundreds of scheduling systems have been developed in industry and academia, but there are still unsolved general problems. In the paper the application of advanced scheduling methods is emphasized.

In the future scheduling systems will be based on the following: simulation, artificial intelligence, graphical presentation, simplifications, self-organisation, fuzzy data and fuzzy logic.

6 LITERATURA
6 REFERENCES

[1] Carlier, J. and P. Chretiénne (1988) Problèmes d'ordonnancement: modelisation / complexité / algorithmes. *Masson*, Paris.

[2] Johnson, S. M. (1954) Optimal two and three stage production schedules with set-up time included, *Naval Research Logistics Quarterly*, Vol. 1, pp. 61-68.

[3] Conway, R. W., W. L. Maxwell and L. W. Miller (1967) Theory of scheduling. *Addison-Wesley*, Reading.

[4] Lawler, E. L., A. H. G. Rinnooy Kan and B. Lageweg (1975) Minimizing total costs in one-machine scheduling, *Operations Research*, Vol. 23, pp. 908-927.

[5] Brah, S. A. and G. E. Wheeler (1998) Comparison of scheduling rules in a flow shop with multiple processors: A simulation, *Simulation*, Vol. 71, No. 5, pp. 302-311.

[6] Polajnar, A., B. Buchmeister, M. Leber, K. Pandža, B. Kalpič, T. Rojs, N. Vujica-Herzog, I. Palčič, T. Fulder and P. Meža (2004) Menedžment proizvodnih sistemov (sodobni pristopi). *Fakulteta za strojništvo*, Maribor.

[7] Brucker, P. (1998) Scheduling algorithms. *Springer-Verlag*, Berlin.

[8] Buchmeister, B., Z. Kremljak, K. Pandža, A. Polajnar (2004) Simulation study on the performance analysis of various sequencing rules, *International Journal of Simulation Modelling (Int J Simul Model)*, Vol. 3, No. 2-3, pp. 80-89.

[9] Gomes, P. J. and L. C. Meile (2002) Leveraging the potential of process technology through workflow scheduling, *International Journal of Service Industry Management*, Vol. 13, No. 1, pp. 7-28.

[10] Blażewicz, J., K. H. Ecker, E. Pesch, G. Schmidt and J. Weglarz (2001) Scheduling computer and manufacturing processes, *Springer-Verlag*, Berlin.

[11] Garcia-Sabater, J. P. (2001) The problem of JIT dynamic scheduling. A model and a parametric procedure. *Proceedings of the ORP3 conference*, Paris, September 2001.

[12] Nyman, D. and J. Levitt (2001) Maintenance planning, scheduling and coordination. *Industrial Press*, New York.

[13] Leung, J. Y.-T. (2004) Handbook of scheduling: algorithms, models and performance analysis. *Chapman & Hall/CRC*, Boca Raton.

[14] Palmer, D. (2006) Maintenance planning and scheduling handbook. *McGraw-Hill*, New York.

[15] Koo, P.-H. and J. Jang (2002) Vehicle travel time models for AGV systems under various dispatching rules, *International Journal of Flexible Manufacturing Systems*, Vol. 14, pp. 249-261.

[16] T'kindt, V. and J.-C. Billaut (2002) Multicriteria scheduling. *Springer-Verlag*, Berlin.

[17] Glaser, H., W. Geiger and V. Rohde (1991) Produktionsplannung und –steuerung. *Gabler Verlag*, Wiesbaden.

[18] Ljubič, T. (2006) Operativni management proizvodnje. *Založba Moderna organizacija*, Kranj.

[19] Pinedo, M. L. (2005) Planning and scheduling in manufacturing and services, *Springer Science+Business Media*, New York.

[20] Kremljak, Z., A. Polajnar and B. Buchmeister (2005) A heuristic model for the development of production capabilities, *Journal of Mechanical Engineering*, Vol. 51, No. 11, pp. 674-691.

[21] Penker, A., M. C. Barbu and M. Gronalt (2007) Bottleneck analysis in MDF-production by means of discrete event simulation, *International Journal of Simulation Modelling (Int J Simul Model)*, Vol. 6, No. 1, pp. 49-57, doi:10.2507/IJSIMM06(1)5.084.

Naslov avtorjev: Tadej Tasič
prof. dr. Borut Buchmeister
prof. dr. Bojan Ačko
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova ulica 17
2000 Maribor
tadej.tasic@uni-mb.si
borut.buchmeister@uni-mb.si
bojan.acko@uni-mb.si

Authors' Address: Tadej Tasič
Prof. Dr. Borut Buchmeister
Prof. Dr. Bojan Ačko
University of Maribor
Faculty of Mechanical Eng.
Smetanova ulica 17
2000 Maribor, Slovenia
tadej.tasic@uni-mb.si
borut.buchmeister@uni-mb.si
bojan.acko@uni-mb.si

Prejeto: Received: 19.7.2007

Sprejeto: Accepted: 28.9.2007

Odprto za diskusijo: 1 leto
Open for discussion: 1 year