6. Imprecise Computations

6.1. Ausgangspunkt

- **Motivation**

  „Akzeptable“ Qualität bei weichen Echtzeit-Anwendungen mit deterministischen Zeitschranken, insbesondere bei Überlast.

  z.B. Bilddekodierung/-darstellung, Näherungsberechnungen
  Zielverfolgung

- **Grundidee**

  (periodische) Task aufteilen in Pflichtteil – Wahlteil

  Pflichtteil muß stets Deadline einhalten, Wahlteil nicht.
  Quantifizierung durch eine Fehlerfunktion (Gütefunktion).

- **Taskklassen**

  N-Tasks: möglichst kleiner mittlerer Fehler;

  C-Tasks: in bestimmten Abständen muß Deadline erreicht werden.

6.2. Task-Modell

- **Allgemeine Voraussetzungen und Bezeichnungen**

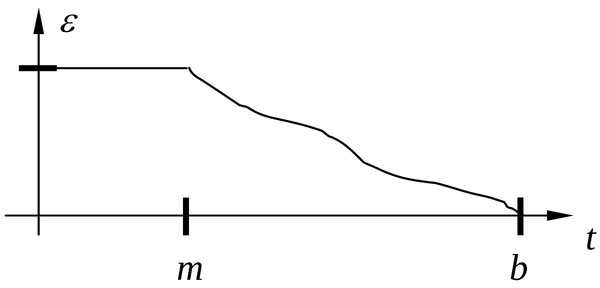
$T = \{\tau_1,\ldots,\tau_n\}$	periodische Tasks
$\tau_i = (\tau_{ij})_{j=1,2,\ldots}$	τ_{ij} : j-ter Job von Task τ_i, $i = 1,\ldots,n$
$\tau_i = (t_i, b_i)$	t_i: Periodenlänge
	b_i: Ausführungszeit (konstant)
m_i	Mindestbearbeitungszeit für τ_{ij}

Periodenanfang = Bereitzeit, Periodenende = Deadline

- **Fehlerfunktion und Einplanbarkeit**

e: einer Task zur Verfügung stehende Zeit (eingeplante Zeit)

$\varepsilon(e)$: Fehler bei $e < b$
monoton fallende Funktion



Exakter Ablaufplan: $e = b \quad \forall \tau \in T$

Ausführbarer Ablaufplan: $e \geq m \quad \forall \tau \in T$

- **Taskaufteilung**

$\tau_i = (t_i, b_i) \mapsto M_i = (t_i, m_i)$ Pflichtteil (mandatory part)

$O_i = (t_i, b_i - m_i)$ Wahlteil (optional part)

Gleiche Bereitzeit und Deadline wie τ_i (Periodenanfang/-ende).

Pflichtteil muß stets Deadline erreichen; Wahlteil wird bei Deadline (Periodenende) abgebrochen (N-Tasks: erreichtes Ergebnis wird gewertet).

6.3. Admission und Scheduling von N-Tasks

- **Aufgabe**

  M einplanen als „harte“ Tasks, O als „weiche“, so daß mittlerer Fehler nach Möglichkeit klein wird.

- **Admission**

  T wird zugelassen bei $$\sum_{i=1}^{n} \frac{m_i}{t_i} \leq n \cdot \left(\sqrt[n]{2} - 1\right).$$

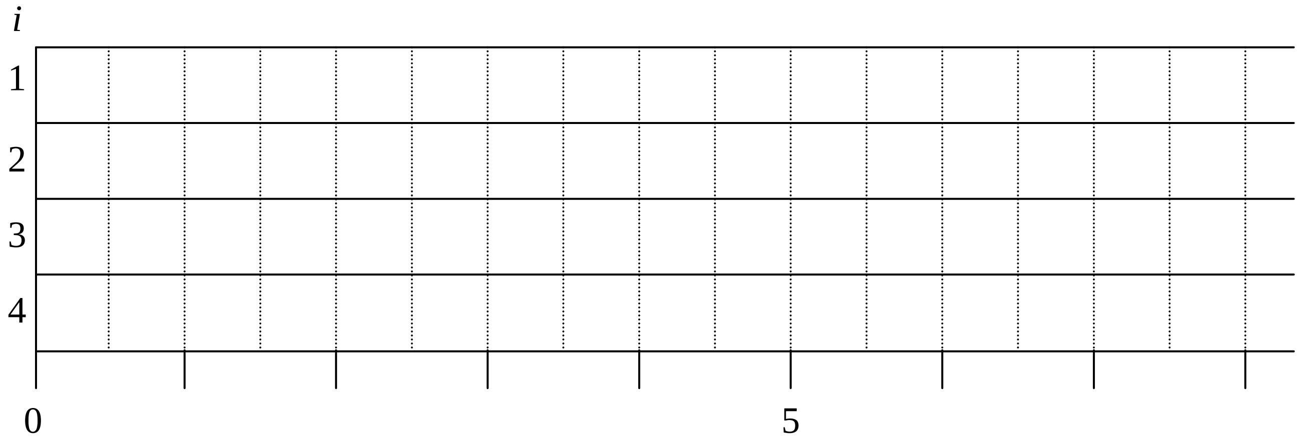
- **Scheduling von M-Jobs**

  2 Prioritätsklassen: „M-Priorität > O-Priorität“ bzgl. ganz T.

  Scheduling von M-Jobs gemäß RMS.

  Beispiel.

i	t_i	b_i	m_i	$U_i = \frac{b_i}{t_i}$	$u_i = \frac{m_i}{t_i}$	
1	2	1,0	0,5			
2	4	0,5	0,2			
3	5	0,5	0,1			
4	6	1,5	1,0			



- **Scheduling von O-Jobs**

LU ***least utilization first*** $v_i = \frac{b_i - m_i}{t_i}$

Beispiel.

Es gilt: Bei linearer Fehlerfunktion und gleichlangen Perioden führt LU zu minimalem mittlerem Fehler.

Aber: nicht optimal bzgl. Fehler bereits bei ungleichen Periodenlängen!

- **Dynamische Algorithmen für O-Jobs**

– ***LAT*** least attained time first = shortest elapsed time first

Motivation: konvexe Fehlerfunktion

– ***LST*** least slack time first slack = laxity

– ***ED*** earliest deadline

– ***SPL*** shortest period length

– ***BIR*** best incremental

6.4. Bewertung

- **Fehlerfunktion**

$$\varepsilon(e_{ij}) = \begin{cases} 1 & e_{ij} < m_i \\ \left(1 - \dfrac{e_{ij} - m_i}{b_i - m_i}\right)^d & e_{ij} \in [m_i, b_i] \\ 0 & e_{ij} > b_i \end{cases}$$

Treppenfunktion statt kontinuierlicher Fehlerfunktion: geringer Effekt.

- **Vergleiche**

– ***Identische Jobs***

d = 1: alle Algorithmen gleicher mittlerer Fehler. $U = \sum_{i=1}^{n} \frac{b_i}{t_i} \leq 1$: exakt.

d = 2: LAT, LST, BIR besser. d = 0,5 umgekehrt.

– ***Gleichlange Perioden***

d = 1: LU optimal bzgl. Fehler, aber ED, SPL genauso gut.

d = 2: LAT, BIR teilweise besser. Auch für d = 0,5.

– ***Harmonische Perioden***

LU recht gut, ED deutlich schlechter!

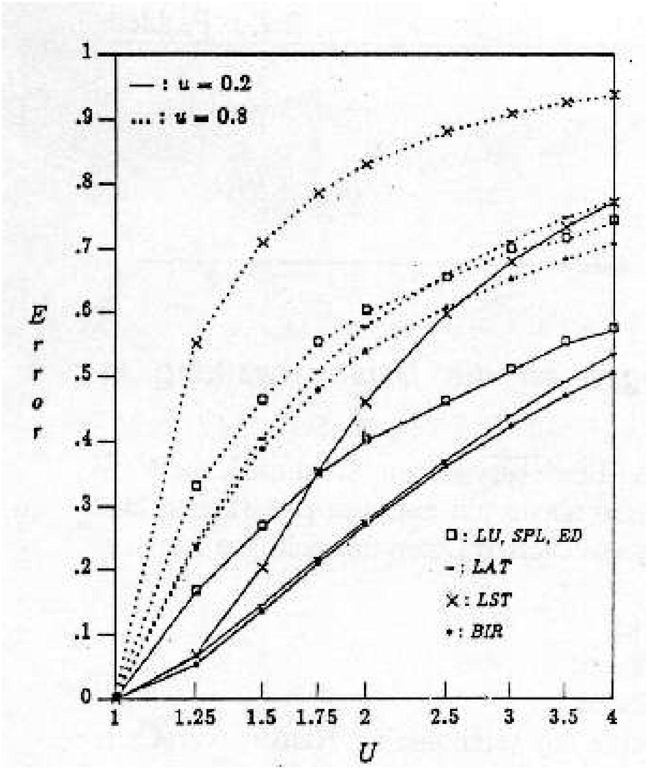
Allgemeiner Fall: ähnliches Verhalten.

- **Exakte Einplanbarkeit**

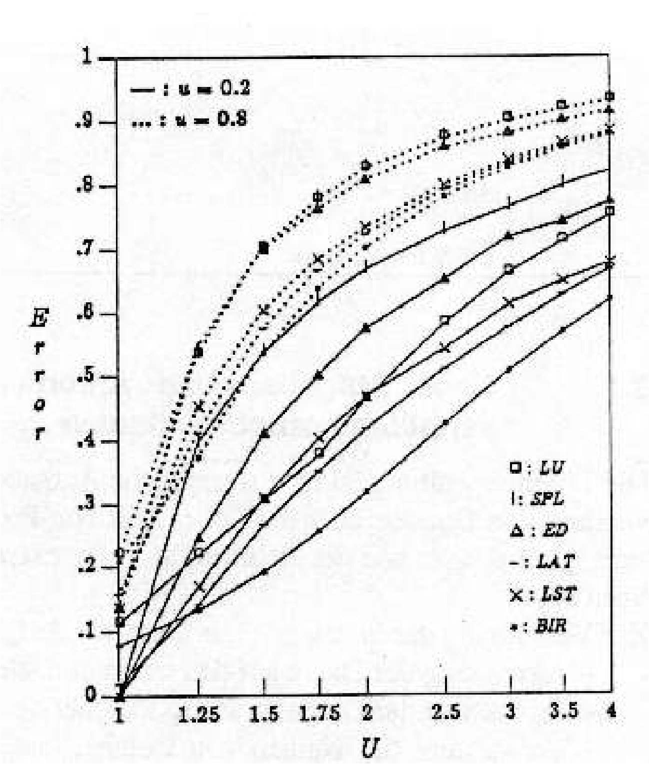
Für eine Taskmenge mit $U \leq 1$ ist ED optimal bzgl. Einplanbarkeit mit Fehler 0 unter allen Algorithmen, die die Pflichtteile mit RMS einplanen.

Gilt auch für LST, für die anderen Algorithmen jedoch nicht.

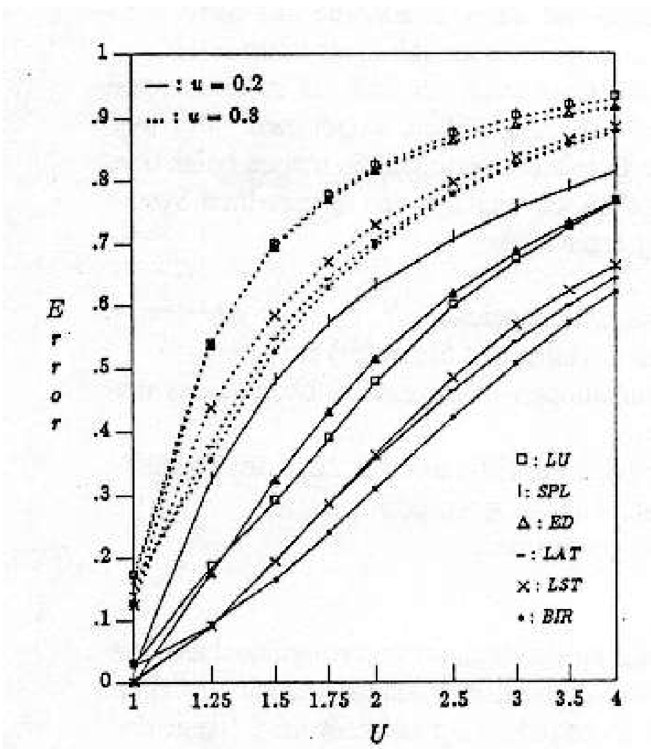
Fehlerfunktion: $d = 2$



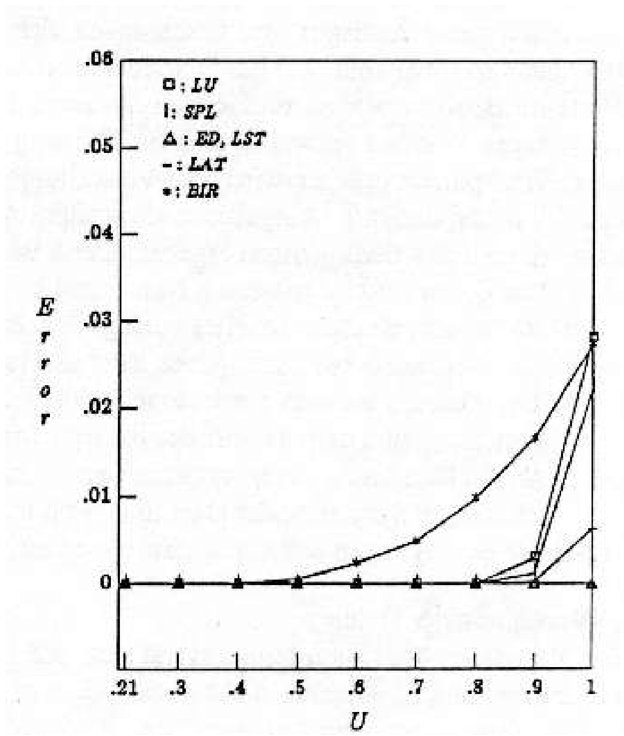
gleichlange Perioden



harmonische Perioden



beliebige Perioden



exakte Einplanbarkeit

6.5. Admission und Scheduling von C-Tasks

- **Fehlerfunktion**

$$\varepsilon(e) = \begin{cases} 1 & e \in [0, b) \\ 0 & e \geq b \end{cases}$$

- **Admission für $T = \{\tau_1,\dots,\tau_n\}$**

  (1) $e_i \geq m_i \quad \forall i = 1,\dots,n.$

  (2) Für mindestens einen Job unter Q_i aufeinander folgenden Jobs von τ_i ist $e_i \geq b_i$.

  Q_i: Qualitätsparameter („cumulation rate“)

- **Gleichlange Perioden *p* und einheitlicher Qualitätsparameter *Q***

  Suche nach ausführbarem Ablaufplan äquivalent zu:

  exakten Ablaufplan ohne Entzug (!) finden für die Wahlteile von T auf Q Prozessoren mit gemeinsamer Deadline $p - \sum_{i=1}^{n} m_i$.

  $\rightarrow$ NP-vollständig! $\rightarrow$ Heuristik.

- **„Längenmonotoner Algorithmus“**

  (1) Pflichtteile bearbeiten

  (2) Wahlteile fallend sortieren nach $b_i - m_i$

  (3) Wahlteile in jeder Periode gemäß first-fit einplanen;
  vollständig bearbeitete Wahlteile danach „bis Q_i“ übergehen!