Wiederholung

Nichtperiodische Aufgaben

Florian Franzmann Tobias Klaus Florian Korschin
Florian Schmaus Peter Wägemann

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)
https://www4.cs.fau.de

16. Dezember 2015

Nichtperiodische Aufgaben

- Definiert durch $T_i = (i_i, e_i, D_i)$
- *Aperiodische* vs. *sporadische* Aufgabe
- *Mischbetrieb*: periodisch ⟷ sporadisch/aperiodisch
  - *dynamische* Einplanung
  - Beeinflussung periodischer Aufgaben?
  - Übernahmeprüfung ⟷ Antwortzeitminimierung

Nichtperiodische Arbeitsaufträge

- Kaum a-priori Wissen (Zeitpunkt, WCET, . . .)
- Herausforderung Mischbetrieb: Erhaltung statischer Garantien
- Abweisung (spor. Aufg.): Schwerwiegende Ausnahmesituation

Basistechniken zur Umsetzung

- Unterbrecherbetrieb ⇝ Bevorzugt nichtperiodische Aufgaben
- Hintergrundbetrieb ⇝ Stellt nichtperiod. Aufgaben hinten an
- Slack Stealing
  - Idee: Termin ist maßgeblich
    ⇝ *Verschieben* periodischer Aufgaben möglich
  - Problem: Schlupfzeit bestimmen
  - Taktsteuerung (mit Rahmen): Einfach ⇝ $f - x_k$
  - Vorrangsteuerung: Schwierig ⇝ dynamischen Berechnung
- Zusteller ⇝ Konvertieren nicht-period. Aufgaben in Periodische
  - Spezielle periodische Aufgabe $T_s = (p_s, e_s)$
  - Ausführungsbudget, Auffüllperiode und -regeln
  - Abbildung auf Prioritätswarteschlange (z. B. AJQ)

Periodische Zusteller

- Verschiedene Ausführungen
  z. B.: Polling, Deferrable, Sporadic Server
- Unterscheiden sich (lediglich) im Regelwerk
- i. d. R. für mehrere Aufgaben zuständig

Beispiel: Abfragender Zusteller (Polling Server)

- Periodische Aufgabe $T_P = (p_s, e_s)$
- Budget e_s verfällt
- Im Falle sporadischer Aufgaben schwierig:
  - $p_P \leq \frac{D_s}{2}$, wobei $D_s \leq i_s$ (quasi Abtasttheorem)

⇝ hohe Abtastfrequenz, Überlastgefahr

Bandweite-bewahrende Zusteller

- Budget bleibt erhalten
  ⇝ Verbesserung des Abfragebetriebs
- Regelwerk wird erweitert
  ⇝ Auffüll- und Konsumregeln
- Betriebssystem (Scheduler) wacht über Budget

Auslegung

- Größe Budget
  ⇝ Berücksichtigung aller periodischer Aufgaben
- Verbesserung Antwortzeit
  ⇝ Kombination mit Hintergrundbetrieb

Beispiel: Aufschiebbarer Zusteller (Deferrable Server)

- Verbrauchsregel: Verbraucht $\frac{1}{Zeiteinheit}$ Budget bei Tätigkeit
- Auffüllregel: periodisches Auffüllen von e_s mit p_s
- keine Akkumulation

Achtung

- aufschiebbarer Zusteller $\neq$ periodische Aufgabe
- *double hit*
  ⇝ Kritischer Zeitpunkt und Auffüllzeitpunkt fallen zusammen
- ⇝ Störung ist bis zu e_s größer als bei periodischer Aufgabe

Lösungsansatz: Sporadischer Zusteller (Sporadic Server)

- Verschiedene Ausprägungen
- Beansprucht niemals mehr Zeit als periodische Aufgabe

Beispiel: SpSL Sporadic Server (Sprunt, Sha & Lehoczky)

- Verbraucht $\frac{1}{Zeiteinheit}$ Budget bei Tätigkeit
- Aufgefüllt wird entsprechend dem Verbrauchsmuster
  - Nächster Auffüllzeitpunkt wird zu Beginn der Tätigkeit bestimmt
  - Aufzufüllendes Budget zum Ende der Tätigkeit
  - ⇝ Auffüllregeln R1 – R3
- SpSL Sporadic Server
  ⇝ *Menge von Aufgaben* T_i mit $p_i = p_s$ und $\sum e_i = e_s$

Forts.: SpSL Sporadic Server, Auffüllregeln

- R1: Initiales Budget ist e_s
- R2: Zeitpunkt $rt_s = t_b + p_s$, wobei:
  - T_s besitzt Budget, dann $t_b = P_s$ wird tätig
  - T_s hat kein Budget, dann $t_b = P_s$ ist/wird tätig und T_s erhält Budget
- R3: Budgetberechnung
  - Sobald P_s untätig wird oder T_s kein Budget mehr hat
  - Budget für rt_s = Verbrauch von T_s seit t_b

Achtung

- P_s bezeichnet das Tasksystem ab der Priorität s (und höher)
- Im Beispiel: Kleinere Zahl ⇝ höherer Priorität

Beispiel