Leistungs- und Präzisionssteigerung des Lastgenerierungsprozesses von UniLoG unter Verwendung echtzeitfördernder Maßnahmen durch das Betriebssystem

Alexander Beifuß
7beifuss@informatik.uni-hamburg.de

Universität Hamburg
MIN Fakultät
Fachbereich Informatik
Telekommunikation und Rechnernetze (TKRN)

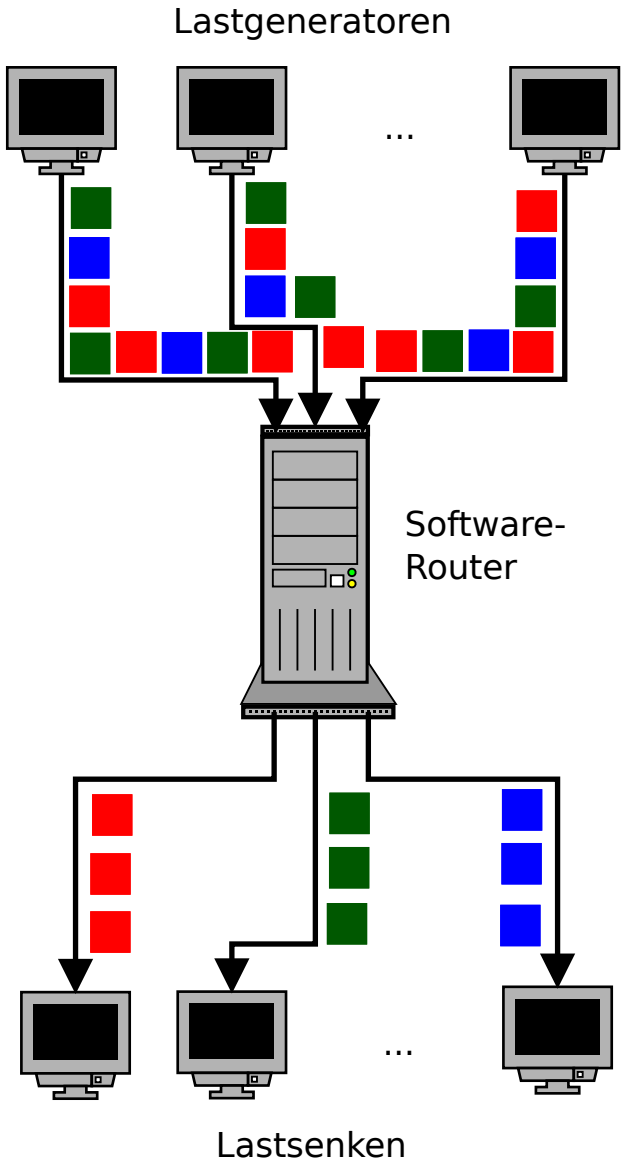
Donnerstag, 21. November 2013
Boppard am Rhein
Echtzeit 2013

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

tkrn
Telekommunikation und Rechnernetze

Inhalt

Motivation



Gründe für Messungen am System

- Verstehen des Systems
  - Modellierung
  - Flaschenhälse
- Leistungsbewertung
- Testen neuer Techniken

Gründe für Lastgeneratoren

- Spezifizierte Lastsituation
  - Skalierbarkeit
  - Reproduzierbarkeit

Motivation

Praxisrelevante Charakteristika von Lastgeneratoren:

- Flexible Lastspezifikationstechnik für ausdrucksstarke Lastmodelle
- Leistung - maximal erreichbare Paketrate
- Präzision - mittlere Verfehlzeit

Lastgeneratoren müssen mit dem Stand der Technik mithalten

Beispiel: Gigabit Ethernet

- ca. 1,49 Millionen Ethernet-Frames/s bei einer Framelänge von 64 Byte (+ 8 Byte Präambel + 12 Byte IFG)
- Verarbeitungszeit: ca. 670 ns/Frame

Motivation

Bisherige Ansätze zur Leistungs- und Präzisionssteigerung:

1. Verbesserung der Implementation
2. Verteilte Ausführung (z.B. D-ITG)
3. Einsatz dedizierter Hardware (z.B. BRUNO)
4. Nutzung der Mittel des Betriebssystems (z.B. BRUTE, pktgen)

Ansätze dieser Arbeit:

- Echtzeitfördernde Maßnahmen bei einem Desktop-Betriebssystem
- Einsatz eines Echtzeitbetriebssystems (RTOS)

Last

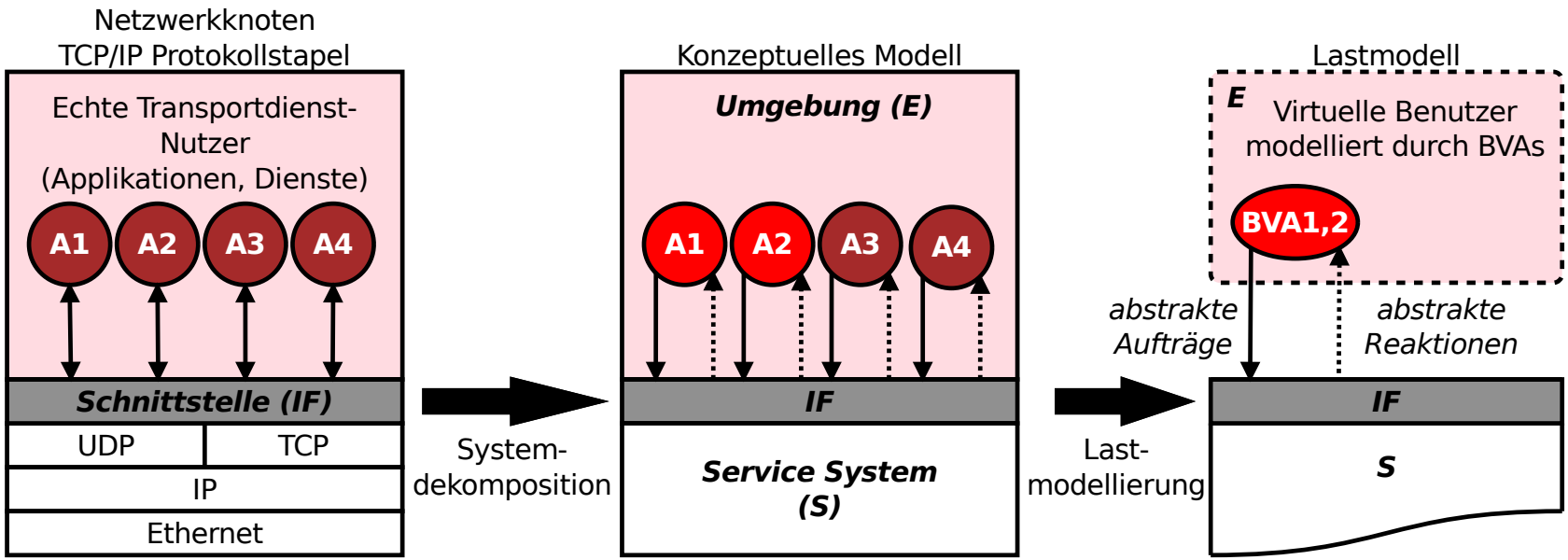


Abbildung: Dekomposition des Systems und Virtualisierung der echten Dienstbenutzer A_i durch Benutzerverhaltensautomaten (BVA).

Definition (Last)

Last ist eine Sequenz von Aufträgen, welche in einem Zeitintervall T, von einer Umgebung E, über eine wohldefinierte Schnittstelle IF, an ein Bediensystem S übergeben wird.

Die Lastspezifikation bei UniLoG

Benutzerverhaltensautomat (BVA)[1]

- Dient der Beschreibung von Sequenzen abstrakter Aufträge
  - I-Zustand: Initialisierung
  - D-Zustand: Verzögerung
  - R-Zustand: Generierung eines abstrakten Auftrags
  - T-Zustand: Terminierung
  - Übergangsfunktion: Probabilistisch

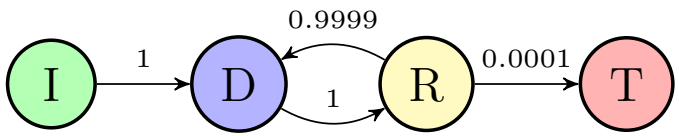
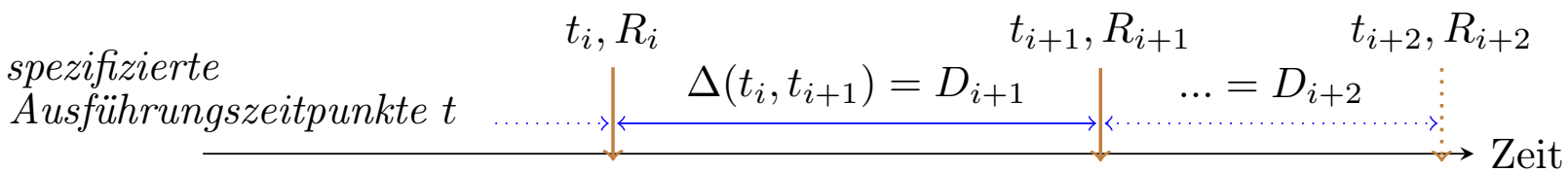
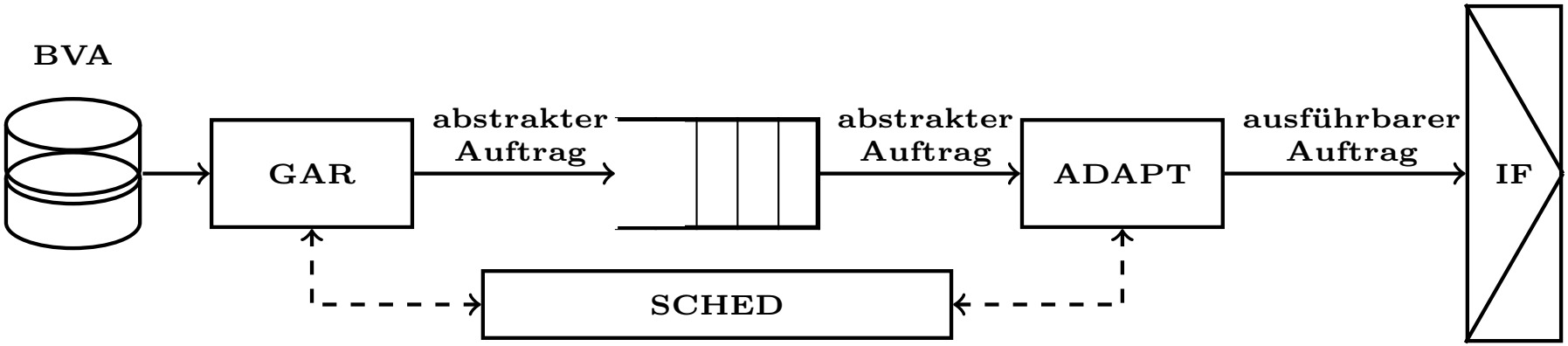


Abbildung: Beispiel eines sehr einfachen Benutzerverhaltensautomaten



[1] A. Kolesnikov, Konzeption und Entwicklung eines echtzeitfähigen Lastgenerators für Multimedia-Verkehrsströme in IP-basierten Rechnernetzen, Echtzeit '08

UniLoGs Architektur



- GAR: Generator abstrakter Aufträge
  - Auswertung des BVA
- ADAPT: Schnittstellen-spezifischer Adapter
  - Konvertierung der abstrakten Aufträge
  - Ausführung der Aufträge
- SCHED: Interner kooperativ arbeitender Scheduler
  - Kontrollzuweisung

Welche Rolle spielt die Ausführungsumgebung?

Desktop-Betriebssystem:

- i.d.R. präemptiver Scheduler (Prioritätsklassen, fair)
  - Unterbrechungen und Verdrängungen
- Nebenläufige Prozesse
  - Wahrscheinlichkeit der Verdrängung steigt.

Ein RTOS stellt die bessere Ausführungsumgebung dar:

- Kontrolle über den Scheduler (Bsp. kooperatives Scheduling)
  - Keine Unterbrechung
  - Keine Verdrängung

Problematik beim Parallelisieren von GAR und ADAPT:

- Single- & Dual-Core Systeme
  - Kontextwechsel
  - Verdrängung

Durchgeführte Leistungsmessungen

- Maximal erreichbare Paketrate P_{max}
  - Infinitesimale Zwischenankunftszeiten: $ZAZ^{infi}_{spec} = \Delta(t_i, t_{i+1}) \stackrel{!}{=} 0, \forall i$
  - GAR und ADAPT kommen alternierend zur Ausführung

$$ZAZ^{min}_{real} \sim T_{gar} + T_{adapt}$$

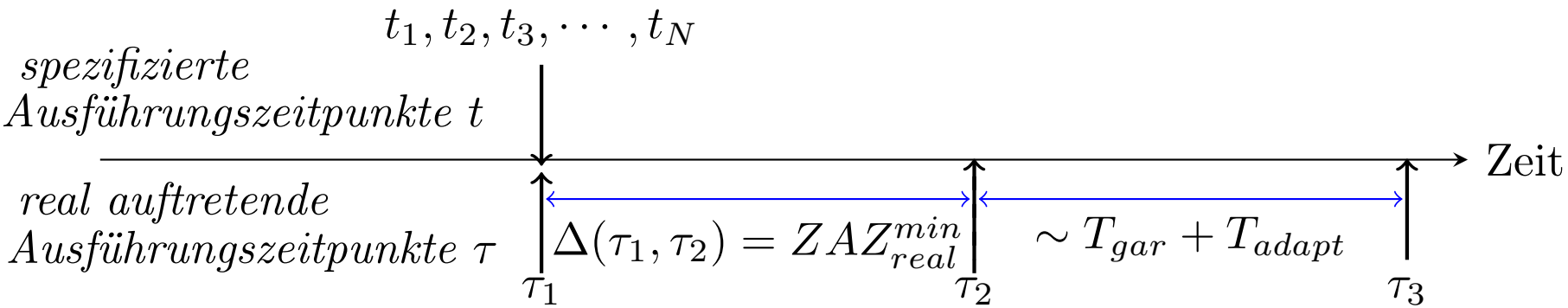


Abbildung: Skizze zur Erläuterung von $ZAZ_{real} = \Delta(\tau_i, \tau_{i+1}) = \tau_{i+1} - \tau_i$, T_{gar} bzw. T_{adapt} sind die Ausführungszeiten der jeweiligen Komponenten

$$P_{max} = \frac{\text{N (Anzahl der ausgeführten Aufträge)}}{\text{T (Länge des betrachteten Zeitintervalls)}}$$

Durchgeführte Präzisionsmessungen

- Mittlere Verfehlzeit ($\overline{VZ}$) in Abhängigkeit von ZAZ_{spec}

$$\overline{VZ} = \frac{1}{N}\sum_{i=1}^{N} VZ_i, \text{ mit } VZ_i = \Delta(t_i, \tau_i) = \tau_i - t_i$$

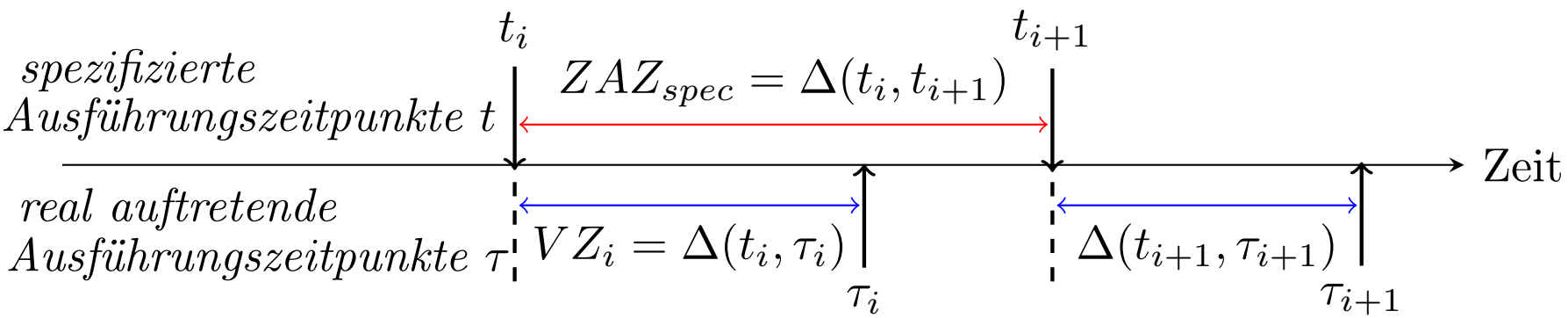


Abbildung: Skizze zur Erläuterung von VZ

Rahmenbedingungen der Messungen

- Schnittstelle: UDP-Sockets (blockierend)
- Hardware: Dell Optiplex 980 Intel Core i5-750 @ 4x2,66GHz, 4 GB RAM, Intel(R) 82578DM Gigabit-Ethernet-Adapter
- Betriebssysteme:
  1. Windows 7 (c_1) - Standardinstallation
  2. Windows 7 (c_2) - Standardinstallation + Maßnahmen
     - Deaktivierung nicht benötigter Dienste
     - Nutzung der Prozess-Priorisierung
     - Nutzung der Prozess-zu-Kernen-Affinität
  3. On TIME RTOS-32
     - Untermenge des Windows Kernels
     - Scheduler: ereignisbasiert, unterbrechend oder kooperativ
     - ca. 400 Win32 API-Funktionen (WinSock 1.1 API)
- 10 Wiederholungen zu 20 s je Parametrisierung

Leistung - Windows 7 (c_1)

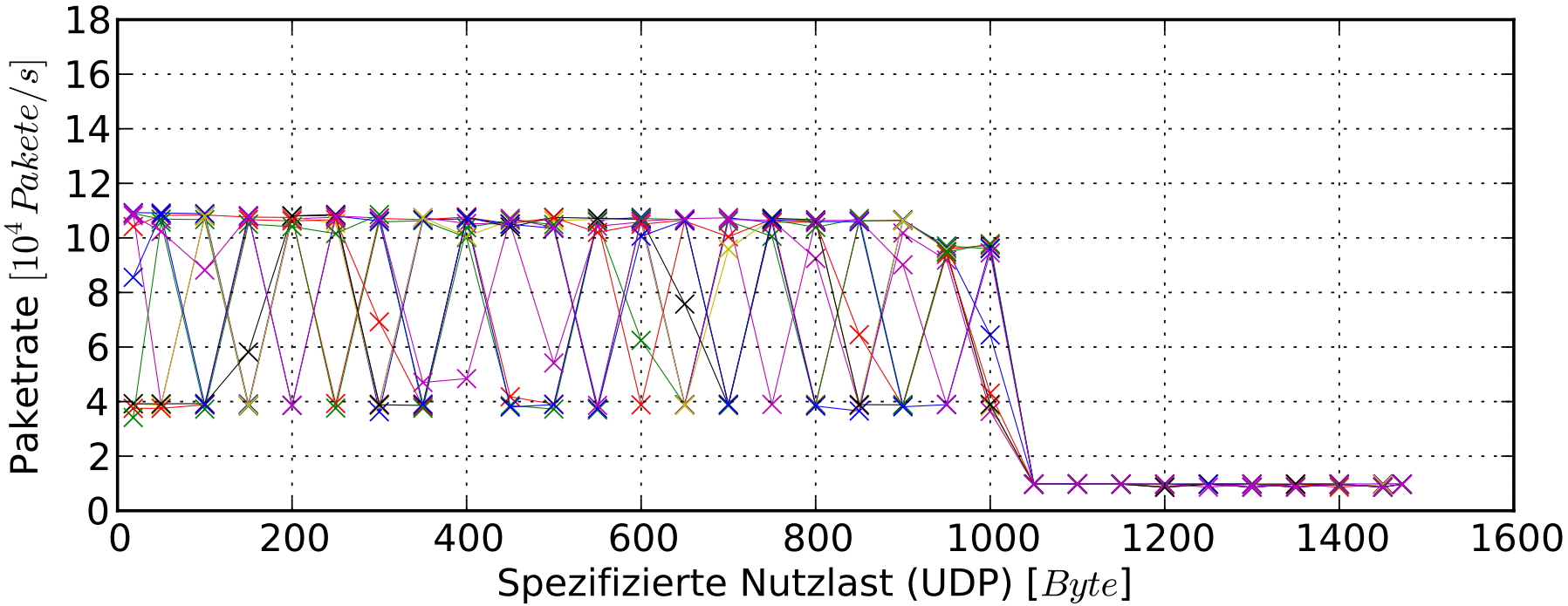
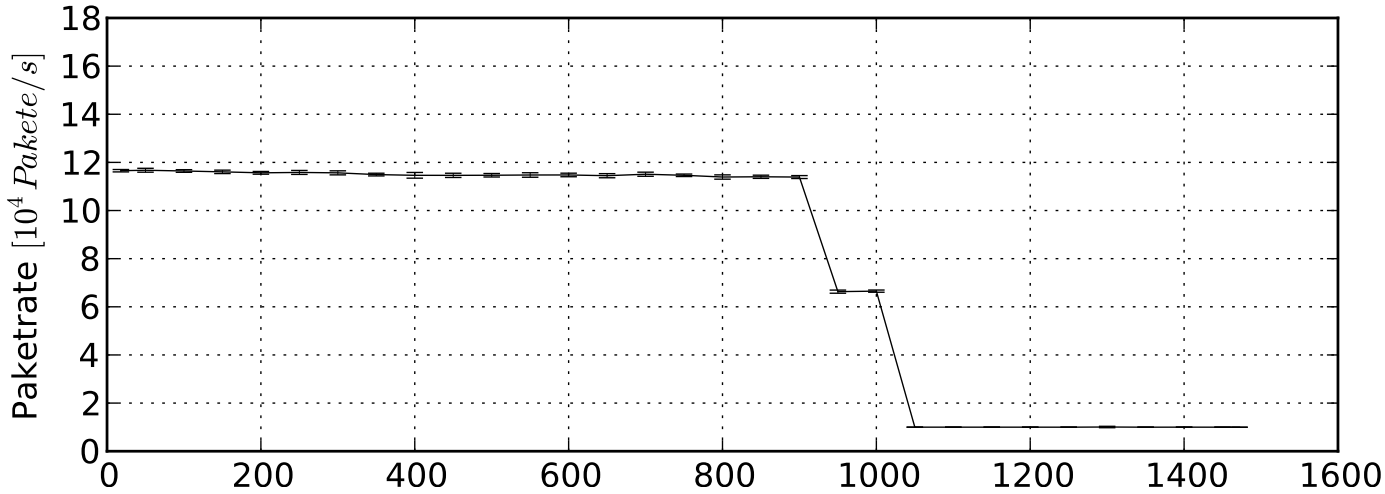


Abbildung: Windows 7 (c_1), Rohdaten der Leistungsmessung

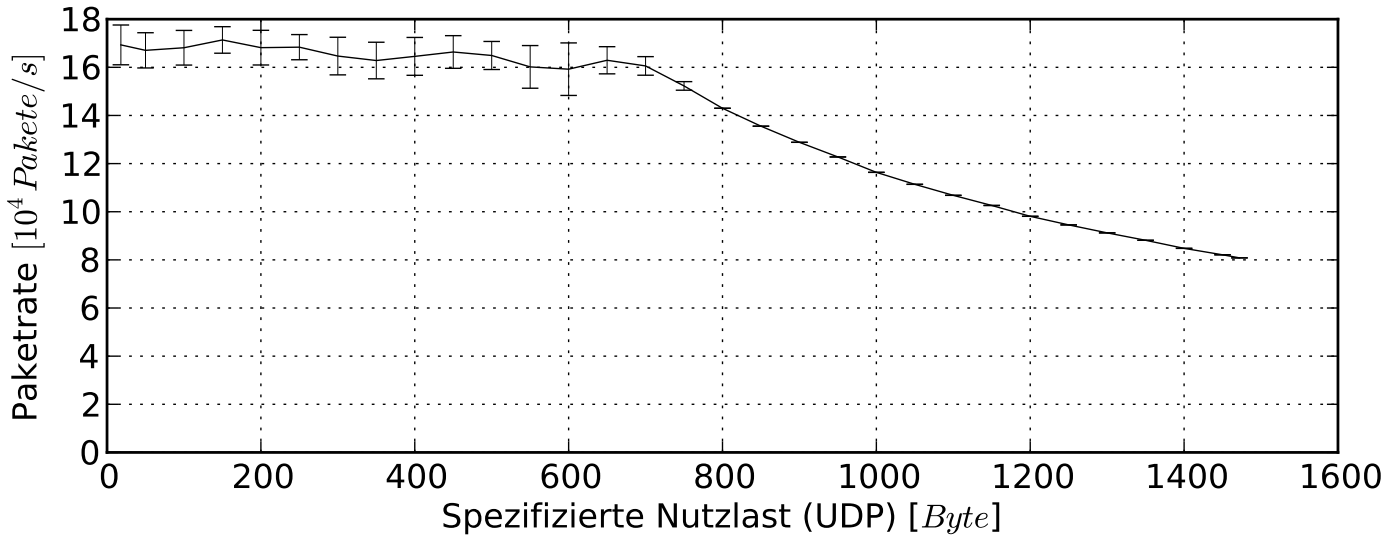
- Ungeeignet für zuverlässige hochfrequente Paketgenerierung
- Ergo: Untersuchung der Präzision entfällt

Leistung - Windows 7 (c_2) vs. RTOS-32

Win 7 c_2

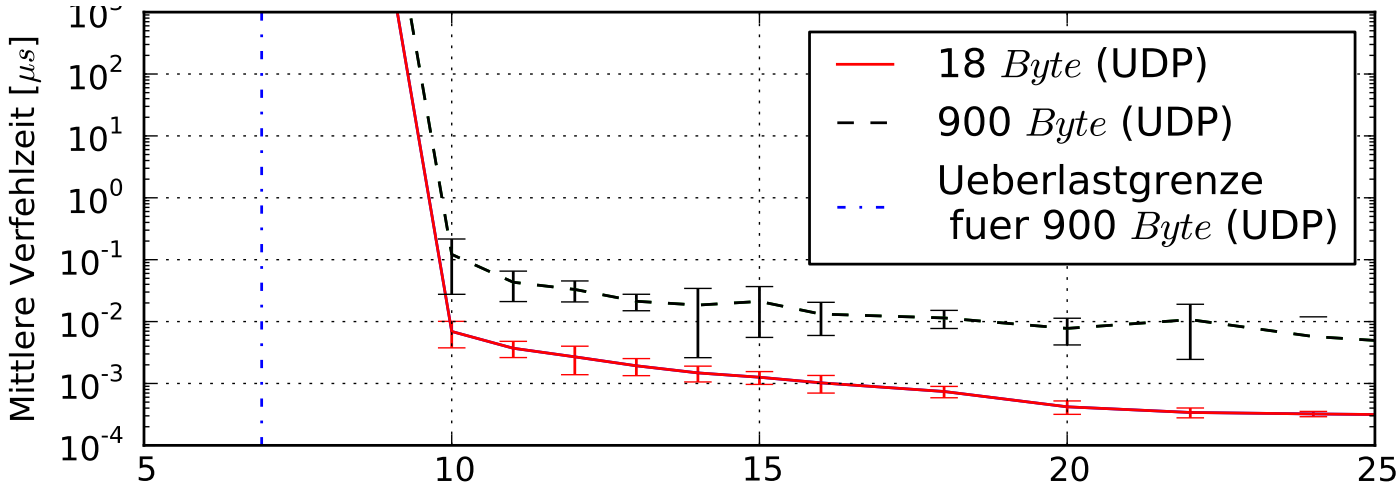


RTOS-32

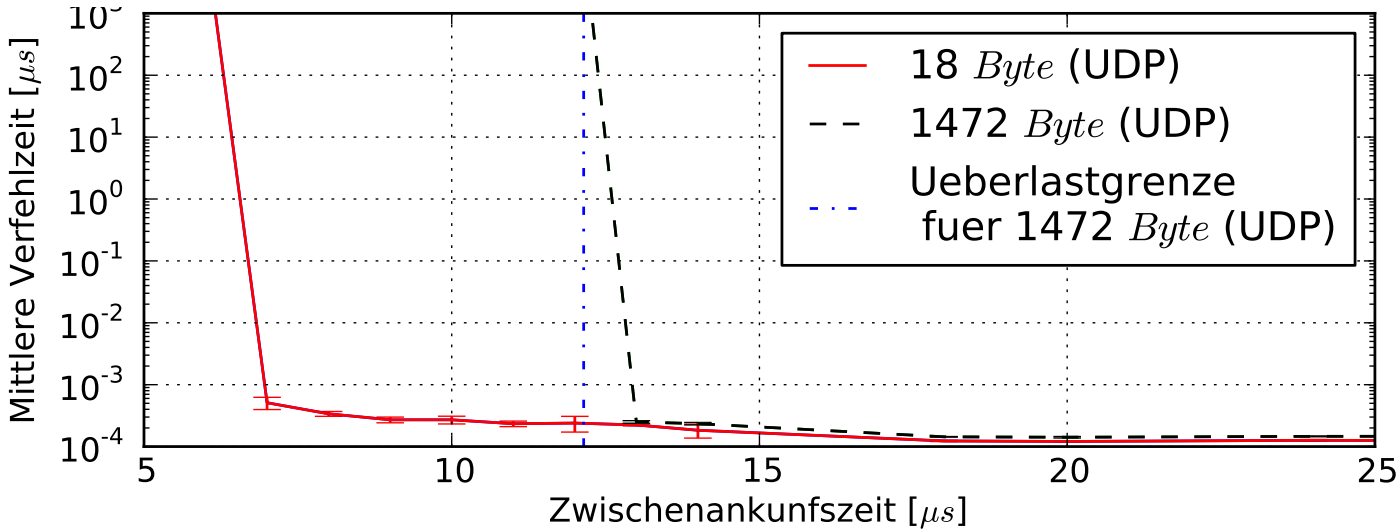


Präzision - Windows 7 (c_2) vs. RTOS-32

Win 7 c_2

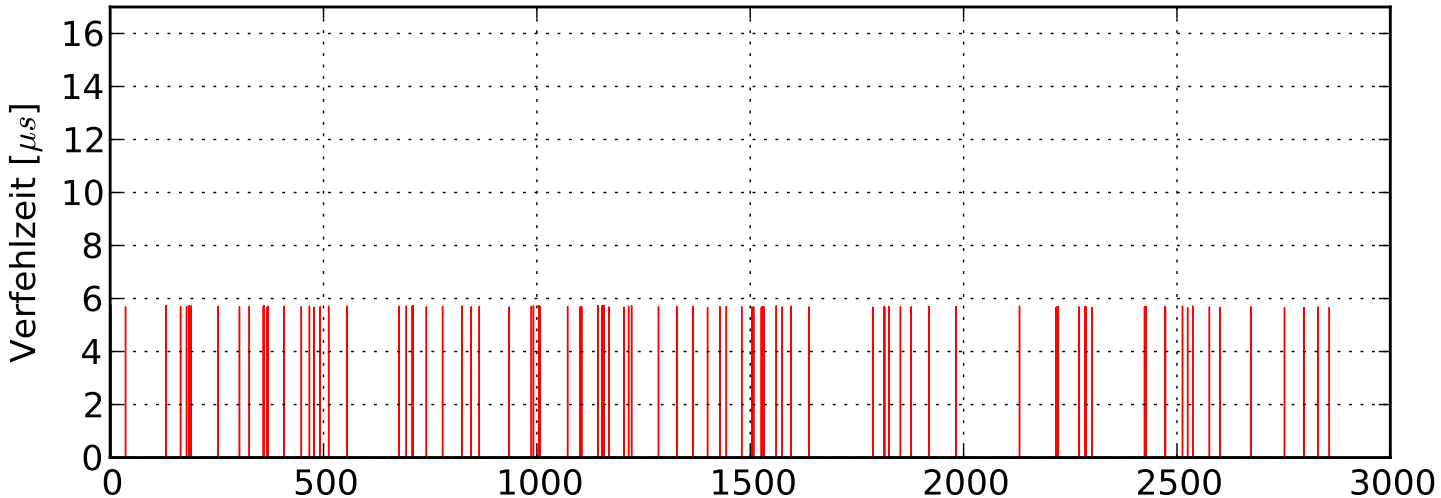


RTOS-32

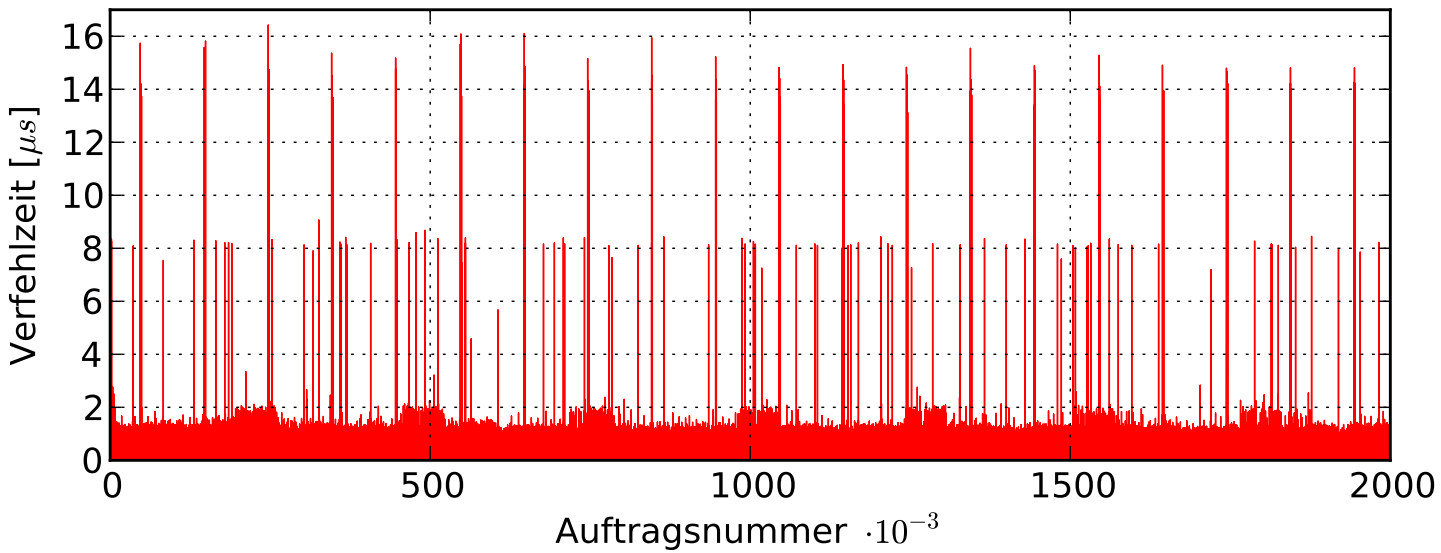


Ergebnisse - Kumulierte Verfehlzeit

RTOS-32
18 *Byte*
7 μs



Win 7 c_2
18 *Byte*
10 μs



Zusammenfassung & Ausblick

Zusammenfassung:

- Echtzeitfördernde Maßnahmen unter Windows 7 notwendig
- Problem mit geringem Sendepuffer und blockierenden Sockets
- Leistung um ca. 30% höher mit RTOS-32
- Präzision um bis zu zwei Größenordnungen geringer mit RTOS-32
- Fehlentscheidungen des internen Schedulers

Ausblick auf Verbesserungen von UniLoG:

- Parallelisierung bei Verdrängungs- und Unterbrechungsfreiheit:
  - Interner Scheduler entfällt
  - Lock-free Queues
  - Präzision ist gewährleistet gdw. $ZAZ_{spec} > max(T_{gar}, T_{adapt})$
- Was bietet Linux?
  - CONFIG_PREEMPT_NONE
  - SCHED_FIFO + PTHREAD_SETAFFINITY_NP
  - CONFIG_NO_HZ_FULL[2]

[2] https://www.kernel.org/doc/Documentation/timers/NO_HZ.txt

Vielen Dank für Ihre Aufmerksamkeit

Mein besonderer Dank gilt:
Herrn Prof. Dr. rer. nat. Bernd E. Wolfinger
Herrn Dipl.-Inform. Andrey Kolesnikov
Der On Time Informatik GmbH