

# Verteilte Systeme

## 13. Lastverteilung

### „Geteilte Last ist halbe Last“

- Potential an Parallelarbeit im Rechnernetz optimal ausnutzen
- „Kreative“ Lastverteilung
  - Verteilte Programmierung
- „Mechanische“ Lastverteilung
  - Auf welchem Rechner soll eine neue „Last“ ausgeführt werden?
  - Ziel: Gleiche Auslastung aller beteiligten Rechner
  - Problem: Fehlender globaler Zustand
- Lastbegriffe
  - Systemspezifisch
    - Prozesse
  - Anwendungsspezifisch
    - Verteilbare Berechnungsabschnitte

### Grundüberlegungen

- ⌘ Viele Lastpakete notwendig, d.h. genügend Parallelarbeit vorhanden
  - Alle Rechner nutzbar
- ⌘ Kleine Lastpakete notwendig, d.h. kurze Aufträge
  - Geringe Lastabweichungen
- ⌘ Nicht zu kleine Lastpakete, da auch das Verteilen kostet
  - Heisenberg läßt grüßen
  - Paketgröße häufig a priori nicht bekannt
- ⌘ Gewisse Trägheit bei der Verteilung sinnvoll
  - Stabiles Systemverhalten

Systemsoftware II, Winter 2002/03 Folie 13.3

### Architektur

- ⌘ Lastmetrik
  - Welche Last hat ein Rechner aktuell
  - Welche Last erzeugt ein Paket auf einem Rechner
- ⌘ Verteilung des Lastwerts der beteiligten Rechner
  - Entscheidungsgrundlage für den Verteiler
- ⌘ Verteilung des Lastpakets auf den gewählten Rechner
  - Welche Informationen müssen übertragen werden

Systemsoftware II, Winter 2002/03 Folie 13.4

## Lastmetriken

- Häufig keine einzelne Zahl, wie z.B. Last 42
- Mehrere Bestandteile
  - Prozessorauslastung
    - Anzahl der Prozesse
    - Anzahl rechnender und rechenwilliger Prozesse
    - CPU-Auslastung
    - Bedienzeit
    - ...
  - Speicherauslastung
    - Hauptspeicherbedarf
    - Hintergrundspeicherbedarf
  - Kommunikationslast
    - E/A-Verkehr in Byte/s pro Netzanschluß
- Vergleichbarkeit zweier Lastvektoren?

Systemsoftware II, Winter 2002/03

Folie 13.5

## Einzelprobleme

- CPU-Auslastung
  - Differenzierung jenseits von 100% nicht möglich
  - Vergleichbarkeit bei verschiedenen Prozessortypen kritisch
- Prozeßanzahl
  - Besondere Unverträglichkeiten mit anderen Prozessen auf demselben Rechner
- Speicherauslastung
  - Systemsoftware wird versuchen, den gesamten physischen Speicher zu nutzen, d.h. Wert immer nah bei 100%
- ...

Systemsoftware II, Winter 2002/03

Folie 13.6

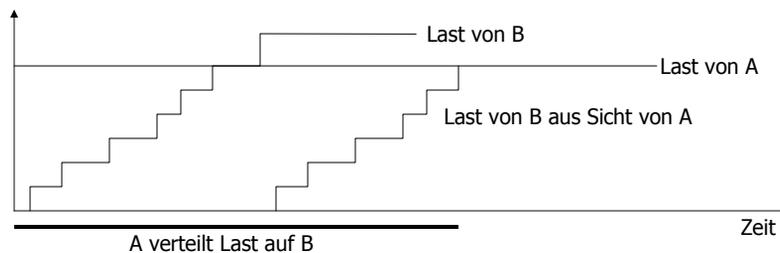
## Verteilung oder Ermittlung der Lastwerte

- Push- und Pull-Ansätze
- Pull
  - Last verdrängen, Lastwerte anziehen
    - Lastwerte bei der Erzeugung eines Pakets ermitteln
    - Varianten
      - Von allen Rechnern (Überflutungsalgorithmus, Broadcast)
      - Von einer festen Teilmenge
      - Zufällige Teilmenge
- Push
  - Last verdrängen, Lastwerte verteilen
    - Unabhängig vom Erzeugungszeitpunkt
    - Varianten
      - An alle Rechner verteilen
      - An eine feste Teilmenge verteilen
      - An eine zufällige Teilmenge verteilen

Systemsoftware II, Winter 2002/03

Folie 13.7

## Alter der Lastwerte



- Endliche Ausdehnungsgeschwindigkeit der Information
- Lösungsansätze
  - Lokale Information selbständig aktualisieren
  - Keine krassen Sprünge
  - Lasten fallen nicht plötzlich auf 0
  - Exponentielle Verzögerung



Systemsoftware II, Winter 2002/03

Folie 13.8

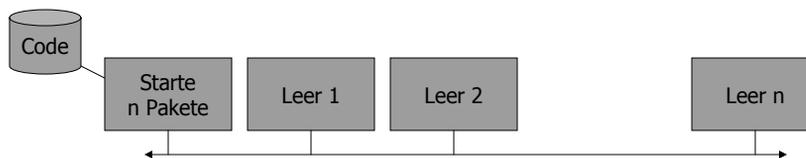
## Übermittlung des Pakets auf den gewählten Rechner

- ☐ Ist der auszuführende Code lokal vorhanden?
  - Ja
    - ☐ Identische Pfadnamen
    - ☐ Achtung Netzwerkdateisystem (siehe nächste Folie)
  - Nein
    - ☐ Zusätzlicher Kommunikationsaufwand
    - ☐ Function-Shipping (Schutzproblematik)
- ☐ Umfang der zu übermittelnden Daten
- ☐ Umfang der Ergebnisdaten

Systemsoftware II, Winter 2002/03

Folie 13.9

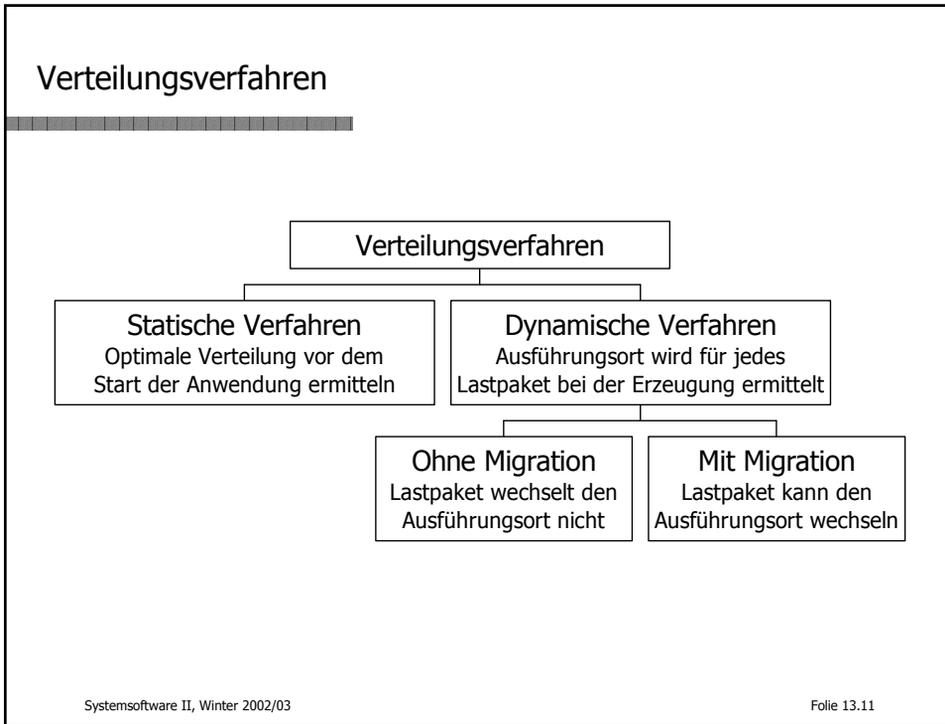
## Verteiler: "Super! der Code ist schon dort !?"



- ☐ n Rechner, Code per NFS o.ä. für jeden Rechner zugreifbar
- ☐ Rechner über Ethernet verbunden
  - Übertragungszeit Code über NFS: 10 ms
  - Übertragungszeit Auftrag und Argumente: 1 ms
- ☐ n autonom ausführbare Lastpakete
- ☐ Zeitpunkt der Fertigstellung
  - Code über NFS?
  - Code lokal vorhanden (z.B. im Cache)?

Systemsoftware II, Winter 2002/03

Folie 13.10

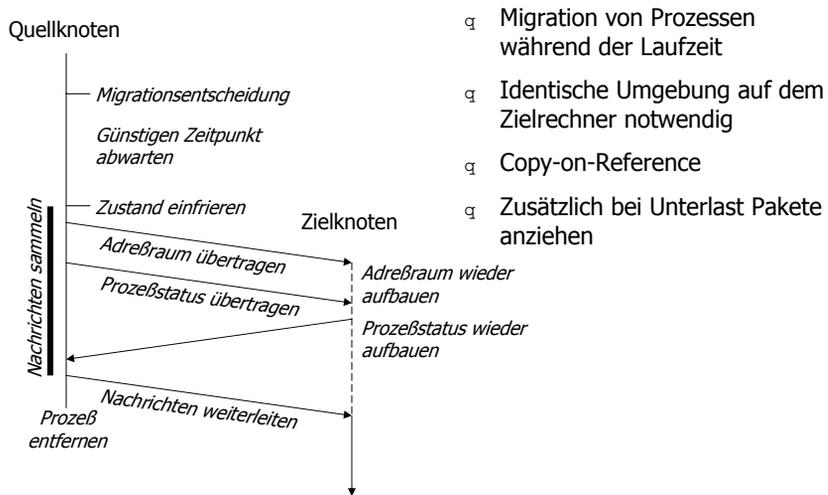


### Statische Lastverteilung

Systemsoftware II, Winter 2002/03 Folie 13.12

- Statisches Anwendungssystem
  - Alle Bestandteile der Lastmetrik sind für alle Prozesse bekannt
- Ausführungsplattform
  - k Prozessoren
- Fragestellung
  - Partitioniere in k Subgraphen, so daß Prozessor- und Speicherlast erfüllbar und Kommunikationslast zwischen den Subgraphen minimal
  - Kommunikationslast innerhalb eines Subgraphen wird vernachlässigt
  - Jeder Subgraph wird auf einem Rechner plaziert
- Probleme
  - Exklusive Prozessorbenutzung
  - Keine Dynamik
- Optimale Lösungen berechenbar

## Dynamische Lastverteilung mit Migration



Systemsoftware II, Winter 2002/03

Folie 13.13

## Dynamische Lastverteilung ohne Migration

- Kritik an den anderen Verfahren
  - Statische Verfahren selten anwendbar
  - Migrationsbasierte Verfahren zu teuer
  - Häufig zu hoher Aufwand bei der Verteilung/Ermittlung des Lastwerts
- Initial Placement
  - Nur bei der Erzeugung eines Pakets wird der Zielknoten ermittelt
  - Im Extremfall zufällige Wahl eines Knotens
- In Untersuchungen liefert dieses Verfahren sehr gute Ergebnisse
  - Hohe Anzahl an Lastpaketen wesentlich
  - In diesem Fall liefern komplexere Verfahren nur unwesentlich bessere Ergebnisse
  - siehe [Eager et al. 1986]

Systemsoftware II, Winter 2002/03

Folie 13.14

## Anwendungsbeispiel

---

- Nutzung ungenutzter Workstations
  - Viele Universitätsrechner meiste Zeit unbenutzt
    - Untersuchung in USA: Selbst zu Hochzeiten 30% der Rechner ungenutzt
  - Viele rechenintensive und verteilbare Probleme
- Aufbau eines Rechnerverbunds
  - Andere können meinen Rechner verwenden, wenn ich nicht da bin!
  - Ich kann die Rechner Anderer verwenden, wenn die nicht da sind!
- Probleme
  - Ermittlung arbeitsloser Rechner
  - Transparente Ausführungsumgebung
  - Was tun, wenn Besitzer wieder kommt?
    - Abbrechen oder Sichern?
- Beispiel: Condor-System [Litzkow et al. 1988]



Systemsoftware II, Winter 2002/03

Folie 13.15

## Literatur

---

- D.L. Eager, E.D. Lazowska, J. Zahorjan  
*Adaptive Load Sharing in Homogeneous Distributed Systems*  
IEEE Transactions on Software Engineering  
Vol. SE-12, pp. 662-675, 1986
- M.J. Litzkow, K. Livny, M.W. Mutka  
*Condor - A Hunter for Idle Workstations*  
Proc. 8th Intl. Conf. on Distributed Computing Systems  
IEEE, pp. 104-111, 1988

Systemsoftware II, Winter 2002/03

Folie 13.16