Fehlertoleranz

- Einführung
- Begriffsbildung
- Techniken der Fehlertoleranz
  - Statisch
  - Dynamisch
- Quantitative Bewertung
- Systeme in der Praxis

Fehlertoleranz
Die zehn wichtigsten Fragen

- Wozu Fehlertoleranz?
- Warum sind Parallelrechner in diesem Bereich wichtige Architekturtypen
- Welche Kenngrößen werden verwendet?
- Welche Fehlerkategorien gibt es?
- Was ist statische Redundanz?
- Welches sind hierbei die typischen Konzepte?
- Was ist dynamische Redundanz?
- Welche Durchführungsphasen finden wir hierbei?
- Wie sieht die qualitative Bewertung eines Systems ohne Reparatur aus?
- Welche Umsetzung gibt es für Linux-Systeme?

Warum als Thema Fehlertoleranz?

Vorlesungsstunde hätte auch „Ausfallsicherheit" oder „Hochverfügbarkeit" heißen können

„Fehlertoleranz" ist der allgemeine Mechanismus, mit dem Ausfallsicherheit und Hochverfügbarkeit erzielt wird

Motivation

Wozu Fehlertoleranzmechanismen?

- Ausfälle im System werden verdeckt und das System läuft mit kurzer Unterbrechung oder mit verminderter Leistung weiter
- Schutz vor Verlust von Menschenleben und/oder Sachwerten

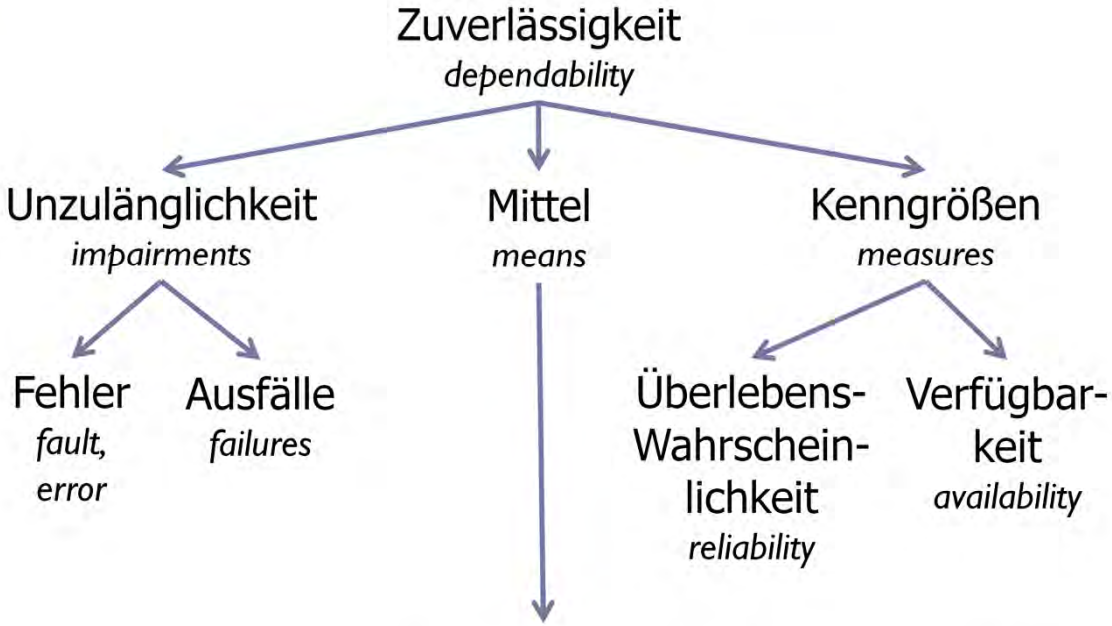
Warum auf Parallelrechnern / in Clustern?

- Redundante Hardware-Strukturen eignen sich besonders gut zur Fehlererkennung und dynamischen Fehlerverdeckung

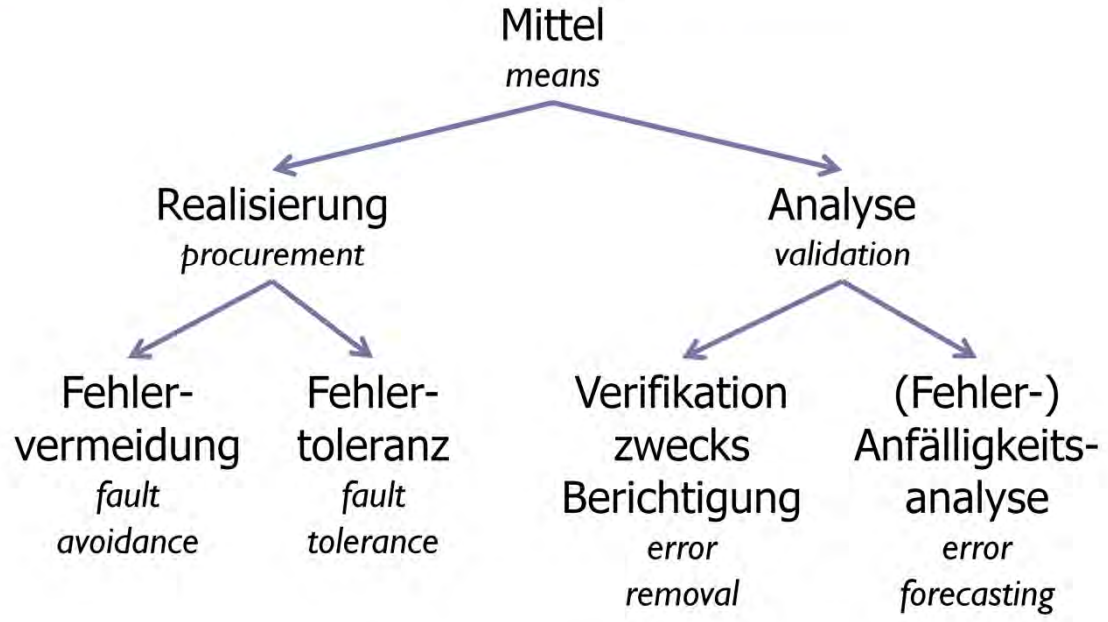
Warum wirklich auf Parallelrechnern / in Clustern?

- Redundante Hardware-Strukturen fallen ständig aus ☹

Begriffsbildung



Begriffsbildung...



Zuverlässigkeit

Definition

Fähigkeit eines Systems, die spezifizierte Anwendungsfunktion während eines Einsatzzeitraumes zu erbringen

Kenngrößen

- Überlebenswahrscheinlichkeit
- Verfügbarkeit

Zuverlässigkeit...

Überlebenswahrscheinlichkeit

- Wahrscheinlichkeit R(t) dafür, dass das System im Zeitintervall [0;t] fehlerfrei bleibt, wenn es zu t=0 intakt war

  [System ohne Reparatur]

Verfügbarkeit

- Wahrscheinlichkeit A(t) dafür, dass das System zum Zeitpunkt t intakt ist

  [System mit Reparatur]
- A(t)=MTBF / (MTBF+MTTR)

  MTBF: mean time between failures

  MTTR: mean time to repair

Wichtige Kennzahlen der Verfügbarkeit

Man spricht z.B. von den fünf Neunen, gemeint ist 0,99999 % Verfügbarkeit

Was bedeutet das als Ausfallzeit im Jahr?

Verfügbarkeit	Ausfallzeit pro Jahr
0,9	876 Stunden
0,99	87 Stunden
0,999	9 Stunden
0,9999	52 Minuten
0,99999	5 Minuten

Unzulänglichkeiten

Beeinträchtigung der Zuverlässigkeit

- Fehler

  Unerwünschter (nicht die Spezifikation erfüllender) Zustand des Systems

  (unterscheide noch Fehlerursache/-zustand)

- Ausfall

  Ist das Ereignis, dass eine benötigte Funktion nicht erbracht wird

Fehler

Einteilung nach ihrem Entstehen im Lebenszyklus des Systems

- Entwurfsfehler
- Herstellungsfehler
- Betriebsfehler

Einteilung nach der Zeitdauer

- Permanent
- Transient
- Intermittierend (pseudotransient)

Fehler...

Entwurfsfehler

Vor Inbetriebnahme des Systems

- Spezifikationsfehler
- Implementierungsfehler
- Dokumentierungsfehler

Herstellungsfehler

System entspricht nicht der entworfenen Implementierung

- Z.B. fehlerhafte Materialen oder Werkzeuge
- Bei uns: Fehler in Compilern und Bibliotheken

Fehler...

Betriebsfehler

In der Nutzungsphase nach Inbetriebnahme

- Zufällige physikalische Fehler
- Verschleißfehler
- Störungsbedingte Fehler
- Bedienungsfehler
- Wartungsfehler
- Absichtliche Fehler (z.B. Sabotage)

Softwarefehler meist Entwurfs- oder Herstellungsfehler

Fehlermodell

Aufgrund der Vielzahl möglicher Fehler:

- Einschränkung auf bestimmte Gruppen
- Fehlermodell nennt betroffene Subsysteme und beschreibt Fehlverhalten
- Damit dann analytische Studien möglich

Z.B. Fehlermodell für Rechner-Cluster

- Nur permanente Fehler in Knoten, Leitung, Switches. Zusätzlich evtl. Prozessoren, Speichermodule, Festplatten
- Fehlerzustände: intakt, defekt
- Systemzustand dargestellt durch Vektor

Mittel zur Realisierung von Zuverlässigkeit

Ergänzen sich gegenseitig:

- Fehlervermeidung
- Fehlertoleranz

Fehlervermeidung (Fehlerintoleranz)

- Sorgfältige Konstruktion
- Ausführliche Tests

Nicht vermeidbare Fehler versuchen zu tolerieren

Fehlertoleranz

Einsatz von Redundanz, um im Fehlerfall weiterarbeiten zu können

- Redundanz
  - HW-Redundanz (zusätzliche Bauteile)
  - SW-Redundanz (zusätzliche Programme)
  - Zeit-Redundanz (zusätzlicher Zeitaufwand)
- Aktivierung der Redundanz
  - Statische Redundanz (funktionsbeteiligte Redundanz)
  - Dynamische Redundanz (Reserveredundanz)

Fehlertoleranz...

Statische Redundanz

- Ressourcen ständig in Betrieb
- Im Fehlerfall direkter Zugriff auf diese Einheiten
- Z.B. Hamming Codes / Triple Modular Redundancy

Dynamische Redundanz

- Aktivierung erst im Fehlerfall
- Bis zu diesem Zeitpunkt:
  - Ungenutzte Redundanz (Standby-Systeme)
  - Fremdgenutzte Redundanz (mit Verdrängung)
  - Gegenseitig nutzbare Redundanz (Fail-Soft-Systeme)

Fehlertoleranztechniken

Tolerierung verschiedener Fehlerarten

- HW-Fehler (siehe folgende Folien)
- SW-Fehler

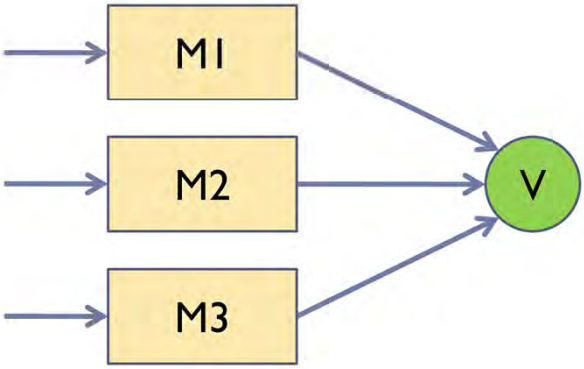
  N-Versionen-Programmierung (gut für Mehrprozessorsysteme)

Betrachtungsebene

- Hier: Prozessoren, Speicher, Verbindungen
- Tiefere Ebenen auch möglich: z.B. fehlerkorrigierende Codes

Statische Redundanz

Wichtigste Technik: Triple Modular Redundancy (TMR)



3 identische Module

1 Voter

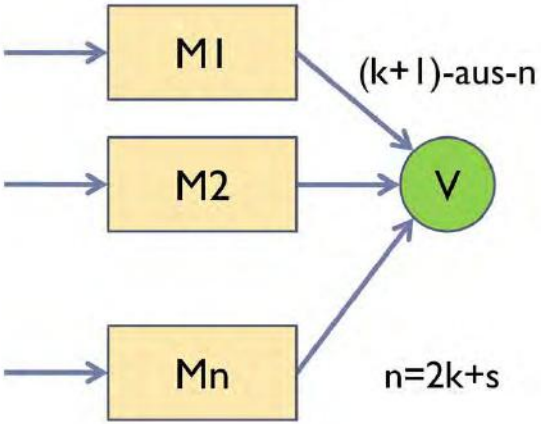
Statische Redundanz...

Konzept TMR

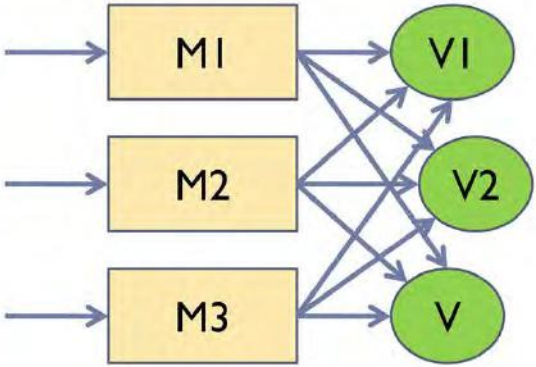
- Drei identische Module führen nebenläufig gleiche Funktion aus (kann HW oder SW sein)
- Voter fällt 2-aus-3-Mehrheitsentscheid
- Erster Ausfall kann toleriert werden
- Zweiter Ausfall kann noch erkannt werden
- Voter kann per Hardware oder Software realisiert werden

Statische Redundanz...

N-Modular-Redundancy (NMR)



Tolerierung von Voter-Ausfällen



Statische Redundanz...

Vorteile TMR/NMR

- Toleriert permanente und transiente Fehler
- Fehlerbehebung kostet keine Zeit

Nachteile

- Hoher Aufwand in HW oder SW begrenzt Einsatzbereich

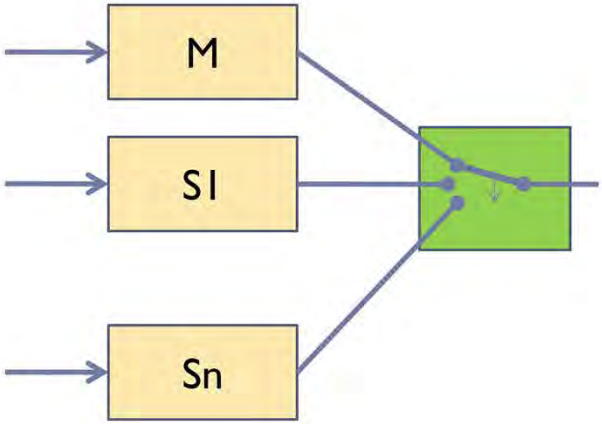
Dynamische Redundanz

Standby-Systeme

Zusätzlich zu den Systemkomponenten noch Reservekomponenten

Nachteile:

- Kein verzögerungsfreies Umschalten
- Ungenutzte Ressourcen (cold/hot standby)



Dynamische Redundanz...

Fail-Soft-Systeme (Graceful Degradation)

- Mehrfach vorhandene Subsysteme werden als Redundanzen genutzt (Eigenredundanz)
- Defekte Systemkomponenten werden deaktiviert
- Mit verminderter Leistung kann die Aufgabe weiterbearbeitet werden

- Ausgangsbedingungen und Verfahren durch ein Rechner-Cluster optimal abgedeckt

Dynamische Redundanz...

Vorgehensweise bei beiden Verfahren

- Fehlererkennung und –lokalisierung (Fehlerdiagnose)
- Rekonfiguration
- Fehlerbehandlung (Recovery)

Dynamische Redundanz...

(1) Fehlerdiagnose

- Aufgabe: Erkennen defekter Knoten
- Ebene: Komponenten- und Systemebene
- SRU (Smallest Replacable Unit)

  Ebene der Rekonfiguration

  Keine Lokalisation auf Komponentenebene
- Fehlererkennung auf Rechnerknotenebene

  Fehlererkennende Codes, Selbsttestprogramme
- Zur Rekonfiguration noch Diagnose auf Systemebene

Dynamische Redundanz...

- Systemdiagnose
  - Fremddiagnose
    - Separater Prozessor für Diagnose (und i.a. auch für Rekonfiguration und Recovery)
    - Nachteile
      - Diagnose- und Wartungsprozessor als perfekt zuverlässig angenommen
      - Zusätzliche Prozessortypen (Heterogenität)
  - Eigendiagnose
    - Zentral: Testrunde in bestimmten zeitlichen Abständen
      Koordinatoreinheit ermittelt defekte Komponenten
      Problem: Wahl des Koordinatorknotens
    - Dezentral: Alle intakten Einheiten haben Diagnosebild
      Problem: Konsistenz der Einzelbilder

Dynamische Redundanz...

Allgemeines Steuerungsproblem: Zentral vs. dezentral

- Zentral
  - Vorteile
    - Konsistente Systemsicht (wichtig!)
  - Nachteile
    - Gefährlich! Sogenannter Single-Point-of-Failure
    - Evtl. Überlastung der Kommunikationswege hin zur Zentrale
- Dezentral
  - Vorteile
    - Kein Single-Point-of-Failure
    - Gut im System verteilbare Last
  - Nachteil
    - Konsistente Systemsicht beliebig schwierig
    - Nutze „Verteilte Algorithmen“ (siehe z.B. Arbeiten von Leslie Lamport)

Dynamische Redundanz...

(2) Rekonfiguration

- Bildet aus den intakten Einheiten ein lauffähiges System
- Ausführung
  - Externer Wartungsprozessor
  - Zentraler Koordinator (hier unkritisch)
  - Dezentral
- Schwierigkeit
  - Suche einer neuen effizienten Abbildung der SW-Komponenten auf die HW-Komponenten

Dynamische Redundanz...

(3) Fehlerbehebung

- Versetzt das System in einen korrekten Zustand
- Setzt die Anwendung mit korrekten Daten wieder auf (Wiederanlauf)
- Methoden
  - Rückwärtsfehlerbehebung
  - Vorwärtsfehlerbehebung

Dynamische Redundanz...

- Rückwärtsfehlerbehebung
  - System zurücksetzen in früheren konsistenten Zustand (*rollback*)
  - Rücksetzpunkte (*recovery points*)
  - Abspeicherung in Haupt- oder Hintergrundspeicher
  - Zeitintervalle durch analytische Methoden
- Vorwärtsfehlerbehebung
  - Zusätzliche Operationen bringen das System in einen korrekten Zustand
  - Problem: Genaue Kenntnis der Anwendung notwendig

Dynamische Redundanz...

Bewertung

- Bei dynamischer Fehlertoleranz kein unterbrechungsfreier Betrieb
  - Für „normale“ Anwendungen akzeptabel, nicht jedoch bei Echtzeitanwendungen
- Je nach Variante können aber alle Komponenten produktiv genutzt werden

Quantitative Bewertung

Überlebenswahrscheinlichkeit R(t)

Exponentiell verteilte Lebensdauer $R(t)=e^{-\lambda t}$

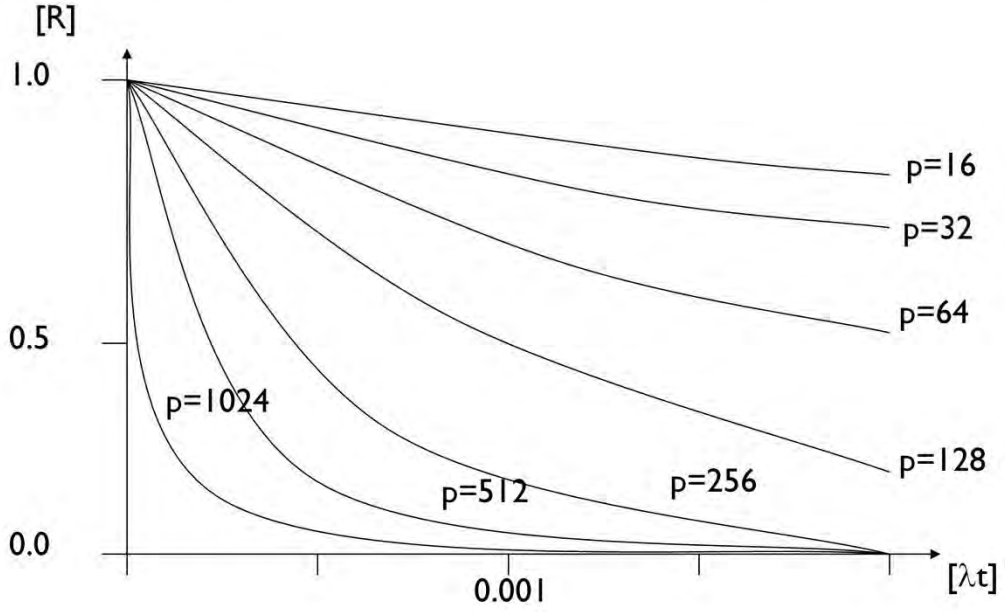
MTTF (mean time to failure) = $1/\lambda$

Zunächst das System ohne Fehlertoleranz:

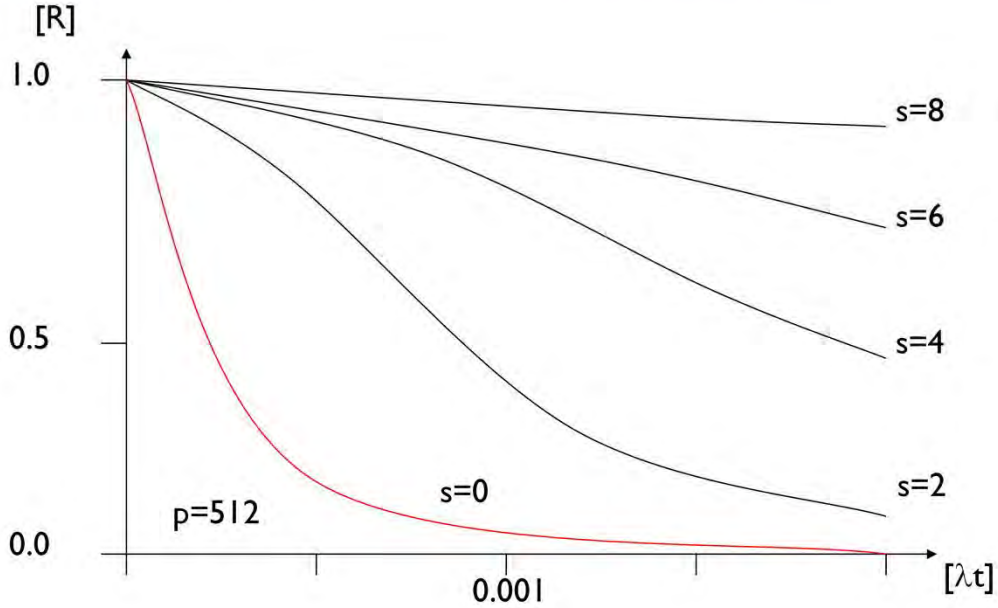
$R(t)p=e^{-p\lambda t}$

Anzahl p PMU´s (*processor memory unit*)

Quantitative Bewertung...



Quantitative Bewertung...



Quantitative Bewertung...

System kann bis zu s ausgefallene PMU´s tolerieren:

$R_{(p\text{-}s) \text{ aus } p}(t) = \sum_{i=0} c^i \, (1-R(t))^i R(t)^{p-i}$

c=P(Fehler wird behandelt | Fehler tritt auf)

(*coverage factor* - Überdeckungsfaktor)

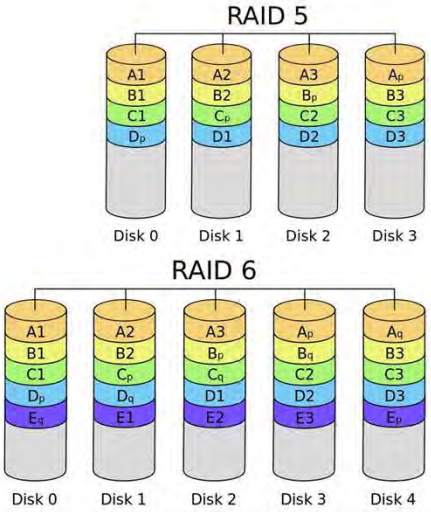
Umsetzungen in der Praxis

Fehlertolerierende Speichersysteme

- Error correcting code memory (ECC memory)
  - Ursachen für Bitfehler: z.B. elektrische/magnetische Einstrahlungen
  - Folgen: Veränderung von Code oder Daten im Hauptspeicher
  - Schutzmechanismus
    - Verwende 9. Bit
    - Nutze Hamming-Codes (statische Redundanz)
  - Problem
    - Erhöht HW-Kosten und Stromverbrauch um mindestens 1/8

Umsetzungen in der Praxis...

- RAID5/6 in Festplattensystemen
  - Probleme: Veränderungen einzelner Bit und Ausfälle von Platten
  - Folgen: Datenverlust droht
  - Schutzmechanismen
    - Verwende zusätzliche Platten
    - Nutze Paritätsinformation (statische Redundanz)
    - Nach Plattendefekte ersetze defekte Platte und rekonstruiere Inhalt
  - Probleme
    - Erhöht HW- und Betriebskosten
    - Geringere Leistung während Plattenrekonstruktion
    - Leistungseinbußen im Betrieb



Umsetzungen in der Praxis...

Fehlertolerierende Systeme in der Praxis

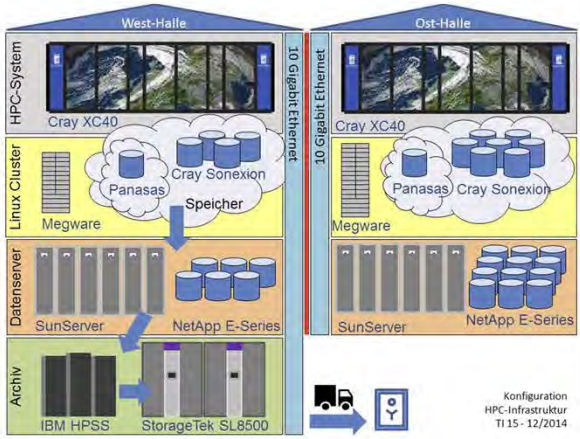
- Parallelrechner und Hochleistungsrechnen
  - Nahezu keine Realisierungen
  - Gründe
    - Systeme zu aufwendig
    - Nutzen zu gering, da Schäden zu gering
  - Ausnahme: Deutscher Wetterdienst
- Parallelrechner und kommerzielle Anwendungen
  - High-Availability-Computing
    - Hochverfügbare Systeme
    - Normalerweise keine parallelen Programme
    - Bereich Datenbanken, Systemsteuerungen u.ä.

- DKRZ
  - Plattensystem Mistral ca. 1% Ausfälle pro Jahr
  - Rechnerknoten ca. 5 Ausfälle am Tag
    - Teilweise nur Reboot
    - Durch Skripte vor Zuteilung durch Scheduler erkannt

Umsetzungen in der Praxis...

Systeme in der Praxis: Deutscher Wetterdienst

- Anbindung an Verkehrsministerium
  Verantwortet Wettervorhersage Flugverkehr
- Rechner- und Speicher-system verdoppelt
- Nutzung des Erstsystems
  - Operationelle Wettervorhersage
- Nutzung des Zweitsystems
  - Meteorologische Forschung
- Dynamische Redundanz
  - Umschaltung (Failover) im Fehlerfall
- Problem: Kosten



Umsetzung in der Praxis...

Verfügbare Software-Unterstützung

- High-Availability (HA) Linux Project
  - Definition von Verfahren, um Parallelrechner ausfallsicher zu machen
  - Cluster hier: Menge von HW-Komponenten, die als wechselseitige Redundanz dienen
- Linux High Availability HOWTO
- Viele kommerzielle HA-Cluster und HA-Lösungen

Fehlertoleranz
Zusammenfassung

- Fehlertoleranz ist ein wichtiges Thema bei allen Systemen, die aus sehr vielen Einzelkomponenten bestehen
- Wir unterscheiden im allgemeinen Systeme mit und ohne Reparatur
- Wir unterscheiden verschiedene Klassen von Fehlern
- Ein Fehlermodell beschreibt das System analytisch
- Redundanz ist notwendig, um Fehler tolerierbar zu machen
- Wir unterscheiden statische und dynamische Redundanzverfahren
- Statische Redundanz: Triple-Modular-Redundancy-Verfahren
- Dynamische Redundanz: Fail-Soft-Verfahren
- In der Praxis Anwendungen nur im Bereich des sogenannten High-Availability-Computing