

Verteiltes System ... mehrere Recheneinheiten arbeiten zusammen

Charakteristik:

- Mehrere Verarbeitungseinheiten
- Verbindungen zwischen den Einheiten
- Gemeinsamer Zustand
- Parallele Verarbeitung der Aufgaben

Ein User soll nicht merken das er vor einem verteilten System sitzt, es soll sich aehnlich anfehlen wie ein Singelcomputing System.

Der Programmierer muss darauf achten, das ein verteiltes System parallel arbeitet; diesem ist entweder im Code oder durch Verwendung entsprechender Bibliotheken rechnung zu tragen.

1.1 Transparenz:

Leistungsstranzperenz; d.h. mehr Knoten soll auch mehr Leistung bedeuten; scheitert oft auch an der parallelsisierbarkeit

Migrationstransperaenz: ein objekt soll erkennbar sein wenn es von einem Knoten auf einen anderen verschoben wird

Replikationstransparenz: Es muss sicher gestellt sein das beim Ausfall eines Knotens dessen Daten nicht verloeren gehen, d.h. mehrfachspeicherung; Der User will als Ergebnis nur die Daten, woher sie kommen ist ihm egal.

Hardwareaufbau:

Siehe Folien

Computer = Speicher & I/O sowie Prozessor (Steuerwerk und Register & ArithmetikLogicalUnit)

Multi Ralu:

Mehrere RALU (Register und aLU), jedoch das Steuerwerk und die I/O und Speichereinheiten nur einfach; micro bis millisekunden Kommunikationszeit; Diese Architektur ist nur anwendbar wenn der Algorithmus auf diese Architektur anwendbar ist; fuer eine Mulit Ralu ist eine sehr feinkoernige Unterteilung des Problem notwendig, d.h. es soll stark paralleliierbar sein.

Multiprozessor:

Mehrere Prozessoren uber ein Verbindungssystem mit dem I/O und Speicher verbunden; Kommunikationszeit im bereich milli bis Sekunden;

Multicomputer:

Jeder Computer hat seinen eigenen Prozessor, I/O und Speicher, ueber ein Kommunikationsnetzwerk sind diese Einheiten verbunden; in der Regel mit handelsueblichen Systemen (Consumer PCs und Speicher, sowie Netzwerksystemen) aufgebaut.

Motivation:

Divide & Conquer; Aufteilung zur Reduktion der Komplexitaet;
Ein Einprozessorsystem unterteilt die Aufgabe in einzelne Tasks, die Abarbeitung dieser erfolgt Sequenziell, man spricht von „pseudoparallel“; gibt es ein Mehrprozessorsysteme kann man Gruppen von Task bilden und diese den einzelnen Rechnern zu; Hier entsteht das Problem wie man die Gruppen (Partitionierung) bildet und wie teilt man diese Gruppen den Rechnern zu (Mapping) da es ev. unterschiedlich schnelle Rechner bzw. verschieden lange Anbindungen durch das Verbindngssystem im Verbund gibt?

Teilprozessbildung:

Problem in unabhangige Teile aufspalten (Inselbetrieb), dies reduziert den Kommunikationsaufwand;
z.B.: Ampelsteuerung in einer Stadt, jede Kreuzung einer einzelnen REcheneinheit zuweisen; fuer eine Verkehrsflussoptimierung sind dann wieder Infos von allen Kreuzungen erforderlich.

Datenverarbeitung vor Ort:

Daten dort verarbeiten wo sie anfallen, da Verkabelung teurer ist als Rechenleistung; ein etwaiges Problem ist die grosse Reaktionszeit und eine fragliche Skalierbarkeit (der Zentralrechner benoetigt fuer jeden Prozess einen eigenen Anschluss). z.B.: Temp. Sensor misst Temperatur, Recheneinheit vor Ort rechnet um in Grad Celcius und schickt sie an die zentrale Regelung; diese entscheidet ob der Heizungsregler auf oder zu dreht.

Hohe Rechenleistung:

Ist mehr Rechenleistung erforderlich als ein Einzelrechner liefern kann ist Parallelisierung notwendig; Anwendung gerne in Prognosen fuer Physik, Wirtschaft, ...
Realisiert wird dies gerne durch Desktop PCs in Cluster oder auch mit einem Verbund aus FPGAs fuer z.B.: Digitale-Bildverarbeitung
Aus dem Bsp. Digitales Filter (folien) ist gut erkennbar das mehr Prozessoren nicht unbedingt weniger Rechenzeit bedeuten, da die beiden Additionen von einander Abhaengig sind.
Darstellung dieser Verbesserungen durch Speedup, d.h. Halbierung des Zeitbedarfs ist ein Speedup von 2.
Durch eine starke Erhoehung der Prozessorzahl kann es durchaus zu einer laengeren Laufzeit kommen, da die Kommunikation zwischen den Knoten ebenfalls Zeit bedarf.
Verbesserung ist ev. noch durch die Vergroesserung des lokalen Speichers; sollte das gesamte Problem lokal gespeichert werden koennen bringt diese eine Performanceverbesserung da nicht staendig auf daten aus der Zentrale „gewartet“ werden muss.
Speedupdaten sind mit vorsicht zu geniesen da es sich um eine sehr spezifische abstrakte Betrachtung handelt; meist sollten sie auch geradzahlig sein.

Hohe Zuverlaessigkeit:

Faellt der zentrale Host aus, faellt das gesamte System aus; beim Verteilten Computing kann durch Umkonfiguration (Degradation, d.h. mit unter auch durch Verzicht auf Leistung) der Ausfall eines oder mehrer Knoten toleriert werden.

Parasive computing

Zugang fuer jeden zu jeder Zeit (ev. auch durch Tragbare Computer die als solches gar nicht mehr erkennbar sind)

Ein HID (Human Interface Device) ist notwendig damit der User mit dem System interagieren kann, ev. gibt es auch keine Daten bzw Energieleitungen mit Kabel zwischen den Systemen.

Anwendung als Location Based Services (PDA zeigt in Museen Iinfos zum Exponat an wenn man davor steht) (RFID Einsatz)