Sistemas Operacionais

Sistemas com Múltiplos Processadores

Introdução

- Sistemas com múltiplos processadores são arquiteturas que possuem duas ou mais UCPs interligadas e que funcionam em conjunto na execução de tarefas independentes ou no processamento simultâneo de uma mesma tarefa.
- Com a implementação de sistemas com múltiplos processadores, o conceito de paralelismo pôde ser expandido a um nível mais amplo.

Introdução

- A evolução desses sistemas se deve, em grande parte, ao elevado custo de desenvolvimento de processadores mais rápidos. Em função disso, passou-se a das ênfase a computadores com múltiplos processadores em vez de arquiteturas com um único processador de alto desempenho.

Introdução

- Os primeiros sistemas com múltiplos processadores surgiram na década de 1960, com o objetivo principal de acelerar a execução de aplicações que lidavam com um grande volume cálculos.
- O mercado corporativo começou realmente a utilizar na década de 1980, para melhorar o desempenho de suas aplicações e reduzir o tempo de resposta dois usuários interativos.

Introdução

- Posteriormente, as empresas reconheceram também nesse tipo de sistema uma maneira de aumentar a confiabilidade, a escalabilidade, e a disponibilidade, além da possibilidade do balanceamento de carga de suas aplicações.
- Atualmente, a maioria dos servidores de banco de dados, servidores de arquivos e servidores Web, utiliza sistemas com múltiplos processadores.

Vantagens

- Desempenho:
  - A princípio, sempre que novos processadores são adicionados à arquitetura de uma máquina, melhor é o desempenho do sistema.
  - O ganho de desempenho pode ser obtido em dois níveis, no primeiro nível, permitem a execução simultânea de diversas tarefas independentes, aumentando o throughput do sistema (ex.: Servidores de Banco de Dados e Servidores Web).

Vantagens

– No segundo nível, permitem a execução de uma mesma tarefa por vários processadores simultaneamente (processamento paralelo). Neste caso, o ganho de desempenho dependerá de diferentes fatores, como a organização dos processadores, a linguagem de programação utilizada e o grau de paralelismo possível na aplicação.

Vantagens

- Escalabilidade:
  - É a capacidade de adicionar novos processadores ao hardware do sistema.

Vantagens

- Relação Custo/Desempenho:
  - Sistema com um único processador, por mais poderosos que sejam, apresentam limitações de desempenho inerentes à arquitetura, devido às limitações existentes na comunicação da UCP com as demais unidades funcionais, principalmente a memória principal. Além disso, o custo do desenvolvimento de um processador que ofereça desempenho semelhante a um sistema com múltiplos processadores é muito elevado.

Vantagens

– Sistemas com múltiplos processadores permitem utilizar UCPs convencionais de baixo custo, interligadas às unidades funcionais através de mecanismos de interconexão. Desta forma é possível oferecer sistemas de alto desempenho com custo aceitável.

Vantagens

- Tolerância a falhas e disponibilidade:
  - Tolerância a falhas é a capacidade de manter o sistema em operação mesmo em casos de falha em algum componente. Nesta situação, se um dos processadores falhar, os demais podem assumir suas funções de maneira transparente aos usuários e suas aplicações, embora com menor capacidade computacional.

Vantagens

– A disponibilidade é medida em número de minutos por ano que o sistema permanece em funcionamento de forma ininterrupta, incluindo possíveis falhas de hardware ou software, manutenções preventivas e corretivas. Uma alta disponibilidade é obtida com sistemas com maior tolerância a falhas. Sistemas de alta disponibilidade são utilizados em aplicações de missão crítica, como sistemas de tráfego aéreo e de comércio eletrônico na Internet.

Vantagens

- Balanceamento de carga:
  - É a distribuição do processamento entre os diversos componentes da configuração, a partir da carga de cada processador, melhorando, assim, o desempenho do sistema como um todo. Servidores de banco de dados, que oferecem esse tipo de facilidade, permitem que as solicitações dos diversos usuários sejam distribuídas entre os vários processadores disponíveis.

Desvantagens

- Novos problemas de comunicação e sincronização são introduzidos, pois vários processadores podem estar acessando as mesmas porções de memória. Além disso, existe o problema de organizar os processadores, memórias e periféricos de uma forma eficiente, que permita uma relação custo/desempenho aceitável.

Sistemas Fortemente e Fracamente Acoplados

- Sistemas Fortemente Acoplados: os processadores compartilham uma única memória principal e são controlados por apenas um único SO.
- Sistemas Fracamente Acoplados: caracterizam-se por possuir dois ou mais sistemas computacionais independentes, conectados por uma rede de comunicação, tendo cada sistema seus próprios processadores, memória principal, dispositivos de E/S e SO.

Sistemas Fortemente e Fracamente Acoplados

- A grande diferença entre os dois tipos de sistemas é que em sistemas fortemente acoplados existe apenas um espaço de endereçamento compartilhado por todos os processadores, também chamado de “memória compartilhada”.
- Nos sistemas fracamente acoplados, cada sistema tem seu próprio espaço de endereçamento individual e a comunicação entre os sistemas é realizada através de mecanismos de troca de mensagens, utilizando operações de send e receive.

Sistemas Fortemente e Fracamente Acoplados

- Podemos relacionar os sistemas fortemente acoplados aos multiprocessadores e os sistemas fracamente acoplados aos multicomputadores.

Sistemas com Multiprocessadores Simétricos

- Symmetric Multiprocessors – SMP: possuem dois ou mais processadores compartilhando um único espaço de endereçamento e gerenciados por apenas um SO.
- Uma característica importante nos sistemas SMP é que o tempo de acesso à memória principal pelos vários processadores é uniforme, independente da localização física do processador. Esse tipo de arquitetura também é conhecido como UMA (uniform memory access).

Sistemas NUMA

- Non-Uniform Memory Aceess – NUMA: O tempo de acesso à memória principal depende da localização física do processador. Nesta arquitetura, existem vários conjuntos, reunindo processadores e memória, sendo cada conjunto conectado aos outros através de uma rede de interconexão. Todos os conjuntos compartilham um mesmo SO e referenciam o mesmo espaço de endereçamento.

Clusters

- São fracamente acoplados, formados por nós conectados por uma rede de interconexão de alto desempenho dedicada. Cada nó da rede é denominado membro do cluster, e possui seus próprios recursos, como processadores, memória, dispositivos de E/S e SO. Geralmente, os membros do cluster são de um mesmo fabricante, principalmente por questões de incompatibilidade dos SO.

- Cada membro do cluster possui seu próprio espaço de endereçamento individual, e a comunicação entre os membros se faz, na maioria das implementações, pelo mecanismo de troca de mensagens através da rede de interconexão.
- Geralmente, a rede de interconexão é restrita aos membros do cluster, e o acesso aos serviços oferecidos é feito a partir de uma outra rede de acesso.

- A razão para o surgimento e a rápida aceitação de sistemas em cluster foi a maior necessidade de tolerância a falhas e a alta disponibilidade, de forma a reduzir o downtime (quantidade de minutos e horas por mês nos quais o servidor fica fora do ar para manutenção técnica preventiva e corretiva).

- O downtime pode ser reduzido não apenas contornando problemas de hardware e software, mas também reduzindo o impacto de manutenções preventivas e atualizações.
- Outras vantagens de um cluster são sua escalabilidade e o balanceamento de cargas.
- Por essas razões, clusters são utilizados em servidores Web, sistemas de comércio eletrônico, servidores de banco de dados e soluções de firewall.

Sistemas Operacionais de Rede

- São o melhor exemplo de um ambiente fracamente acoplados. Cada sistema, também chamado host ou nó, possui seus próprios recursos de hardware, como processadores, memória e dispositivos de E/S. Os nós são totalmente independentes dos demais, sendo interconectados por uma rede de comunicação de dados formando uma rede de computadores.

Sistemas Operacionais de Rede

- São utilizados tanto em rede locais (Local Area Network – LAN), como Ethernet e Token Ring, quanto em redes distribuídas (Wide Area Network – WAN), como a Internet.
- A princípio, não existe um limite máximo para o número de nós que podem fazer parte de uma rede de computadores.

Sistemas Operacionais de Rede

- Cada nó é totalmente independente dos demais, possuindo seu próprio SO e espeço de endereçamento. Os SO podem ser heterogêneos, bastando apenas que os nós comuniquem-se utilizando o mesmo protocolo de rede (ex: TCP/IP).
- Não existe a idéia de imagem única do sistema, como também não existe a tolerância a falhas e a alta disponibilidade presente nos sistemas distribuídos.

Sistemas Operacionais de Rede

- A grande maioria dos sistemas operacionais de rede e seus protocolos de rede implementam o modelo cliente/servidor.

Sistemas Distribuídos

- É um conjunto de sistemas autônomos, interconectados por uma rede de comunicação e que funciona como se fosse um sistema fortemente acoplado. Cada componente de um sistema distribuído possui seus próprios recursos, como processadores, memória principal, dispositivos de E/S, SO e espaço de endereçamento. Os tipo de SO que compõem não precisam ser necessariamente homogêneos.

Sistemas Distribuídos

- O que diferencia um sistema distribuído dos demais sistemas fracamente acoplados é a existência de um relacionamento mais forte entre os seus componentes. Podemos definir um sistema distribuído como sendo um sistema fracamente acoplado pelo aspecto de hardware e fortemente acoplado pelo aspecto de software.

Sistemas Distribuídos

- Para o usuário e suas aplicações, é como se não existisse uma rede de computadores independente, mas sim um único sistema fortemente acoplado. Este conceito é chamado de “imagem única do sistema (single system image)”.

Sistemas Distribuídos

- Os sistemas distribuídos permitem que uma aplicação seja dividida em diferentes partes, que se comuniquem através de linhas de comunicação, podendo cada parte ser executada em qualquer processador de qualquer sistema (aplicações distribuídas). Para que isto seja possível, o sistema deve oferecer transparência e tolerância a falhas em vários níveis, a fim de criar a idéia de imagem único do sistema.

Sistemas Distribuídos - Transparência

- Transparência de acesso: é a possibilidade de acesso a objetos locais ou remotos de maneira uniforme.
- Transparência de localização: significa que o usuário não deve se preocupar onde estão os recursos de que necessita.
- Transparência de migração: permite que os recursos sejam fisicamente movidos para outro sistema, sem que os usuários e suas aplicações seja afetados.

Sistemas Distribuídos - Transparência

- Transparência de replicação: permite a duplicação de informações, com o objetivo de aumentar a disponibilidade e o desempenho do sistema, de forma sincronizada e consistente.
- Transparência de concorrência: permite que vários processos sejam executados paralelamente e os recursos sejam compartilhados de forma sincronizada e consistente

Sistemas Distribuídos - Transparência

- Transparência de paralelismo: possibilita que uma aplicação paralela seja executada em qualquer processador de qualquer sistema, como em um sistema fortemente acoplado.
- Transparência no desempenho: oferece aos usuários tempos de resposta independentes de alterações na estrutura do sistema ou na sua carga. Além disso, operações realizadas remotamente não devem apresentar resultados piores do que realizadas localmente.

Sistemas Distribuídos - Transparência

- Transparência de escalabilidade: permite que o sistema cresça sem a necessidade de alterar as aplicações e seus algoritmos.
- Transparência a falhas: garante que, em caso de falha de um sistema, as aplicações continuem disponíveis sem interrupção.

Sistemas Distribuídos - Transparência

- Ao executar uma aplicação, o usuário não saberá em quais ou quantos componentes a sua aplicação foi dividida. Caso um erro ocorra em um desses componentes, o usuário não terá conhecimento, ficando como responsabilidade do SO a resolução de todos os problemas.

Sistemas Distribuídos – Tolerância a Falhas

- Para que um sistema distribuído possa oferecer transparência é preciso que o sistema implemente “tolerância a falhas” de hardware e, principalmente, de software.
- A tolerância de falhas de hardware é facilmente oferecida utilizando-se componentes redundantes, como fontes duplicadas, vários processadores, memória com detecção e correção de erro e técnicas de RAID. Passa também pela redundância dos meios de conexão entre os sistemas, como placas de rede, linhas de comunicação e dispositivos de rede.

Sistemas Distribuídos – Tolerância a Falhas

- A tolerância a falhas de software é bem mais complexa de implementar. Quando uma falha deste tipo ocorre, como uma falha no SO, a aplicação deve continuar sem que o usuário perceba qualquer problema, que deve ser resolvido de forma transparente, mantendo a integridade e consistência dos dados.

Sistemas Distribuídos – Tolerância a Falhas

- Com a tolerância falhas, é possível também oferecer alta disponibilidade e confiabilidade.

Sistemas Distribuídos – Imagem Única do Sistema

- A maior dificuldade em implementar um sistema distribuído é a complexidade em criar para os usuários e suas aplicações uma imagem única do sistema a partir de um conjunto de sistemas autônomos.
- Para conseguir criar um ambiente fisicamente distribuído e logicamente centralizado é necessário um SO capaz de lidar com os diversos problemas de comunicação existentes em um ambiente fracamente acoplado.

Sistemas Distribuídos – Imagem Única do Sistema

- Um problema encontrado em sistemas fortemente acoplados é o compartilhamento de recursos de forma segura. Em sistemas distribuídos, a utilização de recursos concorrentemente exige mecanismos mais complexos e lentos para manter a integridade e a segurança dos dados.

Sistemas Distribuídos – Imagem Única do Sistema

- Um dos grandes desafios para a adoção de sistemas distribuídos é a dificuldade no desenvolvimento de aplicações paralelas. Enquanto a programação em sistemas fortemente acoplados é relativamente transparente, em sistemas distribuídos o desenvolvimento não é tão simples. Apesar de algumas aplicações serem naturalmente paralelas, como ordenações e processamento de imagens, desenvolver aplicações distribuídas exige uma grande interação do programador com detalhes de codificação e escalonamento da aplicação.