Sistemas de Computação

Mestrado Integrado em
Engenharia de Telecomunicações e Informática

2015/2016

Multiprocessamento

- Sonho dos arquitectos dos computadores
  - Criar computadores poderosos simplesmente ligando muitos computadores pequenos actualmente existentes
  - Visão de num modelo ideal as pessoas encomendam o computador com o número de processadores que podem pagar
  - Os sistemas com multiprocessadores seriam escaláveis: o hardware e o software é desenhado para ser vendido e funcionar com um número variável de processadores
    - Se um processador falhar, o sistemas continua a funcionar com *n-1*

Multiprocessamento (II)

- Algumas questões
  - Como é que os processadores partilham os dados?
  - Como é que se coordena a execução do(s) programa(s)?
  - Quantos processadores?

Multiprocessamento (II)

- Partilha de dados
  - Espaço único de endereçamento (também se chama de *shared-memory processors)* – existe um espaço de endereçamento de memória único. Os processadores comunicam através de variáveis partilhadas em memória, sendo que todos os processadores são capazes de aceder a qualquer posição de memória
    - Necessário coordenação entre os processadores – a isto chama-se *synchronization*
    - Duas opções:
      - UMA – Uniform Memory Access ou SMP – Symmetric Multiprocessors – Leva o mesmo tempo a aceder à memória principal independentemente do processador que quer aceder e da posição da memória que é necessário aceder
      - NUMA – NonUniform Memory Access – Algumas memórias são mais rápidas do que outras, dependendo do processador que está a aceder
  - Comunicação usando mensagens
    - Este modelo é o único possível quando se trata de sistemas com memória "privada".
    - Exemplo extremo: um conjunto de computadores ligados numa rede local e que funcionam como sendo um grande sistema multiprocessador - CLUSTER

Multiprocessamento (III)

- Poucas aplicações foram escritas/reescritas para completarem uma determinada tarefa de forma mais célere em sistemas multiprocessador
- Porque é que os programas com processamento paralelo são muito mais difíceis de desenvolver do que os programas sequenciais?
  - Tem de obter uma boa performance e eficiência de um programa paralelo a ser executado num multiprocessador.
    - Senão o melhor é continuar com o sistema para um só processador que é mais simples de implementar
  - Porque é difícil criar programas para sistemas multiprocessador?
    - *Overhead*
      - *Nota: exemplo em que vários jornalistas escrevem uma história*
    - Necessidade de conhecer o hardware para tirar partido dele. Num sistema uni processador isso não acontece (faz-se sempre da mesma forma, escrevendo numa linguagem de alto nível e ignorando o hardware). Por outro lado, uma solução específica para um determinado modelo paralelo poderá não funcionar noutro modelo.

Multiprocessamento (IV)

- *Clusters*
  - O custo de administrar um cluster com N máquinas é quase o mesmo que administrar N máquinas independentes. O custo de administrar um sistema com N processadores e um espaço de endereçamento partilhado é quase o mesmo que administrar um máquina com um processador;
  - Nos *cluters* as várias máquinas comunicam usando o bus de I/O (rede) enquanto que os sistemas com múltiplos processadores comunicam usando a memória;
  - A memória está dividida pelas N máquinas do cluster que possuem N memórias independentes onde estão N cópias do sistemas operativo;
  - As fraquezas da memória partida por várias máquinas é, afinal, a sua maior força em termos de disponibilidade e expansibilidade:
    - Cada máquina pode ser substituída sem desligar o *cluster*;
    - É quase impossível isolar e substituir um processador de uma máquina sem a desligar (como acontece num sistema SMP).
    - Como o software de cluster é uma camada executada sobre o sistemas operativo então é mais fácil desligar e substituir uma máquina
    - Da mesma forma: torna-se mais fácil expandir o sistema, sem paragem no serviço

Multiprocessamento (IV)

- *Clusters*
  - Outra grande diferença: preço
    - Os sistemas multiprocessador são poucos pelo que os custos de desenvolvimento tem de ser amortizado num número reduzido de sistemas, tornando-os caros.
  - Sistemas híbridos

Cloud computing

- *Cloud computing* – uso de recursos computacionais (hardware e software) que é disponibilizado como um serviço através de uma rede (tipicamente a Internet).
- Os fornecedores de *cloud* gerem a infra-estrutura e a plataforma na qual as aplicações são executadas
- Modelos fundamentais:
  - *Infrastructure as a service* (IaaS)
    - Exemplo: Máquinas virtuais, servidores, espaço em disco, etc.
  - *Platform as a service* (PaaS)
    - Exemlo: Base de dados, web server, etc.
  - *Software as a service* (SaaS)
    - Exemplo: CRM, E-mail, etc.
  - *Network as a service* (NaaS)
    - Exemplo: VPN
  - *Communication as a service* (CaaS)
    - Exemplo: serviço de gravação de voz, serviço de fax, etc.

SaaS (I)

- *Software as a Service* (SaaS)
  - Por vezes designado de "*on-demand software*"
  - Modelo de disponibilização de software no qual as aplicações e os dados associados são executados e guardados algures na Internet
    - Os utilizadores acedem usando pequenos clientes (muitas vezes via *browser*)
  - Nos últimos anos tornou-se muito popular em inúmeras áreas: colaboração, contabilidade, finanças, gestão de clientes (incluindo CRM), etc.
  - Apresentado aos clientes como sendo um excelente modelo porque potencialmente reduz o custo com o suporte de IT, ao fazer o *outsourcing* da manutenção do hardware e do software
  - É considerado parte do "cloud computing", tal como
    - *infrastructure as a service* (IaaS)
    - *platform as a service* (PaaS)
    - *backend as a service* (BaaS)
    - Etc.

SaaS (II)

- *Software as a Service* (SaaS)
  - Custo: em vez de uma licença perpetua, é vendido ao mês/ano, ou com base no número de utilizadores, no número de transacções, etc.
    - Apresenta muitas vezes um custo inicial inferior face à solução "tradicional"
  - Baseado numa arquitectura que permite suportar inúmeros clientes
    - Os parâmetros da aplicação permitem configura-la segundo as necessidades de cada cliente
  - Software actualizado frequência
  - Protocolos normalizados permitem a interacção com aplicações específicas do cliente
  - Desvantagens
    - Segurança dos dados;
    - Velocidade
      - Sendo executado na rede (Internet), o tempo de resposta poderá ser inferior;
    - Sendo uma aplicação que é disponibilizada a inúmeros clientes então o nível de personalização pode ser limitado;
    - Mudança da SaaS pode ser complexo e lento, dado o imenso volume de dados que poderá ser necessário transferir;
    - Algumas organizações podem não querer migrar para as novas versões do software pelos custos que isso representa em termos de formação/aprendizagem.