3. Índices de Carga

Um dos objetivos centrais de todo sistema computacional é a utilização eficiente dos recursos disponíveis. Particularmente, considerando-se os sistemas distribuídos, esse objetivo torna-se fundamental, estando diretamente ligado ao desempenho global obtido.

O gerenciamento efetivo de recursos desse tipo de sistema requer estratégias para o escalonamento de processos entre os múltiplos elementos de processamento disponíveis, visando, por exemplo, ao balanceamento de cargas de trabalho e, com isso, melhor desempenho global.

Independentemente do tipo de escalonamento a ser considerado (estático, dinâmico, tempo-real, etc), um elemento essencial para a política de escalonamento é a aferição da carga.

A aferição da carga, constituindo um índice de carga não é um processo trivial, particularmente considerando-se a natureza dinâmica e não determinística das aplicações que são executadas.

Bons índices de carga e, por extensão, bons índices de desempenho podem levar à obtenção de melhorias no desempenho global observado no sistema.

Neste capítulo são apresentados conceitos e características de diversas métricas utilizadas para a aferição de cargas em sistemas computacionais distribuídos e a caracterização dessas cargas de trabalho, informações indispensáveis para a execução de um bom escalonamento.

3.1 Considerações Iniciais

A avaliação de desempenho em sistemas computacionais constitui uma ampla área de pesquisa, podendo ser vista sob diferentes enfoques em função dos objetivos almejados.

A medida de desempenho, por si só, é um ponto particularmente importante, e é normalmente representada através da definição de métricas.

Quando é considerada a computação paralela/distribuída, a definição e obtenção dessas métricas tornam-se mais complexas, uma vez que, em um sistema

computacional distribuído, a busca por melhor desempenho é planejada através da exploração da potência computacional disponível em um conjunto de elementos de processamento.

Existem na literatura vários modelos para obtenção de índices de carga em ambientes arquitetural e configuracionalmente homogêneos, entretanto, não se encontra tal gama de modelos quando se consideram ambientes heterogêneos, principalmente sob o enfoque de heterogeneidade arquitetural.

3.2 Definição de Métrica e Medida

Na computação, assim como em outras áreas, existem várias quantidades relacionadas com o desempenho e com a confiabilidade dos recursos e componentes das máquinas que se gostariam de se saber quais os seus valores.

Segundo Paxson (Paxson et al, 1998), quando uma dessas quantidades é especificada, ou seja, quando se refere a uma propriedade de um recurso, essa grandeza ou propriedade é denominada *métrica*. Valores quantificados de uma métrica são denominados *medidas*. Quando uma métrica é quantificada, sua medida deve ser apresentada em termos de unidades padronizadas.

Em alguns casos, não existe forma óbvia de se efetivamente medir uma métrica; isto é permitido e até compreensível em alguns casos. Entretanto, a especificação da métrica deve ser o mais clara possível sobre qual quantidade está sendo especificada, ou seja, a métrica pode ser difícil de ser quantificada, mas não pode ser ambígua.

Tão importante quanto apresentar as medidas obtidas nas medições de uma métrica, é apresentar também o contexto da medição. Pode-se definir um *contexto de medição* como aqueles elementos do sistema utilizados para fazer as medições que estão relacionadas com o componente que está sendo medido. Em geral, o perfeito entendimento dos efeitos do contexto é crucial para que se possam fazer medições coerentes.

Existem dois tipos básicos de métricas, *métricas analíticas* e *métricas empíricas*. As primeiras referem-se às métricas definidas em termos das propriedades teóricas e abstratas dos componentes. Estas são as propriedades utilizadas para analisar os componentes matematicamente (Paxson, 1996; Paxson

et al, 1998). Métricas empíricas referenciam as propriedades definidas diretamente a partir das medições.

Neste trabalho as métricas, basicamente empíricas, a serem utilizadas estarão alicerçadas nos conceitos apresentados anteriormente, que se adequam perfeitamente ao contexto deste trabalho.

3.3 Estado da Arte

Em um sistema computacional distribuído, o potencial para compartilhamento de recursos e seus possíveis ganhos são substanciais. As grandes vantagens oferecidas pelo compartilhamento de recursos são o grande número de recursos acessíveis (tanto em tipo quanto em quantidade) e o alto potencial devido à grande variedade desses recursos.

De maneira a obter melhor uso dos recursos disponíveis, algumas medidas e métricas que fornecem informações sobre esses recursos têm sido apresentadas. Uma das informações pertinentes ao estado do sistema e que possui como característica básica ser rapidamente mutável é a carga submetida a esses recursos.

Uma vez que essas cargas alteram-se muito rapidamente (dependendo basicamente do domínio da aplicação), elas tendem a ficar obsoletas muito rapidamente.

Enquanto a necessidade de atualização desse índice de carga é óbvia, a escolha da freqüência dessa atualização constitui, inevitavelmente, um compromisso dependente da característica da aplicação.

A informação da carga de um recurso é geralmente definida em termos de índices de carga. Esse índice de carga é uma métrica que quantifica a carga submetida a um elemento do sistema (Ferrari & Zhou, 1987; Kunz, 1991) e tem por objetivo indicar se o elemento considerado está ocioso (inexistência de carga ou um valor muito pequeno), se está sobrecarregado (condição em que o escalonador deve evitar alocações de processos nesses recursos, e até mesmo considerar a possibilidade de migração, de modo a aliviar a carga do elemento), ou se o elemento considerado está em uma situação normal de processamento, podendo ter disponibilidade para aceitar mais alguns processos.

O índice de carga pode ser, então, definido formalmente como uma variável numérica, não negativa, assumindo valor zero quando o recurso está ocioso e tendo seu valor acrescido positivamente quando a carga desse recurso aumenta (Ferrari & Zhou, 1987; Kunz, 1991).

Identificar o índice de carga apropriado, de maneira a obter um aumento considerável da utilização dos recursos ociosos é, comprovadamente, um ponto importante a ser considerado no projeto de qualquer tentativa de avaliação de desempenho e, principalmente, no projeto de um algoritmo de balanceamento de carga.

Um índice de carga, para ser dito eficaz, deve refletir precisamente, ou o mais próximo possível, o estado do recurso avaliado (Ferrari & Zhou, 1987) tendo, assim, a finalidade de predizer o desempenho de uma tarefa se esta for executada em uma determinada máquina.

Contudo, apesar dessas técnicas utilizadas para a medição da carga serem eficientes, elas devem impor uma sobrecarga mínima ao sistema (Mehra, 1993), de modo que o custo para a obtenção desse índice não interfira no desempenho do sistema (Ferrari & Zhou, 1987; Zhou, 1987; Kunz, 1991; Dantas & Zaluska, 1998).

Sendo assim, o índice de carga deve estar voltado para o objetivo ao qual se propõe o algoritmo de escalonamento de processos, informando não só a carga atual do sistema, mas ser utilizado também para predizer um comportamento futuro, baseando-se em uma situação presente ou em um passado recente (Kunz, 1991; Schnor et al, 1996).

As características que seguem são pontos importantes a serem considerados quando a escolha do índice de carga se faz necessária:

- necessidade de quantificar não só a utilização do processador necessário para os processos, mas também os requisitos de memória e I/O;
- predizer cargas futuras em um recurso, uma vez que essas cargas afetam mais no desempenho de um sistema do que a carga presente;
- permitir uma relação simples com o índice de desempenho, de maneira que esses valores possam facilmente ser traduzidos;

- adotar índices estáveis, desprezando valores flutuantes.

Não existe, entretanto, um consenso quanto à eficácia e à eficiência de um único índice de carga. Dessa forma, existe na literatura uma gama de índices de carga.

Alguns dos índices que têm sido usados, estudados, analisados e avaliados incluem: o tamanho da fila de CPU, a média do tamanho da fila de CPU em um intervalo de tempo determinado, a utilização de CPU, a taxa de chamada do sistema, a quantidade de memória disponível, o tempo de resposta e funções híbridas que utilizam junções e combinações dos índices citados entre outros (Ferrari & Zhou, 1987; Zhou, 1987; Kunz, 1991; Xu & Lau, 1997; Dantas & Zaluska, 1998).

Genericamente falando, os índices de carga podem ser divididos em grupos como: baseados no tamanho da fila de acesso ao recurso, no percentual de utilização do recurso, no tempo de execução/resposta (em medidas de desempenho). Esses grupos de índices podem ser ainda ditos *índices específicos*: quando um único valor é obtido e segundo alguns autores são bons para casos específicos e impõe menor sobrecarga (Kunz, 1991; Kumar et al, 1993; Bolter & Downey, 1995); e *índices genéricos*: quando existe a união de um ou mais valores obtidos, sendo indicados por alguns autores quando não se tem um conhecimento adequado da aplicação nem sobre os objetivos do escalonador (Ferrari & Zhou, 1987; Zhou et al, 1992; Zhou et al, 1993).

Os índices de carga genéricos, além de apresentarem uma tendência à sobrecarga maior, normalmente não apresentam a mesma qualidade de representação de carga de trabalho, quando comparados com os índices específicos utilizados corretamente (Mehra 1993; Ferari & Zhou, 1987).

Nos trabalhos que abordam índices de carga, é também comum o uso de modelos abstratos de filas que representem sistemas computacionais para a obtenção analítica de funções de índices de carga. Destacam-se dentre esses modelos os propostos por Ferrari e Zhou (Ferrari & Zhou, 1987) (Zhou et al., 1992) (Zhou et. al., 1993).

A grande maioria dos modelos para cálculo de índices de carga existentes na literatura é para ambientes configuracionalmente e arquiteturalmente homogêneos.

Apenas uma pequena minoria desses modelos contempla a heterogeneidade e, ainda assim, apenas a heterogeneidade configuracional. Exemplificando esses modelos (tanto os homogêneos como os heterogêneos) têm-se:

- (Ferrari & Zhou, 1987) propõem a utilização de um índice de carga obtido através da combinação linear do tempo de serviço $\mathbf{s_j}$ requerido por uma tarefa para sua execução em um determinado recurso $\mathbf{r_j}$, sendo que o comprimento da fila do recurso $\mathbf{r_j}$ é dado por $\mathbf{q_j}$, de tal modo que o índice de carga (li = load index) é obtido através de:

$$li = \sum_{j=1}^{N} s_j \times q_j \qquad \textbf{(Equação 3.1)}$$

  onde **N** é o número total de recursos que possuem filas. Esse modelo contempla ambientes compostos por máquinas configuracionalmente e arquiteturalmente homogêneas;

- (Lin & Keller, 1987) utilizaram em sua pesquisa um índice de carga que leva em consideração informações obtidas a partir do número de tarefas e da quantidade de memória que está sendo utilizada. Baseados em um simulador (Rediflow (Lin & Keller, 1987)) foi possível a especificação de diversos parâmetros incluindo o número de máquinas, a quantidade de memória, a composição configuracional das máquinas, a largura banda de comunicação entre outros. Partindo-se dos dados fornecidos pelo simulador, o índice de carga (IC) é então calculado através de:

  IC=número-de-tarefas + parâmetros/(1- memória_em_uso)

  onde parâmetros são os parâmetros fornecidos pelo simulador;

- (Theimer & Lantz, 1989) efetuaram os mesmos estudos realizados em (Ferrari & Zhou, 1987), contudo, acrescentaram um maior número de computadores, em torno de 70. Em suas análises, Theimer e Lantz concluíram que algoritmos de escalonamento de processos que utilizam como índice de carga o comprimento da fila de processos. Assumem também a homogeneidade do sistema, o que não reflete a ocupação real dos computadores. Esse tipo de índice degrada o desempenho dos algoritmos de balanceamento de carga.

- (Kunz, 1991) foi apresentado nesse estudo que o escalonamento efetuado fazendo uso de um índice de carga qualquer se apresenta muito melhor que um sistema que não efetua balanceamento (assim como previamente demonstrado por Ferrari & Zhou). Fez-se um estudo onde avaliou o desempenho obtido pelo escalonador quando este efetua balanceamento de carga com relação a seis índices de carga lineares, e foi obtido como resultado que o melhor dentre eles é o número de processos na fila de prontos. Outra avaliação efetuada por Kunz em seus estudos foi a tentativa de utilização de índices de carga agregados, entretanto, através dos experimentos realizados foi constado que esses índices não acrescentavam melhoria no desempenho do sistema quando comparados aos índices lineares e que, por outro lado, ocasionavam sobrecarga na obtenção dos diversos índices lineares que iriam compor os índices agregados. Mais uma vez, os estudos aqui efetuados foram realizados em ambientes tipicamente homogêneos;
- (Zhou et. al., 1993) propõem que os índices de carga variem conforme a natureza do recurso que se está avaliando. Dessa maneira esses índices serão possivelmente o comprimento de fila, a utilização ou a quantidade de recurso livre. A proposta implica que para cada máquina, alguns índices de carga de recursos específicos sejam utilizados. Os índices utilizados são obtidos através da média do comprimento de fila de CPU, quantidade de memória livre, média da taxa de transferência de um disco para todos os outros discos quando efetuado um I/O, quantidade de espaço disponível em disco para troca de páginas e o número de usuários existentes no sistema;
- (Mehra & Wah, 1993) utilizou em sua pesquisa técnicas de redes neurais para obtenção de índices de carga. No modelo é levada em consideração somente a heterogeneidade configuracional. Entretanto, o cálculo do índice de carga das máquinas pertencentes ao sistema requer cinco dias de processamento. Durante esses cinco dias o cálculo do índice é efetuado com base no aprendizado da rede neural. O índice de carga considera: S é o número de máquinas, F a carga de trabalho em *foreground*, T o tempo de captura da carga e B a carga de trabalho em

background. Assim, $l_{b,f(t),s}$ é o vetor que contém o nível de utilização (em uma máquina de **S**) de CPU, memória, disco e rede em cada tempo **T** para **b** e **f** começados em **t**. Na verdade tem-se uma matriz de quatro linhas (representando os recursos: disco memória, rede e disco) e uma coluna (representando o tempo T – o instante de tempo em que está sendo efetuada a medição). Sendo assim, $l_{b,f(t),s}$ é o vetor de valores derivados do comportamento recente das cargas presente nas diferentes máquinas tanto para cargas em *foreground* (**f**) como em cargas *background* (**b**) iniciadas em **t**, nesse caso é assumido que em todas as máquinas o tempo de obtenção do índice é o mesmo, apesar disso não acontecer na realidade. O vetor obtido no tempo t é então dividido por um vetor obtido a partir de uma máquina de referência, isto é, uma máquina ociosa. Dados esses valores o objetivo do aprendizado do índice de carga é:

$$F_s^w \left[\hat{l}_{b,f,s}, f \right]$$

$$F_s \left[\hat{l}_{b,f,s1}, f \right] = \frac{C \left[\hat{l}_{b,f,s1}, f \right]}{C \left[\hat{l}_{(),f,sref}, f \right]} \qquad \textbf{(Equação 3.2)}$$

- (Franklin & Govindan, 1996) propõe um modelo iterativo de matriz para a obtenção de índice de carga considerando que as tarefas são idênticas e que pode existir uma diferença na potência computacional dos processadores, entretanto a homogeneidade arquitetural também deve ser mantida. Segundo essas definições, têm-se que o número de processadores é dado por ***p*** e a potência computacional é dada por C_i, e esta por sua vez é proporcional ao número médio de operações executadas pelos processadores em uma unidade de tempo. Dessa forma, o tempo gasto para a execução de uma tarefa em P_i é inversamente proporcional a C_i. Sendo assim, o índice de carga é obtido através da alocação do número de tarefas de maneira proporcional ao respectivo poder computacional da máquina;
- (Wolffe, Hosseini & Vairavan, 1997) propõe a utilização da Capacidade de Carga como sendo um índice de carga para ambientes heterogêneos. Por

capacidade de carga entende-se a utilização efetiva do processador. É considerado um índice de carga para ambientes heterogêneos por efetuar a normalização da velocidade de cada CPU com relação as demais.

(1 – Utilização_de_CPU) * Velocidade_Relativa_da_CPU

Essa métrica representa a capacidade de processamento efetivamente restante no recurso processador, entretanto não leva em conta os demais recursos que estão envolvidos no processamento como um todo, como, por exemplo, memória, disco, rede, o que o torna um índice muito específico e pouco flexível.

Em adição às considerações tecidas com relação a esse índice, deve ser salientado que os experimentos foram executados em máquinas arquiteturalmente homogêneas.

- (Fontlupt et. al., 1998) propõem a obtenção da carga como sendo o número de itens de dados existente na fila do processador. A função de carga é denotada por *w*. A carga total do sistema é dada por *W*, sendo este obtido através de:

$$W = \sum_{i=1}^{P-1} w[i] \qquad \textbf{(Equação 3.3)}$$

considerando um conjunto de *P* processadores, sendo esses completamente homogêneos e considerando as tarefas a serem executadas nesses processadores também homogêneas, tem-se que a média total nesses sistemas é dada por *W/P* e denotada por w.

De modo geral, o que a literatura recomenda é a adoção de um índice de carga que seja a média de algumas características essenciais de alguns recursos sobre um determinado tempo, diminuindo a possibilidade de escolher algum valor que não corresponda ao estado real do sistema (Ferrari & Zhou, 1987).

Há, dessa maneira, grandes diferenças na eficiência de cada índice de carga, ficando provado através de diversas pesquisas realizadas (Ferrari & Zhou, 1987; Theimer & Lantz, 1989; Shirazi & Hurson, 1992; Kunz, 1991), que os índices de carga mais simples são particularmente os mais eficientes, refletindo melhor o estado do sistema.

O índice de carga mais comumente assumido como melhor índice na literatura é o comprimento de fila de processos na CPU. Esse índice é amplamente utilizado (Shivaratri et al, 1992) por permitir uma modelagem analítica e é freqüentemente utilizado em análise teórica. A maior vantagem por ele proporcionada é o fato de permitir a caracterização matemática precisa, além de ser também muito atrativa para a simulação.

A utilização desse índice não funciona corretamente em ambientes onde os computadores têm capacidade de processamento heterogênea e os processos não têm ocupação semelhante (Shivaratri et al, 1992). Para observar a limitação da técnica de índice de carga baseado no comprimento da fila de processos, suponha-se um ambiente composto de dois computadores. Em dado instante o primeiro computador apresenta dois processos de baixa ocupação em sua fila e o segundo, um processo que ocupa cerca de 99% da CPU. Nessa técnica de cálculo do índice de carga, o primeiro computador apresenta índice igual a 2 e o segundo igual a 1. O computador com menor índice recebe novos processos iniciados no ambiente (Shivaratri et al, 1992). Neste caso, o segundo computador irá receber o processo iniciado. Contudo, ele não tem recursos disponíveis. Enquanto isso, o primeiro computador encontra-se semiocioso.

Entretanto, deve-se levar em conta que o trabalho realizado em (Ferrari e Zhou, 1987; Theimer & Lantz, 1989; Kunz, 1991; Shivaratri et al, 1992) é típico de estudos iniciais de investigação e pesquisa de índices de carga, e difere demasiadamente das características de recursos físicos disponíveis na atualidade, quase vinte anos depois. O trabalho por eles realizado é uma investigação de fatores de índices de carga tais como a freqüência de atualização das cargas, características de I/O e computacional da carga.

Resumindo, foi demonstrado que: escalonamento fazendo uso de resultados de qualquer índice de carga é sempre melhor quando comparado com sistemas que não efetuam balanceamento; que índices de carga baseados na média geralmente são melhores que índices instantâneos; e que a média do comprimento de fila de CPU é sensivelmente melhor que a utilização de CPU. Entretanto, mais uma vez o que deve ser observado é que esses estudos foram conduzidos, em sua maioria, em ambientes homogêneos.

Quando a plataforma utilizada é heterogênea e multiusuária, a importância do índice de carga é ainda maior. Além das alterações feitas pela própria aplicação, a presença de aplicações de outros usuários, e as diferenças entre os processadores influenciam decisivamente o desempenho esperado para a aplicação.

As diferentes potências computacionais existentes nas plataformas computacionais distribuídas e heterogêneas agregam ao índice de carga a tarefa de normalizar as cargas computacionais em relação às potências em cada elemento de processamento. Essa normalização não é trivial, porque dependendo da aplicação utilizada o impacto da heterogeneidade será maior ou menor.

3.4 Fatores que Influenciam o Desempenho Alcançado com o Algoritmo de Escalonamento

Apesar de se fixar um objetivo para o escalonamento de processos, um mesmo algoritmo de escalonamento pode apresentar desempenhos distintos (até mesmo negativo) em conseqüência de, basicamente, três fatores: a maneira como o próprio algoritmo de escalonamento é construído; a plataforma computacional utilizada; e a classe da aplicação executada, caracterizando a carga de trabalho.

3.4.1 Algoritmo de Escalonamento

O algoritmo de escalonamento é um fator decisivo no desempenho atingido com o escalonamento. Além dos benefícios obtidos com um escalonamento melhor elaborado, deve-se considerar também os custos necessários para a obtenção desse escalonamento.

Algumas vezes, um algoritmo simples possui ganho de desempenho melhor por apresentar pouca sobrecarga, do que quando comparado a um algoritmo mais complexo e que possui sobrecarga elevada (Wang & Morris, 1985; Casavant & Kuhl, 1988; Ferrari & Zhou, 1987; Tanenbaum, 1995; Franklin & Govidan, 1996).

Além da sobrecarga imposta pelo algoritmo de escalonamento, existe também a questão da estabilidade e da eficiência do mesmo. (Shivaratri et al, 1992; Krueger & Shivaratri, 1994; Tanenbaum, 1995; Corradi et al, 1998) consideram um algoritmo de escalonamento instável se este piora a situação de carga de uma plataforma,

realizando indefinidamente execuções inúteis que irão contribuir apenas para agravar ainda mais a demanda pelos recursos.

Uma operação inútil, por exemplo, é a procura de processadores ociosos em uma plataforma totalmente sobrecarregada. Por outro lado, um algoritmo eficiente, além de ser estável, melhora, consideravelmente, o desempenho da plataforma quando comparado à execução das aplicações sem o algoritmo de escalonamento.

3.4.2 Classes de Software

Classes de aplicações distintas devem estar presentes em um escalonamento, pois considerar que somente um tipo de aplicação será executada é uma grande limitação.

Diferenças existentes entre as aplicações determinam classes de software. Essas, por sua vez, influenciam diretamente a escolha do algoritmo de escalonamento e, dessa forma, afetam diretamente o desempenho obtido no escalonamento.

Independentemente da plataforma computacional adotada, um sistema computacional pode ser utilizado para executar diferentes classes de software. Essas classes podem ser agrupadas sob diferentes pontos de vista como, por exemplo:

- (Leland & Ott, 1986) mostraram que processos normalmente considerados dividem-se em: CPU, I/O, ordinário (unbound) e extremamente raros (mix).
- Em (Harchol-Balter & Downey, 1997) e em (Kunz, 1991), as aplicações são classificadas em *CPU-Bound* e *I/O-Bound*, representando aplicações que requerem mais processamento e mais comunicação respectivamente.

As aplicações classificadas como *CPU-Bound* são caracterizadas por utilizarem muito tempo de processamento e apresentarem pouca comunicação. Uma vez que os dispositivos de I/O não interferem significativamente no desempenho do sistema. Para esse tipo de aplicações, acreditam-se que máquinas com maior potência computacional, em relação ao recurso processador, devem ser levadas em consideração por mecanismos ou algoritmos de balanceamento de carga, visando, assim, obter maior desempenho do sistema.

Levando em consideração essas características é que algumas medidas de carga baseiam-se simplesmente no número de processos prontos na fila de CPU, apesar de nem sempre o número de processos estar diretamente, e obrigatoriamente, relacionado à complexidade ou demanda por processamento dos processos (Ferrari & Zhou, 1987).

As aplicações classificadas como *I/O-Bound*, por sua vez, caracterizam-se por realizar muitas chamadas aos dispositivos de entrada e saída, quando comparadas à demanda por processamento.

Caracterizando-se dessa forma, a medida de carga deve levar em consideração os fatores relacionados às chamadas de I/O de maneira a contribuir para uma melhor distribuição das cargas. Ainda assim, não é trivial avaliar a carga do sistema sob esse enfoque, tendo em vista que outros recursos, como memória e processador, têm influência direta na avaliação (Ferrari & Zhou, 1987).

- (Devarakonda & Iyer, 1989) confirmaram e observaram ainda uma classe adicional: *Memory-Bound*.
- Em (Saphir, 1995) as aplicações são classificadas em interativas e batch dependendo do grau de interação com o usuário durante o tempo de execução.

As aplicações são classificadas como interativas por que necessitam de grande interação com os usuários. O tempo de resposta é o fator mais importante, devendo ser o mais rápido possível, evitando causar atrasos aos usuários.

Por outro lado, as aplicações batch referem-se aos arquivos de lote. Ao contrário dos usuários interativos, os usuários de aplicações batch não esperam uma resposta imediata, podendo a estratégia de balanceamento desse grupo de aplicações ser, então, mais flexível do que a estratégia de balanceamento para aplicações interativas.

- (Feitelson et al, 1997) por sua vez classificam as aplicações em rígidas, em evolução, moldáveis e maleáveis, considerando para isso a flexibilidade das aplicações em solicitar processadores; entre outras.

Com o advento da internet, e basicamente pela utilização de ambientes distribuídos e o grande uso de aplicações distribuídas, mais uma classe de

aplicações deve ser adicionada: *Network-Bound*, classe essa que apresenta grande número de comunicação entre processos localizados em processadores distintos.

Desse modo, neste trabalho, as aplicações serão classificadas em:

- *CPU-Bound* - da mesma forma que Harchol-Balter e Kunz (Harchol-Balter & Downey, 1997; Kunz, 1991) essas aplicações são assim classificadas por apresentarem muito processamento e pouca atividade de I/O;
- *Disk-Bound* - aplicações que necessitam de muito acesso a disco, tanto para leitura quanto para escrita;
- *Network-Bound* - aplicações que necessitam de muita comunicação entre processos, o que implicará em grande volume de pacotes que entram e saem para a rede; e
- *Memory-Bound* - aplicações que necessitem de muita quantidade de memória, ou até mesmo muito acesso à memória.

Todas os tipos de aplicações mencionados anteriormente podem ainda ser do tipo interativa ou batch, desde de que obedeçam aos mesmos critérios estabelecidos por Saphir, como apresentado na figura 3.1.

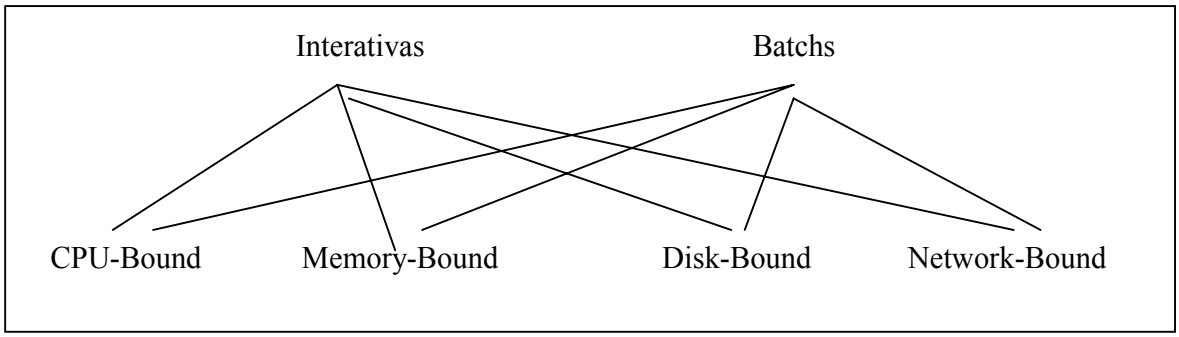

Figura 3.1 - Hierarquia e classificação das aplicações

Um índice de carga para que seja eficiente deve, então, apresentar uma outra característica: ser abrangente. A abrangência deve prever o tipo e a quantidade de trabalho que ocorrerão. Mudadas essas características, por algum motivo qualquer, essas podem afetar o desempenho do escalonamento.

Dimensionar a carga de trabalho não é tão simples quanto parece, uma vez que não existe uma única maneira correta para determinar se um processador está

carregado, ocioso ou com uma carga compatível com sua capacidade. Assim, o uso do termo genérico "carga de um processador" pode não ser representativo o suficiente, uma vez que a demanda pelos recursos disponíveis em um sistema computacional distribuído pode variar de um tipo para outro de aplicação.

3.4.3 Caracterização da Carga de Trabalho

Em um sistema distribuído, usuários executando em diferentes máquinas tipicamente criam tarefas com diferentes demandas por processamento. O resultado é um sistema no qual alguns processadores ficam sobrecarregados enquanto outros estão ociosos ou menos carregados. Para fazer um melhor uso da potência computacional presente no sistema, as tarefas devem ser redistribuídas das máquinas mais sobrecarregas para as máquinas menos carregadas.

A caracterização e a seleção da carga de trabalho são pontos fundamentais para um estudo de avaliação de desempenho. Informação a respeito dessa carga é, então, um dos elementos chaves no processo de balanceamento de cargas, uma vez que tradicionalmente elas têm sido utilizadas para comparar sistemas de computadores.

A eficiência do esquema de balanceamento de cargas pode ser altamente afetada pela maneira através da qual a informação da carga é coletada e de quando essa informação é utilizada. Baseando-se nisso e em diversos resultados analisados (Majundar et al, 1988; Sevcik, 1989; Park & Dowdy, 1989; Ghosal et al, 1991; Parson & Sevcik, 1995; Mello, 2003), tem sido mostrado que o conhecimento prévio das características das tarefas pode aumentar o desempenho de um escalonador de processos.

Uma caracterização de carga de trabalho expressa qualquer carga de trabalho utilizada em estudos de desempenho. Essas caracterizações de carga de trabalho podem ser reais ou sintéticas. Uma carga de trabalho real é aquela observada em um sistema que está sendo utilizado para a execução de operações normais. Essa carga de trabalho não pode ser repetida e, dessa maneira, não é freqüentemente utilizada para efetuar a caracterização da carga de trabalho.

Por outro lado, uma carga de trabalho sintética, possuindo características similares às cargas de trabalho real, pode ser utilizada repetidamente e de maneira controlada.

A razão principal para se utilizar a carga de trabalho sintética de forma mais freqüente é que ela é uma representação ou um modelo de carga de trabalho real. Outras razões secundárias podem ser apresentadas tais como: essa carga de trabalho pode ser facilmente modificada sem afetar a operação do sistema; pode ser facilmente transferida para diferentes sistemas de diferentes tamanhos; pode ser facilmente reproduzida; possui e permite uma maior flexibilidade do que as cargas de trabalhos reais, entre outras (Kunz, 1991).

Uma outra grande vantagem oferecida pelas cargas de trabalho artificiais é a possibilidade de serem escritas especificamente para o experimento a ser executado. Por outro lado, a principal desvantagem dessas cargas artificiais é que elas requerem que todo o ambiente fique dedicado para cada experimento. Algumas técnicas para definição dessas cargas de trabalho são apresentadas em (Ferrari & Zhou, 1987; Devarakonda & Iyer, 1989).

Os recursos de maior relevância analisados na maioria dos casos de geração de carga de trabalho sintética apresentados na literatura são: CPU, memória, disco e a rede (mais precisamente os meios de comunicação) (Mehra, 1993; Dikenelli et al, 1997; Zhou et al, 1993).

Existem na literatura diversas ferramentas, algumas constituindo verdadeiros ambientes, responsáveis pela geração dessas cargas de trabalho sintéticas (Mehra, 1993; Plastino, 1999). Essas ferramentas, que permitem a gravação e repetição da carga de trabalho, se preocupam com as diversas classes de aplicações e com os recursos existentes no ambiente em que elas serão geradas.

Tendo em vista as inúmeras pesquisas e estudos realizados nessa área, observa-se que a carga de trabalho, em si, constitui um ponto crucial no processo de avaliação de desempenho. Obtendo-se um valor inapropriado dessa carga, conclusões errôneas e inaceitáveis podem ser obtidas.

Idealmente, cargas de trabalho sintéticas, objetivando o balanceamento de cargas no sistema, devem ser caracterizadas por um conjunto de métricas de desempenho que satisfaçam aos seguintes critérios:

- baixa sobrecarga, implicando que as medições podem executar freqüentemente atualizações de informações sobre a carga;
- representar a carga em todos os recursos disponíveis; e

- poder ser medida e controlada independentemente uma das outras.

De modo a testar múltiplas alternativas que possam ser oferecidas por condições idênticas de submissão de um sistema, fica caracterizada a necessidade da carga de trabalho ser passível de repetição, permitindo o desenvolvimento de modelos de carga de trabalho que possam ser repetidas quantas vezes forem necessárias.

A caracterização da carga de trabalho necessária para a avaliação de desempenho e, neste trabalho estando voltada mais para as informações que pode oferecer para a definição de índices de carga e de desempenho, é um item que por si só já constitui uma linha de pesquisa e, em função disso, não serão discutidos outros fatores pertencentes à caracterização da carga de trabalho, os quais podem ser encontrados em: (Mehra 1993; Milojicic et al 1993; Kunz 1991; Ferrari& Zhou 1987; Shivaratri et al 1992), entre outros.

(Tannenbaum, 1995) cita que dimensionar a carga de trabalho não é tão simples quanto parece, porque não existe uma única maneira correta para determinar se um processador está carregado, moderado ou ocioso. (Ferrari & Zhou, 1987) citam que devido ao fato da demanda por recursos variar de aplicação para aplicação, o uso do termo genérico "carga de um processador" pode não ser representativo.

Para exemplificar essa situação os autores citam o caso onde um processador pode estar sobrecarregado enquanto o acesso aos discos não está. Nessa situação, para aplicações *CPU-Bound* a carga do processador deve ser considerada alta, enquanto que para processos *Disk-bound* a carga do processador deve ser considerada baixa.

As cargas de trabalho externas à aplicação são chamadas por alguns autores como *background workload* ou *external load* (Mehra, 1993; Russ et al, 1997; Krone et al, 1998).

3.5 Medidas de Desempenho

A preocupação com avaliação de desempenho vai desde o ambiente acadêmico até os ambientes comerciais e industriais. Entretanto, como afirmar que o desempenho de um sistema é satisfatório? A resposta a essa pergunta depende dos indivíduos envolvidos no processo, uma vez que esses é que irão definir os objetivos

que desejam alcançar e a partir do estabelecimento desses objetivos é que se pode avaliar o desempenho obtido.

Segundo (Santana et al, 1997), essas medidas de desempenho podem ser agrupadas em duas categorias: “medidas orientadas a usuários” e “medidas orientadas ao sistema”.

No primeiro caso, os objetivos propostos enfatizam as necessidades de um usuário em especial, normalmente em detrimento às necessidades coletivas. O segundo, por sua vez, dá ênfase às necessidades administrativas, avaliando as necessidades de seus usuários de uma maneira geral e global.

As métricas de desempenho formalizam os objetivos determinados tanto pelo usuário quanto pelo sistema, avaliando se esses objetivos estão ou não sendo atingidos e com qual qualidade isso está sendo feito.

Independentemente de validarem os objetivos do sistema ou do usuário, essas métricas de desempenho existem e são discutidas na literatura. Dentre essas métricas destacam-se: tempo de resposta, *throughput*, percentual de utilização do processador, tempo de execução, entre outros discutidos em (Ferrari & Zhou 1987; Feitelson & Rudolph 1996; Xu & Lau 1997).

Diversos são os métodos utilizados para averiguar o desempenho obtido através do mecanismo de escalonamento, destacando-se: simulação, método analítico, prototipação e *benchmarks*. Entretanto, quando uma avaliação on-line é requerida, a utilização de monitores on-line inteligentes faz-se necessária (Schnor et al 1996; Hofmann et al 1994; Joyce et al 1987; Kunz 1991).

Inúmeros critérios e métricas são utilizados em ambientes acadêmicos com o intuito de avaliar a qualidade do índice de carga para o balanceamento, destacando-se entre eles o índice de desempenho (assim denominado por (Ferrari & Zhou, 1987)). Nesse caso, o índice de desempenho avalia a qualidade e o desempenho proporcionado pelos índices de carga existentes na literatura (Ferrari & Zhou, 1987; Zhou et al, 1993). Entretanto, neste trabalho, índice de desempenho será definido de outra maneira, portanto outras características e finalidades a ele serão impostas como poderá ser observado no capítulo 6.

3.6 Índices de Carga Tradicionais

Uma variedade de índices de carga têm sido utilizada na literatura (Devarakonda & Iyer, 1989; Ferrari & Zhou, 1987; Hac & Johnson, 1990; Svesson, 1990; Kunz, 1991; Mehra, 1993). Uma lista dos possíveis e mais utilizados índices inclui:

- tamanho de fila de CPU: esse índice é obtido em função dos processos dispostos na fila. Ele pode utilizar o tamanho instantâneo da fila de CPU ou uma média do tamanho da fila sobre um determinado tempo. Alguns índices de carga são definidos em função de vários tamanhos de fila, como por exemplo a média de CPU, I/O e tamanho de fila de memória.
- utilização da CPU: algumas pesquisas apontam que a utilização dos elementos de processamento é um bom indicador para a carga, sendo então utilizada como índice de carga.
- tempo de resposta ou tempo de processamento: muitas variações ou combinações de tempo de resposta das tarefas têm sido propostas como índices de carga na literatura. Algumas dessas combinações incluem:
- tempo de resposta normalizado, isto é, uma razão entre o tempo de resposta de uma tarefa em uma máquina sobre o tempo de resposta na mesma máquina quando ela está vazia;
- somatória do tempo de processamento restante das tarefas que estão sendo executadas em uma máquina;
- somatória do tempo de processamento usado por todos os processos ativos sobre o tempo atual; e
- somatória do tempo de processamento total de todos os processos ativos (Ferrari & Zhou, 1987; Devarakonda & Iyer, 1989; Kunz, 1991).

Enquanto um único índice é desejável, tanto para que se possa obter a menor sobrecarga possível, é fácil observar que o comprimento de fila de CPU representa diferentes cargas de CPU cujas capacidades e velocidades são significativamente diferentes.

Embora esses índices tenham sido, e ainda sejam, amplamente estudados e utilizados, eles apresentam alguns problemas. O problema fundamental com a

função tradicional de média de carga é que ela ignora outros recursos senão a CPU. Além disso, apesar da média de carga ser um índice de razoável confiabilidade para as aplicações da classe *CPU-Bound*, torna-se uma métrica questionável para as aplicações que utilizam, primordialmente, recursos como memória, disco e meios de comunicação mais especificamente.

O refinamento de métricas proporcionado pelas estratégias automáticas de obtenção de carga pode ser observado através de sua comparação com as limitações oferecidas pelo projeto e obtenção manual nas estratégias de balanceamento de carga. Esse refinamento pode ser obtido através de estratégias de aprendizado, por exemplo, que permitam um acréscimo gradual do desempenho.

Idealmente, um índice de carga deve permitir a alocação de diferentes tipos de tarefas a diferentes tipos de processadores de acordo com as características de cada uma dessas aplicações e dos processadores em questão. Entretanto, o conhecimento do tempo e da carga de cada tarefa só é claro após o término de sua execução ou em alguns dos casos específicos em tempo de compilação.

Faz-se então necessária uma predição da carga antes de sua execução, uma vez que a sua alocação é assim proferida.

Os diversos índices utilizados tradicionalmente, normalmente, permitem a obtenção de valores para que essa predição possa ser feita apenas em ambientes de domínio conhecido, plataformas específicas e não genéricas, e na sua grande maioria em ambientes homogêneos.

3.7 Índices de Carga Inovadores

Partindo da necessidade de predizer as cargas e as características das aplicações antes que elas sejam executadas, alguns autores têm proposto novos índices de carga que predizem de maneira mais confiável essas características, além de contemplar a heterogeneidade dos ambientes (Schnor et al, 1996; Mehra, 1993).

O método de Mehra propõe a utilização de redes neurais, utilizando um algoritmo *backpropagation* com três camadas (Schnor et al, 1996), para predição das cargas de trabalho e obtenção do melhor índice de carga. Uma vez que esse tipo de método é utilizado para predizer, através da comparação do atraso

apresentado pelo recurso específico, ele pode ser utilizado em ambientes configuracionalmente heterogêneos.

Entretanto, a utilização de redes neurais para a obtenção do índice de carga ainda não apresenta resultados aceitáveis, uma vez que a demora na obtenção desses índices é inaceitável, mesmo sendo seus valores confiáveis.

Segundo estudos efetuados por Schnor (Schnor et al, 1996), o treinamento da rede, de maneira tal que ela possa oferecer informações confiáveis, leva cerca de cinco dias.

Os resultados das métricas calculadas tradicionalmente são mais fáceis de serem obtidos e o desempenho mostra-se satisfatório, enquanto que os calculados pelas redes neurais, apesar de mais confiáveis, mostram-se insatisfatórios, quando levado em consideração o tempo gasto para sua obtenção. Se for imaginado que a informação do índice de carga será utilizada em um balanceamento de carga em um determinado instante, seria tremendamente desagradável e insatisfatório esperar cerca de cinco dias para obtenção do resultado.

Outro índice tido como inovador, por refletir a heterogeneidade do sistema, é o proposto por (Wolffe, Hosseini & Vairavan, 1997). Eles propõem a utilização da Capacidade de Carga como sendo um índice de carga para ambientes heterogêneos em detrimento ao uso dos índices tradicionais. Por capacidade de carga entende-se a utilização efetiva do processador. A capacidade de carga é obtida partindo-se da carga efetiva do processador e da junção velocidade relativa dos processadores. Entretanto essa métrica leva em questão somente um recurso, o processador, deixando de lado os demais recursos.

Apesar de ser tratado como índice heterogêneo e inovador os experimentos foram executados em máquinas arquiteturalmente homogêneas, não sendo assim aplicável a ambientes arquiteturalmente heterogêneos.

Embora todos esses estudos, propondo índices de carga inovadores, apresentem grandes e indiscutíveis avanços na investigação de índices de carga e caracterização de carga de trabalho, eles cobrem somente a heterogeneidade configuracional, deixando de lado a parte arquitetural, que continua sendo tratada de maneira homogênea.

3.8 Estratégia para Obtenção de Índices de Carga

Após as discussões efetuadas sobre escalonamento e índices de carga apresentadas nos capítulos 2 e neste, respectivamente, pode-se observar que o desempenho de um determinado sistema é determinado pela eficiência com que os recursos desse sistema são alocados ou compartilhados. Diversos são os recursos que podem ser considerados em um sistema, indo desde o elemento de processamento até a memória *cache*. Entretanto, apenas quatro recursos têm grande influência no desempenho: CPU, Memória, I/O de disco e de Rede.

Muitos trabalhos assumem que a velocidade de CPU é o fator mais importante quando da preocupação com o desempenho do sistema. Isso pode ser assumido como verdade desde que seja fornecida uma quantidade ilimitada dos outros recursos ou que somente certos tipos específicos de aplicações sejam analisadas (simulações numéricas, entre outras), o que não ocorre na maioria das vezes no mundo real.

Todos os processos consomem uma porção dos recursos do sistema, e esses são por sua vez limitados. Se ainda existir recurso disponível após um processo executar tudo o que ele deseja, então o desempenho do sistema é tão bom quanto deveria ser, caso contrário o processo deverá ficar esperando até que o recurso que ele necessite seja liberado, levando a uma degradação no desempenho do sistema.

Apesar da tarefa de obtenção desses índices de carga para caracterização da utilização dos recursos não ser trivial, a sua obtenção (índices simples) em um sistema real não é impossível. Para exemplificar essa real possibilidade de obtenção, a seção seguinte elenca, a título de exemplo, alguns índices passíveis de obtenção em ambiente UNIX para esses principais recursos que afetam drasticamente o desempenho do sistema.

3.8.1 Índices de CPU

A CPU é, dentre os recursos, o de mais fácil mensuração.

Porcentagem de utilização de CPU - Existe no UNIX uma constante com relação à potência de processamento. Em teoria essa constante é de 100% de ciclos de CPU, entretanto, sobrecargas e outras influências tornam o número real próximo a 95%. Desse modo, um processo que faz uso de 90% da CPU é considerado

inteiramente CPU-Bound e consome com isso a maior parte da potência de processamento disponível.

Existem três tipos de dados de CPU que podem ser conseguidos de maneira razoavelmente fácil: utilização, médias de carga e consumo de CPU por processos. Essas informações resumidas podem ser obtidas através dos comandos **vmstat** ou **sar -u**. Ambos os comandos solicitam dois argumentos: número de segundos a serem monitorados pelo sistema e número de saídas a serem reportadas. O comando **sar -u** reporta as porcentagens de tempo de CPU que estão sendo gastas com código de usuário (%usr), código do sistema (%sys) e ocioso (%idle e %wio). De modo análogo o comando **vmstat** mostra os tempos gastos pela CPU com os processos de usuário, sistema e ociosos como pode ser observado na figura 3.2.

```
hades:~$ vmstat 5 5
procs                  memory      swap        io     system         cpu
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy wa id
 0  0   6500  29688 120392 158816    0    0    13    36  152    65  4  1  0 95
 0  0   6500  29640 120428 158816    0    0     0    47  129    31  0  0  0 100
 0  0   6500  29620 120432 158828    0    0     0     0  209   155  3  1  0 96
 0  0   6500  29488 120456 158832    0    0     0    53  177   107  9  1  0 89
 0  0   6500  28744 120500 158840    0    0     0    56  227   178  2  2  0 96
```

Figura 3.2 - Comando vmstat

Um segundo índice que pode ser facilmente obtido é o **tamanho médio da fila de processos** (média de carga), isto é, a média do número de processos que estão executando. Em geral, essa média inclui processos que estão na fila à espera por I/O de disco e de rede, de modo que esta não é uma medida puramente de uso de CPU. Essa medida pode ser obtida através do comando **uptime** (Figura 3.3).

```
hades:~$ uptime
 15:04:43  up 1 day, 19:54,  1 user,  load average: 0.11, 0.11, 0.06
```

Figura 3.3 - Comando uptime

Três valores são apresentados e que correspondem, respectivamente, às médias de cinco, dez e quinze segundos.

Outras variantes de comandos e aplicativos a partir do qual podem-se obter esses índices são: o comando **ps** que permite observar o quanto de CPU cada processo está consumindo individualmente, e o aplicativo **top**.

A maioria desses comandos e aplicativos obtém essas informações a partir do arquivo **cpuinfo** localizado no diretório **/proc**, de modo que a implementação de funções para obtenção desses valores se torna mais flexível.

Os valores para o índice percentual de utilização têm a vantagem de serem independentes de sistema operacional, enquanto que o tamanho da fila da CPU depende do tipo de sistema operacional e da maneira que se implementa o acesso à CPU. Além disso, o índice percentual de utilização leva em conta apenas a influência das aplicações que acessam a CPU, enquanto que o índice tamanho da fila da CPU sofre influência de quaisquer aplicações que estejam executando no processador, independente de estarem usando a CPU ou não.

O índice percentual de utilização da CPU tem a desvantagem de não representar a carga da CPU em determinadas situações. Por exemplo, em um sistema que esteja executando apenas aplicações CPU-Bound, pode ocorrer que dois elementos de processamento apresentem a mesma utilização da CPU, 100%, mesmo que uma CPU possua dois processos e a outra CPU apenas um processo em execução.

Como o índice tamanho da fila de processos da CPU apenas quantifica a quantidade de processos e não a situação da CPU em termos de sua utilização, máquinas com processos mais “pesados”, ou seja, que consomem mais potência de processamento são tratados da mesma forma que as que possuem processos mais “leves”, com baixa taxa de utilização da CPU. Um índice de carga que consiga refletir essas diferenças presentes nesses tipos de processos constitui a melhor escolha como um índice de carga para processos CPU-Bound.

3.8.2 Índices de Disco

O disco pode se tornar o gargalo de um sistema. O disco é um sistema mecânico, e leva cerca de dez milisegundos atualmente para localizar um bloco, o que muitas vezes faz com que o processo requisitante fique aguardando por ele. Atrasos dessa magnitude levam à degradação de desempenho de todos os outros recursos. Cada acesso a disco causa uma perda de milhões de instruções de CPU.

Em termos de disco, algumas informações que podem ser utilizadas como índices de carga, podem ser obtidas através do comando **iostat** (Figura 3.4). Assim

como o comando **vmstat** ele fornece informações sobre o número de caracteres tanto que entram quanto que saem para e do dispositivo de disco.

hades:~$ iostat -D 5 5

		sd1			sd2			sd3			sd5
rps	wps	util	rps	wps	util	rps	wps	util	rps	wps	util
0	0	1.3	0	0	0.3	0	0	0.5	1	1	4.2
9	8	41.1	1	0	1.8	1	0	2.4	6	8	34.8
11	4	48.4	0	1	2.0	0	0	0.0	3	11	32.6
8	0	15.6	0	0	0.0	0	0	0.0	3	0	9.2
0	0	0.0	0	0	0.0	0	0	0.0	0	0	0.0

Figura 3.4 - Comando iostat

Desse modo, os volumes de disco são representados em leituras e escritas por segundo. O custo de *seek* é o fator mais importante que afeta o desempenho de desse dispositivo.

Discos modernos podem transferir uma grande quantidade de megabytes de dados por segundo, caso sejam lidos a partir de setores contíguos, mas podem apenas executar cerca de 50 a 100 *seeks* por segundo. Dessa maneira, se um disco transfere um setor por *seek*, pode-se facilmente resultar em menos de 2% do pico de vazão do dispositivo.

O número de transferências por segundo obtido a partir do comando **iostat** é a informação mais importante em termos estatísticos que deve ser observada (soma das colunas rps e wps obtidas com esse comando). A utilização desses tempos tem como objetivo a maximização do *throughput* de entrada e saída.

3.8.3 Índices de Memória

Uma vez que a maioria dos sistemas operacionais provê memória virtual, a largura de banda do disco e a memória estão diretamente relacionadas. Em um sistema com uma quantidade limitada de RAM, freqüentemente existe a escrita de páginas para o disco de maneira a renovar páginas da memória virtual. Infelizmente isso significa que fazer uso da memória é tão caro quanto fazer uso do disco.

Existem basicamente dois números que quantificam a quantidade de memória ativa: o tamanho total de memória virtual e a taxa de paginação. O primeiro número informa qual a demanda existente por quantidade de memória, enquanto que o segundo qual porção dessa memória está realmente sendo utilizada.

Para se obter, entretanto, um índice de carga que possa quantificar a memória utilizada ou a quantidade de memória existente, são utilizadas entre outras, informações obtidas a partir do arquivo **/proc/meminfo** (figura 3.5).

```
hades:/proc$ cat meminfo

        total:    used:    free:  shared: buffers:  cached:
Mem:  528039936 499298304 28741632        0 120827904 165904384
Swap: 509956096  6656000 503300096
MemTotal:       515664 kB
MemFree:         28068 kB
MemShared:           0 kB
Buffers:        117996 kB
Cached:         157132 kB
SwapCached:       4884 kB
Active:         149540 kB
Inactive:       226024 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:       515664 kB
LowFree:         28068 kB
SwapTotal:      498004 kB
SwapFree:       491504 kB
```

Figura 3.5 - Informações obtidas do arquivo /proc/meminfo.

A partir desse arquivo pode-se obter a quantidade de memória livre (memfree) facilmente. Outros valores como o somatório das variáveis *memfree*, *Inact_dirty*, *Inact_clean*, *Inact_target* pode ser também obtido, de maneira a representar mais apropriadamente a quantidade de memória efetivamente livre no sistema.

3.8.4 Índices de Rede

A largura de banda da rede assemelha-se à largura de banda do disco em muitos modos devido à latência envolvida. Entretanto, a rede se distingue do disco uma vez que envolve a comunicação como um todo ao invés da computação individual.

Em estudos realizados por (Ishii, 2004), foram consideradas duas diretrizes básicas com objetivo de mensurar a carga de comunicação em um sistema computacional distribuído com propósitos de execução de aplicações paralelas. A primeira diz respeito a quanto de carga de comunicação uma aplicação *Network Bound* gera em um sistema computacional. A outra diz respeito à quantidade de comunicação presente no sistema computacional distribuído, isto é, como se

encontra a carga de trabalho referente à comunicação, na rede de interconexão do sistema computacional antes que a aplicação *Network Bound* seja escalonada.

Latência de comunicação ponto-a-ponto - Essa informação pode ser obtida a partir da confecção e execução de um programa ping-pong, que coleta informações relevantes à carga de trabalho referente à rede de comunicação. O programa avalia o quanto um computador está carregado em termos de acessos à rede de comunicação, calculando o tempo de envio de uma mensagem, considerando que o computador mais carregado em termos de comunicação vai demorar mais para transmitir uma mensagem que um computador menos carregado (esses tempos de transmissão das mensagens em cada computador especificamente podem ser utilizados com índice de carga).

No estudo efetuado por (Ishii, 2004), o programa ping-pong foi implementado em duas versões: uma através do *broadcasting* de uma mensagem, e a partir de então todas as máquinas respondem a essa mensagem. Assim, coleta-se o tempo inicial de envio e o tempo final de recebimento de cada mensagem em diversas execuções e calcula-se a média da diferença desses tempos; outra versão através do conceito ponto-a-ponto, onde os elementos de processamento pertencentes à máquina paralela virtual enviam mensagens à outra máquina que não participa da execução da aplicação e que funciona apenas como um servidor de ping-pong. Como na primeira versão, também se coleta o tempo inicial de envio e o tempo final de recebimento de cada mensagem, executa-se várias vezes o programa e, então, calcula-se a média da diferença desses tempos.

A versão *broadcasting* foi implementada de tal forma a compor apenas um programa em que uma opção de configuração de *socket*[1] (*setsockopt*[2]) habilita o envio de pacotes ICMP *Echo Request*[3] em *broadcasting*. As máquinas devem estar configuradas para aceitar esse tipo de pacote, em estações de trabalho Linux. A *flag*

[1] No UNIX e em outros sistemas operacionais, é um objeto de software que conecta uma aplicação a um protocolo de rede. Por exemplo, um programa pode enviar e receber mensagens TCP/IP por meio de um *socket*, lendo e escrevendo dados do e para o *socket*.

[2] Manipula opções associadas a um *socket*, neste caso, habilita pacotes TCP/IP em *broadcast*.

[3] Um tipo de pacote ICMP usado para *broadcasting.*

/proc/sys/net/ipv4/icmp_echo_ignore_broadcasts deve possuir o valor “0”, o que não ocorre como padrão, máquinas Windows rejeitam esse tipo de pacote.

A implementação ponto-a-ponto além de ser bem mais simples é mais simples de ser gerenciada, e não necessita de configuração específica nas máquinas da rede, como na implementação em *broadcasting*.

Esse tempo de transmissão de uma mensagem (latência de comunicação ponto-a-ponto), pode ser dividido em três partes distintas, cada uma definindo um período de tempo de latência em virtude de suas respectivas sobrecargas (Dillon et al., 1995):

• Latência de transmissão (t_{send}): Antes de ser transmitida, a mensagem precisa ser processada, a fim de acrescentar cabeçalhos, calcular *checksums*, entre outras tarefas. Esta latência é definida pela sobrecarga imposta, por exemplo, pela biblioteca de comunicação;

• Latência da rede (t_{net}): Preparada a mensagem, ela está sujeita a sobrecarga imposta durante a transposição pela rede de comunicação, e aí deve ser incluída a sobrecarga, por exemplo, gerada pelo protocolo TCP/IP;

• Latência de recepção (t_{recv}): Após ser recebida, a mensagem é processada pelo receptor (Por exemplo: verificação de *checksums*, retirada de cabeçalhos, etc.). Essa latência também existe em virtude da sobrecarga da biblioteca de comunicação.

Utilizando-se sintaxe semelhante à definida por (Dillon et al. 1995), pode-se definir o tempo de transferência de uma mensagem entre dois processos A e B, como:

$$T_{A\to B}(n) = T_{\text{sen}\,d}(n) + T_{net}(n) + T_{recv}(n) \qquad \textbf{(Equação 3.4)}$$

onde **n** é o tamanho da mensagem transmitida (em bytes) e t_{send}, t_{net}, e t_{recv} são os tempos relacionados a cada uma das latências na transmissão de uma mensagem.

Um problema para o cálculo desse índice é o fato de não haver sincronização de *clocks* entre os computadores de uma rede Linux. Como os processos transmissor e receptor executam em máquinas diferentes, não há como calcular o tempo de transmissão considerando apenas uma operação de comunicação. Nesse

caso, deve-se calcular o tempo de *round-trip* ($t_{A\leftrightarrow B}$), ou seja, o tempo de transmissão da mensagem do transmissor (por exemplo, processo A) para o receptor (processo B) e o tempo da mesma operação executada em sentido inverso. O tempo obtido é dividido por dois. Colocando em termos mais formais, tem-se:

$$T_{A\leftrightarrow B}(n) = T_{\text{send}_A}(n) + T_{net_{A\to B}}(n) + T_{recv_B}(n) + T_{\text{send}_B}(n) + T_{net_{B\to A}}(n) + T_{recv_A}(n) \quad \textbf{(Equação 3.5)}$$

Considerando $T_{net_{A\to B}}(n) = T_{net_{B\to A}}(n)$, o tempo de latência de comunicação ponto-a-ponto é obtido por:

$$T_{A\to B}(n) = \frac{T_{A\leftrightarrow B}(n)}{2} \quad \textbf{(Equação 3.6)}$$

Quantidade de dados recebidos e transmitidos por segundo - Uma outra maneira de caracterizar os acessos à rede de comunicação é realizado por meio de índices de carga quantitativos que considerem, por exemplo, que processadores que estejam fazendo um maior número de acessos à rede de comunicação estejam mais carregados.

O índice de carga aqui obtido é o número de dados recebidos e transmitidos por segundo de um determinado computador pertencente ao sistema. Esse índice de carga é, assim com outros já mencionados, coletado diretamente no sistema.

De modo análogo aos índices de CPU e memória, o número de acessos à rede pode ser obtido a partir do comando **netstat** (Figura 3.6) que fornece informações para que a rede possa ser supervisionada. Esse comando, assim como os anteriormente apresentados obtêm informações do arquivo **/proc/net**.

```
% netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp        0      0 hades.linsnet.br:http   200-230-88-213.re:36742 TIME_WAIT
tcp        0      0 hades.linsnet.br:3306   200-230-88-6.reve:46322 ESTABLISHED
tcp        0      0 hades.linsnet.br:http   200-230-88-213.re:36816 TIME_WAIT
tcp        0      0 hades.linsnet.br:3306   200-230-88-6.reve:46403 ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node Path
unix  6      [ ]         DGRAM                    400    /dev/log
unix  3      [ ]         STREAM     CONNECTED     372904 /var/run/mysql/mysql.sock
unix  3      [ ]         STREAM     CONNECTED     372903
unix  3      [ ]         STREAM     CONNECTED     610    /var/run/mysql/mysql.sock
unix  3      [ ]         STREAM     CONNECTED     609
unix  2      [ ]         DGRAM                    403
```

Figura 3.6 - Comando netstat

Outra maneira de se obter esses valores é através do programa **top**.

É importante que o número de acessos seja medido em bytes, ou seja, deve-se mensurar o número de bytes transmitidos (ou recebidos) pela rede de comunicação. O número de acessos que se faz a um dispositivo de I/O depende da organização de hardware e do sistema operacional que esteja executando, e não existe relação direta entre o número de acessos e a quantidade de bytes envolvida em cada acesso.

Um possível problema em analisar um índice quantitativo é que processadores mais poderosos tendem a comunicar mais rápido, mesmo que estejam fazendo um maior número de acessos. Ou seja, otimizar o uso do recurso só baseado em quantidade de acessos não necessariamente vai garantir melhor desempenho das aplicações. Uma alternativa para minimizar esse problema é normalizar os índices de carga medidos de acordo com a capacidade computacional do processador. Dessa maneira, o índice considera, além do número de bytes, também a velocidade a qual esses bytes são transferidos.

Uma vez apresentados alguns comandos que possibilitam a captura dos índices de carga em ambiente Unix, a Figura 3.7 apresenta uma estratégia de obtenção de índice de carga para os diferentes recursos existentes e sua explicação é apresentada em seguida.

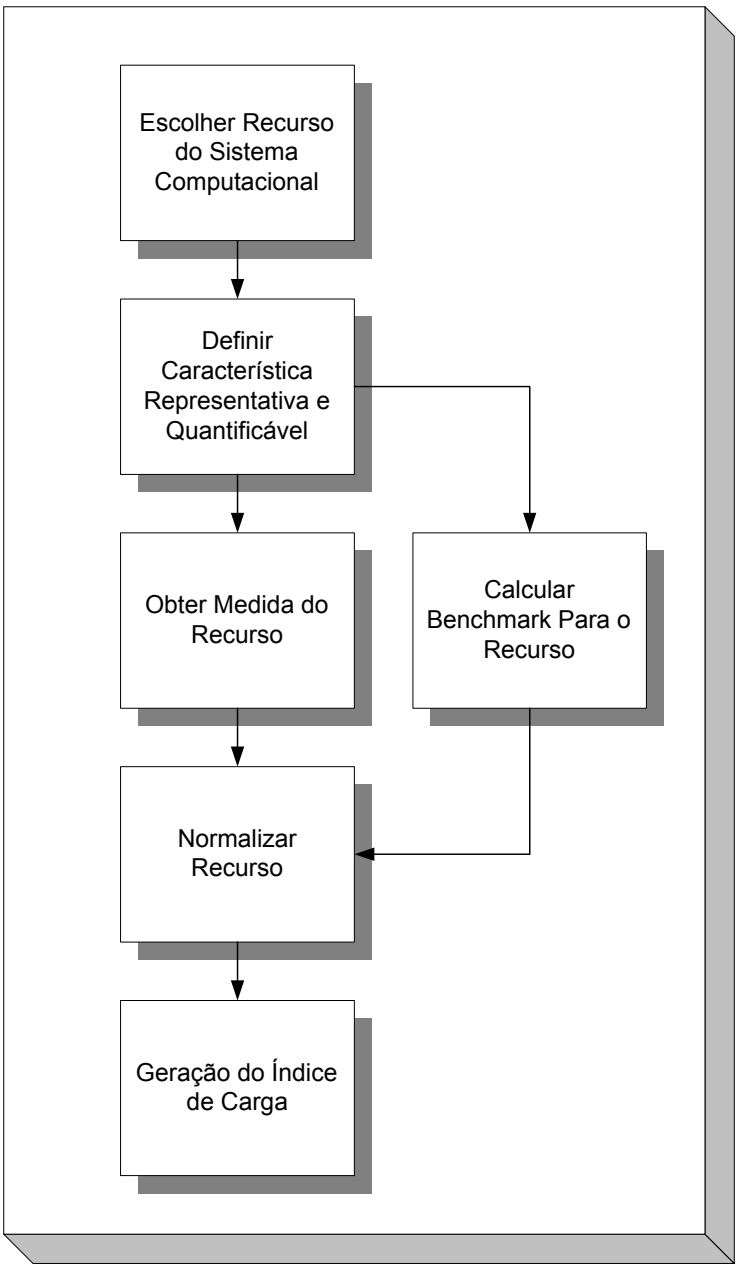

Figura 3.7 - Estratégia para obtenção de Índices de Carga

A obtenção do índice de carga é crucial em um projeto de escalonamento de processos, como já observado anteriormente. Desse modo, alguns passos devem ser seguidos para que esses índices sejam obtidos de maneira apropriada:

1. A melhor forma de se escolher e obter um bom índice é tratar o sistema como um conjunto de recursos:
    a. após observado o tipo de tarefa que será escalonada ou que deverá ser distribuída dentro do sistema um recurso ou diversos recursos devem ser escolhidos como foco do processo de escolha.

2. Características relevantes do recurso devem ser observadas e selecionadas:
    a. as características básicas e que determinam e individualizam um recurso devem ser levantadas e avaliadas;
    b. a observação dessas características e a obtenção dos valores que as representam devem ser consideradas e analisadas nos diferentes sistemas operacionais que compõem o sistema computacional a ser avaliado.
3. A escolha do nível de detalhamento da informação que será obtida deve ser definida para que o nível de confiança do índice seja aceitável, entretanto, deve ser escolhida de modo que não venha causar atrasos ou sobrecargas na obtenção dos valores:
    a. escolha das variáveis que mais representam o recurso devem ser feita de maneira cuidadosa;
    b. tipos de serviços oferecidos por esses recursos e a freqüência com que esses recursos são solicitados podem ser avaliados;
    c. a utilização de arquivos de *trace* ou de históricos para extração das informações relevantes sobre esse recurso pode ser feita quando níveis mais detalhados são exigidos;
    d. média de demanda pelos recursos é utilizada para obtenção mais precisa dos valores, de mesmo modo que a utilização de distribuições de probabilidade em caso de grande variância nos valores obtidos.
4. A representatividade da característica escolhida independente do nível de detalhamento desta:
    a. um índice de carga deve ser representativo do recurso de modo real e deve ser atualizado periodicamente para que as informações obtidas sejam consistentes;
    b. uma vez que a informação se torna obsoleta, essa deixa de caracterizar apropriadamente o recurso, de modo que se faz

necessária a obtenção de um novo fator ou característica para ser considerada como índice de carga;

c. quando o nível de detalhamento do índice de carga é trocado, o cálculo do índice deve ser ajustado para essa mudança.

5. Quantificação dos valores obtidos:

   a. uma vez obtidos os valores a partir das características ou fatores mais relevantes, esses devem poder ser quantificáveis em termos de no mínimo três diferentes estágios: ocioso, moderado e sobrecarregado.

6. Obtido o(s) valor(es) dos recursos a partir dos fatores escolhidos, este(s) valor(es) devem ser normalizados de maneira que as diferenças existente entre os recursos sejam levadas em consideração:

   a. para que a normalização possa ser efetuada faz-se necessária a presença de *benchmarks* que permitam a classificação de mesmos recursos com características distintas;

   b. impacto de fatores externos ao recurso são levados em consideração quando existe a correta escolha do *benchmark* executado.

3.9 Considerações Finais

Apresentou-se aqui uma visão geral dos modelos empregados para obtenção de índices de carga encontrados na literatura.

O índice de carga é uma das questões chave em qualquer projeto de algoritmo de escalonamento de processos, principalmente quando se fala em balanceamento de carga. A sua finalidade é predizer o desempenho de uma tarefa se esta for executada em uma determinada máquina.

Pesquisas apontam para diferenças significativas na eficácia dos índices de carga e que índices simples são particularmente eficientes. (Kunz, 1991) cita que a escolha do índice produz um efeito considerável no desempenho final do

escalonador e que o índice mais eficiente dos mencionados acima é o comprimento de fila da CPU. Não foram verificadas melhorias no desempenho quando o comprimento da fila de CPU foi combinado com outros índices. A escolha do índice de carga deve obedecer dois fatores: eficiência e mínimo *overhead*.

O estudo de índices de carga não é uma tarefa trivial, não sendo encontradas na literatura referências que forneçam uma visão abrangente do assunto, contemplando todas as características e parâmetros necessários para que se possa obter um índice de carga que represente com fidelidade as similaridades, as diferenças e as peculiaridades de máquinas heterogêneas.

Os trabalhos realizados por Ferrari & Zhou em 1987 e por Kunz em 1991, bem como outros trabalhos mais recentes como o de Schnor em 1996 e o de Souza em 2000, indicam que a escolha do índice de carga correto tem influência positiva no desempenho global do algoritmo de escalonamento. Entretanto, não existem estudos completos que considerem a influência de diferentes índices de carga quando da execução de diferentes classes de aplicações.

Com o intuito de investigar essa lacuna, testes foram realizados fazendo uso do PVM (Parallel Virtual Machine) e do ambiente de escalonamento AMIGO (dynAMical flexIble schedulinG envirOnment). Esses testes, apresentados no próximo capítulo, visam a um estudo mais completo da influência das classes de aplicação na escolha do índice de carga e no melhor desempenho do software de escalonamento.