ROMILDO MARTINS DA SILVA BEZERRA

UMA PROPOSTA PARA A GERÊNCIA AUTONÔMICA E ESCALÁVEL DE REDES DE COMPUTADORES

Tese apresentada ao Programa Multiinstitucional de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, Universidade Estadual de Feira de Santana e Universidade Salvador, como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

Orientador: Prof. Dr. Joberto Sérgio Barbosa Martins

Salvador
2012

Ficha catalográfica elaborada pela Biblioteca XXXXXX,
Instituto de Matemática

S111d

Bezerra, Romildo Martins da Silva
XXX
XXX / Romildo Martins da Silva Bezerra. – Salvador, 2012.

93p. : il.

Orientador: Prof. Dr. Joberto Sérgio Barbosa Martins.
Tese (doutorado) – Universidade Federal da Bahia, Instituto de Matemática, Programa Multiinstitucional de Pós-Graduação em Ciência da Computação, 2011.

1. Gerência de Redes. 2. Computação Autonômica. 3. Escalabilidade 4.Qualidade de Serviço. I. Joberto Sérgio Barbosa. II. Universidade Federal da Bahia. Instituto de Matemática. III Título.

CDD 20.ed. 003.83

TERMO DE APROVAÇÃO

ROMILDO MARTINS DA SILVA BEZERRA

UMA PROPOSTA PARA A GERÊNCIA AUTONÔMICA E ESCALÁVEL DE REDES DE COMPUTADORES

Esta tese foi julgada adequada à obtenção do título de Doutor em Ciência da Computação e aprovada em sua forma final pelo Programa Multiinstitucional de Pós-Graduação em Ciência da Computação da UFBA-UEFS-UNIFACS.

Salvador, 05 de setembro de 2012

Orientador: Joberto Sérgio Barbosa Martins, Docteur
Universidade Salvador

Professor Lisandro Zambenedetti Granville, Doutor
Universidade Federal do Rio Grande do Sul

Professor Jose Neuman de Souza, Docteur
Universidade Federal do Ceará

Professor Celso Alberto Saibel Santos, Docteur
Universidade Federal do Espírito Santo

José Augusto Suruagy Monteiro, Ph.D.
Universidade Federal de Pernambuco

Para minha esposa e minha filha

AGRADECIMENTOS

Preciso escrever

Os investimentos em conhecimento geram os melhores dividendos

—BENJAMIN FRANKLIN

RESUMO

A tarefa de gerenciar redes de computadores tem se tornado uma atividade altamente complexa devido a diversidade dos serviços ofertados, à heterogeneidade dos dispositivos e à baixa granularidade da gerência. Neste contexto, os sistemas de gerência autonômicos (*Autonomic Management Systems* – AMS) estão sendo investigados como uma possível abordagem capaz de lidar com a alta complexidade da gerência. Com efeito, espera-se que tais sistemas autonômicos possam, entre outras funcionalidades, ofertar a configuração e otimização dos recursos da rede, mantendo escalável as características de desempenho. Fundamentalmente, o problema de escalabilidade deve ser considerado por soluções atuais de sistemas de gerência autonômicos sob risco de tais soluções não se adequarem em diferentes ambientes gerenciados. Dessa forma, espera-se que as soluções autonômicas sejam encontradas, de preferência, dinamicamente (*on-the-fly*) e, como tal, possam efetivamente substituir a intervenção humana na gestão da rede. Este trabalho objetiva a concepção de uma solução autonômica para que a gerência de redes ofereça, dentre outros aspectos, uma característica de escalabilidade na verificação de novas soluções da rede. Para tal, foi desenvolvido um arcabouço auto-gerenciado para a utilização em diversos cenários escaláveis, que foram alcançados mediante a concepção de uma estratégia de particionamento de rede (NPCE – *Network Partitioning Computing Engine*). A NPCE considera simultaneamente um conjunto de requisitos especificados como parâmetros de QoS e tempo de execução na busca de novas configurações. Os resultados demonstram a viabilidade da solução autonômica através de testes em cenários com diferentes topologias, perfis de tráfego e mediante a aplicação de falhas.

Palavras-chave: Computação Autonômica, Auto-Gerenciamento, Gerência de Redes e Escalabilidade

ABSTRACT

Network services diversity, huge network size and technology heterogeneity are currently computer network characteristics that gradually induce network management to become a highly complex task for administrators. In this context, the Autonomic Management Systems (AMS) are being investigated as possible approaches capable to deal with this inherent complexity. In effect, it is expected that new autonomic systems and solutions could, among other functionalities, both configure and optimize network resources keeping network performance characteristics on a scalable way. This paper addresses, fundamentally, the scalability problem faced by current AMS implementations. In effect, it is expected that autonomic solutions could be computed, preferably, on-the-fly and, as such, can effectively substitute human intervention in network management. We argue that a network partitioning strategy can be applied to manage the scalability problem while considering a set of requirements specified by the network administrator (QoS parameters and execution time). This thesis addresses the set of basic network management requirements considered by the proposal, details the partitioning method and, finally, evaluates the related scalability issues considering a self-managed framework (AMS) in diverse scalable scenarios. Results points to the feasibility of the partitioning method in scenarios with different traffic matrix allocation and, beyond that, under new topologies resulting from failures.

Keywords: Autonomic Computing, Self-Management, Network Management and Scalability

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1

A simplicidade é o último grau de sofisticação – Leonardo da Vinci

INTRODUÇÃO

Nas últimas décadas, a computação tem sido um dos pilares para a evolução da ciência e do progresso da humanidade. A utilização de redes de computadores, serviços baseados na Web, dispositivos portáteis alteram necessariamente o nosso cotidiano e a forma como interagimos na sociedade, basicamente pela mudança na maneira pela qual nos comunicamos. Seja para pessoas ou pequenas e grandes empresas, esta evolução permite uma melhoria significativa de produtividade.

No entanto, enquanto a produtividade dos usuários cresce continuamente, esses avanços criam importantes desafios de gerência para as redes de computadores. e em particular, seus administradores. A sofisticação dos serviços e a maior exigência dos usuários inspira uma nova geração de aplicativos baseados na Web, que compõem diferentes serviços, tornando a gerência da infraestrutura de rede naturalmente mais complexa. Simultaneamente, o crescimento da demanda, da produção de informações e o surgimento de distintas tecnologias tornam a gerência dos ambientes computacionais na área de redes de computadores com baixa manutenibilidade.

A complexidade da gerência de ambientes computacionais associada à necessidade crescente de disponibilidade e heterogeneidade dos serviços impactam diretamente na eficiência dos administradores. Este alto grau de exigência pode tornar a intervenção humana um ponto suscetível a falhas (Jrad et al., 2006), afetando o custo operacional na medida em que aumenta o tempo de busca por soluções. Além disso, o sucesso da gerência destes ambientes complexos está diretamente ligada à *expertise* dos administradores.

Com o objetivo de abordar a complexidade da gerência de ambientes computacionais, maximizando a produtividade dos administradores surge o alternativa de computação autonômica (Horn, 2001). O manifesto publicado em 2001 pela IBM, indica que o crescimento da complexidade dos ambientes computacionais consiste no principal problema a ser atacado pela computação autonômica.

A computação autonômica (*Autonomic Computing*) (Horn, 2001) consiste em um sistema que possui, no limite, a habilidade de se auto governar de acordo com as regras de negó-

cios e objetivos definidos pelos administradores (Kephart; Chess, 2003). A essência dos sistemas autonômicos é a autogerência (*Self-Management*) (Kephart; Chess, 2003), que possui quatro áreas básicas que devem ser atendidas: auto-configuração (*Self-Configuration*), auto-otimização (*Self-Optimization*), auto-cura (*Self-Healing*) e auto-proteção (*Self-Protection*). Este paradigma inovador é considerado uma evolução natural dos sistemas de gerência, em resposta à crescente complexidade e heterogeneidade dos ambientes atuais (Ganek; Corbi, 2003).

1.1 GERÊNCIA AUTONÔMICA DE REDES

O desenvolvimento da gerência autonômica da infraestrutura de redes de computadores é uma área de pesquisa atrativa e também de interesse industrial (Dobson et al., 2006). Analogamente ao sistema nervoso humano, que regula de forma inteligente e inconsciente as funções homeostáticas, a gerência autonômica de redes visa simplificar a administração da complexa infraestrutura de comunicação, reduzindo, por exemplo, a necessidade de intervenção manual na configuração dos dispositivos. A solução dita autonômica requer uma abordagem que pode envolver uma série de áreas existentes tais como: projeto de protocolos, gerência de redes, inteligência artificial, teoria de controle, teoria dos jogos e sistemas sensíveis ao contexto (*context-aware systems*) para garantir requisitos como a manutenção da Qualidade de Serviço (QoS) (*Performability*) e sobrevivência (*Survivability*) da infraestrutura.

Assim é possível afirmar que a caraterística que distingue a computação autonômica de outras formas de gerência é a fusão de técnicas a partir destas disciplinas em busca da possível simplificação na atividade de gerência de infraestruturas complexas, requerendo uma difícil curva de aprendizagem para o desenvolvimento de qualquer trabalho relevante na área de computação autonômica (Dobson et al., 2006).

Ao longo dos anos, as redes de computadores foram projetadas com um conjunto de dispositivos coordenados de forma centralizada com a visão global de um administrador de rede. Tal administrador é responsável em manter o funcionamento desta infraestrutura complexa de dispositivos com capacidades heterogêneas e aplicações com requisitos distintos de acordo com as regras de negócios especificadas. Dessa forma, uma rede de computadores pode ser caracterizada como um ambiente com grande nível de complexidade e heterogeneidade (Yahia; Bertin; Crespi, 2006), dinâmico, pouco confiável e de larga escala, tornando a gerência um processo difícil, senão impossível (Jennings et al., 2007).

Considerando tais características, os princípios da computação autonômica podem ser utilizados em redes de computadores com o objetivo de simplificar o processo de gerência, reduzindo a intervenção humana, ponto suscetível de falhas em sistemas complexos (Jrad et al., 2006).

1.2 MOTIVAÇÃO

O autogerenciamento de redes (Jennings et al., 2007) procura melhorar a capacidade da rede em ofertar seus serviços para lidar com mudanças imprevisíveis, incluindo mudanças físicas e lógicas de acordo com as regras de negócio especificadas. Para tanto, são necessárias soluções visando superar os desafios intrínsecos à computação autonômica (Kephart, 2005).

A utilização dos princípios de computação autonômica no contexto da gerência de redes potencializa uma maior produtividade para a área de gerência e, assim sendo, é de grande interesse da comunidade. A utilização de uma solução de gerência com características autonômicas permite uma melhoria da capacidade da rede e de seus serviços em lidar com mudanças previstas ou imprevistas, incluindo mudanças na topologia, alteração da carga de tráfego e modificações das características físicas e lógicas da rede, dentre outras.

Neste mesmo contexto é importante ressaltar que a factibilidade das respostas às mudanças indicadas depende diretamente do tempo de resposta na busca de soluções – novas configurações – capazes de atender aos requisitos especificados para a rede sendo gerenciada. Assim sendo, existe também grande interesse da comunidade não só em garantir alguma autonomia na solução de gerência aplicada à rede como também de ter uma solução sistêmica que possa ser derivada com tempo de busca dentro de parâmetros considerados razoáveis e exeqüíveis para a gerência.

Em termos práticos isto quer dizer que se busca uma solução menos dependente do ser humano e que pode ser encontrada numa estratégia visando a redução dos tempos de resposta tentando desta forma viabilizar uma solução o mais próxima possível das soluções dinâmicas.

Outro aspecto de grande interesse no contexto da gerência autonômica é a questão da escalabilidade da solução proposta. Nesse sentido, as novas soluções propostas de gerência com características autonômicas devem permitir uma grande quantidade de dispositivos com perfis variados visto que esta é a realidade da própria evolução das redes nos dias atuais.

Em suma, a motivação básica da solução apresentada nesta tese consiste em buscar uma solução escalável e com tempo de resposta factível para a gerência de redes, com um certo nível de autonomia para ambientes de redes que podem ser complexos em termos da quantidade de dispositivos e perfis de utilização.

1.3 OBJETIVOS E ORGANIZAÇÃO DA TESE

O principal objetivo deste tese é propor um conjunto de estratégias de gerência autonômica materializadas através de um arcabouço com características autonômicas para suporte a gerência de redes de computadores. Este arcabouço deve ser estar apto a prover soluções em tempo factível em ambientes com diferentes quantidades de dispositivos gerenciados. Assim, entendemos que o mesmo deve fornecer uma gerência escalável.

O conceito de escalabilidade não está associado a inserção de diversos dispositivos de forma dinâmica no ambiente, mas sim à capacidade do ambiente operar em redes com distintas quan-

tidades de dispositivos e/ou volume de tráfego. Caso novos dispositivos sejam inseridos na rede gerenciada, o arcabouço deve receber a nova topologia e refazer dinamicamente suas configurações. Apesar de ser possível ele se adaptar, este não será o cenário tratado no escopo desta tese.

A provisão de soluções em tempo de resposta aceitável, não deve desconsiderar a maximização da qualidade dos serviços oferecidos pela rede. Certamente, o administrador espera que o arcabouço autonômico para gerência forneça soluções que conciliem o tempo de resposta e a manutenção da QoS. Esta dualidade (*tradeoff*) não é, dentro dos limites de nosso conhecimento, devidamente tratada dentro do contexto da autonomia aplicada a gerência de redes. Além disso, a utilização do histórico do ambiente gerenciado deve ser considerada pelo sistema autonômico e, comotal, faz-se necessária uma especificação de um motor de busca e alguma estratégia para redução desta complexidade.

Dessa forma para permitir o funcionamento adequado da gerência autonômica de rede, de forma autonômica e escalável, é necessário considerar alguns pontos que colocamos como objetivos específicos relacionados ao objetivo mais geral indicado, como:

i) Manutenção da Qualidade de Serviço – *Performability*[1] – O sistema deve ser capaz de oferecer soluções – sempre que possível – que mantenham a qualidade dos serviços ofertados mesmo com a imprevisibilidade do tráfego ou das falhas (Meyer, 1992).

ii) Sobrevivência – *Survavibility*[2] – O sistema de gerência deve prover uma melhor robustez em casos de falhas inesperadas, independentemente de sua localização e duração. Em geral, as falhas abordadas nesta tese serão do tipo *halt failure* (falhas por parada) (Avizienis et al., 2004).

iii) Adaptabilidade – *Adaptability* – O sistema de gerência deve ser aplicável ambientes com distintos topologias, padrões ou volumes de tráfego e não vinculado apenas a uma tecnologia. Não serão impostas restrições quanto ao ambiente utilizado no que diz respeito a sua estrutura e utilização (Jennings et al., 2007).

Esta tese está organizada como segue. O Capítulo 2 apresenta os conceitos e definições relacionadas à computação autonômica, que são aplicáveis ao contexto de redes de computadores. O Capítulo 3 apresenta o arcabouço desenvolvido juntamente com a estratégia de particionamento e o motor de busca de soluções. A validação da escalabilidade e dinamicidade do arcabouço é realizada no Capítulo 4 é materializada através de simulações e de verificação do atendimento dos requisitos especificados. Por fim, o Capítulo 5 encerra esta tese apresentando as conclusões do trabalho e os desdobramentos futuros.

[1] Corresponde a propriedade de um sistema fornecer o desempenho exigido pela especificação do serviço, descrito pela QoS (Sterbenz et al., 2010)

[2] Sobrevivência é a capacidade de um sistema em cumprir, com tempo hábil, seus objetivos mediante a presença de ameaças (Sterbenz et al., 2010)

Capítulo 2

O avanço da civilização aumenta a quantidade de operações importantes que devemos realizar, sem nunca pensar nelas – A. N. Whitehead

GERÊNCIA AUTONÔMICA DE REDES

Neste capítulo serão descritos os conceitos de computação autonômica necessários para o entendimento desta tese e sua aplicação na área da gerência de redes de computadores.

2.1 A EVOLUÇÃO DOS SISTEMAS DE GERÊNCIA

Nas últimas décadas os avanços em áreas como redes de computadores, sistemas distribuídos, *web services*, mobilidade, dentre outros, juntamente com a necessidade de desempenho e produtividade necessitam de uma gerência mais eficaz destes ambientes complexos. Neste contexto, surge o conceito de computação autonômica definido como um processo evolucionário dos sistemas de gerência, motivado do avanço da tecnologia. Na verdade a computação autonômica se apresenta como sucessora natural dos outros quatro níveis de gerência: básico (*basic*), gerenciado (*managed*), preditivo (*predictive*) e adaptativo (*adaptative*) conforme apresentado na Figura 2.1.

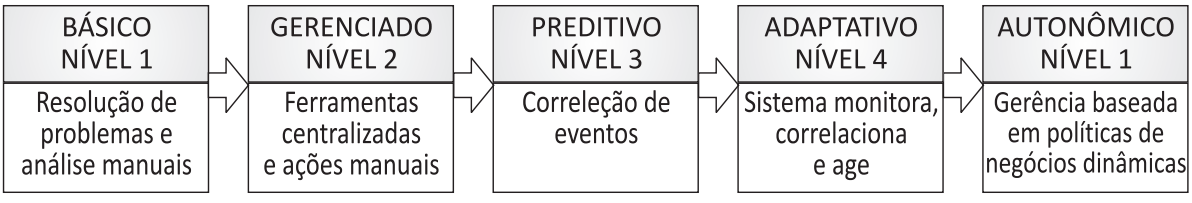


Figura 2.1. Os níveis de evolução da gerência.

O nível básico é ponto de partida dos ambientes gerenciados. Cada elemento da infraestrutura é administrado de forma independente por profissionais de tecnologia da informação (TI), que irão configurá-lo, monitorá-lo e, em caso de falhas, substituí-lo. Neste caso, existem múltiplas fontes gerando informações de gerência, tornando o processo oneroso e necessitando de uma equipe muito qualificada.

No nível gerenciado, sistemas de gerência centralizados coletam informações distintas agregando em menos terminais de gerência, reduzindo o tempo que o administrador leva para de coletar e consolidar as informações do ambiente. Isso aumenta a produtividade em relação ao nível básico. Entretanto no nível gerenciado o administrador pode não ter uma visão global do estado do ambiente gerenciado, pois o foco é a gestão individual dos dispositivos.

Nesta sequência, o nível preditivo possibilita a correlação entre as informações dos elementos gerenciados, podendo não só reconhecer padrões como também possíveis configurações, fornecendo conselhos sobre as possíveis ações que o administrador deve tomar e, consequentemente, permitindo a redução da *expertise* e dependência dos administradores. Esses sistemas tem uma atuação semelhante a um sistema de apoio à decisão.

No nível adaptativo os sistemas podem automaticamente efetuar ações corretivas, em resposta a falhas, baseados na informação oferecida pelos dispositivos e pelas métricas especificadas, com o objetivo de fornecer agilidade e resiliência com a mínima interação humana. Neste ponto, Acordos de Nível de Serviço (*Service Level Agreements*[1]) (SLA) podem ser utilizados como estratégia para determinar dos objetivos do sistema (Ganek; Corbi, 2003).

Por fim, os sistemas autonômicos, que se apresentam como último nível de evoluação, que são capazes de se auto-configurar, em resposta a mudanças ou falhas, de acordo com políticas de negócio (*business policies*) e objetivos, especificadas pelos administradores desses sistemas. Na próxima seção a computação autonômica será apresentada com maiores detalhes.

2.2 COMPUTAÇÃO AUTONÔMICA – UMA BREVE VISÃO

Computação autonômica (*Autonomic Computing*) representa uma evolução inevitável da gerência da infraestrutura de tecnologia da informação (Horn, 2001). Esta evolução se fez necessária pois a complexidade dos ambientes computacionais aumentou, devido a maior sofisticação dos serviços oferecidos, à exigência de qualidade e produtividade., ao crescente volume de dados, a heterogeneidade dos dispositivos, tecnologias e plataformas. Tais características tem aumentado a dificuldade da gerência de infraestrutura, tornando as atividades dos administradores mais onerosa. A complexidade é apresentada como o desafio mais importante a ser tratado por esses sistemas (Ganek; Corbi, 2003).

Em sistemas complexos, a intervenção humana pode ser um ponto de inserção de falhas na gerência (Jrad et al., 2006). Sob este aspecto um sistema autonômico pode ser definido como um sistema livre de administradores para gerência de rotinas administrativas e tarefas operacionais (Ganek; Corbi, 2003).

Contudo, é importante ressaltar que a computação autonômica não foca na eliminação da

[1]Neste contexto, um Acordo de Nível de Serviço – *Service Level Agreement*, SLA – é um acordo entre um cliente e um prestador de um serviço que especifica os níveis de disponibilidade, desempenho, gerenciamento de falhas ou outros atributos do serviço, geralmente estabelecido através de negociação. Em geral, um SLA identifica e define as necessidades do cliente, simplificando as exigências complexas e ajudando a eliminar as expectativas irreais.

intervenção humana (Parashar, 2007), mas sim em uma participação de alto nível com o objetivo de estabelecer metas e regras de negócios a serem seguidas por tais sistemas. Desta forma é possível definir computação autonômica como um sistema que possui a capacidade de gerenciar a si mesmo de acordo com os objetivos estabelecidos pelo administrador (Kephart; Chess, 2003).

De fato, a essência de sistemas autonômicos é o auto-gerenciamento (*Self-Management*) (Kephart; Chess, 2003), que tem como objetivo tornar o ambiente gerenciado capaz de perceber, analisar as suas condições atuais e ter a habilidade de reconfigurar seus componentes e dispositivos, de forma proativa, de acordo com os objetivos definidos pelo administrador.

2.2.1 Requisitos para o *Self-Management*

Para que o *Self-Management* possa ser alcançado, a quatro requisitos fundamentais devem ser atendidos (Kephart; Chess, 2003):

1. **Auto-Configuração (*Self-Configuration*)**
   Corresponde a capacidade do sistema de se adaptar à condições do ambiente, previsíveis ou não, ajustando sua configuração dinamicamente (*on-the-fly*). A auto-configuração não corresponde a capacidade de um sistema de configurar cada dispositivo individualmente de forma automática, mas sim de prover a capacidade de ajustar a configuração dos dispositivos dinamicamente (*on-the-fly*) visando o bem estar do ambiente como um todo.

2. **Auto-Otimização (*Self-Optimization*)**
   Sistemas autonômicos deverão ser capazes de buscar o aperfeiçoamento de suas operações, identificando novas oportunidades de fazer o mesmo as mesmas operações com melhor desempenho ou menor custo utilizando os mesmos recursos. Para isso, eles deverão executar uma busca pró-ativa sendo capazes de identificar, verificar e efetuar mudanças de configuração, que sejam capazes de maximizar a utilização dos recursos sem a intervenção humana.

3. **Auto-Cura (*Self-Healing*)**
   A auto-cura pode prevenir e recuperar uma falha, buscando, diagnosticando e corrigindo pontos que possam causar paradas nos serviços oferecidos. Para isso, o sistema deverá ser capaz de isolar um dispositivo ou componente defeituoso de forma a minimizar o impacto nos serviços, maximizando continuamente a disponibilidade e confiabilidade do ambiente gerenciado.

4. **Auto-Proteção (*Self-Protection*)**
   Corresponde a capacidade de detectar, identificar e defender-se contra ataques e/ou situações não desejáveis. Para que um sistema autonômico seja capaz de se defender é necessário antecipar problemas baseando-se em correção de dados e/ou estudo dos seus estados anteriores. Geralmente, este requisito está vinculado à acesso não autorizado,

vírus, *denial-of-service* ou falhas em geral. Entretanto, auto-proteção também pode ser visualizada como à capacidade de reconhecer e lidar com condições de sobrecarga que possam comprometer a integridade do sistema.

Nota-se a correlação entre os requisitos descritos, uma vez que, por exemplo, não é possível fazer uma auto-otimização ou auto-cura, sem que exista uma auto-configuração do ambiente. Dessa forma, tais requisitos devem permitir uma gerência autonômica colaborativa do ambiente, capaz de melhorar o desempenho do sistema.

Outro ponto importante é a detecção de erro, que consiste em uma peça importante para se desenvolver um sistema de auto-cura eficaz, que esteja diretamente ligado ao monitoramento contínuo do sistema.

É importante ressaltar que existem diversas outras propriedades auto-* (*self-* properties*), que podem estar associadas a escopos específicos de aplicação como, por exemplo, a Auto-estabilização.

A Auto-Estabilização (*Self-Stabilization*) tem emergido como um paradigma promissor para o projeto, controle e manutenção de sistemas tolerantes a falhas distribuídos, pois permite que os sistemas para recuperar automaticamente a ocorrência de falhas. Sua ideia essencial consiste em manter o comportamento desejado de um sistema, independentemente do estado em que o sistema se encontra, mesmo que as falhas o coloquem em um estado arbitrário (não esperado). Neste caso, o sistema pode, eventualmente (se houver recursos disponíveis), retomar o seu comportamento desejado.

Este conceito foi introduzido por Dijkstra, que definiu auto-estabilização com uma característica dos sistemas que chegam a um estado legítimo[2], independentemente do seu estado inicial em um número finito de passos (Dijkstra, 1974) (Schneider, 1993).

2.2.2 Automático *versus* Autonômico

Alguns trabalhos correlacionam um sistema autonômico computacional ao sistema nervoso humano (Strassner; Agoulmine; Lehtihet, 2006) (Murch, 2004) (Parashar; Hariri, 2005). Tal analogia ilustra a importância das funções cognitivas e físicas (ou involuntárias), dentro do escopo computacional para um sistema autonômico. A correlação entre tais sistemas levanta uma discussão pertinente referente à capacidade do sistema de escolher um conjunto de estados pré-definidos ou a possibilidade de propor uma nova solução. Esta análise é de suma importância, uma vez que se espera do sistema autonômico a possibilidade de propor novas soluções de acordo com o estado atual do ambiente gerenciado, mesmo que não existam pré-configurações para tal estado.

Assim, é importante diferenciar um sistema automático de um sistema autonômico. Um sistema automático é capaz de reagir a mudanças de contexto dentro de um conjunto de estados pré-definidos. Em geral, os sistemas automáticos estão vinculados ao conceito de automação. Automatizar um sistema não implica em torná-lo autonômico.

[2]Um estado é considerado legítimo quando sua operação atende aos requisitos esperados.

Um sistema autonômico deve se autoconfigurar mesmo em situações não previsíveis de forma a tentar manter o desempenho, mesmo com falhas, não se contentando com o *status quo* (Ganek; Corbi, 2003). Não são considerados sistemas que ao detectar um estado não esperado, seja por mudanças do ambiente ou falhas, retorna sempre a um estado inicial, pois os sistemas automáticos já fazem isso.

Em (Algomine, 2011) é apresentada uma diferenciação entre automático, autônomo e autonômico com base em outros textos da literatura. A definição de automático (oriundo de *automatically*) está associada a sistemas ou modelos que não possuem qualquer conhecimento além do que está pré-definido, sem qualquer possibilidade de extensão. Na teoria, sistemas automáticos sempre exibem o mesmo comportamento para uma mesma entrada e este comportamento é denominado de função de transferência (*transfer function*). Apesar da palavra automático vir da palavra grega *automatous* – com significado de vontade própria – esta palavra ao longo dos anos associou-se a termos mecânicos, que são pré-estabelecidos e não possuem vontade própria (Feeney; Frisby, 2006). Um sistema automático pode utilizar, por exemplo, o paradigma de raciocínio baseado em regras (*RBR – Rule Based Reasoning*).

Já os sistemas autônomos (oriundo de *autonomously*) são sistemas que exibem um elevado grau de auto-governança (*Self-Governance*) pois este sistema toma a sua decisão com total independência, sem se referir a quaisquer entidades externas para alcançar seus próprios objetivos. Um comportamento autônomo é a liberdade definitiva (Algomine, 2011). Em tais, sistemas as decisões não são pré-estabelecidas, ou seja, o sistema não necessariamente retorna a mesma solução para o mesmo diagnóstico.

Uma ação autonômica é uma tarefa que é realizada por força do hábito ou sem qualquer pensamento consciente. O corpo humano é capaz de realizar uma série de ações reflexas ou involuntárias, enquanto um sistema automático é projetado para executar algumas ações específicas em consequência de alguns fatos ocorridos ou problemas conhecidos. Assim a palavra autonômico, sugere a ideia de auto-governança dentro de uma entidade baseada em políticas ou princípios de funcionamento, como requisitos especificados.

Dentro deste contexto, em (Strassner, 2006) é apresentado conceito de *Self-Governance*, em oposição ao *Self-Management*. Segundo (Strassner, 2006) *Self-Management* é um sistema capaz de assumir valores pré-definidos, através de políticas que serão aplicadas ao sistema de forma prover mudanças esperadas. Já com *Self-Governance* o sistema é pré-carregado de políticas, mas tais políticas podem ser alteradas ou criadas pelo sistema de acordo com as regras de negócio, mudanças do ambiente, demanda de usuários ou qualquer alteração que necessite de um ajuste dinâmico na configuração. Contudo em (Kephart; Chess, 2003) (Jennings et al., 2007) (Dobson et al., 2006) não existem restrições para a não criação de novas políticas para o *Self-Management*. Meste trabalho, assumimos que *Self-Management* é um sistema capaz de criar novos estados visando obter melhor desempenho em resposta a alterações no ambiente gerenciado.

2.2.3 Políticas e Computação Autonômica

Uma política é uma especificação de regras, definida através de um par condição/ação, onde uma ação é executada quando determinada condição é satisfeita (Lupu; Sloman, 1999). Este mapeamento bem definido de condição/ação permite a especificação de um conjunto de estados do ambiente gerenciado e as possíveis soluções decorrentes das ações, para a determinação de um novo cenário mais amigável. Em outras palavras, uma política é uma diretiva que é especificada pelo administrador com certos aspectos desejáveis ou necessários do ambiente, resultando na interação entre aplicações, usuários e recursos (Verma, 2000). Este conjunto de regras é utilizado para gerenciar as mudanças e permitir a manutenção dos estados de um conjunto de objetos gerenciados (Strassner, 2003).

A utilização de políticas é uma realidade nos sistemas de gerência de rede, denominados PBM – *Policy-Based Management*[3] (Davy et al., 2006). Um PBM pode ser visto como um executor de decisões capaz de gerenciar dispositivos, simplificando o processo tradicional de gerência, através da redução da complexidade administrativa e da reconfiguração do ambiente a cada nova mudança de condição, permitindo assim a escolha dinâmica de uma solução. Além de agregar flexibilidade e escalabilidade (Davy et al., 2006), as regras correspondem a representação do conhecimento do ambiente gerenciado da rede através de um mapeamento estático e bem especificado. Estático em relação a pré-definição do conjunto de possíveis estados, e bem especificado no sentido de ser capaz de resolver conflitos e encontrar, sempre que possível, uma solução dentre o conjunto de estados. É importante ressaltar que em algumas situações pode não ser possível encontrar uma solução como por exemplo quando o ambiente gerenciado possuir um alto índice de dispositivos apresentando falhas.

De fato, correlacionar esta inteligência estática ao contexto de autonomia é um desafio. É necessário um modelo capaz de prover uma solução baseada em uma análise dos estados do ambiente gerenciado, através da escolha ou da criação de uma regra para tal. Esta definição autonômica de regras estende o PBM *Policy-Based Management* tradicional, uma vez que não houve intervenções humana na base de políticas. Além disso, políticas tornam-se essenciais na visão da computação autonômica, pois consiste na forma com a qual os humanos irão expressar seus objetivos para os sistemas autonômicos (Kephart, 2005).

Com o uso de políticas na representação dos objetivos, é necessário transformá-las em outras políticas para serem utilizadas pelos sistemas autonômicos (Kephart, 2005), pois políticas podem ajudar a traduzir os SLAs ou regras de negócio em ações visando a configuração e controle dos dispositivos. Dentro deste contexto, alguns trabalhos utilizam políticas em sistemas autonômicos considerando as redes de computadores citeDavy2006, Liu2008, Karmouch2005, Jennings2007, Strassner2011. Entretanto, a metodologia de como e quando usar as políticas, ferramentas de suporte para o refinamento e análise de políticas, bem como ambientes de simu-

[3]Policy-Based Management (PBM) é um paradigma de gerenciamento que separa que separa as normas que regem o comportamento de um sistema de sua funcionalidade (Boutaba; Aib, 2007).

lação para avaliar o uso de políticas e soluções de gerência para grandes domínios são aspectos importantes para pesquisa e necessários para a implantação da política na prática (Boutaba; Aib, 2007).

2.2.4 Aspectos Arquiteturais

Sistemas autonômicos são vistos como uma coleção de elementos autonômicos que possuem recursos capazes de prestar serviços aos administradores ou outros sistemas autonômicos (Kephart; Chess, 2003). A arquitetura de um sistema autonômico pode ser visualizada como uma composição de elementos autonômicos independentes, no que diz respeito a auto-gerenciamento (*Self-Management*), mas com perfil colaborativo ou inteligência social capaz de permitir um comportamento coletivo de acordo com os objetivos especificados. Em outras palavras, as decisões podem ser tomadas por cada elemento, mas sempre almejando o melhor estado para o ambiente como um todo.

De forma geral, um sistema autonômico é composto um elemento ou mais elementos gerenciados por um único gerente autonômico que os controla. O elemento gerenciado é equivalente ao que é encontrado em sistemas não autonômicos comuns, embora possa ser adaptado para permitir o auto-gerenciamento. Já o gerenciador autonômico (*manager*) é responsável pelo monitoramento, análise, planejamento e execução, criando uma sequência de ações necessárias para este ciclo de gerência autonômica (Figura 2.2).

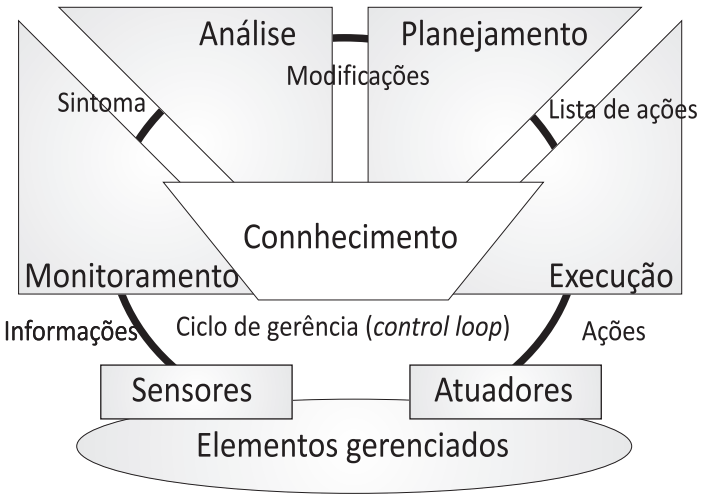


Figura 2.2. Ciclo de gerência (*control loop*) – Adaptado de (Parashar, 2007).

A fase de monitoramento tem como objetivo a coleta de dados de elementos gerenciados, que por sua vez serão filtrados e agregados de forma que possam ser utilizados na fase de análise. Ao receber estes dados, a fase de análise deve ser capaz de correlacioná-los visando extrair informações que são utilizadas para o conhecimento do estado atual do ambiente, identificando as suas mudanças e podendo prever situações futuras. Já o planejamento verifica ou cria as possíveis ações necessárias visando atingir os objetivos especificados. Tais ações podem afetar

apenas um ou vários elementos gerenciados. E por fim a fase de execução controla o conjunto de ações planejadas na fase anterior. É importante que tais ações sejam executadas com atomicidade[4], visando a corretude na configuração dos elementos gerenciados (Bezerra; Martins, 2009b). A atomicidade impede a ocorrência de uma atualização incompleta dos dispositivos, resultando em um estado não desejado.

Este modelo ainda prevê um ciclo contínuo de gerência autonômica, responsável em coletar as informações e buscar ações para controle dos elementos gerenciados, denominado ciclo de controle (*control loop*). Esse ciclo contínuo de gerência autonômica pode ser utilizado visando a verificação de acertos das decisões tomadas no ciclo anterior e, se houver gestão de conhecimento, aprender com erros e acertos, assim como nós humanos. Alguns trabalhos preveem mais de um ciclo de controle visando trabalhar diferentes atividades como a correção e otimização (Jennings et al., 2007). Maiores detalhes serão vistos no Capítulo 3.

Ao monitorar o elemento e seu ambiente externo, e executar ações baseadas neste monitoramento, um gerente autonômico está facilitando o trabalho humano na gerência de cada elemento. Cada elemento autonômico é responsável pelo auto-gerenciamento de seu estado interno e pela interação com outros elementos autonômicos. A relação entre eles pode ser realizada em muitos níveis com o objetivo de fixar estados e relacionamentos permitindo maior dinamismo e flexibilidade (Kephart; Chess, 2003).

Outros trabalhos apresentam conceitos similares com nomenclaturas distintas (Jennings et al., 2007) (Dobson et al., 2006). Em (Dobson et al., 2006) é apresentado um ciclo de controle chamado de ciclo de realimentação (Figura 2.3) onde sistema coleta informações de uma variedade de fontes, incluindo dispositivos de rede tradicional e fluxos de informação, mas também inclui necessidades de aplicações. Estes são analisados para construir uma nova configuração capaz de atender as necessidades definidas pelo administrador. Estas decisões são aplicadas através da rede e/ou podem ser notificadas aos administradores.

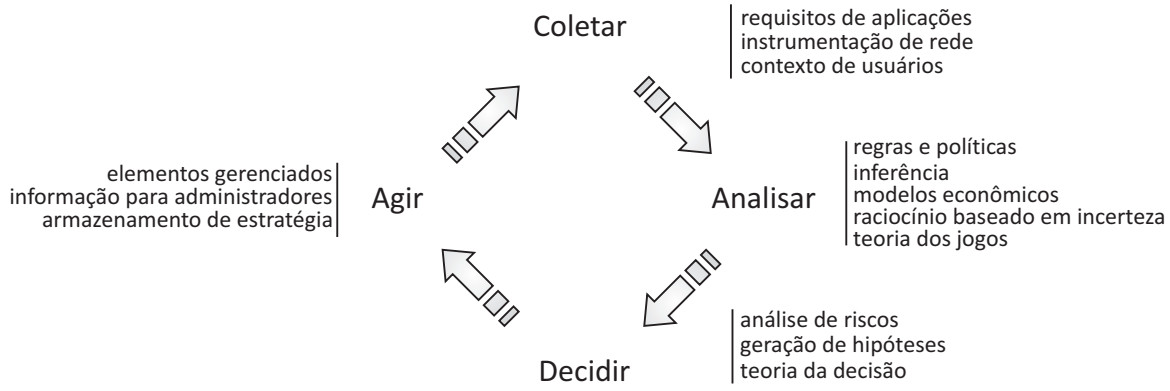


Figura 2.3. Ciclo de gerência (*control loop*) – Adaptado de (Dobson et al., 2006).

O importante é definir de forma modular quais as funções fundamentais para tais sistemas.

[4]Atomicidade é uma propriedade que trata um trabalho composto de um conjunto de tarefas como indivisível (atômico). Com a atomicidade todas as tarefas são executadas em caso de sucesso ou nenhuma delas em caso de falha, não permitindo resultados parciais da realização de um trabalho.

Na proposta desta tese o foco de atuação do arcabouço são as fases intermediárias, ou seja, onde a atuação humana é decisiva na proposta de uma nova configuração para o ambiente gerenciado. Neste trabalho serão utilizadas as definições apresentadas na tabela 2.1.

Função	Descrição
Monitoramento	Permite que o gerente autonômico possa coletar, agregar, filtrar e reportar detalhes como por exemplo as métricas ou topologias), a partir do gerenciamento do(s) dispositivo(s) sob sua responsabilidade.
Análise	Compreende o estado atual do sistema em função dos dados do monitoramento. Em geral, esta fase requer o uso de modelos complexos para visualizar das várias situações possíveis a partir dos dispositivos gerenciados
Planejamento	Consiste na definição do conjunto de ações necessárias para atingir as metas de alto nível e objetivos.
Execução	Esta função permite que o gerente autonômico altere o comportamento do recurso gerenciado.

Tabela 2.1. Separação de funções – Adaptado de (Algomine, 2011)

2.2.5 Princípios de Projeto

A computação autonômica pode ser utilizada em redes de computadores através de arcabouços construídos sobre um conjunto de princípios de projeto já utilizados em outros contextos. Muitos projetos de pesquisa têm seguido direções diferentes visando alcançar a autonomia com diferentes níveis de sucesso (Algomine, 2011).

Os princípios do comportamento autonômico em áreas de computação e engenharia podem ser aplicados, exclusivamente ou em conjunto para construir uma base para sistemas computacionais aplicados a redes autonômicas, onde é possível identificar algumas características gerais de projeto que podem ajudar a construir sistemas autonômicos levando em consideração suas especificidades particulares, como a heterogeneidade, escalabilidade e distribuição (Algomine, 2011). Dentre eles destacam-se:

- Projeto baseado em Sistemas Vivos (*Living Systems Inspired Design*) – Sistemas vivos são uma fonte importante de inspiração para a concepção de sistemas computacionais pois apresentam propriedades que os tornam autônomos. Além disso, seu entendimento é valioso para o projeto destes sistemas (Strassner et al., 2008). Entre essas propriedades, duas delas são especialmente interessantes para a concepção de sistemas autoNõmicos:

    i) Capacidade de sobrevivência bio-inspirada (*Bio-Inspired Survivability*) – O sistema sempre tenta permanecer em uma zona de equilíbrio mesmo quando qualquer estímulo externo ou interno o retira deste estado. Para tal, mecanismos de adaptação –

mudanças de curto prazo[5], somáticas[6] e genótipos[7] – têm um papel primordial na capacidade do sistema biológico de se adaptar às mudanças no ambiente para sua sobrevivência e evolução. Isso deve inspirar o projeto de futuros sistemas autonômicos podendo beneficiar áreas como redes de computadores (Balasubramaniam et al., 2007).

ii) Comportamento coletivo (*Collective Behavior*) – O termo comportamento coletivo referente a processos sociais e eventos que não fazer refletem estruturas sociais existentes, mas que emergem de uma forma espontânea (Gittler, 1975). Em termos de comportamento humano, um movimento social é uma forma de comportamento coletivo que é identificado por um tipo de ação do grupo realizada pelos indivíduos dentro do movimento. É geralmente emergente de um grande agrupamento informal de indivíduos em objetivos específicos não estando por muitas vezes sob nenhum controle e surge em função de circunstâncias ambientais. Em função deste ponto de vista, é interessante observar como uma informação local se torna global e pode influenciar o comportamento de todos os membros da comunidade. Os sistemas autonômicos podem aprender muito com o comportamento coletivo, através do projeto de um sistema com dispositivos verdadeiramente autonômicos interagindo para cumprir metas diferentes e por vezes conflitantes, O objetivo é entender como o sucesso de um indivíduo a fazer escolhas impacta, e ao mesmo tempo, depende das escolhas dos outros. O objetivo é identificar o potencial de equilíbrio que é benéfico para todas as partes.

- Projeto Baseado em Políticas (*Policy-Based Design*)– O conceito de políticas é amplamente utilizado no contexto de redes de computadores. Uma política descreve uma regra que define o comportamento de um sistema (Lupu; Sloman, 1999). Geralmente é composta de um conjunto condição/ação pré-definido pelo administrador e aplicadas na rede (Verma, 2000) atingindo um conjunto de objetivos pré-especificados. A aplicação de políticas em sistemas autonômicos possibilita uma tomada de decisão – fases de análise e planejamento – isenta da ação humana. Entretanto o sistema não estaria preparado para se adaptar a contextos e comportamentos diferentes do especificado, pois nem sempre é possível saber todas as situações e necessidades no contexto de autonomia aplicada a redes de computadores (Kephart; Chess, 2003). A utilização de políticas na computação autonômica visa facilitar o projeto de sistemas, na medida em que possibilita definir o comportamento do sistema e utilizar técnicas para resoluções de conflitos (Aib et al.,

[5] *Short-term changes* – mecanismo de resposta imediato, por exemplo, se a temperatura ambiente aumenta, o corpo humano transpira mais.

[6] *Somatic changes* – A exposição prolongada e/ou contínua à mudança de temperatura pode impactar em alteração na aclimatação do indivíduo (uma mudança de roupa, por exemplo).

[7] *Genotypic changes* – a espécie se adapta à mudança, por exemplo, com um clima frio a espécie pode ter uma pele mais grossa. Em sistemas vivos tal mudança genotípica é gravada em nível celular e se torna hereditária e irreversível na vida do indivíduo. Nota-se que esse é o processo mais lento para convergência.

2003). Políticas podem ser usadas em diferentes níveis do ciclo autonômico (monitoramento, análise, planejamento e execução), podendo aumentar os conflitos da utilização destas políticas a diferentes níveis de decisão, exigindo a definição de mecanismos de resolução eficaz de conflitos (Algomine, 2011).

- Projeto Baseado no Plano de Conhecimento (*Knowledge Plane Design*) – Alguns pesquisadores argumentam que não pode haver auto gerência de redes sem a construção de um plano eficiente e completo conhecimento capaz de capturar todas as propriedades da rede (Algomine, 2011). A chamada Internet do Futuro (*Future Internet*) é baseada no conceito de plano de conhecimento, que é uma espécie de plano de controle para a gerência da Internet do Futuro (Clark et al., 2003). O plano de conhecimento foi originalmente proposto com o objetivo de construir uma rede que pode adaptar-se através de instruções de alto nível para cumprir mudanças de requisitos, corrigir problemas detectados e descobrir automaticamente novos elementos autonômicos (Clark et al., 2004). O plano do conhecimento é uma camada superior que agrega significado semântico a todas as informações de gerência sendo concebido como uma estrutura de alto nível que fornece serviços e opiniões a outros elementos da rede (Siekkinen et al., 2007). Esta abordagem requer o uso de inteligência artificial (IA) e técnicas cognitivas para atingir os objetivos descritos (Algomine, 2011).

- Projeto Adaptativo (*Adaptive Design*) – A auto-adaptação possibilita a mudança de comportamento de um sistema, quando a auto avaliação indica que ele não tem realizado o que é esperado ou quando é possível melhorar seu desempenho. Este projeto apresenta um conceito atraente para o desenvolvimento de sistemas auto-gerenciáveis visando o atendimento total ou parcial das das mudanças espaciais, operacionais ou estratégicas. No contexto de redes de computadores, mudanças espaciais podem ser entendidas como alterações da topologia, enquanto mudanças operacionais relacionam-se com ataques e/ou falhas. Já as mudanças estratégicas podem ser visualizadas como alterações nos objetivos ou SLAs. Espera-se que este projeto adaptativo tenha capacidade para reagir às mudanças, aprendendo a partir das experiências passadas, sendo capaz de responder a situações de forma imediata. Como seu conhecimento aumenta com o tempo, a habilidade em responder de forma eficiente para diversas situações conhecidas e desconhecidas também irá aumentar.

2.3 CLASSIFICAÇÃO DOS SISTEMAS AUTONÔMICOS

As soluções autonômicas podem ser aplicadas a diferentes contextos e possuir distintas funcionalidades, tornando necessário definir critérios abrangentes para avaliar tais soluções. Com a definição destes critérios se tornará mais viável a comparação entre as principais pesquisas da área, inclusive o arcabouço apresentado nesta tese. Assim, nesta tese consideraremos seis

critérios para classificação de sistemas autonômicos que podem ser vistos abaixo (Figura 2.4).

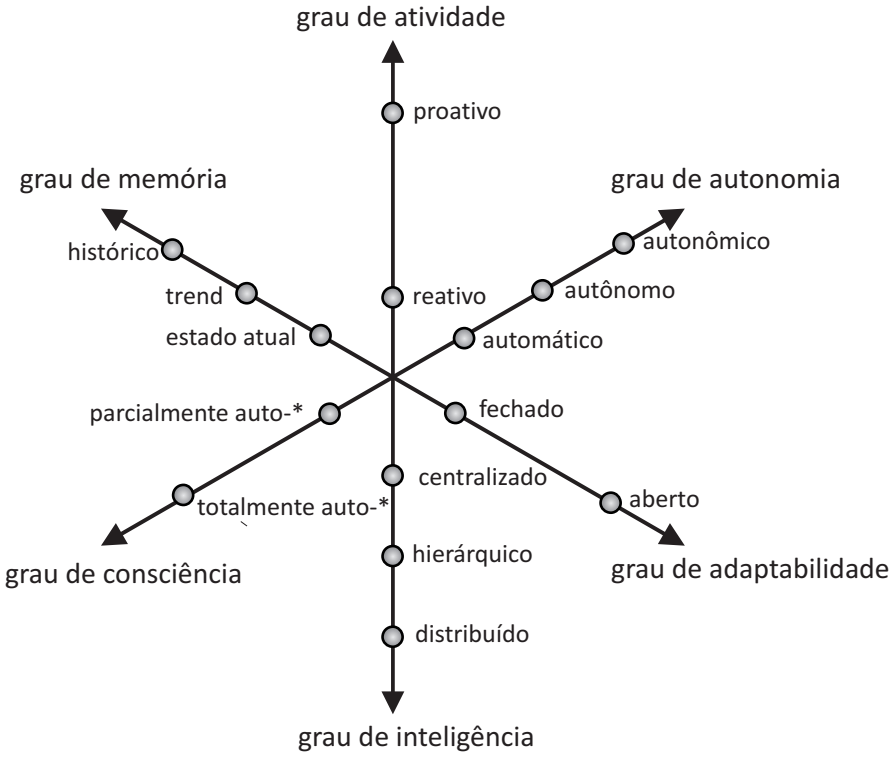


Figura 2.4. Classificação dos sistemas autonômicos (Samaan; Karmouch, 2009).

i) Grau de Atividade (*Activity Degree*) – Um sistema autonômico pode satisfazer as propriedades auto-* através de um comportamento reativo ou proativo. No modo reativo o sistema autonômico tenta identificar eventos significativos ou problemas que podem reduzir o seu desempenho ou prejudicar o cumprimento de seus objetivos, e efetua uma nova configuração baseada no estado atual. Este modo é o mais comum entre os trabalhos apresentados na literatura (Samaan; Karmouch, 2009). Por outro lado, um sistema autonômico proativo fica constantemente monitorando o estado atual do ambiente visando manutenibilidade do desempenho especificado pelo administrador. Esta proatividade do sistema tem características positivas – como a redução de falhas do ambiente gerenciado através de medidas preventivas – e negativas – pois este modelo proativo pode gerar problemas de auto-estabilização (*Self-Stabilization*) em redes de computadores (Boutaba; Xiao; Zhang, 2009). Certamente um sistema proativo é fundamental para ambientes não tolerantes a falhas.

ii) Grau de Adaptabilidade (*Adaptability Degree*) – Assim como as arquiteturas de software, os sistemas autonômicos podem ser classificados como abertos e fechados (Oreizy et al., 1999). A adaptabilidade no contexto autonômico está associada a habilidade/inabilidade de um sistema autonômico adaptar-se a mudanças em função de estados criados ou pré-definidos (Oreizy et al., 1999), ou seja, está associada a estratégia de escolha autonômica

entre estados especificados pelo administrador ou criados pelo próprio sistema. Nesta tese consideraremos apenas os sistemas autonômicos como abertos e os sistemas fechados serão considerados automáticos. Um sistema aberto pode criar estados em função de interações de componentes ou através da experiência adquirida (Tesauro, 2007)

iii) Grau de Inteligência (*Intelligence Degree*) – Em ambientes dinâmicos, como o caso de gerenciamento de rede, um sistema autônomo deve apresentar alguma inteligência na realização das propriedades auto-*. Dessa maneira, o grau de inteligência de um sistema autonômico se refere à capacidade de aprender (por exemplo, a partir de experiências passadas), a fim de melhorar sua operação. Em geral podemos ter um gerente centralizado autonômico (Kephart; Das, 2007) que controla todos os outros componentes no sistema. Neste caso os componentes não precisam de ser autonômicos, uma vez que toda decisão é tomada pelo gerente. A outra visão consiste na inteligência distribuída, que indica a tomada de decisões realizada por todos os componentes do sistema gerenciado (ou pela maioria deles), que podem ser alcançados através da utilização de algoritmos bio-inspirados (Martin-flatin; Srivastava; Westerinen, 2003), nos quais a inteligência pode emergir de um comportamento coletivo de pequenas entidades com inteligência limitada (Truszkowski et al., 2006).

iv) Grau de Consciência (*Awaress Degree*) – Um certo grau de consciência do ambiente gerenciado é necessário para a realização de qualquer funcionalidade autonômica. Quando os sistemas eram não-autonômicos a *expertise* estava associada ao administrador, ou seja, quanto maior a *expertise* do administrador, maior a possibilidade de sucesso da gerência. Além disso, o administrador não precisava armazenar o estado preciso de todos os dispositivos para chegar a uma solução. O mesmo pode acontecer com os sistemas autonômicos, a base de conhecimento (*Knowledge Base*) pode manter um pequeno conjunto de parâmetros operacionais para entender o comportamento atual do ambiente..

v) Grau de Memória (*Memory Srength*) – Este critério se refere à capacidade do sistema para lembrar o histórico do comportamento do sistema gerenciado, ou seja, os problemas e suas soluções encontradas pelo sistema autonômico ao longo do tempo. Em alguns sistemas autonômicos, o conhecimento do estado atual do sistema é suficiente para executar as funcionalidades de auto-gerência (Kephart; Das, 2007). Já em ambientes altamente dinâmicos, conhecimento das tendências de comportamento e histórico de ações passadas pode aumentar significativamente o desempenho do sistema (Samaan; Karmouch, 2005)). Este conhecimento normalmente é armazenado na base de conhecimento do sistema autonômico visando evitar erros cometidos no passado ou reduzir o tempo de busca de soluções, quando o tempo de busca da solução ótima não for factível no contexto em que o sistema autonômico é utilizado. É importante ressaltar que um histórico permanente pode não ser desejável em alguns contextos, pois implica em um maior custo computacional para os sistemas autonômicos, além de não garantir o sucesso do sistema. É altamente

desejável que sistemas autonômicos considerem apenas o histórico mais relevante, que não necessariamente é o mais utilizado.

vi) Grau de Autonomia (*Autonomic Degree or Autonomicity*) – Este critério refere-se ao grau de independência da atuação humana num sistema autonômico. Em geral um sistema automático é aquele que usa um circuito fechado de controle monitorado por um gerenciador que compara as variáveis com um valor pré-especificado ou desejado. O administrador além de especificar os valores pode também definir uma ou mais ações a serem executadas. Em geral estes são similares a sistemas baseados em regras, ou seja, um operador humano especifica, com antecedência, o desempenho do sistema desejado e a solução de cada problema possível que o sistema deve aplicar uma vez que esse problema é encontrado. A introdução da autonomia muda a intervenção humana nesses sistemas (Want; Pering; Tennenhouse, 2003) (Jrad et al., 2006). A computação autonômica introduz algum grau de independência entre o monitoramento (estado atual do ambiente gerenciado) e ação que o sistema deve executar como resposta a mudanças do ambiente. O foco da execução de operações de um sistema autonômico é garantir a continuidade e eficiência da sua funcionalidade esperada. É altamente desejável que estes sistemas autonômicos sejam capazes de aprender com as ações tomadas, sendo necessário a medição da eficiência das soluções já realizadas por estes sistemas.

Estes critérios farão parte da avaliação da solução proposta nesta tese, apresentada no Capítulo 4.

2.4 GERÊNCIA AUTONÔMICA DE REDES E SEUS DESAFIOS

Os princípios da computação autonômica podem ser utilizados no contexto de redes de computadores tendo em vista que tais ambientes são considerados complexos (Bezerra; Martins, 2009a; Denko; Yang; Zhang, 2009), imprevisíveis e de larga escala (Jennings et al., 2007; Gelenbe, 2009). O objetivo de aplicar a abordagem de computação autonômica em redes de computadores é simplificar o processo de gerência e reduzir a intervenção humana (Dobson et al., 2006), que por si só é difícil, se não impossível de gerenciar (Jennings et al., 2007). A complexidade na gerência de redes de computadores pode tornar a intervenção humana como um ponto de falha (Jrad et al., 2006).

Decorrente disso, pesquisadores e profissionais estão perseguindo a abordagem de gerência autonômica para as redes de computadores, que consiste na capacidade de auto-gerência dos dispositivos de rede de acordo com o comportamento do ambiente gerenciado e com os objetivos de negócio que a rede como um todo se propõe alcançar(Jennings et al., 2007). Contudo, aplicar o conceito de computação autonômica na área de redes de computadores possui vários desafios, dos quais destacamos:

1. Gerência de funcionalidade heterogênea (*Management of heterogeneous functionality*)

(Jennings et al., 2007)

Um dos problemas na aplicação do princípio autonômico em redes de computadores é que essas são compostas de vários dispositivos com capacidades(*capabilities*) diferentes. Devido a heterogeneidade de modelos e o não cumprimento de padrões, os dispositivos de rede têm diferentes implementações de suas capacidades e fornecem monitores distintos, geralmente executando conceitos similares. Sendo assim, é necessário abstrair as funcionalidades específicas para facilitar uma maneira padrão de reconfigurar os dispositivos. Esta característica facilitará a auto-configuração (*Self-Configuration*). Isto permitirá também que recursos não autonômicos possam ser administrados por sistemas autonômicos.

2. **Adaptabilidade(*Adaptability*) (Jennings et al., 2007; Dobson et al., 2006)**

   Adaptabilidade consiste na capacidade do sistema de se adaptar em resposta a mudanças nos requisitos dos usuários/aplicações, regras de negócio, e/ou condições ambientais, como falhas. Como exemplo, um sistema autonômico para gerência da qualidade de serviço em redes de computadores deve ser capaz de se adaptar a diferentes redes, independente do tamanho, topologia, carga da rede ou regras de negócio.

3. **Aplicação de técnicas de aprendizagem e raciocínio para suporte a interação inteligente (*Application of learning and reasoning techniques to support intelligent interaction*) (Jennings et al., 2007)**

   Sistemas devem ser capazes de compreender o estado atual da rede através da coleta pontual dos dispositivos gerenciados. por exemplo, as estatísticas, que são dados estáticos de cada dispositivo, podem ser recolhidas e analisadas para determinar se o estado apresenta problemas ou pode ser otimizado. Note que os dados atuais da gerência não informam ao administrador, por exemplo, o motivo de um congestionamento numa rede. Desta forma, tal informação deve ser inferida a partir da correlação dos dados (*reasoning*) e, se possível, um estudo de casos anteriores. Se o sistema também for capaz de guardar informações para referências futuras (*learning*), tal sistema estará caminhando para o estado da arte no contexto da autonomia.

4. **Comportamento cooperativo em face a concorrência (*Cooperative behavior in the face of competition*) (Dobson et al., 2006)**

   Arquiteturas de sistemas autonômicos devem ter a capacidade de possibilitar a cooperação entre elementos autonômicos visando o bem estra global. No caso de sistemas não centralizados, o desafio é evitar o parasitismo, comportamentos egoístas ou maliciosos.

5. **Falta de objetivos e controle centralizados (*Lack of centralized goals and control*) (Dobson et al., 2006)**

   A especificação e controle centralizados dos objetivos facilita o comportamento cooperativo entre elementos. Entretanto a descentralização oferece mais robustez à gerência no que diz respeito ao controle global dos objetivos.

6. **Auto-estabilização (*Self-Stabilization*) (Denko; Yang; Zhang, 2009)**

   O conceito de auto-estabilização (Dijkstra, 1974), proposto inicialmente para sistemas distribuídos, tem muita relevância dentro do contexto de gerência autonômica de redes de computadores. Isso porque, no contexto descentralizado, o estado de cada elemento autonômico deve ser considerado legítimo, pelos outros elementos. Por exemplo, suponha um elemento que inicializa com algum estado e que é passível de falhas transientes. Dessa forma qualquer falha desta natureza, fará com que o mesmo retorne a sua situação inicial e caso não seja tratado, os outros elementos poderão não identificar tal falha.

7. **Confiabilidade (*Reliability*) (Gelenbe, 2009)**

   Dentro do contexto de sistemas autonômicos a confiabilidade de um sistema pode ser medida pela corretude das ações realizadas durante a gerência. Um sistema autonômico confiável é aquele que funciona, sem interrupções, conforme o esperado ou prometido (de acordo com as regras de negócio), fornecendo um serviço adequado sempre que possível.

6. **Robustez (*Robustness* (Kephart; Chess, 2003)**

   O conceito de robustez se confunde ao de confiabilidade. Um sistema robusto pode ser definido como um sistema tolerante a estados não desejáveis, ou seja, um sistema robusto tem o funcionamento persistente mesmo com perturbações contínuas proveniente de falhas, ataques ou comportamento degradante visto no ambiente decorrente da ação de usuários e/ou aplicações.

7. **Escalabilidade (*Scalability*) (Bezerra; Martins, 2009a)**

   Os sistemas autonômicos são utilizados para gerência de sistemas complexos, independente da quantidade de dispositivos gerenciados, que podem variar de dezenas a milhares de elementos. Estudos de caso e pesquisas na área não dão a devida atenção à escalabilidade, limitando-se apenas a informar que existem subdomínios gerenciados, não indicando como estes são determinados. A depender do contexto, uma escolha infeliz dos subdomínios pode impactar na qualidade das soluções de um sistema autonômico.

8. **Análise de dados e visualização (*Data analysis and visualization*) (Pras et al., 2007)**

   Tipicamente, sistemas de gerência de redes coletam uma grande quantidade de informações de monitoramento que geralmente são visualizadas através de mapas ou representações topológicas, não levando em consideração a granularidade. Além disso, visualização de tráfego tem geralmente como objetivo de indicar alto volume de tráfego, não indicando o quanto deste tráfego é não-desejável ou fora do padrão. Isso torna a complexidade ainda maior. Possivelmente, o estudo do perfil e crescimento do tráfego, tenha que ser executado por terceiros ou analisado através de técnicas estatísticas.

2.5 O ESTADO DA ARTE

A necessidade de administrar um ambiente gerenciado, como uma rede de computadores, é um tema de pesquisas que se desenvolve há alguns anos. Entretanto, a complexidade dos ambientes gerenciados cresce constantemente por causa do crescimento e da variedade de serviços disponíveis nas redes de computadores. O principal desafio abordado pela computação autonômica é justamente o crescimento desta complexidade.

A arquitetura de computação autonômica proposta pela IBM (Kephart; Chess, 2003; Farha; Leon-garcia, 2006; Computing, 2006) é o trabalho pioneiro dentro desta abordagem, que define um arcabouço (*framework*) abstrato para a auto-gerenciamento de sistemas. Basicamente um sistema autonômico é visualizado como uma coleção de elementos autonômicos, onde cada um deles pode ser dividido em um gerenciador (ou gerente) autonômico e um recurso gerenciado. A comunicação entre eles é realizada através de sensores (*sensors*) e executores (*effectors*). Os sensores são utilizados para a obtenção de informações do(s) dispositivo(s) gerenciado(s) e os executores são a interface de recebimento de ações para a mudança de comportamento do dispositivo.

O auto-gerenciamento é realizado através de um ciclo de controle contínuo, denominado *control loop*, que possui quatro fases: monitoramento, análise, planejamento execução (conforme visto na Seção 2.2.4) com suporte a gestão de conhecimento, políticas e outras considerações relacionadas.

Entretanto o modelo de computação autonômica proposto pela IBM só fornece a orientação conceitual para a elaboração de sistemas de auto-gerenciáveis, e na prática, o modelo de informação deve ser mapeado para algo implementável. Para tal, técnicas de monitoramento, alocação dinâmica de recursos, desenvolvimento de elementos autonômicos, gerenciamento escalável, coordenação entre sistemas autonômicos, dentre outras, precisam ser desenvolvidas (Cheng et al., 2006). Diversos trabalhos visam propor soluções para tais problemas dentro de um contexto específico (Agarwal et al., 2003; Graupner; Andrzejak, 2003; Serrano et al., 2006; Strassner et al., 2009b). Nesta seção serão destacados três relevantes pesquisas na área de computação autonômica empregadas em redes de computadores.

2.5.1 FOCALE – *Foundation-Observation-Compare-Act-Learning-rEasoning*

A arquitetura FOCALE é baseada na observação de que os objetivos de negócios, requisitos de usuário e bem estar do ambiente sejam garantidos de forma dinâmica. Para isso utiliza mais de um ciclo de controle, considerando que apenas um é insuficiente para permitir que a rede de reaja adequadamente às mudanças observadas. Um dos ciclos do FOCALE foi concebido para o controle de manutenção, sendo utilizado quando não são encontradas anomalias (ou seja, quando o estado atual é satisfatório ou quando o estado está em transição para alcançar um estado satisfatório). O outro ciclo de controle de ajustes que é utilizado quando uma ou mais ações de reconfiguração política deve ser realizada, e/ou novas políticas devem ser codificadas

e implantadas, tendo uma função corretiva.

A justificativa para a manutenção de dois ciclos de controle é a impossibilidade de manter todas as informações necessárias para gerenciar redes de larga escala, contendo um grande número de dispositivos heterogêneos (em termos de funcionalidades disponíveis, específicos de fornecedores modelo de programação e configuração específica). Uma visão dos ciclos de controle pode ser vista na Figura 2.5.

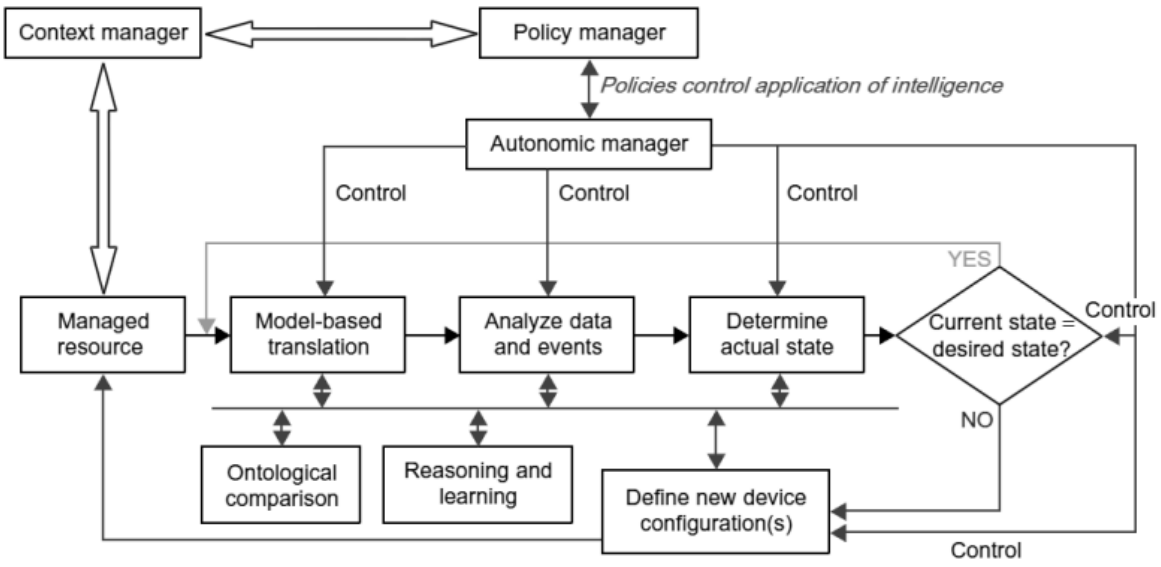


Figura 2.5. Uma visão simplificada da arquitetura do FOCALE e seus ciclos de controle (Strassner, 2011) (Strassner; Hong; Kang, 2009)

.

O FOCALE também assume que qualquer recurso gerenciado (que pode ser tão simples como uma interface de dispositivo ou tão complexo como todo uma rede de computadores) pode ser associado a um elemento de gestão autônomo (AME). Este AME efetua a comunicação do recurso gerenciado com um gerente autonômico (AM) usando uma camada de tradução baseada em modelos (MBTL). Este modelo pode ser modularizado de forma que um AME possa gerenciar um conjunto de AMEs, tornando o gerenciamento hierárquico (Figura 2.6).

Um AM é independente das funcionalidades específicas do objeto gerenciado, o que facilita a comunicação entre os AMEs para a coordenação da tomada de decisões. Cada AM realiza a funcionalidade de gerenciamento autonômico, descrito através de um gerente de eventos (*event manager*), um gerente de estados (*state manager*), um gerente de ação (*action manager*), um pensador (*reasoner*), um aprendiz (*learner*), um analisador de políticas (*policy analyzer*) e um ponto de decisão de políticas (*Policy Decision Point - PDP* (Chan et al., 2001).

A comunicação entre todos estes subcomponentes ocorre através do modelo de informação DEN-ng, um modelo de objeto reflete o estado atual do recurso gerenciado a AME (s), a ontologia do sistema, e o conjunto de políticas que regem o recurso gerenciado (Strassner et al., 2009a).

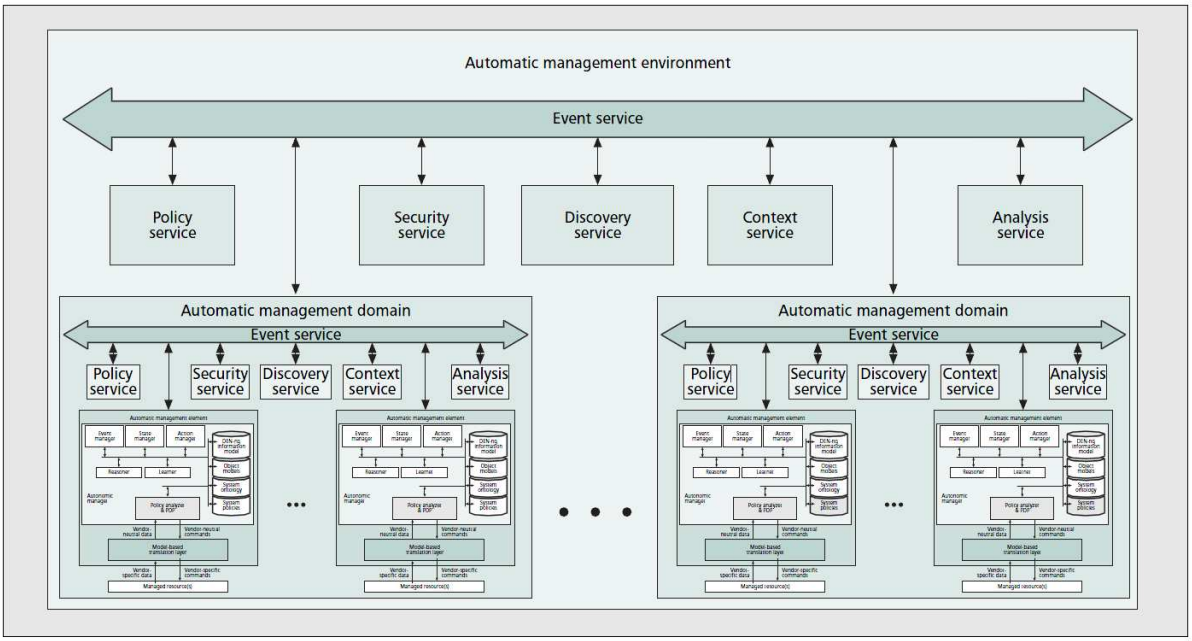


Figura 2.6. A arquitetura funcional do ambiente de gerência do FOCALE (Jennings et al., 2007).

2.5.2 ANEMA – *Autonomic Network Management Architecture*

Arquitetura de Gerência de Redes Autonômicas tem como objectivo introduzir mecanismos diferentes para permitir que redes baseadas em IP se tornem auto-gerenciáveis. Esta arquitetura se baseia na teoria de função de utilidade e introduz vários conceitos, tais como objetivo, políticas comportamentais e políticas função de utilidade para a rede o instrumento. A solução também define um conjunto de mecanismos para transformar requisitos de alto nível humano em objetivos expressos em termos de NUF *(Function Utility Network)*, que são aplicadas nos equipamentos da rede (Derbel; Agoulmine; Salaün, 2009). A NUF representa a utilidade da rede do ponto de vista do operador humano. Esta função é descrita em forma de otimização analítica. Sua avaliação permite especificar o espaço de estados viáveis (Derbel et al., 2011).

2.5.3 ANA – *Autonomic Network Architecture*

O Projeto ANA (*Autonomic Network Architecture*) visa explorar novas formas de organizar e utilizar redes de computares além da atual arquitetura da internet. Tem como objetivo final desenvolver uma arquitetura de rede autonômica, que permita a formação flexível, dinâmica e totalmente autonômica dos nós da rede, assim como de redes inteiras. Tal projeto conta com a participação de universidades e institutos de pesquisa da Europa e América do Norte.

A arquitetura de rede deste projeto permite a adaptação dinâmica e reorganização da rede de acordo com o seu estado atual e/ou necessidades econômicas e sociais dos usuários. Para tanto, o projeto possui dois objetivos complementares:

- Objetivo científico – Visa identificar os princípios fundamentais de redes autonômicas,

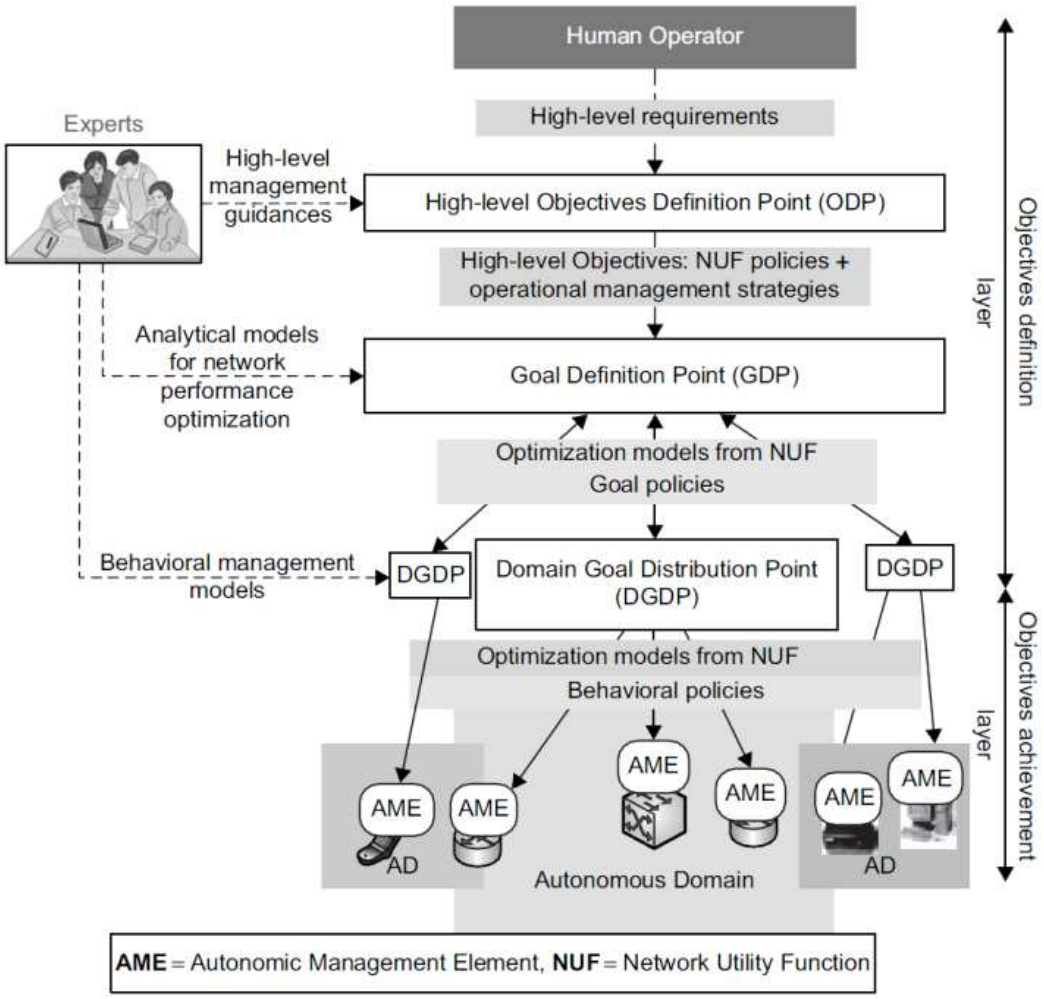


Figura 2.7. Uma visão global do ANEMA (Derbel; Agoulmine; Salaün, 2009).

com atuação em redes de larga escala. A principal premissa deste objetivo é que o dimensionamento de rede é consequência de uma rede que evolui e inclui os atributos auto-* essenciais da concepção autonômica, tais como auto-otimização, auto-monitoramento, auto-reparo, e auto-proteção.

A hipótese é que com a utilização desses atributos auto-* naturalmente as redes estarão aptas a se tornar-se escaláveis. A nova arquitetura de rede necessitará de um arcabouço para permitir sua reconfiguração possibilitando a formação flexível, dinâmica e totalmente autonômica de grandes redes em que as funcionalidades de cada nó da rede também são realizadas de forma autonômica. Esta arquitetura deve permitir a adaptação dinâmica e reorganização da rede de acordo com o tráfego da rede, regras de negócios e SLAs.

- Objetivo tecnológico – A segunda premissa deste projeto é colocar as novas ideias e conceitos bem sucedidos em prática. Ou seja, este projeto assume o desafio de não apenas produzir resultados originais de pesquisa científica e um projeto inovador para a arquitetura, mas também mostrar que eles funcionam em situações reais. Isso permitirá a

utilização da experiência adquirida experimentalmente como retorno para refinar os modelos e resultados de pesquisas. A meta tecnológica do ANA é, portanto, construir uma arquitetura de rede experimental autonômica, e demonstrar a viabilidade desta rede autonômica dentro dos próximos anos. O objetivo é demonstrar a auto-organização dos nós individuais em uma rede compondo topologias de até 105 nós ativos (roteamento). Para demonstrar a a escalabilidade são previstas a interconexão da estrutura de rede dos sites participantes e simulações.

Estes dois objetivos (técnico e científico) se complementam e reforçam mutuamente em um ciclo pois protótipos dos resultados de pesquisa são implementados inicialmente num *testbed*. Os resultados preliminares experimentais poderão ser usados como um retorno para aperfeiçoamento do projeto e para obtenção de resultados de pesquisa mais precisas e realistas.

O conceito fundamental da arquitetura é o ponto de envio de informações (IDP). Os IDPs são inspirados pelo trabalho em ponteiros de rede. Os IDPs são tipicamente ligados a blocos funcionais (FB). Estes últimos por sua vez são unidades de processamento de informações que implementam a funcionalidade de transmissão (por exemplo, envio e recebimento de pacotes IP), ou algumas funcionalidades adicionais como, por exemplo, monitoramento de tráfego. Geralmente, os blocos funcionais podem ser usados para implementar serviços de rede.

O objetivo do IDP é fornecer uma comunicação genérica entre os vários blocos funcionais em execução e oferecer flexibilidade de re-organizar os caminhos de comunicação. Exemplos de caminhos de comunicação são ilustrados na figura 2.8. Nesta figura, um bloco funcional (FB1) envia dados para um IDP *a* que é ligado a outro bloco funcional (FB2).

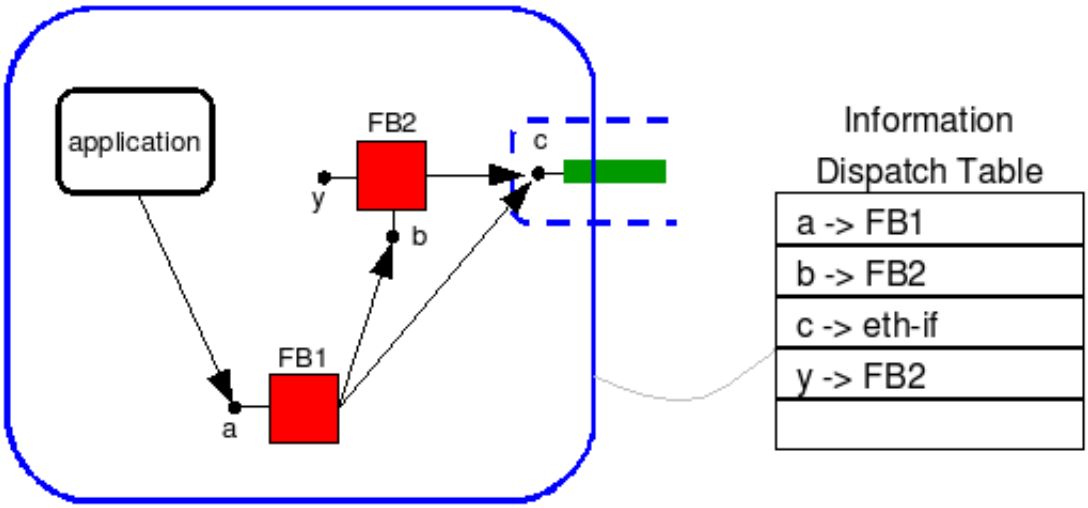


Figura 2.8. Exemplo de operação dos blocos funcionais

Para prover a autonomicidade, o projeto suporta as propriedades auto-* requerendo uma

troca de informações de diferentes tipos dentro de sua arquitetura: informações de configuração, de sinalização e de monitoramento. As informações de configuração são usadas para configurar um dispositivo de rede. As informações relativas à sinalização compreendem essencialmente dados de controle, por exemplo qualquer tipo de mensagem de protocolo de roteamento. Por fim mensagens de monitoramento diz respeito a informações que caracterizam o comportamento dinâmico de um sistema. Normalmente pode-se monitorar o estado interno de uma entidade de rede, por exemplo, a quantidade de memória livre, comprimento da fila atual e largura de banda disponível de cada enlace.

Os trabalhos descritos não tratam devidamente a gerência autonômica escalável, ou seja, o gerente autonômico não tem garantias de atender a uma quantidade maior de dispositivos e o aumento do volume de tráfego da rede com diferentes topologias. Alguns trabalhos citam que podem haver mais de um gerente no ambiente gerenciado, mas a divisão dos domínios gerenciados, bem como a comunicação entre eles não é devidamente citada.

Quanto ao processo de tomada de decisões, o FOCALE apresenta uma estrutura hierárquica que permite centralização do processo. Já o projeto ANA, a descentralização da tomada de decisões aparentemente viabiliza a escalabilidade, mas testes específicos aina não foram realizados.

No próximo capítulo apresentaremos o arcabouço desenvolvido visando a escalabilidade na gerência autonômica de redes.

Capítulo 3

Não é o mais forte que sobrevive, nem o mais inteligente, mas o que melhor se adapta às mudanças – Charles Darwin

UM ARCABOUÇO COM CARACTERÍSTICAS AUTONÔMICAS PARA A GERÊNCIA DE REDES

O capítulo 2 apresentou a evolução dos sistemas de gerência de redes de computadores e alguns de seus problemas inerentes, tais como, o problema da escalabilidade e da adaptabilidade na gerência autonômica de redes. O objetivo deste capítulo é apresentar um arcabouço com características autonômicas que esteja apto a ser utilizado para a gerência autonômica de redes, com um foco de aplicabilidade e estudo de caso específico para a qualidade de serviço).

Para tal, inicialmente serão especificados os requisitos esperados do arcabouço (Seção 3.1), a estrutura do arcabouço estruturado em planos (Seções 3.2), dos quais o plano de decisão é detalhado (Seção 3.2.3). Em seguida, é apresentada uma estratégia de particionamento (NPCE – *Network Partitioning Computing Engine*) integrada ao motor de busca de soluções visando o tratamento da escalabilidade em relação a utilização em ambientes com quantidades diferentes de dispositivos e/ou volume de tráfego (Seções 3.3, 3.4, 3.5 e 3.6). Por fim, a operação (Seção 3.7) do modelo em ambientes computacionais é exemplificada.

De maneira geral, o arcabouço proposto pode ser considerado como a realização de uma modelagem do problema de gerência autonômica de redes com requisitos, características e foco específicos os quais serão detalhados na Seção 3.1).

3.1 MODELAGEM E REQUISITOS DO ARCABOUÇO

O arcabouço com características autonômicas para a gerência de redes pode ser genericamente modelado e percebido como um sistema que recebe o estado atual do ambiente gerenciado (estado da rede) e um conjunto de requisitos de gerência, especificados pelo administrador do ambiente (requisitos de negócio e/ou administração da rede). Este modelo pode ser visualizado na Figura 3.1.

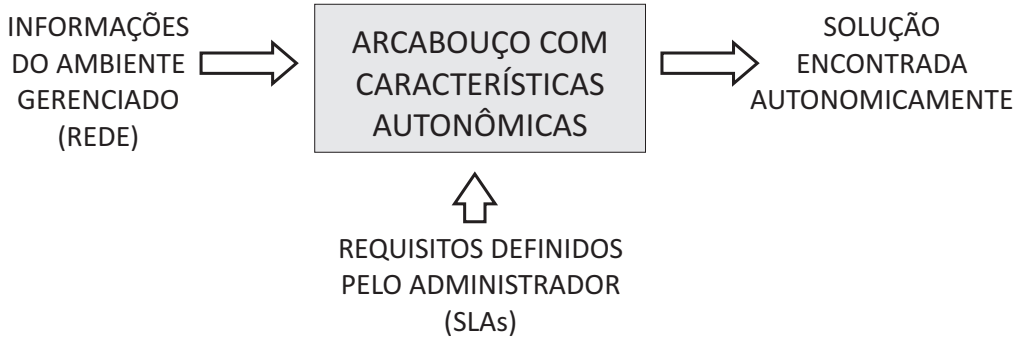


Figura 3.1. Modelagem genérica do arcabouço com características autonômicas

O arcabouço produz então como saída uma nova configuração de rede, que atende aos requisitos especificados pelo administrador. A busca da nova configuração de rede é realizada sem a intervenção direta do administrador (gerente) e assim sendo, assume a característica de auto-gerenciamento no escopo das soluções ditas autonômicas.

Os requisitos básicos definidos para o arcabouço com características autonômicas, conforme o modelo básico definido na Figura 3.1 são os seguintes:

- Autogerenciamento no contexto da gerência de redes de computadores;
- Modularidade;
- Exequibilidade em relação ao tempo de busca de novas soluções para a rede gerenciada;
- Escalabilidade.

O requisito de auto-gerenciamento é efetivamente uma condição básica de implantação do sistema, na medida em que se propõe uma solução com características autonômicas. Em outras palavras, este é um requisito básico e intrínseco. O auto-gerenciamento consiste na alteração do estado atual do ambiente gerenciado (rede) de acordo com os requisitos através da busca de uma solução (nova configuração) a partir de um conjunto de possíveis soluções capazes de atender a tais requisitos. Além disso, uma característica fundamental que deve ser considerada para o arcabouço no contexto do requisito de auto-gerenciamento é a questão do seu foco de atuação e da aplicabilidade da solução proposta.

De faro, o arcabouço proposto tem uma característica de modularidade conforme explicado adiante mas, para efeito de realização e validação da proposta, faz-se necessário a definição de um escopo de aplicação do mesmo. Neste caso, o escopo e estudo de caso considerados são focados na qualidade de serviço em redes.

A modularidade é uma característica importante e relevante do arcabouço proposto, que poderá considerá-la em diferentes perspectivas e níveis. Como requisito geral, inicialmente é necessário que o arcabouço seja capaz de utilizar diferentes algoritmos no escopo da sua estruturação e implementação. Além disso, sua estruturação geral do arcabouço deve ser tal forma que não adote uma estrutura monolítica orientada apenas para o foco de aplicação apresentado

nesta tese, ou seja, o foco desenvolvido nesta tese é orientado para a qualidade de serviço mas o arcabouço poderá ser adaptado para outros cenários de aplicação através da introdução e escolhas de novos algoritmos ou abordagens para os elementos de sua arquitetura. Além disso, a inclusão de novas funcionalidades deve ser facilitada, para que o administrador da rede consiga estender o arcabouço visando o atendimento ou adaptação ao seu contexto, se necessário for. Por exemplo, o modelo não deve impor restrições ao algoritmo de caminho mais curto (*Shortest-Path Algorithm*) para busca de caminhos – durante a alocação dos fluxos na rede – para uma solução visando a qualidade de serviço ou par outra de otimização no contexto de engenharia de tráfego.

A exeqüibilidade em relação ao tempo de busca por novas soluções para a rede gerenciada é um requisito a ser considerado no projeto do arcabouço. Em efeito, a busca de forma autonômica de uma nova configuração para o ambiente gerenciado pode levar a um tempo de resposta inaceitável para os requisitos especificados pelo administrador. Como exemplo ilustrativo, considere que a computação de um novo estado da rede, visando um determinado desempenho esperado para uma configuração com 100 dispositivos (roteadores) pode necessitar de alguns minutos de processamento e, como tal, existe uma forte tendência nos sistemas de gerência não autonômicos para que não sejam executados dinamicamente (execução *offline*).

O requisito de escalabilidade é mais um aspecto importante e significativo para o arcabouço. Atualmente a heterogeneidade e pluralidade das redes atuais em termos tecnológicos são uma realidade. Além de apresentarem tais características, as redes atuais possuem uma tendência a ter um número elevado de dispositivos (sejam eles roteadores, comutadores, dispositivos de acesso, outros equipamentos específicos de redes). Assim sendo, o arcabouço proposto deve ser capaz de atuar em redes de diferentes cardinalidades (nós e enlaces) sem restrições topológicas. Em resumo, é fundamental que a solução apresentada nesta tese possa escalar na busca de novas configurações para a rede pois esta é uma questão de ordem prática que atualmente já representa uma forte tendência de cenário para a gerência de redes.

É importante ressaltar também que existe uma relação entre os requisitos de exequibilidade na busca da solução e escalabilidade. Assim, em termos da implantação, o arcabouço abordará estes dois requisitos de forma correlata e concomitante através de uma abordagem de particionamento vista na Seção 3.3.

A realização dos requisitos indicados leva intrinsecamente ao atendimento de um conjunto de objetivos e características decorrentes da operação do arcabouço como segue:

i) Redução do tempo de alocação dos recursos para a rede gerenciada;

ii) Identificação da melhor qualidade de serviço (solução) possível;

iii) Maximização da capacidade de sobrevivência da rede.

A redução do tempo de alocação de recursos para a rede é uma decorrência direta do requisito de exequibilidade do tempo de resposta. Em suma, os administradores poderão computar

novas alocações de recursos de forma mais eficiente.

A identificação da melhor qualidade de serviço possível (*Performability*) corresponde a um aspecto da operação do arcabouço. Na verdade, o arcabouço permite que o administrador indique requisitos mais ou menos restritivos (nível de exigência) e o mesmo ainda proverá o conjunto de soluções computado como função destes requisitos. Consequentemente, uma característica desejável para o arcabouço é que o administrador tenha a liberdade de identificar a melhor qualidade de serviço, conforme seus requisitos e fazer as escolhas apropriadas, ou seja, o grau de desempenho da nova configuração é, em termos práticos, ajustado pelo administrador. Um aspecto operacional decorrente deste fato é que poderá existir uma situação onde não exista solução para os requisitos apresentados pelo administrador. Por exemplo, numa situação de falha por parada. De maneira geral, o arcabouço na sua operação procurará sempre apresentar a melhor solução encontrada. Outros detalhes de cenários de operação serão vistos na Seção 3.7.

A maximização da capacidade de sobrevivência da rede (*Survivability*) é outro aspecto de operação importante decorrente dos requisitos do arcabouço. De fato, o arcabouço pode ser acionado para a computação de um novo estado de configuração da rede sempre que falhas por parada venham a acontecer nos dispositivos. Logo, como o arcabouço é um sistema com tempo de computação melhorado em relação aos sistemas de gerência não autonômicos, consequentemente a capacidade de sobrevivência da rede fica maximizada com a obtenção de novas soluções para implantação sem a intervenção do ser humano (mais lenta e não executada dinamicamente, ou seja, *offline*).

É importante que estes requisitos sejam atendidos mesmo que as redes gerenciadas tenham dinamicidade de tráfego *traffic dinamicity*) e que em alguns momentos operem com sobrecarga (*traffic overload*). A junção destas duas características corresponde à capacidade de um sistema em tolerar variação de tráfego (Sterbenz et al., 2010). Assim o arcabouço deve estar apto a prover de forma autonômica os requisitos considerando estas necessidades. No escopo desta tese, a característica de tolerância a mudanças enbloga as características de tolerância de variação de tráfego, seja por dinamicidade do perfil e/ou sobrecarga, e de sobrevivência, mediante a falhas por parada.

Segundo (Sterbenz et al., 2010), a sobrevivência, a tolerância de tráfego e a manutenção da qualidade de serviço são características necessárias – mas não suficientes – para a garantia da robustez. Um sistema que possui suporte a tais características não pode ser considerado robusto, uma vez que outras áreas devem ser atendidas como por exemplo: tolerância a falhas e segurança. Podemos indicar que o mesmo possui características que viabilizam a robustez. Uma visualização destas características pode ser vista na Figura 3.2

Na seção seguinte será apresentado o arcabouço com características autonômicas, que atende ao conjunto de requisitos especificados, no que diz respeito à sua estrutura, aspectos funcionais e de operação no contexto de redes de computadores.

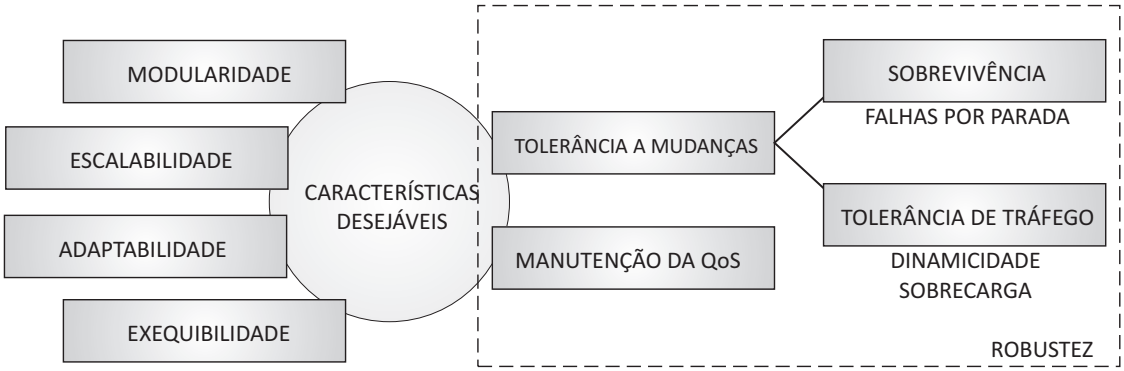


Figura 3.2. Requisitos e características potencialmente decorrentes do arcabouço

3.2 DEFINIÇÃO E ESTRUTURAÇÃO DO ARCABOUÇO

Esta seção define o arcabouço com características autonômicas apresentando a sua estrutura para a gerência de redes. Finalmente, a seção detalha os aspectos mais específicos de modularidade, da implantação e da operação do arcabouço para o foco considerado da gerência da qualidade de serviço em redes.

3.2.1 O Arcabouço Proposto em Relação à Estruturação Geral das Soluções Autonômicas para a Gerência de Redes

No gerenciamento não autonômico a administração dos recursos para a manutenção da satisfação dos usuários e aplicações é realizada pelo administrador da rede que responde diretamente pelas funções intermediárias conforme apresentado na Figura 3.3a. Ferramentas de monitoramento de redes e execução de políticas tem uma maior consolidação, inclusive com as demais ferramentas disponíveis no mundo acadêmico e comercial. Com o auto-gerenciamento, o administrador apenas especifica os requisitos para que o sistema de gerência forneça a solução apropriada, tendo suas atividades de monitoramento, análise, planejamento e execução realizadas por um sistema autonômico o que pode ser observado nas Figuras 3.3a e 3.3b.

As estratégias de implantação da autonomia na gerência são comumente adotadas para fornecerem uma solução que integra as quatro funcionalidades descritas na Figura 3.3b. Conforme visto na Seção 2.2.4 (Aspectos Arquiteturais), nas arquiteturas autonômicas, independentemente da solução específica implementada, existe uma separação em relação às funções de gerência – monitoramento, análise, planejamento e execução.

Com relação ao arcabouço proposto, tem-se uma solução em planos independentes e modulares, onde o plano de decisão concentra as fases de análise e planejamento. Este plano também mantém o requisito de modularidade fazendo com que o arcabouço proposto possa utilizar diferentes algoritmos e/ou estratégias computacionais para a derivação do melhor estado da rede e diferenciando-o das soluções mais clássicas de gerenciamento autonômico.

Para que o arcabouço seja compatível com diferentes sistemas de monitoramento e/ou sis-

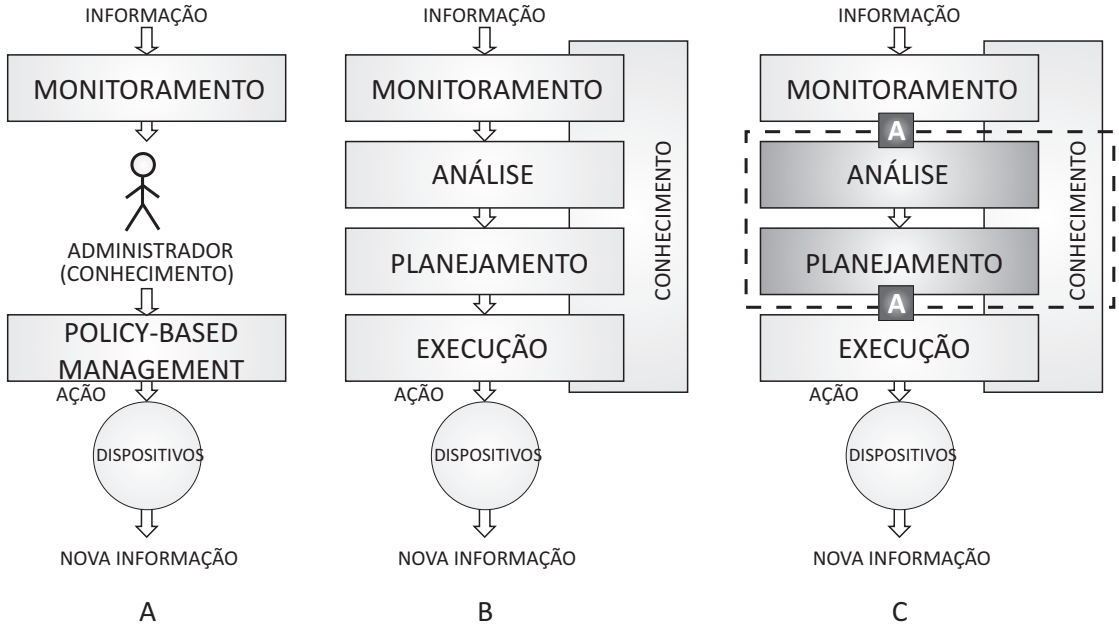


Figura 3.3. Foco de atuação do arcabouço (C) em relação a estruturas clássicas (A) e ao modelo conceitual de gerência autonômica (B)

temas de gerência baseados em políticas torna-se necessário o desenvolvimento de adaptadores entre as fases de monitoramento e análise e as fases de planejamento e execução. Os adaptadores são apenas formatadores dos dados recebidos para a utilização no arcabouço desenvolvido, visando a interoperabilidade do mesmo.

3.2.2 Estrutura do Arcabouço – Uma Visão Baseada em Planos

O arcabouço para a gerência com características autonômicas é definido como uma estrutura com três planos de gerência com características de modularidade como segue (Bezerra; Martins, 2008) (Bezerra; Martins, 2009b) (Figura 3.4):

- Plano de Informação;
- Plano de Decisão;
- Plano de Execução.

O Plano de Informação é o plano responsável pelo recebimento do estado atual dos dispositivos gerenciados (estado da rede), através de ferramentas de monitoração. No arcabouço proposto este plano tem como funcionalidade inerente a conversão das informações de monitoramento para um padrão de formatação baseado em XML[1]. Esta conversão atende o requisito de modularidade e torna o arcabouço mais adaptável, uma vez que a criação de novos conversores

[1] *eXtensible Markup Language é uma recomendação da W3C para gerar linguagens de marcação para necessidades especiais.*

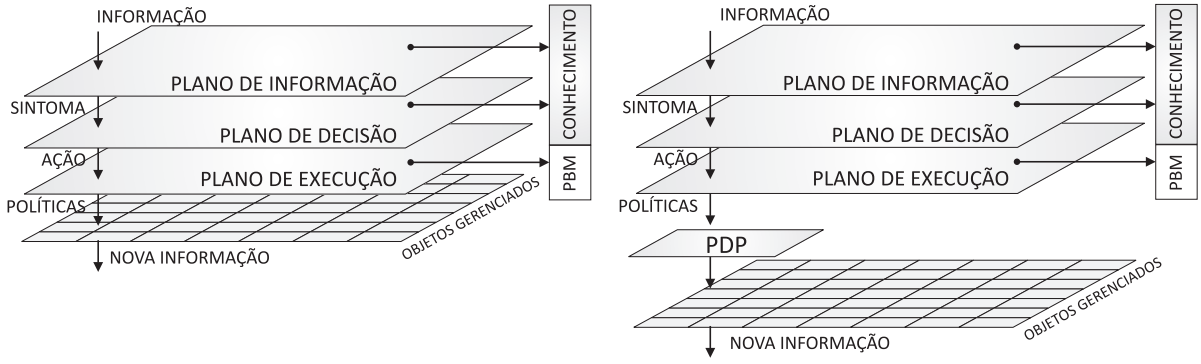


Figura 3.4. Visão geral dos planos de gerência do arcabouço com dois modos de operação do plano de execução sem e com a utilização de Ponto de Decisão de Políticas (PDP – *Policy Decision Point*)

torna-o potencialmente compatível e interoperável com os diversos tipos de monitores (distintas implementações). Assim sendo, o arcabouço permite a interoperabilidade com diferentes fornecedores de serviço que, por sua vez, podem ter estruturas de monitoração distintas.

Outra funcionalidade do plano de informação é a detecção de eventuais anomalias com base no estado atual da rede. De fato, o estado atual da rede é analisado com base na verificação dos níveis de serviços esperados para a manutenção de parâmetros pré-definidos. Na implantação realizada, esta verificação consiste apenas na determinação de anomalias e na quantificação dos níveis de serviço atendidos e em suas respectivas penalidades. Em outras palavras, o plano de informação trabalha com dados de monitoramento normalizados (XML) recebidos do ambiente gerenciado (estado da rede) e retorna os sintomas detectados (em analogia ao corpo humano). A existência de um sintoma indicado nesta fase é um gatilho para a atuação do arcabouço. Em resumo, o plano de informação trata do conhecimento da rede e, fundamentalmente, lida com os dados que definem o estado atual da mesma derivando um eventual sintoma para o plano de decisão.

O plano de execução é o plano responsável pela aplicação das políticas e/ou ações (decisões) geradas pelo plano de decisão do arcabouço. No caso, o plano de execução recebe uma indicação de ação e a aplica (Figura 3.4a) ou gera uma política a ser aplicada nos dispositivos gerenciados através de protocolo ou sistema de aplicação de políticas de gerenciamento (Figura 3.4b). Observa-se que o plano de execução se assemelha a um sistema de Gerência Baseado em Políticas (PBM – *Policy-Based Management*) tradicional, capaz de ler/converter políticas e entregá-las aos dispositivos, seja de forma direta ou através de um Ponto de Decisão de Políticas (PDP – *Policy Decision Point*). No arcabouço proposto é recomendado que a distribuição de políticas seja feita através de um protocolo de difusão de políticas visando a atomicidade das operações de gerência da rede.

A política ou ação aplicada aos objetos gerenciados gera um novo estado da rede a ser avaliado em relação ao sucesso da operação realizada. Dessa forma, é necessária uma nova informação que deve ser analisada pelo arcabouço.

Como comentário final em relação ao plano de execução, um PBM (Bezerra; Martins, 2006) foi adaptado para servir de base para o arcabouço com características autonômicas. Originalmente o PBM foi desenvolvido para administrar todo o ciclo de vida de uma política com ferramentas capazes de criar, editar, converter e ativar políticas de forma integrada e flexível, com o objetivo de garantir uma gerência eficaz de dispositivos e equipamentos. Além disso, O PBM busca também permitir a configuração completa do domínio, oferecer um gerenciador de alertas, dentre outras funções.

Veremos em seguida que o plano de decisão fornece o conhecimento ao processo de gerência com políticas visando à redução da intervenção humana. Vale também ressaltar que o arcabouço proposto é modular no sentido de ter sido especificado de forma a atuar com outros PBMs através de um conversor de políticas, desde que a linguagem de definição de políticas esteja especificada.

O plano de decisão é o ponto chave de contribuições desta tese. Em síntese, ele recebe o sintoma da rede derivado pelo plano de conhecimento, coleta os requisitos definidos pelo administrador para a busca de uma nova solução e, finalmente, executa um "ciclo de processamento"visando a identificação de um novo estado para a rede. Este novo estado de configuração da rede computado gera efetivamente uma ação ou política, que deve ser encaminhada para o plano de execução (Figuras 3.4a e 3.4b). O ciclo de processamento interno do plano de decisão corresponde à computação do novo estado da rede e será detalhado em termos operacionais na seção 3.2.4.

Por definição, o plano de decisão tenta encontrar uma solução ótima, ou seja, o melhor estado possível para a rede dentro de um conjunto de soluções aplicáveis. Obviamente, a qualidade da solução depende dos requisitos definidos pelo gerente e esta questão será discutida na seção 3.5.3 de forma mais objetiva quando da validação do arcabouço proposto.

Em suma, o plano de decisão: (i) concentra as fases de análise e planejamento inerentes aos sistemas autonômicos, (ii) trata os aspectos de conhecimento (*knowledge*) referentes ao estado da rede e seu histórico de soluções e (iii) representa o ponto focal de otimização do arcabouço no sentido da garantia dos requisitos de exequibilidade do tempo de resposta na busca de novas configurações e no tratamento da escalabilidade.

3.2.3 O Plano de Decisão

No contexto da autonomia, as fases de análise e planejamento – agregadas neste trabalho no plano de decisão – são as principais fases no processo de desenvolvimento de sistemas autonômicos. Nestas fases, funções como: compreender o estado atual do sistema em função dos dados do monitoramento e buscar uma solução para otimizar o estado atual são desempenhadas tradicionalmente por administradores na gerência não autonômica. A experiência e a memória dos administradores são essenciais quando não existe autonomia. Dessa maneira, será abordado nesta seção o plano de decisão, o seu motor de busca de soluções integrado a técnica de par-

ticionamento – utilizada para a redução do tempo de resposta na oferta dinâmica de soluções (*on-the-fly*).

O plano de decisão, apresentado Figura 3.5, deve atender os requisitos especificados na Seção 3.1, ao ofertar autonomicamente soluções para o ambiente gerenciado. Para tanto, o plano de decisão é composto de cinco módulos:

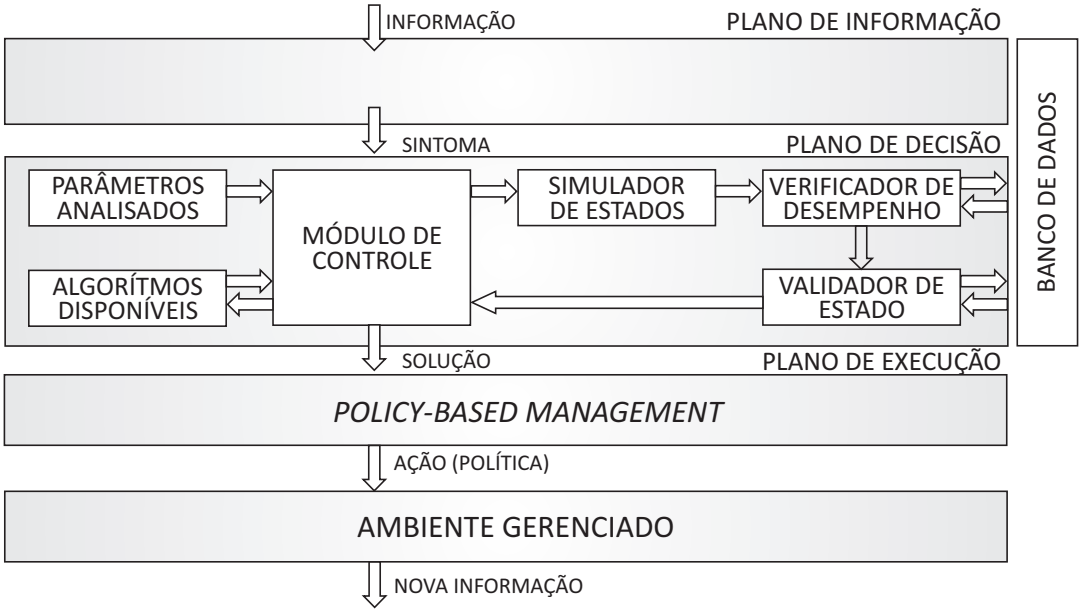


Figura 3.5. Arcabouço para gerência autonômica de redes – detalhamento da camada de decisão

i) Módulo de Controle – Responsável por coordenar as ações entre módulos, base de conhecimento e requisitos especificados pelo administrador. Controla o fluxo de informações entre os módulos deste plano.

ii) Parâmetros Analisados – Este módulo contém os parâmetros que serão levados em consideração na busca de novas soluções. Por exemplo, na disciplina qualidade de serviço os parâmetros largura de banda (*bandwidth*) e atraso (*delay*) podem ser utilizados na especificação do arcabouço.

iii) Algoritmos Disponíveis – Este módulo contém todos os algoritmos que podem ser utilizados na busca de soluções. Por exemplo, para a escolha do menor caminho entre dois nós estão disponíveis os algoritmos de Dijkstra (Dijkstra, 1959), Bellman-Ford (Bellman, 1958) e Jonhson's (Johnson, 1977). Para o cálculo do fluxo máximo entre dois nós quaisquer da rede é utilizado o algoritmo de Goldberg (Goldberg; Tarjan, 1986). Para o particionamento da rede estão disponíveis quatro algoritmos (Newman; Girvan, 2004; Clauset; Newman; Moore, 2004; Newman, 2006; Pons; Latapy, 2006). É importante ressaltar que quaisquer outros algoritmos podem ser utilizados.

iv) Simulador de Estados – Este módulo contém um simulador matemático que calcula a melhor relação (denominado de *tradeoff*) entre a manutenção da qualidade de serviço

e o tempo de resposta em função de parâmetros especificados pelo usuário. Para tal é utilizada a ferramenta R[2] com o pacote igraph[3]. O objetivo deste simulador é ser a plataforma de desenvolvimento para busca e verificação de novas soluções.

v) Validador de Estados – O objetivo deste módulo é filtrar a(s) solução(es) encontrada(s) pelo simulador de estados. Pode ser possível encontrar mais de uma configuração para otimizar os recursos do ambiente gerenciado. Este módulo verificará se todos os requisitos estão sendo atendidos e selecionará qual das soluções possíveis será executada.

Conforme visto anteriormente, o arcabouço apresentado define três planos para a gerência autonômica de redes, onde o plano de decisão precisa oferecer uma nova configuração dinamicamente. Essa busca com restrições temporais implica na necessidade de mecanismos para reduzir o espaço de busca de soluções, visando a oferta de tempo de resposta factível para grandes domínios. No entanto, a redução do espaço de busca implica na restrição da visão global da rede, podendo limitar a qualidade das soluções ofertadas pelo arcabouço. Assim torna-se fundamental encontrar alguma estratégia capaz de minimizar esta restrição, maximizando a qualidade da solução ofertada. Na seção 3.3 será apresentado uma estratégia baseada no conceito de dividir e conquistar que segmenta a rede em domínios em função de informações do perfil de tráfego e topologia.

3.2.4 Ciclo de Atividades do Arcabouço

Para ambientes gerenciados autonomicamente, o ciclo de controle (*control loop*) pode ser caracterizado como uma sequencia de atividades desempenhadas por um sistema autonômico e o seu respectivo fluxo de informações.

Para a gerência autonômica de redes espera-se que o arcabouço procure novas configurações da rede tanto para alcançar uma melhor utilização do seus recursos, quanto para prover respostas a falhas encontradas nos dispositivos. Desta maneira é necessário ter uma visão mais detalhada do ciclo de controle para ambas as situações com foco nas fases de análise e planejamento, uma vez que estas fases compõem o foco deste trabalho. No arcabouço desta tese, o ciclo de controle pode ser visualizado na Figura 3.6.

Observando os possíveis caminhos do ciclo de controle, é possível destacar:

- Para a ocorrência de falhas, o ciclo de controle verifica inicialmente se com a configuração atual a rede consegue manter os requisitos especificados pelo administrador. Caso não consiga, é verificado sob qual configuração o mesmo consegue melhor atender tais necessidades. É importante ressaltar a existência da impossibilidade de encontrar uma solução

[2]R é uma linguagem e ambiente para computação estatística e gráficos que oferece uma ampla variedade de funções estatísticas (modelagem linear e não linear, testes estatísticos clássicos, análise de séries temporais, classificação, clustering, entre outros) e técnicas gráficas, e é altamente extensível. Maiores detalhes vide o Anexo B.

[3]IGRAPH é um pacote de software livre para criação e manipulação de grafos que possui implementações para problemas clássicos de Teoria do Grafos. Maiores detalhes vide o Anexo B.

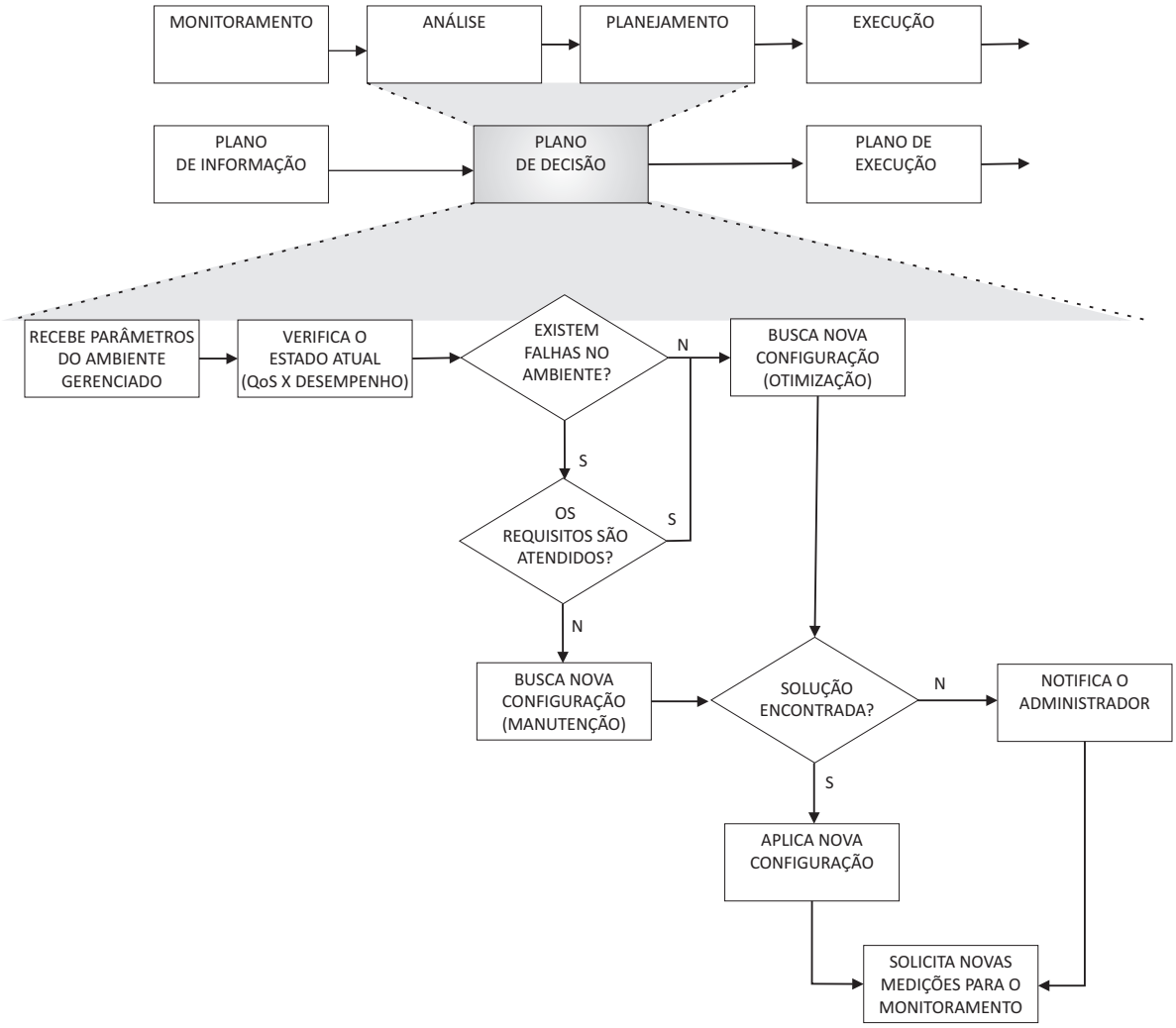


Figura 3.6. Mapeamento do ciclo de controle arquitetural, planos do arcabouço e ciclo de controle do arcabouço

que atenda aos requisitos para uma maior quantidade de falhas ou quando as mesmas se concentram em um nó (ou região). Neste caso, o arcabouço verifica se o administrador aceita a aplicação de uma nova configuração, melhor que o estado atual da rede, mas que não atinge as metas especificadas. É importante ressaltar que o relaxamento dos objetivos especificados só poderá ser realizado mediante a ocorrência de falhas. Se o administrador não aceita tal escolha (se a configuração não atende aos objetivos especificados), ele recebe uma notificação do arcabouço.

- Na inexistência de falhas, o arcabouço busca otimizar os recursos da rede. O processo de busca de soluções é similar, mas durante a busca de uma nova configuração a restrição de tempo não é crítica e o custo operacional de aplicação das alterações, pode ser um parâmetro limitante. Ainda assim o tempo deve ser considerado para que o arcabouço não ofereça uma nova configuração para um estado da rede que não está mais acontecendo.

Para alcançar tais objetivos descritos na Seção 3.1 foi necessária a concepção do plano de

decisão com as seguintes capacidades:

i) Prover uma técnica de particionamento em função da topologia e do perfil de tráfego da rede visando a provisão de soluções em ambientes escaláveis (Seção 3.3).

ii) Incorporar raciocínio e aprendizagem ao arcabouço autonômico permitindo a integração com o particionamento de forma que a busca por soluções atenda aos objetivos especificados, em especial, neste caso, o tempo de resposta e a manutenção da qualidade de serviço (*Performability*), aspectos discutidos na Seção 3.6.

Na Seção 3.3 será apresentada uma estratégia de particionamento para permitir o tratamento da escalabilidade (capacidade i)) e em seguida, na Seção 3.6, tal técnica será integrada ao motor de busca de soluções visando o atendimento da capacidade ii). Ambas capacidades são executadas pelo plano de decisão.

3.3 UTILIZANDO PARTICIONAMENTO PARA TRATAMENTO DA ESCALABILIDADE

Conforme dito anteriormente, quando se trata de gerência autonômica de redes é necessário oferecer suporte dinâmico para pequenos a grandes e complexos domínios, mantendo o tempo de resposta factível. Dessa forma, nota-se que a escalabilidade é um requisito que deve ser considerado (Bezerra; Martins, 2009a). O desempenho no contexto autonômico é um aspecto importante uma vez que é necessário garantir um tempo de resposta viável durante a alocação de recursos. Alocação dinâmica de recursos deve fornecer resultados otimizados com base no estado atual da rede analisando uma base em dados do tráfego coletado e do estado do ambiente gerenciado. Em particular, a escolha de caminhos é um dos fundamentais problemas de otimização em redes e algoritmos para este problema tem sido propostos há algum tempo (Cherkassky; Goldberg; Radzik, 1994). A ideia principal apresentada nesta seção é a utilização do conceito de dividir e conquistar, através de um algoritmo de particionamento que visa a escalabilidade e aspectos de robustez para a gerência de redes.

As grandes redes são altamente complexas de gerenciar, principalmente por conta da natureza dinâmica do roteamento (distribuição da tráfego da matriz, as mudanças de topologia, entre outros) e do tamanho de tais redes (Mortier; Isaacs; Barham, 2005). Como discutido anteriormente, os sistemas de gerência autonômicos (AMS) devem ser capazes de operar em redes com cardinalidades diferentes (nós e ligações) sem restrições topológicas ou de tráfego.

Nesta seção, uma estratégia de particionamento rede é proposta para lidar com este problema. A fim de facilitar a apresentação desta estratégia, será considerado um caso especial do arcabouço autonômico, que consiste na alocação de todos os fluxos (como teste de escalabilidade). Neste quadro o plano de decisão é a entidade responsável que calcula todas as novas definições de rede com características autonômicas (Bezerra; Martins, 2009a). Como tal, a estra-

tégia de particionamento é efetivamente implementada no plano de decisão e a fim de facilitar a discussão, será definido o NPCE (*Network Partitioning Computing Engine*).

Em termos de estratégia de rede de particionamento, o NPCE recebe inicialmente o seguinte conjunto de parâmetros básicos (entradas): a topologia da rede, a matriz de tráfego e os requisitos de rede (tempo de resposta e qualidade de serviço). A estratégia de particionamento, apresentada na Figura 3.7, essencialmente calcula a melhor relação entre o tempo de resposta esperado para encontrar um novo estado de rede e da qualidade do serviço definidos (Figura 3.7). Uma vez que a melhor opção de particionamento é encontrada pelo NPCE, a matriz de tráfego é realocada, a fim de redistribuir os fluxos através da rede configurando um estado de nova rede.

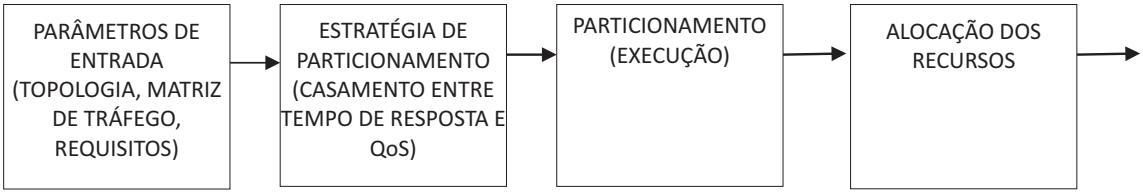
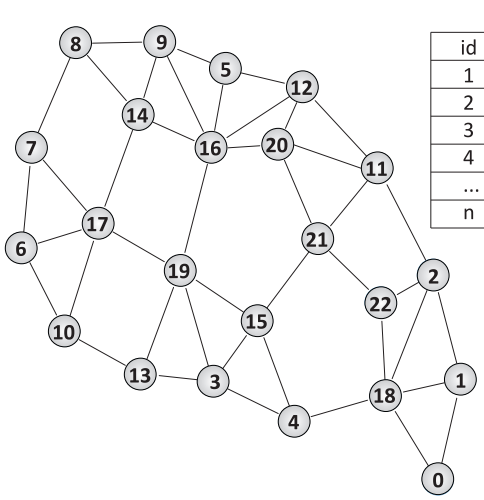


Figura 3.7. Visão geral da estratégia de particionamento – NPCE

Em termos práticos o NPCE recebe uma matriz com todos os fluxos da rede, outra matriz com os possíveis serviços oferecidos e uma matriz de adjacência da topologia da rede. Um simples exemplo ilustrativo pode ser visto na Figura 3.8.



matriz de fluxos

id	origem	destino	serviço
1	9	4	1
2	17	20	4
3	22	10	1
4	3	7	2
...	...	...	...
n	20	9	5

matriz de serviços

id	serviço	banda	...
1	http	128k	
2	voz	32k	
3	vídeo	1024k	
...	...	...	
n	xyz	512k	

matriz de adjacência

vértice	0	1	...	22
0	-	10M	...	-
1	10M	-	...	-
2	-	10M	...	10M
...	...	...	...	...
22	-	-	...	-

Figura 3.8. Exemplo ilustrativo da entrada do arcabouço autonômico

A matriz de adjacência é uma das formas tradicionais de representação de um grafo G qualquer, através de uma matriz n x n denominada como $A(G)$. No exemplo da Figura 3.8, o grafo G (topologia da rede) é não direcionado, simples e com pesos nas arestas (representam a largura dos enlaces), assim basta que as entradas a_{ij} da matriz $A(G)$ contenham o peso se v_i e v_j são adjacentes.

Caso existam dois ou mais enlaces entre nós de redes *A* e *B* quaisquer, o NPCE irá visualizar um único enlace com a largura de banda total sendo o somatório entre todas as larguras de banda dos enlaces que unem os vértices *A* e *B*

O aspecto essencial da estratégia de particionamento é, primeiramente, que a distribuição do tráfego para a rede terá dois componentes novos: o tráfego intra-domínio e o tráfego inter-domínio. O tráfego intra-domínio pode ser entendido como a quantidade de tráfego (fluxos), que permanece dentro dos limites de uma partição de rede recém-calculada (novo segmento de rede). O tráfego inter-domínio corresponde à quantidade de tráfego (fluxos) que necessariamente atravessa as fronteiras do domínio em questão, pois os nós de origem e destino não pertencem ao mesmo domínio.

O segundo aspecto importante da estratégia de particionamento é que o cálculo de redistribuição matriz de tráfego pode agora ser potencialmente executados em paralelo para todo o tráfego intra-domínio e, além disso, esse cálculo será exigido menos recursos computacionais pois o número de parâmetros envolvidos é potencialmente reduzido (nós e links). Como tal, existe um benefício potencial para atingir uma redução no tempo de computação requerida, como previamente indicado, pelos AMSs.

Os aspectos essenciais da estratégia de particionamento são, em resumo, a capacidade potencial para reduzir o tempo de computação para a definição de um novo estado de rede com base na capacidade do segmento da rede e da realocação do tráfego intra e inter-domínios. Este aspecto será considerado nas seções seguintes, em primeiro lugar em termos de seus conceitos analíticos e, posteriormente, mostrando detalhes adicionais da sua efetiva implementação.

3.3.1 Conceitos Básicos para o Partionamento da Rede – Uma Visão Analítica

Definição 3.3.1 *Dado um conjunto V de vértices (nós de rede) e um conjunto E de arestas (enlaces de redes), uma rede de computadores pode ser definida como um grafo $G = (V,E)$ conexo e não direcionado de forma que $E \subseteq |V^2|$, onde cada elemento de E corresponde a uma tupla com dois elementos de V*[4]*.*

Nesta tese serão utilizadas as seguintes notações:

- *A ordem de um grafo G é $|V|$, que corresponde ao número de vértices do grafo. Neste artigo, $|V| = n$. Por exemplo, na Figura 3.9 a quantidade de vértices (nós da rede) é igual a 23 ($n = 23$).*

- *O número de arestas de um grafo é $||G||$. Por exemplo, na Figura 3.9, a quantidade de arestas (enlaces de rede) é igual $||G|| = 44$.*

- *O grau de um vértice é o número de arestas conectadas a ele. O grau do vértice três tem valor quatro.*

[4]Tuplas do tipo (e_i, e_i) não são consideradas e (e_i, e_j) é igual a (e_j, e_i), pois o grafo é não direcionado.

- *A densidade de um grafo é a relação entre o número de arestas do grafo e o total de arestas de um grafo completo com n vértices ($||G||/||K_n||$). Por exemplo, na Figura 3.9 a densidade é igual a* $0,174$*, ou seja existem 17,4 % dos enlaces possíveis entre os nós de rede.*

Um simples exemplo de particionamento de rede, para $n = 23$, com três partições ($d = 3$) pode ser visualizado na Figura 3.9.

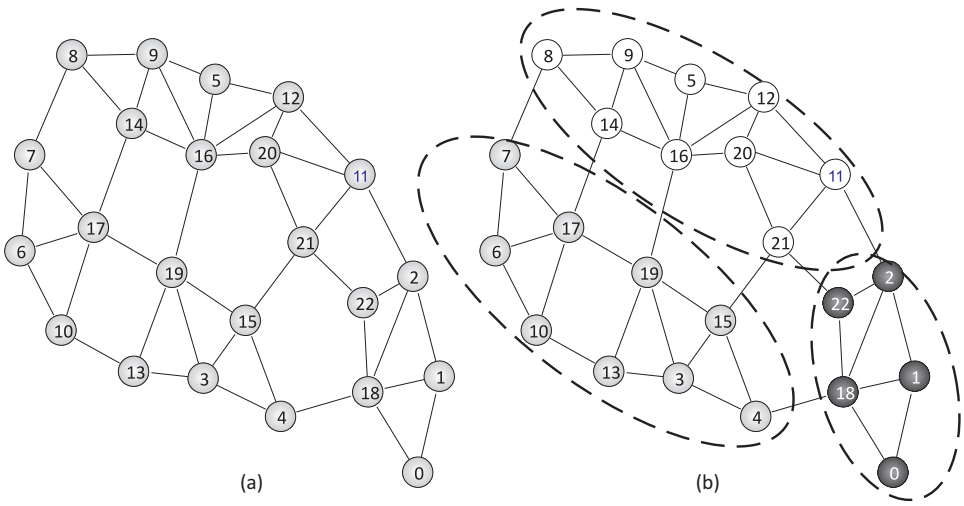


Figura 3.9. Exemplo de particionamento com três partições

Definição 3.3.2 *Dada uma rede qualquer representada por um grafo G, quando a estratégia de particionamento é aplicada gera um conjunto de partições ($d \geq 2$) e consequentemente o tráfego da rede pode ser subdividido em dois grupos disjuntos: tráfego entre partições e tráfego dentro das partições. Nesta tese descreveremos tráfego intra-domínios como sendo todo o tráfego onde os nós de origem e destino pertencem ao mesma partição. Em contrapartida, quando os nós de origem e destino pertencem a partições distintas será denominado tráfego entre tais nós será considerado como tráfego inter-domínios.*

Por exemplo, uma rede de computadores é representada pelo grafo G com $n = (A,B,C,D,E,F,G,H,I,K,J,L,M,N,O,P,Q)$ nós e $tg = (A,B,C,D,E,F)$ geradores de tráfego[5], sua matriz de tráfego é representada por uma matriz quadrada M de dimensão 6*x*6 (*tgxtg*). Neste exemplo, assumimos duas possibilidade de particionamento *A* e *B*, onde o número de partições são respectivamente, dois ($A = p_{A1}, p_{A2}$) e três ($B = p_{B1}, p_{B2}, p_{B3}$), representado da seguinte forma:

$p_{A1} = (A,B,G,H,M)$

$p_{A2} = (C,D,E,F,I,J,L,N,O,P,Q)$

$p_{B1} = (A,B,G,H,M)$

$p_{B2} = (C,D,I,J,O)$

[5]Neste exemplo didático apenas alguns nós são geradores de tráfego. Não existem restrições quanto a quantidade ou percentual de geradores entre os nós

$$p_{B3} = (E, F, K, L, P, Q)$$

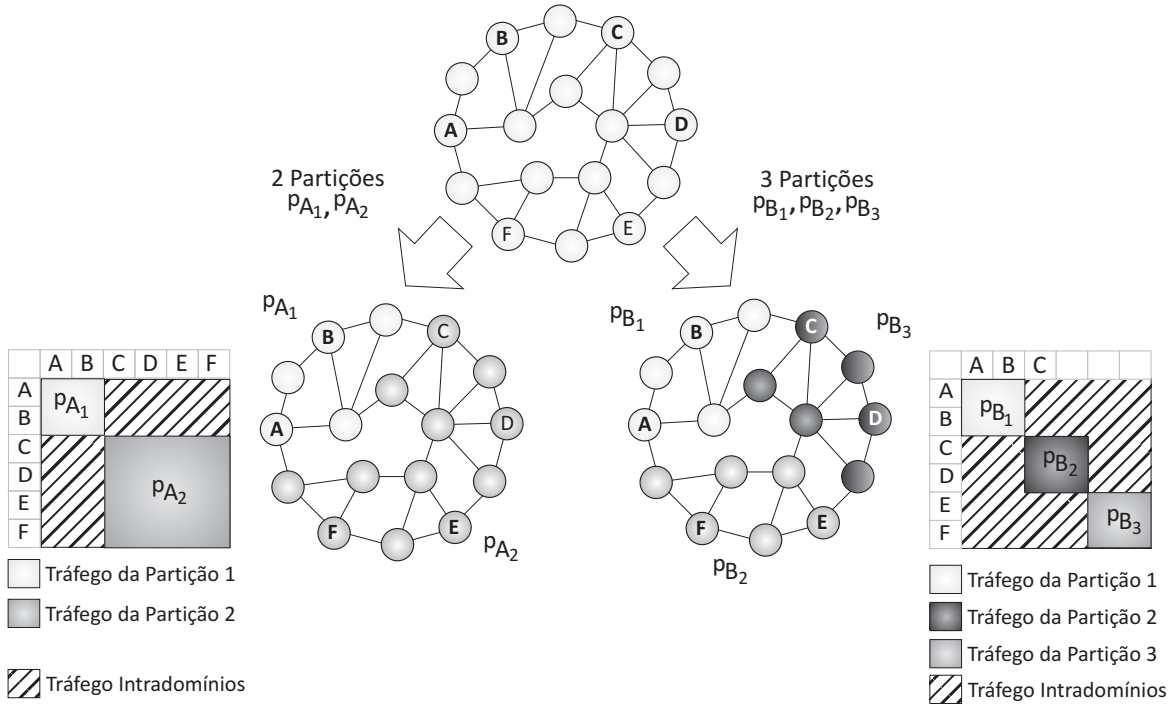


Figura 3.10. A relação entre o tráfego intra-domínio das partições A e B

Definição 3.3.3 *Dada uma rede qualquer representada por um grafo G, quando a estratégia de particionamento for aplicada o custo computacional total de alocação dos fluxos será representado por T'. Caso a estratégia não seja utilizada, ou seja na alocação convencional, a representação será através da variável T.*

Na seção seguinte, as Definições 3.3.1, 3.3.2 e 3.3.3 serão utilizadas para comparação entre os tempos de execução, são denominados T e T' como sendo, respectivamente, o tempo total de alocação dos fluxos com e sem o algoritmo de particionamento, com o objetivo mostrar a redução de tempo de alocação dos fluxos em relação a alocação de fluxos sem a abordagem do particionamento. Não serão considerados outros requisitos além do tempo de resposta.

3.4 ANÁLISE MATEMÁTICA DO PARTICIONAMENTO

Definição 3.4.1 *Considere uma rede com g geradores/consumidores de tráfego, f fluxos e com custo de alocação de fluxos c. O custo total de alocação de fluxos desta rede pode ser escrito como:*

$$T = \frac{g(g-1)}{2} cf \tag{3.1}$$

Se o algoritmo de particionamento for utilizado, o custo total é dividido entre o custo de alocação de fluxos intra-domínios, quando os ($g_{gerador}$[6] e $g_{destino}$[7] pertencem ao mesmo domínio, e o custo de alocação de fluxos inter-domínios, quando os $g_{gerador}$ e $g_{destino}$ não pertencem ao mesmo domínio.

Definição 3.4.2 *O custo de alocação com o particionamento de uma rede representada por* $G = (V,E)$ *pode ser escrito da seguinte forma*

$$T' = T'_{inter} + T'_{intra} \quad = \quad \frac{g_i(g_i-1)}{2} c\, f_{inter} + \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i\, f_{intra} \tag{3.2}$$

onde para cada domínio d, $g \geq 2$, $f \geq 1$ *e* $c \geq 1$.

Considere as equações 3.1 e 3.2, é possível mostrar que a alocação de tempo é menor quando é utilizado o algoritmo de particionamento.

Lema 3.4.1 *O tempo de alocação quando o algoritmo de particionamento é utilizado é sempre menor que o tempo de alocação quando o algoritmo de alocação sem particionamento é adotado, independentemente do algoritmo de alocação, número ou tamanho das partições.*

Prova É necessário provar que $T \geq T'$. Em outras palavras,

$$\frac{g(g-1)}{2} c\, f_t \geq \frac{g_i(g_i-1)}{2} c\, f_{inter} + \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i\, f_{intra} \tag{3.3}$$

onde $f_t = f_{inter} + f_{intra}$ and $f_{intra} = \sum_{i=1}^{d} f_i$. Portanto,

$$\frac{g(g-1)}{2} c \geq \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i \quad = \quad \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} * \sum_{i=1}^{d} c_i \geq \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i \tag{3.4}$$

Para simplificar a notação foi assumido que $a = \frac{g_i(g_i-1)}{2}$. Dessa forma,

$$\begin{aligned} \sum_{i=1}^{d} a_i \sum_{i=1}^{d} c_i \geq \sum_{i=1}^{d} a_i c_i \quad &= \quad (c_1 + \ldots + c_d)(a_1 + \ldots + a_d) \geq (a_1 c_1 + \ldots + a_d c_d) \\ &= \quad (a_2 c_1 + \ldots + a_1 c_n + \ldots + a_n c_1 + \ldots + a_d c_{d-1}) \geq 0 \end{aligned} \tag{3.5}$$

$\forall i \in [1,d]$ and $a_i, c_i \geq 1$,
para $a = \frac{g_i(g_i-1)}{2} \wedge a \geq 1 \wedge g \geq 0 \Rightarrow g \geq 2$ (de acordo com a Equação 3.2)

∎

[6]Nó da rede gerador de tráfego
[7]Nó da rede consumidor de tráfego. No escopo desta tese todos os nós geradores são consumidores.

Lema 3.4.2 *Quanto maior a diferença da relação do tráfego intra-domínio e inter-domínio, menor o tempo de execução do algoritmo proposto.*

Prova Considere uma rede com g geradores/consumidores de tráfego, f fluxos e com custo de alocação de fluxos c. Também considere que ambos os tráfegos intra-domínio e inter-domínio sejam maiores que zero ($f_{inter} > 0$ and $f_{intra} > 0$) e que existem dois cenários distintos A e B, onde a razão do trafego intra-domínio/inter-domínio é maior em A. O objetivo é mostrar que $T'_B > T'_A$.

$$T'_{B_{inter}} + T'_{B_{intra}} > T'_{A_{inter}} + T'_{A_{intra}}$$

$$T'_{B_{inter}} - T'_{A_{inter}} > T'_{A_{intra}} - T'_{B_{intra}} \tag{3.6}$$

$$\frac{g_i(g_i-1)}{2} c f_{B_{inter}} - \frac{g_i(g_i-1)}{2} c f_{A_{inter}} \geq \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i f_{A_{intra}} - \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i f_{B_{intra}} \tag{3.7}$$

$$\frac{g_i(g_i-1)}{2} c (f_{B_{inter}} - f_{A_{inter}}) \geq \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i (f_{A_{intra} - f_{B_{intra}}}) \tag{3.8}$$

onde:
$fA = fb \rightarrow fA_{inter} + fA_{intra} = fB_{inter} + fB_{intra} \rightarrow fB_{inter} - fA_{inter} = fA_{intra} - fB_{intra}$ e c é fixo[8]. Dessa forma,

$$\frac{g_i(g_i-1)}{2} \geq \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2}, \forall g > 1 \tag{3.9}$$

∎

Lema 3.4.3 *Se a quantidade de fluxos é maior, maior será a diferença absoluta entre o tempo total de alocação sem particionamento* **T** *e o tempo total de alocação com particionamento* **T′**, *para qualquer quantidade d de partições.*

Prova Considere uma rede com g geradores/consumidores de tráfego, f fluxos e com custo de alocação de fluxos c. Também considere dois cenários distintos A e B, onde $f_A > f_B$. O objetivo é mostrar que:

$$T_A - T'_A > T_B - T'_B \tag{3.10}$$

$$\frac{g_i(g_i-1)}{2} c f_A - \left[\frac{g_i(g_i-1)}{2} c f_{A_{inter}} + T'_{A_{intra}} \right] \geq \frac{g_i(g_i-1)}{2} c f_B - \left[\frac{g_i(g_i-1)}{2} c f_{B_{inter}} + T'_{B_{intra}} \right] \tag{3.11}$$

$$\frac{g_i(g_i-1)}{2} c f_{A_{intra}} - T'_{A_{intra}} \geq \frac{g_i(g_i-1)}{2} c f_{B_{intra}} - T'_{B_{intra}} \tag{3.12}$$

$$\frac{g_i(g_i-1)}{2} c f_{A_{intra}} - \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i f_{A_{intra}} \geq \frac{g_i(g_i-1)}{2} c f_{B_{intra}} - \sum_{i=1}^{d} \frac{g_i(g_i-1)}{2} c_i f_{B_{intra}} \tag{3.13}$$

[8] Indica que o algoritmo de alocação utilizado será o mesmo para todos os fluxos.

$$f_{A_{intra}} > f_{B_{intra}} \tag{3.14}$$

∎

3.5 ASPECTOS DA ESTRATÉGIA DE PARTICIONAMENTO BASEADA NA DUALIDADE ENTRE PERFIL DE TRÁFEGO E DENSIDADE TOPOLÓGICA DA REDE

Uma rede pode ser visualizada como um grafo conexo sem *loops*[9]. Particionamento de grafos é um problema matemático que consiste em dividir um grafo em subgrafos, de forma que os subgrafos tenham alguma característica em comum entre eles, como por exemplo o tamanho (Newman, 2010). A escolha subgrafos de mesmo tamanho não parece ser a melhor solução para redes de computadores, pois desconsidera informações relevantes como topologia e o tráfego da rede. A estratégia proposta particiona um grafo *G* (rede de computadores) em partições (domínios) disjuntos, para que a alocação de recursos seja realizada de forma concomitante em todas as partições (tráfego intra-domínio[10]). Entretanto existe também um tráfego entre nós de diferentes domínios (tráfego inter-domínios [11] vistos na Figura 3.10.

Conforme visto na Figura 3.7, a estratégia utilizada recebe um grafo *G*, uma matriz de tráfego *M* e uma lista de todos os fluxos ativos na rede *F* e escolhe qual a melhor composição de partições, que minimiza o tempo de resposta e maximiza a manutenção da qualidade de serviço. A ideia básica da estratégia NPCE é encontrar um casamento adequado (*match*) entre o perfil de tráfego e a escolha das partições.

Para tal, a estratégia de particionamento implementada pelo NPCE (*Network Partitioning Computing Engine*) tem que ir além da simples utilização de um algoritmo de segmentação da rede e, efetivamente, está estruturado em uma seqüência de etapas, conforme indicado a seguir:

i) O NPCE executa inicialmente o algoritmo de particionamento (a ser discutido na Seção 3.5.1) gerando um conjunto de possíveis partições da rede ($d \geq 2$) de diferentes tamanhos ocasionando distintos padrões de tráfego intra e inter-domínios.

ii) O NPCE avalia o custo algoritmo de alocação para a matriz de tráfego de entrada (perfil de tráfego de rede atual) para cada particionamento obtido independentemente (a ser discutido na Seção 3.5.2)

iii) O NPCE avalia a melhor relação entre o tempo de resposta e os requisitos de qualidade de serviço (QoS) definidos pelo administrador da rede (a ser discutido na Seção 3.5.3)

O resultado esperado, como discutido anteriormente, é a redução do tempo total computacional necessário para uma nova definição de um novo estado da rede (configuração), preservando a qualidade do serviço. As seções seguintes descrevem cada passo específico associado com a implementação da estratégia de particionamento – NPCE.

[9] Ou seja, arestas do tipo (e_i, e_i)

[10] Este tráfego é caracterizado quando a origem e o destino pertencem ao mesmo domínio

[11] Quando nó de origem e o nó de destino não pertencem a mesma partição

3.5.1 A Operação do NPCE – Fase 01 – Tratando a Escalabilidade com o Particionamento

Um parâmetro de entrada para o algoritmo de particionamento é um grafo conexo qualquer de topologia *G*. Em geral, encontrar uma solução exata para uma partição de grafos é considerado um problema NP-completo, tornando difícil tal tarefa para grandes domínios de redes de computadores. Muitas áreas, como redes sociais, web gráficos, ecossistemas e outros, usam algoritmos de particionamento de gráficos para encontrar propriedades comuns ou de relações entre os vértices de acordo com suas necessidades específicas.

Considerando o cenário de gerenciamento de rede adotado (qualidade de serviço), o princípio geral de particionamento é, basicamente, para dividir os nós da rede (Grafo *G*) em subgrafos com a maior densidade possível. Ao olhar para subgrafos de maior densidade, obtém-se um conjunto de vértices em que a relação entre eles é mais forte, isto é, o número de arestas por subgrafo é substancial. Sendo assim, no contexto de rede de computadores uma maior densidade por partição (subgrafos) representa, potencialmente, um maior número de caminhos entre um conjunto de vértices, um maior número de caminhos de roteamento alocáveis (como LSPs) em cada partição. É importante mencionar que, quando as arestas (*links*) não pertencem a qualquer partição, elas correspondem a enlaces de comunicação inter-domínios.

Os critérios adotados para a estratégia de particionamento foram o tempo de execução, a densidade das partições e o desvio padrão médio do tamanho das partições (Bezerra; Martins, 2009a). Um detalhamento destes critérios é apresentado em detalhes na na Seção 3.5.1.1. O algoritmo *edgebetweenness* apresentou um melhor comportamento – considerando os critérios adotados – na geração de partições da rede gerenciada.

Desta forma nesta fase o NPCE produz grupos de partições com diferentes cardinalidades em função da topologia da rede. Cada um desses grupos de particionamento contém *p* distintas partições e será identificado neste texto como um novo NPT (*Network Partitioned Topology*).

Na seção seguinte será indicado quais os critérios utilizados para a seleção de um algoritmo com o objetivo descrito que melhor se adapte ao contexto utilizado.

3.5.1.1 Seleção do Algoritmo de Particionamento Baseado em Densidade

Muitas áreas que também possuem alta complexidade – como redes sociais (*social networks*), *web graph* e ecossistemas – utilizam algoritmos de particionamento em busca de propriedades entre os vértices e/ou arestas de um grafo. O particionamento utilizado nestas áreas é realizado com base na busca de comunidades (*community structures*) (Newman, 2010).

Algoritmos para busca de comunidades tem sido amplamente estudados e e tal estratégia está intimamente relacionado com a ideia de particionamento em teoria dos grafos (Newman; Girvan, 2004; Clauset; Newman; Moore, 2004; Newman, 2006; Pons; Latapy, 2006). Dentro do contexto de redes de computadores o objetivo é desenvolver um algoritmo dinâmico capaz de buscar um grupo de *d* domínios de alta densidade com *v* vértices e *e* arestas. Ao criar domínios menores,

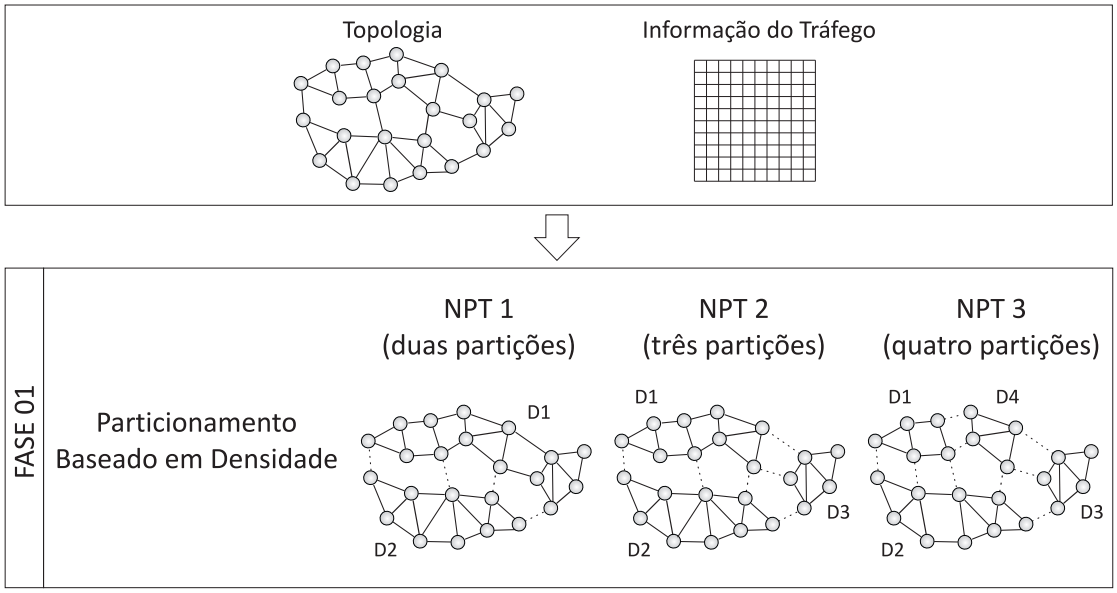


Figura 3.11. Resultados gerados pela primeira fase

um subconjunto de arestas não pertencerá aos domínios – servindo de interligação entre eles – de modo que tais arestas que não pertençam aos domínios são as que tornam os domínios menos densos. A seguir serão apresentados conceitos matemáticos para contextualização dos tópicos desta seção.

Ao procurar domínios de maior densidade em um grafo *G*, obtém-se um conjunto de vértices, onde o relacionamento entre eles é mais forte, ou seja, o número de arestas existentes é maior para um subconjunto de vértices de *G*. No contexto de redes de computadores, isso representa um maior número de caminhos entre este conjunto de vértices. Estes subconjuntos de vértices de maior densidade serão denominados aqui domínios. É importante mencionar que as arestas (enlaces) não pertencentes a nenhum domínio são enlaces que permitem a comunicação inter-domínios. Assim, o algoritmo usado originalmente para encontrar comunidades, será efetivamente utilizado em nosso contexto para identificar os vértices (nós) dos domínios de rede. O algoritmo recebe um grafo *G* (topologia de rede) e o número necessário de domínios *d* e cria um novo gráfico com a interligação das comunidades (domínios).

Alguns algoritmos clássicos para a busca de comunidades tem sido utilizados amplamente em áreas como redes sociais, ecossistemas e sociologia. Desta classe de algoritmos, quatro algoritmos clássicos foram testados: *walktrap* (Pons; Latapy, 2006), *fastgreedy* (Clauset; Newman; Moore, 2004), *eigenvector* (Newman, 2006) e *edgebetweenness* (Newman; Girvan, 2004). Os testes para a escolha do algoritmo utilizaram três métricas: número máximo de arestas pertencentes aos domínios tempo médio de execução, menor desvio padrão da cardinalidade, *n*, dos domínios (subgrafos) gerados. Vale ressaltar que o número máximo de arestas pertencentes aos domínios é o parâmetro que permitirá a criação de subgrafos mais densos e por isso é considerado um parâmetro relevantes. Já a consideração desvio médio padrão da cardinalidade dos domínios visa encontrar um algoritmo que encontre subgrafos com quantidade de vértices e arestas similares.

Os testes foram realizados com grafos de cardinalidade $n = 10, 50, 100, 250$ gerados aleatoriamente. Para a geração dos grafos foram aplicadas as seguintes restrições: grau mínimo dos vértices iguais a dois – visando evitar redes desconexas e/ou com roteamento trivial – grafos não completos e, conforme a definição 3.3.1, que não possuam ciclos (*loops*). Os resultados dos testes podem ser vistos na Tabela 3.1

Tabela 3.1. Resultados dos testes para escolha do algoritmo de particionamento, considerando as métricas especificadas dentro do contexto de redes de computadores. Melhores escolhas estão indicadas em negrito. (Bezerra; Martins, 2009a)

Parametros	tempo de execução			Arestas não usadas			desvio padrão médio		
	segundos			%					
Algoritmos	10	50	100	10	50	100	50	100	250
edgebetweenness $O(e^2v)$	0,141	**0,172**	0,872	20,8	**20,2**	**16,6**	**2,05**	**6,83**	12,09
fastgreedy $O(v\log^2 v)$	**0,125**	0,188	1,106	20,8	**20,2**	19,7	4,27	7,06	14,55
walktrap $O(ev^2)$	0,126	0,181	**0,186**	20,8	22,6	18,3	3,90	11,98	**11,72**

O algoritmo *edgebetweenness* apresentou um melhor comportamento perante os itens especificados, tempo de execução, arestas não usadas e desvio padrão médio que corresponde a um parâmetro relevante pois permite a escolha de um conjunto e domínios com baixa variação de cardinalidade, visando tempo de alocação próximos. Poresta razaão, o algoritmo *edgebetweenness* foi escolhido como base para o particionamento visando a escalabilidade(Bezerra; Martins, 2009a). Resumidamente, o particionamento executado algoritmo *edgebetweenness* resultou em um número máximo de arestas pertencentes aos domínios gerados e o mesmo apresentou um tempo de execução satisfatório. Além disso, a variação da cardinalidade subgrafos foi o menor. Vale ressaltar que o algoritmo *walktrap* não possui um desempenho estável – porque é não deterministicо e utiliza caminhos aleatórios. Já o algoritmo *eigenvector* não teve os resultados apresentados na Tabela 3.1por não conseguir gerar domínios em algumas situações.

A ideia do algoritmo *edgebetweenness*, utilizado originalmente para a detecção de comunidades, é baseada na probabilidade de que as arestas conectando domínios separados tenham pesos altos porque todos os caminhos mais curtos de domínio para domínio, deve percorrê-las. Então, se forem removidas gradualmente as arestas de maior peso, teremos um mapa hierárquico, uma árvore com raízes, chamada de dendrograma[12].

Para encontrar os domínios, o algoritmo *edgebetweenness* segue os passos descritos a seguir (Newman; Girvan, 2004). Na figura 3.9 pode ser visto um exemplo de execução do algoritmo para três partições.

[12]Um Dendrograma (dendro = árvore) é um tipo específico de diagrama que organiza determinados fatores e variáveis. Resulta de uma análise estatística de determinados dados, em que se emprega um método quantitativo que leva a agrupamentos e à sua ordenação hierárquica ascendente - o que em termos gráficos se assemelha aos ramos de uma árvore que se vão dividindo noutros sucessivamente.

i) Calcula os pesos (*betweenness*) para todas as arestas do grafo;

ii) Procura a aresta de maior peso e remove do grafo;

iii) Recalcula os pesos (*betweenness*) para todas as outras arestas;

iv) Repete o item ii, até que não existam mais arestas.

3.5.2 A Operação do NPCE – Fase 02 – Cálculo do Custo Computacional

Neste passo é calculado o custo total para alocação dos fluxos de rede em função dos *n* particionamentos encontrados, ou seja, determina para cada partição qual o custo médio para busca de caminhos. Para tanto, as seguintes tarefas são executadas:

i) Tarefa 01 - Para cada particionamento p $(2 < |p| < n)$ é calculado o percentual tráfego para cada domínio em função do total de tráfego e a quantidade de fluxos por domínio é contabilizada. Em seguida, também para cada particionamento, é calculado o tráfego total inter-domínios e contabilizada a quantidade de fluxos intra-domínios.

ii) Tarefa 02 - O algoritmo do caminho mais curto é selecionado de acordo com a topologia da rede para a obtenção de uma melhor desempenho. Existem diversos algoritmos disponíveis na literatura (Pióro; Medhi, 2004), no arcabouço estão disponíveis os algoritmos de Dijkstra (Dijkstra, 1959) – $O(|e| + |v| * log|v|)$, Bellman-Ford (Bellman, 1958) – $O(|e| * |v|)$ e Jonhson's (Johnson, 1977) – $O(|v|^2 * log|v| + |v| * |e|)$. É importante ressaltar que outros algoritmos podem ser adicionados, viabilizando a modularidade do arcabouço apresentado neste documento.

iii) Tarefa 03 - É calculado o percentual do custo de alocação de recursos com o particionamento em função da solução sem particionamento, denominado de δ. que representa a soma do custo intra-domínios (C_{intra}) e inter-domínios (C_{inter}) – para todo particionamento p $(2 < |p| < n)$ – dividido pelo o custo de alocação de recursos sem particionamento C. O valor de δ pode ser visto na Equação 3.15.

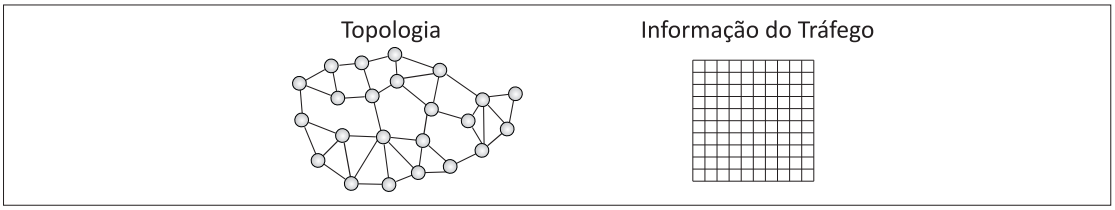
$$\delta = \frac{C_{intra} + C_{inter}}{C} \mid C_{intra} = \sum (C_i * f_i), \tag{3.15}$$

onde:

- C_i – custo estimado para um fluxo calculado na Tarefa 02
- f_i – total de fluxos da partição calculado na Tarefa 01

Ao final desta etapa o NPCE retorna o custo computacional para a alocação de todos os fluxos da rede em função das possíveis NPTs encontradas. Um exemplo pode ser visto na Figura 3.12, onde é mostrado para a NPT com dois domínios de gerência que 57%

do tráfego está concentrado entre os nós do domínio D1, 26% no domínio D2 e 18% do tráfego é entre nós dos domínio D1 e D2.



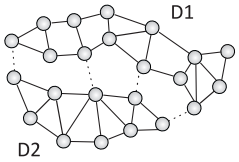
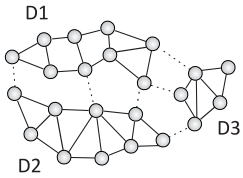
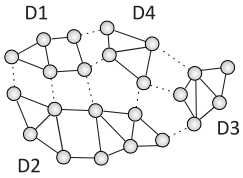
		NPT 1 (duas partições)			NPT 2 (três partições)				NPT 3 (quatro partições)				
FASE 01	Particionamento Baseado em Densidade												
FASE 02	TAREFA 01 Cálculo do Custo Computacional	D1 56%	D2 26%	INTERDOMAIN (I) 18%	D1 32%	D2 26%	D3 15%	I 27%	D1 15%	D2 26%	D3 15%	D4 14%	I 30%
	TAREFA 02 Seleção do Algoritmo do Menor Caminho	Dijsktra			Dijsktra				Dijsktra				
	TAREFA 03 Cálculo do Tempo de Execução Relativo	48,23%			36,77%				38,61%				

Figura 3.12. Visualização do resultado gerado na fase 02 – cálculo do custo computacional

3.5.3 A Operação do NPCE – Fase 03 – Análise da Melhor Relação)

Nesta terceira fase, a melhor relação entre o tempo total de execução dos possíveis particionamentos (Fase 3) e o percentual do somatório do fluxo máximo, que corresponde a razão entre o somatório de todos os fluxos máximos com particionamento e o somatório de todos os fluxos máximos sem particionamento, é calculada.

Além disso, para a definição do problema do fluxo máximo é necessário o entendimento de capacidade, fluxo e o valor de um fluxo que são vistos a seguir.

Definição 3.5.1 *Dado $G = (V,E)$ um grafo conexo e não direcionado com V_1 e $V_2 \in V$ sendo a origem e o destino de um fluxo de rede, respectivamente.*

- *A capacidade de uma aresta é um mapeamento $c : E \to \Re_+$ denotado por $c_{v_1v_2}$ ou $c(v_1,v_2)$, que representa o valor máximo de um fluxo que pode passar em uma aresta (enlace).*
- *Um fluxo é um mapeamento $f : E \to \Re_+$ denotado por $f_{v_1v_2}$ ou $f(v_1,v_2)$, sujeito as seguintes restrições:*

i) $f(v_1, v_2) \leq c(v_1, v_2)$*, pois um fluxo não pode superar a capacidade de uma aresta (enlace)*

ii) $\sum v_1 : (v_1, v_2) \in E_f(v_1, v_2) = \sum v_2 : (v_2, v_1) \in E_f(v_2, v_1)$*, ou seja, corresponde a lei de conservação dos fluxos. Para todo fluxo* f *de origem e* v_1 *e destino em* v_2 *a soma dos fluxos de saída é igual a soma dos fluxos de entrada. Esta é uma adaptação da lei de conservação dos fluxos, uma vez que em redes de computadores geralmente não existem nós apenas geradores ou nós apenas receptores uma vez que as comunicações geralmente são full-duplex.*

- *O valor de um fluxo é definido por* $|f|$*. Em redes de computadores o valor do fluxo corresponde a largura de banda especificada para um serviço esperado pelo arcabouço.*

Assim, o problema do fluxo máximo é maximizar $|f|$*, ou seja, encontrar um conjunto de rotas entre* v_1 *e* v_2*, visando a maior soma de capacidades possível. Nesta tese o fluxo máximo entre dois vértices é calculado com base no algoritmo de Goldberg (Goldberg; Tarjan, 1986) – com complexidade de* $O(|v|^3)$ *– que satisfaz a duas propriedades:*

i) para cada aresta de G o fluxo não pode ser maior que a capacidade da aresta (a capacidade é um parâmetro ou atributo da aresta),

ii) para cada vértice, exceto os vértices de origem e destino, o fluxo de entrada é igual ao fluxo de saída. O fluxo máximo é o somatório de todos os fluxos de entrada em uma aresta, sem restrições de direção.

Assim o somatório do fluxo máximo corresponde a soma de todos os fluxos máximos – calculados individualmente – para cada par de vértices (v_1 , v_2) $\in V$. A representação pode ser realizada utilizando uma matriz quadrada VxV e o somatório corresponde a soma de todas as células desta matriz. O parâmetro de qualidade definido para esta tese é a perda percentual do fluxo máximo que corresponde a razão entre o somatório do fluxo máximo com e sem a utilização do particionamento para todos os nós (vértices), que pertencem a uma mesma NPT. Um exemplo ilustrativo e simples destes parâmetros pode ser visualizado na Figura 3.13.

Observa-se que o fluxo máximo entre os nós B e C ($|f_{(B,C)}|$), que pertencem ao domínio de gerência D1, tem valor igual três sem a utilização do particionamento. Quando o particionamento é utilizado, o valor do fluxo máximo entre os nós citados é reduzido para dois, visto que os caminhos $B \rightarrow D \rightarrow E \rightarrow C$ e $B \rightarrow D \rightarrow F \rightarrow E \rightarrow C$ não podem ser mais utilizados pois os nós D, E e F, agora pertencem a partição 2. O valores de $|f_{(B,C)}|$ com e sem particionamento podem ser visualizados na linha dois e coluna três das matrizes da Figura 3.13). O mesmo problema ocorre com os vértices D e E e pode ser visualizado na linha quatro, coluna cinco.

Esta redução do fluxo máximo não ocorre para todos os pares de vértices de uma mesma partição. Para os pares (A, B) e (A, C) do domínio D1 e (D, F) e (E, F) do domínio D2, o fluxo

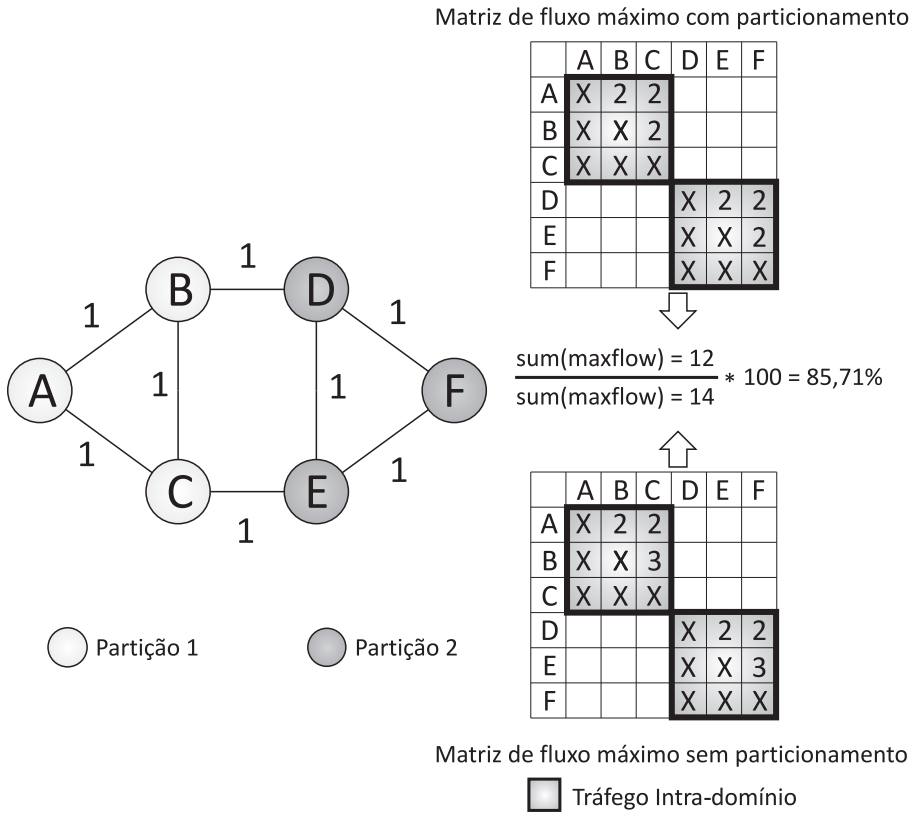


Figura 3.13. Cálculo da perda do somatório do fluxo máximo

máximo é igual a dois ($|f| = 2$) com ou sem o a utilização da estratégia de particionamento. Note que $|f|$ não será maior que dois, uma vez que os vértices A e F tem grau dois ($|d| = 2$).

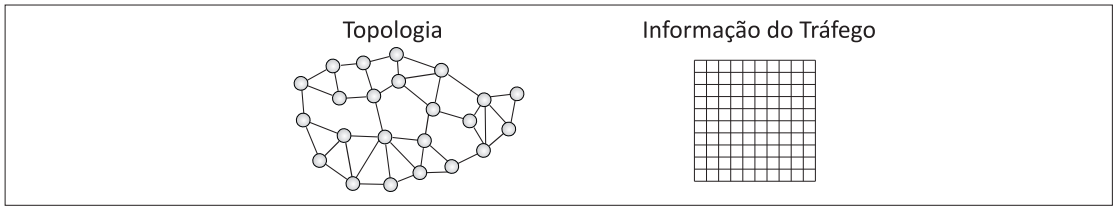
Em termos práticos, o administrador pode especificar a quantidade percentual destes caminhos perdidos em função da aplicação da estratégia de particionamento. Caso o administrador especifique o valor de 90% como o limite de qualidade, o particionamento apresentado na Figura 3.13 não seria aprovado. Apenas seria considerado se o valor da perda percentual do fluxo máximo fosse menor ou igual a 85%. Este parâmetro fica armazenado no módulo de parâmetros analisados do plano de decisão.

Na Figura 3.14 podemos visualizar de forma resumida o funcionamento do algoritmo proposto.

3.6 O MOTOR DE BUSCA DE SOLUÇÕES DO ARCABOUÇO

Em um sistema autonômico é desejável que exista em sua estrutura e/ou concepção um histórico de decisões tomadas para que tais informações armazenadas sirvam de apoio para a análise de novas soluções e consequente determinação de uma nova configuração da rede gerenciada. Espera-se que esta nova configuração possa gerar um estado satisfatório no que diz respeito ao atendimento dos requisitos especificados no arcabouço autonômico pelo administrador da rede gerenciada, sempre quando possível.

Por outro lado, o armazenamento de todas as informações pertinentes ao estado atual de uma



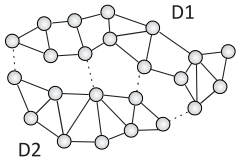
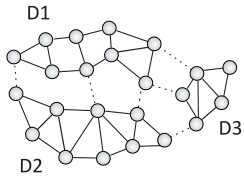
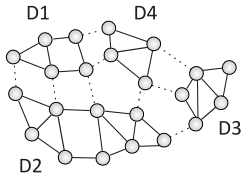
		NPT 1 (duas partições)			NPT 2 (três partições)				NPT 3 (quatro partições)				
FASE 01	Particionamento Baseado em Densidade												
FASE 02	TAREFA 01 Cálculo do Custo Computacional	D1 56%	D2 26%	INTERDOMAIN (I) 18%	D1 32%	D2 26%	D3 15%	I 27%	D1 15%	D2 26%	D3 15%	D4 14%	I 30%
	TAREFA 02 Seleção do Algoritmo do Menor Caminho	Dijsktra			Dijsktra				Dijsktra				
	TAREFA 03 Cálculo do Tempo de Execução Relativo	48,23%			36,77%				38,61%				
FASE 03	Cálculo da Perda do Fluxo Máximo	95,21%			94,44%				94,04%				
	Escolha do NPT Custo x Qualidade				*								

Figura 3.14. Visualização das etapas do particionamento proposto

rede e de seus dispositivos poderá aumentar a complexidade da gerência autonômica, devido à alta granularidade de informações coletadas, e a uma possível grande quantidade de dispositivos gerenciados. Dentro desta realidade espera-se que o armazenamento e recuperação de forma satisfatória atenda a necessidade de especificar um motor de busca de soluções capaz de propor novas configurações considerando:

- A especificação de um motor de busca de soluções, que permita modelar o conhecimento dentro da realidade da gerência de redes de computadores. Esta especificação deve contemplar a possibilidade de operar com baixa aquisição de conhecimento – possibilitando o início da operação de um arcabouço autonômico ainda sem uma base de informações significativa– e que tente evitar a repetição de erros ocorridos no passado.

- Uma base de informação que possibilite a representação das informações de forma flexível, permitindo a adaptação das informações relevantes da estratégia utilizada pelo arcabouço, e uma recuperação de dados em tempo factível para tornar viável o tempo de

resposta das soluções.

Na busca de novas soluções, um modelo autonômico deve incorporar técnicas de aprendizagem e raciocínio *learning and reasoning techniques* para garantir as propriedades auto-* (Jennings et al., 2007). Segundo (Dobson et al., 2006), durante a fase de análise e planejamento diversas técnicas podem ser utilizadas tais como inferência, teoria dos jogos, modelos econômicos, teoria da decisão, análise de risco, dentre outros.

A correlação de eventos também pode ser uma técnica utilizada para gerenciamento integrado em sistemas autonômicos dentro do contexto de redes de computadores(Martin-flatin; Jakobson; Lewis, 2007), uma vez que ela utiliza experiências passadas para encontrar novas soluções através de técnicas como RBR (*Rule-Based Reasoning* – Raciocínio Baseado em Regras) e CBR (*Case-Based Reasoning* – Raciocínio Baseado em Casos) (Martin-flatin; Jakobson; Lewis, 2007). O CBR é uma abordagem para resolução de problemas que tem a habilidade de se adaptar automaticamente para encontrar novas soluções baseadas em casos passados, diferentemente do RBR, que pode ou não ser adaptado.

O funcionamento do CBR é similar a forma humana de resolução de problemas, já que os mesmos são resovidos com base em experiências anteriores (*case based*) e em suas respectivas similaridades. Para tanto, este paradigma faz uso do Algoritmo dos 4R's apresentado na Figura 3.15 que representam *retrieve*, *reuse*, *revise* e *retain* (Aamodt; Plaza, 1994).

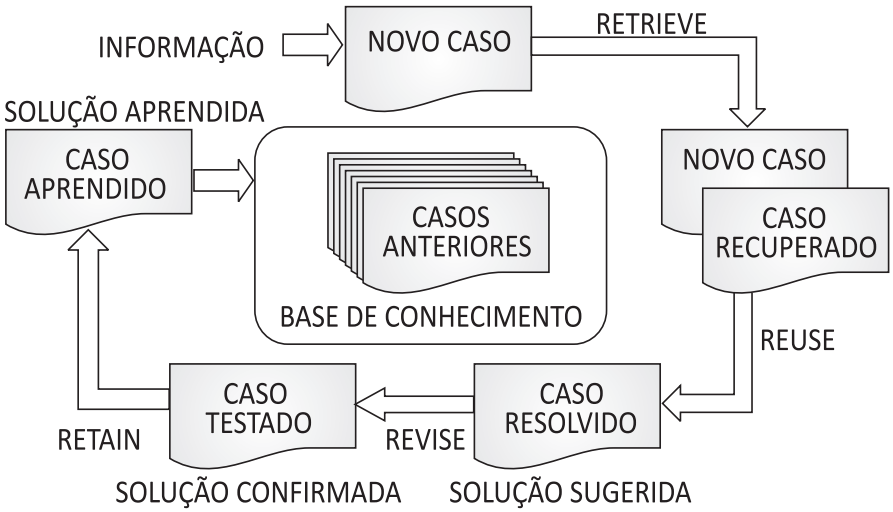


Figura 3.15. O ciclo do algoritmo 4R – Adaptado de (Aamodt; Plaza, 1994)

O paradigma de CBR envolve raciocínio através do armazenamento (*retrieve)* de um caso e sua respectiva solução, para a apoio à resolução de problemas novos. Além disso, o CBR pode ser usado para resolver diversos tipos de problemas em áreas distintas, apesar de de ser indicado para situações onde é impossível entender e prever completamente os domínios, ou quando os fatores de sucesso de uma solução não podem ser modelados de forma explícita.

Em geral a utilização de CBR oferece as seguintes vantagens (Shiu; Pal, 2004a):

- Redução da tarefa de aquisição de conhecimento – Por não trabalhar com modelo baseado em regras, o CBR permite maior facilidade na busca de uma nova solução fruto da comparação com situações anteriores.

- Evita repetição de erros do passado – Ao armazenar os casos que obtiveram êxito ou fracasso o CBR pode evitar que erros se repitam para uma mesma situação.

- Proporciona flexibilidade na modelagem do conhecimento – Diferentemente de sistemas baseados em modelos, o CBR permite que o desenvolvedor modele seu conhecimento em função da modelagem de seu caso.

- Raciocínio em um domínio com um pequeno corpo de conhecimento – Esta vantagem permite que um sistema seja utilizado sem um mapeamento completo do domínio. Esta vantagem é bastante interessante no contexto de autonomia para ambientes gerenciados dinâmicos, uma vez que o sistema pode alcançar o sucesso sem necessariamente ter a solução já gravada na base de conhecimento .

- Raciocínio com dados incompletos ou imprecisos – O casamento (*match*) entre a recuperação de um caso e o estado atual não precisa ser perfeito, possibilitando maior flexibilidade na escolha de uma solução.

Como mencionado anteriormente, CBR usa experiências anteriores para resolver novos problemas e isso geralmente requer a aquisição de conhecimentos significativamente menor. Esta vantagem faz do CBR uma alternativa, em distintas situações, para sistemas que durante sua implantação tenham com pequena quantidade de experiência, que aumentará de acordo com as novas soluções criadas, consequentemente, enriquecendo a base de casos.

Apesar de não ser considerado como um novo paradigma de inteligência artificial, CBR vem sendo recentemente utilizado para resolver problemas de alta complexidade como a gerência autonômica de redes (Samaan; Karmouch, 2004) (Samaan; Karmouch, 2005). Ao considerar a computação autonômica, abordagens recentes sugerem o uso deste paradigma em sistemas autonômicos para melhorar a eficiência de tais sistemas (Khan; Awais; Shamail, 2008).

A seguir será apresentado como foi mapeado um caso para o arcabouço com características autonômicas utilizando a técnica de particionamento.

3.6.1 Mapeamento de um Caso

Um caso tipicamente corresponde ao armazenamento de uma tupla problema/solução que ocorreu no passado. Desta forma, no contexto autonômico de rede de computadores, um caso (c_i) é composto por um problema encontrado no ambiente gerenciado (sintomas) (p_i) e por uma solução para este problema (s_i), descrito da seguinte forma:

$$c_i = \{p_i, s_i\} \tag{3.16}$$

Este modelo básico de caso foi estendido neste trabalho para garantir mais eficiência e flexibilidade na busca de novas soluções. Flexibilidade, pois um problema pode estar em diferentes contextos, por exemplo, um enlace pode falhar e causar diferentes reações a depender do estado atual da rede (distintos perfis de tráfego). Eficiência, com a redução do domínio de casos-base não relevantes no contexto atual (mudança topológica ou baixa ocorrência de casos). Com isso um caso estendido corresponde a seguinte tupla:

$$c_i = \{tp_i, tr_i, npt_i, qs_i, ns_i, ts_i, c_i\} \tag{3.17}$$

onde:

- tp_i – representa a topologia da rede gerenciada. O objetivo é verificar se existem falhas na topologia.
- tr_i – representa o tráfego da rede gerenciada. O total de tráfego é armazenado para cada partição em função do NPT atual.
- npt_i – representa os possíveis particionamentos (NPTs) em função do algoritmo utilizado na NPCE.
- qs_i – indica a qualidade da solução proposta, que é medida de acordo com a verificação dos SLAs não atendidos.
- ns_i – indica o número de ocorrência de um caso c_i; Utilizado para verificar a ocorrência de casos pouco utilizados e refinar a busca por novas soluções.
- ts_i – indica o tempo da última ocorrência de um caso. É desejável remover casos não utilizados recentemente.
- c_i – indica o percentual do custo computacional em realização a não utilização do particionamento, apresentado na Seção 3.5.2);

3.6.2 A Representação do Conhecimento – Associando o Particionamento ao Motor de Busca de Soluções

Para que a utilização da representação do conhecimento através do raciocínio baseado em casos seja possível é necessário que seja utilizado um método capaz de armazenar e indexar tais informações do CBR.

Em geral, os casos podem ser considerados como experiências contextualizadas, úteis para que os objetivos do arcabouço sejam alcançados. Portanto, a representação do caso pode ser vista como a tarefa capaz de permitir que o arcabouço reconheça, armazene, e processe as suas experiências passadas contextualizadas. Ao considerar a representação de um caso, é de fundamental importância observar dois pontos: (a) os modelos conceituais que são usados para projetar e representar casos, especificados na Seção 3.6 e, (b), os meios de implementação das informações no computador, como por exemplo, em uma tabela num SGBD[13] (Sistema de Gerenciamento de Banco de Dados) relacional; algumas estruturas de dados – tais como árvores B – ou numa hierarquia orientada a objetos. Desta forma, a escolha de um método para a representação e indexação de um caso é essencial, porque fornece a estrutura de base para utilização de RBC em sistemas autonômicos.

Como mencionado anteriormente, um caso, consiste principalmente de duas partes: o problema e a solução. Cada parte pode ser dividida em componentes menores que podem ser úteis mais tarde para tarefas de raciocínio , dependendo do problema a ser tratado. No contexto da gerência de redes de computadores são necessárias informações refentes a topologia, ao tráfego e aos requisitos esperados pelo administrador. Por outro lado a representação deve armazenar também informações referentes a estratégia utilizada pela camada de decisão, em especial pelo NPCE.

Uma técnica comumente usada para representação de um caso é a abordagem de banco de dados relacional (Shiu; Pal, 2004b). Este modelo é simples e flexível e tem sido amplamente adotado em muitas aplicações de raciocínio baseado em casos (Shiu; Pal, 2004b). Cada objeto é representado por uma linha de uma tabela relacional onde as colunas são usados para definir os atributos (ou campos) dos objetos, sejam eles informações da rede gerenciada ou dados referentes à NPCE.

Assim como a representação de um caso, poderíamos dividir a(s) tabela(s) em: descrição do problema e parte da solução. Se o caso tem muitas relações com vários objetos ou se o caso pode ser dividido em subcasos, uma rede de relacionamentos pode ser desenvolvida e modelada num SGBD.

De fato o banco de dados relacional pode reduzir a redundância de armazenamento, bem como melhorar a eficiência de recuperação (Shiu; Pal, 2004b). Uma vantagem deste modelo é que ele apresenta uma visão de vários lados dos casos a partir da possibilidade de relacionamento dos dados de forma mais natural. No entanto, este modelo de representação tem a desvantagem de consumir tempo e energia consideráveis para o desenvolvimento de um modelo de banco de dados relacional para a representação de um caso.

Desta forma, um método de armazenamento com a utilização de banco de dados viabiliza a relação entre as informações e ainda tem como vantagens: compartilhamento de dados, suporte

[13]Do inglês *Data Base Management System* (DBMS) - é o conjunto de programas de computador responsáveis pelo gerenciamento de uma base de dados, cujo seu principal objetivo é retirar da aplicação cliente a responsabilidade de gerenciar o acesso, a manipulação e a organização dos dados. O SGBD disponibiliza uma interface para que seus clientes possam incluir, alterar ou consultar dados previamente armazenados.

a múltiplas visões; redução da redundância de dados, minimização da inconsistência, suporte a transações e atomicidade, manutenção da integridade dos dados e flexibilização na representação (Elmasri; Navathe, 2003).

A representação desenvolvida com a utilização de banco de dados é apresentada (Figura 3.16) – de acordo com a especificação contida na equação 3.17 – contém as seguintes tabelas:

- casos – principal tabela do sistema, possui uma chave primária composta contendo a identificação da topologia (informações dos enlaces e verificação das falhas), do particionamento atual (NPT) e do tráfego por partição;
- npt – contém informações do particionamento relativas ao caso armazenado;
- tr – nesta tabela são indexados o percentual de tráfego para o NPT do caso armazenado. Nela consta o percentual de tráfego por partição e entre partições. O objetivo é utilizar essas informações como base para determinação do custo computacional para o NPT definido pelo NPCE.
- tp – contém informações da topologia e a chave primária da tabela de falhas por partição.
- falhas – indica a quantidade de falhas por partição.
- custo – armazena o custo computacional por partição.
- perdadefluxomáximo – armazena a perda de fluxo máximo por partição de um NPT.
- algoritmo – apenas guarda o algoritmo de busca de menor caminho e sua complexidade.

3.7 EXEMPLO DE OPERAÇÃO

Nesta seção serão apresentados exemplos de operação do arcabouço em função do estado da rede no que diz respeito ao percentual de fluxos alocados. Os exemplos tomam como base possíveis estados da rede e o NPT escolhidos. Em seguida são mostrados como o ambiente pode reagir em caso de falhas e/ou de não atendimento das necessidades dos usuários e serviços.

Tecnicamente o arcabouço pode encontrar duas situações distintas: (a) uma falha ocasionando a reconfiguração autonômica dos recursos da rede e (b) uma situação que não existem falhas mas uma reconfiguração pode maximizar a alocação dos recursos na rede gerenciada.

Para exemplificar os possíveis estados e suas transições, serão utilizadas as nomenclaturas $A1, A2, A3 e A4$ para o funcionamento na ocorrência de falhas e $B1, B2, C1 e C1$ quando o objetivo é melhorar a alocação de recursos. A diferença entre os exemplos *B* e *C* é basicamente o estado inicial, pois em *C* o estado *C*1 é considerado não desejável.

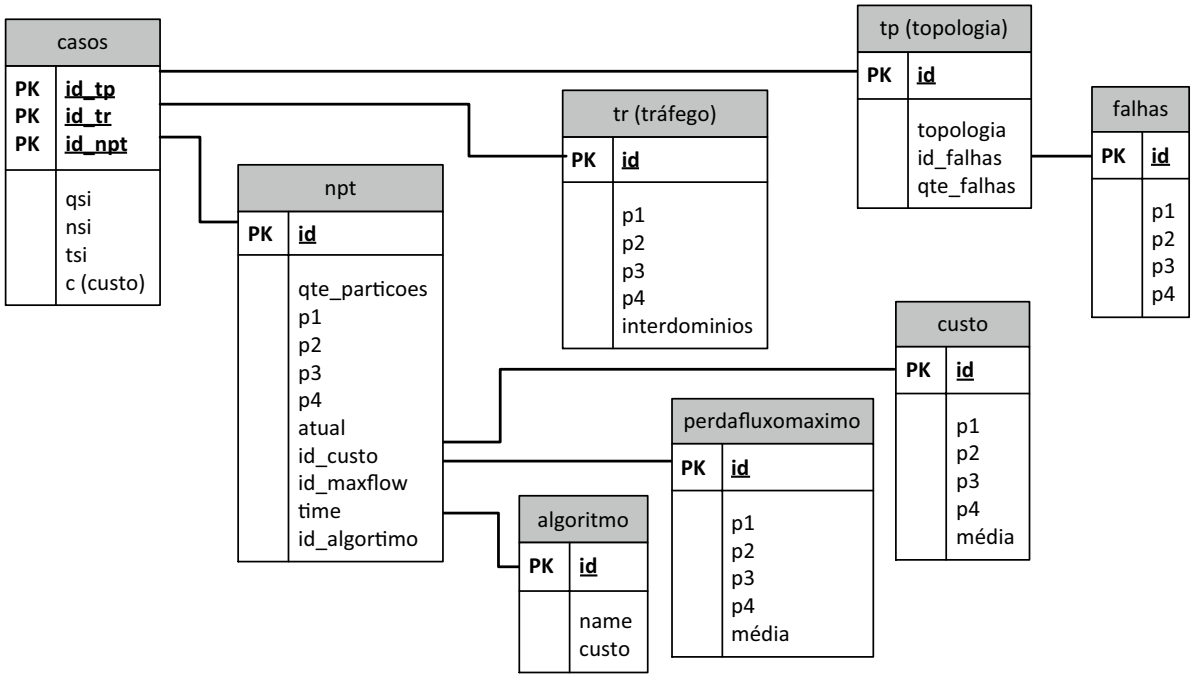


Figura 3.16. Modelo de dados para tomada de decisões

3.7.1 Funcionamento na Ocorrência de Falhas

- $(A1)$ – A rede gerenciada está com 100% dos fluxos alocados e o NPT atual possui cinco partições.

- $(A2)$ – Ocorre um conjunto de falhas ($1 \leq falhas \leq e$) que podem estar presentes em diferentes partições. Na ocasião apresentada, tal conjunto de falhas afetou em torno de 15% dos enlaces da rede.

- $(A3)$ – O arcabouço ao receber uma fotografia (*snapshot*) da rede gerenciada, procura realocar os fluxos em função da NPCE – considerando a relação entre tempo de execução e qualidade da solução. Nesta ocasião dois acontecimentos podem ocorrer:

  - $(A3_1)$ – O NPCE pode optar em manter o NPT atual, realocando todos os fluxos afetados pelas falhas sempre que possível. Caso estes fluxos não consigam ser alocados o NPCE esperará a correção das falhas para realocar os fluxos pendentes.
  - $(A3_2)$ – O NPCE pode sugerir um novo NPT visando a maximação da alocação de fluxos, com foco na qualidade da solução proposta. Neste caso, a nova NPT geralmente opta por uma quantidade menor de partições, visando a redução da perda do fluxo máximo apresentada na Seção 3.5.3. Isso poderá ter um custo computacional maior durante a alocação de fluxos.

- (A_4) – Uma vez que as falhas sejam corrigidas, o NPCE pode manter o particionamento e apenas alocar fluxos eventualmente não alocados (situação apresentada em $(A31)$. O

NPCE também pode reavaliar e utilizar NPTs intermediários ou até mesmo o NPT original. Na Figura 3.17, nota-se em $(A4)$ o NPCE poderia escolher melhorar a utilização de recursos migrando para $(A41)$ ou $(A42)$, onde a quantidade de NPTs pode ser mantida em três ou aumentada para cinco.

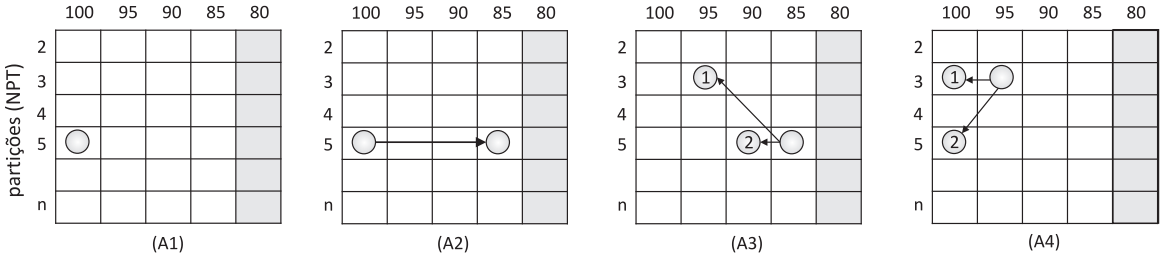


Figura 3.17. Operação 01 – Ambiente com falhas

3.7.2 Funcionamento Visando uma Melhor Alocação de Recursos

- $(B1)\&(C1)$ – Em (B1) a rede gerenciada também possui seus objetivos alcançados, mas o arcabouço verificou que existe uma nova configuração capaz de melhorar o estado atual da rede. O que difere esta situação da $(C1)$ é a identificação de uma não conformidade de alocação dos recursos, verificada para este caso e indicada em cinza na Figura 3.18. É importante ressaltar que não existem falhas neste exemplo de funcionamento.
- $(B2)\&(C2)$ – Uma reconfiguração pode ser executada considerando duas distintas situações:
  - $(B2_1)\&(C2_1)$ – Mantendo a escolha do NPT atual. Neste caso a variável tempo de resposta é considerada prioritária no processo de alocação de recursos.
  - $(B2_2)\&(C2_2)$ – Buscar um NPT mais adequado visando uma melhor alocação dos serviços. Neste caso a variável qualidade é considerada prioritária no processo de alocação de recursos.

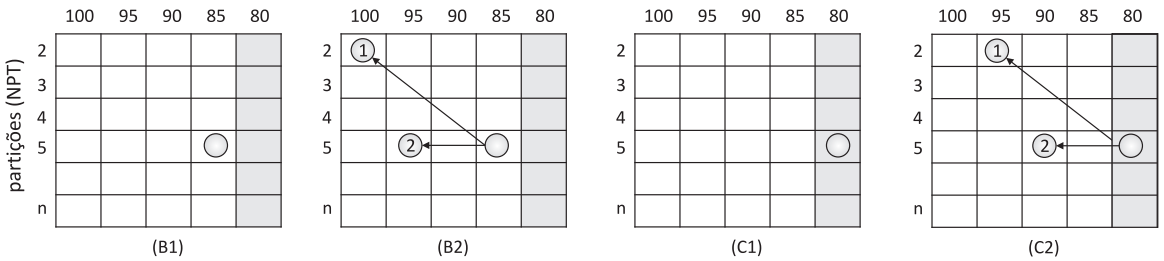


Figura 3.18. Operação 02 – Ambiente sem falhas

No capítulo seguinte serão apresentados os testes realizados através de simulações considerando os ciclos de operações como forma de avaliação do arcabouço com testes de escalabilidade, qualidade e mediante a falhas de enlace . Os testes de escalabilidade serão realizados em

função do aumento da quantidade de fluxos e do aumento da quantidade de nós. As simulações consideram o pior caso, ou seja, alocação de todos os fluxos de uma rede. As medições para este caso envolvem a comparação entre os tempos de resposta. A qualidade da solução é medida através do percentual médio da alocação dos enlaces e do valor médio do comprimento dos caminhos dos fluxos. Em relação as falhas, os testes verificam a capacidade de reação da arcabouço em função do percentual de fluxo realocado em consequência das falhas.

Capítulo 4

Divide as dificuldades que tenhas de examinar em tantas partes quantas for possível, para uma melhor solução – René Descartes

VALIDAÇÃO DO MODELO PROPOSTO

A necessidade do tratamento da escalabilidade motivou a criação do arcabouço para gerência autonômica de redes de computadores apresentado no capítulo anterior. Os elementos do arcabouço foram descritos com um detalhamento especial no plano de controle.

Neste capítulo o arcabouço proposto será analisado em função dos requisitos especificados na Seção 3.1 através de simulações, análises matemáticas e discussões em quesitos menos mensuráveis como adaptabilidade. O objetivo deste capítulo é fornecer um estudo mais aprofundado do arcabouço de gerência utilizando em função dos resultados obtidos, a partir principalmente das simulações.

4.1 ANÁLISE ESTATÍSTICA DOS DADOS

Nesta seção a técnica utilizada para analisar os dados da simulação é apresentada. Na verdade, os resultados são analisados usando o método tradicional de inferência estatística, uma vez que este permite estimar um parâmetro de uma população com base em dados amostrais (Triola, 2008).

De fato, na medida em que não é possível calcular a utilização dos limites ou o melhor caminho para o somatório de fluxos considerando todas as possíveis redes, a análise descrita aqui é baseada em valores amostrais.

Além disso, no sentido de avaliar a dispersão dos dados da amostra, o desvio padrão amostral foi calculado, conforme descrito na Equação 4.1.

$$\delta = \sqrt{\frac{1}{n-1} * \sum (x_i - \overline{x})^2} \tag{4.1}$$

onde:

- *n* é o tamanho da amostra;

- x_i é o valor amostral;
- $\bar{x}$ é a média amostral

O desvio padrão foi calculado usando as funções estatísticas do programa R (R Development Core Team, 2011).

Para determinar a acurácia dos valores obtidos, o intervalo de confiança para cada calor médio foi calculado. O objetivo foi determinar o intervalo válido para os dados com uma confiança de 95%. De fato, de acordo com o Teorema do Link central, quando o parâmtero estimado é a média, 95% dos valores estão a uma distância da média de duas vezes o desvio padrão (Morettin; Bussab, 2004).

4.2 ESPECIFICAÇÃO DOS CENÁRIOS DE SIMULAÇÃO

Os testes do arcabouço proposto utiliza duas classes de cenários de simulação com objetivos distintos de (i) testar o funcionamento da solução mediante a escalabilidade (*stress test*) e (ii) testar se a redução do tempo de execução impacta na qualidade e/ou na sobrevivência da solução apresentada.

Para o teste de escalabilidade o objetivo é medir o tempo de alocação de recursos para v vértices e f fluxos e ampliar estes valores gradativamente, verificando qual o impacto deles na eficiência da solução proposta. Para cada conjunto de simulações, a alocação sem particionamento também é realizada visando a medição quantitativa e qualitativa dos resultados.

Já o teste de sobrevivência da solução apresentada objetiva injetar aleatoriamente um conjunto de falhas nos enlaces e verificar qual a porcentagem de tráfego que foi realocado sem alterar o tráfego dos enlaces não afetados pelas falhas.

Em ambos os casos, o arcabouço proposto (com CBR e particionamento) é comparado a alocação sem estes recursos. O algoritmo utilizado para busca de caminhos em todas as simulações é o algoritmo de Dijkstra por ser uma referência na literatura e para padronizar as simulações realizadas.

Para que a escolha das topologias das redes simuladas sejam definidas pelo administrador – podendo influenciar nos resultados – as redes geradas consideram as seguintes premissas:

i) Todas as redes são conexas;

ii) O grau mínimo de todos os nós (vértices) é maior ou igual a dois ($d(g) \geq 3$), para evitar o roteamento trivial;

iii) Todos as redes são grafos simples, ou seja, são grafos não direcionados que não contem *loops* e nos quais existe mais de uma aresta para cada par de diferentes vértices.

Exemplos das redes geradas são apresenatdos na Figura 4.1 (Figura 4.1).

As simulações utilizaram o software R (R Development Core Team, 2011) com o pacote igraph (Csardi, 2010), descritos no Anexo B.

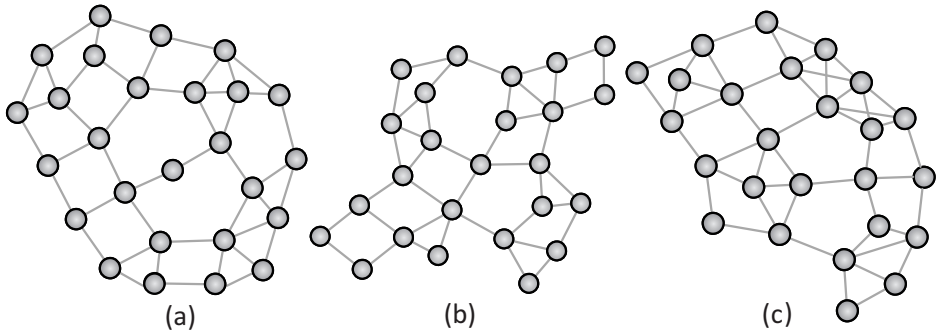


Figura 4.1. Exemplos de grafos gerados com cardinalidade igual a 25

4.3 CENÁRIOS DE SIMULAÇÃO

4.3.1 Cenário de Simulação 01 – Teste de *Stress* – Quantidade de Fluxos

Este cenário de simulação tem como objetivo testar o tempo de execução do algoritmo proposto mediante uma sobrecarga de fluxos. Para tal, o cenário inicial é composto por 25 vértices e 10000 fluxos de rede. Em seguida o tráfego de rede é aumentado para 200 %, resultando 20000 fluxos, e 400 %, resultando 40000 fluxos, tanto com a utilização da solução proposta, quanto na alocação tradicional (sem particionamento).

Em resumo os parâmetros de simulação são:

- 6 topologias distintas com 25 vértices ($v = 25$);
- Para cada rede seis (6) perfis de tráfego distintos;
- Os itens anteriores foram testados com 10000, 20000 e 40000 fluxos ($\approx$ 2,3 GB, 4,7 GB e 9,5 GB), totalizando 108 simulações.

A Figura 4.2 apresenta os resultados da simulação para o primeiro cenário de simulação. Os resultados são apresentados em valores médios percentuais, em função do cenário inicial (25 vértices e 10000), sem a utilização do particionamento.

Além disto, para cada caso são apresentados os valores máimos e mínimos com 95% de confiança.

De fato, o valor médio obtido é apresentado na 4.1. Observe que o desvio padrão e o intervalo de confiança, considerando 95% de confiança, também são apresentados, com o objetivo de verificar a acurária do valor médio obtido na simulação.

Neste primeiro cenário de simulação é possível verificar um melhor desempenho do algoritmo proposto em relação ao tempo de resposta sem a utilização do particionamento para os valores de 10.000, 20.000 e 40.000 fluxos. Nota-se também que a valor máximo com a utilização do particionamento é menor que o valor médio da solução sem o particionamento. Por

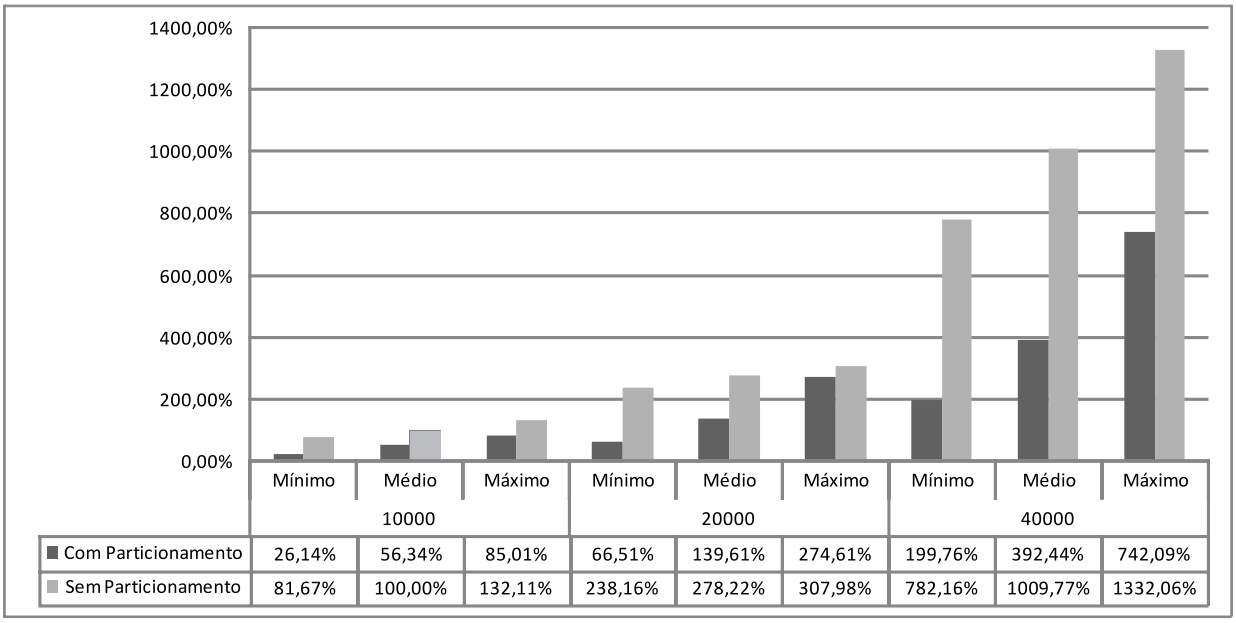


Figura 4.2. Cenário 01 – Aumento da quantidade de fluxos

Fluxos	10.000		20.000		40.000	
Ambiente	Sem Part.	Com Part.	Sem Part.	Com Part.	Sem Part.	Com Part.
Média	52,93	92,71	129,44	257,95	363,84	936,20
Desvio Padrão	11,76	14,18	16,11	53,31	202,97	142,19
Intervalo de Confiança (95%)	[47,60 ; 56,87]	[88,87 ; 96,56].	[112,02 ; 146,85]	[252,69 ; 263,21]	[317,39 ; 410,29]	[869,90 ; 1002,51]

Tabela 4.1. Cálculo do intervalo de confiança (95%) para 25 nós – Valores em segundos

exemplo, para 10.000 fluxos o valor máximo representa 85,01% do valor médio sem o particionamento.

Uma segunda consolidação dos resultados onde os valores percentuais com e sem o particionamento para 10.000, 20.000 e 40.000 são comparados individualmente é apresenrado na Figura 4.3.

Observe que para as simulações que não consideram o particionamento tem valores consideravelmente menores, como por exemplo a solução para 40000 fluxos para a qual há uma economia de 62,15% do tempo de alocação dos recursos.

A seguir, examinaremos o segundo cenário de simulação.

4.3.2 Cenário de Simulação 02 – Teste de *Stress* – Quantidade de Dispositivos

Nesta simulação a análise da escalabilidade foi realizada em função da quantidade de dispositivos da rede, onde o valor foi aumentado para 50 e 100 vértices, representando um aumento em duas e quatro vezes, respectivamente. Neste cenário o total de fluxos foi mantido em 10.000 para todas as simulações executas. Podemos resumir o cenário como:

- 6 topologias distintas com 25 vértices ($|v| = 25$);

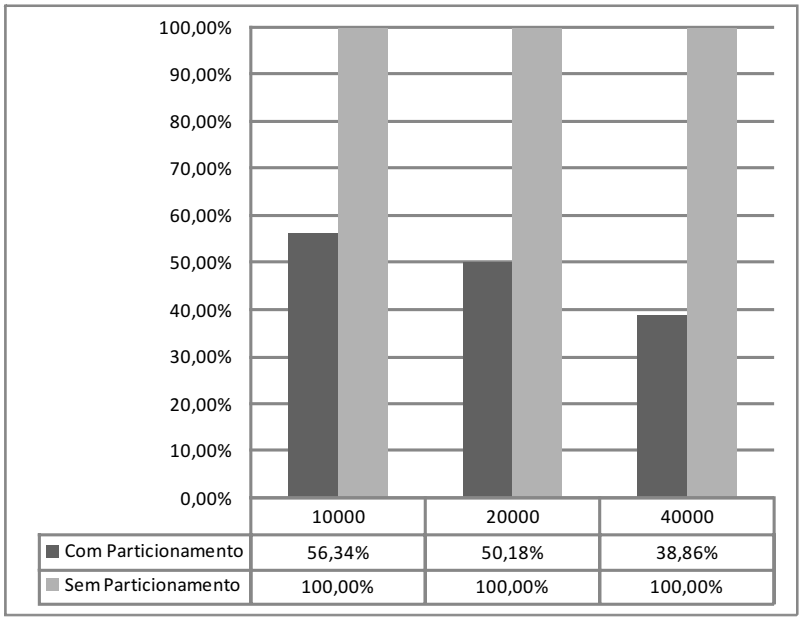


Figura 4.3. Cenário 01 – Aumento da quantidade de fluxos

- 6 topologias distintas com 50 vértices ($|v| = 50$);
- 6 topologias distintas com 100 vértices ($|v| = 100$);
- 3 distintos perfis de tráfego para cada conjunto de topologias;
- 10000 fluxos para todas as 48 simulações ($\approx$ 2,2 GB).

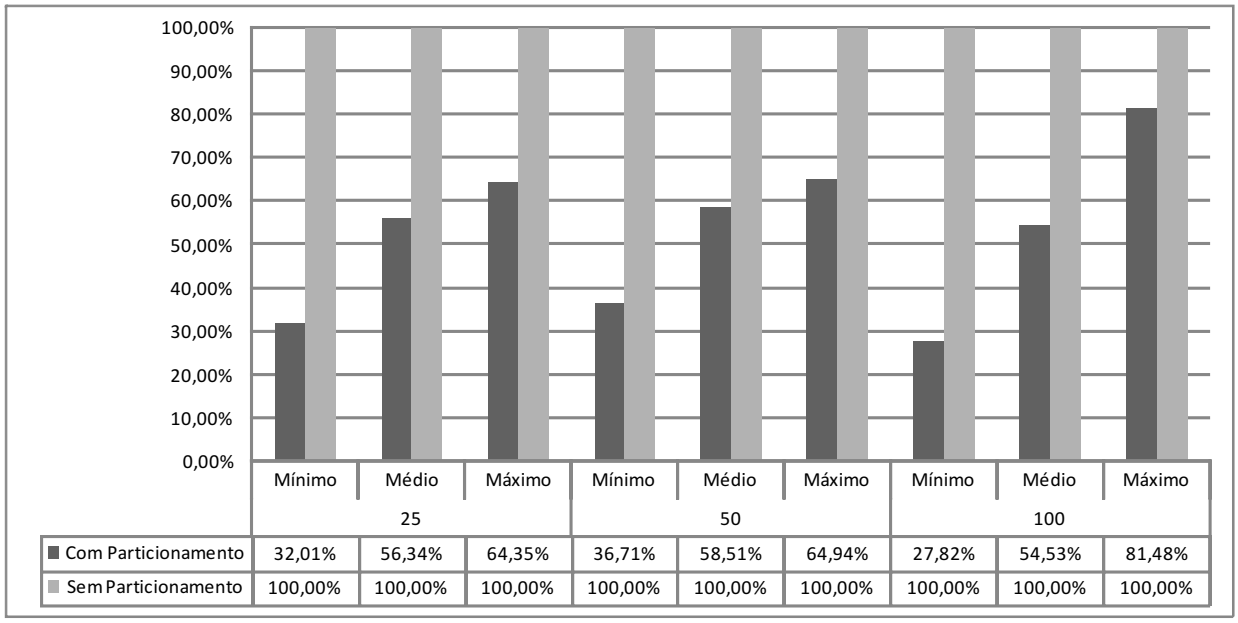
Neste segundo cenário, a validação da escalabilidade é realizada em função da cardinalidade da rede. Em todas as simulações a quantidade de fluxos foi fixada em 10.000 unidades ($f = 10,000$) e a quantidade de nós aumentou de 25 para 50 e 100. O tempo de execução foi calculado para o algoritmo proposto e para o algoritmo tradicional (sem particionamento) (Figure 4.4). Os resultados são vistos na Figura 4.4.

Observe que em todos os casos, a solução proposta apresentou o tempo de resposta médio de 56,34%, 58,51% e 54,53% para 25, 50 e 100 vértices, respectivamente, em relação aos valores médios considerando a abordagem tradicional (sem particonamento).

Nós (vértices)	25		50		100	
Ambiente	Sem Part.	Com Part.	Sem Part.	Com Part.	Sem Part.	Com Part.
Média	52,93	92,71	53,38	91,23	54,85	103,34
Desvio Padrão	11,76	14,18	9,43	13,16	6,65	21,56
Intervalo de Confiança (95%)	[47,60 ; 56,87]	[88,87 ; 96,56].	[49,08 ; 57,68]	[88,15 ; 94,32]	[47,81 ; 61,89]	[101,17 ; 105,52]

Tabela 4.2. Cálculo do intervalo de confiança (95%) para 10000 fluxos – Valores em segundos

Note que para 25, 50 e 100 vértices o valor médio está contido no intervalo de confiança, indicando que a solução proposta pode ser utilizado para ofertar a escalabilidade em redes de computadores.



	Mínimo	Médio	Máximo	Mínimo	Médio	Máximo	Mínimo	Médio	Máximo
	25			50			100		
Com Particionamento	32,01%	56,34%	64,35%	36,71%	58,51%	64,94%	27,82%	54,53%	81,48%
Sem Particionamento	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%

Figura 4.4. Cenário 02 – Aumento da quantidade de nós

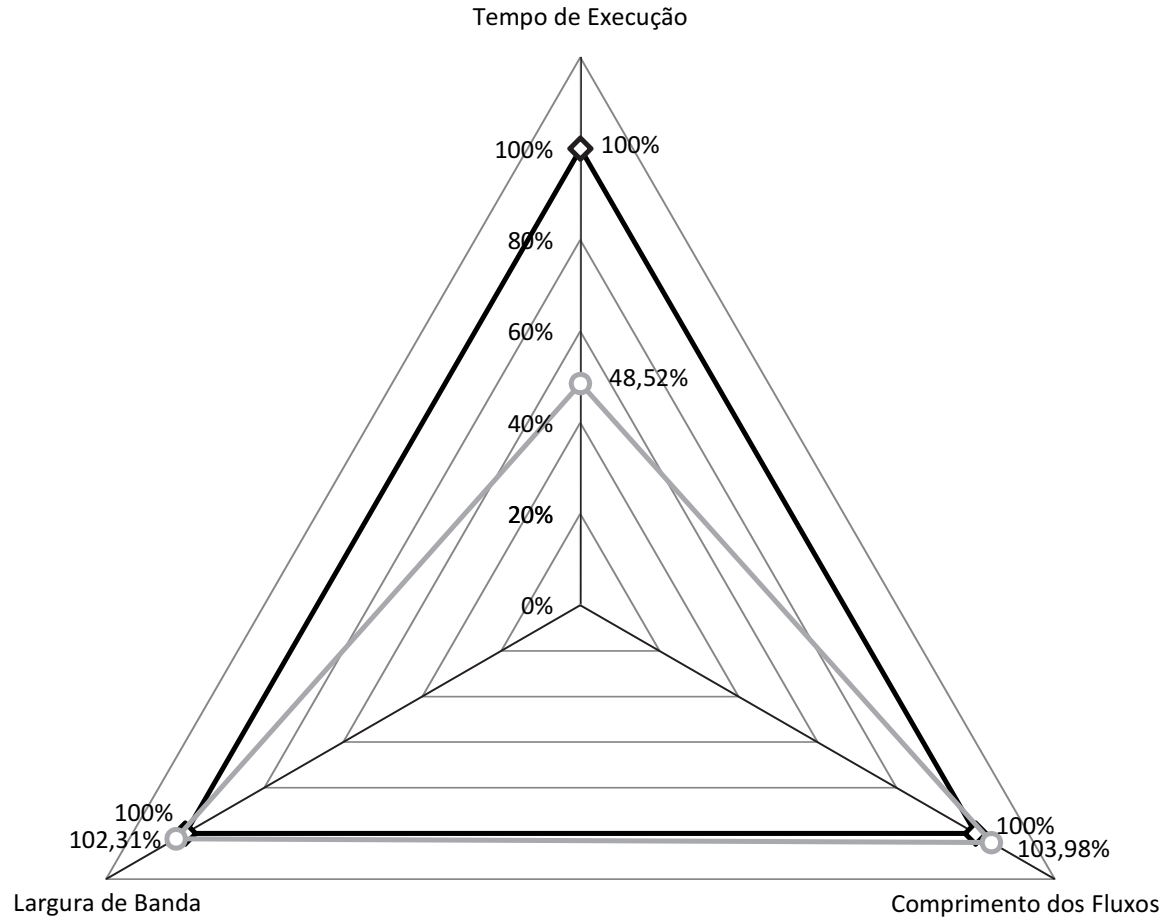
4.3.3 Cenário de Simulação 03 – Manutenção da Qualidade de Serviço

Neste cenário de simulação foram resumidos todas as simulações anteriores onde $v = 25$, que só verificavam o tempo de resposta, e para cada uma delas dois requisitos de qualidade de serviço foram analisados – (i) a largura de banda média utilizada e (ii) o comprimento médio dos caminhos. Em suma os parâmetros utilizados foram:

- 6 topologias distintas com 25 vértices ($|v| = 25$);
- 6 topologias distintas com 50 vértices ($|v| = 50$);
- 6 topologias distintas com 100 vértices ($|v| = 100$);
- 10000 fluxos ($\approx$ 2,3 TB);
- 6 distintos perfis de tráfego para cada conjunto de topologias, totalizando 108 simulações.

Observe que houve uma redução significativa do tempo de resposta com a manutenção da qualidade de serviço, uma vez que, em média, tanto a utilização da largura de banda quanto o comprimento médio dos caminhos escolhidos tiveram um desempenho bastante similar com a utilização da solução proposta, sendo este acréscimo de 2,31% e 3,98% respectivamente. Estes resultados podem ser visualizados na Figura 4.5.

Essa manutenção dos parâmetros de QoS especificados ocorre pois o particionamento proposto, baseado em densidade, permite uma maior concentração de caminhos dentro dos domínios. Além disso, o algoritmo considera também para o limite máximo de partições a perda do fluxo máximo, vista na Seção 3.5.3.



	Tempo de Execução	Comprimento dos Fluxos	Largura de Banda
Sem Particionamento	100%	100%	100%
Com Particionamento	48,52%	103,98%	102,31%

Figura 4.5. Cenário 03 – Manutenção da qualidade de serviço

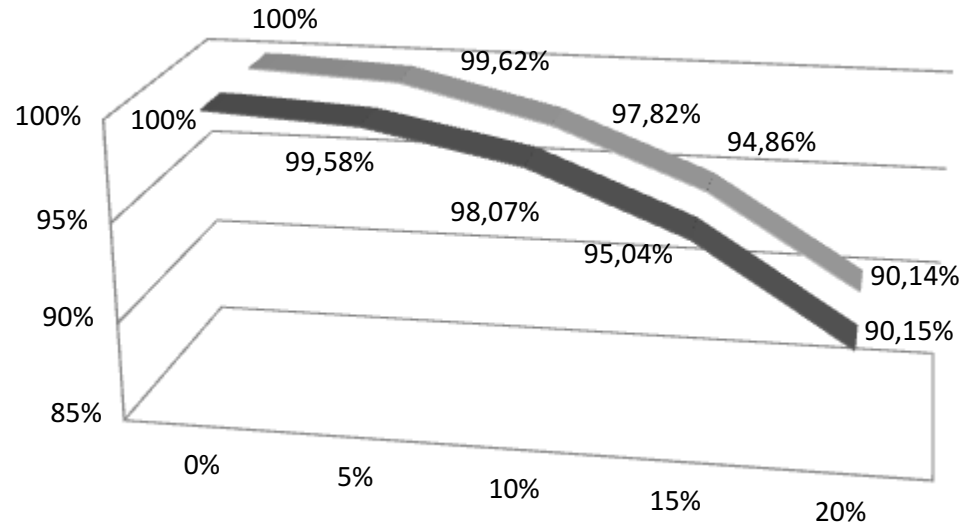
4.3.4 Cenário de Simulação 04 – Sobrevivência

Neste cenário de simulação o principal objetivo é medir a capacidade de reação da solução proposta no contexto de falhas. Para isso, foram injetadas falhas aleatórias numa rede com alocação dos enlaces entre 40% e 60%. As falhas injetadas atingiram 5%, 10%, 15% e 20% dos enlaces.

É importante ressaltar que estas falhas não são contínuas. Ou seja, o segundo conjunto de falhas é aplicado diretamente sobre o primeiro. Por exemplo, dado um grafo G com $|E| = 60$, um possível conjunto de falhas com 5% de enlaces seria e_7, e_12, e_42. Nesta caso, todo o tráfego que flui sobre estas arestas seria realocado. Se um conjunto de falhas que atinge 10% dos enlaces seja aplicado, o estado inicial da rede é recuperado e um novo conjunto de flahas é gerado a partir do original, por exemplo os conjuntos $e_7, e_12, e_42, e_32, e_6, e_51$, $e_7, e_12, e_42, e_15, e_44, e_21$ e $e_7, e_12, e_42, e_2, e_26, e_34$ são válidos.

O ambiente testado para este cenário foi:

- 8 topologias distintas com 25 vértices ($|v| = 25$]);
- 10000 fluxos ($\approx$ 2,3 TB);
- 5%, 10%, 15% e 20% de falhas aleatórias;



	0%	5%	10%	15%	20%
Sem particionamento	100%	99,58%	98,07%	95,04%	90,15%
Com particionamento	100%	99,62%	97,82%	94,86%	90,14%

Figura 4.6. Cenário 04 – Sobrevivência – Fluxo alocado após falhas de enlaces

Os resultados apontam uma equivalência entre a porcentagem dos fluxos alocados para distintos conjuntos de falhas com valores entre 5% e 20%.

4.3.5 Outros Pontos de Investigação

Agora que os resultados do algoritmo de particionamento baseado em densidades foram apresentados, é possível discutir pontos estratégicos considerando a computação autonômica. Sendo assim, alguns aspectos como alocação paralela, visão global versus visão local, entre outros, serão contextualizados.

4.3.5.1 Alocação Paralela

Todas as simulações apresentadas na seção anterior consideraram que a alocação do tráfego intra-domínios foi realizada ao mesmo tempo para cada domínio. Já o tráfego intra-domínios foi alocado separadamente. Para esclarecer melhor, o particionamento da rede e alocação são executados de acordo com a sequência abaixo:

i) O algoritmo de alocação de recursos seleciona todo o tráfego intra-domínios para cada partição;

ii) O algoritmo de alocação de recursos aloca rodo o tráfego intra-domínios;

iii) O algoritmo de alocação de recursos aloca rodo o tráfego inter-domínios;

A alocação de tráfego intra-domínios pode ser executada de forma paralela, pois o conjunto interseção de vértices para qualquer par de domínios é vazio ($V_{di} \bigcap V_{dj} = \emptyset$, $\forall i, j \in d$), ou seja, um vértice só pertence a um único domínio. Assim, a alocação não precisa ser realizada sequencialmente. Isto possibilitou a redução de tempo de execução para o algoritmo de alocação de fluxos.

4.3.5.2 Impacto do Tempo de Particionamento sobre o Tempo Total

Uma preocupação que também deve ser considerada é o tempo do algoritmo de particionamento. Se o tempo de particionamento é alto, isso pode afetar diretamente a viabilidade da solução proposta. Para analisar essa relação entre tais tempos, foram realizados particionamentos para redes com cardinalidade entre 25 a 400 vértices, assumindo um conjunto fixo (onde o conjunto tinha cardinalidade no intervalo $[2;4]$) e dinâmico de partições (onde o conjunto de partições ficou no intervalo $[2, p = \sqrt[3]{n}]$), onde p é a quantidade máxima de partições e n é o número de nós da rede ($n = |v|$). Os resultados deste experimento são apresentados na Tabela 4.3.

Tabela 4.3. Tempo de particionamento em segundos

$\|V\|$	$\|\|E\|\|$	$p \rightarrow fixo$	$p \rightarrow dinamico$
25	40	$0,280$	$0,280$
50	84	$0,358$	$0,359$
100	180	$0,905$	$1,062$
200	364	$3,859$	$5,672$
400	760	$22,214$	$39,972$

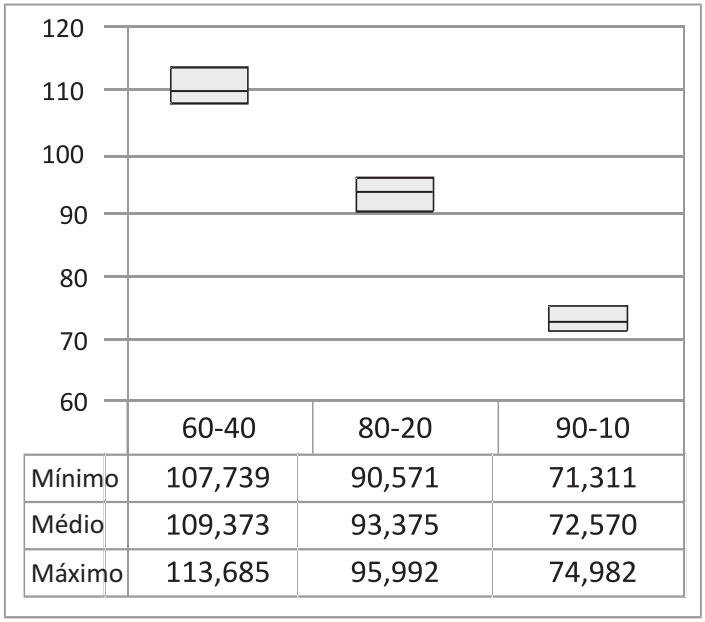
Para entender esta análise, considere, por exemplo, uma rede com 100 vértices, para a qual todos os domínis foram gerados para $p = \{2,3,4\}$(se p é fixo) e para $p = \{2,3,4,5\}$ (se p é dinâmico) totalizando 9 e 14 subgrafos, respectivamente. Em seguida, ostempos para a execução do particionamento foram somados para cada caso. Por fim, na Tabela 4.3 é possível observar que a relação do tempo de particionamento frente ao tempo de alocação não é representativo, uma vez que para 100 nós o mesmo fica em torno de um segundo.

O Tempo de Particionamento para 25, 50 e 100 vértices representa, respectivamente, apenas $0,2291\%$, $0,0995\%$, e 0.0901% do tempo de alocação para quatro partições ($d = 4$), e $0,2291\%$, $0,0998\%$ e $0,1056\%$ para o valor dinâmico de $d = \sqrt[3]{n}$.

4.3.5.3 Análise em Função do Perfil de Tráfego

O termo perfil de tráfego apresentado nesta simulação, se refere a relação entre o tráfego intra-domínios/inter-domínios. Em todas as outras simulações este valor foi mantido fixo, mas neste cenário variou entre 60% – 40% e 90% – 10%. Todos os outros parâmetros foram mantidos fixos.

Observa-se que quanto maior o tráfego intra-domínios, menor será o tempo de execução do algoritmo, conforme mostrado na análise 3.4.2. Os resultados estão apresentados na Figura 4.7.



	60-40	80-20	90-10
Mínimo	107,739	90,571	71,311
Médio	109,373	93,375	72,570
Máximo	113,685	95,992	74,982

Figura 4.7. Análise baseada no perfil de tráfego

Isso ocorre pois a alocação de fluxos inter-domínios utiliza a visão global da rede, enquanto a intra-domínios utiliza uma visão local. Assim a complexidade de busca de caminhos é reduzida em decorrência da menor cardinalidade dos domínios em relação a toda rede. Quanto maior for o tráfego intra-domínios, melhor o algoritmo se comportará em relação ao tempo de execução.

4.4 CLASSIFICAÇÃO DO MODELO PROPOSTO

Na Seção 2.3 foi apresentada uma classificação de sistemas autonômicos em função de seis parâmetros (Samaan; Karmouch, 2009): grau de atividade, grau de adaptabilidade, grau de inteligência, grau de conhecimento, grau de memória e grau de autonomia. De acordo com esta classificação o arcabouço proposto pode ser especificado como:

i) Reativo – Grau de Atividade (*Activity Degree*) – O arcabouço busca otimizar ou reconfigurar a rede gerenciada para estados que já ocorreram. Ele não infere ou predita estados futuros a partir do histórico de estados de sua base de conhecimento. Para tal, é necessária

uma maior integração completa entre a fase de monitoramento com o arcabouço proposto visando estudar as possibilidades de estados futuros.

ii) Aberto – Grau de Adaptabilidade (*Adaptability Degree*) – Conforme citado anteriormente, o conceito de adaptabilidade apresentado por (Samaan; Karmouch, 2009) diz respeito a habilidade/inabilidade de um sistema autonômico adaptar-se a mudanças em função de estados criados ou pré-definidos ((Oreizy et al., 1999)). O arcabouço apresentado propõe novas soluções para estados não previstos.

iii) Hierárquico – Grau de Inteligência (*Intelligence Degree*) – O arcabouço proposto faz tomada de decisões de forma centralizada, mas distribui a busca por caminhos intra-domínios para serem executadas em paralelo.

iv) Parcialmente Autonômico – Grau de Consciência (*Awaress Degree*) – O arcabouço proposto não engloba todas as auto-* propriedades tendo foco na auto-configuração e auto-otimização, conforme especificado em 3.2.1.

v) Histórico – Grau de Memória (*Memory Srength*) – O motor de busca de soluções guarda todos os casos passados e a quantidade de ocorrências de cada um deles.

vi) Autonômico – Grau de Autonomia (*Autonomic Degree or Autonomicity*) – A atividade do administrador está associada a definição de objetivos e configuração inicial do arcabouço, seus parâmetros e algoritmos.

Uma visão geral do sistema pode ser vista na Figura 4.8.

4.5 VERIFICAÇÃO DO CUMPRIMENTO DOS REQUISITOS ESPECIFICADOS

Para a elaboração do arcabouço apresentado nesta tese foram especificadas características desejadas (Figura 4.9) para a oferta de autonomia na gerencia de redes de computadores. Nesta seção será realizada uma verificação do cumprimento das características anteriormente especificadas. Para tal verificação dois métodos serão utilizados: utilização dos resultados de simulação – para os itens indicados em cinza (Figura 4.9)– e discussão conceitual do funcionamento do arcabouço – para os itens indicados em branco(Figura 4.9).

i) Tolerância a mudanças – Conforme citado anteriormente o conceito de tolerância a mudanças apresentado nesta tese está associado a sobrevivência da rede mediante a falhas e à capacidade de adaptar-se mesmo com alterações do perfil e/ou volume de tráfego. A sobrevivência foi validada no cenário de simulação 4 (Seção 4.3.4), enquanto a tolerância a mudanças pode ser vista em todas as simulações (Seções 4.3.1, 4.3.2, 4.3.3, 4.3.4 e) uma vez que o tráfego não foi inicialmnete pré-definido. Além disso, nos dois primeiros cenários de simulação (Seções 4.3.1 e 4.3.2), o total de fluxos foi aumentado até 400% do valor inicial.

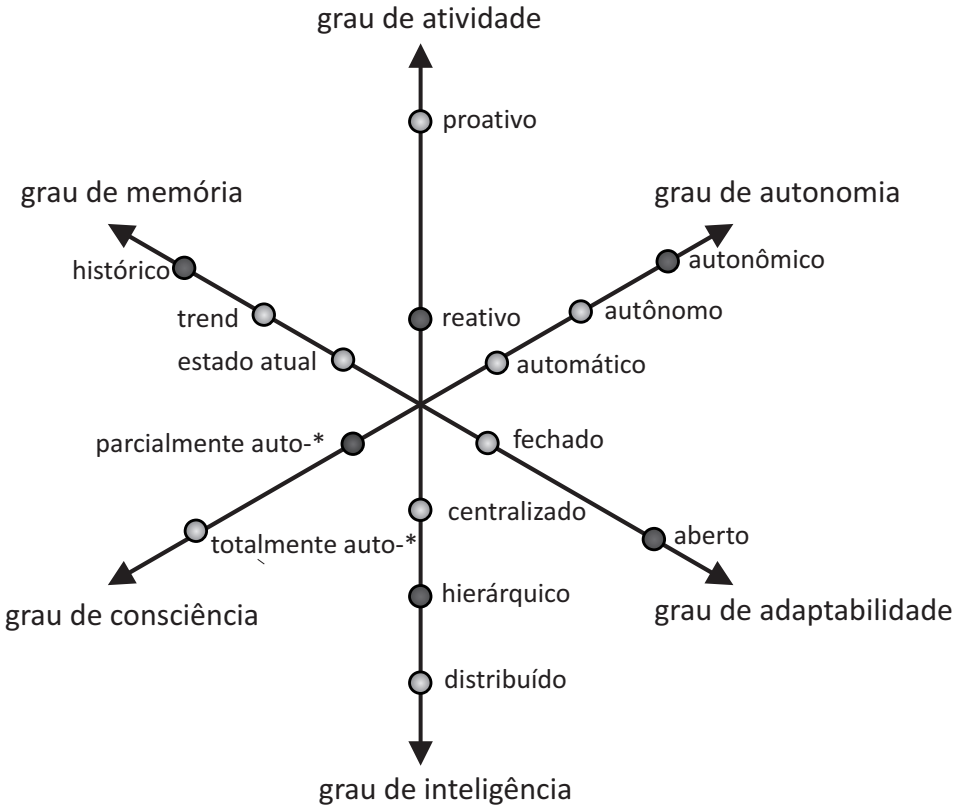


Figura 4.8. Classificação do arcabouço proposto segundo (Samaan; Karmouch, 2009).

ii) Manutenção da QoS – Para validar tal item foi criado um cenário de simulação específico (Seção 4.3.3). Este cenário compara dois parâmetros de QoS – largura de banda média e comprimento médio das alocações realizadas – para redes de cardinalidade igual a 25, 50 e 100 nós. Nesta simulação foi verificado que os parâmetros de QoS foram relativamente pouco afetados em comparação uma redução de tempo considerável com a utilização do arcabouço.

iii) Escalabilidade – A visão de escalabilidade para esta tese compreende a capacidade do arcabouço ofertar soluções mesmo com um aumento significativo em seus parâmetros. Nos cenários de simulação apresentados a escalabilidade está associada ao aumento de

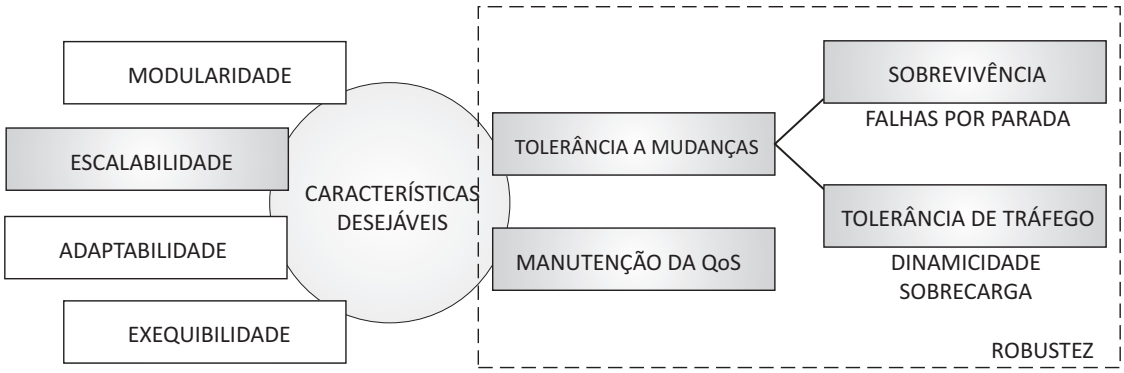


Figura 4.9. Requisitos especificados

tráfego e de quantidade de nós das redes simuladas, conforme visto principalmente nas seções 4.3.1 e 4.3.2.

iv) Modularidade – O arcabouço proposto tem suporte diversos a algoritmos independentemente do objetivo do mesmo. Por exemplo, para o particionamento três algoritmos está disponíveis, para busca de melhor caminho quatro algoritmos podem ser utilizados. Ainda é possível adicionar novos algoritmos caso exista a necessidade por parte do administrador do arcabouço.

v) Adaptabilidade – Não existem restrições técnicas para instanciação do arcabouço em uma rede real, pois o mesmo não está vinculado a tecnologias. A única restrição técnica apresentada é a orientação à fluxos de rede. Ainda assim, não há vínculos específicos com tecnologias que utilizem este conceito como MPLS. O plano de decisão é uma cada independente do plano de execução.

vi) Exequibilidade – Os resultados apresentados durante todo este capítulo apontam que o mesmo pode ser executado. Como o mesmo utiliza a plataforma de simulação do R, pode ser utilizado em qualquer sistema operacional com plataforma Windows ou Linux.

Capítulo 5

A arte de programar consiste em organizar e dominar a complexidade – E. Dijkstra

CONSIDERAÇÕES FINAIS E DESDOBRAMENTOS FUTUROS

Como já discutido, a gerência de redes de computadores é uma tarefa naturalmente complexa, devido a diversos fatores intrínsecos , que dificultam a ação do administrador em grandes domínios. A busca por uma novas soluções de redes em ambientes com alto grau de complexidade impacta diretamente no tempo de resposta inviabilizando a oferta de sistemas que atuem de forma dinâmica. A proposta da computação autonômica é justamente fornecer suporte a gerência de ambientes de alta complexidade como as redes de computadores. Além disso, espera-se que a inserção da autonomia na gerência reduza sensivelmente a atuação humana preferencialmente limitando-a para a especificação dos requisitos esperados para os serviços oferecidos, como por exemplo tempo de resposta e qualidade de serviço.

Uma das contribuições deste trabalho consiste na busca do tratamento da complexidade visando uma gerência escalável e dinâmica de soluções de rede. Isso na medida em que foi verificado, dentro dos limites do nosso conhecimento, que os arcabouços autonômicos disponíveis na literatura fazem um tratamento para a gerência escalável de rede de computadores que pode ser aprimorado.

O estudo de requisitos esperados e atividades necessárias para elaboração de um arcabouço visando a autonomia da gerência de redes foi a primeira contribuição desta tese. Em efeito, as atividades mais críticas para o desenvolvimento de sistemas autonômicos estão associadas as fases de análise e planejamento das soluções.

Um arcabouço com características autonômicas para a gerência escalável de redes foi concebido. Em termos operacionais, o arcabouço visa materializar as propostas de autonomia e escalabilidade de forma dinâmica, permitindo uma validação das mesmas. Seu foco de atuação é justamente nas atividades de análise e planejamento consolidadas no plano de decisão viabilizando assim a aplicação de estratégias operacionais para um melhor tratamento da escalabilidade.

Uma das principais contribuições desta tese foi a integração entre a estratégia utilizada e o motor de busca de soluções realizada no plano de decisão do arcabouço. Em suma, esta integração permite a identificação de novas soluções dinamicamente, e ao mesmo tempo, otimiza o tempo de execução. Para tal, o motor de busca de soluções do arcabouço é baseado em Raciocínio Baseado em Casos *(Case-Based Reasonig – CBR)* e a estratégia – denominada de NPCE *(Network Partitioning Computing Engine)* – consiste no particionamento da rede em função de suas características, do seu perfil de tráfego e dos requisitos especificados pelo usuário (administrador da rede).

O cenário escolhido para a validação das contribuições propostas foi a gerência de qualidade de serviço com foco na alocação de fluxos de redes, com uma atenção especial para a largura de banda utilizada dos enlaces e o comprimento dos fluxos. Os testes foram realizados escalando a quantidade de fluxos das redes (10.000, 20.000 e 40.000 fluxos) que possuíam cardinalidade de 25, 50 e 100 nós, caracterizando um ambiente complexo para testes de escalabilidade.

Os resultados apresentados através de simulações apontam para um tempo de resposta com preservação da qualidade da solução encontrada. O arcabouço também foi avaliado e classificado mediante aos requisitos especificados e à classificação proposta na literatura (Samaan; Karmouch, 2009).

A capacidade do arcabouço em reagir a falhas por parada nos enlaces de rede é um outro resultado relevante decorrente da aplicação das estratégias de escalabilidade desenvolvidas. Observa-se que as estratégias de escalabilidade aplicadas no plano de decisão do arcabouço permitem a sua utilização na área de gerência de falhas, em especial sob a visão da sobrevivência da rede. Embora este não tenha sido o foco do trabalho, observou-se resultados satisfatórios na geração de novas soluções com preservação da qualidade de serviço, sempre que possível, a partir da aplicação de um conjunto de falhas.

5.1 PESQUISAS EM ANDAMENTO E DESDOBRAMENTOS FUTUROS

Em função dos resultados obtidos, uma avaliação de possíveis desdobramentos desta tese segue:

- Análise do limite da sobrevivência mediante a falhas – Um dos desdobramentos em andamento é a análise minuciosa da resposta do arcabouço – materialização das estratégias de tratamento da complexidade e oferta de soluções dinamicamente – mediante falhas por parada ou ataques intencionais à rede. A sobrevivência é a capacidade de um sistema para cumprir sua missão, em tempo hábil, na presença de ameaças como ataques intencionais ou grandes catástrofes naturais – que geralmente produzem muitas falhas – além do modelo de poucas falhas aleatórias comumente associados a problemas em equipamentos ou erros operacionais (Ellison et al., 1997) (Sterbenz et al., 2010).

- Análise de histórico completo da base em CBR – Ao prover uma nova solução, o CBR armazena este novo estado na base de conhecimento para que o mesmo, caso necessário,

seja utilizado novamente. A inserção de novos casos na base geram um histórico de todas as soluções aplicadas pelo arcabouço. Alguns pontos de investigação interessantes podem ser futuramente analisados seguem:

  - Filtro dos casos mais relevantes – À medida que o arcabouço vai provendo novas soluções a complexidade e o tamanho da base de casos aumenta, tornando-se necessário - em algum momento – uma revisão para a identificação dos casos mais relevantes. O objetivo deste filtro é maximizar o tempo de resposta através de um filtro de casos menos relevantes.
  - Análise de transição entre os casos – Caso sejam mapeadas as transições entre os casos escolhidos pelo arcabouço, tal mapeamento de transições pode gerar um gráfico orientado onde os nós representam os casos e os bordos da quantidade de tempo que esta transição ocorreu (Bezerra; Martins, 2008).
  - Análise de Auto-Estabilização – Um ponto de investigação a ser abordado a partir dos resultados desta tese é a análise de auto-estabilização. Com a autonomia, os sistemas passam a propor a solução mais adequada possível para o atendimento dos requisitos. Quando mais forte for esse casamento solução/estado da rede, maior a probabilidade de uma simples alteração do seu estado atual necessitar de uma nova configuração. Dessa forma se faz necessária a escolha de soluções que atendam aos requisitos mas mantenham o ambiente estável o maior tempo possível em resposta a pequenas mudanças no ambiente gerenciado.

- Instanciação do arcabouço – Neste trabalho futuro a proposta é instanciar o arcabouço em termo de uma ferramenta que possa ser utilizada para a gerência efetiva de redes, possibilitando o estudo de ajustes finos de acordo com as peculiaridades do ambiente aplicado.

- Análise de planejamento de capacidade – A base de conhecimento gerada pelo CBR, a estratégica do NPCE e os algoritmos utilizados podem ser utilizados para um planejamento de capacidade para uma rede de computadores. Todos os dados armazenados servem de base para uma análise *offline* entre informações da topologia e do tráfego para estimação de alterações topológicas visando uma melhor conformidade entre tais informações. Para tal seriam utilizadas a computação das Fases 2 e 3 do plano de decisão.

REFERÊNCIAS BIBLIOGRÁFICAS

Aamodt, A.; Plaza, E. Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Commun.*, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 7, n. 1, p. 39–59, 1994. ISSN 0921-7126.

Agarwal, M. et al. Automate: enabling autonomic applications on the grid. In: *Autonomic Computing Workshop, 2003*. [S.l.: s.n.], 2003. p. 48–57.

Aib, I. et al. Analysis of policy management models and specification languages. In: ____. Norwell, MA, USA: Kluwer Academic Publishers, 2003. p. 26–50. ISBN 1-4020-7616-9. Disponível em: <http://dl.acm.org/citation.cfm?id=963962.963966>.

Algomine, N. Introduction to autonomic concepts applied to future self-managed networks. In: Algomine, N. (Ed.). *Autonomic Network Management Principles Form Concepts to Applications*. [S.l.]: Elsevier, 2011. p. 1–26. ISBN 978-0123821904.

Avizienis, A. et al. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, v. 1, n. 1, p. 11 – 33, jan.-march 2004. ISSN 1545-5971.

Balasubramaniam, S. et al. A multi-layered approach towards achieving survivability in autonomic network. In: *Telecommunications and Malaysia International Conference on Communications, 2007. ICT-MICC 2007. IEEE International Conference on*. [S.l.: s.n.], 2007. p. 360 –365.

Bellman, R. On a Routing Problem. *Quarterly of Applied Mathematics*, v. 16, p. 87–90, 1958.

Bezerra, R. M. S.; Martins, J. S. A sense and react plane structured autonomic model suitable for qos management. In: *Proceedings of 5th International IEEE Workshop on Management of Ubiquitous Communications and Services*. [S.l.: s.n.], 2008.

Bezerra, R. M. S.; Martins, J. S. B. Functional decoupling principle applied to network device and qos management. In: *Proceedings of 7ème Colloque Francophone of Gestion of Réseaux et of Services*. [S.l.: s.n.], 2006. v. 1, p. 47–58.

Bezerra, R. M. S.; Martins, J. S. B. Introducing case-based reasoning approach in autonomic model suitable for qos management. In: *LAACS*. [S.l.: s.n.], 2008.

Bezerra, R. M. S.; Martins, J. S. B. Network partitioning and self-sizing methods for qos management with autonomic characteristics. In: *APNOMS*. [S.l.: s.n.], 2009. p. 151–160.

Bezerra, R. M. S.; Martins, J. S. B. A policy-based autonomic model suitable for quality of service management. *Journal of Networks (JNW)*, v. 4, n. 6, p. 495–504, 2009.

Boutaba, R.; Aib, I. Policy-based management: A historical perspective. *Journal Networks Systems Management*, Plenum Press, New York, NY, USA, v. 15, n. 4, p. 447–480, 2007. ISSN 1064-7570.

Boutaba, R.; Xiao, J.; Zhang, Q. Toward autonomic networks: Knowledge management and self-stabilization. In: Zhang, Y.; Yang, L. T.; Denko, M. K. (Ed.). *Autonomic Computing and Networking*. Springer US, 2009. p. 239–260. ISBN 978-0-387-89828-5. 10.1007/978-0-387-89828-510. Disponível em: <http://dx.doi.org/10.1007/978-0-387-89828-510>.

Chan, K. et al. *COPS Usage for Policy Provisioning (COPS-PR)*. United States: RFC Editor, 2001.

Cheng, Y. et al. A generic architecture for autonomic service and network management. *Comput. Commun.*, Butterworth-Heinemann, Newton, MA, USA, v. 29, n. 18, p. 3691–3709, 2006. ISSN 0140-3664.

Cherkassky, B. V.; Goldberg, A. V.; Radzik, T. Shortest paths algorithms: theory and experimental evaluation. In: *SODA '94: Proceedings of the fifth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1994. p. 516–525. ISBN 0-89871-329-3.

Clark, D. et al. *New Arch: Future Generation Internet Architecture*. [S.l.], 2004. Disponível em: <http://www.isi.edu/newarch/iDOCS/final.finalreport.pdf>.

Clark, D. D. et al. A knowledge plane for the internet. In: *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2003. (SIGCOMM '03), p. 3–10. ISBN 1-58113-735-4. Disponível em: <http://doi.acm.org/10.1145/863955.863957>.

Clauset, A.; Newman, M.; Moore, C. Finding community structure in very large networks. *Physical Review E*, v. 70, 2004. Disponível em: <http://dx.doi.org/10.1103/PhysRevE.70.066111>.

Computing, A. An architectural blueprint for autonomic computing. *Quality*, Citeseer, v. 36, n. June, p. 34, 2006. Disponível em: $<http://users.encs.concordia.ca/ac/ac-resources/AC_Blueprint_White_Paper_4th.pdf>$.

Csardi, G. *igraph Manual*. Lausanne, Switzerland, 2010. Disponível em: <http://igraph.sourceforge.net/>.

Davy, S. et al. Policy-based architecture to enable autonomic communications - a position paper. In: *Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE*. [S.l.: s.n.], 2006. v. 1, p. 590–594.

Denko, M. K.; Yang, L. T.; Zhang, Y. *Autonomic Computing and Networking*. [S.l.]: Springer Publishing Company, Incorporated, 2009. ISBN 0387898271, 9780387898278.

Derbel, H. et al. A utility-based autonomic architecture to support qoe quantification in ip networks. In: Algomine, N. (Ed.). *Autonomic Network Management Principles Form Concepts to Applications*. [S.l.]: Elsevier, 2011. p. 67–99. ISBN 978-0123821904.

Derbel, H.; Agoulmine, N.; Salaün, M. Anema: Autonomic network management architecture to support self-configuration and self-optimization in ip networks. *Comput. Netw.*, Elsevier North-Holland, Inc., New York, NY, USA, v. 53, p. 418–430, February 2009. ISSN 1389-1286. Disponível em: <http://dl.acm.org/citation.cfm?id=1501029.1501235>.

Dijkstra, E. W. A note on two problems in connexion with graphs. *Numerische Mathematik*, v. 1, n. 1, p. 269–271, December 1959. Disponível em: <http://dx.doi.org/10.1007/BF01386390>.

Dijkstra, E. W. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, ACM, New York, NY, USA, v. 17, n. 11, p. 643–644, 1974. ISSN 0001-0782.

Dobson, S. et al. A survey of autonomic communications. *ACM Trans. Auton. Adapt. Syst.*, ACM, New York, NY, USA, v. 1, n. 2, p. 223–259, 2006. ISSN 1556-4665.

Ellison, R. et al. *Survivable Network Systems: An Emerging Discipline*. [S.l.], 1997. Disponível em: <http://www.sei.cmu.edu/library/abstracts/reports/97tr013.cfm>.

Elmasri, R.; Navathe, S. B. *Fundamentals of database systems*. Redwood City, CA, USA: Benjamin-Cummings Publishing Co., Inc., 2003. ISBN 0-8053-0145-3.

Farha, R.; Leon-garcia, A. Blueprint for an autonomic service architecture. In: *Autonomic and Autonomous Systems, 2006. ICAS '06. 2006 International Conference on*. [S.l.: s.n.], 2006. p. 16–16.

Feeney, M.; Frisby, R. Autonomic management of ubiquitous computing environments. 2006. Disponível em: <http://repository.wit.ie/879/>.

Gaïti, D. et al. (Ed.). *Autonomic Networking, First International IFIP TC6 Conference, AN 2006, Paris, France, September 27-29, 2006, Proceedings*, v. 4195 de *Lecture Notes in Computer Science*, (Lecture Notes in Computer Science, v. 4195). [S.l.]: Springer, 2006. ISBN 3-540-45891-3.

Ganek, A. G.; Corbi, T. A. The dawning of the autonomic computing era. *IBM Systems Journal*, IBM Corp., Riverton, NJ, USA, v. 42, n. 1, p. 5–18, 2003. ISSN 0018-8670.

Gelenbe, E. Steps toward self-aware networks. *Commun. ACM*, ACM, New York, NY, USA, v. 52, n. 7, p. 66–75, 2009. ISSN 0001-0782.

Gittler, J. *Review of Sociology: Analysis of a Decade*. J. Wiley, 1975. Disponível em: <http://books.google.com.br/books?id=ILFUMAEACAAJ>.

Goldberg, A. V.; Tarjan, R. E. A new approach to the maximum flow problem. In: *Proceedings of the eighteenth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM, 1986. (STOC '86), p. 136–146. ISBN 0-89791-193-8. Disponível em: <http://doi.acm.org/10.1145/12130.12144>.

Graupner, S.; Andrzejak, A. Adaptive control overlay for service management. In: *Workshop on the Design of Self-Managing Systems, International Conference on Dependable Systems and Networks (DSN), 2003*. [S.l.: s.n.], 2003.

Horn, P. *Autonomic computing: IBM's Perspective on the State of Information Technology*. [S.l.]: IBM, 2001.

Jennings, B. et al. Towards autonomic management of communications networks. *Communications Magazine, IEEE*, v. 45, n. 10, p. 112–121, October 2007. ISSN 0163-6804.

Johnson, D. B. Efficient algorithms for shortest paths in sparse networks. *J. ACM*, ACM, New York, NY, USA, v. 24, n. 1, p. 1–13, 1977. ISSN 0004-5411.

Jrad, Z. et al. Artificial intelligence techniques in the dynamic negotiation of qos: A user interface for the internet new generation. In: Gaïti, D. et al. (Ed.). *Autonomic Networking*. [S.l.]: Springer, 2006. (Lecture Notes in Computer Science, v. 4195), p. 146–158. ISBN 3-540-45891-3.

Kephart, J. O. Research challenges of autonomic computing. In: *ICSE '05: Proceedings of the 27th international conference on Software engineering*. New York, NY, USA: ACM, 2005. p. 15–22. ISBN 1-59593-963-2.

Kephart, J. O.; Chess, D. M. The vision of autonomic computing. *Computer*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 36, n. 1, p. 41–50, 2003. ISSN 0018-9162.

Kephart, J. O.; Das, R. Achieving self-management via utility functions. *IEEE Internet Computing*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 11, p. 40–48, January 2007. ISSN 1089-7801. Disponível em: <http://dx.doi.org/10.1109/MIC.2007.2>.

Khan, M. J.; Awais, M. M.; Shamail, S. Enabling self-configuration in autonomic systems using case-based reasoning with improved efficiency. In: *ICAS '08: Proceedings of the Fourth International Conference on Autonomic and Autonomous Systems*. Washington, DC, USA: IEEE Computer Society, 2008. p. 112–117. ISBN 978-0-7695-3093-2.

Lupu, E.; Sloman, M. Conflicts in policy-based distributed systems management. *Software Engineering, IEEE Transactions on*, v. 25, n. 6, p. 852 –869, nov/dec 1999. ISSN 0098-5589.

Martin-flatin, J. P.; Jakobson, G.; Lewis, L. Event correlation in integrated management: Lessons learned and outlook. *J. Netw. Syst. Manage.*, Plenum Press, New York, NY, USA, v. 15, n. 4, p. 481–502, 2007. ISSN 1064-7570.

Martin-flatin, J.-P.; Srivastava, D.; Westerinen, A. Iterative multi-tier management information modeling. *Communications Magazine, IEEE*, v. 41, n. 12, p. 92 – 99, dec. 2003. ISSN 0163-6804.

Meyer, J. F. Performability: a retrospective and some pointers to the future. *Performance Evaluation*, v. 14, n. 3-4, p. 139 – 156, 1992. ISSN 0166-5316.

Morettin, P.; Bussab, W. *Estatística Básica*. [S.l.]: Saraiva, 2004.

Mortier, R.; Isaacs, R.; Barham, P. Anemone: using end-systems as a rich network management platform. *Applications, Technologies, Architectures, and Protocols for Computer Communication*, ACM Press New York, NY, USA, p. 203–204, 2005.

Murch, R. Autonomic attributes and the grand challenge. In: Management, I. P. S. I. (Ed.). *Autonomic computing*. [S.l.]: Prentice Hall PTR, 2004. p. 231–259. ISBN 978-0131440258.

Newman, M. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, v. 74, 2006. Disponível em: <http://dx.doi.org/10.1103/PhysRevE.74.036104>.

Newman, M. *Networks: An Introduction*. New York, NY, USA: Oxford University Press, Inc., 2010. ISBN 0199206651, 9780199206650.

Newman, M.; Girvan, M. Finding and evaluating community structure in networks. *Physical Review E*, v. 69, 2004. Disponível em: <http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0308217>.

Oreizy, P. et al. An architecture-based approach to self-adaptive software. *IEEE Intelligent Systems*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 14, p. 54–62, May 1999. ISSN 1541-1672. Disponível em: <http://dx.doi.org/10.1109/5254.769885>.

Parashar, M. *Autonomic Computing: Concepts, Infrastructure, and Applications / Editor(s): Manish Parashar and Salim Hariri*. Bristol, PA, USA: Taylor & Francis, Inc., 2007. ISBN 1420009354.

Parashar, M.; Hariri, S. Autonomic Computing: An Overview. In: . [s.n.], 2005. p. 257–269. Disponível em: <http://dx.doi.org/10.1007/11527800_20>.

Pióro, M.; Medhi, D. *Routing, Flow, and Capacity Design in Communication and Computer Networks*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004. ISBN 0125571895.

Pons, P.; Latapy, M. Computing communities in large networks using random walks. *Journal of Graph Algorithms and Applications*, v. 10, n. 2, p. 191–218, 2006.

Pras, A. et al. Key research challenges in network management. *Communications Magazine, IEEE*, v. 45, n. 10, p. 104–110, October 2007. ISSN 0163-6804.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2011. ISBN 3-900051-07-0. Disponível em: <http://www.R-project.org>.

Samaan, N.; Karmouch, A. A case-based reasoning approach for automated management in policy-based networks. In: Sahai, A.; Wu, F. (Ed.). *Utility Computing*. [S.l.]: Springer Berlin - Heidelberg, 2004, (Lecture Notes in Computer Science, v. 3278). p. 76–87. ISBN 9783540236313.

Samaan, N.; Karmouch, A. An automated policy-based management framework for differentiated communication systems. *Selected Areas in Communications, IEEE Journal on*, v. 23, n. 12, p. 2236–2247, Dec. 2005. ISSN 0733-8716.

Samaan, N.; Karmouch, A. Towards autonomic network management: an analysis of current and future research directions. *IEEE Communications Surveys Tutorials*, IEEE, v. 11, n. 3, p. 22–36, 2009. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5208731>.

Schneider, M. Self-stabilization. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 25, p. 45–67, March 1993. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/151254.151256>.

Serrano, J. M. et al. Policy-based management and context modelling contributions for supporting services in autonomic systems. In: Gaïti, D. et al. (Ed.). *Autonomic Networking*. [S.l.]: Springer, 2006. (Lecture Notes in Computer Science, v. 4195), p. 172–187. ISBN 3-540-45891-3.

Shiu, S.; Pal, S. K. *Foundations of Soft Case-Based Reasoning*. [S.l.]: John Wiley & Sons, 2004. ISBN 0471086355.

Shiu, S.; Pal, S. K. *Foundations of Soft Case-Based Reasoning*. [S.l.]: John Wiley & Sons, 2004. ISBN 0471086355.

Siekkinen, M. et al. Beyond the future internet–requirements of autonomic networking architectures to address long term future networking challenges. In: *Proceedings of the 11th IEEE International Workshop on Future Trends of Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2007. (FTDCS '07), p. 89–98. ISBN 0-7695-2810-4. Disponível em: <http://dx.doi.org/10.1109/FTDCS.2007.14>.

Sterbenz, J. P. G. et al. Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. *Comput. Netw.*, Elsevier North-Holland, Inc., New York, NY, USA, v. 54, p. 1245–1265, June 2010. ISSN 1389-1286. Disponível em: <http://dx.doi.org/10.1016/j.comnet.2010.03.005>.

Strassner, J. *Policy-Based Network Management: Solutions for the Next Generation (The Morgan Kaufmann Series in Networking)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003. ISBN 1558608591.

Strassner, J. Using autonomic principles to manage converged services in next generation networks. In: *Autonomic Computing, 2006. ICAC '06. IEEE International Conference on*. [S.l.: s.n.], 2006. p. 1–1.

Strassner, J. The design of the focale autonomic networking architecture. In: Algomine, N. (Ed.). *Autonomic Network Management Principles Form Concepts to Applications*. [S.l.]: Elsevier, 2011. p. 231–259. ISBN 978-0123821904.

Strassner, J.; Agoulmine, N.; Lehtihet, E. Focale - a novel autonomic networking architecture. *Multimedia Systems*, v. 3, n. 1, p. 64–79, 2006. Disponível em: <http://eprints.wit.ie/189/>.

Strassner, J.; Hong, J.-K.; Kang, K. A framework for modeling and reasoning about network management resources and services to support information reuse. In: *Information Reuse Integration, 2009. IRI '09. IEEE International Conference on*. [S.l.: s.n.], 2009. p. 85 –90.

Strassner, J. et al. Modelling context for autonomic networking. In: *Network Operations and Management Symposium Workshops, 2008. NOMS Workshops 2008. IEEE*. [S.l.: s.n.], 2008. p. 299 –308.

Strassner, J. et al. An autonomic architecture to manage ubiquitous computing networks and applications. In: *Ubiquitous and Future Networks, 2009. ICUFN 2009. First International Conference on*. [S.l.: s.n.], 2009. p. 116 –121.

Strassner, J. et al. The design of a novel context-aware policy model to support machine-based learning and reasoning. *Cluster Computing*, Kluwer Academic Publishers, Hingham, MA, USA, v. 12, n. 1, p. 17–43, 2009. ISSN 1386-7857.

Tesauro, G. Reinforcement learning in autonomic computing: A manifesto and case studies. *IEEE Internet Computing*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 11, p. 22–30, January 2007. ISSN 1089-7801. Disponível em: <http://dx.doi.org/10.1109/MIC.2007.21>.

Triola, M. F. *Elementary Statistics*. [S.l.]: Pearson, 2008.

Truszkowski, W. F. et al. Autonomous and autonomic systems: a paradigm for future space exploration missions. *IEEE Transactions on Systems Man and Cybernetics Part C Applications and Reviews*, v. 36, n. 3, p. 279–291, 2006.

Verma, D. C. *Policy-Based Networking: Architecture and Algorithms*. Thousand Oaks, CA, USA: New Riders Publishing, 2000. ISBN 1578702267.

Want, R.; Pering, T.; Tennenhouse, D. Comparing autonomic and proactive computing. *IBM Syst. J.*, IBM Corp., Riverton, NJ, USA, v. 42, p. 129–135, January 2003. ISSN 0018-8670. Disponível em: <http://dx.doi.org/10.1147/sj.421.0129>.

Yahia, I. G. B.; Bertin, E.; Crespi, N. Towards autonomic management for next generation services. In: *Proceedings of the International conference on Networking and Services (ICNS '06)*. Washington, DC, USA: IEEE Computer Society, 2006. p. 38. ISBN 0-7695-2622-5.

Apêndice

A

LISTA DE PUBLICAÇÕES

Neste anexo será apresentado o conjunto de publicações e trabalhos submetidos relacionados ao escopo desta tese. Durante o desenvolvimento deste trabalho, os resultados parciais foram submetidos para alguns eventos visando a verificação das abordagens utilizadas e suas respectivas críticas. Os resultados definitivos e publicados colaboraram para a consolidação da tese. Uma visão cronológica dos artigos pode ser vista abaixo (Figura A.1)

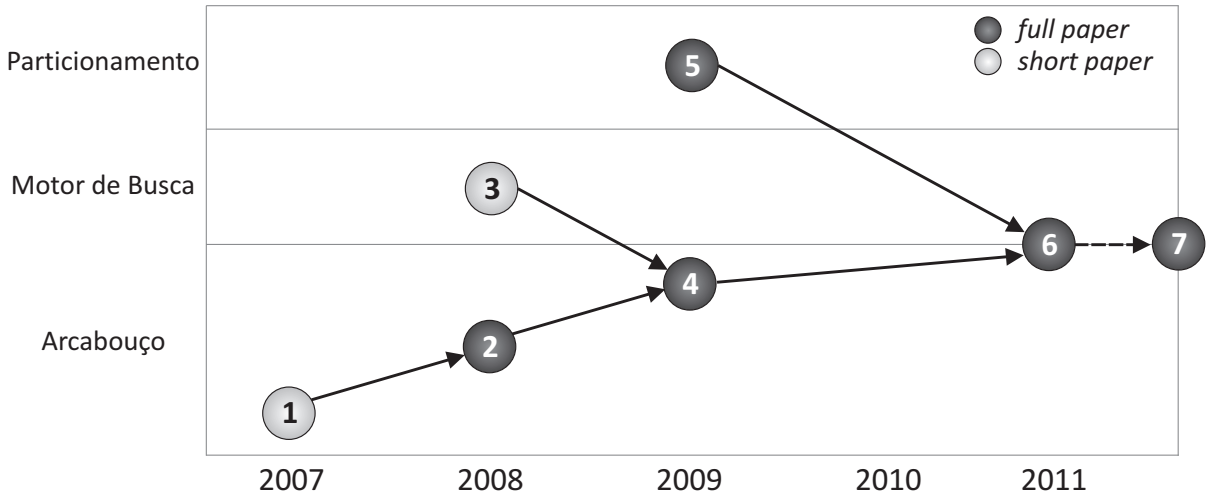


Figura A.1. Visão cronológica das publicações

1 BEZERRA, R. M. S. ; MARTINS, J. S. B. . Using Policy-Based Framework to Support QoS Autonomic Management. In: 2nd Latin American Autonomic Computing Symposium (LAACS2007), 2007, Petrópolis, Brasil.

2 BEZERRA, R. M. S. ; MARTINS, J. S. B. . A Sense and React Plane Structured Autonomic Model Suitable for Quality of Service (QoS) Management. In: 5th International IEEE Workshop on Management of Ubiquitous Communications and Services (MUCS2008), 2008, Salvador, Brasil.

3 BEZERRA, R. M. S. ; MARTINS, J. S. B. . Introducing Case-Based Reasoning Approach in Autonomic Model Suitable for QoS Management. In: 3rd Latin American Autonomic Computing Symposium (LAACS2008), 2008, Gramado, Brasil.

4 BEZERRA, R. M. S. ; MARTINS, J. S. B. A Policy-Based Autonomic Model Suitable for Quality of Service Management. Journal of Networks, v. 4, p. 495-504, 2009. DOI: http://dx.doi.org/10.4304/jnw.4.6.495-504

5 BEZERRA, R. M. S. ; MARTINS, J. S. B. . Network Partitioning and Self-sizing Methods for QoS Management with Autonomic Characteristics. In: 11th Asia-Pacific Network Operations and Management Symposium (APNOMS2009), 2009, Jeju, Korea. Lecture Notes in Computer Science (LNCS) v. 5787. p. 151-160. DOI: http://dx.doi.org/10.1007/978-3-642-04492-2_16

6 BEZERRA, R. M. S. ; MARTINS, J. S. B. . Network Self-Management with a Partitioning Method based on Network Density and Traffic Matrix. In: 13th Asia-Pacific Network Operations and Management Symposium (APNOMS2011), 2011, Taipei, Taiwan. 13th Asia-Pacific Network Operations and Management Symposium (APNOMS2011), 2011.

7 Dealing with the Scalability Issue in Network Self-Management – Submissão ao *Journal of Communications and Networks*.

Inicialmente o primeiro trabalho publicado no LAACS2007 (*shortpaper*), assim como os artigos do MUCS2008 e LAACS2008 (*shortpaper*) e JNW2009 referem-se ao modelo.

Durante a fase de pesquisa, foi verificada a necessidade de provisão de escalabilidade para modelos autonômicos, independentemente do arcabouço utilizado. Dessa forma, no final de 2008 a pesquisa foi direcionada para o particionamento em domínios com o objetiv de integrar esta estratégia ao arcabouço, proposta apresentada no APNOMS2009.

Em dezembro de 2009, recebemos o convite do *Journal of Communication and Computer* (ISSN 1548-7709) para detalhamento dos resultados do artigo publicado no APNOMS2009. Em dezembro também foi submetido para o *Computer Networks* (ISSN 1389-1286) o artigo *A Strategy of Density-Based Partitioning for Scalability Problem in Network Self-Management* que apresenta resultados quantitativos e qualitativos do algoritmo de particionamento.

Atualmente foi submetido um novo artigo com foco na análise da robustez do arcabouço integrando o particionamento e o motor de busca de soluções. Este artigo consiste em um resumo desta tese. O *journal* escolhido foi o *Journal of Communications and Networks*.

Apêndice

B

AMBIENTE DE SIMULAÇÃO

B.1 R

R é uma linguagem e ambiente para computação estatística que fornece uma ampla variedade de estatísticas (modelagem linear e não linear, testes estatísticos clássicos, análise de séries temporais, classificação, clustering, dentre outras) e técnicas gráficas, além de ser altamente extensível. R está disponível como Software Livre sob os termos da *Free Software Foundations GNU General Public License* em forma de código fonte. Ele compila e roda em uma grande variedade de plataformas UNIX e sistemas similares (incluindo FreeBSD e Linux), Windows e MacOS.

O ambiente R um conjunto integrado de instalações de software para manipulação de dados, cálculo e apresentação gráfica que inclui:

- uma de dados eficaz manuseio e instalação de armazenamento,
- um conjunto de operadores para cálculos sobre matrizes, matrizes em particular,
- uma coleção grande, coerente e integrada de ferramentas intermediárias para análise de dados,
- instalações gráfica para análise de dados e exibir ou no ecrã ou na via impressa e
- uma bem desenvolvida, linguagem de programação simples e eficaz que inclui condicionais, loops, funções definidas pelo usuário e recursiva de entrada e saída de instalações.

O termo ambiente destina-se a caracterizá-la como um sistema totalmente planejada e coerente, em vez de um acréscimo incremental de ferramentas muito específicas e inflexíveis, como é frequentemente o caso com outro software de análise de dados.

R tem o seu formato de documentação própria LaTeX-like, que é usado para fornecer documentação completa, tanto on-line em vários formatos e em via impressa.

Muitos usuários pensam de R como um sistema de estatísticas. Preferimos pensar nele de um ambiente no qual as técnicas estatísticas são implementadas. R pode ser estendido (facilmente) através de pacotes. Há cerca de oito pacotes fornecidos com a distribuição R e muitos mais estão disponíveis através da família CRAN de sítios da Internet que cobrem uma vasta gama de estatísticas modernas. Dentre estes pacotes destaco o *igraph* para o desenvolvimento desta tese.

B.2 IGRAPH

IGRAPH é um pacote de software livre para criação e manipulação de gráficos sem direção e direcionado. Ele inclui implementações para problemas gráfico clássico teoria como mínimo árvores geradoras e fluxo de rede, e também implementa algoritmos para alguns métodos de análise de rede recentes, como a busca da estrutura da comunidade.

A aplicação eficaz de IGRAPH lhe permite lidar com gráficos, com milhões de vértices e arestas. A regra chave é que se o grafo se encaixa na memória física, então IGRAPH permite um desempenho satisfatório. Além disso um conjunto de características do IGRAPH são bastante desejáveis no desenvolvimento desta tese:

- contém funções para a geração de grafos regulares e aleatórios de acordo com muitos algoritmos e modelos da literatura sobre teoria dos grafos.
- fornece rotinas para manipulação de gráficos, adição e remoção de arestas e vértices.
- é possível atribuir variáveis numéricas ou textuais para os vértices e/ou arestas do grafo, como peso ou capacidade dos enlaces.
- fornece tipos de dados para implementar o seu próprio algoritmo em C, R, Python ou Ruby.
- fornece também algoritmos de detecção de comunidades usando pesquisas desenvolvidas recentemente

O IGRAPH pode ser instalado como:

- uma biblioteca C, sendo útil se for necessário usá-lo na sua C/C++ projetos, ou deseja implementar sua análise própria rede ou modelo em C/C++ utilizando as estruturas de dados e funções fornecidas pelo IGRAPH.
- um pacote de R, ou seja, um pacote de extensão para o projeto GNU R para computação estatística. A flexibilidade da linguagem R e sua riqueza em métodos estatísticos permitem adicionar uma grande produtividade para o IGRAPH, com uma penalidade de velocidade muito pequena. No contexto desta tese o IGRAPH está sendo utilizado como um pacote do R.

- Python (módulo de extensão) permitindo combinar o a facilidade e enorme conjunto de funções Python disponíveis, também uma penalidade de velocidade pequena.