Planejamento Regional Adaptativo em Sistemas Self-Adaptive de Larga Escala

Eudes S. Andrade, Gabriel F. Sousa, Sandro S. Andrade

Grupo de Pesquisa em Sistemas Distribuídos, Otimização, Redes e Tempo-Real (GSORT)
Instituto Federal de Educação, Ciência e Tecnologia da Bahia (IFBA)
Av. Araújo Pinho, no 39 – Canela – Salvador/BA – CEP: 40.110-150

{eudesandrade, gabrielferreira, sandroandrade}@ifba.edu.br

Resumo. *Com a crescente adoção de sistemas distribuídos de larga escala, tais como clusters para cloud computing, as atividades de implantação e configuração destas soluções se tornam consideravelmente mais complexas. Uma solução promissora é dotar tais sistemas com capacidades de autogerenciamento e diversos padrões de projeto para Sistemas Self-Adaptive estão disponíveis na literatura. Alguns são mais efetivos para operação em ambientes que sugerem a adoção de arquiteturas mais centralizadas. Outros apresentam melhores resultados ao operar de forma mais distribuída. Em ambientes altamente dinâmicos e incertos, no entanto, é difícil encontrar uma única solução efetiva para todos os cenários experimentados. Este trabalho apresenta um modelo adaptativo para implementação de Sistemas Self-Adaptive de larga escala que realizam autogerenciamento com diferentes graus de centralização. Os resultados obtidos mostram que o modelo apresenta desempenho satisfatório tanto em situações onde o sistema monitorado e o ambiente sugerem a adoção de topologias mais centralizadas, quanto em situações onde padrões mais descentralizados apresentam melhor custo-benefício.*

Abstract. *As large-scale distributed systems such as clusters for cloud computing become increasingly adopted, new challenges are posed when deploying and configuring such systems. A promising solution is to endow such systems with self-management capabilities and a number of design patterns for Self-Adaptive Systems is currently available. Some of those patterns are effective when operating in environments that require the adoption of centralized architectures, while others yield better results when controlling large-scale distributed systems. For highly dynamic and uncertain environments, though, a single solution is unlikely to provide effective control in every distinct scenario experienced by the system. This work presents an adaptive model for endowing large-scale Self-Adaptive Systems with self-management capabilities operating with a varying centralization degree. Results show that the proposed model presents improved performance in situations demanding both centralized and decentralized control topologies.*

1. Introdução

Uma série de requisitos e características associados aos sistemas distribuídos modernos tem modificado a forma como tais aplicações são projetadas, desenvolvidas e avaliadas.

Dentre tais características, destacam-se: a utilização de dispositivos multiprocessados, a disponibilidade de paralelismo em soluções altamente distribuídas, a necessidade de integração com outros sistemas e a capacidade de autogerenciamento em ambientes incertos [Northrop 2013]. A despeito da disponibilidade de diversas estratégias de Engenharia de Software para gerência da complexidade gerada por estes desafios, acredita-se que a capacidade humana de compreensão e manipulação de artefatos de software será um fator limitante em um futuro próximo [Huebscher and McCann 2008, Kephart and Chess 2003].

Tendo em vista que a maior parte dos desafios acima apresentados impactam diretamente os requisitos não-funcionais da aplicação – e que estes são amplamente influenciados por decisões arquiteturais tomadas em tempo de projeto – pesquisas recentes indicam que a transferência de decisões de projeto para o tempo de execução (*runtime*) é uma solução promissora no gerenciamento desta complexidade. Para tanto, os sistemas computacionais precisam apresentar alguma capacidade de autogerencimento [Huebscher and McCann 2008, Kephart and Chess 2003, Salehie and Tahvildari 2009]. Adicionalmente, características de Computação Ubíqua comumente presentes, por exemplo, em aplicações da Internet das Coisas e *Cyber-Physical Systems*, frequentemente demandam a necessidade de autogerenciamento. Um Sistema *Self-Adaptive* (SSA) é aquele que avalia o seu próprio comportamento e o modifica quando esta avaliação indica que: *i)* o seu propósito principal não está sendo efetivamente cumprido; ou *ii)* uma melhor funcionalidade e/ou desempenho pode ser alcançado [DARPA 1997].

Muitas arquiteturas para autogerenciamento estão atualmente disponíveis na literatura [Patikirikorala et al. 2012]. Ainda assim, a maioria delas apresenta desempenho satisfatório apenas em sistemas centralizados [Weyns 2010]. A dificuldade em alcançar autogerenciamento efetivo em ambientes descentralizados decorre do *trade-off* comumente encontrado em sistemas distribuídos: conhecimento do estado global do sistema vs. *overhead* de comunicação. Pode-se ilustrar esse *trade-off* analisando *clusters* para suporte a *cloud computing*. Em tais situações, quanto maior o *overhead* de comunicação (e, portanto, o conhecimento sobre o estado global do sistema), maior será a capacidade do sistema coordenar adaptações em caso de mudanças no ambiente. Por outro lado, um maior consumo de recursos pode diminuir a utilização útil do *cluster* para fins de computação. O objetivo da adoção de arquiteturas para autogerenciamento descentralizado, em tais cenários, é reagir ao *trade-off* que envolve a capacidade de atendimento da *cloud* frente às diversas configurações de ambiente apresentadas ao longo do tempo.

Este trabalho apresenta o projeto e avaliação de um modelo adaptativo para autogerenciamento em SSA de larga escala. Esse modelo prevê a reconfiguração dinâmica da topologia e forma de comunicação entre os múltiplos componentes locais de controle. Com esta abordagem, decisões de projeto acerca da topologia de controle a ser adotada são levadas para o tempo de execução. Com isso, obtém-se desempenho satisfatório de controle tanto em situações onde o sistema monitorado e o ambiente sugerem a adoção de topologias mais centralizadas, quanto em situações onde padrões completamente descentralizados apresentam um melhor *trade-off*. O mecanismo de reconfiguração dinâmica da arquitetura de controle aqui proposto foi modelado e avaliado via simulação de eventos discretos. Resultados indicam que houve uma economia estatisticamente significante de recursos computacionais ao utilizar o modelo proposto.

O restante deste trabalho está organizado como segue. A Seção 2 apresenta os

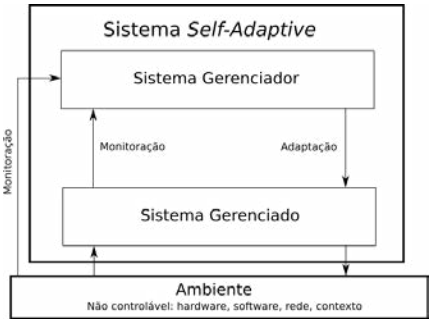

Figura 1. Elementos arquiteturais principais de um Sistema Self-Adaptive.

fundamentos sobre SSA e seus principais desafios. A Seção 3 explica o mecanismo para reconfiguração dinâmica de arquiteturas de controle aqui proposto. A Seção 4 apresenta detalhes sobre as avaliações via simulação realizadas, enquanto a Seção 5 discute e analisa os resultados obtidos. A Seção 6 apresenta os principais trabalhos relacionados e, finalmente, as conclusões e sugestões de trabalhos futuros são discutidos na Seção 7.

2. Sistemas Self-Adaptive de Larga Escala

O autogerenciamento é uma estratégia de projeto utilizada para os mais diversos fins. A despeito disto, a maior parte destas soluções realiza a migração de decisões – anteriormente tomadas em tempo de projeto – para o tempo de execução. Com este artifício, busca-se minimizar possíveis inflexibilidades/ineficiências decorrentes de comprometimentos prévios com determinadas decisões arquiteturais. A ocorrência de situações não previstas pelo arquiteto de *software* é consequência, sobretudo, do dinamismo dos ambientes de execução das aplicações.

Uma definição de SSA comumente adotada na literatura é aquela proposta em [DARPA 1997]: um SSA é aquele que avalia o seu próprio comportamento e o modifica quando esta avaliação indica que: *i)* o seu propósito principal não está sendo efetivamente cumprido; ou *ii)* uma melhor funcionalidade e/ou desempenho podem ser alcançados [DARPA 1997]. Um SSA é geralmente caracterizado pela presença de dois elementos arquiteturais principais, ilustrados na Figura 1. O primeiro – denominado **sistema gerenciado** – é o elemento responsável pela implementação das regras de negócio da aplicação. O segundo elemento – denominado **sistema gerenciador** – é responsável pela implementação das ações necessárias para a realização das adaptações.

Embora diferentes mecanismos para autogerenciamento tenham sido propostos nos últimos anos, a maior parte deles baseia-se na utilização de algum tipo de *loop* de adaptação. Uma arquitetura de referência amplamente adotada no projeto destes *loops* de adaptação é a arquitetura MAPE-K [Kephart and Chess 2003]. Esta arquitetura define a execução, em sequência, das seguintes fases: **M**onitoramento, **A**nálise, **P**lanejamento e **E**xecução – com o auxílio de uma base de conhecimento (**K***nowledge Base*). A fase de monitoramento é responsável pela obtenção de informações sobre o estado atual do sistema/ambiente. A atividade de análise realiza a avaliação das informações obtidas na fase de monitoramento. Após a avaliação destes dados, delibera-se pela necessidade ou não da realização de uma adaptação. A fase de planejamento é responsável pela definição de quais atividades de adaptação devem ser realizadas, com o objetivo que o sistema retorne a

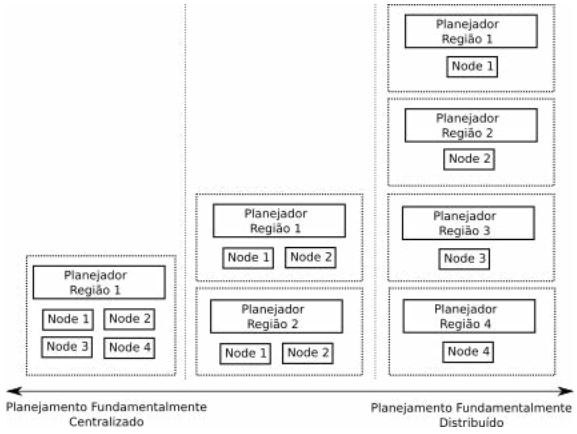

Figura 2. Diferentes instâncias do padrão *Regional Planning*.

um estado particular de interesse. Por fim, a atividade de execução realiza a implantação, no sistema gerenciado, das ações de adaptação definidas na fase de planejamento.

Em ambientes altamente distribuídos, heterogêneos e complexos, um único *loop* de controle pode não ser suficiente para gerenciar toda a sua estrutura [Weyns 2010]. Nestes cenários, portanto, múltiplos *loops* devem ser empregados para controlar as diferentes partes (múltiplos sistemas gerenciados) que compõem o sistema. Esta associação de *loops* de controle aos diversos nós do sistema distribuído traz uma complexidade a mais durante o projeto da solução: a coordenação e interação entre estes múltiplos *loops*.

Em [Weyns 2010], um conjunto de seis padrões arquiteturais para implementação de autogerenciamento descentralizado é apresentado. Para cada padrão, são apresentados os atributos de qualidade decorrentes da sua adoção e as restrições impostas pelas decisões que compõem a solução. Um dos padrões apresentados é o *Regional Planning* – recomendado para situações onde diversas partes integradas (regiões) de um sistema demandam não apenas adaptações locais (dentro das regiões) mas também adaptações globais (que extrapolam os limites das regiões). A solução proposta pelo *Regional Planning* prevê um único componente `P` (*planning*) para cada região. Este planejador regional é responsável por coletar as informações de todos os componentes que fazem parte da sua região, para que o planejamento seja realizado. Planejadores regionais coordenam entre si (sobre múltiplas regiões) para a realização do planejamento da adaptação global.

A Figura 2 apresenta diferentes instâncias do padrão *Regional Planning*, com diferentes tamanhos de região. Na figura, dois tipos de nós de controle são apresentados. O primeiro deles apresenta apenas o componente `P` (planejador da região) e cada região é controlada por apenas um nó do tipo `P`. O segundo tipo apresenta os outros três componentes MAPE-K restantes: `MAE` (Monitoramento, Análise e Execução). Em cada região, existem múltiplos nós de controle do tipo `MAE`, que realizam o monitoramento do seu sistema local e do seu ambiente de execução. O componente `A` realiza uma análise local dos dados monitorados e reporta resultados para o planejador `P` da sua região. O planejador regional `P`, por sua vez, planeja a adaptação local da região após cooperar com os planejadores `P` das outras regiões. Com este arranjo, a adaptação realiza a busca por adaptações globais, extrapolando os limites do monitoramento e análise locais.

Como consequência da adoção do padrão *Regional Planning*, espera-se uma

redução na quantidade de dados trafegados, uma vez que o monitoramento e análise são realizadas localmente. A análise local dos dados monitorados reduz a quantidade de dados e a frequência com que interações serão realizadas com o nó central da região. Por outro lado, a necessidade de agregação local – no nó de planejamento – dos resultados das diversas operações de análise pode implicar em *overhead* excessivo. Outra desvantagem é a necessidade de uma fase de planejamento demasiadamente detalhada, uma vez que as fases de execução não coordenam. Finalmente, a adoção do padrão *Regional Planning* requer a definição – em tempo de projeto – do tamanho de região com o qual o sistema será configurado. Consequentemente, ambientes diferentes podem demandar diferentes tamanhos de região para que o autogerenciamento apresente desempenho satisfatório. Em ambientes altamente dinâmicos e incertos, autogerenciamento satisfatório só é obtido com a variação – em tempo de execução – do tamanho de região adotado.

3. O Mecanismo de Planejamento Regional Adaptativo

Este trabalho apresenta o projeto, implementação e avaliação de um modelo adaptativo para autogerenciamento em SSA de larga escala. O modelo permite a reconfiguração dinâmica da topologia do sistema gerenciador e da forma de comunicação entre os múltiplos componentes locais do controle. Com esta abordagem, decisões referentes à topologia de controle a ser adotada (tamanho da região de planejamento) são levadas para o tempo de execução. Com isso, espera-se obter desempenho satisfatório de autogerenciamento tanto em situações onde o sistema gerenciado/ambiente sugerem a adoção de topologias mais centralizadas, quanto em situações onde padrões completamente descentralizados são mais adequados. Em ambientes altamente dinâmicos e incertos, é comum que uma determinada topologia – adequada para um dado cenário – não seja mais satisfatória após a ocorrência de mudanças no ambiente.

Este trabalho viabiliza o projeto de sistemas gerenciadores de tal modo que eles mesmos sejam tratados como SSA. Dessa forma, decisões acerca dos limites das regiões em arquiteturas que seguem o padrão *Regional Planning* não mais são realizadas *off-line*. Com o modelo proposto, em função do seu estado interno e das condições do ambiente, a arquitetura do sistema gerenciador será reconfigurada para adotar desde abordagens mais centralizadas até os extremos providos pelos padrões mais distribuídos.

3.1. Motivação

Conforme discutido na Seção 2, a adoção do padrão *Regional Planning* apresenta um *trade-off* que envolve a qualidade da adaptação vs. a quantidade de dados trafegados (e localmente mantidos) nos nós responsáveis pelo planejamento. Este *trade-off* é comumente administrado pelo arquiteto de software, em tempo de projeto. Para tanto, os tamanhos das regiões são fixos e definidos antecipadamente, sobretudo considerando as situações de maior sobrecarga previstas para a operação do sistema.

Visto que no padrão *Regional Planning* o tamanho das regiões é definido em tempo de projeto, os sistemas gerenciadores tendem a operar de forma não-ótima quando as condições ambientais se desviam daquelas consideradas em tempo de projeto. Para situações de sobrecarga do sistema – onde um máximo de controle se faz necessário – os controles regionais atuam de maneira satisfatória. Por outro lado, em situações onde a utilização de tamanhos menores de região seria suficiente para garantir a qualidade de operação do sistema, um superprovisionamento de recursos estará sempre presente.

A possibilidade de utilização de um modelo onde a decisão acerca do tamanho da região não mais seja uma definição a ser administrada em tempo de projeto – mas sim uma decisão levada para o tempo de execução – é o que motiva a adoção do Planejamento Regional Adaptativo proposto neste trabalho. Ao utilizar um tamanho variável de região, o sistema gerenciador terá a possibilidade de operar de maneira ótima, independente dos diferentes estados apresentados pelo ambiente e sistema gerenciado ao longo do tempo.

3.2. O Mecanismo Proposto

O Planejamento Regional Adaptativo aqui proposto implementa uma variação do padrão *Regional Planning*. Diferente do *Regional Planning* convencional, o Planejamento Regional Adaptativo transfere para o tempo de execução a decisão acerca do tamanho das regiões. Dessa forma, o sistema gerenciador passa a ser, ele mesmo, um SSA que rejeita perturbações no ambiente através da variação no tamanho das regiões de planejamento.

Para isso, um segundo nível de controle (metacontrole) é inserido na arquitetura. Os tamanhos de região que antes eram fixos – definidos *off-line* de maneira a suportar os cenários de pior caso – passam a ser agora mantidos em tempo de execução pelo metacontrole. Com isso, espera-se obter desempenho satisfatório de autogerenciamento tanto em situações onde o sistema gerenciado/ambiente sugerem a adoção de regiões maiores (planejamento global), quanto em situações onde o uso de pequenas regiões (planejamento local) já é suficiente. Trata-se, portanto, de adicionar ao padrão *Regional Planning* um segundo nível de adaptação.

Em momentos onde o sistema gerenciado e o ambiente demandem menos da adaptação, o tamanho das regiões será diminuído pelo metacontrole. Dessa forma, recursos serão poupados. Por outro lado, em momentos onde o metacontrole verifica que o componente MAPE-K responsável pelo planejamento regional necessita de monitoramento e análises mais globais – de forma a garantir os requisitos de qualidade acordados – o tamanho das regiões é aumentado. A fundamentação para tal investigação envolve os impactos decorrentes da variação do tamanho das regiões. A hipótese aqui investigada considera que a adoção de regiões grandes traz como vantagem a maior presença de informações globais, o que resulta na possibilidade de realização de adaptações mais efetivas. Por outro lado, a adoção de regiões pequenas beneficia aspectos tais como disponibilidade e escalabilidade. Sendo assim, o objetivo geral da solução aqui apresentada é regular, de forma automática, o *trade-off* existente entre a efetividade das adaptações vs. a disponibilidade e escalabilidade da solução.

Algumas premissas foram consideradas nesta proposta. Primeiro, assume-se que o *overhead* introduzido pelo metacontrole é desprezível. Visto que as operações de reconfiguração geralmente envolvem a modificação de parâmetros, tais como tamanho de *buffers* ou *thread pools*, o custo do trânsito de tais parâmetros na rede é consideravelmente menor que transitar os dados da aplicação. Segundo, todas as regiões possuem sempre o mesmo tamanho. Embora tal premissa eventualmente não seja verificada em *clusters* com bastante heterogeneidade, suportar diferentes tamanhos de região traria uma complexidade maior ao modelo e será analisado em trabalhos futuros. Por fim, considera-se que o *cluster* possui *jobs* sendo demandados a todo momento, o que viabiliza o contínuo monitoramento do desempenho das aplicação e do autogerenciamento.

4. Avaliação

O mecanismo de Planejamento Regional Adaptativo aqui proposto foi modelado e avaliado via simulação de eventos discretos. Os resultados da simulação foram avaliados considerando dois grupos de controle: o primeiro, decorrente da execução da simulação utilizando o Planejamento Regional Adaptativo proposto neste trabalho; e o segundo, utilizando o padrão *Regional Planning* convencional, com o uso de um tamanho fixo de região de planejamento. As seguintes hipóteses foram investigadas no experimento:

- Hipótese alternativa (H_1): mantida a mesma qualidade do serviço, o tamanho médio da região (e, portanto, o consumo de recursos) ao utilizar o Planejamento Regional Adaptativo aqui proposto (T_{PRA}) é menor que o tamanho médio da região ao utilizar o *Regional Planning* convencional (T_{PR}).
- Hipótese nula (H_0): mantida a mesma qualidade do serviço, o tamanho médio da região (e, portanto consumo de recursos) ao utilizar o Planejamento Regional Adaptativo aqui proposto (T_{PRA}) não é menor que o tamanho médio da região ao utilizar o *Regional Planning* convencional (T_{PR}).

4.1. Simulação

O modelo da simulação define o sistema gerenciado, gerenciador e metacontrole como um único sistema distribuído. Este sistema distribuído é modelado como um grafo $E(N, C)$. O grafo E é composto por um conjunto finito de nós, denominado $N = \{N_1, N_2, \ldots, N_n\}$ e um conjunto finito de canais de comunicação $C = f(N_i, N_j)|N_i, N_j \in N$. Vértices representam nós (sistemas gerenciados e gerenciadores) e as arestas representam canais de comunicação. O grafo E implementa os detalhes referentes ao sistema simulado, tais como os componentes que simulam os nós do sistema, informações acerca das regiões e eventuais perturbações (P) presentes no ambiente.

Uma vez que os nós N têm o seu tempo de serviço afetado diretamente pela qualidade da adaptação, tem-se que o tempo de serviço das operações (TS) é função do tamanho das regiões (R). Além do tamanho das regiões, nota-se que o tempo de serviço das operações é diretamente influenciado pela perturbação (P) presente no ambiente. Dessa forma, tem-se que TS é função tanto do tamanho das regiões quanto da perturbação existente no ambiente: $TS(R, P)$. A perturbação (P) foi modelada através de uma variável aleatória, cuja intensidade é obtida utilizando a distribuição normal, ao passo que sua frequência de ocorrência é denotada por uma distribuição exponencial.

Além das relações definidas na função $TS(R, P)$, uma variável aleatória K foi inserida no cálculo do tempo de serviço das operações para representar componentes estocásticos internos ao nó. Os valores de K são obtidos de acordo com a distribuição *Erlang* [Jain 1991]. E armazena ainda a fila de operações a ser executada. O modelo proposto considera a existência de uma fila de operações constantemente alimentada e representa *jobs* demandados por usuários. Os nós presentes no sistema atuam consumindo as operações que aguardam por processamento. O tempo de chegada de novas operações na fila foi modelado através de uma variável aleatória com distribuição exponencial.

Dois níveis de adaptação são observáveis no modelo proposto. O primeiro nível implementa o padrão *Regional Planning* e é responsável por garantir a adaptação nos nós. Esse controle é implementado na função que calcula o tempo de serviço das operações

$TS(R, P)$. O tamanho das regiões (R) é a variável que representa a contribuição, do controle de primeiro nível, no tempo de serviço das operações. O segundo nível implementa a operação de metacontrole (MC), responsável por adaptar os controladores de primeiro nível, sobretudo atuando no tamanho das suas regiões. O MC atua no tamanho das regiões da seguinte forma: se a quantidade de ciclos *idle* realizados é maior que um determinado valor de referência, o tamanho das regiões é reduzido. Por outro lado, caso a quantidade de ciclos *idle* mantenha-se abaixo do valor de referência, o MC atuará aumentando o tamanho das regiões. Apesar de o modelo não depender diretamente da lei de controle, para que a simulação pudesse ser executada fez-se necessário optar por uma implementação concreta. Nesta simulação, optou-se por utilizar controladores PID (Proporcional-Integral-Derivativo) [Hellerstein et al. 2004] para realização da adaptação.

A simulação foi implementada utilizando a linguagem de programação *Python* e o *framework* para simulação *SimPy* (http://simpy.readthedocs.org). Pacotes adicionais foram utilizados para geração dos números aleatórios de acordo com as distribuições definidas no modelo. A análise dos resultados foi realizada com a ferramenta `R` (http://www.r-project.org), com o auxílio de módulos para execução dos testes estatísticos.

4.2. Análise dos Resultados

A análise dos dados foi realizada com base em um estudo comparativo com dois diferentes grupos de dados. O primeiro grupo representou a execução da simulação utilizando o padrão *Regional Planning* convencional. O segundo grupo, por sua vez, representou a execução da simulação com a utilização do Planejamento Regional Adaptativo proposto neste trabalho. O experimento apresentou um único fator: o tamanho das regiões em tempo de execução. Esse único fator apresenta dois diferentes níveis: Planejamento Regional Adaptativo (PRA) ou *Regional Planning* convencional (PR).

Para cada nível foram executadas cinquenta replicações utilizando as mesmas sementes de geração de valores para as variáveis aleatórias. A quantidade de ciclos *idle* foi medida para cada uma das replicações. Os resultados das cinquenta replicações foram inicialmente analisados com os testes de *Anderson-Darling* e *Levene*, com o objetivo de analisar a normalidade e homoscedasticidade dos resultados. Com base nos resultados destes testes, decidiu-se pela avaliação da hipótese nula usando ou testes paramétricos (*t-test*) ou testes não-paramétricos (*Wilcoxon*).

5. Resultados

Para realização dos testes considerando o nível PR (tamanho fixo das regiões) foi necessário definir um tamanho fixo para as regiões. Este tamanho foi o mínimo necessário para que o controle tivesse condições de adaptar o sistema satisfatoriamente na situação de pior caso. Uma vez que o tempo de serviço TS é função do tamanho da região e do nível de perturbação verificado em dado instante, calculou-se o tamanho mínimo de região necessário para suportar o ambiente nos momentos de maior perturbação possível. A perturbação gerada no ambiente é uma variável aleatória com distribuição normal. Logo, considerou-se como perturbação máxima o seu valor médio somado ao erro máximo (valor de referência $-$ valor observado). A segunda execução da simulação foi realizada considerando o nível PRA (tamanho variável das regiões). A configuração inicial da simulação inicia o tamanho da região com o mesmo valor utilizado no nível PR, porém, com o decorrer da simulação, o tamanho da região é administrado pelo metacontrole.

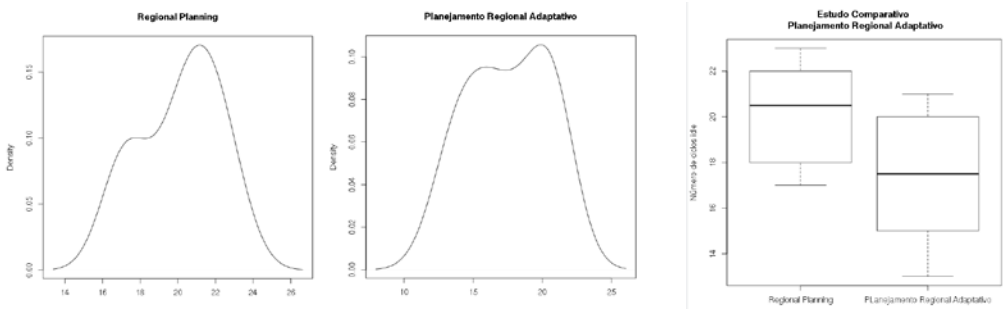

Figura 3. Distribuição dos dados resultantes da simulação.

Uma investigação acerca da normalidade e homoscedasticidade dos dados foi realizada, com o objetivo de decidir se a hipótese nula do experimento poderia ser analisada através de um teste paramétrico (*t-test*). O resultado da análise de normalidade, com um nível de significância $\alpha = 0.01$, apresentou um *p-value* igual a 0.85. A análise de homoscedasticidade apresentou um *p-value* igual a 0.05. Dos resultados anteriores, conclui-se que ambos os testes falharam e, portanto, testes não-paramétricos devem ser utilizados. A Figura 3 apresenta os resultados obtidos.

Considerando as características de normalidade e homoscedasticidade dos dados, concluiu-se que o *t-test* não poderia ser utilizado como ferramenta para investigação da hipótese nula. Optou-se então pela utilização do teste de *Wilcoxon* (não-paramétrico). O resultado do teste de *Wilcoxon* para as amostras anteriores apresentou, com um nível de significância $\alpha = 0.01$, um *p-value* igual a 0.85. O resultado obtido indica que a hipótese nula H_0 pode ser rejeitada e, portanto, que as amostras avaliadas apresentam melhora significativa dos resultados. Uma representação gráfica do estudo comparativo dos resultados é apresentada na Figura 3.

6. Trabalhos Relacionados

Diferentes padrões de projeto podem ser encontrados na literatura para tentar resolver o problema do planejamento [Abuseta and Swesi 2015, Weyns 2010]. Um destes padrões de projeto é o "*Information Sharing*", que restringe a comunicação entre os componentes presentes no padrão descentralizado. Nesta organização, apenas o componente de monitoramento `M` pode se comunicar com outros componentes externos. Esta abordagem resulta em soluções com melhor escalabilidade de comunicação do que o controle totalmente descentralizado, agilizando o processo de execução da adaptação. Porém, este padrão não é indicado para ambientes altamente dinâmicos pois, como as análises e planejamentos são feitos localmente, as adaptações acabam sendo sub-ótimas.

Outra possibilidade é o padrão de projeto denominado controle hierárquico. Esta abordagem define uma separação de *concerns* por camadas, onde cada nível executa seu próprio *loop* MAPE-K. Os níveis inferiores, com seus componentes de monitoramento e execução, lidam diretamente com o subsistema gerenciado. A camada mais alta se preocupa com a adaptação geral do sistema. O padrão de controle hierárquico permite gerenciar a complexidade do SSA. Porém, esta abordagem não é indicada para ambientes em constante mudança pois a divisão em camadas permite que existam dois planejamentos de adaptação, sendo eles global e específico.

7. Conclusão

Este trabalho apresentou o Planejamento Regional Adaptativo – uma extensão do padrão arquitetural *Regional Planning* que promove uma economia de recursos computacionais e menor necessidade de provisionamento de ambiente em SSA de larga escala. Diferente do padrão *Regional Planning*, o modelo proposto dotou, tanto o sistema gerenciador quanto o sistema gerenciado, de capacidades de autogerenciamento. Com isso, viabiliza-se a execução do sistema gerenciador utilizando tamanhos reduzidos de regiões, o que implica em uma economia de recursos. O mecanismo foi modelado e avaliado via simulação de eventos discretos. Dois grupos foram comparados: controle utilizando o padrão *Regional Planning* convencional e controle utilizando o Planejamento Regional Adaptativo proposto. A análise dos resultados indica que, com um nível de significância $\alpha = 0.01$, houve um aumento de desempenho do sistema gerenciador ao utilizar o Planejamento Regional Adaptativo.

Referências

[Abuseta and Swesi 2015] Abuseta, Y. and Swesi, K. (2015). Design patterns for self adaptive systems engineering. *CoRR*, abs/1508.01330.

[DARPA 1997] DARPA, B. A. A. (1997). Self-adaptive software. Technical Report 98-12, DARPA (Defense Advanced Research Projects Agency).

[Hellerstein et al. 2004] Hellerstein, J. L., Diao, Y., Parekh, S., and Tilbury, D. M. (2004). *Feedback Control of Computing Systems*. John Wiley & Sons.

[Huebscher and McCann 2008] Huebscher, M. C. and McCann, J. A. (2008). A survey of autonomic computing - degrees, models, and applications. *ACM Comput. Surv.*, 40(3).

[Jain 1991] Jain, R. (1991). The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling. *SIGMETRICS Performance Evaluation Review*, 19:5–11.

[Kephart and Chess 2003] Kephart, J. O. and Chess, D. M. (2003). The vision of autonomic computing. *IEEE Computer*, 36(1):41–50.

[Northrop 2013] Northrop, L. M. (2013). Does scale really matter? ultra-large-scale systems seven years after the study (keynote). In Notkin, D., Cheng, B. H. C., and Pohl, K., editors, *35th International Conference on Software Engineering, ICSE '13, San Francisco, CA, USA, May 18-26, 2013*, page 857. IEEE / ACM.

[Patikirikorala et al. 2012] Patikirikorala, T., Colman, A., Han, J., and Wang, L. (2012). A systematic survey on the design of self-adaptive software systems using control engineering approaches. In *Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 33 –42.

[Salehie and Tahvildari 2009] Salehie, M. and Tahvildari, L. (2009). Self-adaptive software: Landscape and research challenges. *TAAS*, 4(2).

[Weyns 2010] Weyns, D. (2010). On patterns for decentralized control in self-adaptive systems. In de Lemos, R., Giese, H., Müller, H. A., and Shaw, M., editors, *Software Engineering for Self-Adaptive Systems II*, volume 7475 of *Lecture Notes in Computer Science*, pages 76–107. Springer.