Uma Metodologia para Projeto de Protocolos Utilizando Algoritmos Genéticos e Modelos Markovianos

Marcio Nunes de Miranda[1,2*] Aloysio C. P. Pedroza[1,2]
marcio@gta.ufrj.br aloysio@gta.ufrj.br

Antônio C. de Mesquita Filho[2]
mesquita@coe.ufrj.br

[1] Grupo de Teleinformática e Automação (GTA)
Universidade Federal do Rio de Janeiro

[2] COPPE/EE - Programa de Engenharia Elétrica/Depto. de Eletrônica
Caixa Postal 68504 - CEP 21945-970 - Rio de Janeiro - RJ - Brasil
FAX: + 55 21 2906626

Resumo

Neste trabalho é apresentada uma metodologia para o projeto de protocolos baseada na utilização de algoritmos genéticos e nas técnicas de análise de desempenho que utilizam modelos Markovianos. Na modelagem inicial do protocolo são utilizadas as redes de Petri estocásticas, que podem ser mapeadas diretamente em Cadeias de Markov a tempo contínuo finitas. O modelo em rede de Petri é convertido em um novo modelo, utilizando-se a ferramenta de análise de desempenho Tangram-II. É utilizado um método de otimização baseado em algoritmos genéticos que fornece valores para que o Tangram-II forneça medidas que permitem calcular a função objetivo para cada conjunto aleatório de parâmetros fornecido pelo algoritmo genético. É realizado um processo iterativo até que a função objetivo seja menor que um certo erro especificado pelo projetista.

Abstract

This paper presents a methodology for the design of protocols, based on performance analisys techniques and genetic algorithms. The initial model for the protocol is built using stochastic Petri nets that can be directly mapped onto a Markov model, which is commonly used in performance analysis tools. The Petri net model is converted to the input model of the Tangram-II tool. The genetic algorithm is used to feed the Tangram-II with random values and to calculate the error function based on the results of Tangram-II. This is an iterative process that defines the parameters of the protocol model in order to match a given specification.

---

*Aluno de doutorado.

1 Introdução

Tradicionalmente, os protocolos têm sido implementados inteiramente em software. Entretanto, para que os requisitos de velocidade das redes atuais sejam atendidos, pode ser necessário implementar parte das especificações de um protocolo em hardware o que implica em promover a partição hardware/software de uma determinada especificação [1][2]. Assim, durante o projeto de um protocolo de comunicação o projetista deve definir, a partir de um modelo inicial, as tarefas do protocolo que serão implementadas em hardware e as que serão implementadas em software. Apesar dos tempos de execução em hardware serem, normalmente, muito menores que na implementação em software, o custo deste tipo de solução é maior.

Atualmente o projetista tem poucos recursos disponíveis para auxiliá-lo no refinamento da especificação e na partição hardware/software do protocolo [3]. Geralmente o projetista utiliza apenas a sua experiência e técnicas manuais e informais para definir a partição, o que limita a exploração do espaço de soluções. Neste caso, é útil fornecer ao projetista medidas de desempenho, já a partir do modelo inicial, de maneira a automatizar o processo, auxiliando-o na sua decisão.

Deve-se considerar diversos aspectos na partição hardware/software do projeto de um protocolo de comunicação [3]. Inicialmente deve-se definir a *granularidade* do sistema, ou seja, as menores partes, indivisíveis, usadas durante a partição, tais como: processos, subrotinas, "loops", blocos de instruções, instruções ou operações aritméticas. Quanto maior a granularidade, menor o número de soluções possíveis, o que agiliza o processo mas, por outro lado, diminui o número de partições possíveis. Também devem ser definidos os parâmetros a serem usados para a realização da partição. Alguns parâmetros comumente utilizados incluem custo financeiro, desempenho, taxas de comunicação, área do "chip", confiabilidade, tamanho do programa, quantidade de dados e tamanho da memória. Este trabalho concentra-se nos parâmetros de desempenho.

É necessário combinar os diversos parâmetros utilizados numa função de custo única, denominada *função objetivo*, que fornece uma medida da qualidade da partição. Como os parâmetros especificados normalmente "competem entre si", é útil atribuir-se pesos a cada um dos parâmetros que compõem a função objetivo para equalizar as possíveis diferenças de sensibilidade entre eles. A função objetivo fornece uma maneira de se comparar duas partições e selecionar aquela que satisfaz melhor as especificações.

Atualmente existem diversas abordagens que procuram solucionar o problema da partição hardware/software: algoritmos de "clustering", algoritmos de melhoria incremental (iterative-improvement), algoritmos customizados (custom algorithms) e algoritmos genéticos. Alguns são mais rápidos, como os de "clustering" enquanto que outros são mais lentos porém encontram soluções melhores, como os algoritmos genéticos, que serão explorados com maiores detalhes.

Neste trabalho, é proposto o uso de uma ferramenta de análise de desempenho para auxiliar o projetista na identificação das operações críticas em termos de tempo e outros parâmetros tais como, por exemplo, vazão. Com base nessas medidas de desempenho o projetista pode analisar a sensibilidade das especificações de projeto aos parâmetros do modelo e definir, de forma consistente, a partição hardware/software (HW/SW).

O método proposto utiliza inicialmente um modelo de protocolo baseado nas redes de Petri estocásticas [4][5]. Ao longo do processo, esta descrição é convertida em um novo modelo, apropriado à análise de desempenho, utilizando-se para isto uma ferramenta de análise de desempenho denominada Tangram-II [6][7]. Os valores dos parâmetros do modelo utilizado pela ferramenta Tangram-II serão fornecidos por um método de otimização baseado em algoritmos genéticos (AG's) [8][9]. Para cada *indivíduo* de uma população (ponto no espaço de soluções gerado pelo AG), o Tangram-II fornece resultados que possibilitam o cálculo de medidas de desempenho que indicam o grau de *aptidão* do indivíduo em relação a uma determinada função objetivo especificada pelo projetista. Desta forma, realiza-se uma seleção dos indivíduos, gera-se uma nova população (geração) e realimenta-se o Tangram-II. Este processo se repete até que a função objetivo avaliada esteja dentro de uma precisão especificada ou até que um certo número de gerações do algoritmo seja alcançado. Por exemplo, se a especificação for uma determinada curva de vazão para o protocolo, são fornecidos os parâmetros do modelo que melhor aproximam a curva.

Este trabalho está organizado da seguinte forma: na seção 2 é apresentada a metodologia de projeto proposta, na seção 3 são apresentados os principais conceitos que envolvem a utilização de algoritmos genéticos, na seção 4 é apresentada a ferramenta utilizada e alguns conceitos básicos de análise de desempenho, na seção 5 é realizada uma descrição da implementação da metodologia e são apresentados os resultados obtidos até o presente momento e na seção 6 as conclusões e trabalhos futuros.

2 A Metodologia de Projeto

Uma das técnicas mais utilizadas para a especificação e projeto de protocolos de comunicação é a modelagem em redes de Petri [10][11]. Na metodologia usada neste trabalho, os protocolos a serem projetados são modelados em redes de Petri estocásticas (SPN's), que é uma extensão do modelo de rede de Petri tradicional. Nas SPN's o tempo de disparo de uma transição, após a mesma ser habilitada, possui distribuição exponencial com taxa λ_i, onde λ_i é a taxa associada a cada transição do modelo. Em Molloy [12] é mostrado que as SPN's são equivalentes a uma cadeia de Markov a parâmetro contínuo finita [13], onde cada estado da cadeia corresponde a uma *marca* da rede de Petri [10] e as taxas de transição entre os estados correspondem às taxas com que cada transição da SPN é disparada. Por este motivo torna-se fácil a modelagem de SPN's na linguagem de entrada do Tangram-II, descrita na seção 4.1.2. O problema consiste em determinar as taxas de transição da SPN que modelam o protocolo, de forma a obedecer certas restrições de parâmetros de qualidade de serviço (**Qos**) impostas pelo projetista, ou seja, uma função objetivo especificada. O Tangram-II é utilizado como ferramenta para resolver a cadeia de Markov resultante do modelo especificado que, não por acaso, é igual à cadeia de Markov equivalente à rede de Petri estocástica do modelo inicial.

Neste trabalho, utiliza-se as probabilidades em estado estacionário de cada um dos estados da cadeia de Markov (calculadas pelo Tangram-II) para o cálculo da **vazão** de um protocolo. Este cálculo é realizado para cada conjunto de taxas (indivíduo) de uma população fornecida por um algoritmo genético. Calcula-se a *aptidão* do indivíduo utilizando-se uma função objetivo que é o valor absoluto da diferença entre a curva de vazão desejada e a curva de vazão obtida.

Nesta etapa inicia-se o processo de otimização, onde o valor da função objetivo é determinado para cada um dos indivíduos da população, ou seja, calcula-se a *aptidão* de cada indivíduo da população e verifica-se se algum destes indivíduos representa a solução

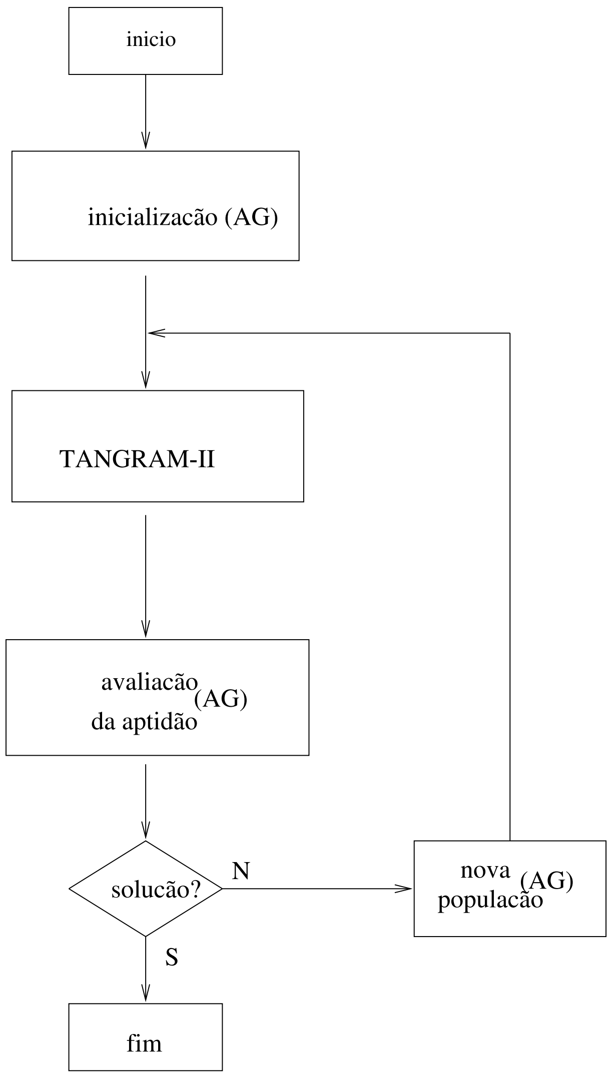

Figura 1: Metodologia para o projeto de protocolos

do problema. Caso a solução não tenha sido encontrada, realiza-se uma seleção dos indivíduos mais aptos e gera-se uma nova população. Realimenta-se o Tangram-II com a nova geração de indivíduos, repetindo-se o processo até que se encontre um conjunto de taxas que atenda à especificação do projetista. O algoritmo genético será descrito com maiores detalhes na seção 3. A figura 1 ilustra o processo.

Este processo permite que se identifique quais são as taxas do modelo que mais influenciam os parâmetros de Qos que o projetista considera mais importante. Neste caso, as tarefas do protocolo relacionadas com essas taxas devem ser implementadas em hardware, para que o protocolo atinja um alto desempenho. O projetista, então, pode se basear em dados concretos para realizar a sua partição hardware/software e não apenas em sua experiência.

3 Algoritmos Genéticos

Os algoritmos genéticos (AG's) vêm tendo larga aceitação devido à simplificação que eles permitem na formulação e solução de problemas de otimização. Esta característica é particularmente útil em problemas de otimização complexos, envolvendo um grande número de variáveis e, consequentemente, espaços de solução de dimensões elevadas. Além disso, em muitos casos onde outras estratégias falham na busca de uma solução, os algoritmos genéticos convergem. A estrutura básica do algoritmo genético é mostrada na figura 2.

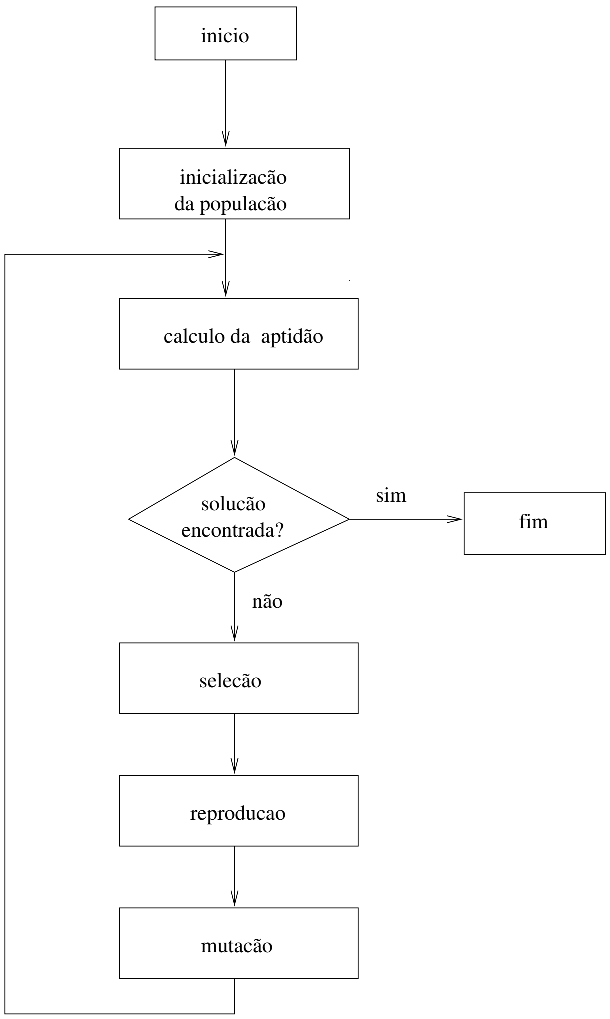

Figura 2: Um algoritmo genético simples

Algoritmos genéticos simples [9] normalmente trabalham com descrições de entrada formadas por cadeias de bits de tamanho fixo. Cada cadeia corresponde a um *cromossomo*, onde um ou mais parâmetros são codificados. Neste caso, cada parâmetro do modelo corresponde a um *gene*. Neste trabalho, o protocolo é modelado por uma rede de Petri estocástica e cada cromossomo será formado pelo conjunto de taxas de transição do modelo, onde cada valor é representado por uma variável real. Consequentemente, utilizou-se um vetor de variáveis reais de dimensão m para representar o cromossomo, onde m é o número de parâmetros do modelo. Deve ser notado que cada cromossomo, chamado de indivíduo no AG, corresponde a um ponto no espaço de soluções do problema de otimização. O processo de solução adotado nos algoritmos genéticos consiste em gerar, através de regras específicas, um grande número de indivíduos, *população*, de forma a promover uma varredura tão extensa quanto necessária do espaço de soluções. Neste trabalho, a população com n indivíduos é representada por uma matriz de números reais de dimensão n x m, onde cada linha corresponde a um cromossomo e n é o tamanho da população.

3.1 Operações Básicas do Algoritmo Genético

Com referência ao diagrama da figura 2, observa-se que cada iteração do algoritmo genético corresponde à aplicação de um conjunto de quatro operações básicas: cálculo de aptidão, seleção, cruzamento e mutação. Ao fim destas operações cria-se uma nova população, chamada de *geração* que, espera-se, representa uma melhor aproximação da solução do problema de otimização que a população anterior. A população inicial é gerada atribuindo-se aleatoriamente valores aos genes de cada cromossomo. A aptidão bruta de um indivíduo da população é medida por uma função de erro, também chamada de função objetivo do problema de otimização. A aptidão bruta é em seguida normalizada (aptidão normalizada), para permitir um melhor controle do processo de seleção. Como critérios de parada do algoritmo em geral são usados a aptidão do melhor indivíduo em conjunto com a limitação do número de gerações. Outros critérios podem envolver, por exemplo, um erro abaixo de um valor especificado pelo projetista para um determinado parâmetro do problema.

- Inicialização

  Uma população de n individuos é gerada aleatoriamente. Cada um dos indivíduos da população representa uma possível solução para o problema, ou seja, um ponto no espaço de soluções.

- Cálculo da aptidão

  Geralmente a aptidão do indivíduo é determinada através do cálculo da função objetivo, que depende das especificações de projeto. Neste trabalho, cada indivíduo é uma entrada para uma ferramenta de análise de desempenho, cuja saída fornece medidas que permitem ao algoritmo genético o cálculo da aptidão do indivíduo. Ainda nesta fase os indivíduos são ordenados conforme a sua aptidão.

- Seleção

  Nesta fase os indivíduos mais aptos da *geração* atual são selecionados. Esses indivíduos são utilizados para gerar uma nova população por cruzamento. Cada indivíduo tem uma probabilidade de ser selecionado proporcional à sua aptidão. Para visualizar este método considere um círculo dividido em n regiões (tamanho da população), onde a área de cada região é proporcional à aptidão do indivíduo (figura 3). Coloca-se sobre este círculo uma “roleta” com n cursores, igualmente espaçados. Após um giro da roleta a posição dos cursores indica os indivíduos selecionados. Este método é denominado “amostragem universal estocástica” [8]. Evidentemente, os indivíduos cujas regiões possuem maior área terão maior probabilidade de serem selecionados várias vezes. Como consequência, a seleção de indivíduos pode conter várias cópias de um mesmo indivíduo enquanto outros podem desaparecer.

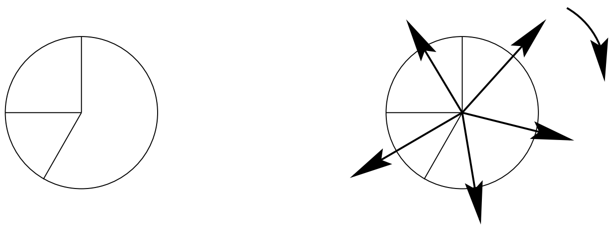

Figura 3: Amostragem universal estocástica

- Cruzamento ("Cross-Over")

  Os indivíduos selecionados na etapa anterior são *cruzados* da seguinte forma: a lista de indivíduos selecionados é embaralhada aleatoriamente criando-se, desta forma, uma segunda lista, chamada lista de parceiros. Cada indivíduo selecionado é então cruzado com o indivíduo que ocupa a mesma posição na lista de parceiros. A forma como se realiza este cruzamento é ilustrada na figura 4. Os cromossomos de cada par de indivíduos a serem cruzados são particionados em um ponto, chamado ponto de corte, sorteado aleatoriamente. Um novo cromossomo é gerado permutando-se a metade inicial de um cromossomo com a metade final do outro. Deve-se notar que se o cromossomo for representado por uma cadeia de bits, como na figura 4, o ponto de corte pode incidir em qualquer posição (bit) no interior de um gene, não importando os limites do gene. No caso de genes representados por números reais, a menor unidade do cromossomo que pode ser permutada é o gene.

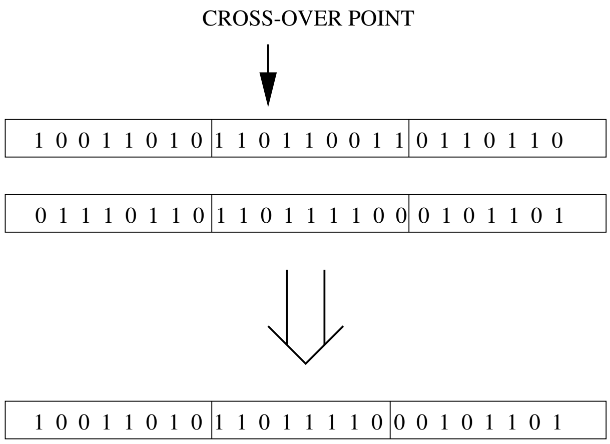

Figura 4: Cruzamento de dois indivíduos num algoritmo genético

- Mutação

  A operação de *mutação* é utilizada para garantir uma maior varredura do espaço de estados e evitar que o algoritmo genético convirja muito cedo para mínimos locais.

A mutação é efetuada alterando-se o valor de um gene de um indivíduo sorteado aleatoriamente com uma determinada probabilidade, denominada probabilidade de mutação, ou seja, vários indivíduos da nova população podem ter um de seus genes alterado aleatoriamente.

3.2 Parâmetros do Algoritmo Genético

Além da forma como o cromossomo é codificado, existem vários parâmetros do algoritmo genético que podem ser escolhidos para melhorar o seu desempenho, adaptando-o às características particulares de determinadas classes de problemas. Entre eles os mais importantes são: o **tamanho da população**, o **número de gerações**, a **probabilidade de cross-over** e a **probabilidade de mutação**. A influência de cada parâmetro no desempenho do algoritmo depende da classe de problemas que se está tratando. Assim, a determinação de um conjunto de valores otimizado para estes parâmetros dependerá da realização de um grande número de experimentos e testes. Na maioria da literatura [9] os valores encontrados estão na faixa de 60 a 65% para a probabilidade de cross-over e entre 0,1 e 5% para a probabilidade de mutação. O tamanho da população e o número de gerações dependem da complexidade do problema de otimização e devem ser determinados experimentalmente. No entanto, deve ser observado que o tamanho da população e o número de gerações definem diretamente o tamanho do espaço de busca a ser coberto.

4 Avaliação da Função Objetivo

Para o cálculo da aptidão de cada indivíduo é necessário avaliar uma função objetivo especificada pelo projetista. Neste trabalho, são utilizadas medidas de desempenho (Qos) do protocolo para a formulação da função objetivo. Estas medidas de desempenho serão calculadas a partir de dados fornecidos por uma ferramenta de análise de desempenho que será discutida a seguir.

Como foram usadas redes de Petri estocásticas para a modelagem inicial do protocolo, para simplificar o cálculo das medidas de desempenho, optou-se por um modelo de cadeias de Markov pela facilidade de mapeamento entre estes modelos [12]. Entre as inúmeras ferramentas de análise de desempenho que permitem o tratamento de modelos Markovianos [14][15], optou-se pela utilização do Tangram-II.

O Tangram-II fornece medidas que permitem o cálculo da aptidão para cada indivíduo da população gerada pelo algoritmo genético. O modelo do protocolo, inicialmente construído em rede de Petri, é transformado de forma simples para a linguagem de especificação de modelos do Tangram-II. As principais características desta ferramenta serão examinadas a seguir.

4.1 A Ferramenta Tangram-II

4.1.1 Características Gerais

O Tangram-II é uma ferramenta utilizada para especificação e análise de desempenho de sistemas, principalmente sistemas de comunicação e computacionais. Foi desenvolvido com objetivos educacionais e de pesquisa, pela equipe do laboratório **Land** da **Coppe/Ufrj**. Entre suas características principais estão: possuir uma interface gráfica bastante flexível e ser baseado no paradigma de orientação a objetos. O Tangram-II possui diversos métodos de solução da cadeia de Markov gerada a partir da especificação do modelo, tanto para estado estacionário quanto para estado transiente. Também possui um simulador interativo que possibilita a simulação de eventos raros e animação.

O usuário inicia a construção do modelo instanciando objetos que já existem numa biblioteca ou construindo novos objetos. Os novos tipos de objetos são especificados em termos de uma representação gráfica e seu comportamento associado. A interface gráfica (GUI) mostra um "template" que auxilia o usuário na construção de cada objeto do modelo. Após a construção do modelo, gera-se a cadeia de Markov associada. O usuário, então, seleciona o método de solução a ser adotado para se obter a solução da cadeia de Markov e, consequentemente, as medidas de interesse, ou seja, as probabilidades em estado estacionário.

4.1.2 Construção de Modelos

Os **eventos** no Tangram-II são gerados espontâneamente por um objeto, caso as **condições** associadas àquele evento estejam satisfeitas. Condições são expressões booleanas avaliadas a partir do estado corrente do objeto e usadas para indicar quando um evento está habilitado ou não. Uma condição deve vir entre parênteses e pode ser composta por variáveis e constantes declaradas, valores numéricos e operadores aritméticos e lógicos [7].

Ações são usadas para modificar o estado do objeto e para enviar mensagens a outros objetos. Uma ação é executada quando um evento é disparado ou uma mensagem é recebida. Ações são descritas em código C-like e podem usar variáveis e constantes declaradas no objeto. Uma ação também pode conter variáveis locais, declaradas dentro do escopo da ação. As variáveis de estado não podem ter o seu valor alterado durante uma ação. Somente ao final da ação é usada a função set_st(<var_name>, <value>) para alterar o valor da variável de estado. Várias ações podem ser associadas a um evento ou a uma mensagem, cada uma com uma probabilidade diferente de ocorrência (soma = 1). As probabilidades são especificadas usando-se a palavra-chave PROB seguida de uma expressão aritmética.

Exemplo de múltiplas ações:

```
action= {
/* chegou um pacote correto */
int q;
q = q + 1;
set_st("queue",q);
} : prob = p;
/* chegou um pacote incorreto , não faz nada */
{
;
} : prob = 1 - p;
```

Mensagens são usadas para representar a interação entre os objetos e são enviadas num tempo igual a zero.

Quando um evento ocorre ou uma mensagem é recebida, o conjunto de ações especificadas pelo usuário é executado com uma dada distribuição de probabilidade. Como resultado da execução de uma ação, o estado do objeto pode mudar e mensagens podem ser enviadas para outros objetos do modelo.

É importante salientar que a metodologia utilizada neste trabalho é baseada em modelos Markovianos (redes de Petri estocásticas). Na figura 5 é apresentado um exemplo da sintaxe usada no Tangram-II para a construção de um modelo de fila M/M/1/k.

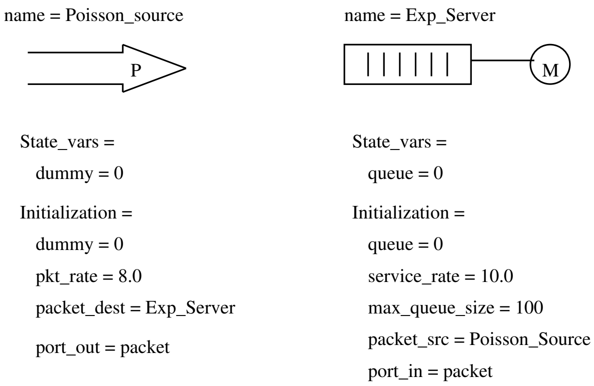

Figura 5: Modelo de fila M/M/1/k

Neste trabalho, o modelo construído no Tangram-II é um modelo literal, ou seja, todos os seus parâmetros são literais. No caso, estes parâmetros são as taxas de transição da cadeia de Markov associada ao modelo, que por sua vez são as mesmas taxas com que cada transição da rede de Petri estocástica é disparada. A cadeia de Markov associada ao modelo é gerada uma única vez, com parâmetros literais nas suas transições. No entanto, como os métodos de solução do Tangram-II são numéricos, é necessário que valores numéricos sejam passados ao Tangram-II antes que ele resolva a cadeia de Markov. Estes valores numéricos são fornecidos pelo algoritmo genético. Os valores fornecidos pelo Tangram-II para as probabilidades em estado estacionário de cada estado da cadeia de Markov são usados para o cálculo da aptidão do indivíduo.

5 Resultados Obtidos

5.1 O Experimento

Para a avaliação da eficiência da metodologia, o protocolo *bit alternante* utilizado em Molloy [12] foi modelado em uma rede de Petri, como é ilustrado na figura 6 e, posteriormente, convertido para o modelo do Tangram-II. Para testar a correção deste modelo, foram usados para as taxas os valores numéricos mostrados na tabela 1 e, a partir da solução da cadeia de Markov gerada, foram determinadas as probabilidades em estado estacionário de cada estado da cadeia. Isto permitiu o cálculo da vazão do protocolo para diversos valores de taxa de chegada de novas mensagens (carga), obtendo-se exatamente a curva da figura 7 (vazão versus carga), que é a mesma curva obtida em Molloy [12]. A seguir, implementou-se o algoritmo genético. Nesta etapa os parâmetros são literais, bem como o modelo equivalente do Tangram-II. A figura 7 mostra a curva a ser aproximada pelo algoritmo, cujo resultado final é um conjunto de valores para as taxas da rede de Petri literal, que devem ser aproximadamente iguais às taxas da tabela 1.

Tabela 1: Taxas da Rede de Petri

Transição	Taxa (disparos/seg.)
Send, Send Ack	9,375
Msg Drop, Ack Drop	3,91
Crc ok, Ack ok	74,22
Timeout	1,00

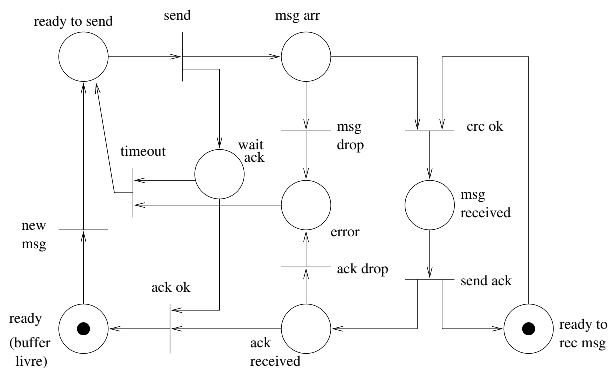

Figura 6: Rede de Petri do protocolo bit alternante

5.2 A Implementação do Algoritmo

O programa foi implementado na linguagem C e executado numa estação de trabalho SUN ULTRA1, no sistema operacional **Solaris**. Para implementar a metodologia apresentada, foi utilizado um algoritmo genético simples dentro do qual foram realizadas as

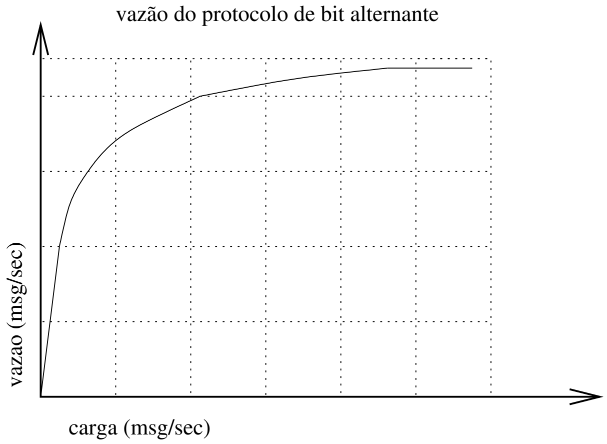

Figura 7: Curva vazão versus carga do protocolo bit alternante

alterações necessárias de forma a adaptá-lo para o problema em questão, ou seja, a codificação do cromossomo como um vetor de de reais que representam os parâmetros (taxas) a serem determinados. Observando a figura 6 pode-se perceber que existem 7 taxas a serem determinadas (o oitavo valor sendo a carga do sistema). A população inicial no algoritmo genético é obtida por uma função que gera números reais aleatoriamente, tomando como semente o relógio do computador. Os valores numéricos correspondentes a cada indivíduo são transferidos ao modelo literal do Tangram-II. Desta forma é possível executar o método de solução do Tangram-II e, consequentemente, calcular as probabilidades em estado estacionário para a cadeia de Markov associada ao indivíduo. A partir dessas probabilidades, determina-se a vazão para cada valor de carga (ponto no eixo x), multiplicando-se a probabilidade do sistema estar no *lugar* [10] "buffer livre" (cf. figura 6) pela carga do sistema (número de novas mensagens). Este processo é realizado k vezes, onde k é o número de pontos através dos quais se deseja aproximar a curva. O valor da função objetivo será dado pela soma dos erros entre os valores de vazão obtidos e os valores de vazão desejados. Desta forma, calcula-se a aptidão de um determinado indivíduo, realiza-se o processo de seleção e uma nova população é gerada, caso nenhum dos indivíduos atinja a precisão determinada no algoritmo.

5.3 Resultados

Como a curva da figura 7 é "bem comportada", acredita-se que 10 pontos sejam suficientes para aproximá-la e obter os parâmetros desejados. Até o presente momento, foram realizados experimentos com uma população de 5000 indivíduos, utilizando-se como precisão o valor de 0.001 e 12 gerações. O algoritmo convergiu para os valores de vazão desejados (erros abaixo de 0.001) para um e para dois pontos. A estrutura do algoritmo tem uma complexidade polinomial (n^3), proporcional ao número de pontos que se deseja aproximar, tamanho da população e número de gerações. Observou-se que a convergência do algoritmo fica mais difícil à medida que se aumenta o número de pontos a aproximar. No presente momento os aspectos relativos à convergência e eficiência (tempo de execução) estão sendo estudados.

6 Conclusões

Neste trabalho foi apresentado um estudo sobre a utilização de algoritmos genéticos e ferramentas de análise de desempenho no auxílio ao projeto de protocolos de comunicação. Apesar da metodologia apresentada ter sido direcionada para aplicação em projeto de protocolos, ela é totalmente genérica e pode ser aplicada a qualquer sistema que seja modelado de forma simples numa ferramenta de especificação e análise de desempenho.

Através dos experimentos descritos na seção 5 foi possível comprovar que a metodologia é possível de ser implementada e possui um grande potencial ainda a ser explorado, principalmente em relação à abrangência das aplicações, apesar de ainda existirem pontos importantes a serem resolvidos em relação à implementação.

Não foi encontrada na literatura nenhuma outra metodologia de auxílio ao projeto de protocolos baseada nos conceitos e ferramentas utilizados neste trabalho, que permitissem comparação.

Espera-se que o tempo de execução seja reduzido substancialmente a partir da implementação em ambiente **Linux** e em máquinas mais poderosas, pois este fator é de fundamental importância para o prosseguimento deste trabalho.

Neste trabalho procurou-se avaliar, fundamentalmente, o potencial da metodologia e a sua praticidade. Em trabalhos futuros, quando a implementação da metodologia estiver sedimentada, será dada maior ênfase aos parâmetros de **Qos**, onde será especificada uma função objetivo mais complexa, que represente de forma mais exata todos os parâmetros utilizados pelo projetista. Estes parâmetros, geralmente no nível de circuitos, deverão ser mapeados em parâmetros de mais alto nível, ou seja, aqueles normalmente utilizados por técnicas de análise de desempenho de redes, tais como: vazão, retardo, probabilidade de perda de pacotes, tamanho de "buffers", entre outros.

Agradecimentos

Agradecemos ao Prof. Edmundo Souza e Silva, coordenador do laboratório LAND da COPPE/UFRJ, e à toda sua equipe por ter fornecido todo o apoio em termos de recursos coputacionais, suporte técnico e constante troca de idéias relativas à utilização da ferramenta Tangram-II, particularmente ao aluno Sidney Cunha de Lucena, ao administrador da rede do LAND, Flavio Pimentel Duarte e a Daniel Rattom Figueiredo.

Este trabalho foi realizado com recursos da UFRJ, FUJB, CNPq, CAPES, COFECUB, FAPERJ e REENGE.

Referências

[1] L. Pirmez, *Uma metodologia para síntese de alto nível de protocolos a partir de uma descrição formal.* PhD thesis, Programa de Engenharia Elétrica - COPPE/UFRJ, 1996.

[2] J. Madsen, J. Grode, P. Knudsen, M. Petersen e A. Haxthausen, “Lycos: the lyngby co-synthesis system”, *Design Automation for Embedded Systems*, vol. 2, no. 2, pp. 1–43, fevereiro de 1997.

[3] D. D. Gajski e F. Vahid, “Specification and design of embedded software/hardware systems”, *IEEE Design and Tests of Computer*, vol. 12, no. 1, pp. 53–67, janeiro de 1995.

[4] H. Choi, V. Kulkarni e K. Trivedi, “Markov regenerative stochastic petri nets”, *Performance Evaluation*, vol. 20, no. 1, pp. 337–357, maio de 1994.

[5] R. German e C. Lindemann, “Analysis of stochastic petri nets by the method of supplementary variables”, *Performance Evaluation*, vol. 20, no. 1, pp. 317–335, maio de 1994.

[6] R. Carmo, L. Carvalho, E. Sousa e Silva, M. Diniz e R. Muntz, “Performance/availability modeling with the Tangram-II modeling environment”, *Performance Evaluation*, vol. 33, no. 1, pp. 45–65, junho de 1998.

[7] D. R. Figueiredo, “Um módulo de simulação da ferramenta Tangram-II: suporte para medidas com recompensa, recursos de eventos raros e aplicações a modelos de rede multimídia”, Tese de Mestrado, Programa de Engenharia de Sistemas - COPPE/UFRJ, julho de 1999.

[8] M. Mitchell, *An Introduction to Genetic Algorithms.* MIT Press, 1996.

[9] H. Horner, “A C++ class library for genetic programming: The Vienna University of Economics - genetic programming kernel”, *Internet Draft*, maio de 1996. `http://www.wu-wien.ac.at/usr/h88/h8850092`.

[10] J. Peterson, “Petri nets*”, *Computing Surveys*, vol. 9, no. 3, pp. 223–252, setembro de 1977.

[11] M. Diaz, “Modeling and analysis of communication and cooperation protocols using petri net based models”, *Computer Networks*, vol. 10, no. 6, pp. 419–441, junho de 1982.

[12] M. Molloy, “Performance analisys using stochastic petri nets”, *IEEE Transactions on Computers*, vol. 31, no. 9, pp. 913–917, setembro de 1982.

[13] S. M. Ross, *Stochastic Processes.* John Wiley & Sons, 1983.

[14] W. J. Stewart, *MARCA: Markov Chain Analyzer, a software package for Markov Chains. in: Numerical Solution of Markov Chains.* Marcel Dekker, 1991.

[15] G. Ciardo, J. Muppala e K. Trivedi, “SPNP: stochastic petri net package”, in *Workshop on Petri Nets and Performance Models*, 1989.