TRABALHO FINAL DE GRADUAÇÃO – JUNHO/2018
UNIVERSIDADE FEDERAL DE ITAJUBÁ
ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Otimização de Desempenho de Sistemas através de Rede de Petri Estocástica e Algoritmos Genéticos

Fernando Oliveira Pereira

Orientador: Prof. Carlos Henrique Valério de Moraes
Coorientador: Prof.: Luiz Edival de Souza
Instituto de Engenharia de Sistemas e Tecnologia da Informação (IESTI)

***Resumo* – Redes de Petri são utilizadas para modelagem de sistemas à eventos discretos utilizando grafos direcionados compostos por transições e estados. A partir do modelo podem ser obtidos vários resultados, neste trabalho o interesse é analisar o desempenho do sistema modelado. Para isso, o modelo precisará considerar eventos aleatórios e utilizar técnicas de processos estocásticos, em particular, Cadeias de Markov. Neste caso, o modelo deve capturar as probabilidades de mudanças dos estados do sistema associando um tempo médio de disparo das transições entre estados e gerar o conjunto de possíveis estados do sistema chamado de Arvore de Alcançabilidade. As combinações dos tempos médios associados as transições tem um impacto muito forte no desempenho do sistema. Por essa perspectiva, o uso de algoritmos genéticos para otimização desta combinação de tempo médio das transições é conveniente devido a capacidade de trabalhar com problemas NP-completos. O objetivo proposto foi atingido, demonstrando a eficácia do método.**

Palavras-Chave: Redes de Petri Estocástica, Cadeias de Markov, análise de desempenho de sistemas, método de seleção, tempo computacional.

I. INTRODUÇÃO

Redes de Petri (RP) são ferramentas matemáticas utilizadas para modelagem de sistemas à eventos discretos. Podem ser utilizadas para descrever processos de uma linha de produção [1], desempenho de serviços Web [2] ou o controle supervisório [3]. Nesse contexto, um evento ocasionará uma transição, como o acionar de um botão, evento determinístico, ou uma falha, evento estocástico. Dado um evento, a rede irá passar de um estado para outro, modificando as características anteriores [4].

Quando a RP é temporizada e estocástica, é relevante calcular a probabilidade de ocorrência do estado. Para isso, usa-se a Teoria de Markov, em que se pode calcular todas as probabilidades dos locais, baseado nos valores de transição [5].

A otimização da rede não é trivial, em uma rede simples é possível a elaboração manual, mas em redes complexas com 20 ou mais transições torna-se inviável. Uma alternativa para se otimizar é através de Algoritmos Genéticos (AG), os quais são algoritmos de busca inspirados na teoria da seleção natural, utilizados para problemáticas de busca e otimização [8]. Os AG são baseados na ideia de que indivíduos mais aptos possuem maiores chances de sobrevivência e de gerarem indivíduos mais adaptados e assim sucessivamente [6]. Análogo a isso, uma RP consiste em achar a melhor combinação de valores para as transições, a fim de deixar um estado com a probabilidade de ocorrência máxima ou mínima, dependendo do problema. Em que, quanto maior a rede, maior a complexidade do indivíduo e da resolução, ou seja, é um problema NP-Completo [7]. Outro fator que dificulta seu ajuste por métodos clássicos de otimização é sua função objetivo implícita de difícil predição, reforçando o uso de um AG para este problema. Tendo em vista a escassez de estudos e aplicações de RP com o objetivo de otimizar um processo, propõe-se uma solução via AG para encontrar uma alternativa viável de otimização de RP estocásticas.

II. DESENVOLVIMENTO

II.1 Redes de Petri Temporizadas e Estocásticas

Uma RP é um grafo direcionado, bipartido, composto por pelo menos dois elementos principais: Lugar, representação dos estados, e Transição, representação dos eventos. Caso o estado esteja ativo ele receberá uma marcação (*Tolkien*), indicando que está ativo.

A RP é formada pela quíntupla (1):

$$R = \{P, T, I, O, M_0\} \quad (1)$$

Em que:
P = {p1, p2, ...pj} é um conjunto finito de lugares;
T = {t1, t2, ... tq} é um conjunto finito de transições;
I = T → P é um conjunto representando o mapeamento de transições para lugares de entrada.
O = T → P é um conjunto que representa o mapeamento de transições para lugares de saída.

M_0: P → {0, 1, 2, ...} é a marcação inicial.

Baseado nos valores dos conjuntos I e O da RP, constrói-se a matriz de incidência Q (2):

$$Q = \begin{bmatrix} V_{11} & V_{21} & V_{31} & \cdots & V_{n1} \\ V_{12} & V_{22} & V_{32} & \cdots & V_{n2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ V_{1m} & V_{2m} & V_{3m} & \cdots & V_{nm} \end{bmatrix} \quad (2)$$

Em que: Os valores sobrescritos indicam primeiramente o lugar de origem e o outro o lugar de destino. V é o valor das taxas de disparos associadas às transições. Por exemplo, V_{21} é uma a transição que tem origem no lugar 2 (p2) e tem como destino o lugar 1 (p1) com um valor de taxa determinado para essa transição.
Para saber a probabilidade de cada estado, usa-se a Teoria de Markov. Na qual precisa dos dados apenas da matriz Q. A teoria diz que a somatória das probabilidades deve ser igual a 1 (3) e a somatória de cada linha da matriz deve ser igual a 0 (4) [5].
A diagonal principal é a soma dos valores da linha, contudo com o sinal multiplicado por (-1). Isto é feito para obedecer a regra que diz que a soma dos valores da linha deve ser igual a zero.

$$\sum_k \pi_k = 1 \quad (3)$$

$$0 = \pi_k * Q \quad (4)$$

Em que : $\pi_k = [\pi_1, \pi_2, \pi_3, \ldots, \pi_k]$ é o de vetor probabilidade de estados.
Com base nisso, gera-se um sistema linear para encontrar os valores de π_k, que permite analisar se o valor da taxa de disparo retorna os valores desejados a cada estado. Caso contrário, pode-se alterar esses valores até conseguir um resultado desejado.
Entretanto, como existem inúmeras possibilidades, o AG é uma alternativa para encontrar os valores ótimos.

II.1 Algoritmo Genético

O código foi feito C#, em que se utilizou AG para encontrar os valores ótimos (Figura 1).
Para a criação da população inicial é preciso entrar com os valores do esqueleto da matriz Q em binário, ou seja, deve-se indicar onde ocorrem as transições de origem para o destino, o valor é 1 para ocorrência de transição e 0 caso não exista. A diagonal principal deve ser iniciada com 1, o algoritmo fará a soma da linha e multiplicará por (-1) para obedecer às regras de Markov.
Em seguida é necessário entrar com os pesos dos estados, estes indicam o grau de importância que é dado aos estados (Quadro 1).

Quadro 1 – Critério de escolha do valor dos pesos

Valor do Peso	Significado
Peso ≥ +1	Onde se deseja uma taxa alta de ocorrência
1 < Peso < 0	É desejável que tenha uma taxa alta, mas não necessário
Peso = 0	O valor não tem relevância
0 < peso < -1	É desejável que tenha uma taxa baixa, mas não necessário
Peso ≤ -1	Onde se deseja uma taxa baixa de ocorrência

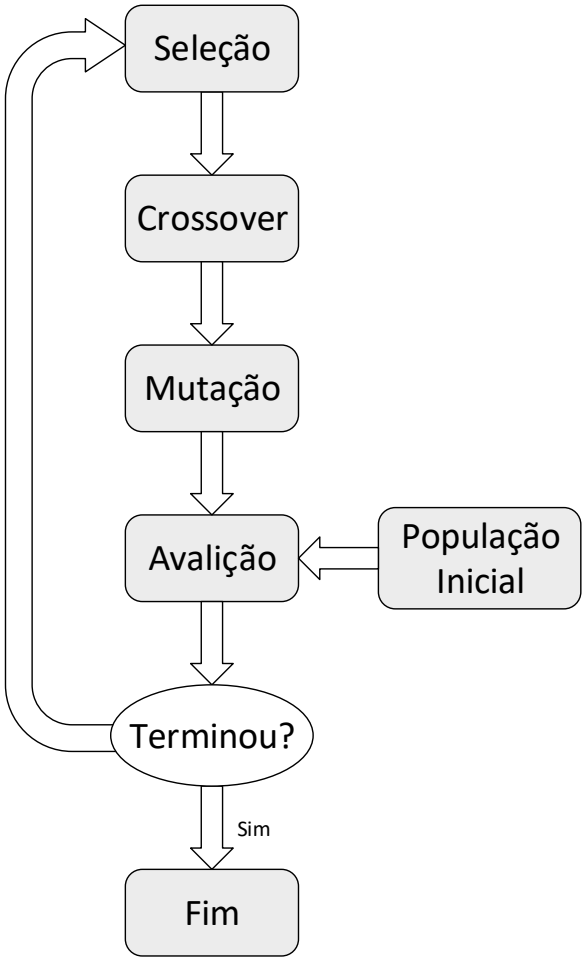

Figura 1 – Funcionamento do AG proposto.

A criação dos valores que compõe a população inicial foi feita aleatoriamente, para isso se utilizou uma função para gerar os valores iniciais com o único critério de serem positivos e diferentes de zero.

O cromossomo, portanto, será formado pelo esqueleto da matriz Q (2), multiplicados pelos valores gerados randomicamente. A cada nova geração somente os valores da população serão alterados, os da matriz serão mantidos.
Os métodos de seleção são a forma como os indivíduos farão a reprodução posteriormente, neste estudo em questão foram utilizados os métodos de seleção Elitista, Ranking e Roleta, que são descritos como:

-Elitista: Primeiro clona os melhores cromossomos para a nova população, depois reproduz os demais.
-Ranking: Classifica a população e então atribui a cada cromossomo um valor de adequação determinado pela sua classificação.
-Roleta: Os pais são selecionados de acordo com sua adequação. Quanto melhores são os cromossomos, mais chances de serem selecionados.

A avalição é feita utilizando o cálculo das probabilidades da Teoria de Markov. Os pesos são usados para o cálculo da função fitness (5). A probabilidade de cada estado será multiplicada pelo valor do peso do estado, após isso, a soma dos valores. Quanto maior o valor, maior a adaptabilidade do indivíduo.

$$\sum_{i=1}^{n} (peso_n * \pi_n) \quad (5)$$

Em que n é a quantidade de lugares.
O critério de parada escolhido foi de 1000 iterações ou até o valor da Taxa de disparo de 99% ser atingido no estado com o maior peso.
Caso um dos critérios de parada tenha sido atingido, o programa entrega os valores das transições e das probabilidades, caso contrário ocorre a reprodução (crossover) e a mutação.
O método de seleção foi previamente escolhido. O crossover é baseado na nota de cada indivíduo dada na equação 4 quanto maior o valor, melhor a nota do indivíduo e vice-versa. Existem 1000 indivíduos por interação. Feito o crossover pode ocorrer a mutação. A taxa de mutação utilizado foi de 10%, isto é, a probabilidade de o valor do indivíduo ser alterado de forma aleatória, fazendo isso previne a estagnação da posição em um valor, evitando problemas de mínimo local.

III. RESULTADOS

Foram testados dois sistemas, a primeira, uma RP simples, chamado "Sistema Teste 1" (S1), com 4 lugares e 6 transições, a segunda RP, chamada "Sistema Teste 2" (S2), mais complexa com 12 lugares e 40 transições.
O valor máximo de iterações foi de 1.000 gerações. A taxa de mutação foi de 0,1 e a taxa de cruzamento de 0,9. O critério de parada não foi utilizado, pois se queria observar o comportamento ao longo das gerações.
Os testes ocorreram da seguinte maneira, rodar o código 50 vezes para cada método de seleção, a fim de avaliar se a escolha aleatória da primeira geração afetava o desempenho do algoritmo. Os valores das iterações são as médias das 50 rodadas.

III.1 Sistema Teste 1 (S1)
O S1 foi modelado na RP mostrada na Figura 2.

Os lugares representam:
P1 máquina pronta para operar;
P2 máquina operando;
P3 máquina em manutenção;
P4 máquina em retrabalho de partes.

As transições representam:
T1: máquina inicia o processamento;
T2: máquina termina o processamento;
T3: máquina falha;
T4: máquina retrabalha após manutenção;
T5: máquina finaliza reprocessamento;
T6: máquina é reiniciada.

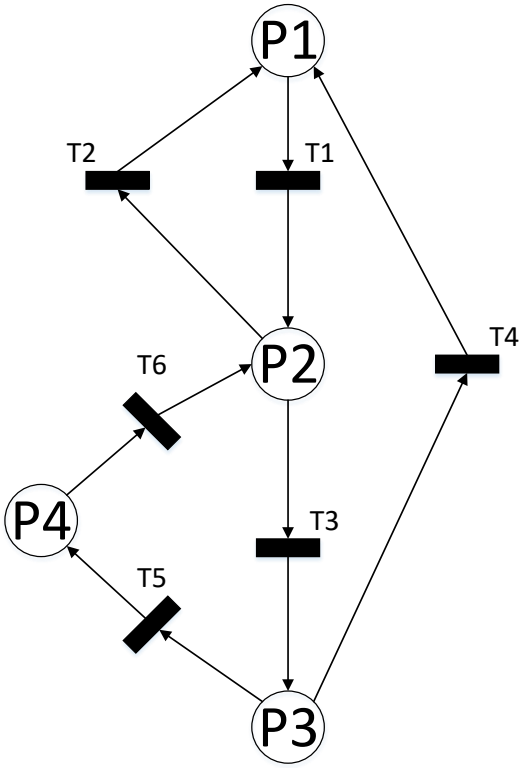

Figura 2 - RP utilizada no S1.

A partir desses dados pode-se escrever a matriz Q (5). Nesse caso ela será um esqueleto para o Algoritmo.

$$Q = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad (5)$$

Os pesos (6) foram escolhidos com o intuito de aumentar a probabilidade de P2, máquina operando e minimizar máquina em manutenção. Os locais P1 e P4 são desejáveis um valor baixo.

$$pesos = [0{,}5 \quad 1 \quad -1 \quad 0{,}5] \qquad (6)$$

Esta legenda da Figura 3, foi utilizada para todos os gráficos e também para o sistema 2.

Figura 3 - Legenda utilizada nos gráficos.

Analisando os resultados para P2 (Figura 4), nota-se que o valor converge rapidamente para 99,9% depois se mantem constante até o fim das 1.000 iterações. Os 3 métodos partem de aproximadamente 80%. Tanto o método elitista quanto o ranking apresentaram valores similares, aproximadamente 99,92% e 99,8% respectivamente. A diferença mais notável foi a convergência do elitista em relação ao ranking. O método roleta se mostrou muito ineficiente, tendo ficado em torno de 67,5%.

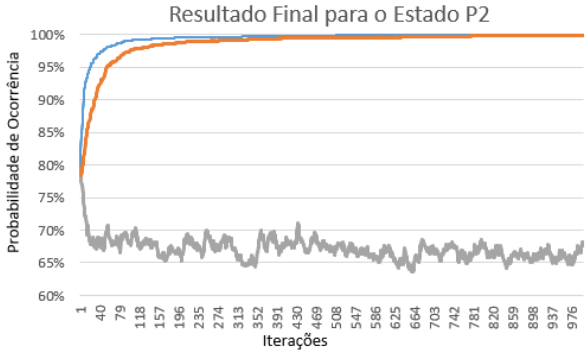

Figura 4 - Convergência para máquina operando (P2).

O desvio padrão de P2 (Figura 5), mostra o valor do desvio padrão das 1.000 iterações. Nota-se que o valor do método elitista nas 200 primeiras iterações é maior em relação ao ranking, isso se deve ao fato do método elitista convergir mais rapidamente. O método roleta demostrou-se instável.

Figura 5 - Desvio para máquina operando (P2).

A média final para os três métodos (Figura 6), após as 50 rodadas, o método elitista e ranking estão estáveis, já o método roleta se manteve instável, tendo mínimo de 55% e máximo de 88%.

Figura 6 - Média para máquina operando no estado (P2).

O tempo computacional (Figura 7), para o método elitista apresenta um pico inicial, pois é o primeiro método a ser rodado no programa, logo pode ser desconsiderado. Ocorreram 3 picos no método elitista e 2 no método de ranking, mas após 25 rodadas, os tempos permaneceram constantes e bem próximos um dos outros. A média de tempo computacional foi de 34 milissegundos para roleta e 39 milissegundos para elitista e ranking.

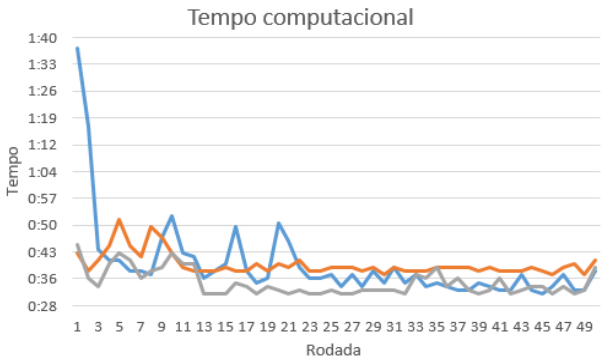

Figura 7 - Tempo computacional para S1.

A Figura 8 é semelhante à Figura 4, entretanto as iterações vão até 100 e não 1.000, foi feito desta maneira para que possibilitasse a melhor visualização do desempenho do algoritmo. O elitista convergiu mais rapidamente em relação ao método de ranking.

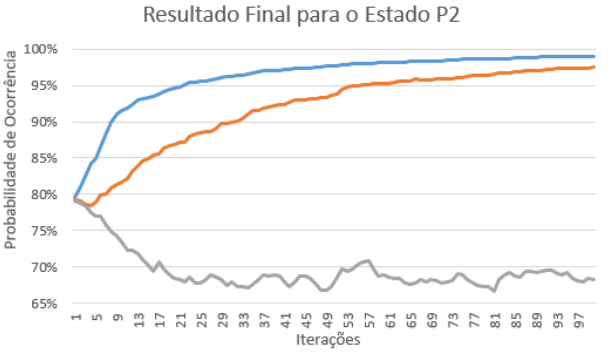

Figura 8 - Destaque da convergência para P2.

A Figura 9 é semelhante à Figura 5, entretanto as iterações vão até 100 e não 1000. O desvio padrão é maior no elitista em relação ao ranking, isso se deve a população inicial, pois afeta mais o método elitista que o método ranking.

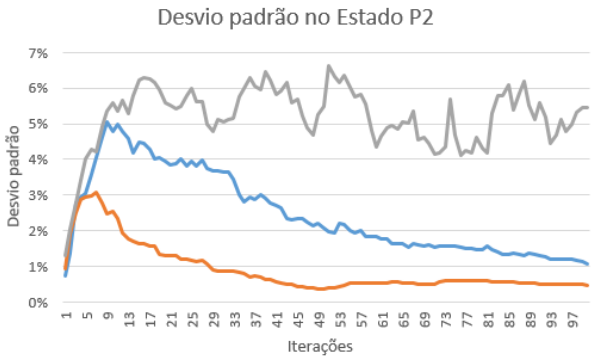

Figura 9 - Destaque para o desvio para P2.

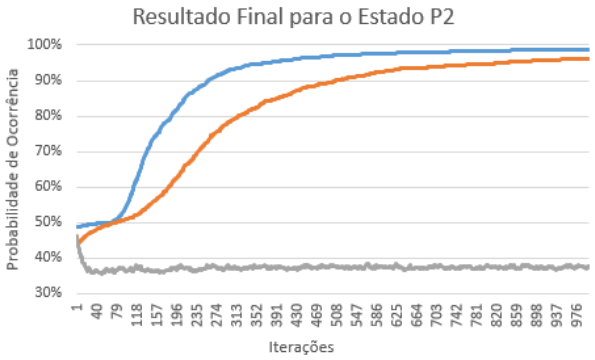

Figura 10 - Melhoria das gerações para P2 em S2.

O método roleta demonstrou-se incapaz de chegar nos valores encontrados pelos outros 2 métodos. Este método de seleção, no problema proposto, aparentemente ficou preso em mínimo local, não conseguindo encontrar o mínimo global, isso explicaria a constante oscilação nos valores.

III.2 Sistema Teste 2 (S2)

Esta RP foi feita para testar a capacidade do algoritmo em resolver problemas mais complexos, contudo o estado desejado para ter a máxima probabilidade foi P2 novamente. O desenho da RP não consta no artigo devido à alta complexidade. A rede possui 12 lugares e 40 transições. A matriz esqueleto Q (7) desse sistema:

$$Q = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

Os pesos dos estados (8):

$$[0.5, 1, -1, 0.5, 0.5, 0.5, \\ 0.5, -1, -1, 0.5, 0.5, 0.5] \quad (8)$$

O sistema 2 teve início em média de 49% para o método elitista e 46 % para ranking e roleta. Chegando em 98,75% para elitista, 96,15% para ranking e 37,34% para roleta.

É possível ver o comportamento do algoritmo ao longo das 1000 iterações (Figura 10). Inicialmente o método elitista e ranking não tiveram resposta até em torno de 120 iterações, após isso começou a aumentar a probabilidade de ocorrência em S2, o método elitista respondeu mais rapidamente em relação ao método de ranking. O método roleta não conseguiu melhorar a probabilidade, ocorrendo o contrário, começou em 46% e chegando em 37%.

O desvio padrão (

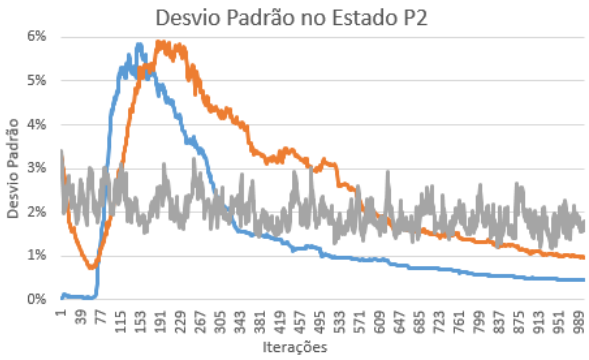

Figura 11) teve um pico para os métodos elitista e ranking, devido a população inicial gerada aleatoriamente. O método roleta manteve-se instável.

A média do elitista (Figura 12) foi melhor em relação ao ranking, com uma diferença de 3%. O método roleta obteve uma resposta média de 37,5%, inferior aos outros métodos.

O tempo computacional (Figura 13) foi parecido entre os métodos, 2 segundos para elitista e ranking, e 1 segundo e 50 milissegundos para o método roleta. Os picos computacionais ocorreram quando a população inicial se encontrou distante do ponto ótimo. O valor inicial de 4 segundos e 17 milissegundos do método elitista pode ser desprezado, devido ao fato de ser a inicialização do programa, isso acarreta um gasto computacional maior.

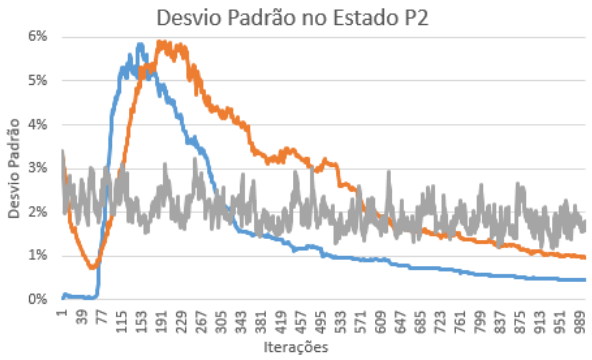

Figura 11 - Desvio de P2 em S2

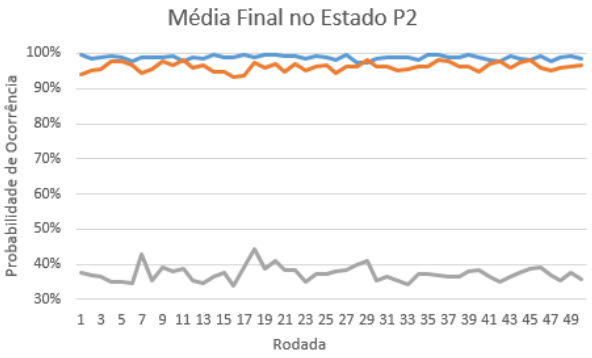

Figura 12 - Média final em P2.

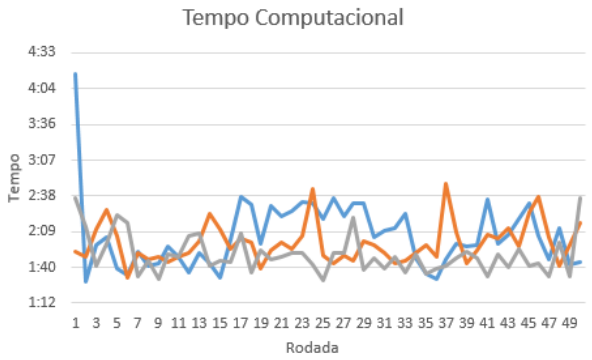

Figura 13 - Tempo computacional para S2.

IV. Conclusão

A utilização de metodologia evolutiva de otimização para ajustar uma RP demonstrou-se funcional, pois no sistema teste 1 inicialmente a probabilidade do estado P2 era de 80% no método manual, chegou a 99,92% e 99,8% com os métodos elitista e ranking no AG. No sistema teste 2 foi 49% e 46% para o método elitista e ranking, e ao término 98,75% e 96,15% respectivamente. Uma melhora de 19,9% no sistema 1 e de quase 50% no sistema 2. O método roleta foi incapaz de melhorar, em ambos os sistemas houve uma queda de 80% para 67% no sistema 1 e de 46% para 37% no sistema 2.

O tempo computacional médio no sistema teste 1 foi de 38 milissegundos para uma rede de 4 lugares, no sistema 2 o tempo médio foi de 2 segundos para uma rede de 12 lugares. Houve um aumento de 3 vezes do número de lugares e de 3,2 vezes no tempo computacional. Como é um problema do tipo NP-completo essa divergência de valores era esperada, a diferença seria maior caso o sistema 2 possuísse mais variáveis. Mesmo no sistema 2 o tempo sendo de 2 segundos, ainda é viável utilizar AG para otimizar RP. A complexidade do cromossomo aumentou de 6 variáveis para 40 variáveis um aumento de mais de 6 vezes.

Pode-se destacar pontos positivos e negativos do método Petri+AG. Em ambos os sistemas testes o resultado final foram satisfatórios, 99% para o S1 e 98% para o S2, demostrando a capacidade do método em buscar o ponto ótimo. O tempo computacional não foi uma adversidade, visto que foi de 34 milissegundos para S1 e 2 segundos para o S2, por essa perspectiva, RP com 12 ou menos lugares o tempo computacional não é relevante.

Entretanto foram observados pontos negativos como no sistema 1, em que o método elitista e ranking obtiveram resultados semelhantes, mas para o sistema 2 o elitista se saiu melhor em relação ao ranking. Para o método roleta ocorreu justamente o contrário, comparado ao elitista e ranking, houve uma queda no valor final em relação ao inicial, ao invés de ocorrer uma melhora. Demonstrando que a escolha do método de seleção é de suma importância. O alto desvio padrão ocorreu em todos os métodos, e isso se deve a inicialização randômica, deixando o resultado final com uma elevada dispersão.

Para trabalhos futuros é relevante a otimização para outros algoritmos evolutivos como Swarm, AIS, evolução diferencial, a fim de comparar o tempo computacional observado no AG. Outro fato relevante, para trabalhos futuros é a colocação da heurística na busca de indivíduos no AG para soluções pré-definidas.

V. Agradecimentos

Agradeço a Larissa Vieira da Silva, pelo apoio e por confiar em mim.

VI. Referências

[1] QIAO, F.; MA, Y.; LI, L. A Petri Net and Extended Genetic Algorithm Combined Scheduling Method for Wafer Fabrication. **IEEE Transactions On Automation Science And Engineering**, v. 10, n. 1, p.197-204 2013.

[2] SILVA, A. N.; LINS, F. A. A.; JUNIOR, J. C. S.; ROSA, N. S.; QUENTAL, N. C.; MACIEL, P. R. M. Avaliação de Desempenho da Composição de Web Services Usando Redes de Petri. In: SBRC, 2006, Curitiba. 24o. Simpósio Brasileiro de Redes de Computadores.

[3] MENEZES, F.A.A.; BARROSO, G.C.; PRATA, B.A. Restrições de controle sobre cores decompostas: uma proposta no controle supervisório de sistemas a eventos discretos utilizando redes de petri coloridas. **Revista Controle & Automação**, Campinas v.23, n.3, p.356-373, 2012.

[4] MACIEL, P.R.M.; LINS, R.D.; CUNHA, P.R.F. **Introdução às Redes de Petri e Aplicações**. 1.ed. Campinas, X Escola de Computação, 1996.

[5] BAUSE, F.; KRITZINGER, P. S. **Stochastic Petri Nets - An Introduction to the Theory**. 2. Ed. Braunschweig, Vieweg, 2002.

[6] LEITE, P.T.; CARNEIRO, A.A.F.M.; CARVALHO, A.C.P.L. Aplicação de algoritmos genéticos na determinação da operação ótima de sistemas hidrotérmicos de potência. **Revista Controle & Automação**, Campinas, v.17, n.1, p.81-88, 2006.

[7] HUANG, Y. Computing quantum discord is NP-complete. New Journal of Physics, v.16, 2014. Cury, J.E.R. Teoria de controle supervisório de Sistemas e Eventos discretos. 1. Ed. Canela – RS, V Simpósio Brasileiro de Automação Inteligente, 2001.

[8] GOLDBERG, D. E.; HOLLAND, J. H. **Genetic algorithms and machine learning**. Machine learning, ed. 1, 1989.

BIOGRAFIA

Fernando Oliveira Pereira
Nasceu em Pouso Alegre (MG), em 1994. Graduando em Engenharia de Controle e Automação na Universidade Federal de Itajubá (UNIFEI). Com interesse em Inteligência Artificial, processos estocásticos, cadeia de Markov e controle adaptativo.