INGEPRO – Inovação, Gestão e Produção
Abril de 2011, vol. 03, nº. 04
ISSN 1984-6193
www.ingepro.com.br

Sistema Computacional para o auxílio à Programação da Produção Reativa utilizando Algoritmos Genéticos

Edilson Reis Rodrigues Kato <kato@dc.ufscar.br>
Orides Morandin Jr. <orides@dc.ufscar.br>
Heito Leon Marson <heitormarson@hotmail.com>
Marcos Abraão de Souza Fonseca <marcos_abraao@dc.ufscar.br>

Resumo: Sistemas Computacionais tem sido muito utilizados no Planejamento da Produção no apoio à operação dos sistemas de manufatura, ou seja, estão relacionados com o chão de fábrica e planejamento e controle da produção levando-se em conta, muitas vezes processos de tomada de decisão. Os sistemas computacionais em desenvolvimento no TEAR - Laboratório de Pesquisa e Inovação em Tecnologias e Estratégias de Automação do Departamento de Computação - DC da UFSCar, tratam em tempo real as informações do chão de fábrica e as informações administrativas de forma a atuarem de uma maneira rápida e eficiente. O objetivo desta proposta é projeto e implementação de um sistema computacional de auxílio ao seqüenciamento da produção que atue de forma reativa, isto é, na ocorrência de um evento inesperado tal como a quebra de uma máquina. Esse sistema computacional de auxílio à decisão deve atuar de forma inteligente, isto é, será baseado em técnicas de Inteligência Artificial (IA). A técnica utilizada será baseada em Algoritmos Genéticos (AG), a qual irá atuar na melhora de desempenho do sistema de seqüenciamento da produção.
Palavras-chaves: Programação da Produção Reativa, Algoritmos Genéticos, Inteligência Artificial.

Computational System for aid to Reactive Production Scheduling using Genetic Algorithms

Abstract: computational systems has been widely used in the production planning in supporting operation of manufacturing systems, i.e. are related to the plant floor and production planning and control given often decision-making processes. Developing computational systems in TEAR-laboratory of research and Innovation in automation technologies and Strategies of the Department of science computer-DC of UFSCar, dealing with real-time information from the factory floor and administrative information to act quickly and efficiently. The purpose of this proposal is to design and implementation of a system of aid to the sequencing of production that acts reactively, i.e. the occurrence of an unexpected event such as the breaking of a machine. This computation system of decision-making aid must act intelligently, i.e. will be based on techniques of Artificial Intelligence (AI). The technique used will be based on genetic algorithms (GA), which will act in improving system performance of production sequencing.
Keywords: Reactive Programming production, genetic algorithms, Artificial Intelligence..

1. Introdução

Nas décadas de 80 e 90, o foco da manufatura era a busca por produtos com baixo custo, o que mudou devido influências de mercado. Atualmente, a manufatura tem relevado fatores como qualidade, confiabilidade e flexibilidade.

Em nível de chão de fábrica, as empresas têm enfrentado um conflito entre produtividade e flexibilidade em sua linha de produção. Situações onde a flexibilidade não é tratada cuidadosamente, a eficiência da produtividade é afetada.

Para manter a situação de flexibilidade do sistema de manufatura, parâmetros devem ser controlados e o impacto da flexibilidade tem que ser considerado. A flexibilidade impacta no desempenho do custo final, na velocidade de produção e na complexidade da atividade de fabricação (AGUIRRE *et al.*, 2007).

Inserido nesse cenário, surgem pesquisas e trabalhos dedicados a estudar sistemas flexíveis de manufatura (FMS). Tais sistemas de grande complexidade possuem sistemas automáticos para transporte de materiais, máquinas de controle numérico, recursos compartilhados, produtos com roteiros alternativos de produção, e são controlados e supervisionados por computador. Assim, esses sistemas representam um estado de grande flexibilidade. Inseridos no contexto de sistemas flexíveis, surgem problemas na área de: despacho de veículos, modelagem do sistema, de tal forma que impacta o planejamento da produção (SLACK, 1993; AGUIRRE *et al.*, 2007).

Este trabalho está inserido na temática de pesquisa do TEAR - Laboratório de Pesquisa e Inovação em Tecnologias e Estratégias de Automação do Departamento de Computação – DC da UFSCar. No ambiente do TEAR há o desenvolvimento de vários projetos relacionados com sistemas computacionais os quais envolvem elementos de automação industriais, técnicas de Inteligência Artificial e métodos computacionais, aplicados às necessidades de sistemas de manufatura reais.

Como motivação para a elaboração deste trabalho, considera-se: trabalhos relacionados à programação da produção como o algoritmo genético híbrido para a programação da produção (GONÇALVES *et al.*, 2004); algoritmos evolutivos híbridos com a união de algoritmos genéticos e técnicas de otimização extrema para a programação da produção (CHEN *et al.*, 2008); programação para sistemas flexíveis *job-shops* via otimização hierárquica (ZRIBI *et al.*, 2007).

Assim, partindo do conceito de sistemas flexíveis de manufatura, este trabalho encontra-se diretamente relacionado à programação da produção de sistemas de manufatura automatizados.

O objetivo deste trabalho é o projeto de um sistema computacional para o auxílio à programação e reprogramação reativa da produção. Tal sistema tem como foco o chão de fábrica. Ao ocorrer eventos imprevistos no decorrer do processo, como a quebra de uma máquina, o sistema possibilitará uma reprogramação da produção. Em um espaço de tempo muito pequeno, o responsável pelo processo de manufatura insere os novos dados relativos ao processo (abrangendo a quebra da máquina) no sistema e, em seguida, obtém a reprogramação da produção de maneira reativa.

O sistema é dividido em dois módulos. O primeiro módulo possibilita a descrição do cenário onde o usuário fornece as características do ambiente em que será realizada a programação da produção. Após a inserção dos dados de entrada do sistema (como, por exemplo, número de máquinas, número de produtos, roteiros de produção de cada produto,

tempo de processamento de cada operação, entre outros), é realizada a busca de uma solução que denota a programação da produção, baseada na técnica de algoritmos genéticos. Prevendo situações adversas como quebra de máquinas e ausência de matéria-prima, o sistema possui uma função de re-planejamento. Para isto, um segundo módulo do sistema foi projetado onde o usuário fornece a alteração ocorrida na linha de produção, que por conseqüência denota a especificação de um novo cenário. Tal cenário possui característica estática, e não dinâmica, por considerar apenas cenário especificado e não por um monitoramento contínuo do ambiente.

2. Revisão da Literatura

É possível tratar problemas de programação da produção utilizando métodos exatos e métodos de aproximação. No problema em consideração, devido sua característica *NP-hard*, métodos exatos sobrecarregam o processamento computacional, tornando-se inviável para sua aplicação em cenários de grande escala. Ao contrário, métodos de aproximação não garantem a otimalidade da solução, porém apresentam soluções próximas a ótima em um tempo significativamente reduzido (BLUM e ROLI, 2003). Outra vantagem dos métodos de aproximação é a busca por todo o espaço de soluções, não se limitando a mínimos locais. Portanto, para este trabalho, escolheu-se método de aproximação como forma de tratar a programação da produção, mais especificamente a metaheurística algoritmos genéticos.

Chen *et al.* (2008) propõem um algoritmo evolutivo híbrido denominado GEO (*Genetic Extremal Otimization*) que utiliza as técnicas de algoritmos genéticos (AG) e otimização extrema (EO). O algoritmo é aplicado em um problema de programação da produção ambientado na indústria siderúrgica. A abordagem desenvolvida procura combinar a busca exploratória do AG com a busca local de grão fino (*fine-grained*) EO. O trabalho é apresentado de forma construtiva, começando a partir de um algoritmo genético modificado (MGA) que utiliza uma busca local como operador de mutação e algumas variantes nos procedimentos de inicialização, de seleção e na operação de crossover, o resultado gerado pelo MGA é então melhorado pelo algoritmo de otimização extrema (EO). Por fim a abordagem proposta utiliza a técnica EO como operador de mutação do MGA definindo assim o GEO. Os resultados são uma comparação entre as três variantes e a abordagem GEO obtém resultados mais significativos.

Gonçalves *et al.* (2004) propõem um algoritmo genético hibrido pra o problema de programação em sistemas *job shop*. A principal contribuição da proposta está em uma classe de programação denominada Programação Ativa Parametrizada (*Parameterized Active Schedules – PAS*), que está incorporada no processo de seleção. Uma das características do algoritmo é o comprimento do cromossomo ser estabelecido por *2n* genes onde *n* é o número de operações. A decodificação do cromossomo considera que os *n* primeiros genes representam a prioridade das operações em um intervalo [0,1] e os *n* genes seguintes representam o tempo de atraso utilizado por cada iteração da programação. A mutação consiste em gerar um novo indivíduo aleatoriamente, ao invés de inversão de gene por gene. O procedimento de programação, como mencionado, de certa forma faz parte da seleção. Ele consiste em limitar o espaço de busca por restringir a busca somente às operações que estão dentro de um limite máximo de atraso permitido.

Um procedimento de busca local é utilizado para melhorar o valor do makespan, ele consiste em identificar o caminho crítico na solução obtida no procedimento de programação, e então efetuar movimento de troca entre as operações da mesma máquina. Se a aptidão do indivíduo melhorar então a troca é mantida, do contrário é desfeita.

Os resultados alcançados pela abordagem são apresentados a partir de uma comparação entre outras nove abordagens. Uma Heurística, quatro Algoritmos Genéticos, duas abordagens utilizando GRASP, um Algoritmo Genético híbrido com *Simulate Annealing* e por fim uma abordagem com Busca Tabu. Em relação a todas as instâncias de testes, a abordagem proposta obteve um melhor resultado que as demais abordagens, com exceção ao algoritmo de Busca Tabu.

Zribi *et al.* (2007) propõem um método hierárquico para problemas de programação job-shop em sistemas flexíveis de manufatura. Na abordagem do problema, são considerados três critérios: carga total de trabalho das máquinas, carga de trabalho da máquina crítica, e o tempo total de utilização da linha de produção (*makespan*). Dividiu-se o problema, inicialmente, em subproblemas e em seguida tratou-se do seqüenciamento dos primeiros.

Um dos problemas tratado foi a sobrecarga de trabalho nas máquinas. Para isto, foi proposto um algoritmo de busca local, onde a operação a ser realizada sempre é alocada na máquina com menor carga de processamento acumulada. Para otimizar a busca local, visando a minimização do *makespan*, também tratou-se o problema com método exato, o *Branch-and-Bound*. De forma geral, para o problema tratado foi proposto um algoritmo de duas fases e várias abordagens heurísticas foram adotadas (métodos de localização e busca tabu com algotimo *branch-and-bound*). E para avaliar as possíveis soluções, um algoritmo genético híbrido foi utilizado. Os resultados mostraram qualidade e adequação ao cenário em questão.

Por conseguinte, utilizando os trabalhos descritos acima como motivação, o tema deste trabalho tem como proposta o seqüenciamento de roteiros de produção. Utilizando Algoritmos Genéticos, propõe-se a modelagem de um ambiente flexível de manufatura onde objetiva-se a otimização do *makespan* e a obtenção de uma ordem de produção que contenha a seqüência de processamento de produtos com seus respectivos roteiros de produção. Destacando que o objetivo é um seqüenciamento em alto nível, não se preocupando com o seqüenciamento das operações, mas dos roteiros de operações.

3. Descrição do Problema

O problema consiste em determinar os roteiros de produção para *P* produtos com uma disponibilidade de *M* máquinas. O tempo de operação de um produto em uma determinada máquina é pré-definido. São ainda consideradas as seguintes restrições:

- Cada máquina executa somente uma operação por vez.
- Não é permitido preempções das operações.
- Todas as máquinas estão disponíveis no tempo t=0.
- As máquinas não quebram.
- O tempo de transporte de um recurso para a máquina M_k não é considerado.
- O tempo de configuração é o mesmo para todas as máquinas.
- A ordem de execução das operações para cada tarefa é fixa e não pode ser alterada.

Em outras palavras, o problema consiste em alocar todas as tarefas às máquinas, o que nos permite, por conseqüência, determinar a ordem de execução de cada tarefa de acordo com a função objetivo. A função objetivo usada será o makespan, o tempo necessário para executar todas as tarefas programadas para produzir os *P* produtos com *M* máquinas disponíveis. O makespan pode ser denotado por:

$$C_{Max} = TF_{ij} \text{ Onde } (i \leq N, j \leq M) \quad (1)$$

Onde, C_{Max} indica o valor da última operação j do último produto i. TF_{ij} é o tempo de término de processamento.A solução do problema é fornecida com a ordem de produção de cada produto com seus respectivos roteiros.

Os roteiros de produção são definidos por P_iR_k onde P_i representa o i-ésimo produto e R_k o k-ésimo roteiro do produto P_i. As informações sobre os roteiros são previamente conhecidas e adquiridas como ilustrado na Tabela 1.

P_1	R_1	M1	M2	M3	-
	R_2	M2	M3	M5	-
P_2	R_1	M2	M3	M4	-
	R_2	M1	M2	M4	M5
P_3	R_3	M1	M2	M5	-

Tabela 1: Roteiros de produção para os produtos P_1, P_2 e P_3.

Neste exemplo os produtos P_1 e P_2 possuem cada um dois roteiros distintos e o produto P_3 apenas um. Como a tabela indica, o produto P_1 estará completo após a execução de suas operações nas máquinas *M1*, *M2* e *M3* ou *M2*, *M3* e *M5*. A partir destas informações, a matriz da Tabela 2 é construída, onde cada valor indicado representa o índice da máquina presente no roteiro. Por exemplo, os valores 1, 2 e 3 do produto P_1 representam as máquinas *M1*, *M2* e *M3* da Tabela 1 e o valor 0 indica a inexistência de demais máquinas para este roteiro. Este último valor é necessário devido ao número de elementos no roteiro do produto P_2 em seu segundo roteiro, sendo que uma matriz deve respeitar os valores de suas dimensões.

P_1	R_1	1	2	3	0
	R_2	2	3	5	0
P_2	R_1	2	3	4	0
	R_2	1	2	4	5
P_3	R_1	1	2	5	0

Tabela 2: Roteiros de produção com o índice das máquinas para os produtos P_1, P_2 e P_3.

3.1. Cenário

O cenário, referente ao ambiente de manufatura, é criado a partir de informações características da linha de produção e dados referentes ao lote de produtos a serem produzidos. Para o algoritmo ser executado, é necessário que o cenário esteja definido e dimensionado com as informações citadas.

Esta condição de possuir o cenário já definido para que o algoritmo seja executado, caracteriza o cenário como estático. Do contrário, se o cenário fosse monitorado, seria caracterizado como dinâmico.

4. Algoritmos Genéticos

Algoritmos Genéticos são métodos de otimização e busca inspirados nos mecanismos de evolução de populações de seres vivos. Introduzidos por John Holland (HOLLAND, 1975) e popularizado por um de seus alunos, David Goldberg, (GOLDBERG, 1989). Estes algoritmos seguem o princípio da seleção natural e sobrevivência do indivíduo mais apto, declarado pelo inglês Charles Darwin, em 1859, em *A Origem das Espécies.*

Os mecanismos de evolução de populações de seres vivos partem do princípio que, em uma dada população, o indivíduo com boa característica genética possui maiores probabilidades de sobrevivência e de reprodução, produzindo indivíduos cada vez mais aptos.

Assim, os indivíduos menos aptos tendem a desaparecer.

Na utilização de Algoritmos Genéticos em problemas reais, cada indivíduo da população é denominado cromossomo, representando uma possível solução para o problema. Mecanismos de reprodução, baseados no processo evolutivo, são utilizados para realizar a busca pelo espaço de solução e encontrar a solução de maior aptidão.

Os Algoritmos Genéticos se diferenciam dos métodos tradicionais de busca e otimização, principalmente, por quatro fatores:

– Trabalham com codificação de conjunto de parâmetros e não com os parâmetros em si.
– Atuam em populações e não em um único ponto;
– Utilizam informações de custo e não derivadas ou conhecimento auxiliar;
– Possuem regras de transição probabilística e não determinística (RESENDE *et al.*, 2003).

O fluxograma abaixo ilustra o funcionamento dos Algoritmos Genéticos.

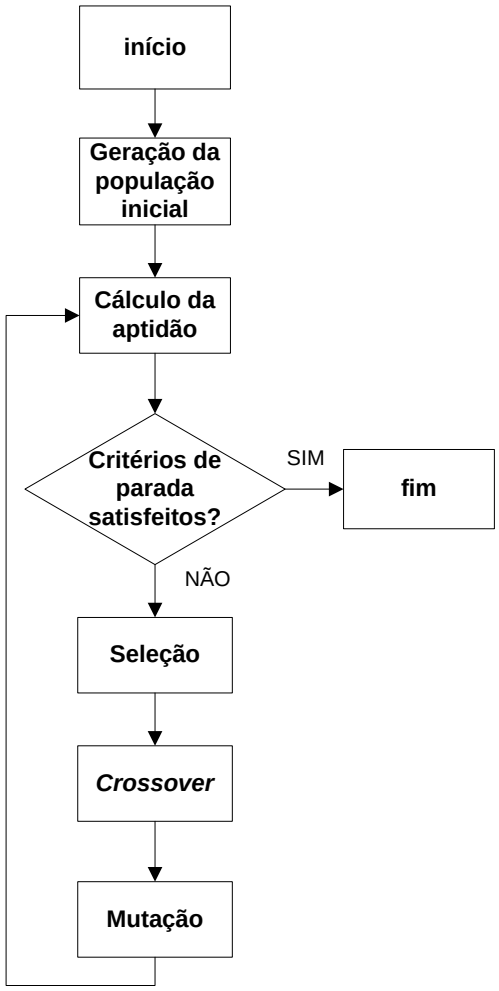

Figura 1 – Funcionamento de um AG

5. Representação do Problema baseado em Algoritmos Genéticos

5.1 Considerações Iniciais

O problema foi baseado em A Search Method using Genetic Algorithmn for Production Reactive Scheduling of Manufacturing Systems (MORANDIN *et al.*,2008) como

fundamento para um programa de iniciação científica.

5.2 Função *Fitness*

A função *fitness*, ou função objetivo, no nosso problema, será representada pelo *makespan*.

*Makespa*n é o tempo total de produção. Utilizando o princípio de que cada máquina realiza apenas uma operação por vez, nosso objetivo é minimizar o m*akespan* do roteiro de produção. O *makespan* é calculado quando o último produto é obtido.

O gráfico abaixo ilustra a ocupação das máquinas em função do tempo. No exemplo abaixo, obtivemos um *makespan* de 3,3 unidades de tempo.

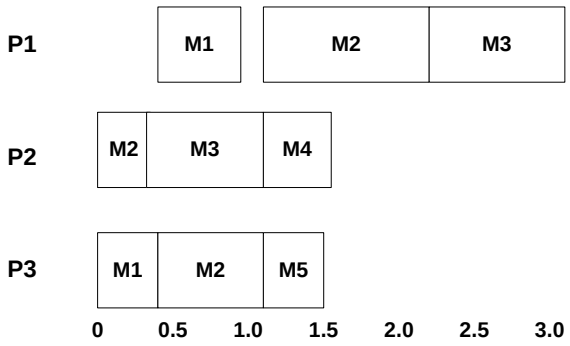

Figura 2- Gráfico de Gantt com as operações dos roteiros e suas respectivas máquinas

5.3 Cromossomo

No problema abordado, o cromossomo nos fornecerá, por ordem de produção, o produto e seu respectivo roteiro. Há a possibilidade de os produtos possuírem roteiros de produção de tamanhos diferentes. Nesse caso, adota-se como referência o maior roteiro, e os roteiros menores recebem zeros nas posições onde não possuem máquinas para realizarem tarefas.

5.4 Geração da População Inicial

Na geração da população inicial é utilizada a Codificação por Permutação.

Na codificação por Permutação, cada cromossomo é uma série de números que representa uma posição em uma seqüência. Tal codificação é muito utilizada para solução de problemas de ordenação, exemplo: Problema do Caixeiro Viajante (Travelling Salesman Problem).

O problema é descrito de tal forma: são dadas cidades e as respectivas distâncias entre elas. O caixeiro viajante tem que visitar todas elas, sem percorrer uma distância maior que a necessária. A solução se resume em encontrar a seqüência de cidades em que as viagens devem ser feitas de forma que a distância percorrida seja a mínima possível.

A codificação ocorre da seguinte forma: Os cromossomos descrevem a ordem que o caixeiro percorrerá as cidades.

Aplica-se a Codificação por Permutação em programação da produção, roteamento de veículos, projeto de circuito integrado, etc.

Em nosso problema, ao invés de cidades, utilizaremos as máquinas como destinos para

os produtos.

Para exemplificar, usaremos os dados da Tabela 2 e Tabela 3. O cenário será a produção de três produtos com cinco máquinas disponíveis. Os possíveis roteiros dos produtos já foram determinados na Tabela 2, sendo que os produtos 1 e 2 possuem dois roteiros cada um. Outro dado a ser considerado é o tempo de processamento de cada produto nas respectivas máquinas de seu roteiro.

	P_1	P_2	P_3
M_1	0.5	0.9	0.4
M_2	1.1	0.3	0.7
M_3	0.9	0.8	0.7
M_4	0.4	0.5	0.7
M_5	0.8	0.6	0.4

Tabela 3 - Exemplo de tempo de processamento das operações.

A população inicial é gerada de maneira aleatória. Ocorre a permutação de todas as possíveis ordens de produção de produtos. Para ilustrarmos, escolheremos a permutação 3-1-2, ou seja, a ordem de produção se inicia com o produto 3, seguido do produto 1 e termina com o produto 2. Após definirmos a ordem dos produtos, escolhe-se um dos roteiros disponíveis para cada produto. Utilizando a Tabela 2, é atribuído o primeiro roteiro (R1) para os produtos 1 e 2 e o roteiro R1 para o produto 3. Gerando o cromossomo ilustrado na Figura 2.

3	1	2	5	0	**1**	1	2	3	0	**2**	2	3	4	0

Figura 3 – Cromossomo gerado após a permutação dos roteiros

5.5 Mutação

Neste exemplo, a mutação é aplicada no roteiro de um determinado produto que compõe o cromossomo.

A mutação acontece selecionando-se qual produto do cromossomo sofrerá alteração no seu roteiro de produção.

5.6 Cruzamento

O cruzamento, ou *crossover*, é aplicado em dois cromossomos, chamados pais, gerando 2 cromossomos filhos. No problema abordado, aplicar o cruzamento significa selecionar dois cromossomos e trocar entre eles o roteiro de um determinado produto. O cruzamento ocorre após definir o ponto de corte, ou seja, qual o pedaço do cromossomo será transferido. Os cromossomos selecionados são chamados de pais, e os cromossomos gerados são chamados de filhos.

6. Roteiro de Implementação de um Algoritmo Genético em Matlab

Nesta sessão é apresentada a aplicação de um algoritmo genético para o problema de Programação da Produção em um Sistema Flexível de Manufatura.

6.1 Arquivos

Para executar a implementação de um algoritmo genético em Matlab, cria-se um arquivo principal, neste exemplo denominado como *agManufacturing*, onde serão invocados os demais arquivos que contém os operadores genéticos sendo esses denominados como *crossover ,fitness , mutate, permutations*.

- *agManufaturing*: é onde aloca-se a variável que declara o tempo de operação de cada produto em cada máquina. Este arquivo é responsável por invocar os arquivos *permutations*, *fitness*, *crossover* e *mutate*, e iniciar o funcionamento do algoritmo.
- *permutations:* este arquivo é responsável por determinar como a população inicial será gerada.
- *fitness:* neste arquivo consta o procedimento para o cálculo do *makespan*.
- *crossover:* arquivo onde ocorre o cruzamento dos cromossomos.
- *mutate:* arquivo onde ocorre o procedimento de mutação.

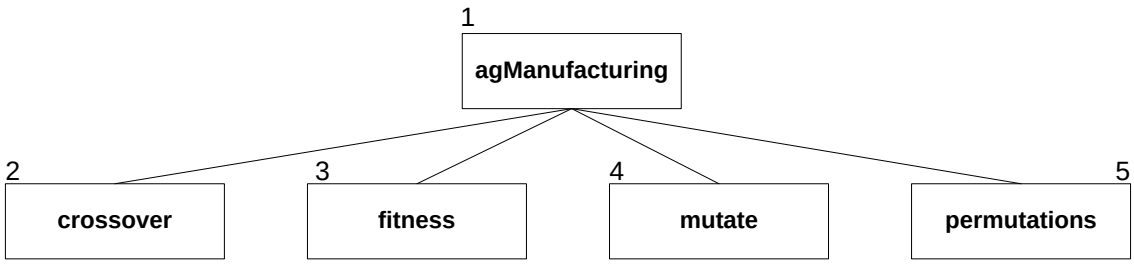

Figura 4 – Fluxograma com o funcionamento do arquivo AgManufacturing

7. Desenvolvimento Gráfico do Sistema

O desenvolvimento gráfico do sistema, ou seja, o projeto da interface do sistema que irá se comunicar com o usuário foi desenvolvido no ambiente do Microsoft Visual Studio 2008.

O projeto das janelas foi baseado em simplicidade de *design* e funcionalidades.

A janela inicial foi projetada para o usuário escolher entre 2 opções: Planejamento ou Replanejamento. Através de botões o usuário seleciona a opção desejada.

Seguindo o projeto do sistema, projetou-se uma janela de planejamento (Figura 5). Esta tem como objetivo a coleta dos dados de entrada. Esta janela possui campos para o usuário digitar: número de máquinas, número de produtos, roteiros de produção dos produtos e tempo de processamento de cada produto em cada máquina. Após a inserção de todos os dados, o usuário seleciona o botão “Realizar Planejamento”. Após tal comando, o sistema irá iniciar a execução do algoritmo genético.

Outra janela (Figura 6) foi projetada para realizar o replanejamento da produção. No cotidiano da indústria, eventos não convenientes ao processo de manufatura acontecem como a quebra de uma máquina, tornando esta inativa, e o replanejamento se mostra necessário. E para reduzir o tempo de replanejamento, esta segunda janela do sistema é projetada para o usuário citar quais máquinas se encontram inativas. Assim, o algoritmo será processado e fornecerá um novo planejamento com as máquinas disponíveis.

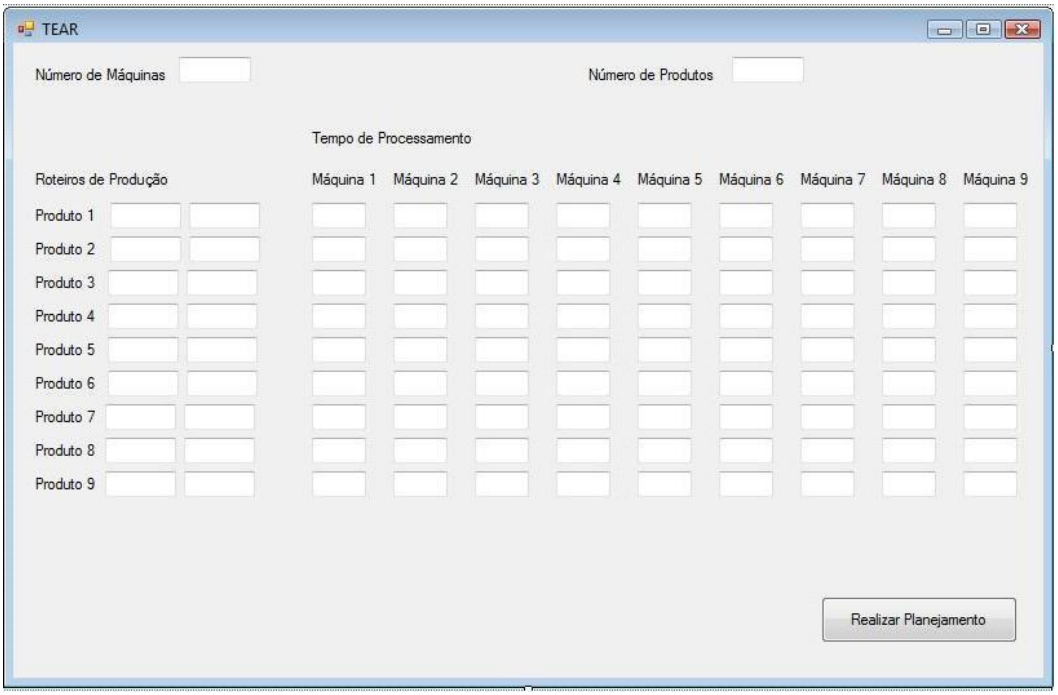

Figura 5 – Janela de Planejamento

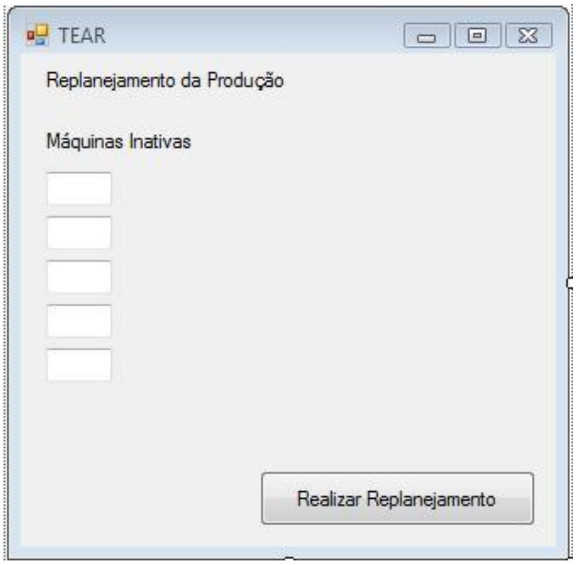

Figura 6 – Janela de Replanejamento

Por fim, uma última janela foi projetada para exibir o resultado do planejamento ou replanejamento da produção. Esta janela possui como informação final ao usuário: o makespan e a ordem de produção.

Atualmente, o sistema se encontra em fase de operação no ambiente TEAR.

8. Resultados Obtidos

A implementação foi baseada no trabalho A Search Method using Genetic Algorithmn for Production Reactive Scheduling of Manufacturing Systems (MORADIN *et al.*, 2008).

O cenário é caracterizado como 9x9 (Produtos x Máquinas). A Tabela 4 contem os produtos e os respectivos roteiros para o cenário em questão.

Produto	Roteiros
1	$M_1M_2M_4M_5M_7M_9$
	$M_3M_4M_5M_6M_8M_9$
2	$M_1M_2M_3M_4M_5M_6M_7$
	$M_2M_3M_5M_7M_8M_9$
3	$M_4M_5M_6\ M_7\ M_8$
	$M_2M_3M_7\ M_8\ M_9$
4	$M_2M_3M_4M_5M_6\ M_7$
	$M_1M_5M_6M_8M_9$
5	$M_4M_5M_6M_8M_9$
	$M_1M_2M_3M_5M_6$
6	$M_2M_4M_5M_6M_7M_8M_9$
	$M_1M_3M_6M_7M_8M_9$
7	$M_1M_2M_4M_5M_6M_9$
	$M_1M_2M_3M_7M_8M_9$
8	$M_4M_5M_6M_7M_8M_9$
	$M_3M_4M_5M_7M_8M_9$
9	$M_3M_5M_6M_7M_8M_9$
	$M_2M_4M_6M_7M_8M_9$

Tabela 4 – Produtos e Roteiros

Após 35 execuções do algoritmo, o *makespan* médio obtido foi de 4831,314 unidades de tempo.

O melhor resultado apresentado foi de 4673 unidades de tempo. O gráfico abaixo ilustra o desempenho do algoritmo que obteve o melhor resultado.

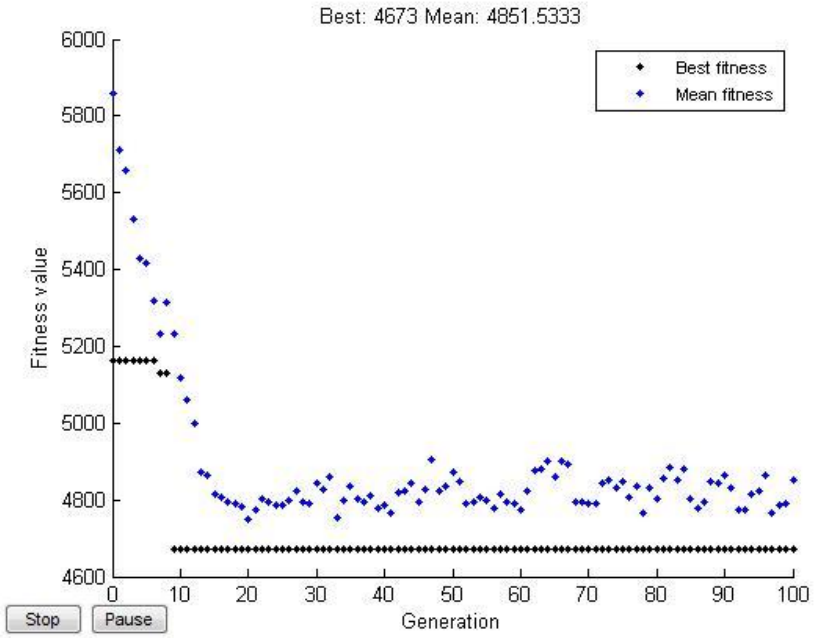

Figura 7 – Desempenho de algoritmo

Morandin *et al.* (2008) realizaram comparações com outros métodos de busca propostos (Maggio, YU, RCR Cost, Reyes) para o mesmo problema e cenário em questão. Morandin *et al*. obtiveram resultado de qualidade superior aos outros métodos.

A implementação do trabalho aqui apresentado mostrou melhor qualidade em relação ao resultado apresentado por Morandin *et al.*

9. Agradecimentos

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro (Programa Institucional de Bolsas de Iniciação Científica- PIBIC), fornecendo amparo e suporte para a realização deste trabalho.

Referências

AGUIRRE, L. A. *Enciclopédia de Automática Controle e Automação*. Vol 3. São Paulo: Blucker, 2007.

CARVALHO, A.C.P.L.F.; BRAGA, A.P. & LUDERMIR, T.B. *Computação Evolutiva*. In: REZENDE, S.O. *Sistemas Inteligentes: fundamentos e aplicações*. Manole, 2003. p 225-239.

C. BLUM, E A. ROLI *"Metaheuristics in combinatorial optimization: Overview and conceptual comparison"* ACM Comput. Surv. Vol. 35, 3, 268-308, Sep. 2003.

CHEN, Y.-W.; LU, Y.-Z.; YANG, G.-K. *Hybrid Evolutionary Algorithm with Marriage of Genetic Algorithm and Extremal Optimization for Production Scheduling*. The International Journal of Advanced Manufacturing Technology. vol. 36, 959-958. DOI-10.1007/s00170-006-0904-9, Springer, London, April, 2008.

GOLDBERG, D. E. (1989). *Genetic algorithm s in search,*

optimization, and machine learning. Addison-Wesley.

GONÇALVES, J. F., MAGALHÃES MENDES J. J., AND RESENDE, M. G. C. *A Hybrid Genetic Algorithm for the Job Shop Scheduling Problem.* European Journal of Operational Research, vol. 167, 2002.

HOLLAND, J. H. (1975). *Adaptation in Natural and Artificial*

Systems. MIT Press.

MORANDIN JR., O.; KATO, E. R. R.; DERIZ, A. C.; SANCHES, D. S.; (2008) *A Search Method using Genetic Algorithmn for Production Reactive Scheduling of Manufacturing Systems*. ISIE 2008.

SLACK, N., CHAMBERS, S., HARLAND, C., HARRISON, A.; JOHNSTON, R. *"Administração da Produção"*, p 55 – 72, Atlas, 1993.

ZRIBI, N.; KACEM, I.; El KAMEL, A.; BORNE, P. *Assignment and Scheduling in Flexible Job-Shops by Hierarchical Optimization. Systems, Man, and Cybernetics*, Part C: *Applications and Reviews*, Vol. 37, Issue 4, July 2007 Page(s):652 – 661