Boletim Técnico da Escola Politécnica da USP

Deptº de Engª de Computação e Sistemas Digitais

BT/PCS/9408

Modelamento de Sistemas com Redes de Petri Interpretadas

Carlos Alberto Sangiorgio
Wilson V. Ruggiero

São Paulo - 1994

O presente trabalho e baseado na dissertação de mestrado "Uma Ferramenta para Descrição e Análise de Paralelismo e Sincronizacão ", apresentada em 1993, por Carlos Alberto Sangiorgio, sob orientação do Prof. Dr. Wilson V. Ruggiero.

A íntegra da dissertação encontra-se à disposição com o autor e na Biblioteca de Engenharia Elétrica da Escola Politécnica da USP.

Sangiorgio, Carlos Alberto
Modelamento de sistemas com redes de Petri interpretadas / C.A. Sangiorgio, W.V. Ruggiero. -- São Paulo : EPUSP, 1994.
19p. -- (Boletim Técnico da Escola Politécnica da USP, Departamento de Engenharia de Computação e Sistemas Digitais, BT/PCS/9408)

1. Análise de sistemas 2. Redes de Petri I. Ruggiero, Wilson Vicente II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais III. Título IV. Série CDD 003
003

Modelamento de Sistemas com Redes de Petri Interpretadas

(Systems Modelling using Interpreted Petri Nets)

Carlos Alberto Sangiorgio

Prof. Dr. Wilson V. Ruggiero

Resumo.

Este trabalho apresenta um método para modelamento de sistemas com paralelismo e pontos de sincronização. O método utiliza um programa de computador chamado ***RP_SIM*** para modelamento dos sistemas. O ***RP_SIM*** é um simulador de Redes de Petri interpretadas que abrange tanto a parte de descrição dos modelos como a extração de resultados.

Além de discussões sobre modelamento de sistemas e da descrição do ***RP_SIM,*** este trabalho apresenta um exemplo de utilização do método para análise de um sistema de manufatura.

Abstract.

This work presents a methodology to model parallel systems with syncronization points. The method uses a computer program called ***RP_SIM*** to model the systems. ***RP_SIM*** is a interpreted Petri nets simulator that permits description and analysis of the models.

Besides systems modelling discussions and ***RP_SIM*** presentation, this work describes a manufacturing system modelling example.

1. INTRODUÇÃO.

Existe ao nosso redor um grande número de sistemas que apresentam paralelismo e pontos de sincronização. Muitas vezes precisamos determinar o comportamento destes sistemas, sendo que, em boa parte dos casos, não podemos realizar esta tarefa, de forma adequada, pela simples observação do sistema real. Nestes casos podemos fazer o modelamento do sistema real e a análise deste modelo. Quando o sistema a ser modelado apresenta paralelismo e pontos de sincronização as Redes de Petri (RPs) se apresentam como uma ferramenta bastante atraente. Devido

a sua utilizacão no modelamento dos mais variados tipos de sistemas, uma série de variantes foram propostas em relação a RP original. Este grande número de variantes provocou o inconveniente de gerar uma série de ferramentas especificas para cada tipo de sistema.

Este artigo apresenta parte do que foi abordado em [SANG93] (UMA FERRAMENTA PARA DESCRICÃO e ANALISE DE PARALELISMO e SINCRONIZACÃO). Neste trabalho foram utilizadas RPs com interpretacão para modelamento de diversos sistemas, sendo que a grande vantagem desta classe de RPs e ser bastante generica. Serão apresentados:

a. Comentarios sobre modelamento de sistemas;

b. Descricão dos programas que permitem utilizar RPs com interpretacão para modelamento de sistemas;

c. Um exemplo de aplicacão.

Em [SANG93] são apresentados outros pontos relevantes tais como, o mapeamento de outras variantes de RPs (ou mesmo outras formas de descricão de paralelismo) para RPs com interpretacão, problemas de cada tipo de método utilizado para extracão dos resultados do modelamento, etc. São ainda apresentados uma serie de exemplos procurando mostrar a flexibilidade das ferramentas geradas, bem como tornar clara suas utilizações.

2.COMENTÁRIOS SOBRE MODELAMENTO DE SISTEMAS.

Independente da natureza de um sistema, quando desejamos analisar seu comportamento devemos definir a forma de extrair resultados que permitam determiná-lo. Em alguns casos, a definicão da forma de extrair resultados é uma tarefa muito simples, sendo que muitas vezes basta a simples observacão do próprio sistema. Em outros casos porém, esta definição pode ser uma tarefa muito complexa e que demanda, acima de tudo, imaginacão e capacidade de abstração. Entre os inúmeros fatores que contribuem para dificultar a determinacão do comportamento de dado sistema podemos citar:

a. Inexistência ou indisponibilidade de aparelhos de medida adequado para medir alguns de seus parâmetros;

b. Impossibilidade ou dificuldade de acesso aos pontos de interesse para medida;

c. Dificuldade de definição dos pontos de medida relevantes;

d. Inexistência do sistema que se pretende fazer a análise;

e. Tempo excessivo para coleta das informacões.

Quando este tipo de dificuldade se apresenta, uma possível solução é o modelamento do sistema e a posterior análise deste modelo para obtencão de resultados. A tarefa de modelamento está balisada em duas premissas básicas:

a. O modelo deve capturar o maior número de parâmetros relevantes possível do sistema real. Quanto maior o número de parâmetros relevantes capturados maior será a aderência do modelo;

b. O modelo deve ser o mais simples possivel para que possam ser feitas análises de forma simples e eficiente.

Evidentemente as duas premissas definidas acima apresentam afirmacões conflitantes. Cabe ao modelador a tarefa de capturar os pontos relevantes no sistema, de forma a ser o mais aderente possível e suficientemente simples para ser resolvido em tempo razoável.

Uma vez tendo o modelo do sistema, devemos cuidar da etapa de sua resolução. Novamente temos uma variedade de opções para execução desta etapa, sendo que entre elas podemos citar:

a. A simples observação do modelo permite muitas vezes identificar diversos pontos relevantes;

b. Resolução do modelo de forma a obter fórmulas (ou equações algébricas) que definam o comportamento de alguns parâmetros do sistema;

c. Resolução do modelo de forma a obter valores numéricos que definam o comportamento de alguns parâmetros do sistema, sendo estes valores obtidos, por exemplo, por resolução de sistemas de equacões lineares;

d. Resolução do modelo de forma a obter valores numéricos que definam o comportamento de alguns parâmetros do sistema, sendo estes valores obtidos por simulacão do modelo;

De forma geral, as opções que estabelecem equações algébricas para determinado parâmetro do sistema, impõem uma série de restrições ao modelo. Porém estas produzem resultados que claramente definem o comportamento do parâmetro em questão. Já as soluções numéricas permitem a utilização de modelos mais genéricos, necessitando porém

uma análise mais rigorosa para se definir, a partir dos valores obtidos, o comportamento do parâmetro em questão.

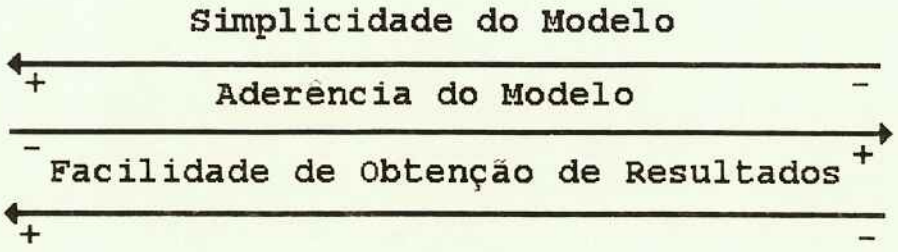

fig.2.1:Modelamento de Sistemas.

Como exemplo vamos realizar a análise de um sistema de memória com cache. Estamos interessados em determinar o tempo médio de acesso a está memória. Podemos, através de analisadores lógicos ou outros equipamentos deste tipo, realizar medidas no próprio sistema para determinação deste parâmetro. Podemos também, modelar o sistema através de uma rede de Petri ou rede de filas ou algo semelhante e em seguida resolver este modelo. A solução do modelo poderá apresentar uma equação tal como:

Ta = Tm (1 - x) + Tc (x)

onde: **Tm**: tempo de acesso da memória sem cache

Tc: tempo de acesso do cache

x: "hit-rate" do cache

Outro tipo de solução do modelo poderia produzir uma tabela do tipo:

x	0.5	0.8	0.95
Ta	500	200	100

É evidente que a primeira solução é mais confortável para se verificar a dependência do parâmetro tempo de acesso em relação aos outros parâmetros. Porém, é provável que uma série de simplificações tenham sido impostas ao modelo original para obtenção da equação. A segunda solução, embora não apresente de forma explicita as dependências, pode ter sido obtida a partir da resolução de um modelo mais completo e que portanto deve ser mais aderente ao sistema modelado.

No trabalho de modelamento são empregadas, em geral, um conjunto das diversas técnicas citadas. Na verdade, a resolução de um modelo simplificado pode servir de base para validação do modelo mais sofisticado.

3.LINGUAGEM DE DESCRIÇÃO DE RPs E OS PROGRAMAS *RP_NUM* E *RP_SIM*.

Em [SANG93] são apresentadas duas ferramentas de análise de RPs. Estas ferramentas são dois programas de computador chamados ***RP_NUM*** e ***RP_SIM***. O programa ***RP_NUM*** chega a resultados sobre o comportamento da RP através de metódos algébricos e numéricos, montando e resolvendo sistemas de equacões lineares. Já o programa ***RP_SIM*** chega aos mesmos resultados através de processo de simulação da RP. As descricões detalhadas de cada ferramenta, vantagens e desvantagens de cada uma delas, limitacões, etc podem ser encontradas em [SANG93]. Neste item iremos abordar superficialmente os pontos mais importantes das duas ferramentas

3.1. A linguagem de descrição da estrutura das RPs.

A linguagem proposta para descricão das RPs que é utilizada tanto pelo programa ***RP_NUM*** como pelo programa ***RP_SIM*** consiste em uma extencão da linguagem C++. O C++ permite a criacão de novos tipos de objetos que se comportam como se já fossem parte integrante da linguagem. Assim, por exemplo, pode-se criar um tipo "Complexo" e definir as operações "+", "-", "*", "/" (em C++ esta definicão é denominada sobrecarga de operadores). Uma vez criado o tipo e definidas as operações, o usuário pode utilizar os números complexos da mesma forma como utiliza números inteiros. Isto permite a definicão em C++ de outras linguagens. Outra caracteristica do C++ muito utilizada nos programas é sua capacidade de criar novos tipos de objetos que herdem características de outros objetos (mecanismo de herança) e que podem modificar parcialmente ou totalmente seus comportamentos (mecanismo de funções virtuais). As principais extencões criadas na linguagem C++ para torná-la capaz de descrever RPs foram as criações do objeto (classe) **"Transicao"** e do objeto **"Lugar"** e dos métodos (operações) **"ti << pj"** (ti é uma Transição e pj é um Lugar) que define que existe um arco de entrada que sai de **pj** e vai para **ti** e e **"ti >> pj"** que que define que existe um arco de saída que sai de **ti** e vai para **pj**, além da função "=" que é capaz de atribuir Marcas a Lugares.

Com apenas estes tipos definidos a linguagem C++ foi expandida de forma a poder descrever RP. Um recurso muito interessante do C++, que é utilizado nas descrições de RPs, é a capacidade de descrever blocos que se repetem. Assim, RPs complexas que possuam blocos que se repetem (como ocorre em grande parte das RPs) podem ser descritas de forma bastante simples.

Abaixo segue um exemplo de modelamento de um sistema multiprocessador em RPs e sua descrição na linguagem proposta.

Ex.3.1: Sistema Multiprocessador - Dois processadores que acessam uma memória compartilhada

RP que modela o sistema:

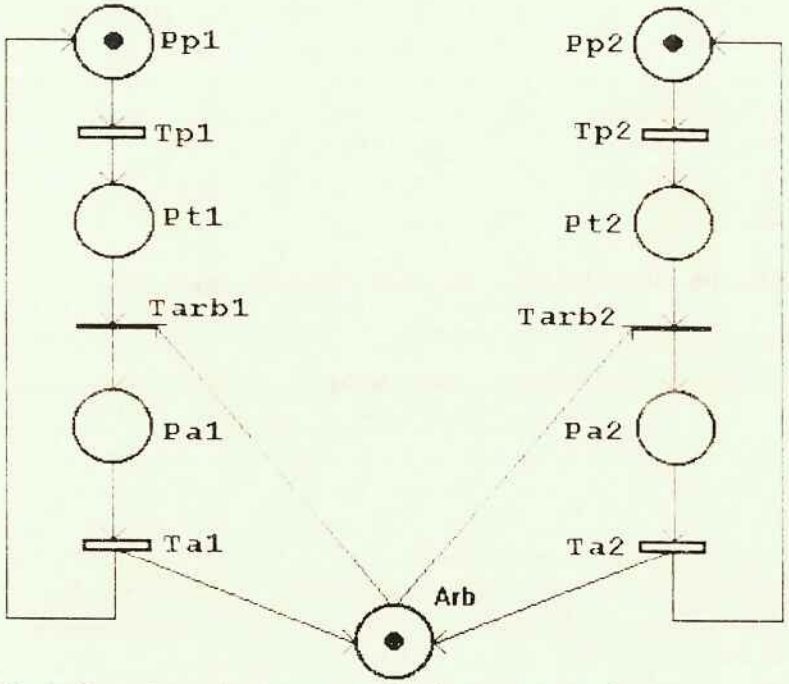

fig.3.1: Modelo do sistema multiprocessador

Significado dos lugares e transições

Lugares	Significado	Marcação Inicial
Pp1,Pp2	Processador PROCESSANDO	1
Pt1,Pt2	Processador Requisitanto uso do Barramento	0
Pa1,Pa2	Processador Acessando Recurso Global	0
Arb	Arbitro do Barramento	1

Transições	Significado	Taxas de Disparo
Tp1,Tp2	Tempo Processando	1/T_PROC
Tarb1,Tarb2	Arbitração	-
Ta1,Ta2	Tempo Recurso Global	1/T_ACESS

Descrição na linguagem proposta

```
struct Proc
{
// cria uma  estrutura que descreve um processador
  Lugar PL[3]; // PL[0] = ppi do exemplo
               // PL[1] = pti do exemplo
               // PL[2] = pai do exemplo
  Transition TR[3]; // TR[0] = tpi do exemplo
                    // TR[1] = tarbi do exemplo
                    // TR[2] = tai do exemplo
// no construtor da estrutura e que serão efetuas as
// ligações Transição <-> place
  Proc ()
  {
    PL[0] = 1;        // coloca uma Marca na rede indicando
                      //    que    o    processador    está
inicialmente
                      //  no estado processando
    for (char I = 0; I < 3; I++)
    {
// estabelece os arcos de entrada da Transição TR[I]
      TR[I] << PL[I];
// estabelece os arcos de saida da Transição TR[I]
      TR[I] >> PL[(I + 1) % 3];
    }
  }
};

struct RedeFinal
{
// Descreve o sistema com dois processadores que
// compartilham  uma mesma memória
  Lugar Arb;      // árbitro da memória
  Proc    Pr[2];       //  os  dois  processadores.  Cada
processador
                  // é uma estrutura "Proc" criada acima
  RedeFinal ()
  {
    Arb = 1         // uma Marca para indicar
                    // memória disponível
    for (char I = 0; I < 2; I++)
    {
//   arcos de entrada
      Pr[I].TR[1] << Arb;
//   arcos de saida
      Pr[I].TR[2] >> Arb;
    }
  }
};
```

A RP do exemplo está estruturalmente representada pela "struct RedeFinal". Veja que a descrição ficou muito facilitada pela possibilidade de utilizacão dos recursos do C++, com a definicão de variáveis, loops, etc, bem como pela capacidade de criacão de blocos. A linha de C++

```
RedeFinal RededePetri;
```

cria uma variável chamada "RededePetri" que é uma instância do objeto "RedeFinal". Com esta variável podemos manipular diversos campos da RP da mesma forma como manipulamos uma estrutura em C ou C++. Assim, por exemplo, se desejarmos colocar uma Marca no Lugar processando do processador 1 escreveriamos

```
RededePetri.Pr[0].Pl[0] = 1;
// RededePetri.Pr[0] é uma instância do objeto "Proc"
// RededePetri.Pr[0].Pl[0] é uma instância do objeto
// "Lugar"
```

Outra operacão que poderia ser realizada é saber quantas Marcas existem em determinado Lugar da RP. A linha de C++

```
printf ("Marcas no arbitro: %d",
          (unsigned) RededePetri.Arb);
```

imprimiria quantas Marcas existem no Lugar "Arb".

3.2. Método utilizado para extrair o comportamento das RPs.

Independentemente de se estar utilizando ***RP_NUM*** ou ***RP_SIM*** o procedimento para interfacear com estes programas é, de forma geral, o mesmo. O usuário deve:

a. Descrever a estrutura do sistema de acordo com a lingugem proposta no item anterior. Nesta descricão podem ser utilizados os tipos e objetos básicos da linguagem C++, mais os objetos descritos acima (basicamente "Lugar" e "Transicao"), mais os objetos criados pelo usuário. De fato, na maioria dos casos, o usuário irá criar uma série de novos objetos que herdem objetos básicos, completando ou alterando o comportamento destes. Isto é feito para que o usuário possa ajustar as ferramentas a classe de RP que está em análise. Assim, por exemplo, o usuário poderá criar um objeto chamado "TransicaoTemporal" que herde características do objeto "Transicao", mas que possua métodos (rotinas) capazes de manipular uma variável que represente o tempo. De fato, as implementações atuais dos

programas ***RP_NUM*** e ***RP_SIM*** já possuem incorporadas uma série de extensões deste tipo;

b. Descrever um conjunto de rotinas que indiquem qual deve ser o comportamento seguido pelos programas ***RP_NUM*** e ***RP_SIM***. Estas rotinas irão executar basicamente as seguintes funcões:

- interface com o usuário. Os programas ***RP_NUM*** e ***RP_SIM*** chamam algumas rotinas, em pontos convenientes da análise, para que o usuário possa realizar entrada e saída de dados. Nenhum tipo de entrada e saída de dados é realizada pelos programas sem ser atraves destas rotinas. Assim, o usuário pode adequar o formato dos dados a sua necessidade. Pode, por exemplo, realizar a entrada de dados atraves de arquivos em disco ou através de entrada pelo teclado. A saída de dados pode apresentar valores no video na forma de texto ou na forma de graficos, pode gravar valores em arquivo ou até inserir dados numa planilha eletrônica ou num banco de dados. Na verdade, a utilizacão destas rotinas de entrada e saída poderiam provocar atuações fisicas em determinado sistema (esta idéia é abordada rapidamente no capitulo que descreve o programa ***RP_SIM***).

- interface com os núcleos dos programas. Os programas ***RP_NUM*** e ***RP_SIM*** chamam algumas rotinas, em pontos convenientes da análise, que definem qual deve ser o comportamento que este deve seguir. Estas rotinas são utilizadas principalmente pelo programa ***RP_SIM***. Através da implementacão destas rotinas, o usuário poderá interferir no andamento do processo de análise. Apenas a título de exemplo podemos citar as seguintes rotinas:

."RotinaPosCiclo" do programa ***RP_SIM***. Esta rotina é chamada após cada ciclo de simulacão. Através dela o usuário poderá imprimir a marcação gerada, o número da Transição que disparou, ou mesmo fazer com que o programa execute passo a passo. Para este último caso, basta que na rotina seja colocada uma linha que espere a digitação de uma tecla. Após cada ciclo de simulacão o simulador ficará parado, aguardando a digitação.

.rotina "ArmazenaMarcacoes" do programa ***RP_SIM***. Esta rotina indica ao programa se é para ele armazenar, ou não, as marcações que estão sendo geradas pelo processo de simulação. O usuário poderá optar por armazenar todas as marcações desde o início até o final da simulação, armazenar as marcacões durante um período da simulação, ou até que o número de marcações tenha atingido determinado limite.

Além de definirem o comportamento dos programas com relação ao que se espera extrair do sistema em análise,

estas rotinas são de vital importância para a determinação da velocidade de obtenção de resultados e da utilização de memória. Por exemplo, o armazenamento de todas as marcações geradas por uma RP de porte razoável pode tornar o tempo de simulação e a utilização de memória muito elevados, sendo que boa parte das vezes o conjunto completo das marcações atingidas pelo sistema não é um dado relevante.

3.3. A utilização dos programas *RP_NUM* e *RP_SIM*.

Os programas ***RP_NUM*** e ***RP_SIM*** utilizam o processo de compilação da RP descrita para chegar a um programa executável. Portanto quem for utilizar um destes programas deverá seguir os seguintes passos:

1. Descrever a RP e as rotinas de interface como descrito em 2.1 e 2.2;

2. Compilar o programa com um compilador C++ comercial. Este programa é na verdade a RP descrita, mais as rotinas de interpretação associadas, mais as rotinas do núcleo do programa ***RP_NUM*** ou ***RP_SIM***;

3. Executar o programa gerado no passo 2.

Deve-se notar, que o programa gerado pode ser flexível em vários de seus pontos, possibilitando-se assim variação de parâmetros da análise sem que seja necessário nova compilação. Esta flexibilidade é atingida, como já foi explicado, pela existência das rotinas de interface. Nestas rotinas podemos ler valores do teclado (ou arquivos) com chamadas às funções padrão do C++ (por exemplo getch () ou scanf ()), permitindo que parâmetros da RP descrita (como número de Marcas em determinados Lugares, tempos associados a determinadas Transições, etc) sejam variados com facilidade, sem que seja necessário o trabalho de recompilação dos programas.

4. Análise de um Micro Sistema de Manufatura utilizando o *RP_SIM*.

As RPs e suas variantes vem sendo intensamente utilizadas em diversos pontos das células flexíveis de manufatura. A utilização de RPs pode ser tanto na parte de modelamento dos sistemas reais como no próprio processo de controle destes sistemas.

Alguns trabalhos ([VAL92], [VAL90], [SILV90]) vem sendo publicados mostrando a utilização conjunta de RPs

com inteligência artificial e RPs com "fussy-logic" para implementação de algoritmos de controle de sistemas.

Uma das principais vantagens da utilização de RPs nas células de manufatura flexíveis reside justamente no fato da utilizacão de uma ferramenta única para descrição, análise e implementacão do controle do sistema.

O programa ***RP_SIM*** é especialmente indicado para este tipo de operação. Isto devido a capacidade de implementacão, por parte do usuário, das rotinas de entrada e saida de dados permitindo que a mesma descricão do sistema seja utilizada para sua análise e para o seu controle efetivo, ou seja, dependendo do conjunto de rotinas de entrada e saída que esteja sendo usado, o ***RP_SIM*** irá realizar a tarefa de análise ou de controle do sistema.

No exemplo apresentado neste ítem vamos mostrar o trabalho de modelamento de um pequeno sistema de manufatura utilizando RPs coloridas com interpretacão

O sistema de manufatura aqui apresentado é uma linha de producão de torneiras. Cada torneira é composta por 5 peças, sendo 2 do mesmo tipo. Chamaremos os tipos das peças de Pc1, Pc2, Pç3 e Pc4. Cada uma destas peças é produzida por uma máquina diferente. Chamaremos estas máquinas por Mq1, Mq2, Mq3 e Mq4 respectivamente. As peças Pç1, Pç2 e Pç4 devem passar por uma etapa de cromeacão. A peca Pç3 não necessita de nenhuma etapa adicional antes de sua inserção no produto final. Existem duas cromeadoras na linha de produção que chamaremos por Cr1 e Cr2. As peças podem ser cromeadas indistintamente em alguma destas cromeadoras. A montagem final da torneira é realizada manualmente. Para esta montagem deve estar disponivel 1 peça Pç1, 2 peças Pc2, 1 peça Pç3 e 1 peça Pç4, ou seja:

Torneira = 1*Pç1 + 2*Pç2 + 1*Pç3 + 1*Pç4

As pecas Pç1, Pç2 e Pc4, após serem produzidas, escorrem automaticamente por uma bica e entram em uma fila para serem cromeadas. A escolha da cromeadora é feita, através de processo automático, como a que tenha menor fila naquele momento. Este processo garante um equilíbrio nas filas de peças para cromear.

Os tempos necessários para cada uma das etapas dos processo são os seguintes:

a. Máquinas Mq1, Mq2, Mq3 e Mq4: Em média 10.0 unidades de tempo para produzir uma peça. O tempo de produção obedece basicamente uma distribuição uniforme com 30% de variação em relação ao valor médio;

b. Cromeadoras Cr1 e Cr2: Em média 10.0 unidades de tempo para cromear uma peça. O tempo de cromeacão obedece basicamente uma distribuicão uniforme com 40% de variação em relação ao valor médio;

c. Montagem: A montagem final da torneira leva em média 10.0 unidades de tempo. O tempo de montagem obedece basicamente uma distribuicão uniforme com 20% de variacão em relacão ao valor médio;

O empresário, dono da fábrica, resolve realizar investimentos para melhorar o número médio de torneiras produzidas em determinado período de tempo. Inicialmente ele compra uma nova maquina Mq2. Este investimento foi baseado no fato de que na torneira final existe o dobro de peças do tipo 2. Animado com os resultados obtidos, realiza novos investimentos. Moderniza suas cromeadoras, fazendo com que estas passem a apresentar um tempo de cromeacão 30% inferior ao apresentado anteriormente. Verificando os resultados resolve fazer um ultimo investimento que é o de modernizar mais ainda as cromeadoras, fazendo com que estas passem a apresentar um tempo de cromeacão 50% inferior ao apresentado antes das modernizacões.

Nossos objetivos neste exemplo serão os seguintes:

a. Apresentar o modelamento da linha de produção utilizando RPs coloridas;

b. Verificar os investimentos realizados pelo empresário;

d. Propor um outro plano de aumento da capacidade de produção da linha.

4.1. Modelamento utilizando RPs coloridas:

A utilizacão de RPs convencionais para modelamento deste sistema [SANG93] requer que uma série de Lugares e Transicões sejam colocados nos modelos das cromeadoras e da montadora para distinguir o tipo da peça. A utilização de RPs coloridas pode simplificar consideravelmente a complexidade do modelo, já que em um único Lugar as peças podem ser distinguidas através de cores. A figura 4.1 mostra o modelo da linha de produção utilizando RPs coloridas. O modelamento merece os seguintes comentários:

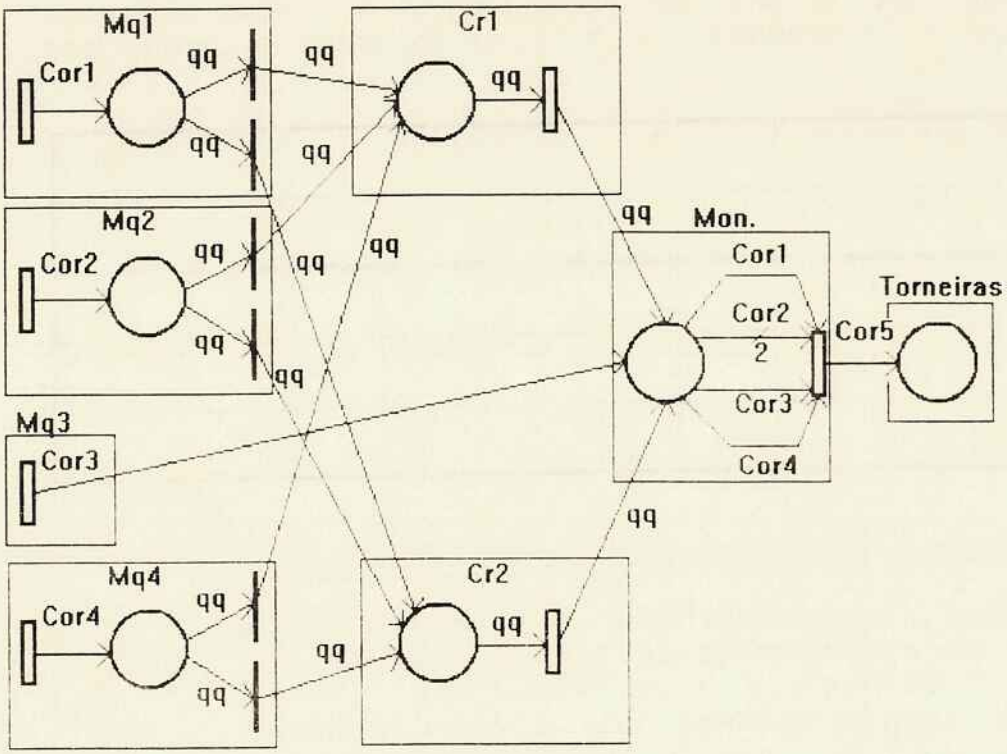

fig.4.1: Modelo em RP colorida da fábrica de torneiras.

a. Os Lugares das cromeadoras podem possuir marcas da cor 1, cor 2 e cor 4. As marcas de cada cor permanecem separadas neste Lugar único permitindo a identificacão da máquina onde as pecas foram produzidas;

b. O Lugar da montadora pode possuir marcas da cor 1, cor 2, cor 3 e cor 4. As marcas de cada tipo permanecem separadas neste Lugar único permitindo a identificação da máquina onde as peças foram produzidas;

c. Um arco de entrada com denominacão "qq" indica que no Lugar de origem pode ter Marca com qualquer cor;

d. Um arco de saída com denominação "qq" indica que a cor da Marca que vai ser colocada no Lugar de destino deve manter a cor da Marca que veio do Lugar de origem.

e. A producão de peças pelas 4 máquinas está representada pelas transições fonte presentes nos modelos das máquinas (transições sempre habilitadas).

A utilização do programa ***RP_SIM*** permitiu a obtenção da produção da fábrica antes e depois das alterações efetuadas em sua estrutura. Todos os resultados foram obtidos para 1500.0 unidades de tempo. Os resultados estão mostrados em forma tabular. Cada tabela apresenta a quantidade de cada tipo de peça e o total de peças existente em cada fila (uma fila para cada Cromeadora e uma fila para a Montadora), bem como o total de Torneiras

produzidas. Na coluna das Cromeadoras a linha superior representa a Cromeadora 1 e a linha inferior a Cromeadora 2.

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pc1	Pc2	Pc4	Tot	Pc1	Pc2	Pc3	Pc4	
70	29	26	15						
				238	63	0	108	67	50
72	17	33	22						

4.1.1.Investimentos realizados pelo empresario:

Abaixo iremos alterar o modelo definido para refletir as mudancas ocorridas na producão de torneiras com os investimentos promovidos pelo empresário. Cada alteracão sera comparada com o valor anterior e com o valor original da linha de producão.

a. Acrescimo de mais uma maquina Mq2. Para fazer o modelo refletir esta alteração, apenas dividimos por 2 o valor do tempo da Transição que representa a máquina Mq2. Com isso ela possou a disparar, em média, 2 vezes mais, produzindo assim o dobro do número de pecas. Os resultados obtidos foram os seguintes:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pç1	Pc2	Pç4	Tot	Pç1	Pc2	Pç3	Pç4	
146	29	87	30						
				201	56	2	99	44	60
146	16	104	26						

Aumento na producão em relação ao original: 20%

Note que o aumento na producão de 20% pode ter parecido significativo para o empresário, justificando seus novos investimentos;

b. Reformulacão das cromeadoras para diminuir o tempo médio de cromeação em 30%. Para fazer o modelo refletir esta alteracão apenas dividimos por 1.3 o valor do tempo das Transições que representam as cromeadoras. Os resultados obtidos foram os seguintes:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pç1	Pç2	Pç4	Tot	Pç1	Pç2	Pç3	Pç4	
111	7	95	9						
				175	55	2	78	41	**81**
110	20	60	30						

Aumento na producão em relação ao anterior: 35%
Aumento na producão em relação ao original: 62%

Note que o aumento na produção de 35% em relação ao valor anterior pode ter deixado o empresário ainda mais motivado para novos investimentos. O empresário ainda deve ter raciocinado que as cromeadoras são o gargalo do sistema e portanto valeria a pena continuar otimizando-as;

c. Reformulação das cromeadoras para diminuir o tempo médio de cromeação em 50% em relação ao seu valor original. Para fazer o modelo refletir esta alteração, apenas dividimos por 1.5 o valor do tempo das Transições que representam as cromeadoras. Os resultados obtidos foram os seguintes:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pc1	Pc2	Pc4	Tot	Pc1	Pc2	Pc3	Pc4	
75	1	67	7						
				189	62	0	71	56	**92**
76	6	59	11						

Aumento na produção em relação ao anterior: 13.5%
Aumento na produção em relação ao original: 84%

O aumento na produção de 13.5% em relação ao valor anterior pode ter esfriado o ânimo de investir do empresário. Este pode inclusive ter concluido que, com os recursos hora apresentados, a linha de produção havia atingido um valor próximo ao máximo.

4.1.2.Proposta alternativa de investimentos:

A proposta alternativa de investimentos está, obviamente, embasada nos resultados gerados pela resolução dos modelos. Observando a resolução do modelo

(tanto os valores finais como a evolucão da simulacão) podemos apresentar os seguintes fatos:

a. Na montadora ocorrem diversos instantes em que não existem pecas do tipo 2 disponíveis (fato observado principalmente com a evolucão da simulacão);

b. Em geral, nos instantes em que está faltando pecas do tipo 2 na montadora, a cromeadora possui este tipo de peca em sua fila de entrada. No entanto ela está ocupada cromeando outro tipo de peca (fato observado principalmente com a evolucão da simulacão).

Destes fatos faremos a seguinte proposta inicial. Vamos acrescentar um algoritmo para a escolha da peça a ser cromeada, levando em conta as necessidades da montadora. O algoritmo é bastante simples. No instante da escolha de uma peca a ser cromeada, se na montadora não estiver disponivel pelo menos duas pecas do tipo 2, então a peca escolhida para cromeacão será uma deste tipo. Este algoritmo foi acrescentado ao modelo da linha (através de interpretacões convenientes) e os resultado obtidos foram os seguintes:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pç1	Pc2	Pc4	Tot	Pç1	Pç2	Pç3	Pç4	
72	43	0	29						
				97	1	1	81	14	79
73	36	1	36						

Aumento na producão em relacão ao original: 61%

A proposta seguinte vem do fato de observarmos que na fila de cromeação, em alguns instantes, faltam peças do tipo 2. Portanto, a compra de uma nova máquina do tipo 2 pode se justificar. Os resulatdos obtidos foram os seguintes:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pç1	Pç2	Pç4	Tot	Pç1	Pç2	Pç3	Pç4	
148	43	68	37						
				82	1	2	78	1	83
148	32	75	41						

Aumento na produção em relação ao anterior: 5%
Aumento na produção em relação ao original: 66%

Notamos agora que a fila nas cromeadoras, basicamente dobrou. Portanto as cromeadoras devem estar representando pontos de estrangulamento na linha. Faremos análises para quedas no tempo de cromeação de 30%, 50% e 70%. Os resultados obtidos foram os seguintes:

30% de diminuição:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pc1	Pc2	Pc4	Tot	Pc1	Pc2	Pc3	Pc4	
108	26	60	22						
				61	2	2	56	1	105
106	26	43	37						

Aumento na producão em relacão ao anterior: 26.5%
Aumento na producão em relação ao original: 110%

50% de diminuição:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pc1	Pc2	Pc4	Tot	Pc1	Pc2	Pc3	Pc4	
66	20	34	12						
				44	2	2	39	1	122
66	18	25	23						

Aumento na producão em relação ao anterior: 16.2%
Aumento na producão em relação ao original: 144%

70% de diminuição:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pc1	Pc2	Pc4	Tot	Pc1	Pc2	Pc3	Pc4	
42	5	25	12						
				28	2	2	23	1	136
42	19	11	12						

Aumento na producão em relação ao anterior: 11.5%
Aumento na producão em relação ao original: 172%

As propostas de alteração poderiam continuar, sempre procurando-se diminuir as filas das cromeadoras e da montadora e identificando pontos de estrangulamento.

5. CONCLUSÕES.

Das análises efetuadas acima podemos, entre muitas outras coisas, concluir que:

a. A simples observação de um processo industrial pode muitas vezes ocultar determidados gargalos, fazendo com que haja o sub-aproveitamento de recursos já existentes e que investimentos sejam direcionados para pontos errados do processo;

b. O processo de modelamento do sistema pode, por si só, apresentar de forma clara diversos destes gargalos;

c. Uma ferramenta eficiente para modelar e obter resultados deste tipo de processo pode ser de elevada valia. Diversas alterações a planta podem ser propostas, implementadas e testadas no modelo, com investimento muito baixo. So quando o conjunto de alterações estiver devidamente definido é que ele será implementado na planta, com resultados praticamente assegurados. Veja que no exemplo conseguimos obter um aumento de produção da ordem de 17% contra 8% obtido pelo empresario. Apenas com a primeira alteração, que possui um custo de implementação muito baixo, conseguimos uma elevação de produção próxima a conseguida com os dois primeiros investimentos do empresário;

d. Nem sempre o aumento ou a otimização de um recurso isolado produz um resultado positivo no número de itens produzidos. Para ratificar esta conclusão fizemos mais uma alteração no modelo original. Esta alteração prevê a inclusão de um algoritmo que inativa a produção das máquinas se já existirem peças suficientes do tipo correspondente na montadora. Os resultados obtidos foram os seguintes:

Fila Cromeadoras				Fila Montadora					Torneiras
Tot	Pc1	Pc2	Pc4	Tot	Pc1	Pc2	Pc3	Pc4	
6	2	1	3						
				29	9	5	14	1	75
7	1	1	5						

Aumento na produção em relação ao original: 50%

Note que conseguimos o aumento de 50% no número de torneiras produzidas com a DIMINUIÇÃO do número de peças produzidas pelas máquinas. Este resultado não pode ser classificado como intuitivo;

e. RPs coloridas simplificam o modelamento deste tipo de sistema;

Além das conclusões relativas ao exemplo acima, podemos ainda apresentar os seguintes comentários em relacão a utilizacão do programa ***RP_SIM***:

a. A utilizacão de uma linguagem tipo C++ como base para criacão de outras linguagens possui diversas vantagens, como:

1. incorporacão imediata à nova linguagem de ferramentas, bibliotecas de funcões e ambientes amigáveis e integrados presentes em compiladores C++ comerciais;

2. maior facilidade e rapidez para criacão do compilador da linguagem proposta;

3. permitir que a nova linguagem possa ser extendida/modificada de acordo com as necessidades de cada usuario. Esta caracteristica é herdada direntamente do C++.

b. Ao lado das vantagens apresentadas no item acima a utilizacão de uma linguagem tipo C++ como base para criacão de outras linguagens possui uma desvantagem que é a sintaxe hostil das linguagens de programacão para pessoas que não sejam desta área. Esta dificuladade pode ser contornada com a criação de um módulo adicional que implemente uma interface iterativa (possivelmente com recursos gráficos) para criacão das descricões das redes e das interpretacões. Este módulo converteria cada descricão para a linguagem proposta.

BIBLIOGRAFIA

[SANG93]. C. A. Sangiorgio: "Uma Ferramenta para Descrição e Análise de Paralelismo e Sincronização" - Dissertacão de Mestrado - EPUSP - 1993;

[VALL92]. Prof. Dr. Robert Vallete: "Redes de Petri em Processos de Manufatura" - L.A.A.S./C.N.R.S. Toulouse - Curso de Pós Graduacão EPUSP - 1992;

[VALL90]. Prof. Dr. Robert Vallete: "Redes de Petri" - L.A.A.S./C.N.R.S. Toulouse - Curso de Pós Graduacão EPUSP - 1990;

[SILV90]. M.Silva, R.Vallete.: "Petri Nets and Flexible Manufacturing, in Advances in Petri Nets" - Lecture Notes in Computer Science - Springer Verlag - 1990;

[DEWH90]. S. C. Dewhurst, K. T. Stark: "Programming in C++" - Prentice-Hall - 1990.

BOLETINS TÉCNICOS - TEXTOS PUBLICADOS

BT/PCS/9301 - Interligação de Processadores através de Chaves Ômicron - GERALDO LINO DE CAMPOS, DEMI GETSCHKO

BT/PCS/9302 - Implementação de Transparência em Sistema Distribuído - LUÍSA YUMIKO AKAO, JOÃO JOSÉ NETO

BT/PCS/9303 - Desenvolvimento de Sistemas Especificados em SDL - SIDNEI H. TANO, SELMA S. S. MELNIKOFF

BT/PCS/9304 - Um Modelo Formal para Sistemas Digitais à Nível de Transferência de Registradores - JOSÉ EDUARDO MOREIRA, WILSON VICENTE RUGGIERO

BT/PCS/9305 - Uma Ferramenta para o Desenvolvimento de Protótipos de Programas Concorrentes - JORGE KINOSHITA, JOÃO JOSÉ NETO

BT/PCS/9306 - Uma Ferramenta de Monitoracão para um Núcleo de Resolução Distribuída de Problemas Orientado a Objetos - JAIME SIMÃO SICHMAN, ELERI CARDOSO

BT/PCS/9307 - Uma Análise das Técnicas Reversíveis de Compressão de Dados - MÁRIO CESAR GOMES SEGURA, EDIT GRASSIANI LINO DE CAMPOS

BT/PCS/9308 - Proposta de Rede Digital de Sistemas Integrados para Navio - CESAR DE ALVARENGA JACOBY, MOACYR MARTUCCI JR.

BT/PCS/9309 - Sistemas UNIX para Tempo Real - PAULO CESAR CORIGLIANO, JOÃO JOSÉ NETO

BT/PCS/9310 - Projeto de uma Unidade de Matching Store baseada em Memória Paginada para uma Máquina Fluxo de Dados Distribuído - EDUARDO MARQUES, CLAUDIO KIRNER

BT/PCS/9401 - Implementacão de Arquiteturas Abertas: Uma Aplicacão na Automacão da Manufatura - JORGE LUIS RISCO BECERRA, MOACYR MARTUCCI JR.

BT/PCS/9402 - Modelamento Geométrico usando do Operadores Topológicos de Euler - GERALDO MACIEL DA FONSECA, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/9403 - Segmentação de Imagens aplicada a Reconhecimento Automático de Alvos - LEONCIO CLARO DE BARROS NETO, ANTONIO MARCOS DE AGUIRRA MASSOLA

BT/PCS/9404 - Metodologia e Ambiente para Reutilização de Software Baseado em Composição - LEONARDO PUJATTI, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/9405 - Desenvolvimento de uma Solucão para a Supervisão e Integração de Células de Manufatura Discreta - JOSÉ BENEDITO DE ALMEIDA, JOSÉ SIDNEI COLOMBO MARTINI

BT/PCS/9406 - Método de Teste de Sincronização para Programas em ADA - EDUARDO T. MATSUDA, SELMA SHIN SHIMIZU MELNIKOFF

BT/PCS/9407 - Um Compilador Paralelizante com Detecção de Paralelismo na Linguagem Intermediária - HSUEH TSUNG HSIANG, LÍRIA MATSUMOTO SAITO