
Uma Regra de Sincronização Baseada na
Lógica Linear para Prevenção de *Deadlock*
em *WorkFlow nets* Interorganizacionais

Vinícius Ferreira de Oliveira



UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Vinícius Ferreira de Oliveira

Uma Regra de Sincronização Baseada na
Lógica Linear para Prevenção de *Deadlock*
em *WorkFlow nets* Interorganizacionais

Dissertação de mestrado apresentada ao
Programa de Pós-graduação da Faculdade
de Computação da Universidade Federal de
Uberlândia como parte dos requisitos para a
obtenção do título de Mestre em Ciência da
Computação.

Área de concentração: Engenharia de Software

Orientador: Dr. Stéphane Julia

Coorientadora: Dra. Lúcia Maria Soares Passos

Uberlândia

2017

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU, MG, Brasil.

- O48r
2017
- Oliveira, Vinícius Ferreira de, 1989-
Uma regra de sincronização baseada na lógica linear para prevenção de Deadlock em WorkFlow nets interorganizacionais / Vinícius Ferreira de Oliveira. - 2017.
155 f. : il.
- Orientador: Stéphane Julia.
Coorientadora: Lígia Maria Soares Passos.
Dissertação (mestrado) - Universidade Federal de Uberlândia, Programa de Pós-Graduação em Ciência da Computação.
Disponível em: <http://dx.doi.org/10.14393/ufu.di.2018.1114>
Inclui bibliografia.
1. Computação - Teses. 2. Redes de Petri - Teses. 3. Fluxo de trabalho - Teses. I. Julia, Stéphane. II. Passos, Lígia Maria Soares, 1984- III. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

CDU: 681.3

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Os abaixo assinados, por meio deste, certificam que leram e recomendam para a Faculdade de Computação a aceitação da Dissertação intitulada "**Uma Regra de Sincronização Baseada na Lógica Linear para Prevenção de *Deadlock* em *WorkFlow nets* Interorganizacionais**" defendida por **Vinícius Ferreira de Oliveira** como parte dos requisitos exigidos para a obtenção do título de **Mestre em Ciência da Computação**.

Uberlândia/MG, 30 de agosto de 2017.

Orientador: _____

Prof. Dr. Stéphane Julia
Universidade Federal de Uberlândia

Coorientadora: _____

Profa. Dra. Lígia Maria Soares Passos
Universidade Federal Rural do Rio de Janeiro

Banca Examinadora:

Prof. Dr. Carlos Roberto Lopes
Universidade Federal de Uberlândia

Profa. Dra. Emília Villani
Instituto Tecnológico de Aeronáutica

*Dedico este trabalho aos meus pais Vaine e Márcia, ao meu irmão Raphael e aos meus
sobrinhos Daniel e Ana Beatriz.*

Agradecimentos

Este trabalho não teria sido possível sem a colaboração e a boa vontade de todos aqueles à que agora me refiro.

Primeiramente à Deus, pois sem Ele acredito que nada é possível.

Aos meus pais Vaine de Oliveira e Helena Márcia Ferreira, que sempre me apoiaram nos momentos mais difíceis, dando-me a atenção e tendo a paciência necessária.

Ao meu irmão Raphael Ferreira de Oliveira, à minha cunhada Ana Carolina Rodrigues Henrique e aos meus sobrinhos Daniel Henrique de Oliveira e Ana Beatriz Henrique de Oliveira pelo apoio durante todo o tempo.

À minha namorada Mariana Nunes que faz os meus dias melhores.

Agradeço também aos meus colegas de curso que foram fundamentais nesse período, Amanda Gaspar, Anderson Leonardo, Franciny Medeiros, Herbert Rezende, Leiliane Rezende, Luciane Silva e Regis Maicon, compartilhando sempre que necessário seu tempo e conhecimento.

Em especial ao professor Dr. Stéphane Julia e à professora Dra. Lígia Maria Soares Passos, por terem acreditado em meu potencial, pela paciência que tiveram e por todo o apoio e conhecimento necessário que foram compartilhados.

Não poderia deixar de agradecer também à todo o corpo docente do Programa de Pós-Graduação em Ciência da Computação que foi de suma importância em minha formação.

À CAPES e a FAPEMIG, pelo apoio financeiro.

À todos, o meu muito obrigado!

“Não basta conquistar a sabedoria, é preciso usá-la.”
(Cícero)

Resumo

Este trabalho apresenta um método de prevenção para situações de *deadlock*, por meio de uma regra de sincronização, em processos de *workflow* interorganizacionais que são localmente *sound*, porém não globalmente *sound*. Tal método considera a teoria das redes de Petri e é baseado na análise das árvores de prova da Lógica Linear. O trabalho apresenta também um método para análise qualitativa e quantitativa do modelo de processo de *workflow* interorganizacional antes e após a aplicação da regra de sincronização usada para remover situações de *deadlock*. Após a aplicação da regra de sincronização, modelos que representam processos de *workflow* interorganizacionais que não são *sound* se transformam em modelos *sound* devido à substituição de certos lugares de comunicação assíncrona por mecanismos de comunicação parcialmente síncrona. Como consequência da aplicação da regra de sincronização, o processo poderá sofrer pequenos atrasos, diminuindo o desempenho do processo. A utilização da Lógica Linear tem como vantagem a possibilidade de reutilizar fragmentos da árvore de prova produzidos antes da aplicação da regra de sincronização. Consequentemente a análise do modelo somente é realizada na parte do modelo responsável pela situação de *deadlock*.

Palavras-chave: Processos de negócio, redes de Petri, Lógica Linear, *WorkFlow net* Interorganizacional, *Soundness*, *Deadlock*, Regra de Sincronização.

Abstract

This work presents a prevention method for deadlock situations, through a synchronization rule, in interorganizational workflow processes that are locally sound, but not globally sound. Such a method considers the Petri net theory and is based on the analysis of Linear Logic proof trees. The work also presents a method for the qualitative and quantitative analysis of the interorganizational workflow model process, before and after the application of the synchronization rule used to remove deadlock situations. Following the application of the synchronization rule, those models that represent the interorganizational workflow processes that are not sound are transformed into sound models due to the replacement of certain asynchronous communication places by partially synchronous communication mechanisms. As a consequence of the application of the synchronization rule, the process may suffer minor delays, reducing the process performance. The use of Linear Logic has the greatest advantage the possibility of reusing fragments of the proof tree produced before the application of the synchronization rule. Consequently the analysis of the model is only performed in that part of the model responsible for the deadlock situation.

Keywords: Business process, Petri net, Linear Logic, Interorganizational WorkFlow Net, Soundness, Deadlock, Synchronization Rule.

Lista de ilustrações

Figura 1 – Rede de Petri e seus elementos básicos.	34
Figura 2 – Exemplos de sensibilização e disparo de transição em uma rede de Petri.	36
Figura 3 – Exemplo de rede de Petri t-temporal.	38
Figura 4 – Exemplo de rede de Petri colorida.	39
Figura 5 – Exemplo de WF-net.	40
Figura 6 – Exemplo de IOWF-net.	43
Figura 7 – $U(\text{IOWF-net})$ correspondente à IOWF-net da Figura 6.	44
Figura 8 – Versão 4.0.1 do CPN Tools.	46
Figura 9 – Redes de Petri para exemplificação da tradução de redes de Petri em fórmulas da Lógica Linear.	47
Figura 10 – Exemplo de rede de Petri com sifão.	54
Figura 11 – Antes e depois da regra de sincronização.	61
Figura 12 – Antes e depois da regra de sincronização com lugar de comunicação com mais de um arco de entrada.	62
Figura 13 – Antes e depois da regra de sincronização com lugar de comunicação com mais de um arco de saída.	63
Figura 14 – Adição do lugar auxiliar <i>aux</i> no modelo da Figura 11B.	65
Figura 15 – Aplicando a regra de sincronização na $U(\text{IOWF-net})$ apresentada na Figura 7.	71
Figura 16 – Antes e depois da regra de sincronização na $U(\text{IOWF-net})$ apresentada na Figura 7.	72
Figura 17 – Modelo final após a aplicação da regra de sincronização na $U(\text{IOWF-net})$ da Figura 7.	74
Figura 18 – Implementação no CPN Tools do modelo apresentado na Figura 7.	77
Figura 19 – Grafo das marcações acessíveis para a rede de Petri da Figura 7 antes da regra de sincronização, gerado pelo simulador CPN Tools.	78
Figura 20 – Relatório de Vivacidade para a rede de Petri da Figura 7 antes da regra de sincronização, gerado pelo simulador CPN Tools.	79

Figura 21 – Implementação no CPN Tools do modelo apresentado na Figura 17. . .	80
Figura 22 – Grafo das marcações acessíveis para a rede de Petri da Figura 17 depois da regra de sincronização, gerado pelo simulador CPN Tools.	81
Figura 23 – Relatório de Vivacidade para a rede de Petri da Figura 17 depois da regra de sincronização, gerado pelo simulador CPN Tools.	81
Figura 24 – Datas simbólicas antes e depois da regra de sincronização.	83
Figura 25 – Rede de Petri t-temporizada correspondente à $U(IOWF-net)$ da Figura 7.	88
Figura 26 – Implementação no CPN Tools do modelo apresentado na Figura 25. . .	90
Figura 27 – Implementação no CPN Tools do cenário Sc_4 antes da regra de sincronização.	91
Figura 28 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_4 antes da regra de sincronização.	92
Figura 29 – Implementação no CPN Tools do modelo t-temporizado correspondente a $U(IOWF-net)$ da Figura 17.	93
Figura 30 – Implementação no CPN Tools do cenário Sc_4 depois da regra de sincronização.	94
Figura 31 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_4 depois da regra de sincronização.	95
Figura 32 – Processo para compra virtual de bilhetes.	99
Figura 33 – Módulos de <i>workflow</i> Cliente (C1) e <i>Ticket</i> (T1).	99
Figura 34 – O sistema composto $C1 \oplus T1$	101
Figura 35 – Aplicando a regra de sincronização para o cenário Sc_3 do sistema composto apresentado na Figura 34.	105
Figura 36 – Antes e depois da regra de sincronização no cenário Sc_3	106
Figura 37 – Aplicando a regra de sincronização para o cenário Sc_4 do sistema composto apresentado na Figura 34.	107
Figura 38 – Antes e depois da regra de sincronização no cenário Sc_4	108
Figura 39 – Modelo final após a aplicação da regra de sincronização no sistema composto da Figura 34.	109
Figura 40 – Implementação no CPN Tools do modelo apresentado na Figura 34. . .	111
Figura 41 – Grafo das marcações acessíveis para a rede de Petri da Figura 34 antes da regra de sincronização, gerado pelo simulador CPN Tools.	112
Figura 42 – Relatório de Vivacidade para a rede de Petri da Figura 34 antes da regra de sincronização, gerado pelo simulador CPN Tools.	112
Figura 43 – Implementação no CPN Tools do modelo apresentado na Figura 39. . .	113
Figura 44 – Grafo das marcações acessíveis para a rede de Petri da Figura 39 depois da regra de sincronização, gerado pelo simulador CPN Tools.	113

Figura 45 – Relatório de Vivacidade para a rede de Petri da Figura 39 depois da regra de sincronização, gerado pelo simulador CPN Tools.	114
Figura 46 – Rede de Petri t-temporizada correspondente ao sistema composto da Figura 34.	118
Figura 47 – Implementação no CPN Tools do modelo apresentado na Figura 46. . .	119
Figura 48 – Implementação no CPN Tools do cenário Sc_1 antes da regra de sincronização.	120
Figura 49 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_1 antes da regra de sincronização.	120
Figura 50 – Implementação no CPN Tools do cenário Sc_2 antes da regra de sincronização.	121
Figura 51 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_2 antes da regra de sincronização.	122
Figura 52 – Implementação no CPN Tools do modelo t-temporizado correspondente a rede de Petri apresentada na Figura 39.	123
Figura 53 – Implementação no CPN Tools do cenário Sc_1 depois da regra de sincronização.	124
Figura 54 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_1 depois da regra de sincronização.	124
Figura 55 – Implementação no CPN Tools do cenário Sc_2 depois da regra de sincronização.	125
Figura 56 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_2 depois da regra de sincronização.	126

Lista de tabelas

Tabela 1 – Datas simbólicas de produção e consumo dos átomos da rede de Petri t-temporal da Figura 3.	52
Tabela 2 – Intervalos de datas simbólicas de produção e consumo dos átomos da rede de Petri t-temporal da Figura 3.	52
Tabela 3 – Intervalos de datas numéricas de produção e consumo dos átomos da rede de Petri t-temporal da Figura 3.	53
Tabela 4 – Datas simbólicas de produção e consumo dos átomos.	85
Tabela 5 – Data de produção do átomo o no cenário Sc_4	86
Tabela 6 – Intervalos de datas simbólicas, $[D_{Pmin}, D_{Cmax}]$, para a finalização do cenário Sc_4	87
Tabela 7 – Intervalos de datas numéricas para a finalização do cenário Sc_4	89
Tabela 8 – Data de produção do átomo o para o cenário Sc_1	115
Tabela 9 – Intervalo de datas simbólicas do cenário Sc_1	116
Tabela 10 – Data de produção do átomo o para o cenário Sc_2	116
Tabela 11 – Intervalo de datas simbólicas do cenário Sc_2	117
Tabela 12 – Intervalo de datas numéricas do cenário Sc_1	117
Tabela 13 – Intervalo de datas numéricas do cenário Sc_2	118
Tabela 14 – Intervalos de datas simbólicas de execução para as tarefas do cenário Sc_4	150
Tabela 15 – Intervalos de datas simbólicas de execução para as tarefas do cenário Sc_1	155
Tabela 16 – Intervalos de datas simbólicas de execução para as tarefas do cenário Sc_2	155

Lista de siglas

BPMN *Business Process Model and Notation*

IOWF-net *Interorganizational WorkFlow net*

LWF-nets *Local WorkFlow nets*

RdP *redes de Petri*

SOA *Service-Oriented Architecture*

UML *Unified Modeling Language*

U(IOWF-net) *Unfolded Interorganizational WorkFlow net*

WfMS *WorkFlow Management Systems*

WF-net *WorkFlow net*

WS *Web Services*

Sumário

1	INTRODUÇÃO	25
1.1	Motivação	28
1.2	Objetivos e Desafios da Pesquisa	29
1.3	Hipóteses	30
1.4	Contribuições	31
1.5	Organização da Dissertação	31
2	FUNDAMENTAÇÃO TEÓRICA	33
2.1	Redes de Petri	33
2.1.1	Propriedades das redes de Petri	36
2.1.2	Rede de Petri Ordinária	37
2.1.3	Rede de Petri Temporal e rede de Petri Temporizada	37
2.1.4	Rede de Petri Colorida	38
2.2	<i>Workflow net</i>	39
2.2.1	Processos	39
2.2.2	Roteamentos	40
2.2.3	<i>Soundness</i>	41
2.3	<i>Workflow net</i> Interorganizacional	41
2.4	CPN Tools	45
2.5	Redes de Petri e Lógica Linear	46
2.6	<i>Deadlock</i> em redes de Petri	53
3	TRABALHOS RELACIONADOS	57
4	REGRA DE SINCRONIZAÇÃO PARA PREVENÇÃO DE <i>DEADLOCK</i>	59
4.1	Aplicação da Regra de Sincronização	59
4.2	Análise Qualitativa	75

4.3	Análise Quantitativa	82
5	ESTUDO DE CASO	97
5.1	Módulo de <i>Workflow</i> e Sistema Composto	98
5.2	Regra de sincronização para prevenção de <i>deadlock</i> em Composição de <i>Web Services</i>	101
5.3	Análise Qualitativa de Composição de <i>Web Services</i>	110
5.4	Análise Quantitativa de Composição de <i>Web Services</i>	114
6	CONCLUSÃO	127
6.1	Principais Contribuições	127
6.2	Trabalhos Futuros	129
6.3	Contribuições em Produção Bibliográfica	129
	REFERÊNCIAS	131

APÊNDICES 137

APÊNDICE A	– ÁRVORES DE PROVA DA LÓGICA LINEAR	139
A.1	Árvore de Prova da Lógica Linear	140
A.2	Árvores de Prova para o exemplo apresentado na Proposta do Capítulo 4	141
A.3	Árvores de Prova Canônica para o exemplo apresentado na Proposta do Capítulo 4	147
A.4	Árvores de Prova para o exemplo apresentado no Estudo de Caso do Capítulo 5	151
A.5	Árvores de Prova Canônica para o exemplo apresentado no Estudo de Caso do Capítulo 5	153

Introdução

Os Sistemas de Gerenciamento de *WorkFlow*, do inglês *WorkFlow Management Systems* (WfMS), possuem como principal objetivo, executar processos de *workflow* (AALST; HEE, 2004). Os processos de *workflow* representam as sequências de atividades que devem ser executadas em uma organização para tratar casos específicos e alcançar uma meta bem definida (AALST; HEE, 2004). O termo “*workflow*” é usado em Aalst e Hee (2004) como um sinônimo para “processos de negócio”.

Os WfMS são a chave tecnológica para melhoria da efetividade e eficiência dos *workflows* dentro de uma organização (AALST, 1998). Eles focam nos processos que são limitados pelas fronteiras de uma organização. Entretanto, o número de *workflows* onde múltiplas organizações estão envolvidas está crescendo rapidamente, ou seja, a maioria dos *workflows* atravessam os limites organizacionais (AALST, 1998).

Os *workflows* de uma organização podem interagir com *workflows* de clientes, fornecedores e subcontratantes; além disso, tecnologias tais como a Troca Eletrônica de Dados, a Internet e a *World Wide Web* permitem que múltiplas organizações participem em *workflows* compartilhados (AALST, 1998). O crescimento do comércio eletrônico, das organizações virtuais, e a extensão das empresas e soluções, como a utilização de *Web Services* (WS), destacam o fato de mais e mais *workflows* estarem cruzando os limites organizacionais (AALST, 1998). Isto significa que o gerenciamento de *workflow* deve ser capaz de lidar com processos de *workflow* que abrangem várias organizações (AALST, 1998).

Processos de *workflow* que envolvem vários processos de *workflow* locais pertencentes à diferentes organizações e que precisam coordenar suas ações a fim de alcançar um objetivo em comum são conhecidos como processos de *workflow* interorganizacionais (CAPTARENCU, 2012). Existem vários mecanismos para implementar processos de *workflow* interorganizacionais (AALST, 1997). O presente trabalho considera os processos de *workflow* fracamente acoplados: processos que interagem através de mecanismos de comunicação. Cada parceiro de negócio tem um processo de *workflow* privado que é conectado à processos de *workflow* de outros parceiros (AALST, 1997). Dois mecanismos

de comunicação são então usados para interação: assíncrono e síncrono (AALST, 1997).

Uma situação de *deadlock* em um *workflow* se dá caso a execução de um processo seja interrompida e não possa ser continuada. Existem vários motivos pelos quais podem ocorrer situações de *deadlock* em um *workflow*. Os mais comuns são motivos estruturais que impedem um processo de ser concluído (MARUTA et al., 1998). Em particular, processos fracamente acoplados operam essencialmente de forma independente, porém precisam ser sincronizados em certos pontos para garantir a correta execução do processo de *workflow* global (AALST, 1997). A sincronização de processos paralelos é conhecida como uma potencial fonte de erros, incluindo *deadlocks* (GALVIN; GAGNE; SILBERSCHATZ, 2013). Dessa forma, o risco de situações de *deadlock* aumenta proporcionalmente em relação ao aumento de subprocessos que comunicam e do conhecimento parcial das estruturas dos processos que colaboram (SILVA, 2014). Assim, *workflows* interorganizacionais estão mais suscetíveis a situações de *deadlock*.

Como processos são elementos essenciais no gerenciamento de *workflow*, é importante usar um padrão estabelecido para modelagem e análise dos processos de *workflow* (AALST, 1998). Neste trabalho são utilizadas as redes de Petri (RdP), consideradas uma ferramenta matemática, gráfica e formal que permitem a modelagem, análise e controle de sistemas a eventos discretos que comportam atividades paralelas, concorrentes e assíncronas (MURATA, 1989). Os motivos para a utilização das redes de Petri para modelagem de *workflow* é o fato destas redes possuírem semântica formal, natureza gráfica, expressividade, análises técnicas e ferramentas que fornecem um *framework* para modelagem e análise dos processos de *workflow* (AALST, 1998).

De forma complementar, Aalst (1998) apresenta a definição formal das *WorkFlow nets* (WF-*nets*), que são redes de Petri que modelam processos de *workflow* simples e, na sequência, Aalst (1998) define as *WorkFlow nets* Interorganizacionais (IOWF-*nets*), que são redes de Petri que modelam processos de *workflow* interorganizacionais.

Basicamente três tipos de análise de processos de *workflow* são definidos por Aalst (1998) sendo: validação, verificação e análise de desempenho. A validação tem como objetivo testar se o processo de *workflow* se comporta como esperado, enquanto a verificação preocupa-se com a correção de um processo de *workflow* e a análise de desempenho é responsável por avaliar a capacidade de atender aos requisitos em relação aos tempos de produção, níveis de serviço e utilização de recursos de um processo de *workflow* (AALST, 1998). Assim, no contexto de verificação das *WorkFlow nets*, é definida a propriedade *Soundness* (AALST, 1998). Tal propriedade é considerada como o principal critério de correção para as *WorkFlow nets*, pois se o *workflow* é *sound*, haverá a garantia de que: se uma instância de processo começou a ser tratada, o tratamento desta será finalizado; quando este tratamento for finalizado, não poderá haver nenhuma pendência não tratada no processo para tal instância e finalmente, não haverá tarefas que não serão executadas para nenhuma instância do processo de *workflow*, ou seja, haverá ausência de tarefas

mortas no processo verificado (AALST, 1998). Essas condições da propriedade *Soundness* garantem a ausência de *deadlock* no modelo de *workflow*.

No contexto dos processos de *workflow* interorganizacionais, a correção do critério *Soundness* nos processos de *workflow* locais não garante a correção *Soundness* no modelo do *workflow* interorganizacional, como é mostrado em Aalst (1998). Quando um processo de *workflow* interorganizacional é localmente *sound*, porém não globalmente, um *deadlock* pode ser produzido pela troca de mensagens como mostrado em Xiong, Zhou e Pu (2009). Uma maneira possível para prevenir as situações de *deadlock* causadas pela troca de mensagens é sincronizar partes dos processos locais (AALST, 1998).

Muitos trabalhos já consideraram a prevenção de situações de *deadlock* em processos de *workflow* interorganizacionais utilizando as redes de Petri. Em Silva et al. (2013), um lugar de controle é adicionado à uma IOWF-net para evitar situações de *deadlock*. Entretanto, este novo lugar pode introduzir um novo *deadlock* ao modelo e aumentar gradualmente a complexidade do modelo original. Em Rezende e Julia (2015a), uma abordagem baseada nas estruturas de sifão e redes de Petri possibilísticas é proposta para tratar situações de *deadlock* em processos de *workflow*. Devido ao grande número de mecanismos de propagação que precisam ser executados a fim de evitar a ocorrência de situações de *deadlock*, tal abordagem pode ser difícil de implementar em tempo real. Dessa forma, apesar de trazer bons resultados na prevenção de *deadlocks*, essas soluções podem não ser de fácil implementação ou podem não remover de fato o problema de *deadlock* do modelo.

Existe uma tradução quase direta entre a estrutura de uma rede de Petri e um sequente da Lógica Linear, dessa forma, trabalhos apresentados por Girault, Pradier-Chézalviel e Valette (1997), Pradin-Chézalviel et al. (1999) e Champagnat, Pradin-Chézalviel e Valette (2000) mostram a ligação entre a teoria das redes de Petri e a Lógica Linear.

No contexto dos processos de *workflow* simples, Passos e Julia (2009) mostram como a Lógica Linear pode ser utilizada para provar o critério de correção *Soundness* para *WorkFlow nets* sem que seja necessária a construção de um grafo das marcações acessíveis, considerando diretamente a própria estrutura da *WorkFlow net* em vez de considerar seu autômato correspondente. Já no contexto de *workflows* interorganizacionais, Passos e Julia (2013) formalizam métodos baseados na Lógica Linear para verificação de processos modelados por IOWF-nets.

Dentro do contexto apresentado, a proposta deste trabalho é formalizar uma nova abordagem baseada na Lógica Linear para tratar as situações de *deadlock* em processos de *workflow* interorganizacionais modelados por *WorkFlow nets* Interorganizacionais que são localmente, mas não globalmente *sound*. A estratégia fundamenta-se na substituição de certos elementos de comunicação do tipo assíncrona (aqueles responsáveis pelas situações de *deadlock*) por um tipo de mecanismo de comunicação síncrona, forçando certos processos de *workflow* locais a iniciarem tarefas específicas simultaneamente, criando um

tipo de regra de sincronização.

A aplicação da abordagem proposta poderá alterar o critério de correção *Soundness* assim como o desempenho da execução dos processos do modelo; dessa forma, é realizada neste trabalho uma análise qualitativa e uma análise quantitativa após a aplicação da regra de sincronização na IOWF-net.

A aplicação prática da modelagem e análise do modelo depende da existência de ferramentas computacionais para apoiar a criação e manipulação dos modelos. Para auxiliar na criação e nas análises qualitativa e quantitativa do modelo, a ferramenta de simulação CPN Tools será utilizada (RATZER et al., 2003).

A ideia do presente trabalho foi inspirada em uma abordagem proposta em Mugarza et al. (2001), onde uma regra de redução baseada na Lógica Linear é proposta para produzir a concatenação de várias transições e remover as situações de *deadlock*; porém essa regra de redução é somente utilizada no contexto de processos puramente sequenciais (processos sem roteiros paralelos e alternativos). Essa pesquisa também é inspirada no trabalho de Passos e Julia (2014), onde através da análise dos sequentes da Lógica Linear correspondentes aos potenciais cenários em uma IOWF-net, é possível identificar os cenários do modelo que possuem *deadlock*. Já na abordagem deste trabalho em particular, utiliza-se a análise dos sequentes da Lógica Linear correspondentes aos potenciais cenários para detecção do lugar de comunicação assíncrona responsável pela situação de *deadlock* no cenário onde a propriedade *Soundness* não é respeitada.

1.1 Motivação

Quando se trata de processos *workflow*, as situações de total travamento são indesejadas. Esses travamentos são conhecidos como *deadlocks* e descrevem uma situação desfavorável quando a execução de um processo é bloqueada e não pode ser devidamente encerrada ou concluída. De um modo geral, tais situações surgem devido as exigências que não podem ser satisfeitas (HOLT, 1972).

O problema abordado nesse trabalho é como possibilitar a comunicação de *workflows* organizacionais, criando um *workflow* interorganizacional, de forma que ele não possua o problema de *deadlock*. É importante que o problema de *deadlock* não esteja presente em *workflows* interorganizacionais para que *workflows* organizacionais consigam se comunicar, finalizando os cenários de sucesso e não ocorrendo travamentos na execução dos processos. Para isso, é necessário detectar e remover o *deadlock* na definição do processo de *workflow* antes da execução real do processo correspondente, considerando que detectar situações de *deadlock* em processos interorganizacionais em tempo de execução é difícil e usualmente requer um custo computacional alto (SHIM; HAN; KIM, 2002).

A modelagem de processos de *workflow* pode ser feita utilizando diferentes linguagens de modelagem como *Business Process Model and Notation* (BPMN) (WHITE; MI-

ERS, 2008), os Diagramas de Atividades da *Unified Modeling Language* (UML) (DUMAS; HOFSTEDE, 2001) e as redes de Petri. Porém, linguagens de modelagem consideradas semiformais ou informais, como é o caso de BPMN e UML, não permitem a verificação dos processos de *workflow*.

De acordo com Murata (1989), as redes de Petri são apropriadas para modelar Sistemas de Tempo Real, uma vez que elas permitem a representação de situações de conflito, compartilhamento de recursos, representação de comunicações entre processos e restrições de precedência. Em Aalst (1998) são citados os motivos para utilizar as redes de Petri para modelar processos de *workflow* como a sua semântica formal, natureza gráfica, expressividade, propriedades e técnicas de análise.

A situação ideal é aquela em que os processos interorganizacionais são *sound*, mas pode ser difícil estabelecer a correção *Soundness* de processos de *workflow* interorganizacionais complexos. De fato, a correção do critério *Soundness* nos processos de *workflow* locais não garante a correção *Soundness* no modelo do *workflow* interorganizacional, como é mostrado em Aalst (1998). Quando um processo de *workflow* interorganizacional é localmente, porém não globalmente *sound*, a situação de *deadlock* pode ter sido introduzida pela troca de mensagens, como mostrado em Xiong, Zhou e Pu (2009). Uma maneira possível de prevenir as situações de *deadlock* que são causadas pela troca de mensagens é então sincronizar partes dos processos locais (AALST, 1998).

Em Pradin-Chézalviel, Valette e Kunzle (1999), é apresentada a tradução de uma rede de Petri em fórmulas da Lógica Linear. A vantagem de utilizar a Lógica Linear, como mostrado em Passos e Julia (2009), é que a complexidade de tempo para provar um sequente da Lógica Linear, que corresponde à um cenário de um processo de *workflow*, é linear.

Diante dos argumentos expostos, o presente trabalho apresenta uma abordagem baseada na análise dos sequentes da Lógica Linear para tratar as situações de *deadlock* em processos de *workflow* interorganizacionais modelados por IOWF-*nets* que são localmente, porém não globalmente *sound*. A substituição de elementos de comunicação assíncrona por uma regra de sincronização forçará, em particular, os processos de *workflow* locais a executarem simultaneamente certas atividades, removendo a situação de *deadlock* do modelo.

1.2 Objetivos e Desafios da Pesquisa

Considerando os problemas apresentados na seção 1.1, o objetivo geral desta pesquisa é de formalizar um método baseado na Lógica Linear para remover as situações de *deadlock* em processos de *workflow* interorganizacionais onde os processos de *workflow* locais são *sound*, porém o processo de *workflow* global não é *sound*, permitindo que todos os cenários finalizem com sucesso. Os objetivos específicos da presente pesquisa são:

- a) definir uma regra de sincronização para eliminar as situações de *deadlock* provocadas por trocas de mensagens assíncronas;
- b) definir um método de análise qualitativo baseado no cálculo dos sequentes da Lógica Linear para verificar, após aplicação da regra de sincronização, que o novo modelo produzido respeita a propriedade *sound* (que todos os cenários finalizam corretamente);
- c) definir um método de análise quantitativa baseado numa álgebra derivada das árvores de prova do cálculo dos sequentes da Lógica Linear para avaliar a alteração do desempenho de certos cenários *sound* no novo modelo após a sincronização de certas partes que se comunicam nos processos;
- d) validar a abordagem proposta através de um estudo de caso, utilizando o método formalizado no contexto dos processos de *workflow* interorganizacionais ao contexto da composições de *Web Services*; em particular, realizar um estudo comparativo entre os resultados produzidos através das análises qualitativas e quantitativas baseadas no cálculo dos sequentes da Lógica Linear e resultados de simulação dos modelos de processos interorganizacionais implementados na ferramenta de edição, análise e simulação de redes de Petri coloridas CPN Tools.

1.3 Hipóteses

Com base nos objetivos definidos na seção 1.2, as seguintes hipóteses são consideradas:

- a) a utilização da Lógica Linear, mais especificamente a construção e análise de árvores de prova da Lógica Linear, permite identificar situações de *deadlock* em *WorkFlow nets* Interorganizacionais que são localmente *sound*; em particular é possível identificar também os elementos de comunicação assíncrona responsáveis pelas situações de travamento nas comunicações entre processos de *workflow* distintos;
- b) a substituição dos elementos de comunicação assíncrona responsáveis por um estado de *deadlock* em uma *WorkFlow net* Interorganizacional por uma regra de sincronização entre partes específicas de processos distintos constitui uma condição necessária e suficiente para que todas atividades dos processos que estão diretamente envolvidas com a situação de *deadlock* possam ser corretamente executadas e os processos finalizados;
- c) em caso de sincronização de certas partes dos processos de uma *WorkFlow net* Interorganizacional, somente as partes diretamente alteradas do modelo deverão ser reanalisadas, considerando somente fragmentos das árvores de prova do cálculo dos sequentes da Lógica Linear que foram geradas durante a análise qualitativa responsável pela detecção de situações de *deadlock*;

- d) a sincronização de certas partes dos processos de uma *WorkFlow net* Interorganizacional pode gerar atrasos na finalização das atividades dos processos e, consequentemente, provocar uma degradação do desempenho global do sistema;
- e) modelos e métodos de análise baseados em uma *WorkFlow net* Interorganizacional e no cálculo dos sequentes da Lógica Linear, apresentados neste trabalho, podem ser diretamente estendidos a modelos de *Web Services* modelados por redes de Petri.

1.4 Contribuições

Com a realização desta pesquisa, pretende-se contribuir com a área de processos de *workflow* no que diz respeito à prevenção de situações de *deadlock* em processos de *workflow* interorganizacionais.

Uma vez que os objetivos traçados na seção 1.2 forem alcançados e com a confirmação das hipóteses levantadas na seção 1.3, as contribuições adquiridas serão as seguintes:

- a) um método para remoção de situações de *deadlock* em processos de *workflow* interorganizacionais, baseado na construção e análise de árvores de prova da Lógica Linear;
- b) um método de análise qualitativa (verificação da propriedade *sound*) de processos de *workflow* interorganizacionais modelados por IOWF-*nets* baseado no cálculo dos sequentes da Lógica Linear;
- c) um método de análise quantitativa (cálculo de datas simbólicas de início e fim de realização de atividades) de processos de *workflow* interorganizacionais modelados por IOWF-*nets* baseado no cálculo dos sequentes da Lógica Linear;
- d) realização de um estudo de caso que mostra a remoção das situações de *deadlock* em composições de *Web Services* modeladas por redes de Petri, utilizando o método proposto nesta pesquisa.

1.5 Organização da Dissertação

A presente dissertação encontra-se estruturada em 6 capítulos, organizados da maneira como se segue.

O Capítulo 2 apresenta os fundamentos teóricos necessários para entendimento da abordagem proposta. Os fundamentos teóricos relacionados às redes de Petri, às *WorkFlow nets* e às *WorkFlow nets* Interorganizacionais são apresentados respectivamente nas seções 2.1, 2.2 e 2.3. A seção 2.5 apresenta a Lógica Linear e suas relações com as redes de Petri. Por fim, na seção 2.6 é apresentado o conceito de sifão para definição de situações de *deadlock* nas redes de Petri.

No Capítulo 3, os trabalhos relacionados são apresentados. Nele, as principais contribuições existentes na literatura abordando o tema de “prevenção de *deadlocks* em *WorkFlow nets* Interorganizacionais” são descritas juntamente com as limitações existentes em cada proposta apresentada.

O Capítulo 4 apresenta a abordagem proposta neste trabalho. Em particular, a seção 4.1 apresenta a regra de sincronização para remover situações de *deadlock* em *WorkFlow nets* Interorganizacionais. A seção 4.2 apresenta a análise qualitativa antes e após a aplicação da regra formalizada nesta pesquisa, preocupando-se com o critério de correção *Soundness* do modelo. Já a seção 4.3 apresenta a análise quantitativa antes e após a aplicação da regra apresentada neste trabalho, preocupando-se com o desempenho do tempo de execução do modelo através do cálculo de datas simbólicas das árvores de prova da Lógica Linear e a utilização da ferramenta de simulação CPN Tools.

O Capítulo 5, por sua vez, apresenta um estudo de caso ilustrativo aplicado à composição de *Web Services* para ilustrar a abordagem proposta neste trabalho. Primeiramente na seção 5.1 é apresentada a relação dos *Web Services* com as *WorkFlow nets* Interorganizacionais. Na seção 5.2 é apresentada a aplicação da regra de sincronização no modelo. Na seção 5.3 é feita a análise qualitativa do modelo antes e depois da aplicação da regra, enquanto na seção 5.4 é feita a análise quantitativa.

Finalmente, no Capítulo 6, a conclusão desta pesquisa é apresentada. A seção 6.1 apresenta as principais contribuições. Na seção 6.2, os trabalhos futuros são propostos. Finalmente, na seção 6.3, as contribuições em produção bibliográfica obtidas no contexto da presente pesquisa são listadas.

Fundamentação Teórica

Neste capítulo são introduzidos os principais conceitos teóricos existentes na literatura e que serão necessários para a apresentação da proposta no Capítulo 4. Este capítulo está organizado como segue: a seção 2.1 apresenta as terminologias, propriedades e notações básicas relacionadas à teoria das redes de Petri; na seção 2.2 são descritos os principais conceitos relacionados às *WorkFlow nets*; já na seção 2.3 são introduzidos os conceitos necessários para o entendimento das *WorkFlow nets* Interorganizacionais; na seção 2.4 é apresentado o *software* CPN Tools; na seção 2.5 é feita a relação entre a teoria das redes de Petri e a Lógica Linear; por fim, o problema de *deadlock* no contexto das redes de Petri é apresentado na seção 2.6.

2.1 Redes de Petri

A teoria das redes de Petri foi proposta inicialmente na tese de doutorado de Carl Adam Petri, apresentada em 1962 à Universidade de Darmstadt na Alemanha. O seu trabalho consistiu na apresentação de um tipo de grafo bipartido com estados associados, com o objetivo de estudar os mecanismos de comunicação entre autômatos (PETRI, 1962). Nos anos seguintes, impulsionado por um grupo de pesquisadores do MIT (Massachusetts Institute of Technology) e interessado pelo trabalho inicial de Carl Adam Petri, Anatol W. Holt lançou entre 1968 e 1976 as bases teóricas das redes de Petri (CARDOSO; VALETTE, 1997). Após Petri apresentar seus primeiros modelos, alguns pesquisadores, no sentido de prestarem uma homenagem a sua descoberta, os denominaram de redes de Petri (RdP).

Desde então, o estudo e uso das redes de Petri tem crescido consideravelmente. Apesar de ser considerada uma teoria relativamente jovem, a teoria se adapta bem a um grande número de aplicações em que as noções de eventos e de evoluções simultâneas são importantes (CARDOSO; VALETTE, 1997). Uma rede de Petri pode ser vista como uma ferramenta matemática, gráfica e formal que permite a modelagem, análise e controle de sistemas a eventos discretos que comportam atividades paralelas, concorrentes e assíncronas (MURATA, 1989).

Formalmente, uma rede de Petri é definida da seguinte forma (MURATA, 1989) (CARDOSO; VALETTE, 1997):

Definição 1. (Rede de Petri) Uma rede de Petri PN é uma tripla $PN = (P, T, F)$ onde:

- a) P é um conjunto finito de lugares de dimensão n ;
- b) T é um conjunto finito de transições de dimensão m ;
- c) $F \subseteq (P \times T) \cup (T \times P)$ é um conjunto de arcos (relação que define os arcos que ligam as transições aos lugares e os lugares às transições).

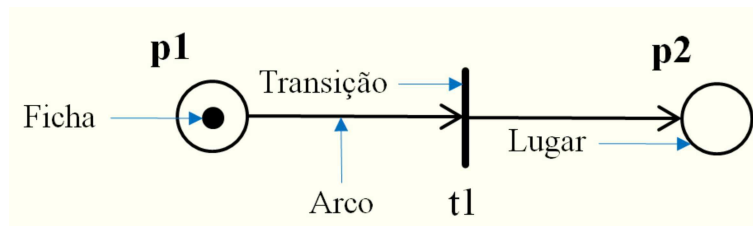


Figura 1 – Rede de Petri e seus elementos básicos.

Segundo Murata (1989) e Cardoso e Valette (1997), os elementos básicos que permitem a definição de uma rede de Petri, que podem ser vistos na Figura 1, são polivalentes e em grande medida podem ser interpretados livremente. Estes elementos são apresentados a seguir (MURATA, 1989):

- a) **lugar (representado por um círculo):** pode ser interpretado como uma condição, um estado parcial, uma espera, um procedimento, um conjunto de recursos, um estoque, uma posição geográfica num sistema de transporte, etc. Em geral, todo lugar tem um predicado associado; por exemplo, máquina livre, peça em espera, etc;
- b) **transição (representada por uma barra ou um retângulo):** é associada a um evento que ocorre no sistema, como o evento iniciar uma operação, por exemplo;
- c) **ficha (representada por um ponto em um lugar):** é um indicador significando que a condição associada ao lugar é verificada. Pode representar um objeto (recurso ou peça), numa certa posição geográfica (num determinado estado), ou ainda uma estrutura de dados. Por exemplo, uma ficha no lugar máquina livre indica que uma máquina está livre (predicado verdadeiro). Se não tem fichas neste lugar, o predicado é falso, consequentemente a máquina não está livre. Se no lugar peças em espera houvesse três fichas, indicaria que existem três peças em espera;
- d) **arco (representado por uma seta):** liga as transições aos lugares e os lugares às transições.

Vale ressaltar que um lugar nunca é ligado a outro diretamente e uma transição nunca é ligada a outra diretamente, ou seja, um lugar sempre está ligado a uma ou mais transições por meio de arcos e uma transição; por sua vez, sempre está ligada a um ou mais lugares também por intermédio dos arcos (MURATA, 1989).

Um lugar p é chamado de lugar de entrada de uma transição t se, e somente se, existe um arco direcionado de p para t . Por exemplo, considerando a rede de Petri da Figura 1, o lugar $p1$ é lugar de entrada da transição $t1$. Já um lugar p é chamado de lugar de saída de uma transição t se, e somente se, existe um arco direcionado de t para p . Ainda na Figura 1, o lugar $p2$ é um lugar de saída de $t1$.

O modelo de uma rede de Petri marcada é formalizado na Definição 2 (MURATA, 1989) (CARDOSO; VALETTE, 1997).

Definição 2. (Rede de Petri marcada) Uma rede de Petri marcada PN é uma dupla $PN = (R, M)$ onde:

- a) R é uma rede de Petri.
- b) M é a marcação inicial dada pela aplicação $M: P \times T \rightarrow \mathbb{N}$, com \mathbb{N} sendo o conjunto dos números naturais.

$M(p)$ representa o número de fichas contidas no lugar p . A marcação M é a distribuição das fichas nos lugares, sendo representada por um vetor coluna cuja dimensão corresponde ao número de lugares e os elementos são $M(p)$.

Um lugar p , quando se considera uma rede de Petri marcada, contém, em um dado momento, zero ou mais fichas (*tokens*), que são representadas por pontos pretos (\bullet). Por exemplo, considerando a rede de Petri da Figura 2(a), o lugar $p1$ contém uma ficha e os lugares $p2$ e $p3$ contêm zero fichas. A marcação de uma rede de Petri refere-se à distribuição de fichas nos lugares, sendo que o número de fichas pode mudar durante a execução da rede.

As transições são componentes ativos em uma rede de Petri, pois elas mudam a marcação da rede de acordo com as seguintes regras de disparo (MURATA, 1989):

- a) uma transição t é dita sensibilizada se, e somente se, cada lugar de entrada p de t contém pelo menos uma ficha. Por exemplo, a transição $t1$ da rede de Petri da Figura 2(a) não está sensibilizada. Já a transição $t1$ da rede de Petri da Figura 2(b) está sensibilizada, pois há uma ficha em cada lugar de entrada desta transição;
- b) uma transição sensibilizada pode disparar. Se a transição t disparar, então t consome uma ficha de cada lugar de entrada p de t e produz uma ficha em cada lugar de saída p de t . As redes de Petri das Figuras 2(b) e 2(c) mostram um exemplo de disparo de transição, isto é, a marcação inicial é representada pela Figura 2(b) e a marcação final, obtida após o disparo da transição $t1$, é representada pela Figura 2(c). Neste caso, $t1$ consome uma ficha de cada lugar de entrada ($p1$ e $p2$) e produz uma ficha em cada lugar de saída ($p3$).

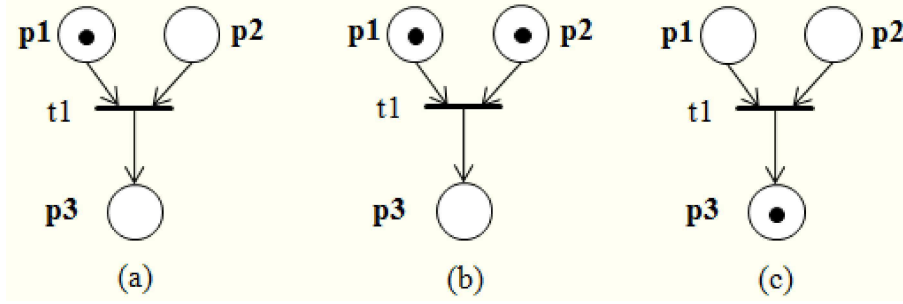


Figura 2 – Exemplos de sensibilização e disparo de transição em uma rede de Petri.

2.1.1 Propriedades das redes de Petri

Uma rede de Petri não se resume apenas a uma ferramenta que permite a modelagem de problemas que tenham atividades concorrentes. Ela pode ser utilizada também para descrever e analisar sistemas (DAVID; ALLA, 2010). A dinâmica de um sistema descrito por uma rede de Petri é dada pela evolução das marcações. Em (MURATA, 1989) são definidas as propriedades relativas às redes de Petri marcadas de acordo com o conjunto de marcações acessíveis a partir da marcação inicial. Essas propriedades são conhecidas na literatura como boas propriedades das redes de Petri e são apresentadas a seguir (MURATA, 1989):

- a) **Alcançabilidade** (*Reachability*): a alcançabilidade é a base fundamental para o estudo das propriedades dinâmicas de qualquer sistema. Uma marcação M_n é dita alcançável se existe uma sequência de disparo de transições que a partir da marcação inicial M_0 possibilita a chegada à M_n . Essa propriedade garante que certos estados sempre serão atingidos;
- b) **Limitabilidade** (*Boundedness*): uma rede de Petri marcada N é dita k -limitada ou simplesmente limitada se, e somente se, todos os seus lugares são k -limitados. Essa propriedade garante que o número de fichas em cada lugar não excede um número finito k para qualquer marcação alcançável a partir da marcação inicial;
- c) **Vivacidade** (*Liveness*): uma rede de Petri é dita viva se qualquer transição t da rede é disparável a partir de qualquer marcação acessível do conjunto das marcações acessíveis (grafo de marcações produzidas a partir da marcação inicial), não importando a sequência de disparos seguida para disparar t . Isto significa que, para ser viva, a transição t deve poder ser sensibilizada a partir de qualquer marcação. Essa propriedade garante que o sistema é livre de *deadlock*, isto é, todas as transições da rede são disparáveis;
- d) **Reiniciabilidade** (*Reversibility*): uma rede é dita reiniciável se, qualquer que seja a sequência de disparo de transições seguida, é sempre possível voltar à marcação inicial M_0 a partir de qualquer marcação do conjunto das marcações acessíveis (grafo de marcações produzidas a partir da marcação inicial). Essa propriedade é de

vital importância quando um sistema modelado tem seu funcionamento repetitivo. Uma rede de Petri é repetitiva se existir uma sequência de disparos, associada a uma dada marcação, na qual todas as ações são executadas um número de vezes infinito. Se a sequência existir, mas apenas algumas ações forem disparadas ilimitadamente, a rede é dita parcialmente repetitiva.

2.1.2 Rede de Petri Ordinária

Na terminologia original de Carl Adam Petri, uma rede lugar/transição é uma rede de Petri não marcada, que é ao mesmo tempo generalizada e de capacidade finita (DAVID; ALLA, 2010). Conforme terminologia utilizada por David e Alla (2010) essa subclasse é formalmente definida como redes de Petri ordinárias. Por serem as mais divulgadas, são normalmente referenciadas apenas como redes de Petri e estão definidas na subseção 2.1 deste capítulo.

Uma rede de Petri ordinária permite a modelagem de estados, eventos e condições. As redes de Petri correspondem a uma técnica de descrição formal poderosa que possibilita a modelagem de sistemas concorrentes de forma natural, porém a sua aplicação para processos em sistemas reais pode ser inadequada, pois os modelos obtidos podem ser complexos, de grandes dimensões e não hierárquicos, dificultando a sua compreensão (MACIEL; LINS; CUNHA, 1996) (AALST, 1998). Além disso, não permitem a modelagem de dados e tempo. Para resolver esse problema é necessária a apresentação de extensões do modelo básico da rede de Petri como a extensão com cor e a extensão com tempo. Essas extensões são conhecidas como redes de Petri de alto nível (AALST, 1998). Na sequência são apresentados os modelos de rede de Petri de alto nível necessários para a compreensão do presente trabalho.

2.1.3 Rede de Petri Temporal e rede de Petri Temporizada

Para sistemas reais, muitas vezes é importante descrever o comportamento temporal do sistema, ou seja, é necessário modelar durações e atrasos. Uma vez que uma rede de Petri ordinária não é capaz de manusear quantitativamente o tempo, o conceito de tempo é acrescentado ao modelo. Os primeiros modelos que relacionavam as redes de Petri com o tempo foram as redes de Petri temporais introduzidas por Merlin (1974), as redes de Petri temporizadas por Ramchandani (1974) e as redes de Petri estocásticas por Molloy (1981). De um modo geral, as redes de Petri temporais são adequadas para serem utilizadas em aplicações de simulação, diagnóstico, supervisão e análise de desempenho (JULIA; OLIVEIRA; VALETTE, 2008).

Nas redes de Petri p-temporizadas, o tempo é representado por durações associadas aos lugares do modelo (SIFAKIS, 1977), enquanto nas redes de Petri t-temporizadas o tempo é representado por durações (números racionais positivos ou nulos) associadas às

transições (RAMCHANDANI, 1974). Nas redes de Petri t-temporais (MERLIN, 1974) o tempo é representado por um intervalo $[\theta_{\min}, \theta_{\max}]$ associado a cada transição e a duração da sensibilização deve ser maior que o θ_{\min} e menor que θ_{\max} , ou seja, a transição só pode ser disparada dentro deste intervalo de tempo.

A Figura 3 apresenta o exemplo de uma rede de Petri t-temporal. O modelo é baseado em um exemplo de rede de Petri apresentado em Aalst (1997) que trata do processamento de pedidos relacionados a danos de carros. Para cada atividade do processo foi atribuído um intervalo de tempo nas transições, com tempo mínimo e tempo máximo para disparo. Por exemplo, a transição $t4$ indica que esta transição irá disparar no mínimo em 2 unidades de tempo e no máximo em 7 unidades de tempo após ter sido sensibilizada.

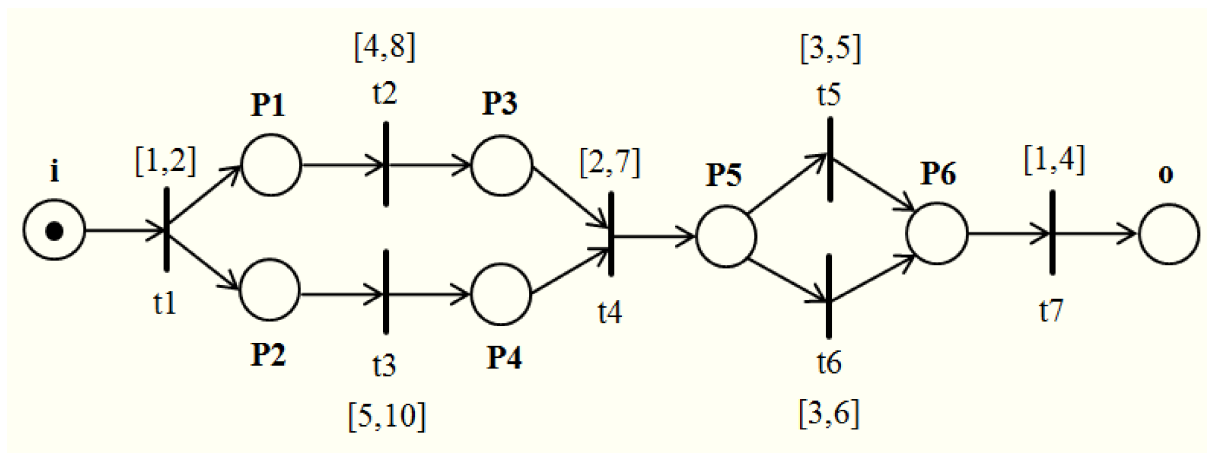


Figura 3 – Exemplo de rede de Petri t-temporal.

2.1.4 Rede de Petri Colorida

As fichas representam frequentemente objetos nos sistemas modelados, como por exemplo, recursos, produtos e pessoas. Para modelar sistemas constituídos de muitos componentes idênticos que interagem entre si, o modelo das redes de Petri do tipo ordinária pode exibir grande redundância (MACIEL; LINS; CUNHA, 1996). Isto se deve ao fato de que para diferenciar dois componentes idênticos é preciso especificar uma estrutura idêntica de sub-rede para cada um dos componentes. Portanto, muitas vezes é necessária a representação das características desses itens. As redes ordinárias não possibilitam ainda níveis de refinamento do modelo, ou seja, não existe a noção de hierarquização (MACIEL; LINS; CUNHA, 1996).

Para solucionar este problema, Jensen (1991) criou as redes de Petri coloridas, que são uma extensão das redes de Petri clássicas nas quais as fichas podem ter diferentes tipos de informação associada. O principal objetivo das redes de Petri coloridas é a redução do tamanho do modelo, permitindo que cada ficha tenha um valor, normalmente referido

Para apresentar as redes de Petri coloridas, a Figura 4 é utilizada; trata-se de um exemplo apresentado em Maciel, Lins e Cunha (1996) onde uma rede modela a manufatura de dois tipos de produtos distintos através de duas linhas de manufatura onde recursos (máquinas) são compartilhados nos processos de fabricação.

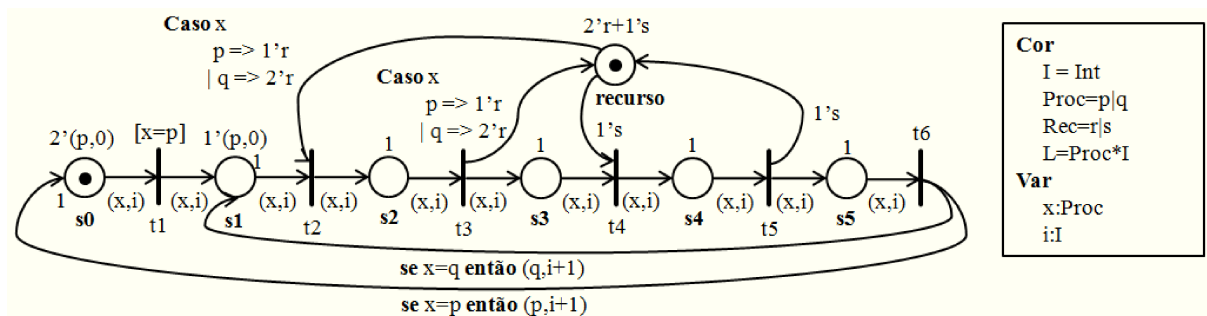


Figura 4 – Exemplo de rede de Petri colorida.

Uma rede de Petri que modela os aspectos de um processo de *workflow* é chamada de *Workflow net* (WF-*net*) (AALST, 1998).

A definição formal de uma *WF-net* dada por Aalst (1998) é apresentada a seguir.

Definição 3. (*Workflow net*) Uma rede de Petri $PN = (P, T, F)$ pode ser considerada uma *WF-net* se satisfaz as seguintes propriedades:

- a) possui apenas um lugar de início (i) e apenas um lugar de término (o), sendo estes dois lugares tratados como lugares especiais. O lugar i tem apenas arcos de saída e o lugar o tem apenas arcos de entrada. Uma ficha em i representa um caso que precisa ser tratado e uma ficha em o representa um caso que já foi tratado;
- b) todo nó da rede, lugar e transição, deve estar em um caminho entre os lugares de início e fim.

2.2.1 Processos

Um processo define quais tarefas precisam ser executadas e em qual ordem a execução deve ocorrer. Modelar um processo de *workflow* em termos de uma *WF-net* é bem direto,

sendo que as transições são componentes ativos e modelam as tarefas, enquanto que os lugares são componentes passivos e modelam as condições (pré e pós). Por fim, têm-se as fichas que modelam os casos (AALST, 1998).

Para ilustrar o mapeamento de processos de *workflow* em *WF-nets*, considera-se a adaptação do exemplo apresentado em Aalst (1997) que trata do processamento de pedidos relacionados a danos de carros. As tarefas necessárias para processar um pedido são *verificar seguro* ($t2$), *contatar garagem* ($t3$), *pagar danos* ($t5$), *enviar carta* ($t6$) e *finalizar pedido* ($t7$) que são modeladas pelas transições. As tarefas $t2$ e $t3$ determinam se o pedido é aceito; essas tarefas são executadas em paralelo. Se o pedido é aceito, então o dano é pago (tarefa $t5$). Caso contrário, uma carta de rejeição é enviada ao requerente (tarefa $t6$). Por fim, independente do pedido ter sido aceito ou não, ele é finalizado (tarefa $t7$) e o processo é concluído.

A Figura 5 ilustra a *WF-net* que representa este processo.

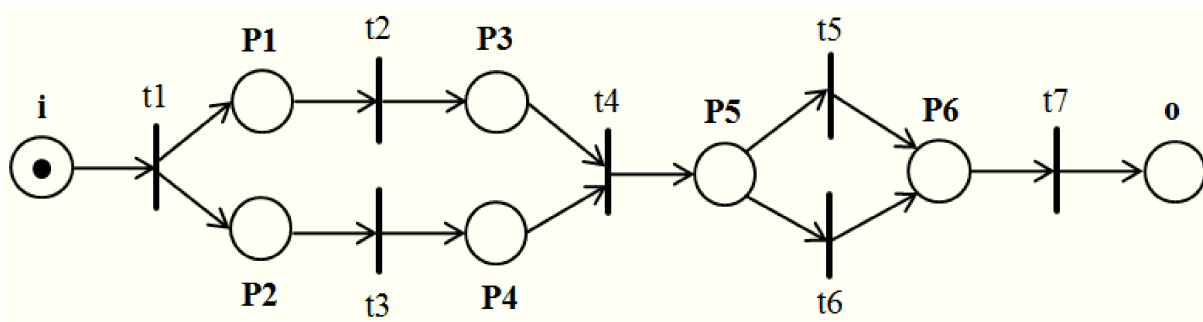


Figura 5 – Exemplo de *WF-net*.

2.2.2 Roteamentos

De acordo com Aalst e Hee (2004), pela rota de um caso, ao longo de uma série de tarefas, pode-se determinar quais tarefas precisam ser executadas (e em que ordem). Quatro construções básicas para o roteamento de tarefas são consideradas:

- sequencial:** a forma mais simples de execução de tarefas, na qual uma tarefa é executada após a outra, havendo claramente, dependência entre elas;
- paralela:** mais de uma tarefa pode ser executada simultaneamente, ou em qualquer ordem. Neste caso, as tarefas podem ser executadas sem que o resultado de uma interfira no resultado da outra;
- condicional (ou rota seletiva):** quando há uma escolha entre duas ou mais tarefas;
- iterativa:** quando é necessário executar uma mesma tarefa múltiplas vezes.

Considerando o processo de pedidos relacionados a danos de carros, apresentado na Figura 5, as tarefas *verificar seguro* ($t2$) e *contatar garagem* ($t3$) são um exemplo de

roteamento paralelo, enquanto que as tarefas *pagar danos* ($t5$) e *enviar carta* ($t6$) são um exemplo de roteamento condicional.

2.2.3 Soundness

A propriedade *Soundness* é o principal critério de correção definido para as WF-nets. Ela se relaciona com a dinâmica de uma WF-net. Em Aalst (1998), são apresentados os requisitos necessários para uma WF-net atender o critério de correção *Soundness*:

- a) o primeiro requisito afirma que a partir do lugar de início (lugar i), é sempre possível chegar no lugar de término (lugar o) com uma (e apenas uma) ficha;
- b) o segundo requisito afirma que no momento em que uma ficha é colocada no lugar de término (lugar o), todos os outros lugares devem estar vazios;
- c) o último requisito afirma que não há transições mortas (tarefas) a partir do lugar inicial (lugar i).

É comum o termo término adequado ser usado para descrever os dois primeiros requisitos. A definição formal da propriedade *Soundness* é apresentada a seguir (AALST, 1998).

Definição 4. (*Soundness*) Um processo modelado por uma WF-net $PN = (P, T, F)$ é *sound*, se e somente se:

- a) para cada marcação alcançável M a partir do lugar i , existe uma sequência de disparo partindo da marcação M que leva ao lugar o . Formalmente:

$$\forall_M (i \xrightarrow{*} M) \Rightarrow (M \xrightarrow{*} o);$$

- b) a marcação o é o único estado alcançável a partir da marcação i com exatamente uma ficha no lugar o quando a execução terminar. Formalmente:

$$\forall_M (i \xrightarrow{*} M \wedge M \geq o) \Rightarrow (M = o);$$

- c) não existem transições mortas em (PN, i) . Formalmente:

$$\forall_{t \in T} \exists_{M, M'} i \xrightarrow{*} M \xrightarrow{t} M'.$$

2.3 WorkFlow net Interorganizacional

Na seção anterior foram apresentadas as redes de Petri aplicadas à modelagem e análise de *workflows* que consideram uma única organização. Porém, no presente trabalho, são considerados também os *workflows* interorganizacionais. Um *workflow* interorganizacional é essencialmente um conjunto de processos de *workflow* fracamente e/ou fortemente acoplados que possui n parceiros de negócios envolvidos em um processo de *workflow* “global” (AALST, 1998).

Cada um dos parceiros tem o seu próprio processo de *workflow* “local” que é privado, isto é, o parceiro de negócio correspondente tem total controle sobre a parte local do *workflow*. Contudo, estes processos de *workflow* locais precisam se comunicar, pois eles interagem entre si para a correta execução de dados cenários de negócio (AALST, 1998).

Um processo de *workflow* global consiste em processos de *workflow* locais e uma estrutura de interação (comunicação). Existem duas maneiras de interagir entre processos distintos: a comunicação síncrona e comunicação assíncrona. A comunicação síncrona corresponde à fusão de um certo número de transições que força a execução simultânea de certas atividades dos processos de *workflow* locais, enquanto que a comunicação assíncrona corresponde à troca de mensagens entre processos de *workflow* locais através de simples lugares de comunicação. Uma rede de Petri que modela os aspectos de um processo de *workflow* interorganizacional é chamada de *WorkFlow net* Interorganizacional, em inglês *Interorganizational WorkFlow net* (IOWF-net) (AALST, 1998).

A definição formal de uma IOWF-net é apresentada na sequência (AALST, 1998).

Definição 5. (*WorkFlow net* Interorganizacional) Uma IOWF-net é definida como uma tupla $IOWF = (PN_1, PN_2, \dots, PN_n, P_{AC}, AC, T_{SC}, SC)$, onde:

- a) $n \in \mathbb{N}$ é o número de *WorkFlow nets* Locais, do inglês *Local WorkFlow nets* (LWF-nets);
- b) para cada $k \in \{1, \dots, n\}$: PN_k é uma WF-net com lugar de início i_k e lugar de término o_k ;
- c) para cada $k, l \in \{1, \dots, n\}$: se $k \neq l$, então $(P_k \cup T_k) \cap (P_l \cup T_l) = \emptyset$;
- d) $T^* = \bigcup_{k \in \{1, \dots, n\}} T_k$, $P^* = \bigcup_{k \in \{1, \dots, n\}} P_k$, $F^* = \bigcup_{k \in \{1, \dots, n\}} F_k$ (relação entre os elementos das LWF-nets);
- e) P_{AC} é o conjunto de elementos de comunicação assíncrona (lugares de comunicação);
- f) T_{SC} é o conjunto de elementos de comunicação síncrona (fusão de transições);
- g) $P_{AC} \cap T_{SC} = \emptyset$, $(P_{AC} \cup T_{SC}) \cap (P^* \cup T^*) = \emptyset$;
- h) $AC \subseteq P_{AC} \times \mathbb{P}(T^*) \times \mathbb{P}(T^*)$ é a relação de comunicação assíncrona¹;
- i) $SC \subseteq T_{SC} \times \mathbb{P}(T^*)$ é a relação de comunicação síncrona;
- j) para cada $(t1, x1), (t2, x2) \in SC$: se $t1 \neq t2$, então $x1 \cap x2 = \emptyset$.

Cada elemento de comunicação assíncrona corresponde a um lugar em P_{AC} e a relação AC especifica um conjunto de transições de entrada e um conjunto de transições de saída para cada elemento de comunicação assíncrona (AALST, 1998).

Para ilustrar os conceitos definidos acima, é considerado o processo de *workflow* interorganizacional apresentado em Aalst (1998), que modela um processo que precede a

¹ $\mathbb{P}(T^*)$ é o conjunto de todos os subconjuntos não vazios de T^*

apresentação de um artigo em uma conferência. A Figura 6 apresenta a IOWF-net que modela este processo.

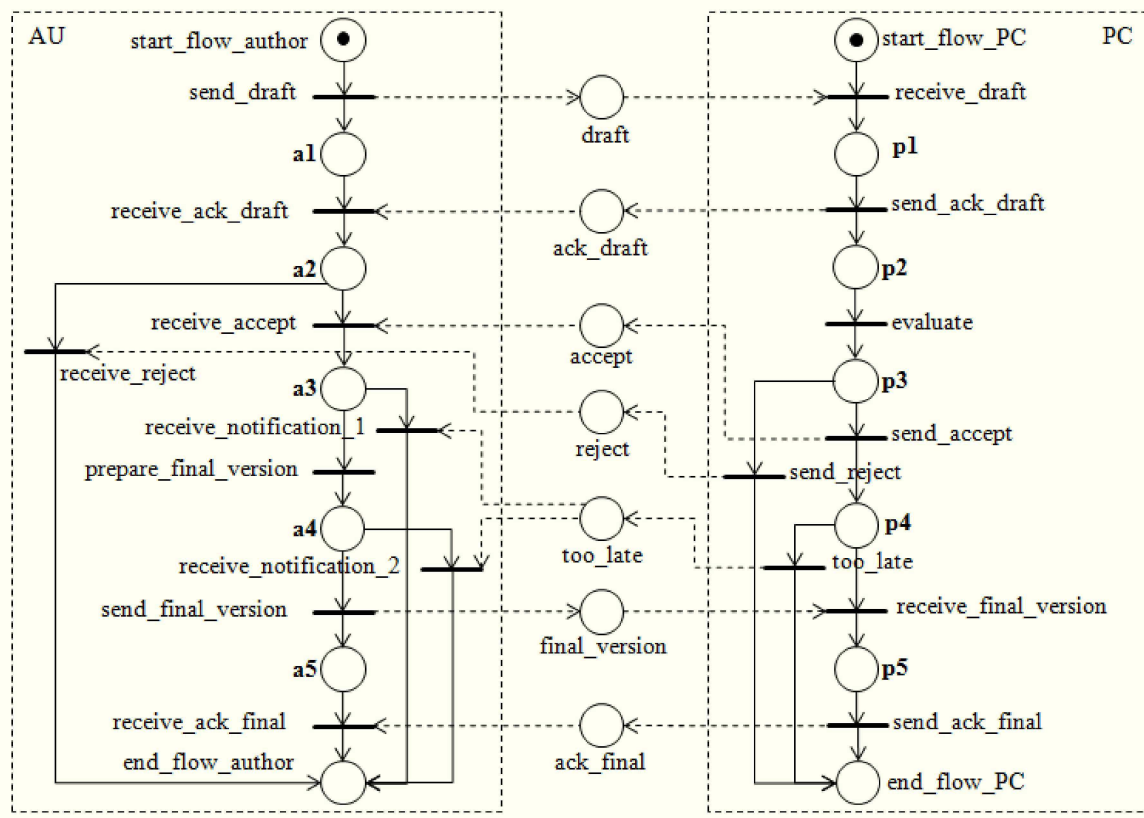


Figura 6 – Exemplo de IOWF-net.

Esta IOWF-net possui duas LWF-nets: *Author* e *PC*. Cada uma destas LWF-nets possui um lugar de início e um lugar de término. No caso da LWF-net *Author*, o lugar de início é *start_flow_author* e o lugar de término é *end_flow_author*. Na LWF-net *PC*, o lugar de início e término são *start_flow_PC* e *end_flow_PC*, respectivamente. Os lugares *draft*, *ack_draft*, *accept*, *reject*, *too_late*, *final_version* e *ack_final* são os lugares de comunicação.

É possível utilizar a técnica de verificação de correção de um processo de *workflow* para os processos de *workflow* locais de um *workflow* global. Porém, um *workflow* interorganizacional que é composto por um número de *workflows* locais que são *sound* pode estar sujeito a erros de sincronização. É possível que *deadlocks* sejam introduzidos pelos elementos de comunicação (AALST, 1998).

Devido a estes problemas, é interessante uma noção sobre a propriedade *Soundness* para IOWF-nets. Para definir uma noção global de *Soundness*, é apresentado o conceito de *Unfolded Interorganizational WorkFlow net* (*U(IOWF-net)*).

Conforme Aalst (1998), em uma *U(IOWF-net)* todas as LWF-nets são conectadas por uma transição de início *ti* e uma transição de término *to*. Além disso, é adicionado um lugar de início global *i* e um lugar de término global *o*. O resultado corresponde a

uma nova rede de Petri do tipo $U(\text{IOWF-net})$. A Figura 7 apresenta a $U(\text{IOWF-net})$ correspondente a IOWF-net da Figura 6.

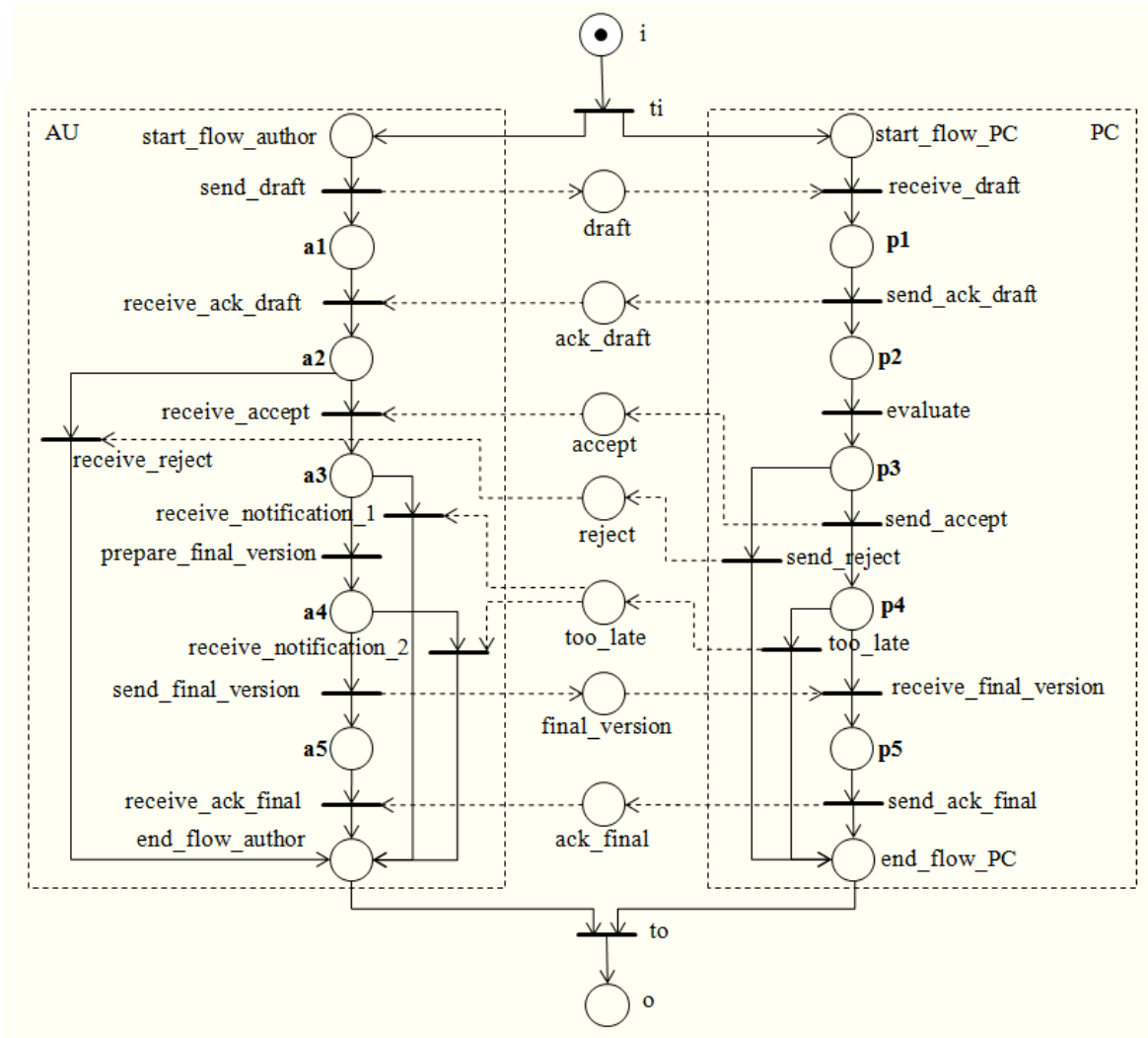


Figura 7 – $U(\text{IOWF-net})$ correspondente à IOWF-net da Figura 6.

Definição 6. (*Soundness IOWF*) De acordo com Aalst (1998), uma IOWF-net é *sound* se e somente se é localmente *sound* e também é globalmente *sound*. Uma IOWF-net é localmente *sound* se e somente se cada uma de suas WF-nets PN_k é *sound*. Uma IOWF-net é globalmente *sound* se e somente se a sua $U(\text{IOWF-net})$ é *sound*.

Considerando a $U(\text{IOWF-net})$ mostrada na Figura 7, as LWF-nets *Author* e *PC* são ambas *sound*. Porém a $U(\text{IOWF-net})$ não é *sound*. Portanto, a IOWF-net não satisfaz o critério de correção *Soundness*. É fácil observar que se a transição *too_late* da LWF-net *PC* é disparada e as transições *prepare_final_version* e *send_final_version* da LWF-net *Author* são disparadas, o modelo será levado a um estado de *deadlock* com uma ficha no lugar *a5* e as mensagens *too_late* e *final_version* nunca serão recebidas (uma ficha fica presa no lugar *too_late* e uma ficha fica presa no lugar *final_version*).

2.4 CPN Tools

A aplicação prática da modelagem e análise de modelos baseados em redes de Petri depende da existência de ferramentas computacionais para apoiar a criação e manipulação de tais modelos. O CPN Tools é uma ferramenta adequada para edição, simulação e análise de redes de Petri Coloridas. O *software* foi originalmente desenvolvido no projeto de pesquisa CPN2000, pelo Grupo CPN da Universidade de Aarhus na Dinamarca entre 2000 e 2010 e foi patrocinado por grandes organizações como a Hewlett Packard, Microsoft e Nokia (RATZER et al., 2003).

Utilizando o CPN Tools é possível investigar o comportamento de um sistema modelado no ponto de vista qualitativo, verificando as suas boas propriedades através de métodos de espaço de estado (*state space*). Um relatório padrão de espaço de estados pode ser gerado contendo informações de limitabilidade e vivacidade da rede. Além de informações do ponto de vista qualitativo, é possível gerar simulações e investigar o comportamento de um sistema modelado no ponto de vista quantitativo, verificando a análise de desempenho do modelo ao gerar replicações.

O *software* permite também estruturar os modelos de redes de Petri em módulos. O conceito de módulos em redes de Petri colorida é baseado em um mecanismo de estruturação hierárquica. A ideia básica da hierarquia em redes de Petri é permitir a construção de um amplo modelo combinando várias redes de Petri em um único modelo (RATZER et al., 2003). O conceito de hierarquia utilizado no CPN Tools é útil na modelagem e na análise das IOWF-*nets*, onde existem várias LWF-*nets* que se comunicam.

Esta ferramenta permite também modelar redes de Petri Temporais ou Temporizadas. É possível adicionar o conceito de tempo nas fichas, nas transições e/ou nos arcos, sendo permitido especificar um tempo fixo ou um intervalo de tempo de simulação nesses componentes. Com essas características, é possível realizar análises quantitativas e verificar a performance dos modelos construídos no CPN Tools.

O CPN Tools usa a linguagem CPN ML para declarações e atribuições de estruturas de dados complexas, sendo que se trata de uma linguagem baseada no padrão Standard ML (RATZER et al., 2003). O CPN ML faz com que o CPN Tools possibilite uma certa capacidade de programação, permitindo declaração de variáveis, criação de funções e estrutura de controles que poderão ser utilizadas nos lugares, arcos e transições, possibilitando a manipulação de dados abstratos complexos e a criação de condições de guarda associadas às transições do modelo.

Com todos os recursos apresentados, a ferramenta fornece as condições necessárias para modelar, simular e verificar IOWF-*nets*.

A Figura 8 apresenta a interface gráfica do *software* CPN Tools com sua área de trabalho e as paletas das suas funcionalidades.

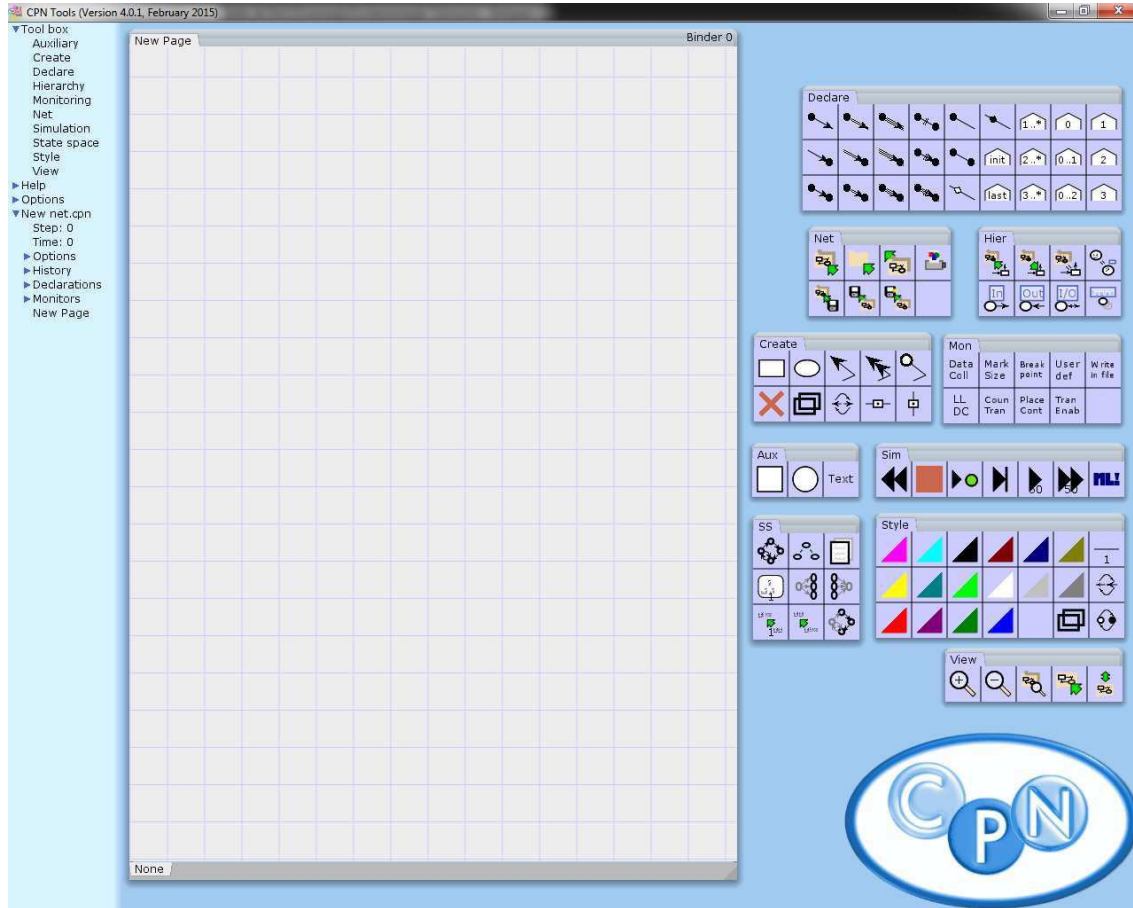


Figura 8 – Versão 4.0.1 do CPN Tools.

2.5 Redes de Petri e Lógica Linear

A Lógica Linear foi proposta por Jean-Yves Girard em 1987 em sua obra *Linear Logic* (GIRARD, 1987). Enquanto na Lógica Clássica as proposições possuem um valor *booleano* (*verdadeiro* ou *falso*), na Lógica Linear, as proposições são consideradas como recursos que são consumidos e produzidos em cada mudança de estado (PRADIN-CHÉZALVIEL; VALETTE; KUNZLE, 1999).

A Lógica Linear introduz sete novos conectivos divididos em três grupos (GIRARD, 1987):

- os conectivos multiplicativos: “times”(\otimes), “par”(\wp) e implicação linear(\multimap), que são versões bilineares do “e”, “ou” e “implica” respectivamente;
- os conectivos aditivos: “plus”(\oplus) e “with”($\&$), que são versões lineares do “ou” e “e” respectivamente;
- os conectivos exponenciais: “of course”(!) e “why not”(?), que possuem alguma similaridade com os modais \Box (necessariamente) e \Diamond (possivelmente) e são essenciais para preservar a força da lógica.

A tradução de uma rede de Petri em fórmulas da Lógica Linear é apresentada em

Pradin-Chézalviel, Valette e Kunzle (1999). Para representar uma rede de Petri através da Lógica Linear, apenas dois conectivos da Lógica Linear são necessários e serão considerados neste trabalho. Os dois conectivos considerados para este trabalho são:

- a) o conectivo *times*, denotado pelo símbolo \otimes , que representa a disponibilidade simultânea de recursos. Por exemplo, $X \otimes Y$ representa a disponibilidade simultânea dos recursos X e Y ;
- b) o conectivo *implicação linear*, denotado pelo símbolo \multimap , que representa uma mudança de estado. Por exemplo, $X \multimap Y$ denota que consumindo X , Y é produzido. Deve-se notar que após a produção do recurso Y , o recurso X será consumido e não estará mais disponível.

A Lógica Linear possui características como a noção de mudança de estado (PRADIN-CHÉZALVIEL; VALETTE; KUNZLE, 1999) e a manipulação de recursos (PRADIN-CHÉZALVIEL et al., 1999) que possibilitam uma aproximação com a teoria das redes de Petri. Apesar dos modelos em redes de Petri poderem ser reescritos utilizando Lógica Linear, as duas teorias não são equivalentes e cada uma delas possuem suas vantagens específicas (CHAMPAGNAT; PRADIN-CHÉZALVIEL; VALETTE, 2000), como o fato das rede de Petri permitirem o cálculo de invariantes de lugar e de transição, enquanto a Lógica Linear trata claramente cenários que envolvem a produção e consumo de recursos (SOARES, 2004).

Uma marcação M é um monômio (expressão lógica que possui apenas o conectivo “times” entre as fórmulas) em \otimes , ou seja, uma marcação representada por $M = A_1 \otimes A_2 \otimes \dots \otimes A_k$, onde A_i são nomes de lugares: se algum lugar A_k contém várias fichas (n , por exemplo), n instâncias do átomo A_k devem aparecer no monômio (RIVIERE; PRADIN-CHÉZALVIEL; VALETTE, 2001). Por exemplo, considerando as redes das Figuras 9(a) e 9(c), tem-se a marcação inicial $M = p1$. Já a marcação inicial da rede de Petri da Figura 9(b) é dada por $M = p1 \otimes p2$.

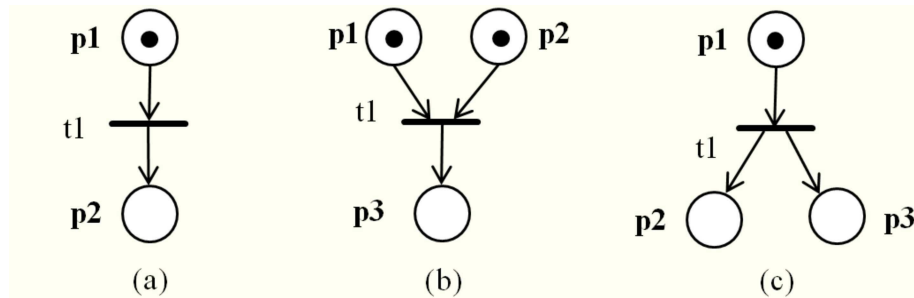


Figura 9 – Redes de Petri para exemplificação da tradução de redes de Petri em fórmulas da Lógica Linear.

Uma transição é uma fórmula $M_1 \multimap M_2$ onde M_1 e M_2 são marcações. Esta expressão representa o disparo da transição: ela aparecerá em um sequente o número de vezes em que esta transição for disparada (RIVIERE; PRADIN-CHÉZALVIEL; VALETTE, 2001).

Por exemplo, considerando a transição $t1$ da rede de Petri apresentada na Figura 9(a), tem-se que $t1 = p1 \multimap p2$. Para a rede da Figura 9(b), tem-se que $t1 = p1 \otimes p2 \multimap p3$. Já para a rede da Figura 9(c), tem-se que $t1 = p1 \multimap p2 \otimes p3$.

Um sequente $M, t_i \vdash M'$ representa um cenário onde M e M' são, respectivamente, a marcação inicial e final e t_i é uma lista de transições não ordenadas (JULIA; SOARES, 2003). Por exemplo, considerando a rede de Petri da Figura 9(a) e sua marcação final $p2$, tem-se o seguinte sequente linear $p1, p1 \multimap p2 \vdash p2$. Para a rede da Figura 9(b), considerando a marcação final $p3$, tem-se o sequente $p1 \otimes p2, p1 \otimes p2 \multimap p3 \vdash p3$. Considerando a rede da Figura 9(c) e a marcação final $p2 \otimes p3$, tem-se o sequente $p1, p1 \multimap p2 \otimes p3 \vdash p2 \otimes p3$.

É importante destacar que o sistema de dedução linear é similar ao sistema de dedução de Gentzen, proposto em 1934 (GOCHET; GRIBOMONT, 1992). Um sequente linear tem a forma $\Gamma \vdash \Delta$, onde Γ e Δ são conjuntos finitos de fórmulas, isto é, $\Gamma = \Gamma_1, \Gamma_2, \dots, \Gamma_n$ e $\Delta = \Delta_1, \Delta_2, \dots, \Delta_n$. O símbolo Γ é o antecedente da fórmula e o símbolo Δ o consequente.

Um sequente pode ser provado através de aplicações sucessivas de regras do cálculo de sequentes. De acordo com Girault, Pradier-Chézalviel e Valette (1997), há equivalência entre a prova de sequentes da Lógica Linear e o problema de alcançabilidade em uma rede de Petri.

Uma árvore de prova da Lógica Linear é construída para provar se um dado sequente linear é ou não sintaticamente correto (DIAZ, 2009). A árvore de prova é lida de baixo para cima (*bottom-up*) e termina quando todas as folhas forem sequentes identidade (sequentes do tipo $X \vdash X$), considerando os casos em que o sequente for sintaticamente correto (DIAZ, 2009).

O presente trabalho considera apenas as regras da Lógica Linear que são necessárias para representar uma rede de Petri através da Lógica Linear. Assim, somente três regras serão explicadas e terão suas aplicações exemplificadas. As demais regras do cálculo de sequentes da Lógica Linear podem ser encontradas em Girard (1987) e em Girard (1995). Inicialmente, considere que F , G e H são fórmulas e que Γ e Δ são blocos de fórmulas da Lógica Linear. As regras a seguir, apresentadas em Riviere, Pradin-Chézalviel e Valette (2001), são as regras utilizadas nos métodos propostos nesta pesquisa:

- a) a regra \multimap_L expressa um disparo de transição e gera dois sequentes. O sequente da direita representa o subsequente restante a ser provado. O sequente da esquerda representa as fichas consumidas pelo disparo da transição. A regra \multimap_L é dada por:

$$\frac{\Gamma \vdash F \quad \Delta, G \vdash H}{\Gamma, \Delta, F \multimap G \vdash H} \multimap_L$$

Por exemplo, considerando o disparo da transição $t_1 = i \multimap p1 \otimes p2$ da rede de Petri apresentada na Figura 5 apresentada na subseção 2.2.1, dois sequentes são gerados: $i \vdash i$, representando as fichas consumidas por esse disparo, e o subsequente remanescente, do qual a marcação $p1 \otimes p2$ fará parte, ou seja:

$$\frac{i \vdash i \quad p1 \otimes p2, p1 \multimap p3, t_3, t_4, t_5, t_6, t_7 \vdash o}{i, i \multimap p1 \otimes p2, t_2, t_3, t_4, t_5, t_6, t_7 \vdash o} \multimap_L$$

- b) a regra \otimes_L é usada para transformar uma marcação em uma lista de átomos. Essa regra é dada por:

$$\frac{\Gamma, F, G \vdash H}{\Gamma, F \otimes G \vdash H} \otimes_L$$

Por exemplo, a marcação $p1 \otimes p2$ gerada pelo disparo da transição $t_1 = i \multimap p1 \otimes p2$ da rede de Petri apresentada na Figura 5 utilizará a regra \otimes_L para ser transformada em uma lista de átomos $p1, p2$, ou seja:

$$\frac{p1, p2, t_2, t_3, t_4, t_5, t_6, t_7 \vdash o}{p1 \otimes p2, t_2, t_3, t_4, t_5, t_6, t_7 \vdash o} \otimes_L$$

- c) a regra \otimes_R transforma um sequente como $A, B \vdash A \otimes B$ em dois sequentes identidade $A \vdash A$ e $B \vdash B$: assim, aplicando tais regras, folhas finais são atingidas para o ramo considerado. A regra \otimes_R é dada por:

$$\frac{\Gamma \vdash F \quad \Delta \vdash G}{\Gamma, \Delta \vdash F \otimes G} \otimes_R$$

Por exemplo, considerando o disparo da transição $t4 = p3 \otimes p4 \multimap p5$ da rede de Petri da Figura 5, o sequente que representa as fichas consumidas pelo disparo desta transição, $p3, p4 \vdash p3 \otimes p4$, deverá ser provado utilizando a regra \otimes_R , ou seja:

$$\frac{p3 \vdash p3 \quad p4 \vdash p4}{p3, p4 \vdash p3 \otimes p4} \otimes_R$$

Para a exemplificação da construção de uma árvore de prova canônica da Lógica Linear, considere a rede de Petri da Figura 5. Transformando as transições desta rede de Petri em fórmulas da Lógica Linear, tem-se que:

$$\begin{aligned} t1 &= i \multimap p1 \otimes p2, \\ t2 &= p1 \multimap p3, \\ t3 &= p2 \multimap p4, \\ t4 &= p3 \otimes p4 \multimap p5, \\ t5 &= p5 \multimap p6, \\ t6 &= p5 \multimap p6, \\ t7 &= p6 \multimap o. \end{aligned}$$

Segundo Passos e Julia (2013), um cenário no contexto das *WF-nets* corresponde a um caminho bem definido no processo de *workflow*. Se a *WF-net* tem mais de um caminho (lugares com dois ou mais arcos de saída), é necessário então construir um sequente da Lógica Linear diferente para cada cenário existente (PASSOS; JULIA, 2013). Cada um desses cenários é então representado por um sequente da Lógica Linear específico que considera as marcações iniciais e finais da *WF-net* e uma lista não ordenada de transições envolvidas nelas (PASSOS; JULIA, 2013).

Considerando a rede de Petri da Figura 5 pode-se observar dois arcos saindo do lugar $p5$; conseqüentemente pode-se concluir que existem dois caminhos, dois cenários diferentes e dois sequentes da Lógica Linear para serem provados. Assim, os dois sequentes linear a serem provados são dados por:

$$SC_1 = i, t1, t2, t3, t4, t5, t7 \vdash o,$$

$$SC_2 = i, t1, t2, t3, t4, t6, t7 \vdash o.$$

sendo i e o , respectivamente, a marcação inicial e final da rede de Petri apresentada no exemplo para ambos os cenários.

Aplicando as regras do cálculo de sequentes ao sequente linear do primeiro cenário, é possível provar se o mesmo é ou não um sequente sintaticamente válido. A árvore de prova do cenário SC_1 é mostrada na sequência.

$$\begin{array}{c}
 \frac{p6 \vdash p6 \quad o \vdash o}{\multimap_L} \\
 \frac{p5 \vdash p5 \quad p6, p6 \multimap o \vdash o}{\multimap_L} \\
 \frac{\frac{p3 \vdash p3 \quad p4 \vdash p4}{p3, p4 \vdash p3 \otimes p4} \otimes_R \quad p5, p5 \multimap p6, t7 \vdash o}{\multimap_L} \\
 \frac{p2 \vdash p2 \quad p3, p4, p3 \otimes p4 \multimap p5, t5, t7 \vdash o}{\multimap_L} \\
 \frac{p1 \vdash p1 \quad p3, p4, p2 \multimap p4, t4, t5, t7 \vdash o}{\multimap_L} \\
 \frac{p1, p2, p1 \multimap p3, t3, t4, t5, t7 \vdash o}{\otimes_L} \\
 \frac{i \vdash i \quad p1 \otimes p2, p1 \multimap p3, t3, t4, t5, t7 \vdash o}{\multimap_L} \\
 i, i \multimap p1 \otimes p2, t2, t3, t4, t5, t7 \vdash o
 \end{array}$$

Todos os nós folha da árvore de prova acima são sequentes identidade, ou seja, sequentes do tipo $A \vdash A$. Dessa forma, tem-se que o sequente linear $i, t1, t2, t3, t4, t5, t7 \vdash o$, que representa o primeiro cenário da rede de Petri apresentada na Figura 5, é sintaticamente válido.

Deve-se provar para o sequente $i, t1, t2, t3, t4, t6, t7 \vdash o$, referente ao segundo cenário da rede de Petri da Figura 5, se ele também é um sequente sintaticamente válido. A árvore de prova construída para a prova do sequente do cenário SC_2 é apresentada no Apêndice A.1 deste trabalho.

Como todos os nós folha da árvore de prova do segundo cenário são sequentes identidade, tem-se através da árvore de prova do cenário SC_2 que o sequente desse cenário

também é sintaticamente válido.

No contexto das redes de Petri t-temporais, em uma árvore de prova da Lógica Linear, cada disparo de transição pode gerar uma data simbólica associada a cada átomo (*token*), como mostram Riviere, Pradin-Chézalviel e Valette (2001).

A importância da derivação de datas simbólicas em vez de datas numéricas se dá pela sua flexibilidade, sendo que elas podem ser utilizadas para o cálculo de quaisquer datas numéricas que venham a ser associadas às transições (PASSOS, 2016). Dessa forma, uma vez calculadas, estas podem ser reutilizadas quando os valores numéricos associados às transições são alterados.

No presente trabalho, D_i denota uma data e d_i uma duração associada a um disparo de uma transição (t_i). Um par (D_p, D_c) será associado a cada átomo da árvore de prova, tal que D_p e D_c representam, respectivamente, as datas de produção e consumo de um átomo. O cálculo de datas em árvores de prova canônica é dado pelos seguintes passos (RIVIERE; PRADIN-CHÉZALVIEL; VALETTE, 2001):

- determinar uma data de produção D_i para todas as fichas da marcação inicial;
- para cada instância da regra \multimap_L , calcule a data de disparo desta transição: isto é igual ao maior valor de data de produção dos átomos consumidos por esta transição, acrescido pela duração de sensibilização d_i associada à transição considerada;
- atualizar as datas de todos os átomos que foram consumidos e produzidos.

A Figura 3 apresenta uma rede de Petri t-temporal correspondente à *WF-net* da Figura 5. Lembrando que essa rede de Petri possui dois cenários, dessa forma é necessário construir uma árvore de prova canônica da Lógica Linear com cálculo de datas para cada árvores correspondente a cada cenário do modelo. Considerando o cenário SC_1 da rede de Petri da Figura 3, as fórmulas da Lógica Linear para cada uma de suas transições (definidas anteriormente nessa seção), sua marcação inicial i e que $Seq = D_i + d_1 + \max(d_2, d_3) + d_4$, tem-se a seguinte árvore de prova com cálculo de datas:

$$\begin{array}{c}
\frac{p6(Seq+\max(d5,d6),Seq+\max(d5,d6)+d7)\vdash p6 \quad o(Seq+\max(d5,d6)+d7,.)\vdash o}{\multimap_L} \\
\frac{p5(Seq,Seq+\max(d5,d6))\vdash p5 \quad p6(Seq+\max(d5,d6),.)\vdash p6\multimap o\vdash o}{\multimap_L} \\
\frac{\frac{p3(Di+d1+d2,Seq)\vdash p3 \quad p4(Di+d1+d3,Seq)\vdash p4}{p3(Di+d1+d2,Seq),p4(Di+d1+d3,Seq)\vdash p3\otimes p4} \otimes_R \quad p5(Seq,.)\vdash p5\multimap p6,t7\vdash o}{\multimap_L} \\
\frac{p2(Di+d1+,Di+d1+d3)\vdash p2 \quad p3(Di+d1+d2,.)\vdash p3 \quad p4(Di+d1+d3,.)\vdash p4\otimes p5,t5,t7\vdash o}{\multimap_L} \\
\frac{p1(Di+d1+,Di+d1+d2)\vdash p1 \quad p2(Di+d1,.)\vdash p2 \quad p3(Di+d1+d2,.)\vdash p3\multimap p4,t4,t5,t7\vdash o}{\multimap_L} \\
\frac{p1(Di+d1,.)\vdash p1 \quad p2(Di+d1,.)\vdash p2\multimap p3,t3,t4,t5,t7\vdash o}{\otimes_L} \\
\frac{i(Di,Di+d1)\vdash i \quad p1(Di+d1,.)\otimes p2(Di+d1,.)\vdash p1\multimap p3,t3,t4,t5,t7\vdash o}{\multimap_L} \\
i(Di,.)\vdash i\multimap p1\otimes p2,t2,t3,t4,t5,t7\vdash o
\end{array}$$

A árvore de prova canônica da Lógica Linear com cálculo de datas construída para a prova do sequente do cenário SC_2 é apresentada no Apêndice A.1 deste trabalho.

Tabela 1 – Datas simbólicas de produção e consumo dos átomos da rede de Petri t-temporal da Figura 3.

Átomo	Data de Produção	Data de Consumo
i	D_i	$D_i + d_1$
$p1$	$D_i + d_1$	$D_i + d_1 + d_2$
$p2$	$D_i + d_1$	$D_i + d_1 + d_3$
$p3$	$D_i + d_1 + d_2$	Seq
$p4$	$D_i + d_1 + d_3$	Seq
$p5$	Seq	$Seq + \max(d_5, d_6)$
$p6$	$Seq + \max(d_5, d_6)$	$Seq + \max(d_5, d_6) + d_7$
o	$Seq + \max(d_5, d_6) + d_7$	<i>desconhecido</i>

A Tabela 1 mostra as datas simbólicas de produção e consumo de cada átomo da rede de Petri t-temporal da Figura 3.

Em um modelo de rede de Petri t-temporal, toda duração de sensibilização d_i de uma transição t_i tem um valor que pertence a um intervalo de tempo $\Delta_i = [\delta_{imin}, \delta_{imax}]$. Logo, uma vez que as datas simbólicas computadas dependem de d_i , seus domínios também serão em função de intervalos de tempo. A Tabela 2 mostra os intervalos de datas simbólicas de produção e consumo de cada átomo da rede de Petri t-temporal da Figura 3, considerando que $Seq_{min} = D_i + d_{1min} + \max(d_{2min}, d_{3min}) + d_{4min}$ e $Seq_{max} = D_i + d_{1max} + \max(d_{2max}, d_{3max}) + d_{4max}$.

Tabela 2 – Intervalos de datas simbólicas de produção e consumo dos átomos da rede de Petri t-temporal da Figura 3.

Átomo	Data de Produção	Data de Consumo
i	D_i	$[D_i + d_{1min}, D_i + d_{1max}]$
$p1$	$[D_i + d_{1min}, D_i + d_{1max}]$	$[D_i + d_{1min} + d_{2min}, D_i + d_{1max} + d_{2max}]$
$p2$	$[D_i + d_{1min}, D_i + d_{1max}]$	$[D_i + d_{1min} + d_{3min}, D_i + d_{1max} + d_{3max}]$
$p3$	$[D_i + d_{1min} + d_{2min}, D_i + d_{1max} + d_{2max}]$	$[Seq_{min}, Seq_{max}]$
$p4$	$[D_i + d_{1min} + d_{3min}, D_i + d_{1max} + d_{3max}]$	$[Seq_{min}, Seq_{max}]$
$p5$	$[Seq_{min}, Seq_{max}]$	$[Seq_{min} + \max(d_{5min}, d_{6min}), Seq_{max} + \max(d_{5max}, d_{6max})]$
$p6$	$[Seq_{min} + \max(d_{5min}, d_{6min}), Seq_{max} + \max(d_{5max}, d_{6max})]$	$[Seq_{min} + \max(d_{5min}, d_{6min}) + d_{7min}, Seq_{max} + \max(d_{5max}, d_{6max}) + d_{7max}]$
o	$[Seq_{min} + \max(d_{5min}, d_{6min}) + d_{7min}, Seq_{max} + \max(d_{5max}, d_{6max}) + d_{7max}]$	<i>desconhecido</i>

Considerando os intervalos de datas numéricas, apresentados na Figura 3, e os intervalos de datas simbólicas, apresentados na Tabela 2, pode-se substituir as datas simbólicas pelas numéricas, obtendo datas numéricas de produção e consumo de cada átomo da rede de Petri t-temporal da Figura 3. O resultado obtido, considerando que $D_i = 0$, é apresentado na Tabela 3.

Tabela 3 – Intervalos de datas numéricas de produção e consumo dos átomos da rede de Petri t-temporal da Figura 3.

Átomo	Data de Produção	Data de Consumo
i	0	[1,3]
$p1$	[1,3]	[4,12]
$p2$	[1,3]	[5,10]
$p3$	[4,12]	[6,17]
$p4$	[5,10]	[6,17]
$p5$	[6,17]	[9,23]
$p6$	[9,23]	[10,27]
o	[10,27]	<i>desconhecido</i>

2.6 Deadlock em redes de Petri

O assunto *deadlock* tem sido tema de uma série de estudos em Ciência da Computação (HEBALKAR, 1970). Em um ambiente onde existem vários processos, esses processos podem competir entre si por um número finito de recursos. Um processo solicita um recurso e se ele não está disponível naquele momento, o processo entra em um estado de espera. Podem existir cenários onde esse processo de espera não consegue mais alterar o seu estado; esta situação é chamada de *deadlock* (GALVIN; GAGNE; SILBERSCHATZ, 2013).

No contexto dos processos de *workflows*, a situação de *deadlock* é causada por um erro de *design* no modelo de *workflow* (MARUTA et al., 1998), podendo ocorrer em virtude da competição entre os processos concorrentes pela utilização de recursos compartilhados ou por falhas de sincronização de comunicação entre esses processos concorrentes (SILVA, 2014). A definição formal de *deadlock* no contexto das redes de Petri é dada a seguir (BANASZAK; KROGH, 1990).

Definição 7. (*Deadlock*) Uma rede de Petri $PN = (P, T, F)$ está em situação de *deadlock* para uma dada marcação M se, e somente se, nenhuma transição (ou um conjunto de transições) pode ser disparada. Formalmente,

$$\forall t \in T, (M \not\stackrel{t}{\rightarrow}).$$

Relacionado à propriedade da vivacidade, um sifão, também conhecido como *deadlock* estrutural, é o responsável pela presença de situações de *deadlock* em sistemas modelados pelas redes de Petri (BARKAOUI; ABDALLAH, 1995). A definição de sifão é apresentada formalmente na sequência (MURATA, 1989).

Definição 8. (Sifão) Seja $PN = (P, T, F)$ uma rede de Petri, um sifão é um conjunto de lugares S não vazio de $P (S \subset P)$ tal que $\bullet S \subseteq S\bullet$.

De uma forma simplificada, um sifão é um conjunto de lugares S , tal que o conjunto de transições de entrada de S (representado formalmente por $\bullet S$) esteja contido no conjunto de transições de saída de S (representado formalmente por $S\bullet$) (MURATA, 1989). Como

existem mais saídas de fichas que entradas, o conjunto de lugares S pode ficar livre de fichas, com todos os lugares totalmente desmarcados (vazios), podendo provocar uma situação de *deadlock* (JÚNIOR, 2003).

As definições formais de sifão básico e sifão mínimo são descritas a seguir (MURATA, 1989).

Definição 9. (Sifão Básico) Um sifão S é dito básico se ele não puder ser expresso pela união de outros sifões.

Definição 10. (Sifão Mínimo) Um sifão S é dito mínimo se, e somente se, nenhum outro sifão está contido nele como um subconjunto próprio.

Para que não ocorra o completo esvaziamento de fichas em um sifão, ele deve conter pelo menos uma *trap*. A definição formal de *trap* é apresentada na sequência (MURATA, 1989).

Definição 11. (Trap) Seja $PN = (P, T, F)$ uma rede de Petri, uma *trap* é um conjunto de lugares Q não vazio de $P(Q \subset P)$ tal que $Q \bullet \subseteq \bullet Q$.

Em outras palavras, uma *trap* é um conjunto de lugares Q , tal que o conjunto de transições de saída de Q (representado formalmente por $Q \bullet$) esteja contido no conjunto de transições de entrada de Q (representado formalmente por $\bullet Q$) (MURATA, 1989). Isto significa que se algum dos lugares da *trap* tem uma ficha, então sempre existirá uma ficha em algum dos lugares da *trap*, ou seja, o disparo das transições podem movimentá-las nos lugares da *trap* mas não podem removê-las.

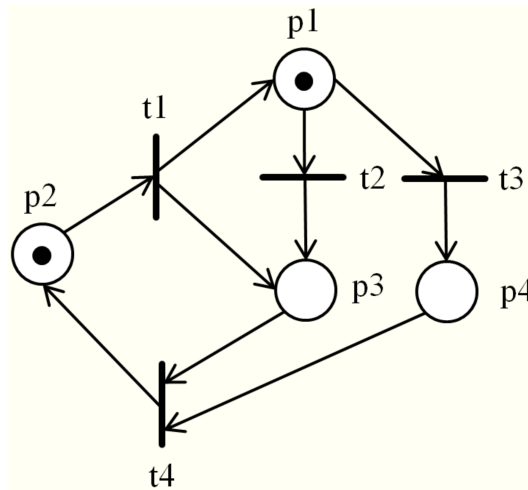


Figura 10 – Exemplo de rede de Petri com sifão.

Uma condição necessária para a existência de uma situação de *deadlock* em uma rede de Petri é a presença de pelo menos um sifão vazio quando se considera o conjunto das marcações acessíveis (IORDACHE; MOODY; ANTSAKLIS, 2002). Além disto, uma

condição necessária e suficiente para a vivacidade de uma rede de Petri marcada existe se todos os sífios mínimos da rede possuem uma *trap* suficientemente marcada, garantindo assim que nenhuma situação de *deadlock* ocorrerá (HACK, 1972). Para exemplificar a presença de sífios e *traps* em uma rede de Petri, será considerada a rede de Petri apresentada na Figura 10.

Seja S_1 , S_2 , S_3 , S_4 e S_5 , conjuntos de lugares da Figura 10 com os respectivos valores $S_1 = \{p_1, p_2, p_3\}$, $S_2 = \{p_1, p_2, p_4\}$, $S_3 = \{p_1, p_2, p_3, p_4\}$, $S_4 = \{p_2, p_3\}$ e $S_5 = \{p_2, p_3, p_4\}$. Tem-se que $\bullet S_1 = \{t_1, t_2, t_4\} \subseteq S_1 \bullet = \{t_1, t_2, t_3, t_4\}$. Como S_1 é um conjunto de lugares, tal que o conjunto de transições de entrada de S_1 está contido no conjunto de transições de saída de S_1 , então esse conjunto é um sífio. Uma vez que $S_4 \bullet = \{t_1, t_4\} \subseteq \bullet S_4 = \{t_1, t_2, t_4\}$, ou seja, o conjunto de transições de saída de S_4 está contido no conjunto de transições de entrada de S_4 , logo S_4 é uma *trap*. Similarmente, é fácil verificar que S_2 é um sífio, S_3 é tanto um sífio quanto uma *trap* e S_5 é uma *trap*. De fato, tanto S_1 quanto S_2 não podem ser expressos pela união de outros sífios e nenhum outro sífio está contido nesses conjuntos como um subconjunto próprio; então S_1 e S_2 são sífios básicos e mínimos.

Trabalhos Relacionados

Neste capítulo são apresentados os trabalhos relacionados que utilizam as redes de Petri para resolver o problema de *deadlock* em processos de *workflow* interorganizacionais, ou seja, os principais trabalhos correlatos à abordagem proposta nesta dissertação.

Em Aalst (1998) é apresentada a definição de IOWF-*net* e junto com ela o problema de *deadlock* que comumente é encontrado em *workflows* interorganizacionais. Para o exemplo de *workflow* interorganizacional apresentado em Aalst (1998) é acrescentada uma nova transição com o objetivo de remover o problema de *deadlock* encontrado nos elementos de modelagem que expressam a criação da comunicação entre as *WorkFlow nets* locais. Nesta pesquisa é utilizada a definição de IOWF-*net* apresentada em Aalst (1998), além dos conceitos de tipos de comunicação e a indicação da necessidade de um tipo de comunicação síncrona para remover problemas de *deadlock* gerados por lugares de comunicação assíncrona.

Referente à detecção e correção de situações de *deadlock* em IOWF-*nets*, Silva et al. (2013) apresenta um procedimento sistemático em seu trabalho, seguido de uma proposta de abordagem para uma arquitetura distribuída livre de *deadlock*; porém essa abordagem consiste na inclusão de novos lugares na rede podendo aparecer outras situações de *deadlock*. Uma nova análise da estrutura e das propriedades precisa ser feita para garantir que a inclusão destes lugares não gera novos *deadlocks*. Dessa forma, a abordagem pode alterar o modelo original, aumentando indefinidamente a quantidade de lugares e tornando-o grande e mais complexo para análise. Na presente pesquisa, pretende-se definir uma abordagem que não deve inserir novos lugares, de forma que não sejam adicionadas novas possíveis situações de *deadlocks* e nem aumente a complexidade do modelo.

Em Passos e Julia (2014) é apresentada uma abordagem que formaliza um método baseado na Lógica Linear para detecção dos cenários livres de *deadlock* em processos modelados por IOWF-*nets*. Porém em seu trabalho, Passos e Julia (2014) não identificam nesses cenários quais as transições e lugares de comunicação são responsáveis pela situação de *deadlock*. Atividades da abordagem proposta em Passos e Julia (2014) como a representação dos cenários das IOWF-*nets* em sequentes da Lógica Linear e a análise das

árvores de prova da Lógica Linear também são utilizadas nesta pesquisa.

Ainda dentro do contexto das IOWF-*nets*, em Rezende e Julia (2015b) é apresentada uma abordagem baseada em estrutura de sifão e redes de Petri possibilísticas para garantir a existência de pelo menos uma sequência de disparo permitindo a finalização do processo sem se deparar com situações de *deadlock*. Cardoso (1998) define as redes de Petri possibilísticas como um modelo onde um local marcado corresponde a um possível estado parcial, uma transição a uma possível mudança de estado e uma sequência de disparo a um comportamento possível. O conceito de sifão em IOWF-*nets* pode ser utilizado de acordo com Rezende e Julia (2015b); porém não serão utilizadas as redes de Petri possibilísticas para a abordagem apresentada neste trabalho, devido ao grande número de mecanismos de propagação para frente (*forward reasoning*) que precisam ser executados para evitar a ocorrência de situações de *deadlock* e que podem tornar difícil a implementação de tal abordagem (encontrar uma sequência que finaliza corretamente o processo) em tempo real.

Em Mugarza et al. (2001) é apresentada uma proposta de aplicação de um método de redução não clássico em uma rede de Petri baseado no cálculo dos seqüentes da Lógica Linear. Após a realização das reduções, as boas propriedades da rede de Petri não são preservadas; de forma que somente os cenários livres de *deadlock* são preservados. Em particular, todos os cenários que poderiam levar o sistema à situações de *deadlock* são removidos no modelo reduzido. O modelo reduzido só será equivalente ao modelo original quando são considerados cenários livres de *deadlock*. Ao aplicar as regras de redução no modelo, a rede de Petri não será mais temporizada nas transições e sim nos arcos de saída das transições. Esta regra de redução entretanto somente é utilizada em processos puramente sequenciais (processos sem caminhos paralelos e alternativos), não sendo aplicada especificamente às WF-*nets* e IOWF-*nets*. Essa pesquisa serviu como inspiração para a criação da abordagem apresentada nesta dissertação. Na abordagem proposta nesta dissertação, uma regra de sincronização deverá forçar os processos de *workflow* locais à executarem tarefas específicas, respeitando uma política de escalonamento local com o objetivo de remover os cenários responsáveis pelos *deadlocks*.

Regra de Sincronização para prevenção de *deadlock*

O presente capítulo apresenta uma abordagem para remoção de situações de *deadlock* em processos de *workflow* interorganizacionais modelados pelas *WorkFlow nets* Interorganizacionais que são localmente, mas não globalmente *sound*. A seção 4.1 apresenta o método para aplicação da regra de sincronização para remover situações de *deadlock* em *WorkFlow nets* Interorganizacionais. A seção 4.2 apresenta a análise qualitativa antes e depois da aplicação da regra de sincronização, preocupando-se com o critério de correção *Soundness* do modelo. Já a seção 4.3 apresenta a análise quantitativa antes e depois da aplicação da regra apresentada neste trabalho, preocupando-se com o desempenho do tempo de execução do modelo através do cálculo de datas simbólicas das árvores de prova da Lógica Linear.

4.1 Aplicação da Regra de Sincronização

No contexto das *WorkFlow nets* Interorganizacionais, as situações de *deadlock* se originam da troca de mensagens, como mostrado em Xiong, Zhou e Pu (2009), e são introduzidas por elementos de comunicação assíncrona (AALST, 1998). De acordo com Aalst (1998), uma situação de *deadlock* pode ser corrigida pela substituição de um elemento de comunicação assíncrona (um lugar de comunicação) por um elemento de comunicação síncrona (uma transição de sincronização).

A abordagem proposta nesta pesquisa consiste na substituição de lugares de comunicação assíncrona (aqueles responsáveis pelas situações de *deadlock*) em uma IOWF-net por novos mecanismos de comunicação parcialmente síncronos com o objetivo de prevenir a ocorrência de mensagens perdidas que, em caso de situações de *deadlock*, ficam presas em lugares de comunicação, impedindo assim que o modelo do processo de *workflow* respeite a propriedade *Soundness*. Tal substituição será um tipo de regra de sincronização.

Para aplicar tal regra de sincronização, será necessário primeiro identificar dentro

dos potenciais cenários do modelo, quais são os responsáveis pelas situações de *deadlock*. Mais especificamente será necessário identificar os elementos de comunicação assíncrona responsáveis por levar o sistema a estados de *deadlock* (estados terminais com fichas presas em um ou mais lugares de comunicação assíncrona).

Para a abordagem apresentada, deverá ser considerado o respectivo modelo *unfolded* de uma *IOWF-net* ($U(IOWF-net)$) que será analisado para detecção do *deadlock*.

A detecção de estados inconsistentes neste trabalho será baseada na análise dos sequentes das árvores de prova da Lógica Linear que correspondem a cenários potenciais de uma $U(IOWF-net)$. Será necessário que os elementos da $U(IOWF-net)$ sejam representados através do uso de fórmulas da Lógica Linear. Em particular, para cada cenário potencial da $U(IOWF-net)$, um sequente da Lógica Linear será então produzido. O método detalhado para obtenção de todos os sequentes candidatos da Lógica Linear para possíveis colaborações entre dois ou mais processos de *workflow* é apresentado em Passos e Julia (2013).

Depois da definição dos sequentes da Lógica Linear que representarão todos os possíveis cenários da $U(IOWF-net)$, as árvores de prova da Lógica Linear, que correspondem aos sequentes identificados, serão produzidas e analisadas. Caso o último sequente da árvore de prova da Lógica Linear construída para um cenário específico seja diferente do sequente identidade $o \vdash o$, então será identificada uma situação de *deadlock* na $U(IOWF-net)$ e consequentemente o cenário não será *sound*.

Após identificação de um cenário que não é *sound*, será necessário identificar a causa do *deadlock*, isto é, a última transição disparada antes da produção da marcação final que corresponde ao estado de *deadlock*. No caso de uma $U(IOWF-net)$, devido a um problema estrutural, em particular na parte de comunicação que relaciona dois processos locais distintos, a última transição disparada vai produzir um átomo específico na árvore de prova que não será consumido até o final da prova. Tal transição será chamada nessa abordagem de t_{d1} . Para identificar tal transição, será necessário verificar na árvore de prova, o disparo da transição que produzirá o átomo que permanecerá em um dos lugares de comunicação (um lugar pertencente à P_{AC}) até a conclusão da prova.

Para cada transição do tipo t_{d1} , uma transição correspondente do tipo t_{d2} vai também existir. Tal transição, quando a situação de *deadlock* ocorre, corresponde então à transição de saída do lugar de comunicação que contém a ficha presa, caracterizando o estado de *deadlock*. Tal lugar de comunicação será chamado nessa abordagem de cp e corresponde ao lugar de saída da transição de tipo t_{d1} . O mesmo lugar corresponde também ao lugar de entrada da transição de tipo t_{d2} . O lugar cp terá o seu respectivo átomo produzido na árvore de prova pelo disparo da transição do tipo t_{d1} , porém tal átomo não será consumido até o final da prova e poderá ser visto no último sequente da árvore. É então a transição morta de tipo t_{d2} que caracterizará a situação de *deadlock* no modelo e o não respeito da propriedade *Soundness* da $U(IOWF-net)$.

Como o objetivo da abordagem é prevenir a morte da transição t_{d2} , é necessário introduzir um tipo de regra de sincronização que corresponde à seguinte estratégia: cada vez que uma transição do tipo t_{d1} é disparada, a transição correspondente do tipo t_{d2} tem que ser disparada na sequência com o objetivo de esvaziar o lugar de comunicação no qual o átomo foi produzido. Na prática, tal política corresponde em garantir que para cada mensagem (a tarefa associada à transição t_{d1}) enviada por um processo local, a mensagem correspondente do outro processo local (a tarefa associada à transição t_{d2}) vai certamente ser recebida.

Para apresentar a aplicação da regra de sincronização, considere que $p1$ é um lugar de entrada da transição do tipo t_{d1} pertencente à LWF-net $L1$, $p2$ um lugar de entrada da transição do tipo t_{d2} pertencente à LWF-net $L2$, $p3$ um lugar de saída da transição do tipo t_{d1} pertencente à LWF-net $L1$, $p4$ um lugar de saída da transição do tipo t_{d2} pertencente à LWF-net $L2$ e cp o lugar de comunicação assíncrona que estabelece a comunicação entre os *workflows* locais $L1$ e $L2$. A Figura 11A representa a estrutura genérica onde a regra de sincronização será aplicada em caso de *deadlock*.

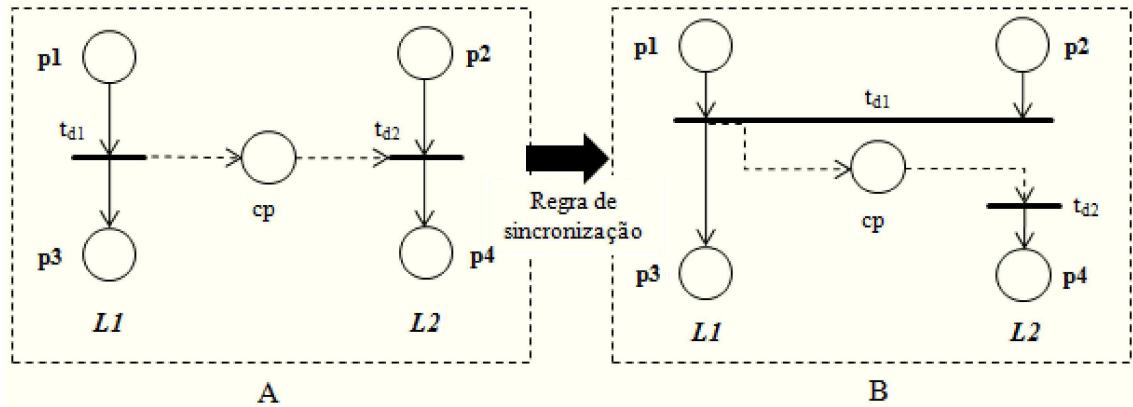


Figura 11 – Antes e depois da regra de sincronização.

As transições apresentadas na Figura 11A, que representa o modelo de comunicação antes da aplicação da regra de sincronização, podem ser representadas pelas seguintes fórmulas da Lógica Linear:

$$t_{d1} = p1 \multimap p3 \otimes cp,$$

$$t_{d2} = cp \otimes p2 \multimap p4.$$

A fórmula da transição t_{d1} mostra claramente que para produzir os átomos $p3$ e cp , somente o átomo $p1$ é necessário, o que não garante que para o próximo disparo da transição t_{d2} , transição dependente dos átomos cp e $p2$, que o átomo $p2$ estará sempre disponível. A solução é então exigir que ao disparar t_{d1} , tanto o átomo em $p1$ quanto o átomo em $p2$ estejam disponíveis para garantir que a próxima atividade t_{d2} aconteça. Isso se traduz em Lógica Linear pela fórmula $p1 \otimes p2 \multimap p3 \otimes cp$. Tal alteração é diretamente expressa na Figura 11B e corresponde então à sincronização parcial do mecanismo de comunicação assíncrona apresentado na Figura 11A, garantindo que ao sincronizar a transição do tipo

t_{d1} nas duas LWF-nets $L1$ e $L2$, a transição do tipo t_{d2} será obrigatoriamente executada na sequência.

Tal sincronização da transição específica, em particular, o fato de que a atividade associada à transição t_{d1} (pertencente à LWF-net $L1$) só poderá ser iniciada caso existam fichas em espera tanto em $p1$ quanto em $p2$. Antes da aplicação da regra de sincronização, a marcação no lugar $p1$ era condição suficiente para o disparo de t_{d1} , o que não garante na sequência o disparo da transição t_{d2} , caso nenhuma ficha esteja disponível no lugar $p2$ até a finalização da execução da LWF-net $L2$. Dessa forma, após a aplicação da regra, as transições alteradas no modelo podem ser redefinidas nas seguintes fórmulas da Lógica Linear:

$$t_{d1} = p1 \otimes p2 \multimap p3 \otimes cp,$$

$$t_{d2} = cp \multimap p4.$$

Antes da aplicação da regra de sincronização, a transição do tipo t_{d2} do modelo era sensibilizada somente com a presença de uma ficha em seu lugar de entrada ($p1$), porém para disparar a transição seguinte (transição do tipo t_{d2}), a transição precisava de uma ficha no lugar de entrada da LWF-net $L2$ ($p2$) e outra ficha no lugar de comunicação (cp). Ao sincronizar o modelo, os lugares de entrada da transição do tipo t_{d1} ($p1$ e $p2$) vão ter as suas fichas disparadas de forma simultânea nas LWF-nets $L1$ e $L2$, fazendo com que o disparo da transição do tipo t_{d2} dependa apenas de uma ficha no lugar de comunicação cp .

Existem casos onde o lugar de comunicação cp , pode possuir mais de um arco de entrada ou mais de um arco de saída. Nessas situações, será considerado mais de um cenário para ser tratado no modelo global.

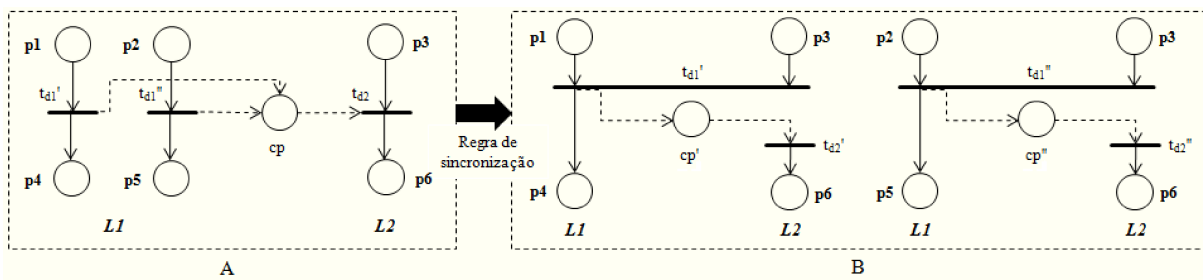


Figura 12 – Antes e depois da regra de sincronização com lugar de comunicação com mais de um arco de entrada.

A Figura 12A apresenta o modelo genérico antes da aplicação da regra quando o lugar de comunicação cp possui mais de um arco de entrada. Conforme apresentado na Figura 12B, após a aplicação da regra de sincronização, cada transição do tipo t_{d1} deverá ser sincronizada nas LWF-nets, forçando que a transição do tipo t_{d2} seja disparada na sequência. Como consequência da sincronização o lugar de comunicação cp e a transição t_{d2} deverão ser duplicados para cada arco de entrada que o lugar cp tiver.

Como existem dois arcos de entrada no lugar de comunicação cp , serão gerados dois cenários para o modelo da Figura 12. Os dois diferentes cenários e respectivos sequentes da Lógica Linear são então representados na sequência:

$$Sc_1 = p1, p3, t_{d1}', t_{d2} \vdash p4, p6,$$

$$Sc_2 = p2, p3, t_{d1}'', t_{d2} \vdash p5, p6.$$

As transições apresentadas na Figura 12A, que representam o modelo de comunicação antes da aplicação da regra de sincronização, podem ser representadas pelas seguintes fórmulas da Lógica Linear:

$$t_{d1}' = p1 \multimap p4 \otimes cp,$$

$$t_{d1}'' = p2 \multimap p5 \otimes cp,$$

$$t_{d2} = cp \otimes p3 \multimap p6.$$

Em casos onde o lugar de comunicação assíncrona cp do modelo possui mais de um arco de entrada, após a aplicação da regra de sincronização, cada lugar cp e cada transição do tipo t_{d2} será duplicado. Tais elementos serão repetidos na rede de Petri para cada cenário que será gerado pelo arco de entrada em cp . Assim, será criado um lugar de comunicação cp' e uma transição t_{d2}' para o cenário Sc_1 e um lugar de comunicação cp'' e uma transição t_{d2}'' para o cenário Sc_2 , elementos derivados respectivamente do lugar de comunicação cp e da transição t_{d2} do modelo antes da regra de sincronização. Dessa forma, após a aplicação da regra de sincronização, as transições alteradas no modelo podem ser redefinidas nas seguintes fórmulas da Lógica Linear:

$$t_{d1}' = p1 \otimes p3 \multimap p4 \otimes cp',$$

$$t_{d1}'' = p1 \otimes p3 \multimap p5 \otimes cp'',$$

$$t_{d2}' = cp' \multimap p6,$$

$$t_{d2}'' = cp'' \multimap p6.$$

Dessa forma, é possível concluir que para cada arco de entrada no lugar de comunicação cp , após a aplicação da regra de sincronização, um cenário será gerado com um lugar de comunicação cp e uma transição t_{d2} , derivados do modelo antes da aplicação da regra de sincronização.

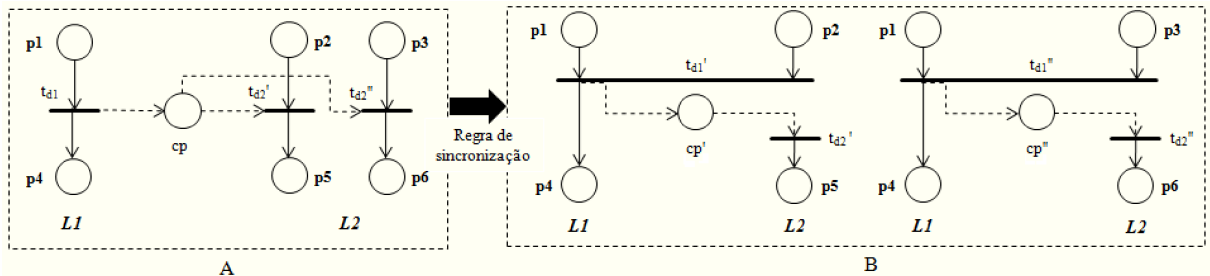


Figura 13 – Antes e depois da regra de sincronização com lugar de comunicação com mais de um arco de saída.

A Figura 13A apresenta um modelo genérico antes da aplicação da regra quando o

lugar de comunicação cp possui mais de um arco de saída. Conforme apresentado na Figura 12B, após a aplicação da regra de sincronização, a transição do tipo t_{d1} deverá ser sincronizada com as atividades (transições t_{d2}' e t_{d2}'') da LWF-net $L2$. Como consequência da sincronização o lugar de comunicação cp e a transição t_{d1} deverão ser duplicados para cada arco de entrada que o lugar cp tiver.

Como existem dois arcos de saída no lugar de comunicação cp , serão gerados dois cenários para o modelo da Figura 13. Os dois diferentes cenários e respectivos sequentes da Lógica Linear são então representados na sequência:

$$Sc_1 = p1, p2, t_{d1}, t_{d2}' \vdash p4, p5,$$

$$Sc_2 = p1, p3, t_{d1}, t_{d2}'' \vdash p4, p6.$$

As transições apresentadas na Figura 13A, que representa o modelo de comunicação antes da aplicação da regra de sincronização, podem ser representadas pelas seguintes fórmulas da Lógica Linear:

$$t_{d1} = p1 \multimap p4 \otimes cp,$$

$$t_{d2}' = cp \otimes p2 \multimap p5,$$

$$t_{d2}'' = cp \otimes p3 \multimap p6.$$

Em casos que o lugar de comunicação assíncrona cp do modelo possui mais de um arco de saída, após a aplicação da regra de sincronização, cada lugar cp e cada transição do tipo t_{d1} será duplicado. Tais elementos serão repetidos na rede de Petri para cada cenário que será gerado pelo arco de saída em cp . Assim, será criado um lugar de comunicação cp' e uma transição t_{d1}' para o cenário Sc_1 e um lugar de comunicação cp'' e uma transição t_{d1}'' para o cenário Sc_2 , elementos derivados respectivamente do lugar de comunicação cp e da transição t_{d1} do modelo antes da regra de sincronização. Dessa forma, após a aplicação da regra de sincronização, as transições alteradas no modelo podem ser redefinidas nas seguintes fórmulas da Lógica Linear:

$$t_{d1}' = p1 \otimes p2 \multimap p4 \otimes cp',$$

$$t_{d1}'' = p1 \otimes p3 \multimap p4 \otimes cp'',$$

$$t_{d2}' = cp' \multimap p5.$$

$$t_{d2}'' = cp'' \multimap p6.$$

É possível identificar que tanto em casos onde o lugar de comunicação cp que possui mais de um arco de entrada, como pode ser visto na Figura 12, quanto em casos onde o lugar de comunicação cp que possui mais de um arco de saída, como pode ser visto na Figura 13, a regra de sincronização apresentada para a Figura 11, onde existe apenas um arco de entrada e um arco de saída no modelo, poderá ser utilizada normalmente, apenas tendo que tratar cada arco de entrada ou saída do lugar de comunicação assíncrona cp como um cenário diferente.

Depois da aplicação da regra de sincronização, a causa do *deadlock* (uma ficha presa em um lugar de comunicação) é removida e a prova da Lógica Linear pode ser finalizada corretamente com apenas um átomo produzido no lugar o (último lugar da $U(IOWF-net)$).

Os cenários que possuem situações de *deadlock* são removidos e não podem mais ocorrer quando a execução do modelo *unfolded* é iniciado. Na prática, esses cenários são então removidos de cenários potenciais existentes quando se considera a $U(IOWF-net)$. Como, após a aplicação da regra, os cenários com situações de *deadlock* são removidos, vão existir apenas cenários potenciais que são *sound* no modelo.

Como consequência da remoção dos cenários com situações de *deadlock*, as transições do tipo td_1 e td_2 são alteradas após a aplicação da regra de sincronização. Dessa forma, é necessário representar tais transições por novas fórmulas da Lógica Linear. Como os cenários são representados através do uso dos sequentes da Lógica Linear, as fórmulas da Lógica Linear que representam as transições do tipo td_1 e td_2 são alteradas, e os cenários *sound* onde essas transições são executadas também precisam ser provados novamente.

As árvores de prova dos sequentes alterados, após a aplicação da regra de sincronização, só precisam ser reprocessadas a partir da parte modificada pela regra de sincronização, ou seja, a partir do disparo da transição do tipo td_1 .

Após a alteração no modelo das partes que foram sincronizadas, os cenários que antes da aplicação da regra de sincronização não levavam o modelo à situações de *deadlock* permanecem *sound*.

Para respeitar a definição das *WorkFlow nets* Interorganizacionais, após aplicação da regra de sincronização, para cada cenário que possuía situação de *deadlock*, um lugar adicional deverá ser adicionado ao modelo final. Tais lugares, serão lugares de saída das transições de tipo t_{d1} e lugares de entrada das transições de tipo t_{d2} . Esses lugares têm como objetivo principal respeitar a estrutura básica de uma *WorkFlow net*. Considerando a teoria das redes de Petri (MURATA, 1989), são lugares implícitos que não alteram as boas propriedades do modelo correspondente. A Figura 14 apresenta a adição do lugar auxiliar *aux* no modelo apresentado na Figura 11B para respeitar a estrutura básica de uma *WorkFlow net*.

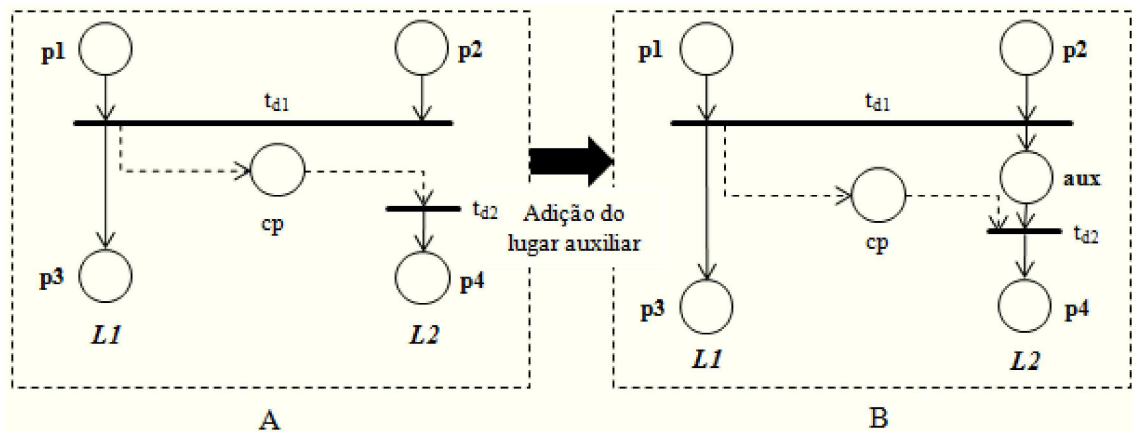


Figura 14 – Adição do lugar auxiliar *aux* no modelo da Figura 11B.

De acordo com os procedimentos anteriormente apresentados, o seguinte método para

aplicar a regra de sincronização é apresentado:

- a) criar a $U(\text{IOWF-net})$ correspondente à IOWF-net estudada;
- b) identificar os cenários existentes para a respectiva $U(\text{IOWF-net})$;
- c) produzir os sequentes da Lógica Linear correspondentes aos cenários encontrados (que consideram as marcações iniciais e finais e uma lista não ordenada de transições envolvidas neles);
- d) provar os sequentes através da construção das árvores de prova da Lógica Linear para cada cenário e identificar as possíveis situações de *deadlock*;
- e) identificar a transição do tipo t_{d1} , o lugar de comunicação cp e a transição do tipo t_{d2} , através da análise dos átomos que compõem os sequentes das árvores de prova dos cenários que possuem *deadlock*:
 - a transição t_{d1} corresponde à última transição disparada, que produz os átomos não consumidos na árvore de prova; tais átomos são remanescentes no último sequente da árvore de prova para o cenário que possui o *deadlock*;
 - o lugar cp corresponde a um lugar de comunicação da IOWF-net ($cp \in P_{AC}$) e também a um lugar de saída da transição do tipo t_{d1} . Além disso, no último sequente da árvore de prova, deverá existir um átomo que corresponde a uma ficha neste lugar de comunicação, fazendo com que a propriedade *Soundness* não seja respeitada;
 - a transição do tipo t_{d2} corresponde a uma transição morta de uma LWF-net no modelo global (diferente da LWF-net da transição de t_{d1}); tal transição não será sensibilizada por falta de recebimento da mensagem enviada por t_{d1} . Esta transição será identificada através da análise do modelo global, identificando qual transição corresponde à transição de saída do lugar de comunicação cp ;
- f) para cada cenário com situação de *deadlock*, aplicar a regra de sincronização alterando a transição do tipo t_{d1} , a transição do tipo t_{d2} e o lugar de comunicação cp no modelo global, conseqüentemente removendo a situação de *deadlock* conforme mostra esquematicamente a Figura 11;
- g) para cada sincronização introduzida no modelo, acrescentar um lugar implícito auxiliar aux_i , alterar as fórmulas da Lógica Linear correspondente às mudanças realizadas no modelo e refazer as árvores de prova a partir do disparo da transição do tipo t_{d1} para os cenários que possuem transições do tipo t_{d1} e t_{d2} que foram alteradas pela regra de sincronização.

Para ilustrar a abordagem proposta, a IOWF-net apresentada na Figura 6 é considerada. Como mostrado em Aalst (1998), esta IOWF-net é localmente mas não globalmente *sound*, dado que a sua respectiva $U(\text{IOWF-net})$, representada pela Figura 7, não é *sound*. Observa-se prontamente que se a transição *too_late* da WF-net local

PC e a transição $send_final_version$ da $WF-net$ local AU são disparadas, o modelo é levado a um estado de *deadlock* com uma ficha no lugar $a5$, uma ficha no lugar too_late e uma ficha no lugar $final_version$ (as mensagens enviadas por uma $LWF-net$ nunca serão recebidas pela outra $LWF-net$).

Conforme apresentado na abordagem, para o exemplo considerado é necessário inicialmente identificar quais são os cenários potenciais existentes no modelo, representá-los através do uso dos sequentes da Lógica Linear e identificar através da análise dos sequentes da Lógica Linear quais destes cenários são responsáveis pelas situações de *deadlock*.

De acordo com Passos (2016), para a $U(IOWF-net)$ exibida na Figura 7, existem cinco cenários diferentes a serem estudados. O primeiro cenário, Sc_1 , onde a tarefa $send_reject$ será executada (disparo da transição $send_reject$). O segundo cenário, Sc_2 , onde as tarefas too_late e $receive_notification_1$ serão executadas (disparo das transições too_late e $receive_notification_1$). O terceiro cenário, Sc_3 , onde as tarefas too_late e $receive_notification_2$ serão executadas (disparo das transições too_late e $receive_notification_2$). O quarto cenário, Sc_4 , onde as tarefas $send_final_version$ e $receive_final_version$ serão executadas (disparo da transição $send_final_version$ e $receive_final_version$). E, finalmente, o quinto cenário, Sc_5 , onde as tarefas too_late e $send_final_version$ serão executadas (disparo das transições too_late e $send_final_version$).

Cada um dos cenários é então representado por um sequente específico da Lógica Linear que considera as marcações iniciais e finais da $U(IOWF-net)$ e uma lista não ordenada de transições (PASSOS; JULIA, 2013).

As transições da $U(IOWF-net)$ apresentada na Figura 7 são representadas pelas seguintes fórmulas da Lógica Linear:

$$\begin{aligned}
t_i &= i \multimap start_flow_author \otimes start_flow_PC, \\
t_1 &= send_draft = start_flow_author \multimap a1 \otimes draft, \\
t_2 &= receive_ack_draft = a1 \otimes ack_draft \multimap a2, \\
t_3 &= receive_accept = a2 \otimes accept \multimap a3, \\
t_4 &= receive_reject = a2 \otimes reject \multimap end_flow_author, \\
t_5 &= prepare_final_version = a3 \multimap a4, \\
t_6 &= receive_notification_1 = a3 \otimes too_late \multimap end_flow_author, \\
t_7 &= send_final_version = a4 \multimap a5 \otimes final_version, \\
t_8 &= receive_notification_2 = a4 \otimes too_late \multimap end_flow_author, \\
t_9 &= receive_ack_final = a5 \otimes ack_final \multimap end_flow_author, \\
t_{10} &= receive_draft = start_flow_PC \otimes draft \multimap p1, \\
t_{11} &= send_ack_draft = p1 \multimap ack_draft \otimes p2, \\
t_{12} &= evaluate = p2 \multimap p3, \\
t_{13} &= send_reject = p3 \multimap reject \otimes end_flow_PC, \\
t_{14} &= send_accept = p3 \multimap accept \otimes p4,
\end{aligned}$$

$$\begin{aligned}
t_{15} &= \text{too_late} = p4 \multimap \text{too_late} \otimes \text{end_flow_PC}, \\
t_{16} &= \text{receive_final_version} = \text{final_version} \otimes p4 \multimap p5, \\
t_{17} &= \text{send_ack_final} = p5 \multimap \text{ack_final} \otimes \text{end_flow_PC}, \\
t_o &= \text{end_flow_author} \otimes \text{end_flow_PC} \multimap o.
\end{aligned}$$

Os cinco diferentes cenários e respectivos sequentes da Lógica Linear são então apresentados na sequência:

$$\begin{aligned}
Sc_1 &= i, t_i, t_1, t_2, t_4, t_{10}, t_{11}, t_{12}, t_{13}, t_o \vdash o, \\
Sc_2 &= i, t_i, t_1, t_2, t_3, t_6, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, t_o \vdash o, \\
Sc_3 &= i, t_i, t_1, t_2, t_3, t_5, t_8, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, t_o \vdash o, \\
Sc_4 &= i, t_i, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o, \\
Sc_5 &= i, t_i, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, t_o \vdash o.
\end{aligned}$$

Os átomos nas árvores de prova serão representados apenas pelas letras iniciais do seu nome completo. Por exemplo, *draft*, *too_late* e *final_version* serão representados, respectivamente, por *d*, *tl* e *fv*. Um conjunto de transições t_i, t_j, t_k e t_l podem também ser representado como $t_{i,j,k,l}$.

As árvores de prova completas construídas para a prova de cada um destes sequentes são apresentadas em Passos (2016) e podem ser encontradas no Apêndice A.2 deste trabalho.

Na sequência são apresentados os resumos das árvores de prova para os cenários Sc_1 , Sc_2 e Sc_3 e as árvores de prova completas para os cenários Sc_4 e Sc_5 .

A árvore de prova do cenário Sc_1 é a seguinte:

$$\frac{\begin{array}{c} \underline{o \vdash o}_{\multimap_L} \\ \vdots \end{array}}{i, t_i, t_1, t_2, t_4, t_{10}, t_{11}, t_{12}, t_{13}, t_o \vdash o}$$

A árvore de prova para o cenário Sc_2 é apresentada a seguir:

$$\frac{\begin{array}{c} \underline{o \vdash o}_{\multimap_L} \\ \vdots \end{array}}{i, t_i, t_1, t_2, t_3, t_6, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, t_o \vdash o}$$

A árvore de prova para o cenário Sc_3 é dada por:

$$\frac{\begin{array}{c} \underline{o \vdash o}_{\multimap_L} \\ \vdots \end{array}}{i, t_i, t_1, t_2, t_3, t_5, t_8, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, t_o \vdash o}$$

Para uma melhor apresentação da abordagem, a árvore de prova do cenário Sc_4 foi dividida em duas partes, facilitando a identificação dos estados onde a regra de sincronização interferiu diretamente no processo. Lembrando que uma árvore da Lógica Linear é lida de baixo pra cima (modo *bottom-up*). A Parte 1 da árvore de prova do cenário Sc_4 é apresentada na sequência:

$$\begin{array}{c}
\vdots \\
\frac{\frac{a2 \vdash a2}{a2, a \vdash a2 \otimes a} \otimes_R \quad p4, a3, a3 \multimap a4, t_7, t_9, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{a2, a, p4, a2 \otimes a \multimap a3, t_5, t_7, t_9, t_{16}, t_{17}, o \vdash o}{\otimes_L} \\
\frac{p3 \vdash p3 \quad a2, a \otimes p4, t_3, t_5, t_7, t_9, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{p2 \vdash p2 \quad a2, p3, t_3, t_5, t_7, t_9, p3 \multimap a \otimes p4, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{\frac{a1 \vdash a1}{a1, ad \vdash a1 \otimes ad} \otimes_R \quad p2, a2, t_3, t_5, t_7, t_9, p2 \multimap p3, t_{14}, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{a1, ad, p2, a1 \otimes ad \multimap a2, t_3, t_5, t_7, t_9, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\otimes_L} \\
\frac{p1 \vdash p1 \quad a1, ad \otimes p2, t_2, t_3, t_5, t_7, t_9, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{\frac{sfp \vdash sfp}{sfp, d \vdash sfp \otimes d} \otimes_R \quad a1, p1, t_2, t_3, t_5, t_7, t_9, p1 \multimap ad \otimes p2, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{sfp, a1, d, t_2, t_3, t_5, t_7, t_9, sfp \otimes d \multimap p1, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\otimes_L} \\
\frac{sfa \vdash sfa \quad sfp, a1 \otimes d, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{sfa, sfp, sfa \multimap a1 \otimes d, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\otimes_L} \\
\frac{i \vdash i \quad sfa \otimes sfp, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
i, t_i, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o
\end{array}$$

A Parte 2 da árvore para o cenário Sc_4 é a seguinte:

$$\begin{array}{c}
\frac{\frac{efa \vdash efa}{efa, efp \vdash efa \otimes efp} \otimes_R \quad o \vdash o}{\multimap_L} \\
\frac{\frac{a5 \vdash a5}{a5, af \vdash a5 \otimes af} \otimes_R \quad efp, efa, efa \otimes efp \multimap o \vdash o}{\multimap_L} \\
\frac{a5, af, efp, a5 \otimes af \multimap efa, t_o \vdash o}{\otimes_L} \\
\frac{p5 \vdash p5 \quad a5, af \otimes efp, t_9, t_o \vdash o}{\multimap_L} \\
\frac{\frac{fv \vdash fv}{fv, p4 \vdash fv \otimes p4} \otimes_R \quad a5, p5, p5 \multimap af \otimes efp, t_9, t_o \vdash o}{\multimap_L} \\
\frac{p4, a5, fv, fv \otimes p4 \multimap p5, t_9, t_{17}, o \vdash o}{\otimes_L} \\
\frac{a4 \vdash a4 \quad p4, a5 \otimes fv, t_9, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\frac{a3 \vdash a3 \quad p4, a4, a4 \multimap a5 \otimes fv, t_9, t_{16}, t_{17}, o \vdash o}{\multimap_L} \\
\vdots
\end{array}$$

A árvore de prova para o cenário Sc_5 é apresentada a seguir:

$$\begin{array}{c}
\frac{}{tl,efpc,a5,fv,t9,to \vdash o} \otimes_L \\
\frac{a4 \vdash a4 \quad tl,efpc,a5 \otimes fv,t9,to \vdash o}{\vdash o} \multimap_L \\
\frac{a3 \vdash a3 \quad tl,efpc,a4,a4 \multimap a5 \otimes fv,t9,to \vdash o}{\vdash o} \multimap_L \\
\frac{}{a3,tl,efpc,a3 \multimap a4,t7,t9,to \vdash o} \otimes_L \\
\frac{p4 \vdash p4 \quad a3,tl \otimes efpc,t5,t7,t9,to \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{a2 \vdash a2}{a2,a \vdash a2 \otimes a} \otimes_R \quad p4,a3,t5,t7,t9,p4 \multimap tl \otimes efpc,to \vdash o}{\vdash o} \multimap_L \\
\frac{}{a2,a,p4,a2 \otimes a \multimap a3,t5,t7,t9,t15,to \vdash o} \otimes_L \\
\frac{p3 \vdash p3 \quad a2,a \otimes p4,t3,t5,t7,t9,t15,to \vdash o}{\vdash o} \multimap_L \\
\frac{p2 \vdash p2 \quad a2,p3,t3,t5,t7,t9,p3 \multimap a \otimes p4,t15,to \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{a1 \vdash a1}{a1,ad \vdash a1 \otimes ad} \otimes_R \quad p2,a2,t3,t5,t7,t9,p2 \multimap p3,t14,t15,to \vdash o}{\vdash o} \multimap_L \\
\frac{}{a1,ad,p2,a1 \otimes ad \multimap a2,t3,t5,t7,t9,t12,t14,t15,to \vdash o} \otimes_L \\
\frac{p1 \vdash p1 \quad a1,ad \otimes p2,t2,t3,t5,t7,t9,t12,t14,t15,to \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{sfpc \vdash sfpc}{sfpc,d \vdash sfpc \otimes d} \otimes_R \quad a1,p1,t2,3,5,7,9,p1 \multimap ad \otimes p2,t12,t14,t15,to \vdash o}{\vdash o} \multimap_L \\
\frac{}{sfpc,a1,d,t2,t3,t5,t7,t9,sfpc \otimes d \multimap p1,t11,t12,t14,t15,to \vdash o} \otimes_L \\
\frac{sfa \vdash sfa \quad sfpc,a1 \otimes d,t2,t3,t5,t7,t9,t10,t11,t12,t14,t15,to \vdash o}{\vdash o} \multimap_L \\
\frac{sfa,sfpc,sfa \multimap a1 \otimes d,t2,t3,t5,t7,t9,t10,t11,t12,t14,t15,to \vdash o}{\vdash o} \otimes_L \\
\frac{i \vdash i \quad sfa \otimes sfpc,t1,t2,t3,t5,t7,t9,t10,t11,t12,t14,t15,to \vdash o}{\vdash o} \multimap_L \\
i,t_i,t_1,t_2,t_3,t_5,t_7,t_9,t_{10},t_{11},t_{12},t_{14},t_{15},to \vdash o
\end{array}$$

Após a identificação dos cenários existentes no modelo e a representação destes através do uso dos sequentes da Lógica Linear é necessário identificar quais deste cenários possuem situações de *deadlock*, isto é, não finalizam corretamente, apresentando o último sequente da árvore de prova do cenário diferente do sequente identidade $o \vdash o$.

Conforme é possível identificar nas árvores de prova apresentadas acima, os últimos sequentes das árvores de prova para os cenários Sc_1 , Sc_2 , Sc_3 e Sc_4 é $o \vdash o$, logo estes cenários são livres de *deadlock*.

O último sequente para o cenário Sc_5 é $tl, efpc, a5, fv, t9, to \vdash o$; então o cenário Sc_5 é onde a situação de *deadlock* ocorre. É necessário então aplicar a regra de sincronização no cenário Sc_5 para remover a situação de *deadlock*.

Identificado o cenário em que ocorre a situação de *deadlock*, é necessário identificar quais são as transições do tipo t_{d1} e t_{d2} , e o lugar de comunicação assíncrona cp , conforme proposto por esta abordagem, através da análise do sequente da Lógica Linear para esse cenário.

Para o cenário Sc_5 , onde a situação de *deadlock* ocorre, a última transição que foi disparada antes da situação de *deadlock* é a transição t_7 (*send_final_version*) marcada em

negrito na árvore de prova. Em particular, os últimos átomos *a5* e *fv* (*final_version*) do último sequente da árvore de prova para o cenário Sc_5 são produzidos quando a transição t_7 é disparada; entretanto, devido à situação de *deadlock*, estes átomos não são consumidos e aparecem no último sequente da árvore de prova para o cenário.

O lugar de comunicação assíncrona *fv*, presente no último sequente da árvore de prova, é um lugar de saída da transição t_7 (última transição disparada na árvore de prova). Este lugar de comunicação é um lugar de entrada da transição t_{16} (*receive_final_version*) da *WF-net* local PC. A transição t_{16} corresponde então a uma transição morta do modelo global quando se executa o cenário Sc_5 . A transição t_7 irá corresponder então à transição t_{d1} da abordagem, a transição t_{16} à transição t_{d2} , e o átomo *fv* (*final_version*) ao lugar de comunicação assíncrona marcado *cp*. Nesta abordagem, as transições do tipo t_{d1} e t_{d2} , mais o lugar de comunicação assíncrona *cp*, serão substituídos por um elemento de comunicação parcialmente síncrono após a aplicação da regra de sincronização. A Figura 15 apresenta a aplicação da regra de sincronização no modelo.

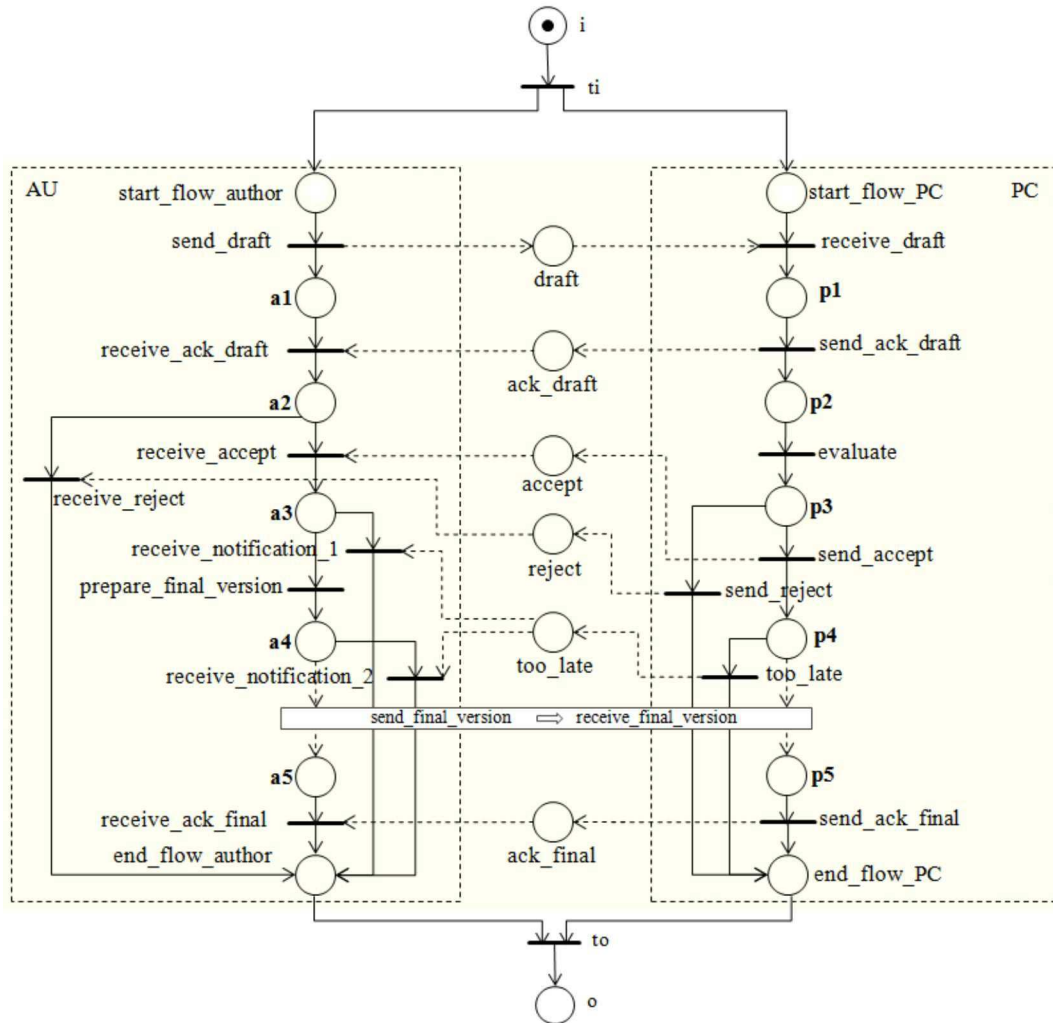


Figura 15 – Aplicando a regra de sincronização na $U(IOWF-net)$ apresentada na Figura 7.

A regra de sincronização, apresentada na Figura 15, não é uma transição de verdade, é uma regra de transição que sincroniza parte da estrutura de comunicação entre as LWF-nets *AU* e *PC*.

A transformação do mecanismo de comunicação assíncrono puro em um mecanismo de sincronização parcial, depois da aplicação da regra de sincronização, é apresentada na Figura 16.

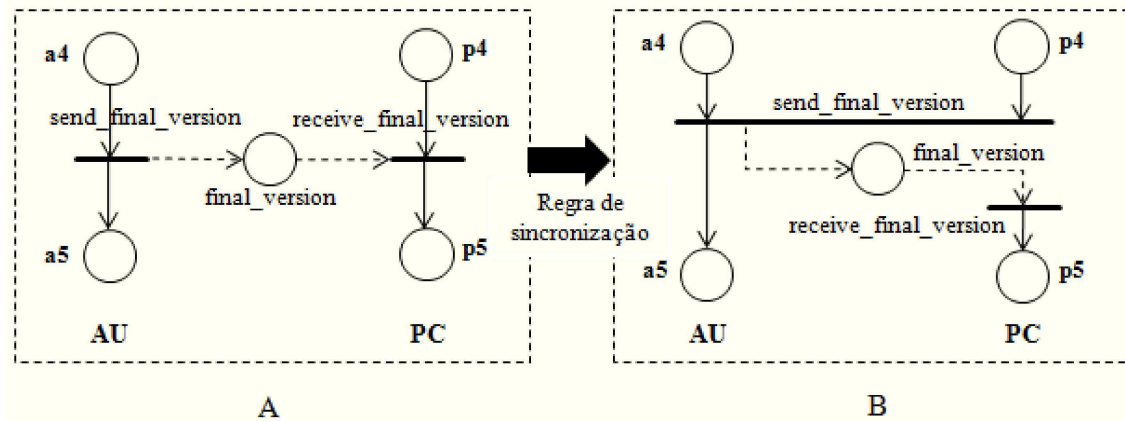


Figura 16 – Antes e depois da regra de sincronização na $U(IOWF-net)$ apresentada na Figura 7.

Na Figura 16A, o disparo da transição t_7 (transição do tipo t_{d1}) corresponde somente a uma condição necessária para o disparo da transição t_{16} (transição do tipo t_{d2}). Tal protocolo de comunicação assíncrono não é uma garantia então para o recebimento da mensagem enviada. Depois do disparo da transição t_7 , uma ficha irá eventualmente permanecer presa no lugar de comunicação *final_version*. Por outro lado, na Figura 16B, o fato de sincronizar a execução da primeira atividade associada à transição t_7 (transição do tipo t_{d1}) em ambos os processos locais *AU* e *PC*, corresponde à garantia da execução da atividade *receive_final_version* associada à transição t_{16} (transição do tipo t_{d2}). De fato, o disparo da transição do tipo t_{d1} , depois da aplicação da regra de sincronização, corresponde a uma condição necessária e suficiente para o disparo da transição do tipo t_{d2} , e até o fim do cenário associado nenhuma ficha irá permanecer nos lugares de comunicação intermediários.

Para o exemplo apresentado, tem-se que depois da aplicação da regra de sincronização, a situação de *deadlock* é removida e o cenário Sc_5 não será mais executado na rede *unfolded*. Na prática, o cenário Sc_5 não é mais considerado dentre os cenários potenciais quando se considera a $U(IOWF-net)$.

Considerando agora os quatro cenários potenciais restantes após a remoção do cenário Sc_5 , têm-se que as transições t_7 e t_{16} não aparecem nos seguintes dos cenários Sc_1 , Sc_2 e Sc_3 . Como essas transições não ocorrem nesses cenários, a alteração do modelo não modifica as árvores de prova dos cenários correspondentes. Por outro lado, ambas transições aparecem no cenário Sc_4 .

Após identificar os cenários onde as transições td_1 e td_2 aparecem, é necessário reprocessar as árvores de prova da Lógica Linear para esses cenários. Considerando que até o momento de disparo de td_1 o cenário não é alterado, é necessário reprocessar as árvores de prova apenas a partir do sequente onde a transição td_1 é disparada. Uma grande vantagem desta abordagem é que quando uma situação de *deadlock* ocorre, somente a parte alterada pela regra de sincronização precisa ser reprocessada, o restante da árvore de prova da Lógica Linear permanecerá inalterada. A análise do modelo somente acontece então na parte modificada. Considerando a análise do último sequente da árvore de prova para o cenário que foi alterado após a aplicação da regra de sincronização, verifica-se que o cenário ainda é *sound*.

De acordo com a Figura 16B, as transições t_7 (*send_final_version*) e t_{16} (*receive_final_version*) da $U(\text{IOWF-net})$ são representadas agora pelas seguintes novas fórmulas da Lógica Linear:

$$t_7 = a4 \otimes p4 \multimap a5 \otimes \text{final_version},$$

$$t_{16} = \text{final_version} \multimap p5.$$

A Parte 1 da árvore de prova para o cenário Sc_4 continua inalterada após a aplicação da regra de sincronização. A Parte 2 da árvore de prova precisa ser reprocessada e é como se segue:

$$\begin{array}{c} \frac{\frac{efa \vdash efa \quad efp \vdash efp}{efa, efp \vdash efa \otimes efp} \otimes_R \quad o \vdash o}{\multimap_L} \\ \frac{\frac{a5 \vdash a5 \quad af \vdash af}{a5, af \vdash a5 \otimes af} \otimes_R \quad efp, efa, efa \otimes efp \multimap o \vdash o}{\multimap_L} \\ \frac{\quad}{a5, af, efp, a5 \otimes af \multimap efa, t_o \vdash o} \otimes_L \\ \frac{p5 \vdash p5 \quad a5, af \otimes efp, t_9, t_o \vdash o}{\multimap_L} \\ \frac{fv \vdash fv \quad p5, p5 \multimap af \otimes efp, t_9, t_o \vdash o}{\multimap_L} \\ \frac{\quad}{a5, fv, fv \multimap p5, t_9, t_{17}, t_o \vdash o} \otimes_L \\ \frac{\frac{a4 \vdash a4 \quad p4 \vdash p4}{a4, p4 \vdash a4 \otimes p4} \otimes_R \quad a5 \otimes fv, t_9, t_{16}, t_{17}, t_o \vdash o}{\multimap_L} \\ \frac{a3 \vdash a3 \quad a4, p4, a4 \otimes p4 \multimap a5 \otimes fv, t_9, t_{16}, t_{17}, t_o \vdash o}{\multimap_L} \\ \vdots \end{array}$$

O último sequente na árvore de prova para o novo cenário Sc_4 é $o \vdash o$; então depois da aplicação da regra de sincronização, o cenário Sc_4 , mesmo alterado, ainda é um cenário livre de *deadlock*.

A Figura 17 mostra o modelo final depois da aplicação da regra de sincronização na $U(\text{IOWF-net})$ exibida na Figura 7, removendo o cenário Sc_5 que era responsável pela situação de *deadlock* no modelo original e alterando o cenário Sc_4 . Note que o lugar adicional aux_1 é produzido no modelo final com o objetivo de respeitar a definição de

uma *Workflow net* (de fato, a transição *receive_final_version* (t_{16}) precisa de pelo menos uma transição de entrada com o objetivo de respeitar a estrutura básica de uma *Workflow net*).

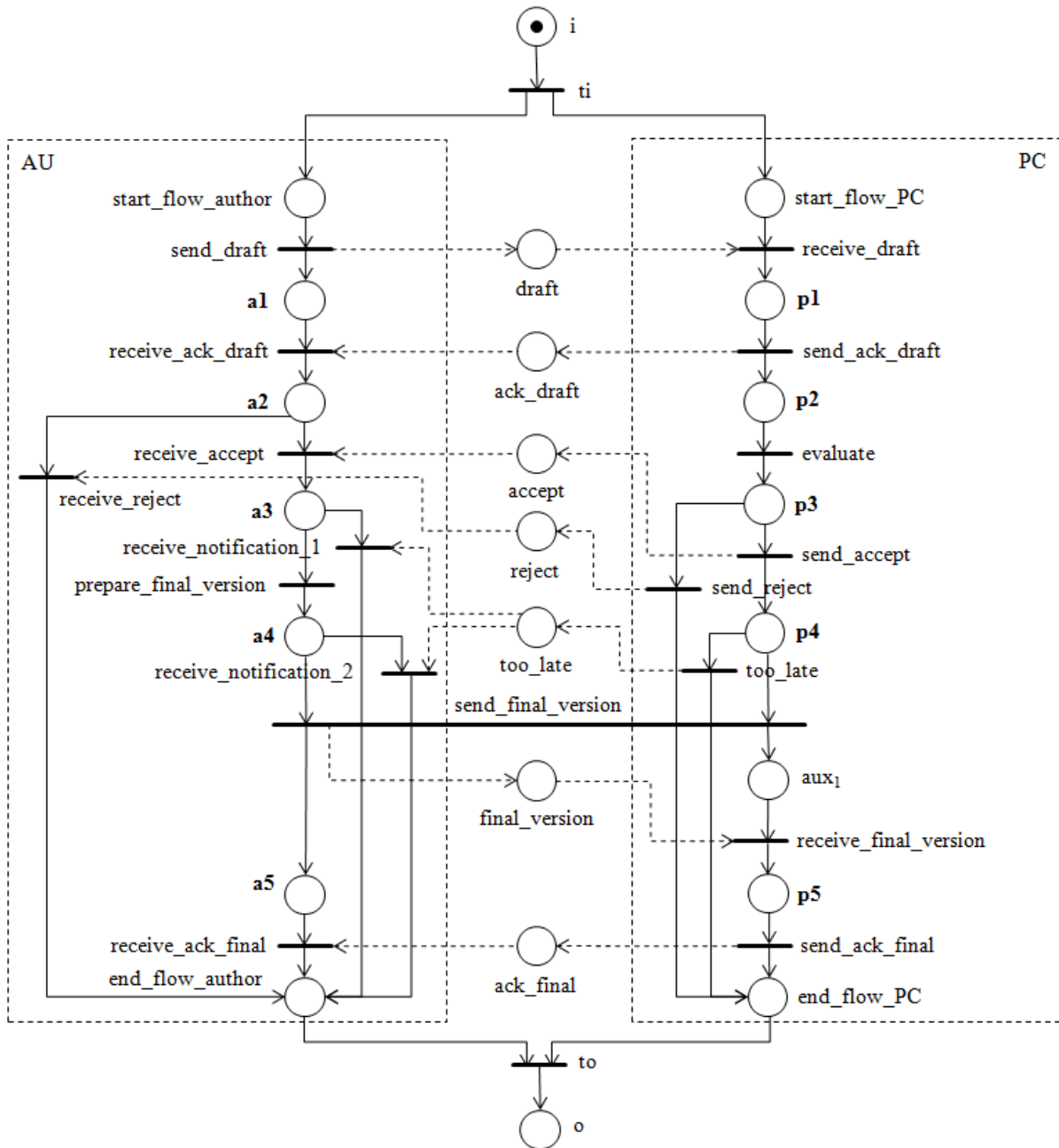


Figura 17 – Modelo final após a aplicação da regra de sincronização na $U(IOWF-net)$ da Figura 7.

4.2 Análise Qualitativa

De acordo com Aalst (1998), a análise qualitativa se refere à prova do critério de correção *Soundness* definido para *WorkFlow nets*. As condições da propriedade *Soundness* garantem a ausência de *deadlock* no modelo de *workflow*. Dessa forma, é importante que um modelo seja *sound* (AALST, 1998) para garantir a finalização de seus processos com sucesso.

A regra de sincronização apresentada neste trabalho tem como principal objetivo remover situações de *deadlock* do modelo original e gerar um novo modelo *sound*. Para transformar um modelo que não respeita a propriedade *Soundness* em um modelo *sound* é necessário indentificar os cenários que são responsáveis pela situação de *deadlock*. Após a identificação desses cenários, a regra de sincronização é aplicada e os cenários responsáveis por situações de *deadlock* são então removidos do modelo global. Além dos cenários removidos, pelo menos um cenário precisará ser alterado para cada aplicação da regra com o objetivo de contemplar a alteração do modelo gerada pela remoção do cenário que era responsável pelo estado de inconsistência.

Em Passos (2016) é formalizado um método para a análise qualitativa, onde é feita a verificação do critério de correção *Soundness* em processos de *Workflow Interorganizacionais*. O método consiste em construir a árvore de prova de cada cenário que compõe o modelo global e depois analisar o último sequente da árvore de prova do cenário e verificar se esse sequente corresponde ao sequente identidade $o \vdash o$. Caso o último sequente seja igual à $o \vdash o$, o cenário é *sound*. Se todos os cenários que compõem o modelo global são *sound*, então o modelo global também será *sound*.

Como mostrado na seção 4.1, para aplicar o método proposto é necessário provar os sequentes de todos os cenários do modelo global através da construção das árvores de prova da Lógica Linear e identificar através da análise do último sequente da árvore de prova se o cenário é *sound*. O cenário que possui a situação de *deadlock* é removido do modelo após a aplicação da regra e a prova do cenário que precisou ser alterado é refeita, provando que mesmo depois da aplicação da regra o cenário alterado continua *sound*. Dessa forma, o método proposto em Passos (2016) é utilizado durante a abordagem proposta neste trabalho, de forma que a análise qualitativa do modelo já é realizada na aplicação da regra de sincronização.

Essa seção irá validar a regra de sincronização proposta nesta pesquisa, através de métodos clássicos das redes de Petri e o apoio da ferramenta de simulação CPN Tools. Será possível através dessa validação confirmar se uma situação de *deadlock* que existia no modelo original realmente é removida após a aplicação da regra de sincronização.

A propriedade da Vivacidade, apresentada na seção 2.1.1, garante que o sistema seja livre de *deadlock* (MURATA, 1989). Uma rede de Petri é dita viva se todas as transições podem ser sensibilizadas a partir de qualquer marcação do grafo das marcações acessíveis (MURATA, 1989).

A aplicação da regra de sincronização altera as boas propriedades do modelo global, transformando este modelo em uma rede de Petri viva. Os estados de *deadlock* no grafo das marcações acessíveis são alcançados através do disparo das transições pertencentes aos sequentes da Lógica Linear dos cenários responsáveis pelas situações de *deadlock*. Removendo o cenário responsável pela situação de *deadlock*, a marcação atingida no estado de *deadlock* no grafo das marcações também será removida, permitindo que todas as transições possam ser sensibilizadas a partir de qualquer marcação do grafo das marcações acessíveis.

O cenário que precisa ser alterado após a aplicação da regra não altera a propriedade da Vivacidade da rede de Petri, considerando que apesar das fórmulas lógicas das transições desse cenário serem alteradas, o seu sequente linear a ser provado permanece o mesmo. Considerando que o sequente linear a ser provado para o cenário é mantido e a sequência de disparos do cenário não é alterada, então não é gerada uma nova marcação no grafo das marcações acessíveis.

Para validar a abordagem proposta, o simulador CPN Tools foi utilizado (RATZER et al., 2003). Inicialmente é necessário implementar o modelo no CPN Tools. Para isso é necessário clicar com o botão direito do mouse sobre algum lugar da área de trabalho do *software* e escolher a opção *New Net* (Nova Rede). O sistema cria um nome padrão para a nova rede criada, que pode ser alterado clicando sobre *New Page* (Nova Página) na barra superior da janela e substituído por um novo nome desejado.

Através da paleta *Create* (Criar), os elementos do modelo são inseridos e nomeados, podendo modelar a rede de Petri conforme necessidade do usuário. Ao usuário criar um lugar, ele poderá dar um nome e um tipo para esse elemento, além de definir as fichas, caso o lugar tenha uma marcação. Na criação da transição, será possível definir o nome da transição, definir o tempo de disparo da transição, em caso de redes de Petri t-temporizadas, e a criação de condições de guarda em caso de redes de Petri coloridas.

Para inserir um arco no modelo, basta clicar sobre um lugar ou uma transição com o botão direito e selecionar a opção *New Arc* (Novo Arco). Respeitando as regras de criação de uma rede de Petri, ao acrescentar um arco é possível ligar uma transição a um lugar ou um lugar a uma transição. Quando é criado um arco, é possível o usuário definir condições, tempos e tipos sobre esse elemento; por padrão ele vem em branco.

Por padrão todos os lugares também vêm sem fichas e com o tipo *UNIT*, para inserir fichas em um lugar basta o usuário clicar uma vez sobre o lugar que deseja marcar, apertar duas vezes a tecla *Tab* e digitar a quantidade. Caso o usuário tenha alterado o tipo padrão do lugar, é possível também informar o tipo das fichas que serão inseridas.

Para verificar a Vivacidade em um modelo antes e depois da aplicação da regra de sincronização, a $U(IOWF-net)$ da Figura 7 é considerada. A Figura 18 apresenta a implementação do modelo da Figura 7 no *software* CPN Tools.

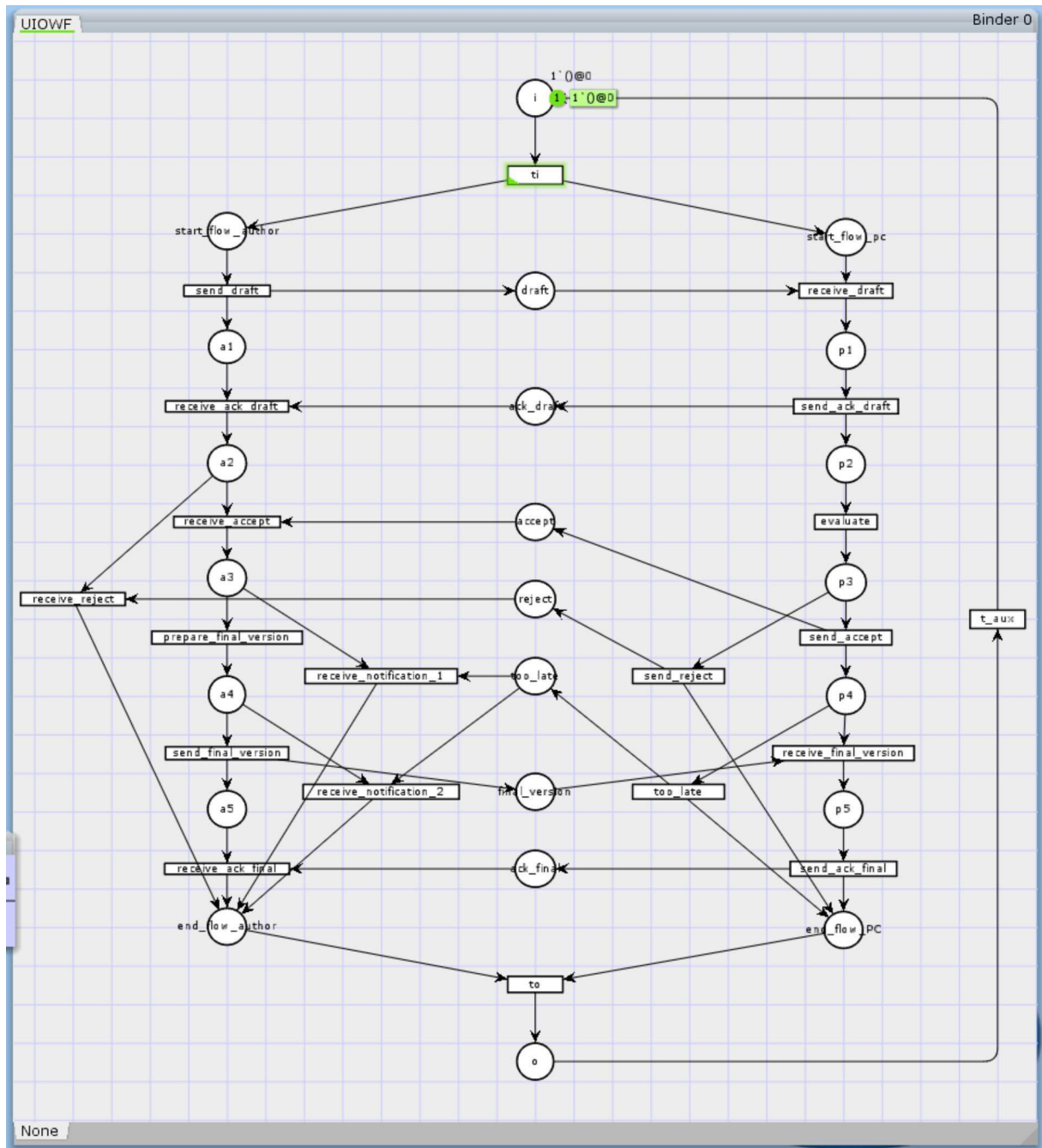


Figura 18 – Implementação no CPN Tools do modelo apresentado na Figura 7.

Para realizar a análise qualitativa no CPN Tools é utilizado o método proposto em Aalst e Hee (2004) em que é adicionada à rede uma nova transição auxiliar que tem como entrada o lugar final e como saída o lugar início. Pode-se notar que no modelo implementado apresentado na Figura 18, a transição auxiliar *t_aux* foi adicionada.

A paleta SS (Espaço de Estados, em inglês *State Space*) é disponibilizada pelo simulador CPN Tools e permite que uma análise, baseada no grafo de marcações, seja feita para todos os caminhos possíveis no modelo apresentado (RATZER et al., 2003). Para realizar a análise das propriedades da rede de Petri, inicialmente deve ser utilizada a ferramenta

Calculate State Space (Calcular Espaço de Estados). Após o cálculo do espaço de estados, é possível, através da ferramenta *Displays the node with the specified number* (Exibe o nó com o número especificado), iniciar a construção do grafo das marcações acessíveis acrescentando a primeira marcação do grafo. Para dar continuidade na construção do grafo, é necessário escolher a ferramenta *Display the successors to this node* (Exibir os sucessores deste nó). A cada clique na área de trabalho com essa ferramenta selecionada, uma nova marcação do grafo das marcações acessíveis é acrescentada, até que o grafo seja criado completamente.

Assim sendo, para o modelo implementado na Figura 18, o grafo das marcações acessíveis, que permite uma visualização gráfica dos estados alcançáveis a partir do estado inicial, é apresentado na Figura 19. Nessa figura também é apresentada a paleta SS que possui as funcionalidades do CPN Tools necessárias para a geração do grafo das marcações acessíveis.

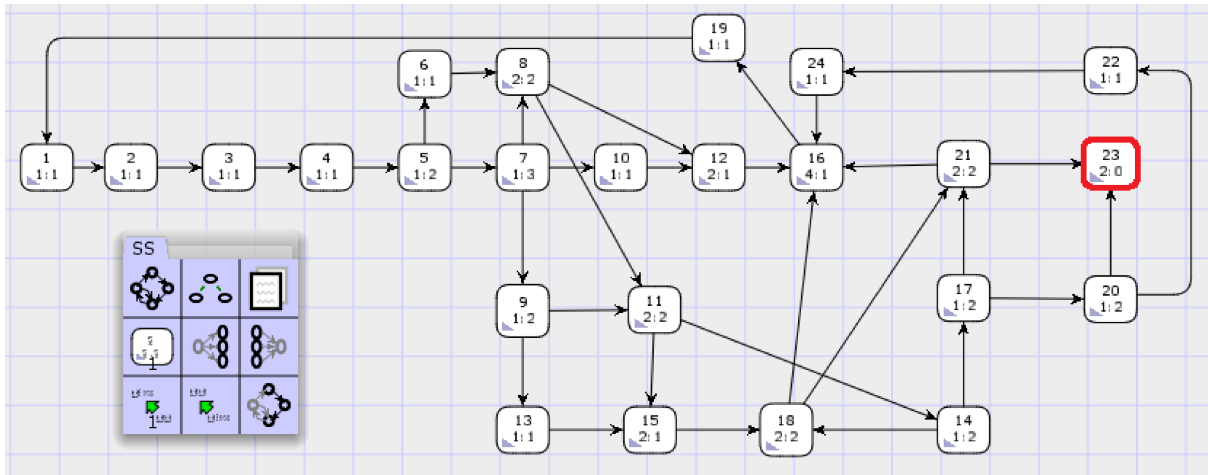


Figura 19 – Grafo das marcações acessíveis para a rede de Petri da Figura 7 antes da regra de sincronização, gerado pelo simulador CPN Tools.

A Figura 19 apresenta o grafo das marcações acessíveis para a Figura 7 que representa o modelo antes da aplicação da regra de sincronização. É possível verificar no estado, destacado com uma borda mais espessa no grafo, representado pela marcação 23, que uma situação de *deadlock* pode ser alcançada, já que existe nessa marcação dois arcos de entrada e nenhum arco de saída. Esse estado de *deadlock* é alcançado no grafo quando as transições *too_late* e *send_final_version* são disparadas, o que corresponde à execução do cenário Sc_5 onde essas mesmas transições são disparadas.

O CPN Tools possibilita também gerar o resultado da análise da rede de Petri em um arquivo através da ferramenta *Save Report* (Salvar Relatório), localizada na paleta SS. No relatório gerado são fornecidas informações sobre as boas propriedades do modelo como a Vivacidade e a Limitabilidade. Para salvar o relatório com o resultado da análise é necessário inicialmente utilizar a ferramenta *Calculate State Space* (Calcular Espaço de

Estados) e depois a ferramenta SSC (Componente Fortemente Conexo, em inglês *Strongly Connected Component*).

Como o método proposto nesse trabalho tem como principal objetivo a remoção do *deadlock* no modelo global, é apresentado na Figura 20 o resultado da propriedade da Vivacidade do relatório gerado através do *State Space* no *software* CPN Tools para a implementação apresentada na Figura 18. É possível verificar nas informações da propriedade de Vivacidade do relatório gerado que uma marcação morta foi identificada; logo o modelo possui uma situação de *deadlock*. Assim como foi possível verificar no grafo das marcações acessíveis, a marcação 23 é morta.

Liveness Properties

```

Dead Markings
  [23]

Dead Transition Instances
  None

Live Transition Instances
  None

```

Figura 20 – Relatório de Vivacidade para a rede de Petri da Figura 7 antes da regra de sincronização, gerado pelo simulador CPN Tools.

Após a aplicação da regra de sincronização, algumas transições do modelo original são alteradas, fazendo com que um novo modelo seja gerado; dessa forma, uma nova rede de Petri precisa ser implementada no *software*. A Figura 21 apresenta a implementação do modelo da Figura 17 que mostra o modelo final após a aplicação da regra de sincronização, no *software* CPN Tools.

Para verificar se o novo modelo proposto, após a aplicação da regra de sincronização, é livre de *deadlock*, um novo cálculo de espaço de estados (através da ferramenta *Calculate State Space*) deve ser feito.

Após o cálculo de espaço de estados, um novo grafo de marcações acessíveis deve ser iniciado pela ferramenta *Displays the node with the specified number* e completado através da ferramenta *Display the successors to this node*. Para o modelo implementado na Figura 21, é apresentado o grafo das marcações acessíveis que permite uma visualização gráfica dos estados alcançáveis a partir do estado inicial na Figura 22.

Na Figura 22 é possível verificar que após a aplicação da regra de sincronização um novo grafo fortemente conexo é gerado e a marcação morta é removida, permitindo que todas as transições possam ser sensibilizadas a partir de qualquer marcação do grafo

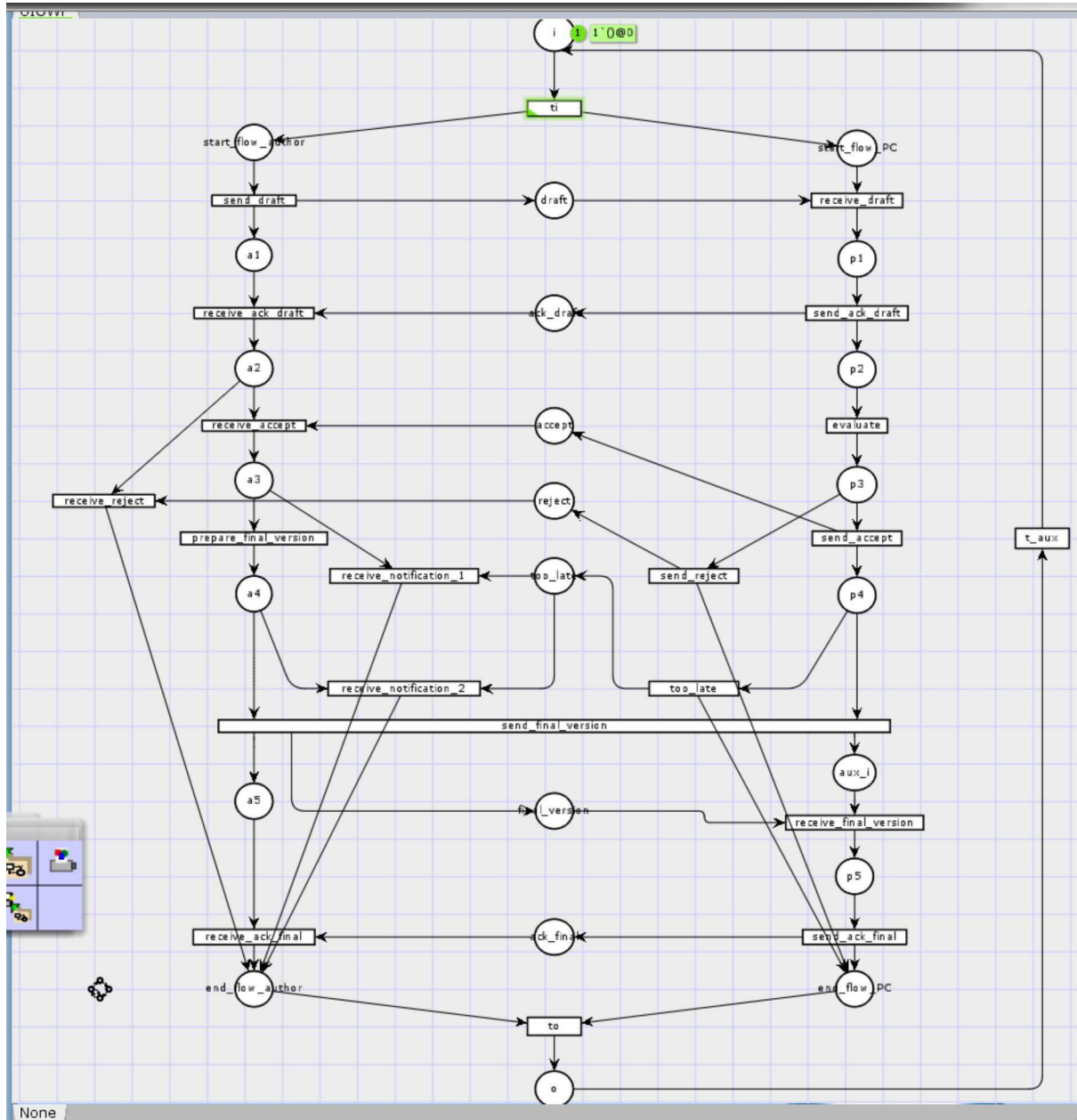


Figura 21 – Implementação no CPN Tools do modelo apresentado na Figura 17.

das marcações acessíveis. Como o cenário Sc_5 , responsável pela situação de *deadlock* no exemplo, é removido após a aplicação da regra de sincronização, não terá mais uma sequência de disparos das transições *too_late* e *send_final_version* que conduza o grafo das marcações acessíveis a um estado de *deadlock*.

Dessa forma, é possível concluir que após a aplicação da regra de sincronização a nova rede de Petri gerada é viva. Logo livre de *deadlock*.

A Figura 23 apresenta o resultado da propriedade da Vivacidade do relatório gerado através do *State Space* no *software* CPN Tools para a implementação apresentada na Figura 21.

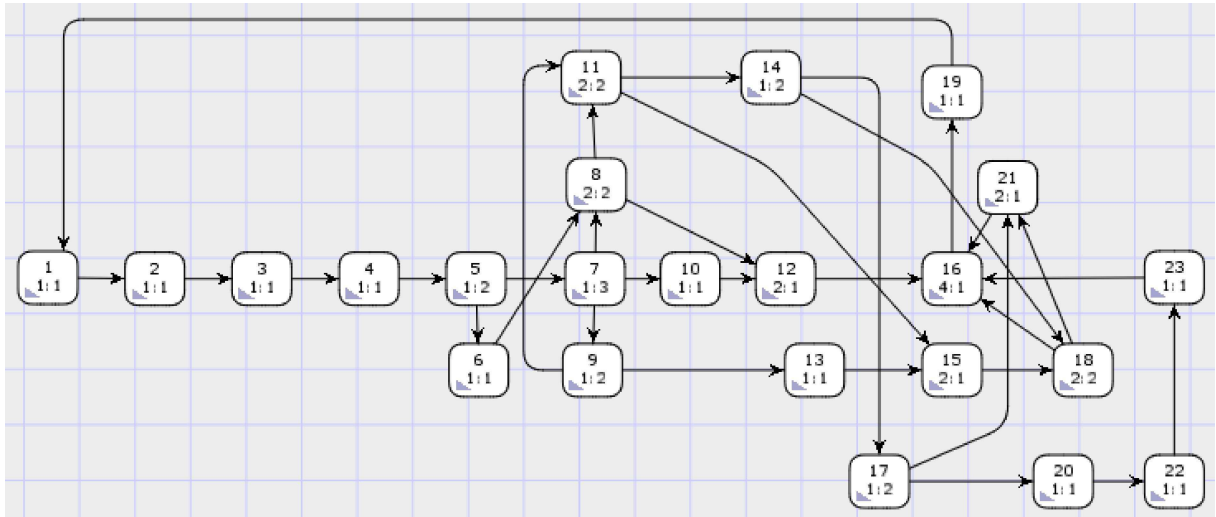


Figura 22 – Grafo das marcações acessíveis para a rede de Petri da Figura 17 depois da regra de sincronização, gerado pelo simulador CPN Tools.

É possível verificar, nas informações da propriedade de Vivacidade do relatório gerado, que nenhuma marcação foi identificada para o novo modelo gerado após a aplicação da regra de sincronização; sendo assim, a nova rede é viva. Logo livre de situações de *deadlock*. Assim como esperado, as situações de *deadlock* são removidas após a aplicação da regra de sincronização no modelo global e os cenários alterados não introduzem novas situações de *deadlock* e permanecem *sound*.

Liveness Properties

Dead Markings

None

Dead Transition Instances

None

Live Transition Instances

All

Figura 23 – Relatório de Vivacidade para a rede de Petri da Figura 17 depois da regra de sincronização, gerado pelo simulador CPN Tools.

4.3 Análise Quantitativa

De acordo com Aalst e Hee (2004), a análise quantitativa se refere aos requisitos de desempenho e capacidade de um processo de *workflow*. A análise dos aspectos quantitativos foca na definição de indicadores de desempenho, tais como tempo médio de conclusão, nível de serviço e utilização de capacidade (AALST; HEE, 2004). A presente pesquisa considera a análise quantitativa ao verificar o tempo de execução de um cenário alterado em uma IOWF-net após a aplicação da regra de sincronização. Através de tal análise será possível identificar em particular o impacto que a sincronização de processos terá no tempo global de execução do serviço proposto pela IOWF-net.

Inicialmente, será necessário identificar os cenários que serão alterados pela aplicação da regra de sincronização. Tais cenários são apresentados na seção 4.1. São cenários que possuem as transições do tipo t_{d1} e t_{d2} , que representam, respectivamente, a última transição disparada antes da situação do *deadlock* assim como a transição morta do modelo analisado. Esses cenários precisam ser representados através de árvores de prova canônica da Lógica Linear antes e após a aplicação da regra de sincronização.

As datas de execução das tarefas serão representadas através de datas simbólicas. A maior vantagem da utilização de datas simbólicas é que quando estas já estiverem calculadas, poderão então ser utilizadas diretamente para qualquer caso que será tratado pelo processo de *workflow* analisado - mesmo considerando casos cujo instante de iniciação e/ou duração global de atividades sejam diferentes (PASSOS, 2009). Como mostrado em Passos (2009), através das árvores de prova, datas simbólicas para a execução de tarefas podem ser derivadas.

A regra de sincronização apresentada em 4.1 consiste em substituir lugares de comunicação assíncrona da IOWF-net por novos mecanismos de comunicação parcialmente síncronos com o objetivo de prevenir a ocorrência de mensagens perdidas, permitindo que o modelo *unfolded* do processo de *workflow* respeite a propriedade *Soundness*. O fato de sincronizar a execução da atividade representada pela transição do tipo t_{d1} em ambos os processos locais (representados pelas LWF-nets) corresponde de fato à garantia da execução da atividade representada pela transição do tipo t_{d2} .

Os lugares de entrada das transições do tipo t_{d1} e t_{d2} , anteriormente a aplicação da regra de sincronização, possuíam datas de consumo dos átomos de forma independente entre si, considerando que cada uma das atividades (representadas pelas transições do modelo) estão em LWF-nets distintas. Ao substituir um lugar de comunicação assíncrona na IOWF-net por um novo mecanismo de comunicação parcialmente síncrono, a transição de tipo t_{d1} é substituída por uma regra de sincronização que força os dois *workflows* locais a executarem a tarefa simultaneamente. Como consequência dessa sincronização, as datas simbólicas de consumo dos átomos dos lugares de entrada dessa transição poderão sofrer um atraso após a aplicação da regra de sincronização, pois as datas de consumo desses lugares de entrada de t_{d1} tornam-se dependentes entre si. O atraso que pode ser gerado ao

aplicar a regra de sincronização vai depender da estrutura da rede de Petri no momento em que as tarefas serão sincronizadas, além dos tempos associados às atividades dos processos.

Considere que $p1$ é um lugar de entrada da transição do tipo t_{d1} pertencente à LWF-net $L1$, $p2$ um lugar de entrada da transição do tipo t_{d2} pertencente à LWF-net $L2$, $p3$ um lugar de saída da transição do tipo t_{d1} pertencente à LWF-net $L1$, $p4$ um lugar de saída da transição do tipo t_{d2} pertencente à LWF-net $L2$ e cp um lugar de comunicação assíncrona que interage os dois *workflows* locais $L1$ e $L2$. A Figura 24 apresenta as datas simbólicas (*minimum* e *maximum*) das atividades associadas com t_{d1} e t_{d2} .

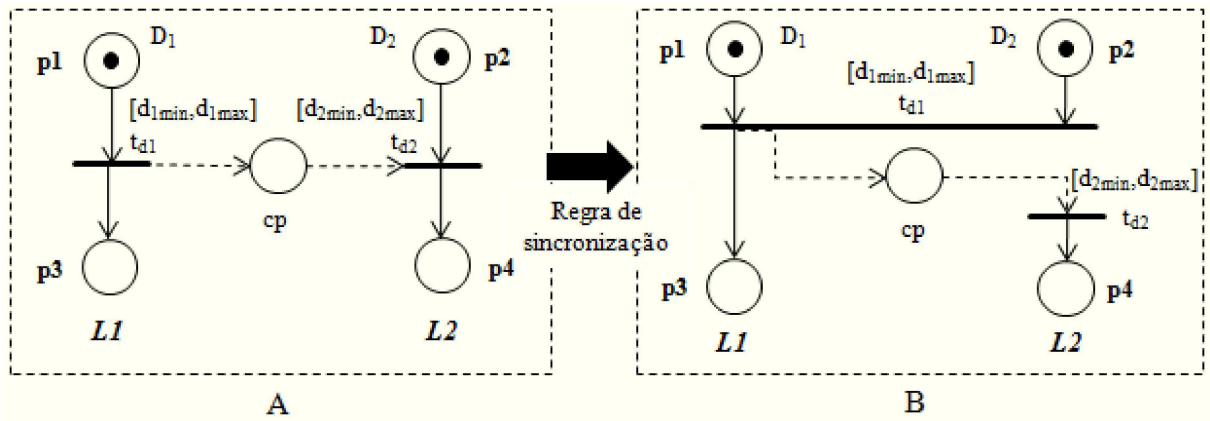


Figura 24 – Datas simbólicas antes e depois da regra de sincronização.

Lembrando que, no presente trabalho, D_i denota uma data e d_i uma duração associada a um disparo de uma transição (t_i). O atraso ocorrido pela aplicação da regra pode ser apresentado através do uso da Lógica Linear.

O sequente da Lógica Linear a seguir considera as marcações iniciais e finais da rede de Petri apresentada na Figura 24A, além de uma lista não ordenada de transições envolvidas nessas marcações:

$$S = p1, p2, t_{d1}, t_{d2} \vdash p3, p4.$$

As transições envolvidas no sequente antes da aplicação da regra, apresentadas no modelo da Figura 24A, são representadas pelas seguintes fórmulas da Lógica Linear:

$$t_{d1} = p1 \multimap p3 \otimes cp,$$

$$t_{d2} = cp \otimes p2 \multimap p4.$$

Aplicando as regras da Lógica Linear ao sequente linear S , apresentado para o modelo da Figura 24A, é possível construir uma árvore de prova canônica da Lógica Linear com cálculo de datas, conforme a seguir:

$$\begin{array}{c}
\frac{cp(D_1+d_1, \max\{D_1+d_1, D_2\}) \vdash cp \quad p2(D_2, \max\{D_1+d_1, D_2\}) \vdash p2}{cp(D_1+d_1, \max\{D_1+d_1, D_2\}), p2(D_2, \max\{D_1+d_1, D_2\}) \vdash cp \otimes p2} \otimes_R \quad p3(D_1+d_1, .), p4(\max\{D_1+d_1, D_2\}+d_2, .) \vdash p3, p4 \multimap_L \\
\frac{}{p2(D_2, .), p3(D_1+d_1, .), cp(D_1+d_1, .), cp \otimes p2 \multimap p4 \vdash p3, p4} \otimes_L \\
\frac{p1(D_1, D_1+d_1) \vdash p1 \quad p2(D_2, .), p3 \otimes cp, t_{d2} \vdash p3, p4}{p1(D_1, .), p2(D_2, .), p1 \multimap p3 \otimes cp, t_{d2} \vdash p3, p4} \multimap_L
\end{array}$$

Para verificar a diferença no tempo de execução, após a aplicação da regra de sincronização, é necessário verificar as datas de produção e consumo do modelo após a sincronização. Dessa forma, é necessário representar as novas transições apresentadas na Figura 24B através de novas fórmulas da Lógica Linear:

$$t_{d1} = p1 \otimes p2 \multimap p3 \otimes cp,$$

$$t_{d2} = cp \multimap p4.$$

Lembrando que apesar das fórmulas da Lógica Linear que representam as transições da Figura 24B terem sido alteradas em relação as transições da Figura 24A, a definição do sequente a ser provada em relação às estruturas básicas apresentadas na Figura 24 não é alterada, pois as marcações iniciais e finais do modelo, além da lista de transições não ordenada, são mantidas. Assim, a árvore de prova da Lógica Linear com cálculo de datas para o sequente S , considerando o modelo da Figura 24B, é apresentada na sequência:

$$\begin{array}{c}
\frac{cp(\max\{D_1, D_2\}+d_1, \max\{D_1, D_2\}+d_1+d_2) \vdash cp \quad p3(\max\{D_1, D_2\}+d_1, .), p4(\max\{D_1, D_2\}+d_1+d_2, .) \vdash p3, p4}{cp(\max\{D_1, D_2\}+d_1, \max\{D_1, D_2\}+d_1+d_2), p3(\max\{D_1, D_2\}+d_1, .), p4(\max\{D_1, D_2\}+d_1+d_2, .) \vdash p3, p4} \multimap_L \\
\frac{}{p3(\max\{D_1, D_2\}+d_1, .), cp(\max\{D_1, D_2\}+d_1, .), cp \multimap p4 \vdash p3, p4} \otimes_L \\
\frac{\frac{p1(D_1, \max\{D_1, D_2\}+d_1) \vdash p1 \quad p2(D_2, \max\{D_1, D_2\}+d_1) \vdash p2}{p1(D_1, \max\{D_1, D_2\}+d_1), p2(D_2, \max\{D_1, D_2\}+d_1) \vdash p1 \otimes p2} \otimes_R \quad p3 \otimes cp, t_{d2} \vdash p3, p4}{p1(D_1, .), p2(D_2, .), p1 \otimes p2 \multimap p3 \otimes cp, t_{d2} \vdash p3, p4} \multimap_L
\end{array}$$

Para uma melhor visualização e comparação das datas calculadas nas árvores de prova, as datas simbólicas de produção e de consumo dos átomos do modelo antes e depois da aplicação da regra de sincronização são apresentadas a seguir na Tabela 4.

Tabela 4 – Datas simbólicas de produção e consumo dos átomos.

Átomo	Data de Produção Antes da Regra	Data de Produção Depois da Regra	Data de Consumo Antes da Regra	Data de Consumo Depois da Regra
p1	D_1	D_1	$D_1 + d_1$	$\max\{D_1, D_2\} + d_1$
p2	D_2	D_2	$\max\{D_1 + d_1, D_2\}$	$\max\{D_1, D_2\} + d_1$
cp	$D_1 + d_1$	$\max\{D_1, D_2\} + d_1$	$\max\{D_1 + d_1, D_2\}$	$\max\{D_1, D_2\} + d_1 + d_2$
p3	$D_1 + d_1$	$\max\{D_1, D_2\} + d_1$	<i>desconhecido</i>	<i>desconhecido</i>
p4	$\max\{D_1 + d_1, D_2\} + d_2$	$\max\{D_1, D_2\} + d_1 + d_2$	<i>desconhecido</i>	<i>desconhecido</i>

Analisando a Tabela 4 é possível verificar que a data de produção dos átomos $p1$ e $p2$ permanece inalterada mesmo após a aplicação da regra de sincronização; dessa forma, todo o modelo anterior à aplicação da regra não possui o seu tempo de execução alterado. Esse comportamento pode ser verificado ao analisar as duas primeiras linhas das duas primeiras colunas da Tabela 4.

Por outro lado, a data de consumo dos átomos $p1$ e $p2$, no momento em que a sincronização do modelo ocorre, são alteradas. Conforme anteriormente apresentado, a data de consumo dos átomos do lugar $p1$ da LWF-net $L1$ e do lugar $p2$ da LWF-net $L2$ se tornam dependentes uma da outra, necessitando assim verificar qual o tempo máximo de consumo entre os dois átomos. O tempo máximo de consumo dos dois átomos é somado ao tempo de execução da transição que consome esses átomos, que será a transição do tipo t_{d1} . Esse comportamento pode ser verificado ao analisar as duas primeiras linhas das duas últimas colunas da Tabela 4. O atraso gerado pela aplicação da regra de sincronização só ocorre no modelo quando a data de produção D_2 (referente ao lugar $p2$ da LWF-net $L2$) for maior que a data de produção D_1 (referente ao lugar $p1$ da LWF-net $L1$).

Para verificar o cálculo de datas proposto, a $U(\text{IOWF-net})$ apresentada na Figura 7 é considerada. Como mostrado na seção 4.1, o cenário Sc_4 possui tanto a transição do tipo t_{d1} quanto a transição do tipo t_{d2} . O sequente do cenário Sc_4 deve ser analisado através da construção de uma árvore de prova da Lógica Linear com cálculo de datas. A Figura 17 corresponde à Figura 6 após a aplicação da regra de sincronização. As árvores de prova da Lógica Linear com cálculo de datas para o cenário Sc_4 , antes e depois da aplicação da regra de sincronização, são apresentadas no Apêndice A.3.

Uma vantagem da utilização da Lógica Linear, é que as árvores de prova com cálculo de datas construídas para o cenário Sc_4 , antes e depois da aplicação da regra de sincronização, apresentadas no Apêndice A.3, são as mesmas árvores construídas para a verificação da propriedade *Soundness* apresentadas no Apêndice A.2. Dessa forma, os resultados derivados da verificação da propriedade *Soundness* na seção 4.1 são diretamente reaproveitadas para o cálculo de datas.

Para a análise quantitativa desta pesquisa, é utilizado o método apresentado em Passos (2009) onde para cada tarefa (representada no modelo por uma transição) na $U(IOWF-net)$ analisada, deve-se extrair as datas de produção, D_P , e consumo, D_C , do átomo que representa a pré-condição da transição correspondente a esta tarefa. Quando há mais de uma pré-condição associada à transição, considera-se a data máxima das produções dos átomos correspondentes a estas pré-condições (PASSOS, 2009).

A data de produção do átomo, D_P , corresponde ao início da execução da tarefa associada à transição e a data de consumo, D_C , corresponde ao término da execução da mesma. Assim, é gerado um intervalo $[D_P, D_C]$ de datas onde o recurso que executará a referida tarefa deverá ser disponibilizado para realizá-la. Uma vez que as datas de produção e consumo são dependentes de durações de sensibilização d_i , cujo valor pertence a um intervalo de tempo $\Delta_i = [\delta_{imin}, \delta_{imax}]$, pode-se considerar vários intervalos possíveis de execução das tarefas, de acordo com um planejamento estratégico (PASSOS, 2009).

A Tabela 14 apresentada no Apêndice A.3 mostra os intervalos de datas simbólicas de execução para as tarefas do cenário Sc_4 , considerando o intervalo de execução $I_{Exec} = [D_{Pmin}, D_{Cmax}]$ e as respectivas árvores de prova da Lógica Linear com cálculo de datas para o modelo antes e depois da aplicação da regra de sincronização.

A informação sobre quando o processo global finalizará para o cenário Sc_4 é dada pela data de produção do átomo o . Esta data de produção é extraída da árvore de prova com cálculo de datas para este cenário. A Tabela 5 mostra as datas simbólicas de produção do átomo o no cenário Sc_4 antes e depois da aplicação da regra de sincronização.

Tabela 5 – Data de produção do átomo o no cenário Sc_4 .

Cenário Sc_4	Data de produção do átomo o
Antes	$D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3 + d_5 + d_7 + d_{16} + d_{17} + d_9 + d_o$
Depois	$\max\{D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3 + d_5, D_i + d_i + d_1 + d_{10} + d_{11} + d_{12} + d_{14}\} + d_7 + d_{16} + d_{17} + d_9 + d_o$

Devido a aplicação da regra de sincronização, a data simbólica de produção do cenário Sc_4 é alterada. Como é possível verificar na Tabela 14 apresentada no Apêndice A.3, o intervalo de data simbólica de execução para a tarefa *send_final_version*, correspondente ao disparo da transição do tipo td_1 , é alterado devido à sincronização dos processos locais. Antes da regra de sincronização o disparo da transição *send_final_version* dependia apenas do consumo do átomo do lugar a_4 da LWF-net AU , porém, após a regra de

sincronização, o disparo de *send_final_version* passa a depender também do consumo do átomo de *p4* da LWF-net PC.

Apenas a partir do disparo da transição *prepare_final_version*, cujo tempo máximo de disparo depende do tempo de consumo do átomo *a4*, é que as datas simbólicas de execução das tarefas do modelo são alteradas.

Conforme é possível verificar na Tabela 5, após aplicar a regra de sincronização, a data de produção do átomo *o* no cenário *Sc₄* é alterado de $D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3 + d_5 + d_7 + d_{16} + d_{17} + d_9 + d_o$ para $\max\{D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3 + d_5, D_i + d_i + d_1 + d_{10} + d_{11} + d_{12} + d_{14}\} + d_7 + d_{16} + d_{17} + d_9 + d_o$.

Considerando que as transições *t₃* e *t₅* no modelo são disparadas após as transições *t₁₂* e *t₁₄*, logo as datas de *t₃* e *t₅* serão sempre maiores que as datas de *t₁₂* e *t₁₄*. Dessa forma, tem-se que $D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3 + d_5$ será sempre maior que $D_i + d_i + d_1 + d_{10} + d_{11} + d_{12} + d_{14}$, uma vez que tanto $D_i + d_i + d_1 + d_{10} + d_{11} + d_2 + d_3 + d_5$ quanto $D_i + d_i + d_1 + d_{10} + d_{11} + d_{12} + d_{14} + d_3 + d_5$ é maior que $D_i + d_i + d_1 + d_{10} + d_{11} + d_{12} + d_{14}$. Pode-se reescrever então a data de produção do átomo *o* no cenário *Sc₄* após a regra de sincronização como $D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3 + d_5 + d_7 + d_{16} + d_{17} + d_9 + d_o$.

Uma vez que é possível reescrever a data de produção do átomo *o* no cenário *Sc₄* após a aplicação da regra de sincronização como $D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3 + d_5 + d_7 + d_{16} + d_{17} + d_9 + d_o$, devido a estrutura do modelo, a data de produção do átomo *o* no cenário *Sc₄* é mantida e o tempo em que o processo global finalizará para tal cenário não será alterado.

A Tabela 14, apresentada no Apêndice A.3, mostra os intervalos de datas simbólicas de execução para as tarefas do cenário *Sc₄*, considerando o intervalo de execução $I_{Exec} = [D_{Pmin}, D_{Cmax}]$ e as respectivas árvores de prova da Lógica Linear com cálculo de datas para o modelo antes e depois da aplicação da regra de sincronização.

Tabela 6 – Intervalos de datas simbólicas, $[D_{Pmin}, D_{Cmax}]$, para a finalização do cenário *Sc₄*.

Cenário <i>Sc₄</i>	Intervalo
Antes	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + \max\{d_{2min}, d_{12min} + d_{14min}\} + d_{3min} + d_{5min} + d_{7min} + d_{16min} + d_{17min} + d_{9min} + d_{omin}, D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + \max\{d_{2max}, d_{12max} + d_{14max}\} + d_{3max} + d_{5max} + d_{7max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$
Depois	$[max\{D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + \max\{d_{2min}, d_{12min} + d_{14min}\} + d_{3min} + d_{5min}, D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + d_{12min} + d_{14min}\} + d_{7min} + d_{16min} + d_{17min} + d_{9min} + d_{omin}, max\{D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + \max\{d_{2max}, d_{12max} + d_{14max}\} + d_{3max} + d_{5max}, D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + d_{12max} + d_{14max}\} + d_{7max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$

Considerando as datas de produção do átomo *o* para o cenário *Sc₄*, apresentadas na Tabela 5, pode-se calcular a data ao mais cedo de finalização do processo, dada pela data mínima D_{Pmin} de produção do átomo *o*, e a data ao mais tarde de finalização do processo, dada pela data máxima D_{Pmax} de produção do átomo *o*. A Tabela 6 mostra os

intervalos $[D_{Pmin}, D_{Cmax}]$ de produção do átomo o antes e depois da aplicação da regra de sincronização.

Para ilustrar a abordagem proposta a partir de valores de datas e durações numéricas, considera-se a $U(IOWF-net)$ t-temporal apresentada na Figura 25. O modelo apresentado na Figura 25 corresponde à $U(IOWF-net)$ da Figura 7 com intervalos de tempo associados às transições. Considera-se também que o processo a ser tratado inicia-se na data 0 (zero), ou seja, $D_i = 0$.

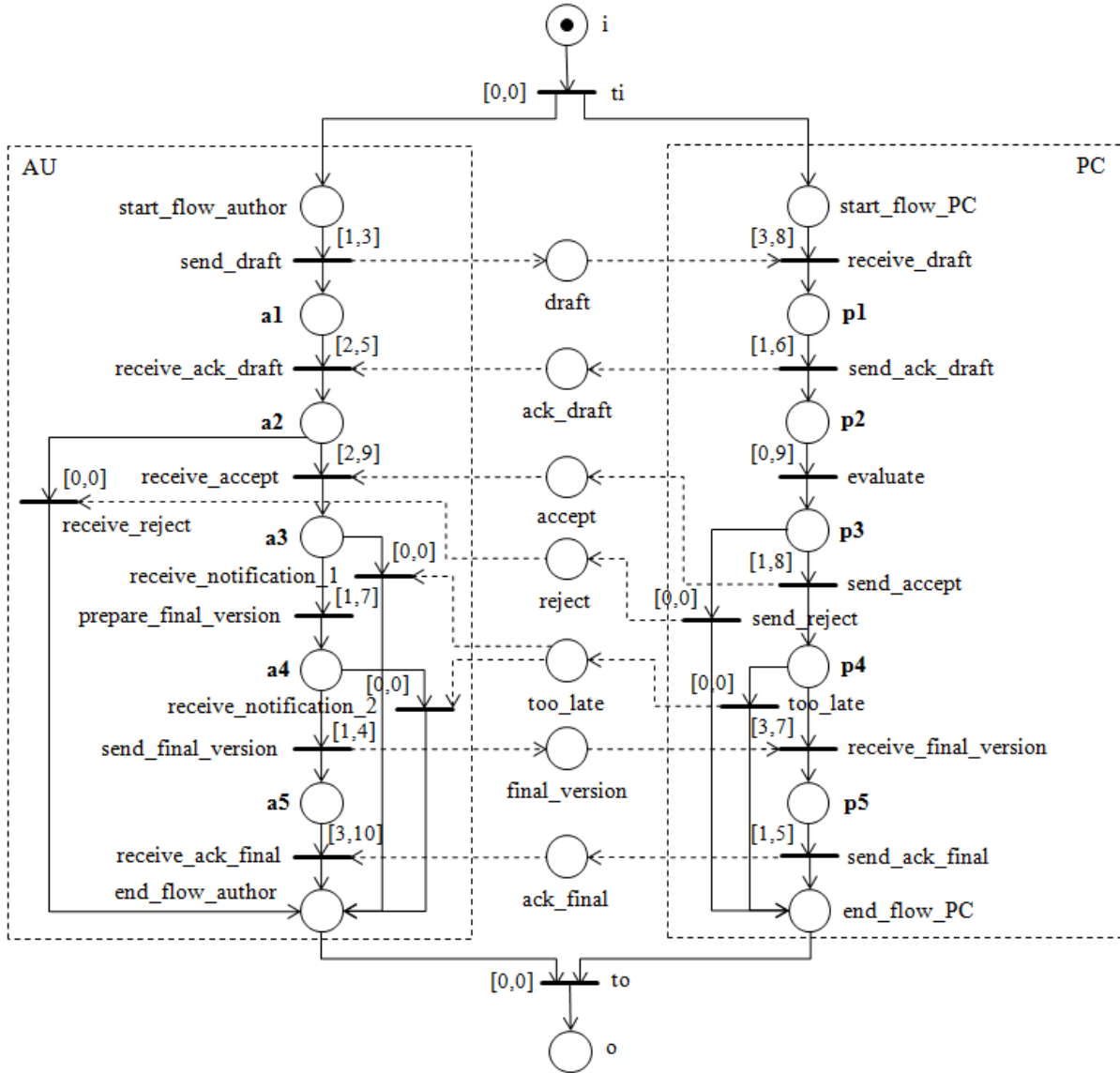


Figura 25 – Rede de Petri t-temporalizada correspondente à $U(IOWF-net)$ da Figura 7.

Uma vez que as datas simbólicas do modelo já foram calculadas, os intervalos de datas numéricas para a finalização do processo podem ser calculados apenas substituindo as datas simbólicas presentes na Tabela 6 pelas datas numéricas associadas às transições apresentadas na Figura 25. A Tabela 7 mostra o resultado deste processo.

Tabela 7 – Intervalos de datas numéricas para a finalização do cenário Sc_4 .

Cenário Sc_4	Intervalo numérico
Antes	[18,76]
Depois	[18,76]

Como a data de produção do átomo o é mantida para o cenário Sc_4 após a aplicação da regra de sincronização, logo o seu intervalo de datas simbólicas $[D_{Pmin}, D_{Cmax}]$ também será mantido. Consequentemente, o intervalo de datas numéricas para o cenário Sc_4 será igual, tanto antes quanto depois da aplicação da regra de sincronização, como pode ser verificado na Tabela 7.

Para validar a abordagem proposta no ponto de vista quantitativo, foram realizadas simulações no simulador CPN Tools (RATZER et al., 2003). Inicialmente é necessário implementar a rede de Petri t-temporizada no *software* CPN Tools.

A Figura 26 apresenta a implementação no *software* CPN Tools da rede de Petri t-temporizada apresentada na Figura 25.

O CPN Tools oferece algumas funções de distribuições estatísticas. Para essa pesquisa foi escolhida a distribuição estatística uniforme, na qual a probabilidade de cada tempo ocorrer é a mesma, que é implementada no CPN Tools através da função $uniform(a : real, b : real) : real$. Por padrão, o CPN Tools carrega o modelo com o tipo *UNIT*, que não é um tipo temporizado. Para transformar o tipo *UNIT* em um tipo temporizado é necessário adicionar a palavra *timed* em sua declaração.

Para acrescentar tempo em uma transição no CPN Tools é necessário clicar sobre a transição, apertar duas vezes a tecla *TAB* e informar a quantidade de tempo desejada. A função *uniform* é acrescentada para gerar um tempo aleatório dentro de um intervalo fornecido. Como a função *uniform* é utilizada apenas para tipos inteiros no CPN Tools, é necessário acrescentar a função *round()* para converter o resultado da distribuição *uniform* em um valor inteiro arredondado.

Através da paleta *Monitoring*, utilizando a ferramenta *Data Collection*, é possível acrescentar um monitor ao modelo. Essa ferramenta tem como objetivo selecionar um lugar do modelo para criar um ponto de parada. Ao alcançar o lugar marcado pelo monitor, a execução do modelo é finalizada.

O CPN Tools oferece a funcionalidade de replicação da simulação (do tipo Monte Carlo), através do uso da função $CPN'Replications.nreplicationsn$ onde n é o número de simulações que serão realizadas pelo simulador. É possível gerar essa função clicando com o botão direito sobre um espaço vazio da área de trabalho do *software*, escolhendo a opção *Create aux text*. Ao criar um novo texto é necessário informar o nome da função de replicação e a quantidade de simulações que serão realizadas. Segundo Rozinat et al. (2008), quanto maior o número de replicações, maior a precisão do resultado. No entanto, é necessário decidir um limite aceitável de replicações. Sendo assim, para as simulações

realizadas nesta pesquisa foram consideradas 50 replicações.

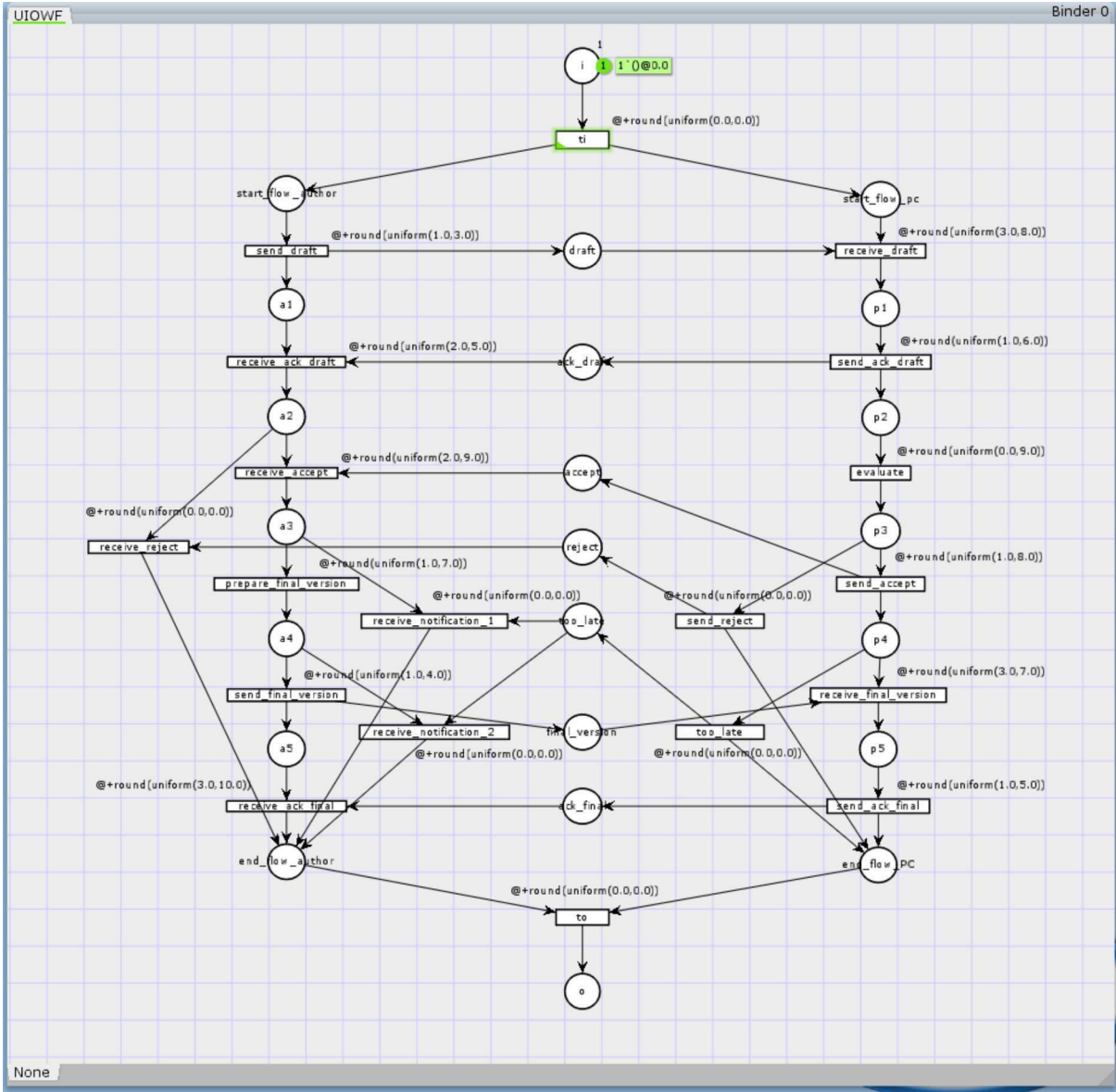


Figura 26 – Implementação no CPN Tools do modelo apresentado na Figura 25.

Para a análise quantitativa desta pesquisa apenas serão considerados os cenários alterados no modelo após a aplicação da regra de sincronização, uma vez que o objetivo é validar o método apresentado, considerando somente os cenários que podem sofrer uma perda de desempenho devido à sincronização introduzida. Dessa forma, a análise proposta neste trabalho não precisa considerar o conjunto de todos os cenários possíveis e sim somente os cenários alterados após a aplicação da regra de sincronização. Assim, os outros cenários da *U(IOWF-net)* são removidos e o cenário Sc_4 , alterado após a aplicação da regra, é implementado de forma isolada no CPN Tools. A Figura 27 apresenta a implementação do cenário Sc_4 do modelo da Figura 25 no simulador CPN Tools.

Para executar as replicações das simulações é necessário clicar com o botão direito sobre o texto auxiliar criado na área de trabalho da ferramenta com o nome da função de replicação. Após clicar sobre o texto com o botão direito é necessário escolher a opção “*Evaluate ML*” para iniciar a execução das replicações e gerar os arquivos com os resultados da simulação. Ao executar as replicações das simulações do cenário Sc_4 , o relatório estatístico com o resultado das replicações, apresentado na Figura 28, é gerado.

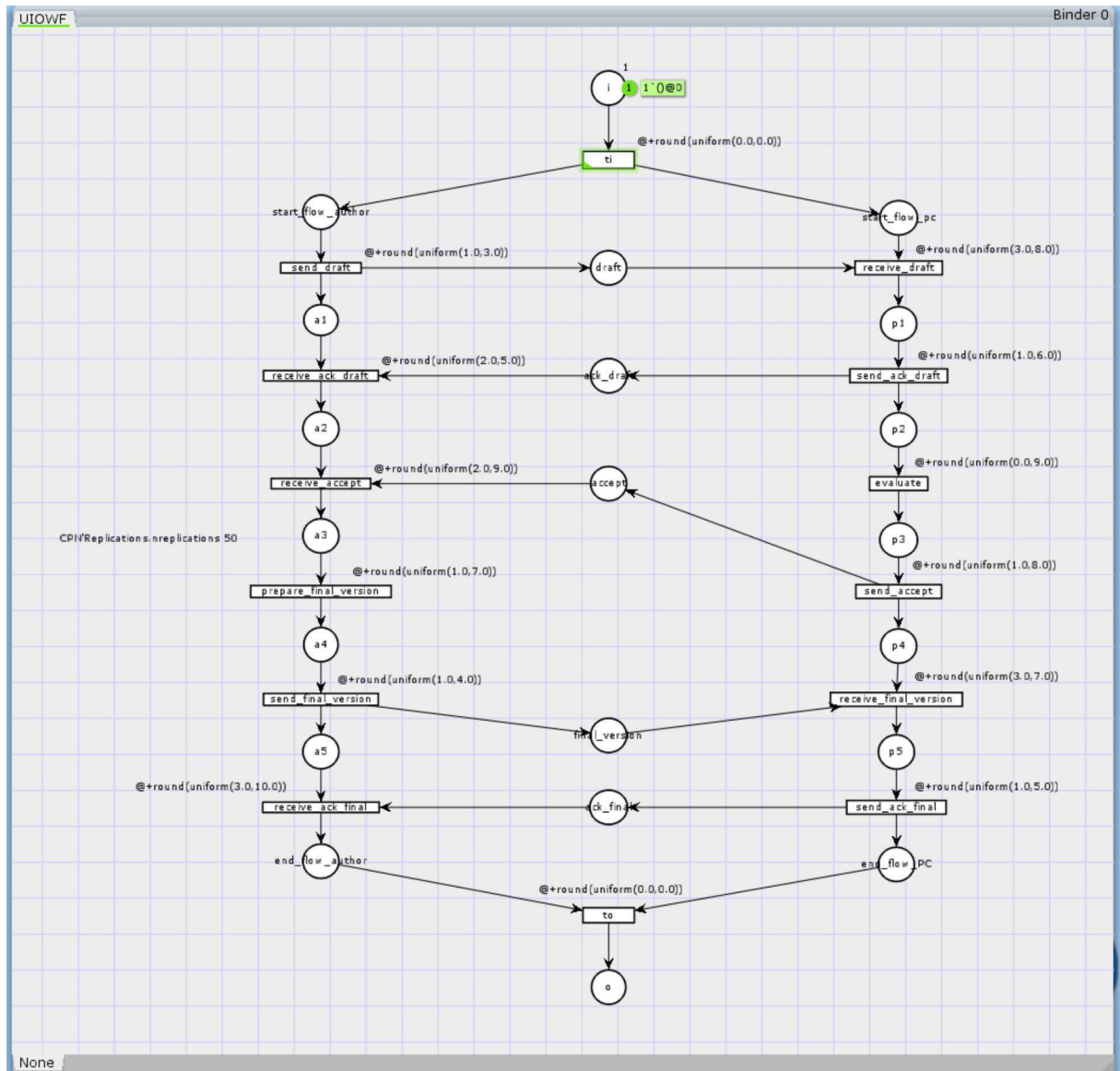


Figura 27 – Implementação no CPN Tools do cenário Sc_4 antes da regra de sincronização.

Conforme é possível observar na área destacada com uma borda mais espessa na Figura 28, o término mais cedo ocorre na data 35 e o término mais tarde na data 58. Os resultados analíticos produziram, como mostra a Tabela 7, o intervalo de datas [18,76], ou seja, o término mais cedo na data 18 e o término mais tarde na data 76. Desta forma, os valores obtidos na simulação pertencem ao intervalo numérico definido pelo método

analítico.

CPN Tools Performance Report
 Net: /cygdrive/C/Users/Vinicius/Desktop/CPN/sc4.cpn
 Number of replications: 50

Statistics							
Name	Avrg	90% Half Length	95% Half Length	99% Half Length	StD	Min	Max
ParadaO							
count_iid	14.000000	0.000000	0.000000	0.000000	0.000000	14	14
max_iid	47.200000	1.273893	1.528823	2.045491	5.349041	35	58
min_iid	0.000000	0.000000	0.000000	0.000000	0.000000	0	0
avrg_iid	20.969488	0.579931	0.695986	0.931196	2.435114	15.485714	25.879310

Figura 28 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_4 antes da regra de sincronização.

Considerando o intervalo de confiança de 95%, as replicações podem variar 1.528823 entre o limite inferior e o limite superior do intervalo, como mostra a coluna *95% Half Length* da Figura 28. Se esse valor (arredondado) for subtraído ao limite inferior e somado ao limite superior, tem-se que o término mais cedo ocorre na data 33,47 e o término mais tarde na data 59,53. Desta forma, considerando o intervalo de confiança de 95%, os valores obtidos na simulação ainda pertencem ao intervalo de datas [18,76], apresentado na Tabela 7, definido pelo método analítico.

Após a aplicação da regra de sincronização, a $U(IOWF-net)$ da Figura 7 é alterada e dá origem a um novo modelo, representado pela Figura 17. Esse novo modelo, gerado após a aplicação da regra de sincronização, também precisa ser implementado no CPN Tools para efeito de comparação. A Figura 29 apresenta a implementação do modelo da Figura 17 no *software* CPN Tools com os mesmos tempos associados nas transições do modelo da Figura 25.

Para analisar apenas o cenário alterado após a aplicação da regra, os outros cenários do modelo implementado na Figura 29 são removidos e o cenário Sc_4 é implementado de forma isolada no CPN Tools. A Figura 30 apresenta a implementação do cenário Sc_4 do modelo da Figura 29 no *software* CPN Tools.

Ao executar as replicações das simulações do cenário Sc_4 , após a regra de sincronização, o relatório estatístico com o resultado das replicações, apresentado na Figura 31, é gerado.

Conforme é possível observar na área destacada com uma borda mais espessa na Figura 31, o término mais cedo ocorre na data 34 e o término mais tarde na data 60. Os resultados analíticos produziram, como mostra a Tabela 7, o intervalo de datas [18,76], ou seja, o término mais cedo na data 18 e o término mais tarde na data 76. Desta forma,

os valores obtidos na simulação pertencem ao intervalo numérico definido pelo método analítico.

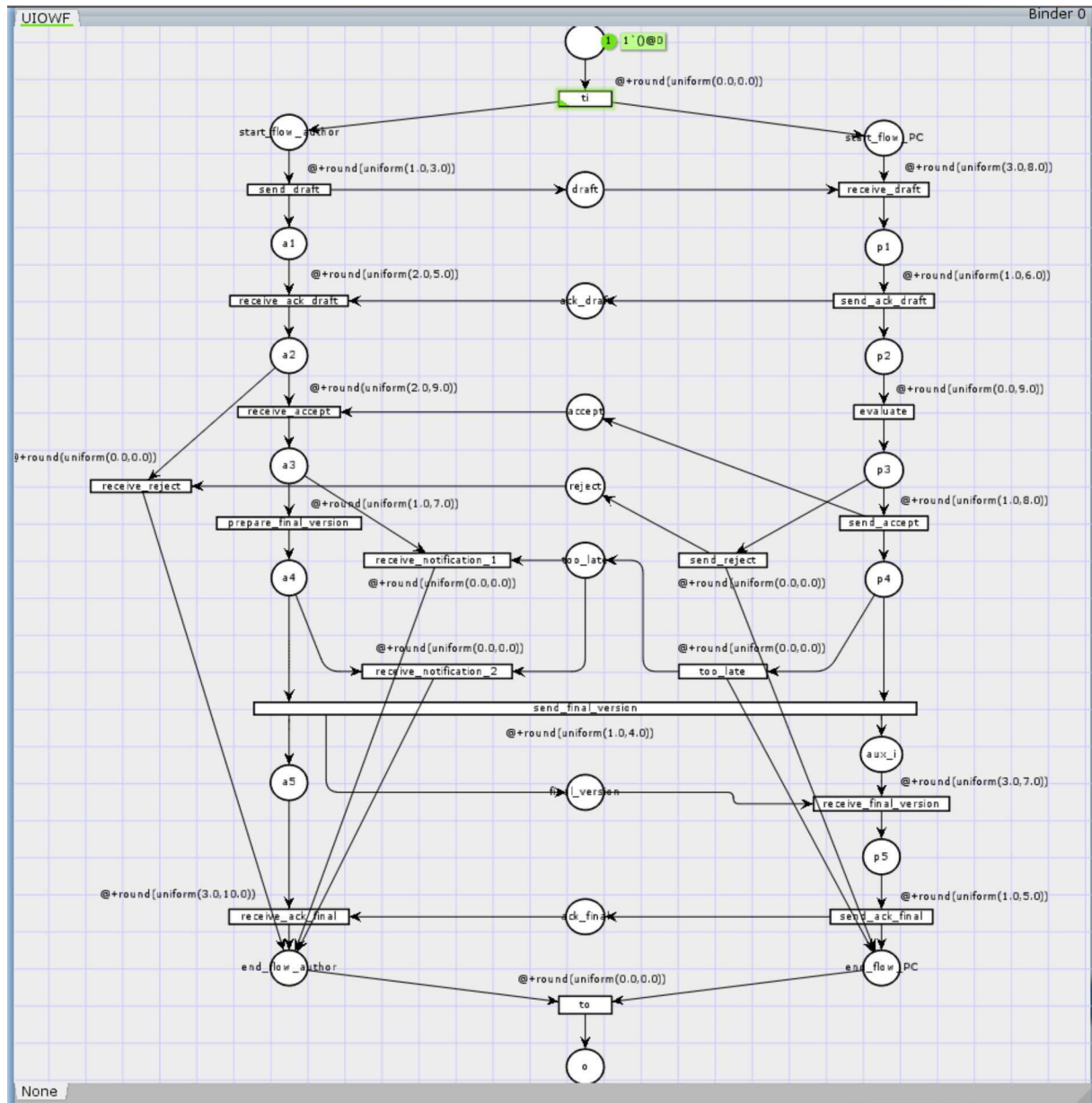


Figura 29 – Implementação no CPN Tools do modelo t-temporizado correspondente a $U(IOWF-net)$ da Figura 17.

Considerando o intervalo de confiança de 95%, as replicações podem variar 1.632898 entre o limite inferior e o limite superior do intervalo, como mostra a coluna *95% Half Length* da Figura 31. Se esse valor (arredondado) for subtraído ao limite inferior e somado ao limite superior, tem-se que o término mais cedo ocorre na data 32,37 e o término mais tarde na data 61,63. Desta forma, considerando o intervalo de confiança de 95%, os valores obtidos na simulação ainda pertencem ao intervalo de datas [18,76], apresentado na Tabela 7, definido pelo método analítico.

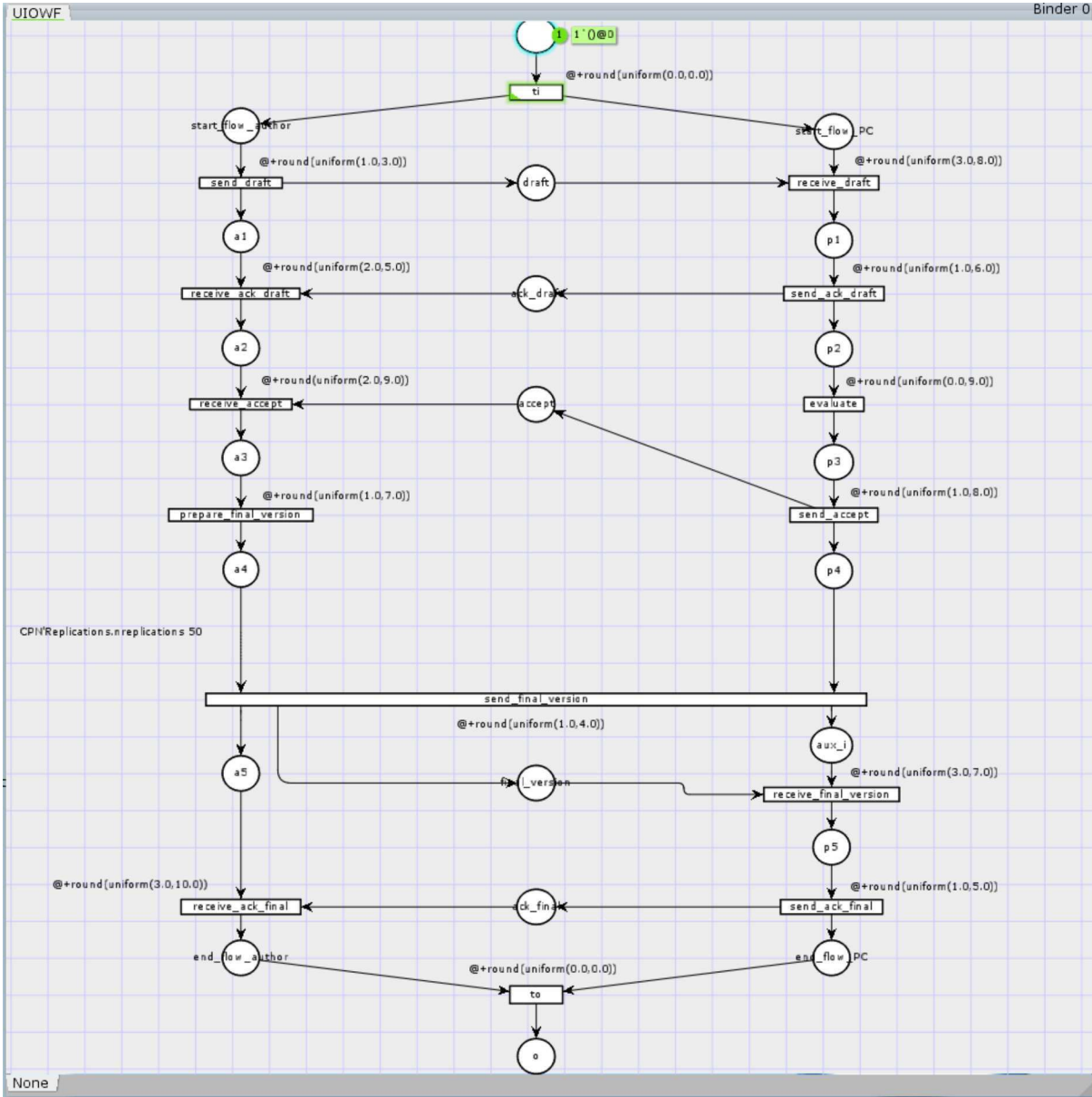


Figura 30 – Implementação no CPN Tools do cenário Sc_4 depois da regra de sincronização.

Como apresentado nesta seção, o cenário Sc_4 , mesmo após a aplicação da regra de sincronização, pode ter a data simbólica de produção do átomo o igual ao do cenário antes da alteração. Dessa forma, tanto ao calcular os intervalos de datas numéricas quanto ao simular o cenário no *software* CPN Tools, é possível verificar que para este exemplo, devido a estrutura da rede de Petri, a aplicação da regra de sincronização para o caso específico do modelo da Figura 7 não altera o tempo de finalização do processo global para o cenário Sc_4 .

É possível concluir através da análise quantitativa feita nesta seção, que a aplicação da regra de sincronização no modelo não vai gerar necessariamente uma degradação de

desempenho. O atraso após a utilização do método poderá eventualmente ocorrer, dependendo da estrutura da rede de Petri analisada e também dos tempos numéricos associados às transições do modelo.

CPN Tools Performance Report

Net: /cygdrive/C/Users/Vinicius/Desktop/CPN/sc4r.cpn

Number of replications: 50

Statistics							
Name	Avrg	90% Half Length	95% Half Length	99% Half Length	StD	Min	Max
ParadaO							
count_iid	14.000000	0.000000	0.000000	0.000000	0.000000	14	14
max_iid	45.820000	1.360614	1.632898	2.184739	5.713178	34	60
min_iid	0.000000	0.000000	0.000000	0.000000	0.000000	0	0
sum_iid	275.460000	9.112700	10.936322	14.632268	38.263965	169	371
avrg_iid	19.675714	0.650907	0.781166	1.045162	2.733140	12.071429	26.500000

Figura 31 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_4 depois da regra de sincronização.

Estudo de Caso

Existe uma crescente aceitação da Arquitetura Orientada a Serviço (do inglês, *Service-Oriented Architecture* (SOA)) como um paradigma para a integração de diferentes componentes de *software*, dentro e fora dos limites organizacionais, usando protocolos de *Internet* (KLAI; OCHI; TATA, 2013) (XIONG; FAN; ZHOU, 2010). Estes componentes, chamados *Web Services* (WS), estão disponíveis em ambientes distribuídos na *Internet*. Como consequência, as organizações se esforçam para fornecer os seus próprios serviços através de tarefas complexas que podem ser resolvidas usando uma combinação (composição) de vários WS (KLAI; OCHI; TATA, 2013).

Em muitos casos, múltiplos WS precisam interagir dinamicamente para atender às exigências do solicitante do serviço (XIONG; FAN; ZHOU, 2010). Orquestração e coreografia são dois mecanismos de interação diferentes de um WS (BARROS; DUMAS; OAKS, 2006). O primeiro considera um serviço particular que direciona a ordem lógica de todos os outros serviços, enquanto o outro considera o caso onde serviços individuais trabalham juntos em uma rede fracamente acoplada (XIONG; FAN; ZHOU, 2010) (BARROS; DUMAS; OAKS, 2006).

Quando dois ou mais serviços modelados por redes de Petri são compostos, os lugares de entrada e saída comuns aos serviços são combinados e dão origem a nova interface do sistema (MARTENS, 2005). Mesmo quando os serviços possuem uma interface sintaticamente compatível, o processo resultante distribuído pode levar a uma situação de *deadlock* (MARTENS, 2005). De acordo com Klai, Ochi e Tata (2013), isto é devido ao fato da construção dos fluxos de controle dos WS terem sido projetados de uma forma que garante que a execução “individual” de serviço não possua nenhum *deadlock*, porém algumas combinações de atividades estruturadas que interagem através de estruturas assíncronas podem levar o processo a situações onde algumas atividades são “inacessíveis”.

Muitos estudos já consideraram as redes de Petri como um modelo apropriado para especificação e análise de composição de WS. Em Martens (2005), um *framework* para modelagem e análise de WS baseado em processos de negócio com a ajuda das redes de Petri é apresentado. Nesta abordagem, os serviços, chamados de módulos, são classificados

como utilizável ou não utilizável (módulos que podem ser usados em uma composição). Em Klai, Ochi e Tata (2013), os autores endereçam o problema de abstração e conferência da correção da composição dos WS, tomando em consideração quatro variantes da propriedade Soundness (AALST et al., 2011) (*Soundness*, *Weak Soundness*, *Relaxed Soundness* e *Easy Soundness*). Para informação adicional a respeito das variações da propriedade *Soundness*, consultar Aalst et al. (2011). Em Passos e Julia (2015), uma abordagem é apresentada para garantir que os cenários livres de *deadlock* existentes na composição de WS garantam o relacionamento do negócio, evitando situações particulares de *deadlock*. Contudo, em tal abordagem, os cenários que podem levar o sistema a uma situação de *deadlock* não são removidos da arquitetura.

Assim, considerando a regra de sincronização apresentada no Capítulo 4, a proposta apresentada nesta pesquisa é aplicada através de um estudo de caso ilustrativo que utiliza o método formalizado no contexto dos processos de *workflow* interorganizacionais ao contexto das composições de WS.

Primeiramente, as definições de módulo e sistema composto são apresentadas na seção 5.1, com o objetivo de fazer a relação direta entre a modelagem de processos de *workflow* e a modelagem de WS. Na seção 5.2 é aplicado o método para remoção de *deadlocks* em composição de WS. Na seção 5.3 é realizada a análise qualitativa (verificação da propriedade *sound*) do modelo antes e depois da aplicação da regra de sincronização. Por fim, na seção 5.4 é realizada a análise quantitativa (cálculo de datas simbólicas de início e fim das atividades do processo) do modelo antes e depois da aplicação da regra de sincronização.

5.1 Módulo de *Workflow* e Sistema Composto

Segundo Martens (2005), um *Web Service* consiste em estruturas internas que executam processos locais e uma interface que permite a comunicação com outros *Web Services*. Assim, um WS pode ser modelado através de uma *WorkFlow net* (AALST et al., 2011) suplementada por uma interface; tal modelo é chamado de um módulo de *workflow* (ou apenas módulo de uma maneira resumida) (MARTENS, 2005).

A definição formal de módulo de *workflow* é apresentada na sequência (MARTENS, 2005).

Definição 12. (Módulo de *workflow*) Uma rede de Petri finita $M = (P, T, F)$ é chamada de um módulo de *workflow*, se e somente se:

- a) o conjunto de lugares é dividido em três conjuntos disjuntos: lugares internos P^N , lugares de entrada P^I e lugares de saída P^O ;
- b) a relação de fluxo é dividida em fluxo interno $F^N \subseteq (P^N \times T) \cup (T \times P^N)$ e fluxo de comunicação $F^C \subseteq (P^I \times T) \cup (T \times P^O)$;

- c) a rede $N(M) = (P^N, T, F^N)$ é uma *WorkFlow net*;
- d) nenhuma transição é conectada a um lugar de entrada e a um lugar de saída ao mesmo tempo.

Para ilustrar o conceito definido acima, é considerado um processo adaptado do exemplo apresentado em Xiong, Fan e Zhou (2010), que representa os *Web Services* responsáveis pela compra virtual de bilhetes, conforme mostrado na Figura 32.

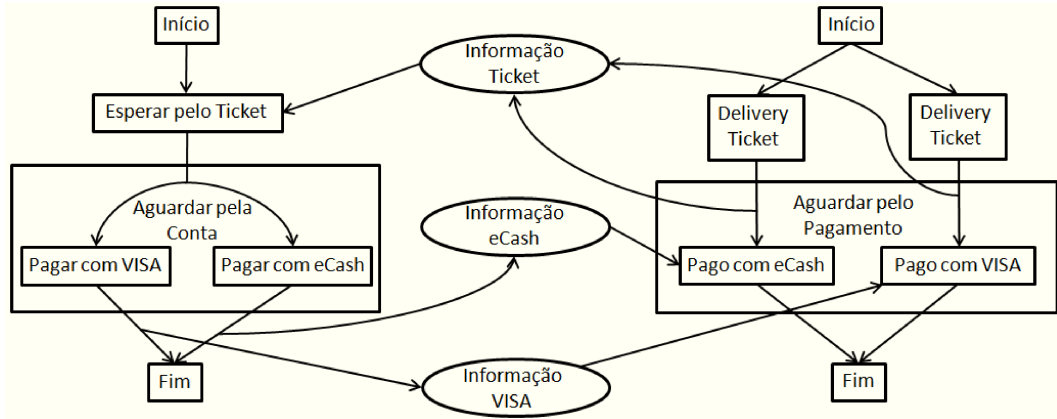


Figura 32 – Processo para compra virtual de bilhetes.

A Figura 33 modela o processo apresentado na Figura 32 através da utilização de uma *WF-net*. O módulo C1 representa um cliente e o módulo T1 modela um serviço de *Ticket*. Os lugares internos do módulo C1 são os lugares $q0$, $q1$ e $q2$. O lugar de entrada de C1 é *Ticket* e os lugares de saída desse módulo são *VISA* e *eCash*. Considerando o módulo T1, os seus lugares internos são $p0$, $p1$, $p2$ e $p3$. O lugar de saída de T1 é *Ticket* e os lugares de entrada desse módulo são *VISA* e *eCash*. As transições t_1 , t_2 , t_3 , t_4 , t_5 , t_6 e t_7 representam as atividades do processo modelado na Figura 32.

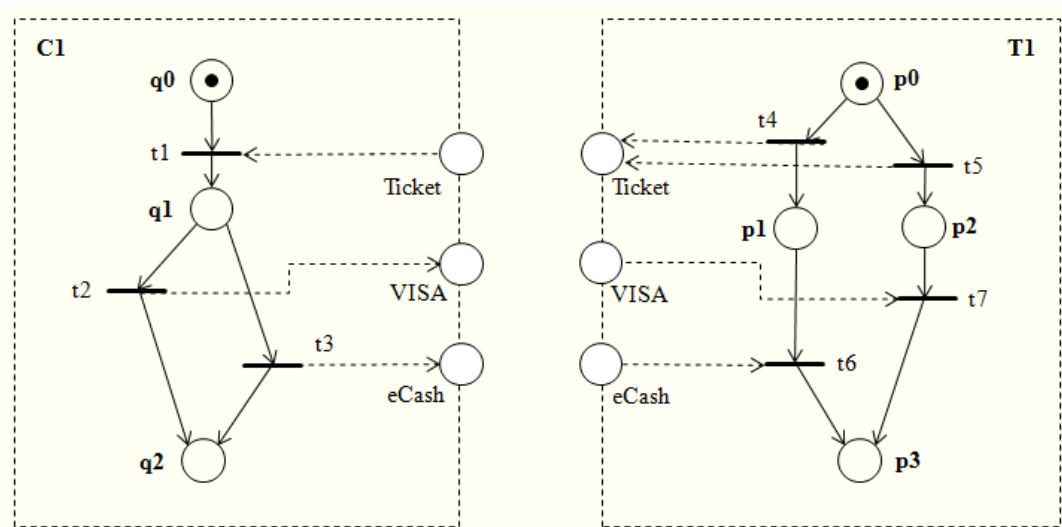


Figura 33 – Módulos de *workflow* Cliente (C1) e *Ticket* (T1).

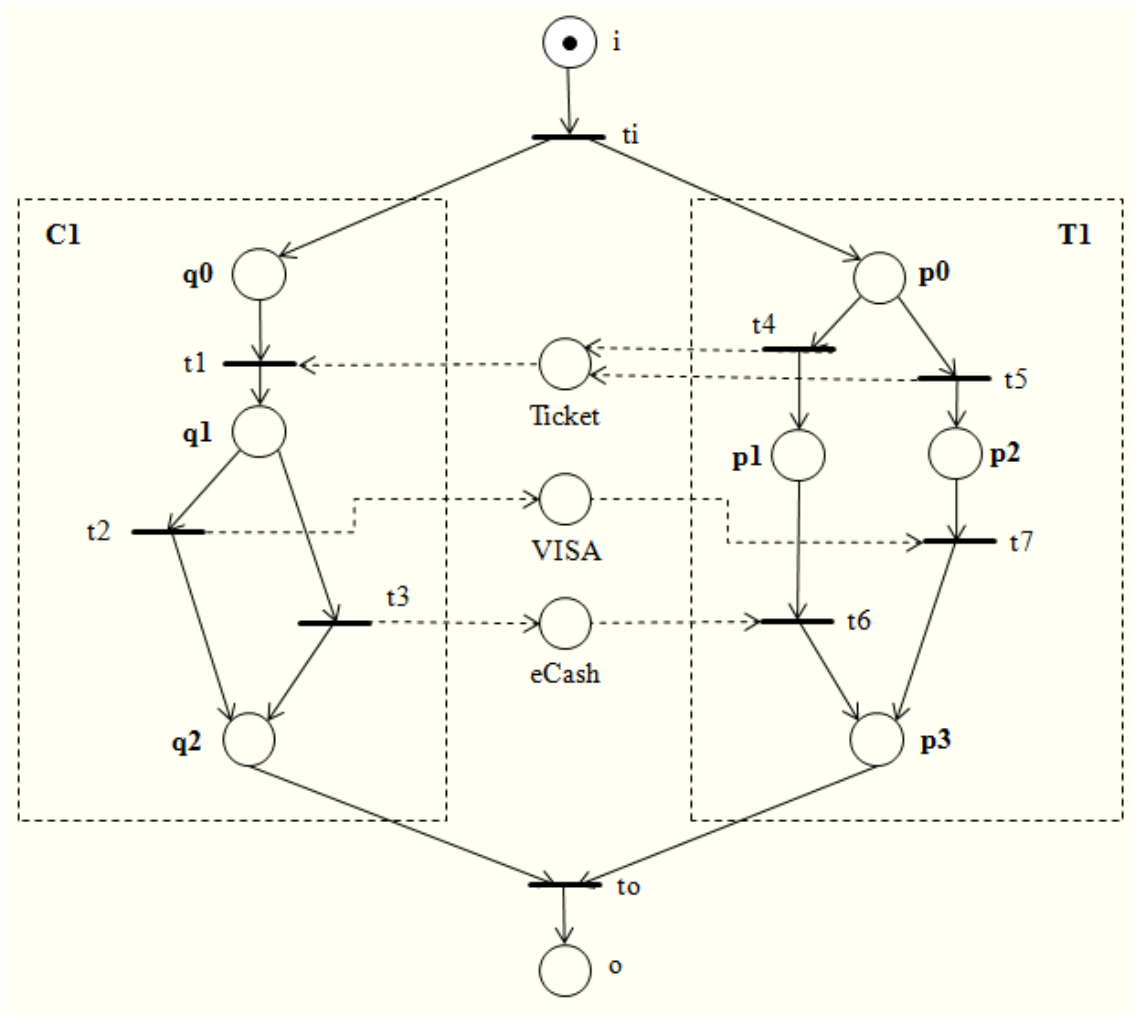
De acordo com Martens (2005), dois módulos são chamados de sintaticamente compatíveis se ambos processos internos são disjuntos e cada lugar comum é um lugar de saída de um módulo e um lugar de entrada do outro. Considerando os módulos C1 e T1 apresentados na Figura 33, é possível perceber que estes módulos são sintaticamente compatíveis. Na abordagem apresentada em Martens (2005), quando dois módulos são compostos, os seus lugares em comum são fundidos e os lugares de entrada e saída intermediários tornam-se a nova interface. Para obter um módulo de *workflow* sintaticamente correto, é necessário adicionar novos componentes para início e término do processo. Neste contexto, Martens (2005) definiu o conceito de sistema composto apresentado formalmente a seguir.

Definição 13. (Sistema composto) Seja $A = (P_a, T_a, F_a)$ e $B = (P_b, T_b, F_b)$ dois módulos sintaticamente compatíveis. Sejam $i, o \notin (P_a \cup P_b)$ dois novos lugares, e $t_i, t_o \notin (T_a \cup T_b)$ duas novas transições. O sistema composto $A \oplus B$ é dado por (P_s, T_s, F_s) , tal que $P_s = P_a \cup P_b \cup \{i, o\}$, $T_s = T_a \cup T_b \cup \{t_i, t_o\}$ e $F_s = F_a \cup F_b \cup \{(i, t_i), (t_i, \alpha_a), (t_i, \alpha_b), (\omega_a, t_o), (\omega_b, t_o)\}, (t_o, o)$.

A Figura 34 apresenta o sistema composto $C1 \oplus T1$ dos módulos de *workflow* C1 e T1 apresentados na Figura 33. Note que o serviço *Ticket* (T1) resolve um conflito interno e envia a mensagem *Ticket* ao módulo Cliente (C1). Consequentemente, o módulo T1 está no estado $p1$ esperando por *eCash* ou no estado $p2$ esperando por *VISA*. O cliente (C1) recebe o *Ticket* e pode escolher entre outros tipos de pagamento, *VISA* ou *eCash*. É importante notar que o módulo C1 não conhece o estado interno do módulo T1, ou seja, $p1$ ou $p2$. De acordo com Martens (2005), quando uma decisão interna é feita e não é comunicada adequadamente a um WS externo, um problema já bem conhecido na literatura ocorre, o problema de escolha não local, que pode ser responsável pela ocorrência de situações de *deadlock*.

De acordo com Passos (2016), comparando a definição de um módulo de *workflow* apresentada por Martens (2005) e a definição de uma IOWF-*net* apresentada Aalst (1998), é possível observar que um módulo de *workflow* corresponde a uma *WorkFlow net* Local (LWF-*net*) adicionada de sua estrutura de comunicação (lugares e fluxos de comunicação). Além disso, é possível notar que a definição de um sistema composto é equivalente à definição de uma $U(\text{IOWF-net})$ que considera apenas duas LWF-*nets* (PASSOS, 2016). Assim, o método apresentado na presente pesquisa, para remoção de situações de *deadlocks* em *workflow* interorganizacionais modelados por IOWF-*nets*, pode ser diretamente aplicado à remoção de situações de *deadlocks* em composições de *Web Services*. É importante destacar ainda que Martens (2005) afirma que a composição de mais de dois módulos de *workflow* é realizada por aplicação recorrente da composição em pares.

Assim, a seção 5.2 apresenta a aplicação da regra de sincronização proposta no Capítulo 4 no contexto de composições de *Web Services*.

Figura 34 – O sistema composto $C1 \oplus T1$.

5.2 Regra de sincronização para prevenção de *deadlock* em Composição de *Web Services*

A presente seção apresenta a aplicação do método para remoção de situações de *deadlock*, proposto no Capítulo 4, ao contexto das composições de *Web Services*.

O sistema composto apresentado na Figura 34 é considerado para a aplicação do método proposto. Como mostrado em (MARTENS, 2005), os módulos *Costumer* (C1) e *Ticket* (T1) que formam o sistema composto, apresentados na Figura 33, são livres de *deadlock* mas o sistema composto da Figura 34 não é livre de *deadlock*.

Para o sistema composto considerado, existem quatro diferentes cenários para serem estudados. O primeiro cenário, Sc_1 , onde as tarefas *Pagar com eCash* e *Entregar Ticket* serão executadas (disparo das transições t_3 e t_4). O segundo cenário, Sc_2 , onde as tarefas *Pagar com VISA* e *Entregar Ticket* serão executadas (disparo das transições t_2 e t_5). O terceiro cenário, Sc_3 , onde as tarefas *Pagar com VISA* e *Entregar Ticket* serão executadas (disparo das transições t_2 e t_4). Finalmente, o quarto cenário, Sc_4 , onde as tarefas *Pagar*

com *eCash* e *Entregar Ticket* serão executadas (disparo das transições t_3 e t_5).

Cada um destes cenários é então representado por um sequente específico da Lógica Linear que considera as marcações iniciais e finais do sistema composto e uma lista não ordenada de transições envolvidas nele (PASSOS; JULIA, 2015).

As transições do sistema composto mostrado na Figura 34 são representadas pelas seguintes fórmulas da Lógica Linear:

$$\begin{aligned}
 t_i &= i \multimap q0 \otimes p0, \\
 t_1 &= q0 \otimes Ticket \multimap q1, \\
 t_2 &= q1 \multimap VISA \otimes q2, \\
 t_3 &= q1 \multimap eCash \otimes q2, \\
 t_4 &= p0 \multimap Ticket \otimes p1, \\
 t_5 &= p0 \multimap Ticket \otimes p2, \\
 t_6 &= p1 \otimes eCash \multimap p3, \\
 t_7 &= p2 \otimes VISA \multimap p3, \\
 t_o &= q2 \otimes p3 \multimap o.
 \end{aligned}$$

Os quatro diferentes cenários e consequentemente sequentes da Lógica Linear são, portanto, os seguintes:

$$\begin{aligned}
 Sc_1 &= i, t_i, t_1, t_3, t_4, t_6, t_o \vdash o, \\
 Sc_2 &= i, t_i, t_1, t_2, t_5, t_7, t_o \vdash o, \\
 Sc_3 &= i, t_i, t_1, t_2, t_4, t_6, t_o \vdash o, \\
 Sc_4 &= i, t_i, t_1, t_3, t_5, t_7, t_o \vdash o.
 \end{aligned}$$

A seguir são apresentadas as árvores de prova para cada um destes cenários. Para uma melhor visualização da abordagem, a árvore de prova para o cenário Sc_1 foi dividida em duas partes, lembrando que a árvore de prova da Lógica Linear é lida de baixo para cima. A Parte 1 da árvore de prova para o cenário Sc_1 é a seguinte:

$$\begin{array}{c}
 \vdots \\
 \frac{q0, Ticket, p1, q0 \otimes Ticket \multimap q1, t_3, t_6, t_o \vdash o}{p0 \vdash p0 \quad q0, Ticket \otimes p1, t_1, t_3, t_6, t_o \vdash o} \otimes_L \\
 \frac{q0, p0, t_1, t_3, p0 \multimap Ticket \otimes p1, t_6, t_o \vdash o}{i \vdash i \quad q0 \otimes p0, t_1, t_3, t_4, t_6, t_o \vdash o} \multimap_L \\
 \frac{i, i \multimap q0 \otimes p0, t_1, t_3, t_4, t_6, t_o \vdash o}{i, i \multimap q0 \otimes p0, t_1, t_3, t_4, t_6, t_o \vdash o} \multimap_L
 \end{array}$$

A Parte 2 da árvore para o cenário Sc_1 é a seguinte:

$$\begin{array}{c}
\frac{\frac{q2 \vdash q2 \quad p3 \vdash p3}{q2, p3 \vdash q2 \otimes p3} \otimes_R \quad o \vdash o}{\neg_L} \\
\frac{\frac{p1 \vdash p1 \quad eCash \vdash eCash}{p1, eCash \vdash p1 \otimes eCash} \otimes_R \quad q2, p3, q2 \otimes p3 \multimap o \vdash o}{\neg_L} \\
\frac{}{p1, eCash, q2, p1 \otimes eCash \multimap p3, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p1, eCash \otimes q2, t_6, t_o \vdash o}{\neg_L} \\
\frac{\frac{q0 \vdash q0 \quad Ticket \vdash Ticket}{q0, Ticket \vdash q0 \otimes Ticket} \otimes_R \quad p1, q1, q1 \multimap eCash \otimes q2, t_6, t_o \vdash o}{\neg_L} \\
\vdots
\end{array}$$

Para uma melhor análise da abordagem, a árvore de prova da Lógica Linear para o cenário Sc_2 também foi dividida em duas partes. A Parte 1 da árvore de prova para o cenário Sc_2 é a seguinte:

$$\begin{array}{c}
\vdots \\
\frac{q0, Ticket, p2, q0 \otimes Ticket \multimap q1, t_2, t_7, t_o \vdash o}{\neg_L} \\
\frac{p0 \vdash p0 \quad q0, Ticket \otimes p2, t_1, t_2, t_7, t_o \vdash o}{\neg_L} \\
\frac{q0, p0, t_1, t_2, p0 \multimap Ticket \otimes p2, t_7, t_o \vdash o}{\neg_L} \\
\frac{i \vdash i \quad q0 \otimes p0, t_1, t_2, t_5, t_7, t_o \vdash o}{\neg_L} \\
i, i \multimap q0 \otimes p0, t_1, t_2, t_5, t_7, t_o \vdash o
\end{array}$$

A Parte 2 da árvore de prova para o cenário Sc_2 é a seguinte:

$$\begin{array}{c}
\frac{\frac{q2 \vdash q2 \quad p3 \vdash p3}{q2, p3 \vdash q2 \otimes p3} \otimes_R \quad o \vdash o}{\neg_L} \\
\frac{\frac{p2 \vdash p2 \quad VISA \vdash VISA}{p2, VISA \vdash p2 \otimes VISA} \otimes_R \quad q2, p3, q2 \otimes p3 \multimap o \vdash o}{\neg_L} \\
\frac{}{p2, VISA, q2, p2 \otimes VISA \multimap p3, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p2, VISA \otimes q2, t_7, t_o \vdash o}{\neg_L} \\
\frac{\frac{q0 \vdash q0 \quad Ticket \vdash Ticket}{q0, Ticket \vdash q0 \otimes Ticket} \otimes_R \quad p2, q1, q1 \multimap VISA \otimes q2, t_7, t_o \vdash o}{\neg_L} \\
\vdots
\end{array}$$

A árvore de prova para o cenário Sc_3 é a seguinte:

$$\begin{array}{c}
\frac{}{p1, \mathbf{VISA}, q2, t_6, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p1, \mathbf{VISA} \otimes q2, t_6, t_o \vdash o}{\quad} \multimap_L \\
\frac{\frac{q0 \vdash q0 \quad Ticket \vdash Ticket}{q0, Ticket \vdash q0 \otimes Ticket} \otimes_R \quad p1, q1, q1 \multimap \mathbf{VISA} \otimes q2, t_6, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, Ticket, p1, q0 \otimes Ticket \multimap q1, t_2, t_6, t_o \vdash o} \otimes_L \\
\frac{p0 \vdash p0 \quad q0, Ticket \otimes p1, t_1, t_2, t_6, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, p0, t_1, t_2, p0 \multimap Ticket \otimes p1, t_6, t_o \vdash o} \otimes_L \\
\frac{i \vdash i \quad q0 \otimes p0, t_1, t_2, t_4, t_6, t_o \vdash o}{\quad} \multimap_L \\
i, i \multimap q0 \otimes p0, t_1, t_2, t_4, t_6, t_o \vdash o
\end{array}$$

E, finalmente, a árvore de prova para o cenário Sc_4 é dada por:

$$\begin{array}{c}
\frac{}{p2, \mathbf{eCash}, q2, t_7, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p2, \mathbf{eCash} \otimes q2, t_7, t_o \vdash o}{\quad} \multimap_L \\
\frac{\frac{q0 \vdash q0 \quad Ticket \vdash Ticket}{q0, Ticket \vdash q0 \otimes Ticket} \otimes_R \quad p2, q1, q1 \multimap \mathbf{eCash} \otimes q2, t_7, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, Ticket, p2, q0 \otimes Ticket \multimap q1, t_3, t_7, t_o \vdash o} \otimes_L \\
\frac{p0 \vdash p0 \quad q0, Ticket \otimes p2, t_1, t_3, t_7, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, p0, t_1, t_3, p0 \multimap Ticket \otimes p2, t_7, t_o \vdash o} \otimes_L \\
\frac{i \vdash i \quad q0 \otimes p0, t_1, t_3, t_5, t_7, t_o \vdash o}{\quad} \multimap_L \\
i, i \multimap q0 \otimes p0, t_1, t_3, t_5, t_7, t_o \vdash o
\end{array}$$

O último sequente nas árvores de prova para os cenários Sc_1 e Sc_2 é $o \vdash o$; então estes cenários são livres de *deadlock*. O último sequente para o cenário Sc_3 é $p1, \mathbf{VISA}, q2, t_6, t_o \vdash o$, e o último sequente para o cenário Sc_4 é $p2, \mathbf{eCash}, q2, t_7, t_o \vdash o$; então os cenários Sc_3 e Sc_4 são os cenários onde as situações de *deadlock* ocorrem. É necessário então aplicar a regra de sincronização em cada um desses cenários para remover a situação de *deadlock*.

Considerando primeiramente o cenário Sc_3 , a última transição que foi disparada antes da situação de *deadlock* é a transição t_2 , marcada em negrito na árvore de prova. Em particular, os átomos $q2$ e \mathbf{VISA} do último sequente da árvore de prova para o cenário Sc_3 são produzidos quando a transição t_2 é disparada. Entretanto, devido à situação de *deadlock*, estes átomos não são consumidos até o final da prova. Consequentemente, a transição t_2 vai corresponder à transição t_{d1} da abordagem, a transição t_7 à transição t_{d2} , e o átomo \mathbf{VISA} ao lugar de comunicação assíncrono marcado (interface).

O lugar de comunicação assíncrona \mathbf{VISA} , presente no último sequente da árvore de prova do cenário Sc_3 , é um lugar de saída da transição t_2 (última transição disparada na

árvore de prova). Este lugar de comunicação é um lugar de entrada da transição t_7 do módulo T1. Portanto, a transição t_7 corresponde, no cenário Sc_3 , à uma transição morta.

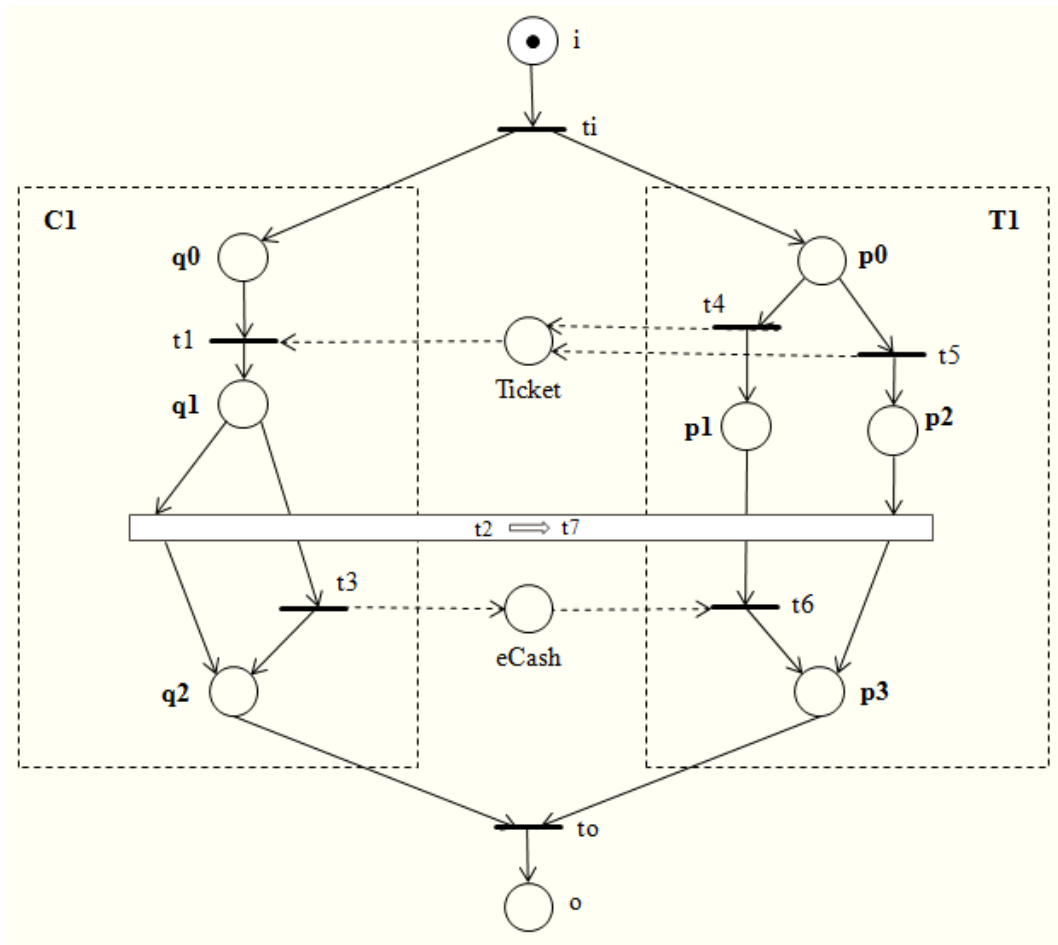


Figura 35 – Aplicando a regra de sincronização para o cenário Sc_3 do sistema composto apresentado na Figura 34.

A Figura 35 apresenta a aplicação da regra de sincronização no cenário Sc_3 do sistema composto apresentado na Figura 34. A regra de sincronização, apresentada na Figura 35, não é uma transição verdadeira e sim uma regra de transição que sincroniza parte da estrutura de comunicação entre os módulos C1 e T1. A transformação do mecanismo de comunicação assíncrono puro em um mecanismo de sincronização parcial, depois da aplicação da regra de sincronização é apresentada na Figura 36. Na Figura 36A, o disparo da transição t_2 (transição do tipo t_{d1}) corresponde somente a uma condição necessária para o disparo da transição t_7 (transição do tipo t_{d2}). Tal protocolo de comunicação assíncrono não fornece a garantia para o recebimento da mensagem enviada. Em particular, para o cenário Sc_3 , depois do disparo da transição t_2 , uma ficha permanecerá presa no lugar de comunicação *VISA*.

Por outro lado, na Figura 36B, o fato de sincronizar a execução da primeira atividade associada à transição t_2 (transição do tipo t_{d1}) em ambos módulos C1 e T1, corresponde à garantia da execução da atividade t_7 (transição do tipo t_{d2}). De fato, o disparo da

transição do tipo t_{d1} , depois da aplicação da regra de sincronização, corresponde a uma condição necessária e suficiente para o disparo da transição do tipo t_{d2} .

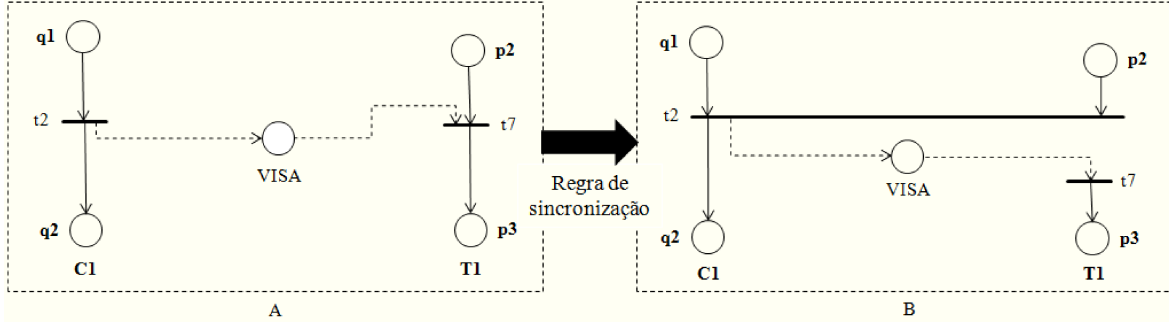


Figura 36 – Antes e depois da regra de sincronização no cenário Sc_3 .

Depois da aplicação da regra de sincronização, a situação de *deadlock* é removida e o cenário Sc_3 não ocorrerá mais no sistema composto. Na prática, o cenário Sc_3 é então removido de cenários potenciais que existem quando se considera o sistema composto apresentado na Figura 34.

De acordo com a Figura 36B, as transições t_2 e t_7 do sistema composto são modificados depois da regra de sincronização. Elas são representadas agora pelas seguintes novas fórmulas da Lógica Linear:

$$t_2 = q1 \otimes p2 \multimap VISA \otimes q2,$$

$$t_7 = VISA \multimap p3.$$

As transições t_2 e t_7 não aparecem nos sequentes da Lógica Linear do cenário Sc_1 e Sc_4 , e a alteração do modelo não modifica as árvores de prova dos cenários correspondentes. Por outro lado, ambas transições aparecem no cenário Sc_2 . A Parte 1 da árvore de prova para o cenário Sc_2 continua inalterado depois da regra de sincronização. A Parte 2 da árvore de prova precisa ser processada novamente e se torna então:

$$\frac{\frac{\frac{q2 \vdash q2 \quad p3 \vdash p3}{q2, p3 \vdash q2 \otimes p3} \otimes_R \quad o \vdash o}{o \vdash o} \multimap_L}{VISA \vdash VISA \quad q2, p3, q2 \otimes p3 \multimap o \vdash o} \multimap_L$$

$$\frac{VISA, q2, VISA \multimap p3, t_o \vdash o}{VISA \otimes q2, t_7, t_o \vdash o} \otimes_L$$

$$\frac{\frac{\frac{q1 \vdash q1 \quad p2 \vdash p2}{q1, p2 \vdash q1 \otimes p2} \otimes_R \quad VISA \otimes q2, t_7, t_o \vdash o}{p2, q1, q1 \otimes p2 \multimap VISA \otimes q2, t_7, t_o \vdash o} \multimap_L}{\frac{q0 \vdash q0 \quad Ticket \vdash Ticket}{q0, Ticket \vdash q0 \otimes Ticket} \otimes_R \quad p2, q1, q1 \otimes p2 \multimap VISA \otimes q2, t_7, t_o \vdash o} \multimap_L$$

$$\vdots$$

O último sequente da árvore de prova para o novo cenário Sc_2 é $o \vdash o$; como consequência, depois da aplicação da regra de sincronização, o cenário Sc_2 ainda é um cenário livre de *deadlock*.

Uma vez que o modelo possui mais de um cenário com *deadlock*, é necessário também aplicar a regra de sincronização ao cenário Sc_4 . Considerando agora o cenário Sc_4 , a última transição que foi disparada antes da situação de *deadlock* é a transição t_3 marcada em negrito na árvore de prova. Em particular, os últimos átomos $q2$ e $eCash$ do último sequente da árvore de prova para o cenário Sc_4 são produzidos quando a transição t_3 é disparada. Entretanto, devido à situação de *deadlock*, estes átomos não são consumidos até o fim da árvore de prova. Portanto, a transição t_3 vai corresponder a uma transição do tipo t_{d1} , a transição t_6 a uma transição do tipo t_{d2} , e o átomo $eCash$ a um lugar de comunicação assíncrona marcado.

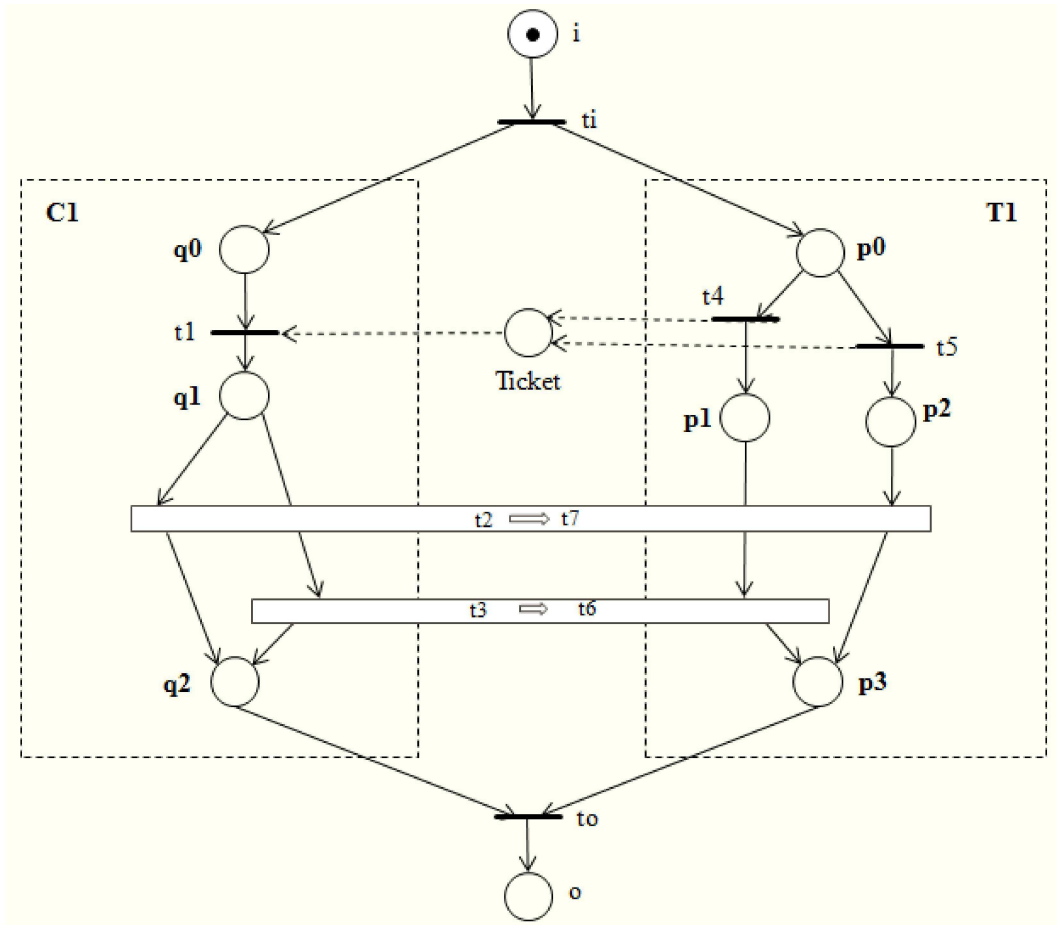


Figura 37 – Aplicando a regra de sincronização para o cenário Sc_4 do sistema composto apresentado na Figura 34.

O lugar de comunicação assíncrona $eCash$ presente no último sequente da árvore de prova do cenário Sc_4 é um lugar de saída da transição t_3 (última transição disparada na árvore de prova). Este lugar de comunicação é um lugar de entrada da transição t_6 do módulo T1. Consequentemente, a transição t_6 corresponde, no cenário Sc_4 , à transição morta.

A Figura 37 apresenta a aplicação da regra de sincronização no cenário Sc_4 do sistema composto apresentado na Figura 34, considerando que a regra de sincronização já

foi aplicada uma vez nesse sistema composto para a remoção do cenário Sc_3 . Como mencionado no cenário Sc_3 , a regra de sincronização, apresentada na Figura 37, não é um transição verdadeira e sim uma regra de transição que sincroniza parte da estrutura de comunicação entre os módulos C1 e T1. A transformação do mecanismo de comunicação assíncrona puro em um mecanismo de sincronização parcial depois da aplicação da regra de sincronização é apresentada na Figura 38. Na Figura 38A, o disparo da transição t_3 (transição do tipo t_{d1}) corresponde somente a uma condição necessária para o disparo da transição t_6 (transição do tipo t_{d2}). Tal protocolo de comunicação assíncrona portanto, não fornece a garantia para o recebimento da mensagem enviada. Em particular, para o cenário Sc_4 , depois do disparo da transição t_3 , uma ficha permanecerá presa no lugar de comunicação $eCash$.

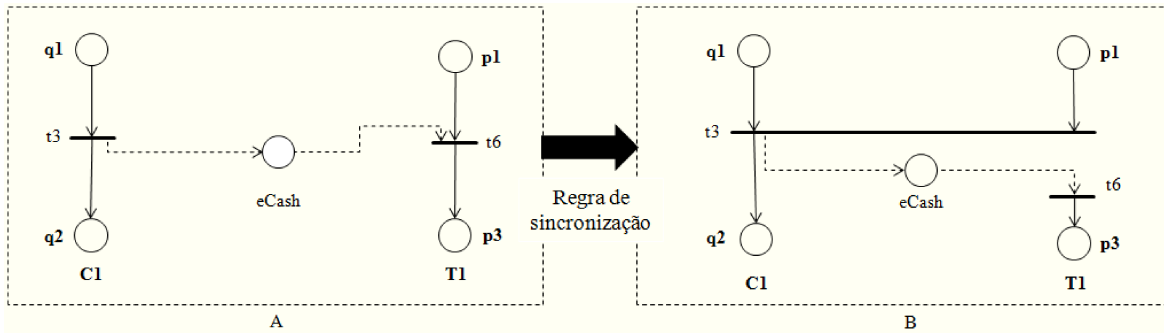


Figura 38 – Antes e depois da regra de sincronização no cenário Sc_4 .

Por outro lado, na Figura 38B, o fato de sincronizar a execução da primeira atividade associada à transição t_3 (transição do tipo t_{d1}) em ambos os módulos C1 e T1, corresponde à garantia da execução da atividade chamada t_6 (transição do tipo t_{d2}).

Depois da aplicação da regra de sincronização, a situação de *deadlock* é removida e o cenário Sc_4 não ocorrerá mais no sistema composto. Na prática, o cenário Sc_4 é então removido dos potenciais cenários que existem quando se considera o sistema composto como mostrado na Figura 34.

De acordo com a Figura 38B, as transições t_3 e t_6 do sistema composto são modificadas após aplicação da regra de sincronização. Elas são agora representadas pelas seguintes fórmulas da Lógica Linear:

$$t_3 = q1 \otimes p1 \multimap eCash \otimes q2,$$

$$t_6 = eCash \multimap p3.$$

As transições t_3 e t_6 não aparecem nos seguintes da Lógica Linear do cenário Sc_2 e a alteração do modelo não modifica a árvore de prova do cenário correspondente. Por outro lado, ambas transições aparecem no cenário Sc_1 . A Parte 1 da árvore de prova para o cenário Sc_1 continua inalterado depois da aplicação da regra de sincronização. A Parte 2 da árvore de prova precisa ser processada novamente e torna-se então:

A Figura 39 mostra o modelo final após a aplicação da regra de sincronização nos cenários Sc_3 e Sc_4 do sistema composto mostrado na Figura 34. Note que os lugares adicionais $aux1$ e $aux2$ são acrescentados no modelo final com o objetivo de respeitar a definição de módulo de *workflow* (de fato, as transições t_6 e t_7 precisam de pelo menos uma transição de entrada com o objetivo de respeitar a estrutura básica de uma *Work-Flow net*). Tais lugares, considerando a teoria da rede de Petri (MURATA, 1989), são lugares implícitos e não alteram as boas propriedades do modelo correspondente; então os resultados produzidos pelo estudo das árvores de prova da Lógica Linear (remoção dos cenários que levam a situações de *deadlock*) ainda serão válidos. Em particular, os cenários Sc_3 e Sc_4 , que possuíam *deadlock*, são removidos do modelo final, ficando somente com dois cenários livres de *deadlock*, Sc_1 e Sc_2 . Lembrando aqui que os cenários Sc_1 e Sc_2 são levemente modificados depois da aplicação da regra de sincronização.

5.3 Análise Qualitativa de Composição de *Web Services*

A presente seção apresenta a validação da abordagem proposta do ponto de vista qualitativo, que considera a aplicação da regra de sincronização a uma composição de *Web Services* que não satisfaz o critério de correção *Soundness*, com o objetivo de eliminar as situações de *deadlock* presentes no modelo original, de forma que o critério *Soundness* seja satisfeito para as composições de *Web Services*.

Para realizar tal validação, o sistema composto da Figura 34, que equivale a uma $U(IOWF-net)$, é considerado. Além disso, para auxiliar na validação da abordagem proposta é utilizado o simulador CPN Tools (RATZER et al., 2003). Dessa forma, inicialmente, é necessário implementar o modelo no *software*. A Figura 40 apresenta a implementação do modelo da Figura 34 no CPN Tools.

No modelo implementado, apresentado na Figura 40, uma transição auxiliar t_{aux} , que tem como entrada o lugar final e como saída o lugar início, foi adicionada com o objetivo de realizar a análise qualitativa. Essa transição auxiliar foi adicionada no modelo, devido o método proposto para verificar a correção *Soundness* apresentado em Aalst e Hee (2004) que também é utilizado na validação da abordagem proposta nesta pesquisa. Nesta pesquisa, este método é utilizado particularmente para verificar a presença de situações de *deadlock* no modelo.

É necessário construir o grafo das marcações acessíveis para o modelo apresentado na Figura 40, que representa o modelo antes da aplicação da regra de sincronização, para verificar as situações de *deadlock* encontradas no sistema composto. Assim sendo, o grafo das marcações acessíveis que permite uma visualização gráfica dos estados alcançáveis a partir do estado inicial é apresentado na Figura 41.

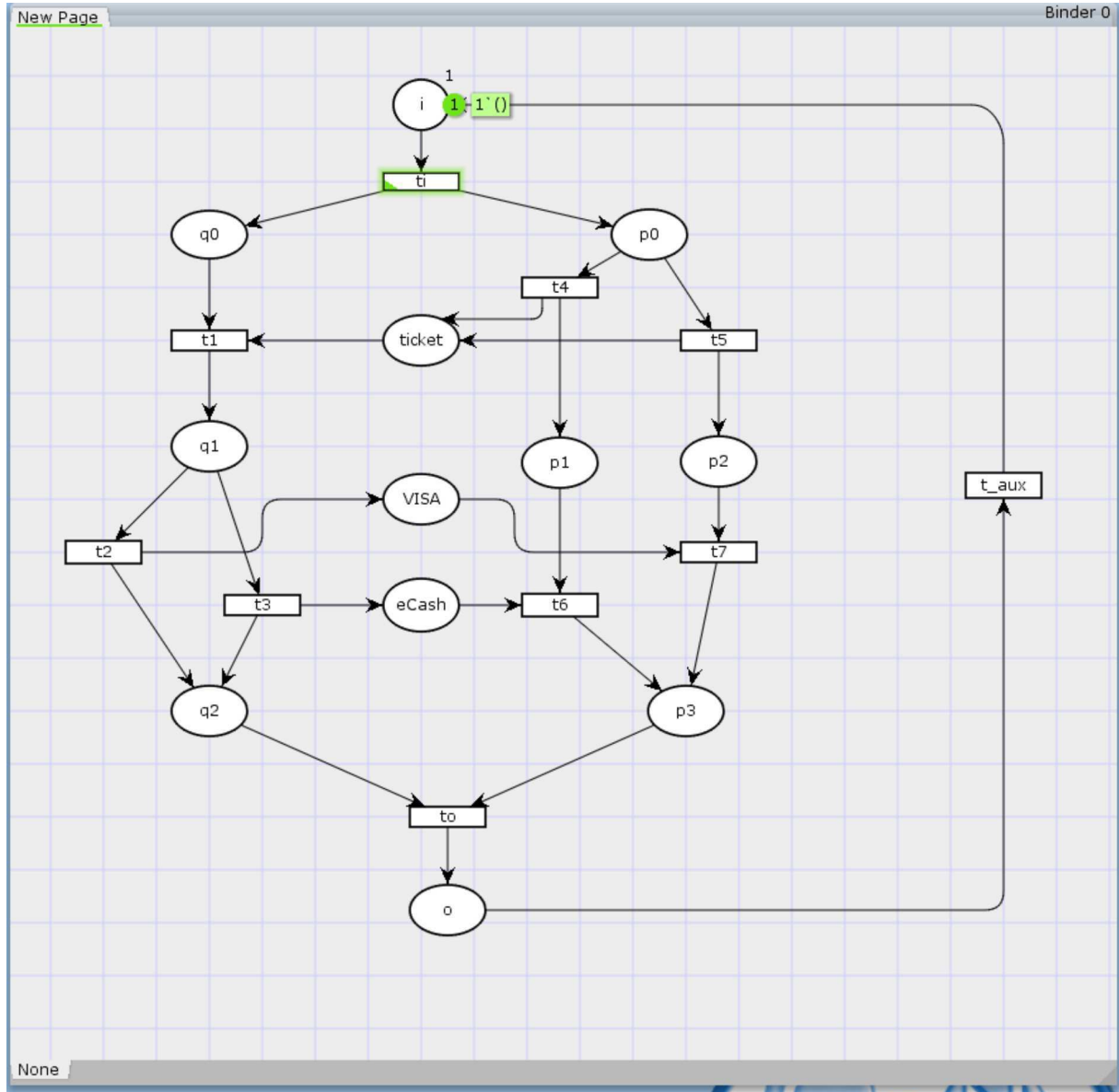


Figura 40 – Implementação no CPN Tools do modelo apresentado na Figura 34.

É possível verificar nos estados destacados com uma borda mais espessa no grafo das marcações acessíveis da Figura 41, representados pelas marcações 7 e 10, que duas situações de *deadlock* podem ser alcançadas, já que existem nessas marcações dois arcos de entrada e nenhum arco de saída. O primeiro estado de *deadlock* é alcançado no grafo quando as transições t_4 , t_1 e t_2 são disparadas, o que corresponde à execução do cenário Sc_3 onde essas mesmas transições são disparadas. Já o segundo estado de *deadlock* é alcançado no grafo quando as transições t_5 , t_1 e t_3 são disparadas, o que corresponde à execução do cenário Sc_4 , onde essas mesmas transições são disparadas.

O resultado da análise das boas propriedades do modelo apresentado na Figura 40, utilizando o simulador CPN Tools, é apresentado na Figura 42. Como o método proposto nessa pesquisa tem como principal objetivo a remoção da situação de *deadlock* no sistema

composto, é apresentado na Figura 42 o resultado da propriedade da Vivacidade.

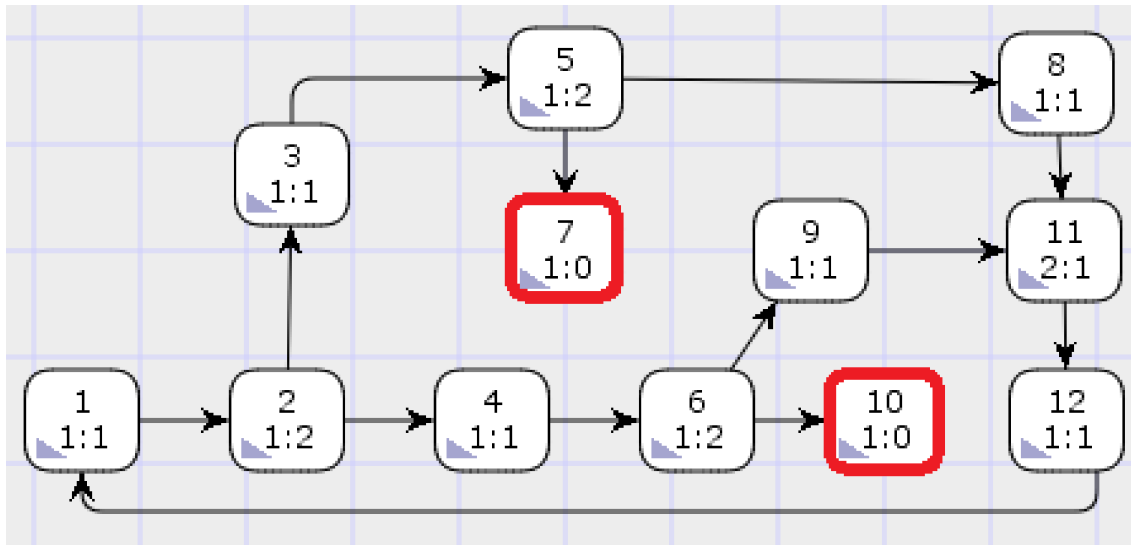


Figura 41 – Grafo das marcações acessíveis para a rede de Petri da Figura 34 antes da regra de sincronização, gerado pelo simulador CPN Tools.

Conforme é possível verificar nas informações da propriedade de Vivacidade do relatório gerado, as mesmas duas marcações mortas apresentadas no grafo da Figura 41, marcações 7 e 10, foram identificadas. Logo o modelo possui situações de *deadlock*, conforme apresentado para esse modelo, na seção 5.2.

Liveness Properties

Dead Markings

[7,10]

Dead Transition Instances

None

Live Transition Instances

None

Figura 42 – Relatório de Vivacidade para a rede de Petri da Figura 34 antes da regra de sincronização, gerado pelo simulador CPN Tools.

Após a aplicação da regra de sincronização algumas transições do modelo original são alteradas, fazendo com que um novo modelo seja gerado. Dessa forma, uma nova rede de Petri precisa ser implementada no *software*. A Figura 43 apresenta a implementação do novo modelo no CPN Tools.

Para verificar se o novo modelo proposto, após a aplicação da regra de sincronização, é livre de *deadlock*, um novo cálculo de espaço de estados (através da ferramenta *Calculate State Space*) deve ser feito.

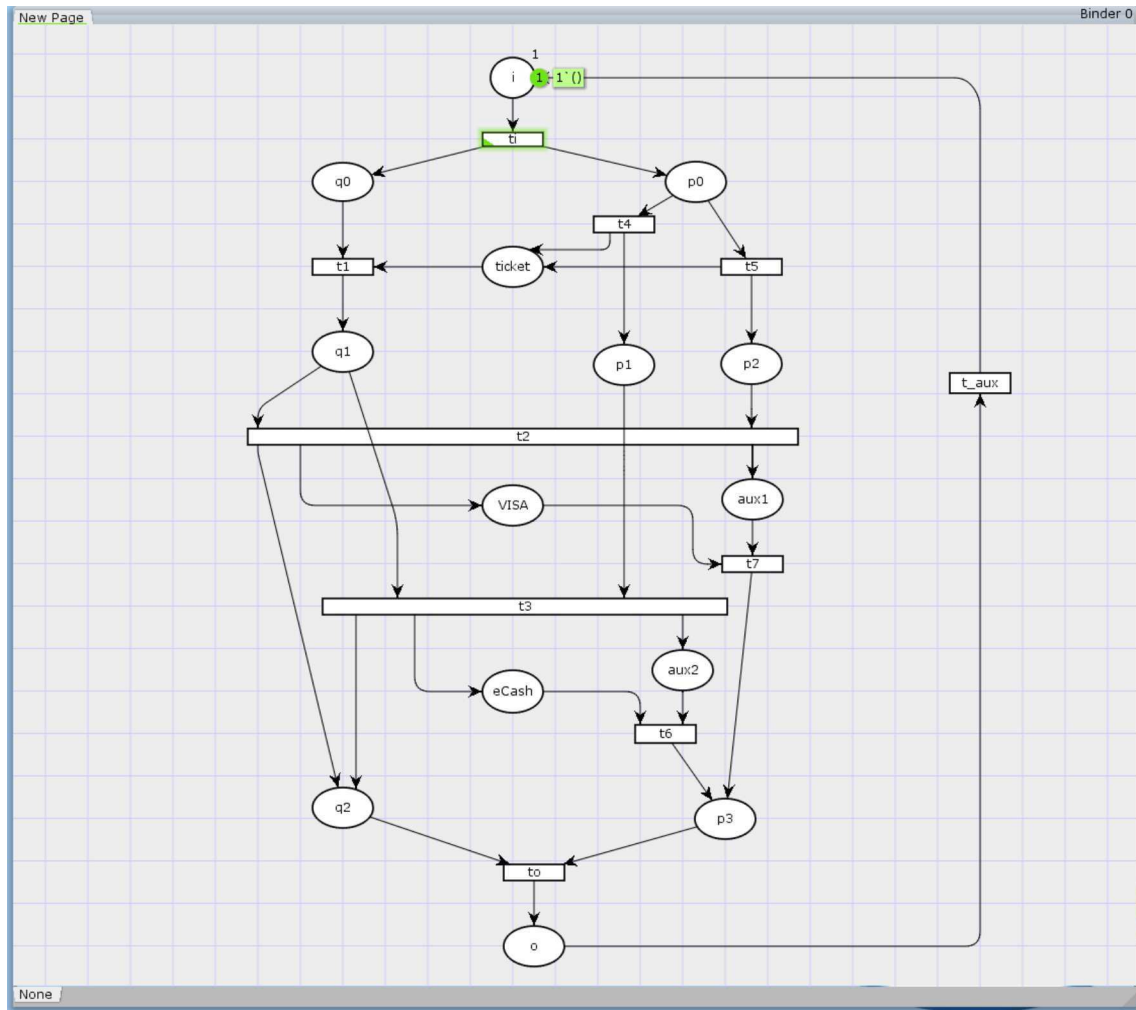


Figura 43 – Implementação no CPN Tools do modelo apresentado na Figura 39.

Na Figura 43, é apresentado o grafo das marcações acessíveis que permite uma visualização gráfica dos estados alcançáveis a partir do estado inicial da Figura 44.

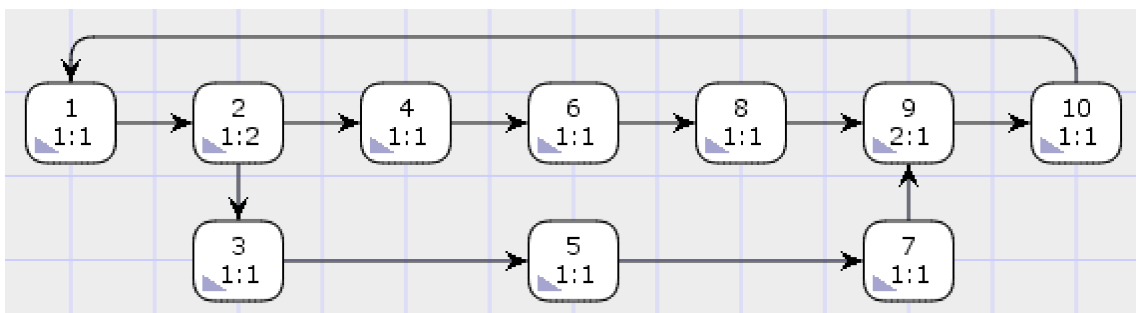


Figura 44 – Grafo das marcações acessíveis para a rede de Petri da Figura 39 depois da regra de sincronização, gerado pelo simulador CPN Tools.

Na Figura 44 é possível verificar que, após a aplicação da regra de sincronização, um novo grafo fortemente conexo é gerado e as marcações mortas, 7 e 10 do grafo apresentado na Figura 41, são removidas. A remoção das marcações mortas permite que todas as

transições possam ser sensibilizadas a partir de qualquer marcação do grafo das marcações acessíveis. Como os cenários Sc_3 e Sc_4 , responsáveis pelas situações de *deadlock* no exemplo, são removidos após a aplicação da regra de sincronização, não terá mais uma sequência de disparos das transições t_4 , t_1 e t_2 ou uma sequência de disparos das transições t_5 , t_1 e t_3 que conduza o grafo das marcações acessíveis a um estado de *deadlock*.

Liveness Properties	

Dead Markings	None
Dead Transition Instances	None
Live Transition Instances	All

Figura 45 – Relatório de Vivacidade para a rede de Petri da Figura 39 depois da regra de sincronização, gerado pelo simulador CPN Tools.

Conforme é possível verificar nas informações da propriedade de Vivacidade do relatório gerado, nenhuma marcação morta foi identificada. Dessa forma, é possível concluir que a nova rede de Petri apresentada pela Figura 39 é viva. Logo livre de *deadlock*.

5.4 Análise Quantitativa de Composição de *Web Services*

A presente seção apresenta a validação da abordagem proposta no ponto de vista quantitativo que considera a aplicação da regra de sincronização a uma composição de *Web Services*. A pesquisa considera a análise quantitativa ao verificar o tempo de execução de um cenário alterado em composições de *Web Services* após a aplicação da regra de sincronização. Com essa análise será possível identificar o impacto que a abordagem proposta terá no tempo de execução do modelo.

Inicialmente é necessário identificar os cenários que são alterados pela aplicação da regra de sincronização. Esses cenários são apresentados na seção 5.2. São cenários que possuem as transições do tipo t_{d1} e t_{d2} , e que representam, respectivamente, a última transição disparada antes da situação do *deadlock* e a transição morta do modelo analisado. Esses cenários precisam ser representados através de árvores de prova canônica da Lógica Linear antes e após a aplicação da regra para verificar o método proposto.

Para verificar o atraso que pode ser ocasionado pelo método proposto, o sistema composto apresentado na Figura 34, que equivale a uma $U(IOWF-net)$, é considerado.

Como mostrado na seção 5.2, os cenários Sc_1 e Sc_2 desse modelo possuem tanto a transição do tipo t_{d1} quanto a transição do tipo t_{d2} . Os sequentes dos cenários Sc_1 e Sc_2 devem ser analisados através da construção de uma árvore de prova da Lógica Linear com cálculo de datas. A Figura 39 corresponde ao modelo final da Figura 34 após a aplicação da regra de sincronização. As árvores de prova da Lógica Linear com cálculo de datas para os cenários Sc_1 e Sc_2 , antes e depois da aplicação da regra de sincronização, são apresentadas no Apêndice A.5.

As Tabelas 15 e 16 apresentadas no Apêndice A.5 mostram respectivamente os intervalos de datas simbólicas de execução para as tarefas dos cenários Sc_1 e Sc_2 , considerando o intervalo de execução $I_{Exec} = [D_{Pmin}, D_{Cmax}]$ e as respectivas árvores de prova da Lógica Linear com cálculo de datas para o modelo antes e depois da aplicação da regra de sincronização.

Como a regra de sincronização teve que ser aplicada duas vezes no modelo original, devido às duas situações de *deadlock* que haviam no modelo original, serão analisados os dois cenários alterados no modelo final, iniciando pela análise do cenário Sc_1 .

A informação sobre quando o processo global finalizará para o cenário Sc_1 é dada pela data de produção do átomo o . Esta data de produção é extraída da árvore de prova com cálculo de datas para este cenário. A Tabela 8 mostra as datas simbólicas de produção do átomo o no cenário Sc_1 antes e depois da aplicação da regra de sincronização.

Devido a aplicação da regra de sincronização, a data simbólica de produção do cenário Sc_1 é alterada. Como é possível verificar na Tabela 15 apresentada no Apêndice A.5, o intervalo de data simbólica de execução para a tarefa t_3 , correspondente ao disparo da transição do tipo t_{d1} , é alterado devido à sincronização dos processos locais. Antes da regra de sincronização, o disparo da transição t_3 dependia apenas do consumo do átomo do lugar $q1$ da LWF-net $C1$, porém após a regra de sincronização, o disparo de t_3 passa a depender também do consumo do átomo de $p1$ da LWF-net $T1$.

Tabela 8 – Data de produção do átomo o para o cenário Sc_1 .

Cenário Sc_1	Data de produção do átomo o
Antes	$D_i + d_i + d_4 + d_1 + d_3 + d_6 + d_o$
Depois	$\max\{D_i + d_i + d_4, D_i + d_i + d_4 + d_1\} + d_3 + d_6 + d_o$

Apenas a partir do disparo da transição t_4 , que possui o seu tempo máximo de disparo dependente do tempo de consumo do átomo do lugar $p1$, é que as datas simbólicas de execução das tarefas do modelo são alteradas.

Conforme é possível verificar na Tabela 8, após aplicar a regra de sincronização, a data de produção do átomo o no cenário Sc_1 é alterado de $D_i + d_i + d_4 + d_1 + d_3 + d_6 + d_o$ para $\max\{D_i + d_i + d_4, D_i + d_i + d_4 + d_1\} + d_3 + d_6 + d_o$.

Considerando que $D_i + d_i + d_4 + d_1$ será sempre maior ou igual à $D_i + d_i + d_4$, então é possível reescrever a data de produção do átomo o no cenário Sc_1 após a aplicação da regra de sincronização como $D_i + d_i + d_4 + d_1 + d_3 + d_6 + d_o$. Dessa forma, a data de

produção do átomo o é igual para o modelo tanto antes quanto depois da aplicação da regra de sincronização, ou seja, o tempo em que o processo global finalizará para o cenário Sc_1 não é alterado pela regra de sincronização.

Baseando-se nas datas de produção do átomo o para o cenário Sc_1 apresentadas na Tabela 8, pode-se calcular a data ao mais cedo de finalização do processo, dada pela data mínima D_{Pmin} de produção do átomo o e a data ao mais tarde de finalização do processo, dada pela data máxima D_{Pmax} de produção do átomo o . A Tabela 9 mostra os intervalos $[D_{Pmin}, D_{Cmax}]$ de produção do átomo o antes e depois da aplicação da regra de sincronização para o cenário Sc_1 .

Tabela 9 – Intervalo de datas simbólicas do cenário Sc_1 .

Cenário Sc_1	Intervalo
Antes	$[D_i + d_{imin} + d_{4min} + d_{1min} + d_{3min} + d_{6min} + d_{omin},$ $D_i + d_{imax} + d_{4max} + d_{1max} + d_{3max} + d_{6max} + d_{omax}]$
Depois	$[max\{D_i + d_{imin} + d_{4min}, D_i + d_{imin} + d_{4min} + d_{1min}\} + d_{3min} + d_{6min} + d_{omin},$ $max\{D_i + d_{imax} + d_{4max}, D_i + d_{imax} + d_{4max} + d_{1max}\} + d_{3max} + d_{6max} + d_{omax}]$

Como a regra de sincronização teve que ser aplicada duas vezes no modelo original, devido às duas situações de *deadlock* que haviam no modelo original, após a análise do cenário Sc_1 é feita a análise do cenário Sc_2 .

A informação sobre quando o processo global finalizará para o cenário Sc_2 é dada pela data de produção do átomo o . Esta data de produção é extraída da árvore de prova com cálculo de datas para este cenário. A Tabela 10 mostra as datas simbólicas de produção do átomo o no cenário Sc_2 antes e depois da aplicação da regra de sincronização.

Tabela 10 – Data de produção do átomo o para o cenário Sc_2 .

Cenário Sc_2	Data de produção do átomo o
Antes	$D_i + d_i + d_5 + d_1 + d_2 + d_7 + d_o$
Depois	$max\{D_i + d_i + d_5, D_i + d_i + d_5 + d_1\} + d_2 + d_7 + d_o$

Devido a aplicação da regra de sincronização, a data simbólica de produção do cenário Sc_2 é alterada. Como é possível verificar na Tabela 16 apresentada no Apêndice A.5, o intervalo de data simbólica de execução para a tarefa t_2 , correspondente ao disparo da transição do tipo t_{d1} , é alterado. Antes da regra de sincronização, o disparo da transição t_2 dependia apenas do consumo do átomo do lugar $q1$ da LWF-net $C1$, porém após a regra de sincronização o disparo de t_2 passa a depender também do consumo do átomo de $p2$ da LWF-net $T1$.

Apenas a partir do disparo da transição t_5 , que possui o seu tempo máximo de disparo dependente do tempo de consumo do átomo do lugar $p2$, é que as datas simbólicas de execução das tarefas do modelo são alteradas.

Conforme é possível verificar na Tabela 10, após aplicar a regra de sincronização, a data de produção do átomo o no cenário Sc_2 é alterado de $D_i + d_i + d_5 + d_1 + d_2 + d_7 + d_o$ para $max\{D_i + d_i + d_5, D_i + d_i + d_5 + d_1\} + d_2 + d_7 + d_o$.

Considerando que $D_i + d_i + d_5 + d_1$ será sempre maior ou igual à $D_i + d_i + d_5$, então é possível reescrever a data de produção do átomo o no cenário Sc_2 após a aplicação da regra de sincronização como $D_i + d_i + d_5 + d_1 + d_2 + d_7 + d_o$. Dessa forma, a data de produção do átomo o é igual para o modelo tanto antes quanto depois da aplicação da regra de sincronização, ou seja, o tempo em que o processo global finalizará para o cenário Sc_2 não é alterado pela regra de sincronização.

Baseando-se nas datas de produção do átomo o para o cenário Sc_2 apresentadas na Tabela 10, pode-se calcular a data ao mais cedo de finalização do processo, dada pela data mínima D_{Pmin} de produção do átomo o , e a data ao mais tarde de finalização do processo, dada pela data máxima D_{Pmax} de produção do átomo o . A Tabela 11 mostra os intervalos $[D_{Pmin}, D_{Cmax}]$ de produção do átomo o antes e depois da aplicação da regra de sincronização para o cenário Sc_2 .

Tabela 11 – Intervalo de datas simbólicas do cenário Sc_2 .

Cenário Sc_2	Intervalo
Antes	$[D_i + d_{imin} + d_{5min} + d_{1min} + d_{2min} + d_{7min} + d_{omin}, D_i + d_{imax} + d_{5max} + d_{1max} + d_{2max} + d_{7max} + d_{omax}]$
Depois	$[max\{D_i + d_{imin} + d_{5min}, D_i + d_{imin} + d_{5min} + d_{1min}\} + d_{2min} + d_{7min} + d_{omin}, max\{D_i + d_{imax} + d_{5max}, D_i + d_{imax} + d_{5max} + d_{1max}\} + d_{2max} + d_{7max} + d_{omax}]$

Para exemplificação, considera-se o modelo temporizado apresentado na Figura 46. O modelo da Figura 46 corresponde ao sistema composto apresentado na Figura 34 com intervalos de tempo associados às transições. Dessa forma, deve-se considerar os intervalos de tempo numérico apresentados na Figura 46 e o fato que o processo a ser tratado inicia-se na data 0 (zero), ou seja, $D_i = 0$. Uma vez que as datas simbólicas do modelo já foram calculadas, os intervalos de datas numéricas para a finalização do processo podem ser calculados apenas substituindo as datas simbólicas presentes na Tabela 9 e na Tabela 11 pelas datas numéricas associadas às transições apresentadas na Figura 46.

A Tabela 12 mostra o resultado da substituição das datas simbólicas por datas numéricas para o cenário Sc_1 .

Tabela 12 – Intervalo de datas numéricas do cenário Sc_1 .

Cenário Sc_1	Intervalo numérico
Antes	[2,30]
Depois	[2,30]

A Tabela 13 mostra o resultado da substituição das datas simbólicas por datas numéricas para o cenário Sc_2 .

Como a data de produção do átomo o é mantida para os cenários Sc_1 e Sc_2 após a aplicação da regra de sincronização, logo o seus intervalos de datas simbólicas $[D_{Pmin}, D_{Cmax}]$ também serão mantidos. Consequentemente, os intervalos de datas numéricas para os cenários Sc_1 e Sc_2 serão mantidos após a aplicação da regra de sincronização, como pode ser verificado respectivamente nas Tabelas 12 e 13.

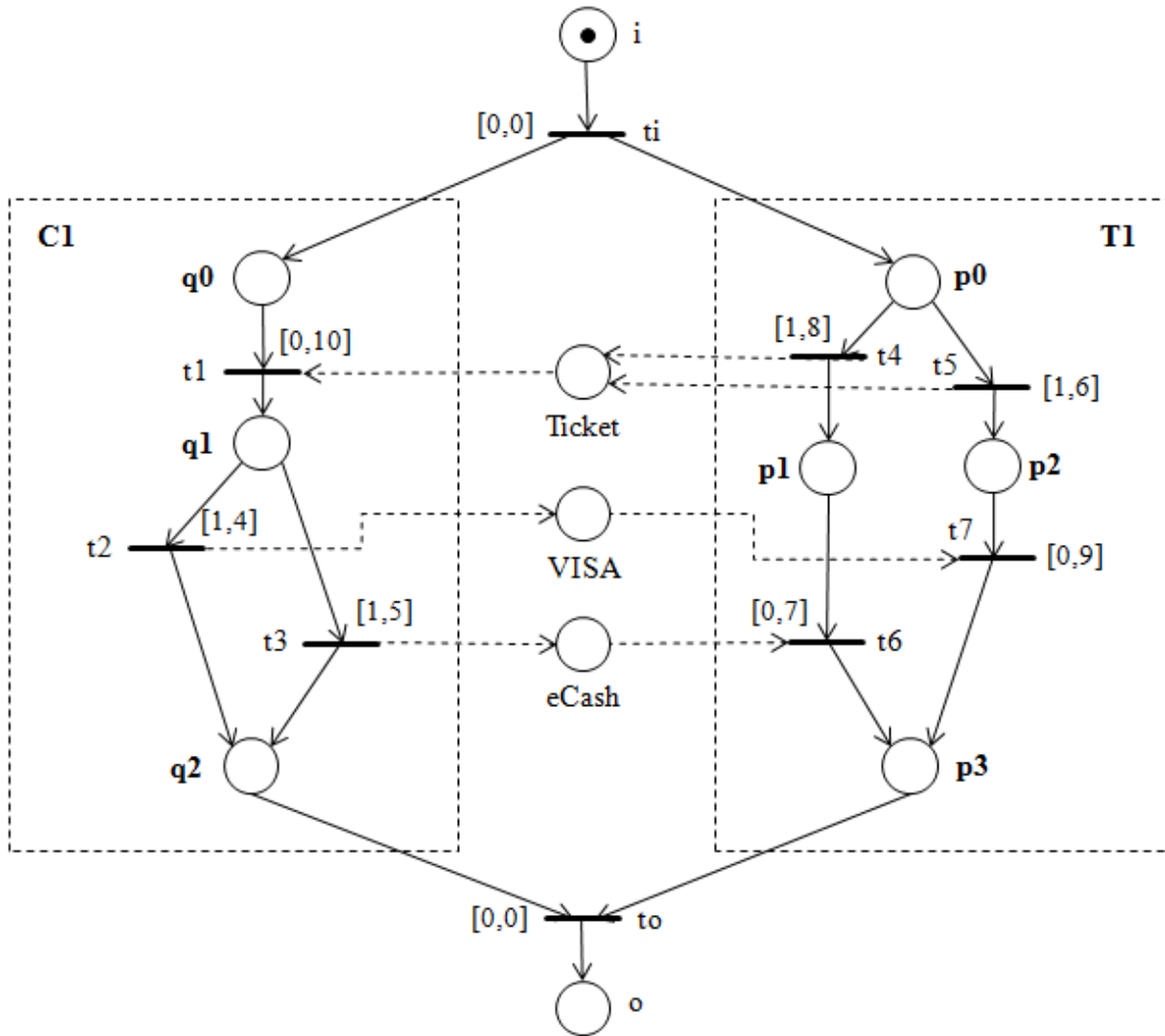


Figura 46 – Rede de Petri t-temporizada correspondente ao sistema composto da Figura 34.

Tabela 13 – Intervalo de datas numéricas do cenário Sc_2 .

Cenário Sc_2	Intervalo numérico
Antes	[2,29]
Depois	[2,29]

Para validar a abordagem proposta no ponto de vista quantitativo, foram realizadas simulações no simulador CPN Tools. Inicialmente é necessário implementar o modelo temporizado no simulador CPN Tools antes da aplicação da regra de sincronização. A Figura 47 apresenta a implementação no *software* CPN Tools da rede de Petri t-temporizada apresentada na Figura 46.

A análise quantitativa desse trabalho irá considerar somente os cenários alterados após a aplicação da regra de sincronização, dessa forma, é necessário implementar de maneira isolada os cenários que possuem as transições do tipo t_{d1} e t_{d2} . Assim, os outros cenários

do sistema composto são removidos e os cenários Sc_1 e Sc_2 , alterados após a aplicação da regra, são implementados de forma isolada no CPN Tools.

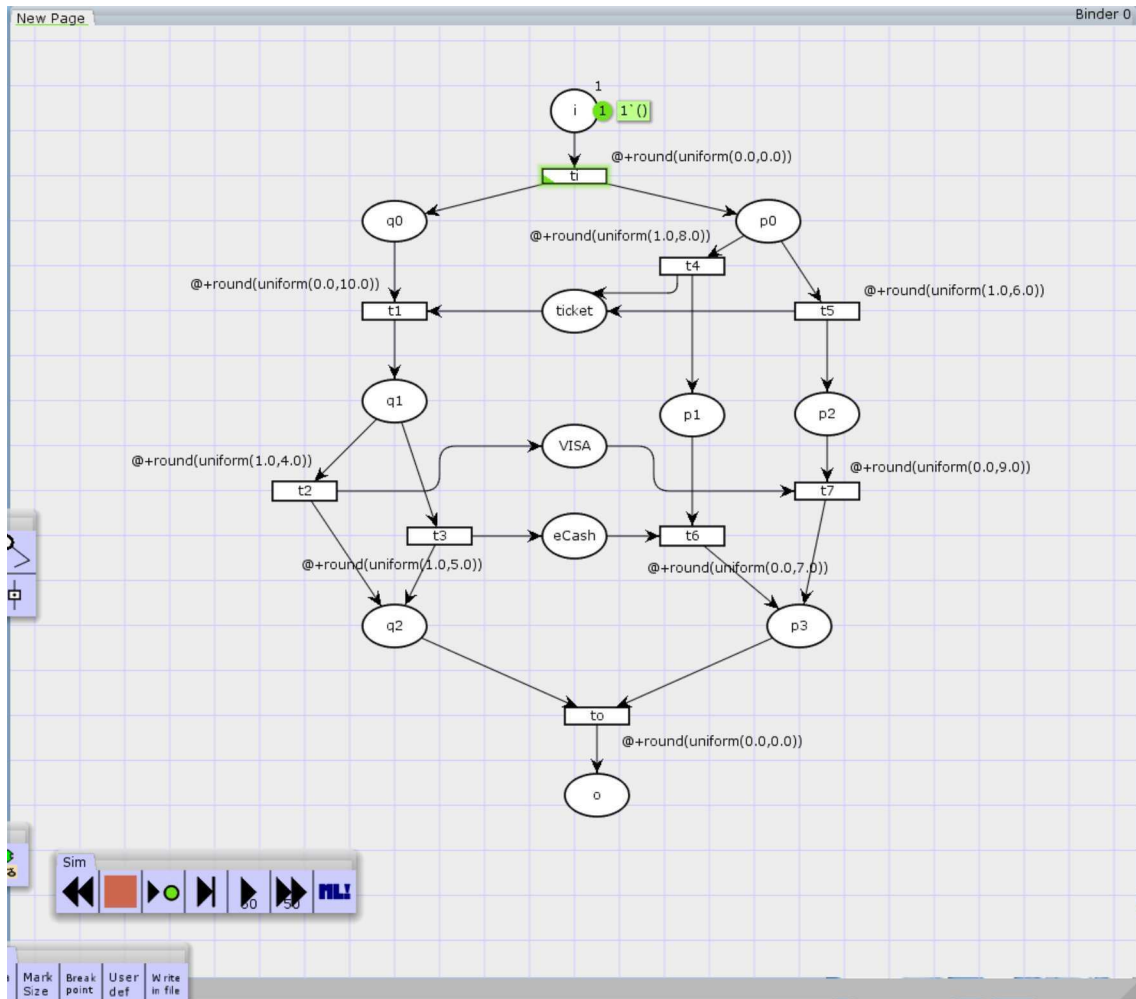


Figura 47 – Implementação no CPN Tools do modelo apresentado na Figura 46.

A Figura 48 apresenta a implementação no *software* CPN Tools do cenário Sc_1 antes da aplicação da regra de sincronização no modelo da Figura 46.

Segundo Rozinat et al. (2008), quanto maior o número de replicações, maior a precisão do resultado. No entanto, é necessário decidir um limite aceitável de replicações. Sendo assim, para as simulações realizadas nesta pesquisa foram consideradas 50 replicações.

Após o cenário Sc_1 ser implementado é possível executar as replicações das simulações para esse cenário. O relatório estatístico com o resultado das replicações das simulações para o cenário Sc_1 , gerado pelo CPN Tools, é apresentado na Figura 49.

Conforme é possível observar na área destacada com uma borda mais espessa na Figura 49, o término mais cedo ocorre na data 6 e o término mais tarde na data 27. Os resultados analíticos produziram, como mostra a Tabela 12, o intervalo de datas [2,30], ou seja, o término mais cedo na data 2 e o término mais tarde na data 30. Desta forma, os valores obtidos na simulação pertencem ao intervalo numérico definido pelo método analítico.

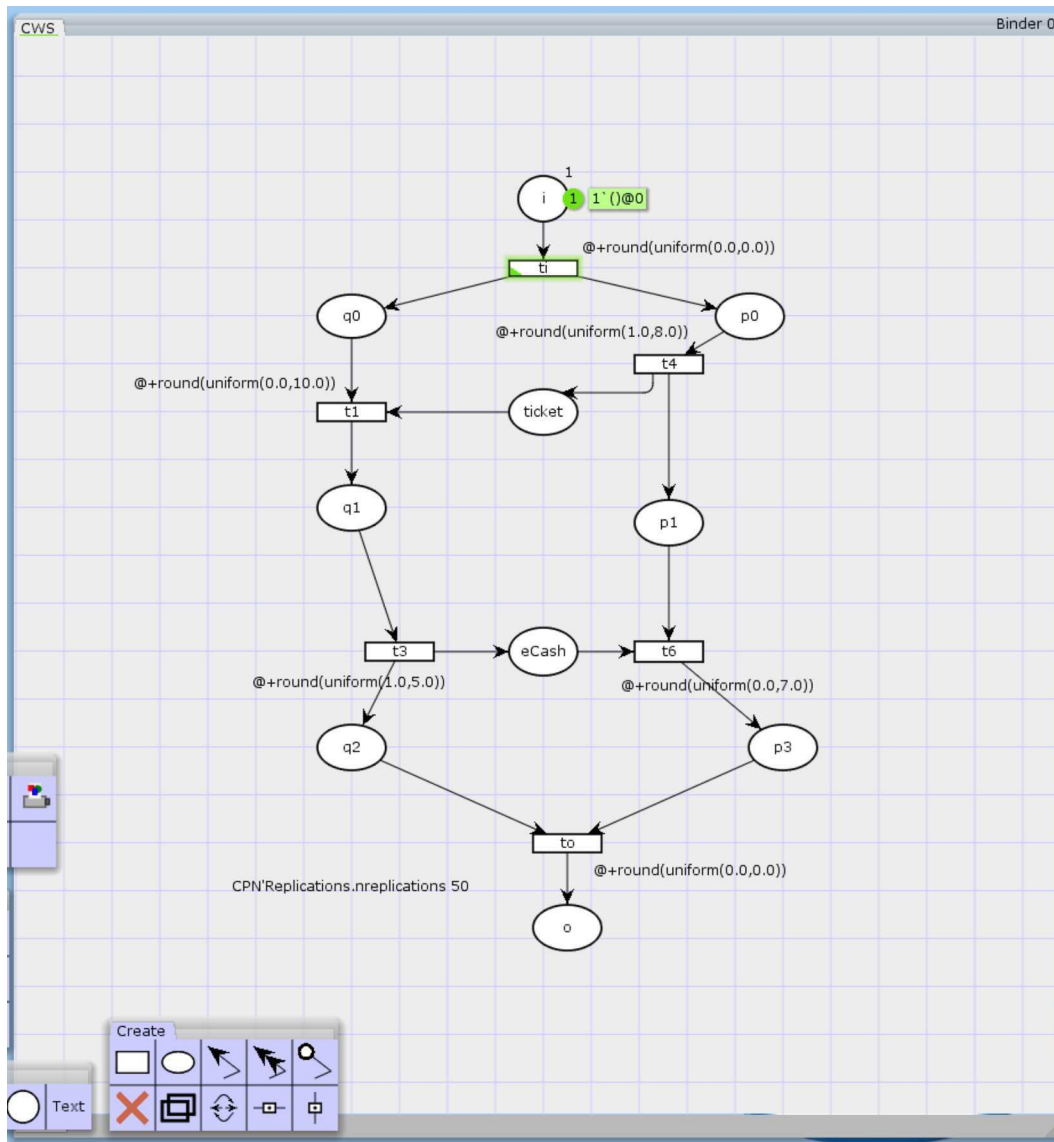


Figura 48 – Implementação no CPN Tools do cenário Sc_1 antes da regra de sincronização.

CPN Tools Performance Report
 Net: /cygdrive/C/Users/Vinicius/Desktop/CPN/WS/sc1.cpn
 Number of replications: 50

Statistics							
Name	Avrg	90% Half Length	95% Half Length	99% Half Length	Std	Min	Max
ParadaO							
count_iid	6.000000	0.000000	0.000000	0.000000	0.000000	6	6
max_iid	15.740000	1.053923	1.264832	1.692284	4.425391	6	27
min_iid	0.000000	0.000000	0.000000	0.000000	0.000000	0	0
sum_iid	40.800000	3.066507	3.680173	4.923893	12.876176	14	74
avrg_iid	6.800000	0.511085	0.613362	0.820649	2.146029	2.333333	12.333333

Figura 49 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_1 antes da regra de sincronização.

Considerando o intervalo de confiança de 95%, as replicações podem variar 1.264832 entre o limite inferior e o limite superior do intervalo, como mostra a coluna 95% *Half Length* da Figura 48. Se esse valor (arredondado) for subtraído ao limite inferior e somado ao limite superior, tem-se que o término mais cedo ocorre na data 4,74 e o término mais tarde na data 28,26. Desta forma, considerando o intervalo de confiança de 95%, os valores obtidos na simulação ainda pertencem ao intervalo de datas [2,30], apresentado na Tabela 12, definido pelo método analítico.

A Figura 50 apresenta a implementação no *software* CPN Tools do cenário Sc_2 antes da aplicação da regra de sincronização no modelo da Figura 46.

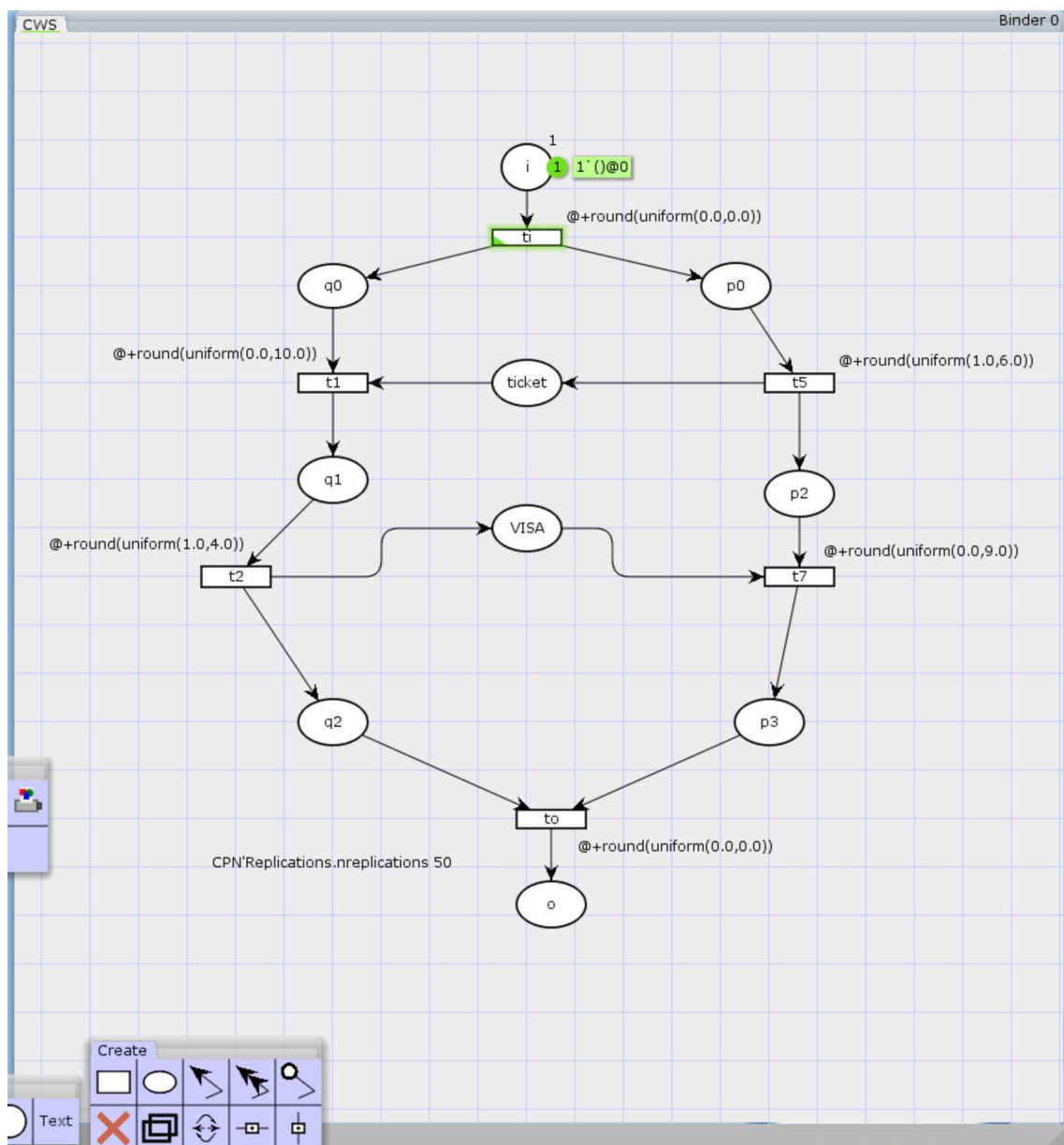


Figura 50 – Implementação no CPN Tools do cenário Sc_2 antes da regra de sincronização.

Após o cenário Sc_2 ser implementado é possível executar as replicações das simulações para esse cenário. O relatório estatístico com o resultado das replicações das simulações

para o cenário Sc_2 , gerado pelo CPN Tools, é apresentado na Figura 51.

CPN Tools Performance Report
 Net: /cygdrive/C/Users/Vinicius/Desktop/CPN/WS/sc2.cpn
 Number of replications: 50

Statistics							
Name	Avrg	90% Half Length	95% Half Length	99% Half Length	StD	Min	Max
ParadaO							
count_iid	6.000000	0.000000	0.000000	0.000000	0.000000	6	6
max_iid	15.860000	0.885891	1.063174	1.422476	3.719831	10	25
min_iid	0.000000	0.000000	0.000000	0.000000	0.000000	0	0
sum_iid	38.740000	2.220989	2.665451	3.566244	9.325869	24	61
avrg_iid	6.456667	0.370165	0.444242	0.594374	1.554312	4.000000	10.166667

Figura 51 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_2 antes da regra de sincronização.

Conforme é possível observar na área destacada com uma borda mais espessa na Figura 51, o término mais cedo ocorre na data 10 e o término mais tarde na data 25. Os resultados analíticos produziram, como mostra a Tabela 13, o intervalo de datas [2,29], ou seja, o término mais cedo na data 2 e o término mais tarde na data 29. Desta forma, os valores obtidos na simulação pertencem ao intervalo numérico definido pelo método analítico.

Considerando o intervalo de confiança de 95%, as replicações podem variar 1.063174 entre o limite inferior e o limite superior do intervalo, como mostra a coluna *95% Half Length* da Figura 51. Se esse valor (arredondado) for subtraído ao limite inferior e somado ao limite superior, tem-se que o término mais cedo ocorre na data 8,9 e o término mais tarde na data 26,1. Desta forma, considerando o intervalo de confiança de 95%, os valores obtidos na simulação ainda pertencem ao intervalo de datas [2,29], apresentado na Tabela 13, definido pelo método analítico.

Após a aplicação da regra de sincronização, o sistema composto da Figura 34 é alterado e dá origem a um novo modelo representado pela Figura 39. Esse novo modelo, gerado após a aplicação da regra de sincronização, precisa também ser implementado no CPN Tools para efeito de comparação. A Figura 52 apresenta a implementação do modelo da Figura 39 no *software* CPN Tools com os mesmos tempos associados nas transições do modelo da Figura 46.

Para analisar apenas os cenários alterados após a aplicação da regra, inicialmente o cenário Sc_2 do modelo implementado na Figura 52 é removido e o cenário Sc_1 é implementado de forma isolada no CPN Tools. A Figura 53 apresenta a implementação do cenário Sc_1 do modelo da Figura 52 no *software* CPN Tools.

Ao executar as replicações das simulações do cenário Sc_1 , após a regra de sincronização, o relatório estatístico com o resultado das replicações, apresentado na Figura 54, é gerado.

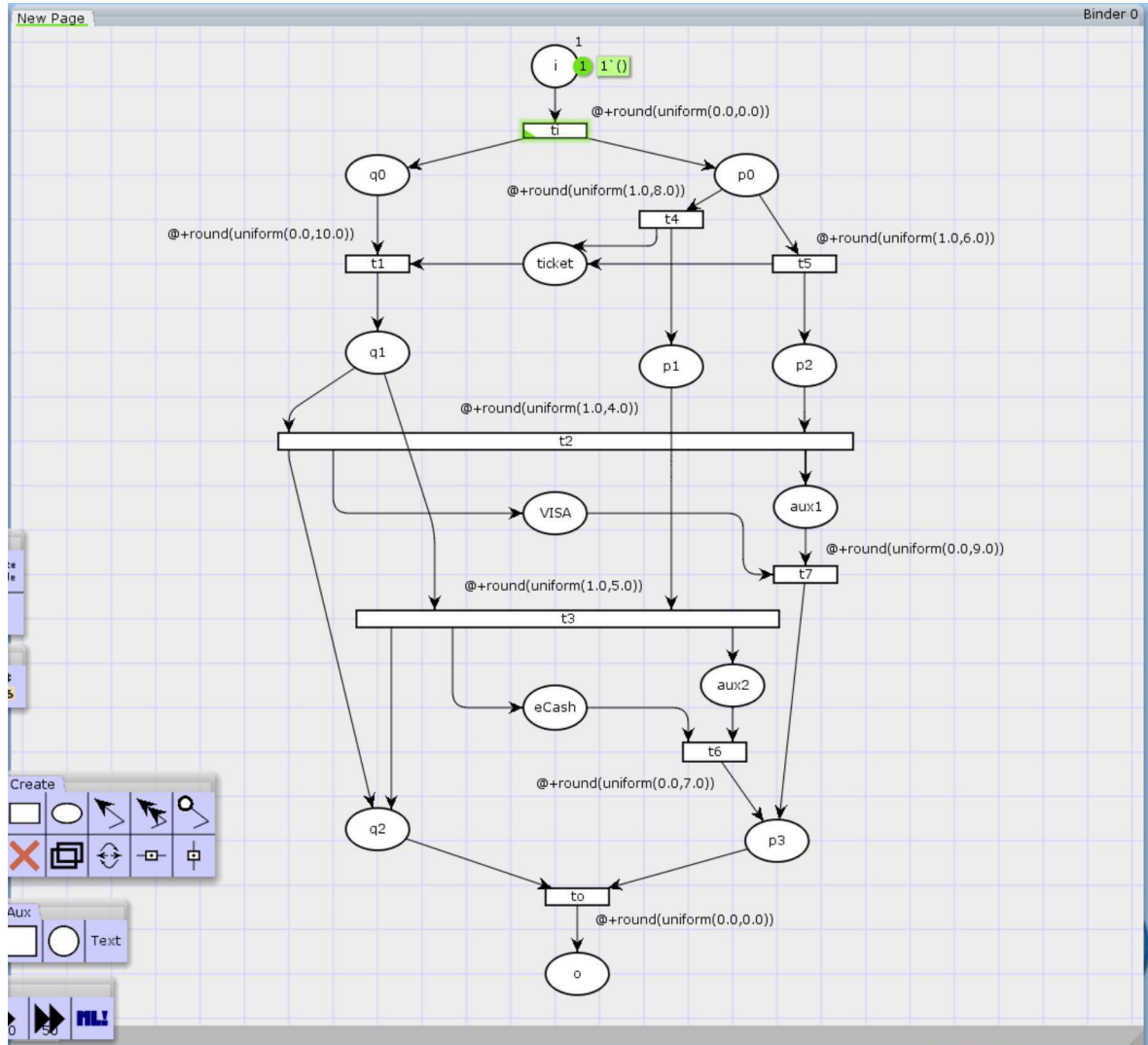


Figura 52 – Implementação no CPN Tools do modelo t-temporizado correspondente a rede de Petri apresentada na Figura 39.

Conforme é possível observar na área destacada com uma borda mais espessa na Figura 54, o término mais cedo ocorre na data 10 e o término mais tarde na data 25. Os resultados analíticos produziram, como mostra a Tabela 12, o intervalo de datas [2,29], ou seja, o término mais cedo na data 2 e o término mais tarde na data 29. Desta forma, os valores obtidos na simulação pertencem ao intervalo numérico definido pelo método analítico.

Considerando o intervalo de confiança de 95%, as replicações podem variar 1.107760 entre o limite inferior e o limite superior do intervalo, como mostra a coluna 95% *Half Length* da Figura 54. Se esse valor (arredondado) for subtraído ao limite inferior e somado ao limite superior, tem-se que o término mais cedo ocorre na data 8,9 e o término mais tarde na data 26,10. Desta forma, considerando o intervalo de confiança de 95%, os valores obtidos na simulação ainda pertencem ao intervalo de datas [2,29], apresentado na Tabela 12, definido pelo método analítico.

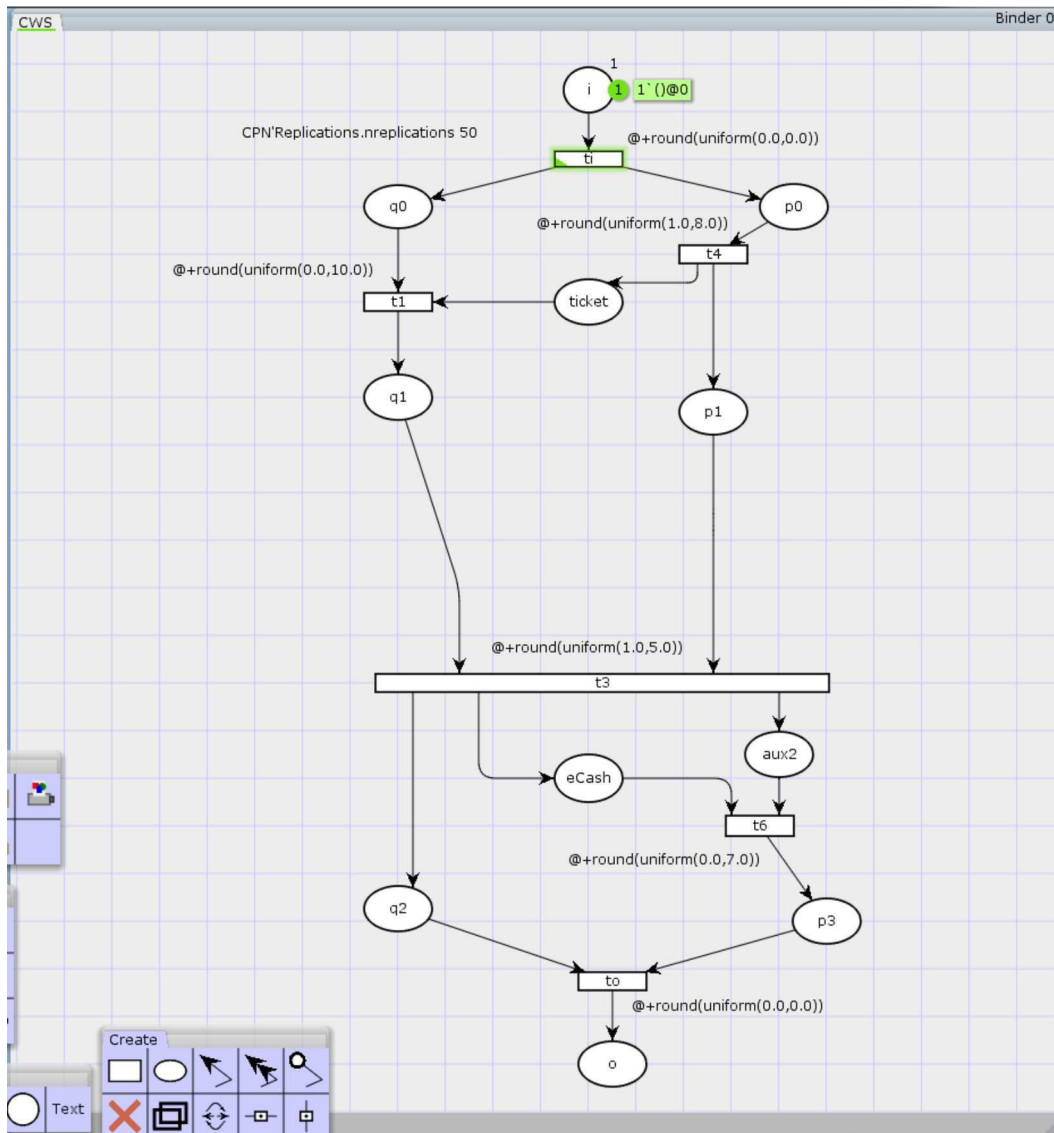


Figura 53 – Implementação no CPN Tools do cenário Sc_1 depois da regra de sincronização.

CPN Tools Performance Report
 Net: /cygdrive/C/Users/Vinicius/Desktop/CPN/WS/sc1r.cpn
 Number of replications: 50

Statistics							
Name	Avrg	90% Half Length	95% Half Length	99% Half Length	StD	Min	Max
ParadaO							
count_iid	6.000000	0.000000	0.000000	0.000000	0.000000	6	6
max_iid	16.280000	0.923042	1.107760	1.482130	3.875828	10	25
min_iid	0.000000	0.000000	0.000000	0.000000	0.000000	0	0
avrg_iid	5.692444	0.359920	0.431946	0.577923	1.511292	3.100000	9.080000

Figura 54 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_1 depois da regra de sincronização.

Para analisar apenas os cenários alterados após a aplicação da regra, o cenário Sc_1 do modelo implementado na Figura 52 que já foi analisado é removido e o cenário Sc_2 é implementado de forma isolada no CPN Tools. A Figura 55 apresenta a implementação do cenário Sc_2 do modelo da Figura 52 no *software* CPN Tools.

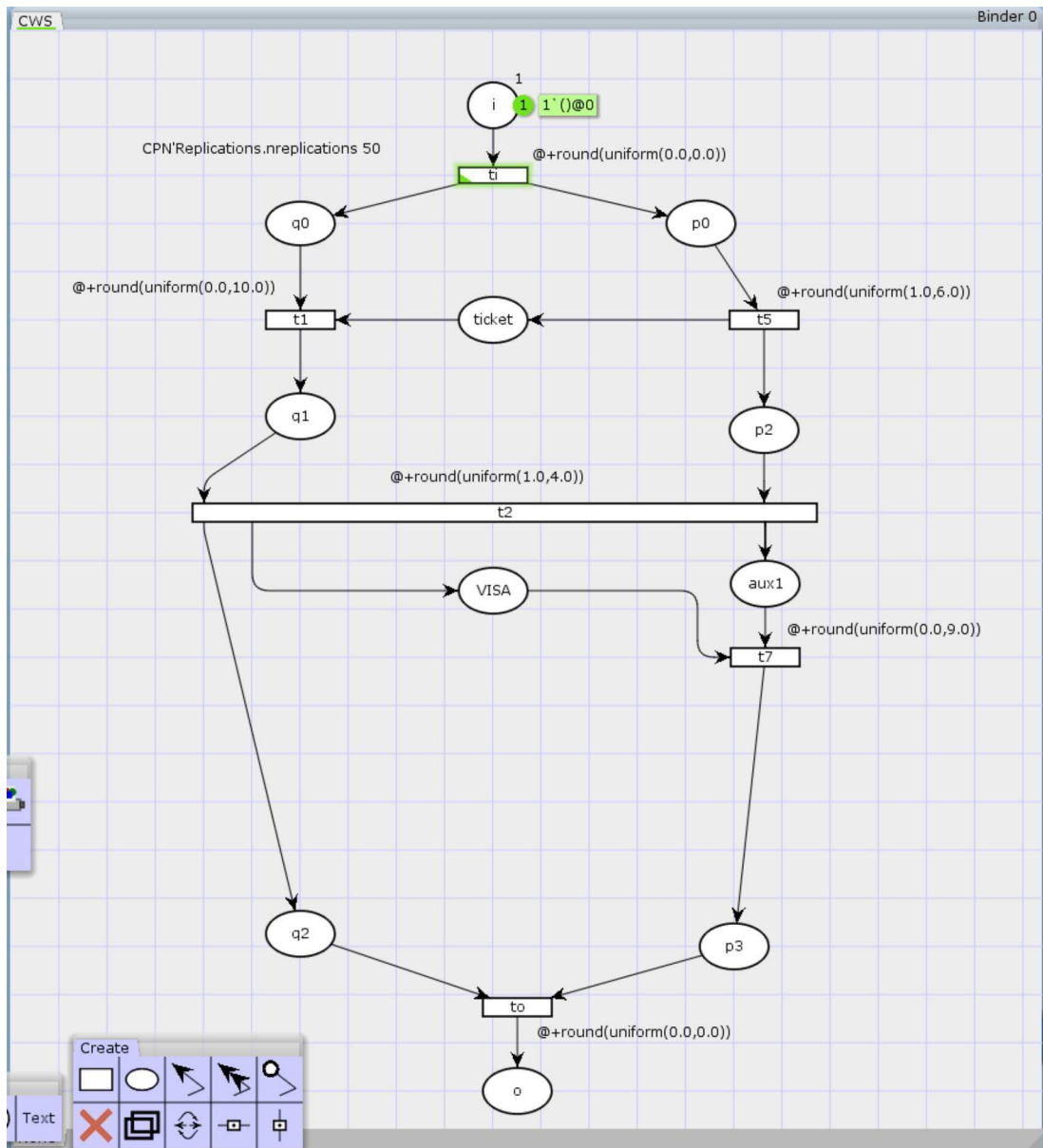


Figura 55 – Implementação no CPN Tools do cenário Sc_2 depois da regra de sincronização.

Ao executar as replicações das simulações do cenário Sc_2 , após a regra de sincronização, o relatório estatístico com o resultado das replicações, apresentado na Figura 56, é gerado.

Conforme é possível observar na área destacada com uma borda mais espessa na Figura 56, o término mais cedo ocorre na data 6 e o término mais tarde na data 26. Os resultados analíticos produziram, como mostra a Tabela 13, o intervalo de datas $[2,29]$, ou seja, o término mais cedo na data 2 e o término mais tarde na data 29. Desta forma, os valores

obtidos na simulação pertencem ao intervalo numérico definido pelo método analítico.

CPN Tools Performance Report

Net: /cygdrive/C/Users/Vinicius/Desktop/CPN/WS/sc2r.cpn

Number of replications: 50

Statistics							
Name	Avrg	90% Half Length	95% Half Length	99% Half Length	StD	Min	Max
ParadaO							
count_iid	6.000000	0.000000	0.000000	0.000000	0.000000	6	6
max_iid	15.240000	1.063489	1.276313	1.707645	4.465560	6	26
min_iid	0.000000	0.000000	0.000000	0.000000	0.000000	0	0
avrg_iid	5.151039	0.382145	0.458620	0.613611	1.604617	1.833333	9.076923

Figura 56 – Relatório de replicações de simulação que considera a monitoração do lugar o no cenário Sc_2 depois da regra de sincronização.

Considerando o intervalo de confiança de 95%, as replicações podem variar 1.276313 entre o limite inferior e o limite superior do intervalo, como mostra a coluna *95% Half Length* da Figura 56. Se esse valor (arredondado) for subtraído ao limite inferior e somado ao limite superior, tem-se que o término mais cedo ocorre na data 4,72 e o término mais tarde na data 27,28. Desta forma, considerando o intervalo de confiança de 95%, os valores obtidos na simulação ainda pertencem ao intervalo de datas [2,29], apresentado na Tabela 13, definido pelo método analítico.

Como apresentado nesta seção, os cenários Sc_1 e Sc_2 mesmo após a aplicação da regra de sincronização, podem ter as datas simbólicas de produção do átomo o mantido, ou seja, igual aos dos próprios cenários antes da alteração sofrida pela regra. Dessa forma, tanto ao calcular os intervalos de datas numéricas quanto ao simular o cenário no *software* CPN Tools, é possível verificar que devido a estrutura da rede considerada neste exemplo específico, a aplicação da regra de sincronização para o caso específico do modelo da Figura 34, não altera o tempo de finalização do processo global para os cenários Sc_1 e Sc_2 .

Conclusão

Neste capítulo é apresentada a conclusão desta pesquisa. Inicialmente, na seção 6.1 são apresentadas as principais contribuições. Já a seção 6.2 apresenta os trabalhos futuros que poderão ser desenvolvidos considerando os resultados obtidos nesta pesquisa. Finalmente, as contribuições em produção bibliográfica, obtidas no contexto da presente pesquisa, são apresentadas na seção 6.3.

6.1 Principais Contribuições

Este trabalho tem como principal contribuição a apresentação de um método para remoção de situações de *deadlock* em processos de *workflow* interorganizacionais modelados por IOWF-*nets* que são localmente *sound*, porém não globalmente *sound*. É apresentado também um tipo de análise qualitativa e quantitativa de IOWF-*nets* baseada no cálculo dos sequentes da Lógica Linear.

Na seção 4.1 é apresentado um método que utiliza a construção e análise das árvores de prova da Lógica Linear para verificar quais os cenários possuem situações de *deadlock* no modelo. Além disso, em particular, é possível identificar quais os elementos de comunicação assíncrona responsáveis pelas situações de *deadlock* entre processos de *workflow* distintos. Foi possível então validar a primeira hipótese desta pesquisa, ou seja, *a utilização da Lógica Linear, mais especificamente a construção e análise de árvores de prova da Lógica Linear, permite identificar situações de deadlock em WorkFlow nets Interorganizacionais que são localmente sound; em particular é possível identificar também os elementos de comunicação assíncrona responsáveis pelas situações de travamento nas comunicações entre processos de workflow distintos.*

O método apresentado na seção 4.1 substitui os elementos de comunicação assíncrona responsáveis por situações de *deadlock* por um novo mecanismo de comunicação parcialmente síncrono. Tal regra de sincronização tem como objetivo sincronizar a execução das atividades que levam o processo interorganizacional a situações de *deadlock*. Na seção 4.2 é feita uma análise qualitativa, onde é possível validar que após a aplicação do mé-

todo proposto em 4.1, as situações de *deadlock* são removidas do modelo. Assim, tem-se que a segunda hipótese desta pesquisa foi validada, isto é, *a substituição dos elementos de comunicação assíncrona responsáveis por um estado de deadlock em uma WorkFlow net Interorganizacional por uma regra de sincronização entre partes específicas de processos distintos constitui uma condição necessária e suficiente para que todas atividades dos processos que estão diretamente envolvidas com a situação de deadlock possam ser corretamente executadas e os processos finalizados.*

O método apresentado na seção 4.1 considera somente as partes alteradas de uma IOWF-net quando mecanismos de sincronização são introduzidos entre certas atividades que comunicam em um processo interorganizacional. Tal método em particular reprocessa somente os sequentes específicos de partes de árvores de provas já parcialmente calculadas e não é necessária uma análise completa do modelo do sistema a cada vez que ocorre uma modificação na estrutura de controle. Dessa forma, é possível concluir que a terceira hipótese desta pesquisa foi validada, ou seja, *em caso de sincronização de certas partes dos processos de uma WorkFlow net Interorganizacional, somente as partes diretamente alteradas do modelo deverão ser reanalisadas, considerando somente fragmentos das árvore de provas do cálculo dos sequentes da Lógica Linear que foram geradas durante a análise qualitativa responsável pela detecção de situações de deadlock.*

A análise quantitativa, apresentada na seção 4.3, permite deduzir que o modelo final de processo interorganizacional, após aplicação da regra de sincronização, poderá em certos casos sofrer um acréscimo no tempo global na finalização do processo de acordo com a estrutura da rede de Petri ou as datas numéricas atribuídas ao disparo das transições, atrasando a finalização do processo global. Portanto, conclui-se que a quarta hipótese desta pesquisa também foi validada, ou seja, *a sincronização de certas partes dos processos de uma WorkFlow net Interorganizacional pode gerar atrasos na finalização das atividades dos processos e, conseqüentemente, provocar uma degradação do desempenho global do sistema.*

Finalmente, a quinta hipótese desta pesquisa é validada através do estudo de caso apresentado no Capítulo 5. No Capítulo 5, o método apresentado no Capítulo 4 é aplicado no contexto de composições de *Web Services*, já que existe uma relação próxima entre a modelagem de um processo de *workflow* interorganizacional e uma composição de *Web Services*. Assim, tem-se a validação da última hipótese desta pesquisa, isto é, *os modelos e métodos de análise baseados em uma WorkFlow net Interorganizacional e no cálculo dos sequentes da Lógica Linear, apresentados neste trabalho, podem ser diretamente estendidos a modelos de Web Services modelados por redes de Petri.*

6.2 Trabalhos Futuros

Como trabalho futuro, pretende-se automatizar o processo de construção das árvores de prova da Lógica Linear para o sequente linear de um cenário da IOWF-*net* analisada, facilitando, em particular, a detecção automática dos cenários não *sound* e dos elementos de comunicação assíncrona responsáveis pelas situações de *deadlock* do modelo.

Além disso, deve-se propor um método para identificação de todos os elementos necessários para aplicação da regra de sincronização apenas através do uso da Lógica Linear, considerando que a identificação da transição do tipo t_{d2} do método atual depende da análise do modelo global em redes de Petri.

Considerando estes dois trabalhos futuros, pretende-se também automatizar o processo da aplicação da regra de sincronização e geração da nova árvore de prova para o cenário alterado após aplicação do método.

Um outro trabalho interessante é a verificação da aplicação da regra de sincronização, proposta neste trabalho, nas IOWF-*nets* com mais de duas LWF-*nets*, onde estas são *sound*, mas o modelo global possui situações de *deadlock*.

De maneira semelhante, é interessante apresentar exemplos de cenários onde a regra de sincronização proposta neste trabalho seria aplicada em *WorkFlow net* Interorganizacionais com mais de um arco de entrada ou mais de um arco de saída nos lugares de comunicação responsáveis pelas situações de *deadlock*.

Finalmente, deve-se aprofundar os estudos no que se diz respeito ao relacionamento entre a aplicação da regra de sincronização proposta neste trabalho, que remove as situações de *deadlock* em *WorkFlow net* Interorganizacionais introduzidos por lugares de comunicação do tipo assíncrono, e a presença de sifões que esvaziam (*deadlocks* estruturais) em sistemas modelados por redes de Petri, mostrando, em particular, como a sincronização de certos elementos do modelo permite remover sifões que podem se esvaziar de suas fichas.

6.3 Contribuições em Produção Bibliográfica

Na sequência são apresentadas as produções bibliográficas resultante desta pesquisa.

O artigo Oliveira et al. (2018), intitulado “A Synchronization Rule Based on Linear Logic for Deadlock Prevention in Interorganizational WorkFlow Nets”, foi publicado na décima quarta edição da International Conference on Information Technology - New Generations (ITNG) em 2017. Tal artigo descreve a regra de sincronização para remoção de situações de *deadlock* em *WorkFlow nets* Interorganizacionais, conforme apresentado na seção 4.1.

Já o artigo Oliveira et al. (2017), intitulado “A Linear Logic based Synchronization Rule for Deadlock Prevention in Web Service Composition”, foi publicado e apresentado na décima nona edição da International Conference on Enterprise Information Systems

(ICEIS) em 2017. Tal artigo descreve e aplica a regra de sincronização para remoção de situações de *deadlock* ao contexto de composições de *Web Services* modelados através de *WorkFlow nets* Interorganizacionais, conforme apresentado nas seções 5.1 e 5.2.

Referências

AALST, M. P. V. D. Modeling and analyzing interorganizational workflows. In: **Proceedings 1998 International Conference on Application of Concurrency to System Design**. [S.l.: s.n.], 1998. p. 262–272.

AALST, W. M. P. van der. Verification of workflow nets. In: _____. **Application and Theory of Petri Nets 1997: 18th International Conference, ICATPN'97 Toulouse, France, June 23–27, 1997 Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997. p. 407–426. ISBN 978-3-540-69187-7.

_____. Three good reasons for using a petri-net-based workflow management system. In: _____. **Information and Process Integration in Enterprises: Rethinking Documents**. Boston, MA: Springer US, 1998. p. 161–182. ISBN 978-1-4615-5499-8. Disponível em: <https://doi.org/10.1007/978-1-4615-5499-8_10>.

AALST, W. M. P. van der et al. Soundness of workflow nets: classification, decidability, and analysis. **Formal Aspects of Computing**, v. 23, n. 3, p. 333–363, May 2011. ISSN 1433-299X. Disponível em: <<https://doi.org/10.1007/s00165-010-0161-4>>.

AALST, W. M. Van der. The application of petri nets to workflow management. **Journal of Circuits, Systems and Computers**, v. 08, n. 01, p. 21–66, 1998. Disponível em: <<https://doi.org/10.1142/S0218126698000043>>.

AALST, W. V. D.; HEE, K. M. V. **Workflow Management: Models, Methods, and Systems**. [S.l.]: MIT Press, 2004. (Information Systems). ISBN 9780262296908.

BANASZAK, Z. A.; KROGH, B. H. Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows. **IEEE Transactions on Robotics and Automation**, v. 6, n. 6, p. 724–734, Dec 1990. ISSN 1042-296X.

BARKAOUI, K.; ABDALLAH, I. B. Deadlock avoidance in fms based on structural theory of petri nets. In: **Emerging Technologies and Factory Automation, 1995. ETFA '95, Proceedings., 1995 INRIA/IEEE Symposium on**. [s.n.], 1995. v. 2, p. 499–510 vol.2. Disponível em: <<https://doi.org/10.1109/ETFA.1995.496690>>.

BARROS, A.; DUMAS, M.; OAKS, P. Standards for web service choreography and orchestration: Status and perspectives. In: _____. **Business Process Management Workshops: BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005. Revised Selected**

- Papers**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 61–74. ISBN 978-3-540-32596-3.
- CAPTARENCU, O. O. Modelling and verification of interorganizational workflows with security constraints: A petri nets-based approach. In: _____. **Advanced Information Systems Engineering Workshops: CAiSE 2012 International Workshops, Gdańsk, Poland, June 25-26, 2012. Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 486–493. ISBN 978-3-642-31069-0. Disponível em: <https://doi.org/10.1007/978-3-642-31069-0_39>.
- CARDOSO, J. Time fuzzy petri nets. **Fuzziness in Petri Nets, Studies in Fuzziness and Soft Computing**, v. 22, p. 115–145, 1998.
- CARDOSO, J.; VALETTE, R. **Redes de Petri**. [S.l.]: Editora da UFSC, 1997.
- CHAMPAGNAT, R.; PRADIN-CHÉZALVIEL, B.; VALETTE, R. Petri nets and linear logic as an aid for scheduling batch processes. **ADPM 2000, Automation of mixed processes: Hybrid Dynamic Systems**, p. 107–112, 2000.
- DAVID, R.; ALLA, H. **Discrete, Continuous, and Hybrid Petri Nets**. 2nd. ed. Springer Publishing Company, Incorporated, 2010. ISBN 3642106684, 9783642106682. Disponível em: <<https://doi.org/10.1007/978-3-642-10669-9>>.
- DIAZ, M. **Petri Nets: Fundamental Models, Verification and Applications**. Wiley-IEEE Press, 2009. ISBN 1848210795, 9781848210790. Disponível em: <<https://doi.org/10.1002/9780470611647>>.
- DUMAS, M.; HOFSTEDE, A. H. M. ter. Uml activity diagrams as a workflow specification language. In: _____. **UML 2001 — The Unified Modeling Language. Modeling Languages, Concepts, and Tools: 4th International Conference Toronto, Canada, October 1–5, 2001 Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. p. 76–90. Disponível em: <https://doi.org/10.1007/3-540-45441-1_7>.
- GALVIN, P. B.; GAGNE, G.; SILBERSCHATZ, A. **Operating system concepts**. [S.l.]: John Wiley & Sons, Inc., 2013.
- GIRARD, J.-Y. Linear logic. **Theoretical Computer Science**, v. 50, n. 1, p. 1 – 101, 1987. ISSN 0304-3975.
- _____. Linear logic: Its syntax and semantics. In: **Proceedings of the Workshop on Advances in Linear Logic**. New York, NY, USA: Cambridge University Press, 1995. p. 1–42. ISBN 0-521-55961-8. Disponível em: <<https://doi.org/10.1017/CBO9780511629150.002>>.
- GIRAULT, F.; PRADIER-CHÉZALVIEL, B.; VALETTE, R. A logic for petri nets. **Journal européen des systèmes automatisés**, Lavoisier, v. 31, n. 3, p. 525–542, 1997.
- GOCHET, P.; GRIBOMONT, P. **"Logique", méthodes pour l'informatique fondamentale**. [S.l.]: A. Colin, etc., 1992. v. 97. 573 p.
- HACK, M. **Analysis of production schemata by Petri nets**. Cambridge, MA, USA, 1972.

HEBALKAR, P. G. **Deadlock-free sharing of resources in asynchronous systems**. Cambridge, MA, USA, 1970.

HOLT, R. C. Some deadlock properties of computer systems. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 4, n. 3, p. 179–196, set. 1972. ISSN 0360-0300.

IORDACHE, M. V.; MOODY, J.; ANTSAKLIS, P. J. Synthesis of deadlock prevention supervisors using petri nets. **IEEE Transactions on Robotics and Automation**, v. 18, n. 1, p. 59–68, Feb 2002. ISSN 1042-296X.

JENSEN, K. Coloured petri nets: A high level language for system design and analysis. In: _____. **Advances in Petri Nets 1990**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991. p. 342–416. ISBN 978-3-540-46369-6. Disponível em: <https://doi.org/10.1007/3-540-53863-1_31>.

JULIA, S.; OLIVEIRA, F. F. de; VALETTE, R. Real time scheduling of workflow management systems based on a p-time petri net model with hybrid resources. **Simulation Modelling Practice and Theory**, v. 16, n. 4, p. 462 – 482, 2008. ISSN 1569-190X.

JULIA, S.; SOARES, M. Verification of real time uml specifications through a specialized inference mechanism based on a token player algorithm and the sequent calculus of linear logic. In: **Proceedings of the 15th European Simulation Symposium (ESS 2003)**. [S.l.: s.n.], 2003. p. 65–70.

JÚNIOR, S. B. Arquitetura de software baseada numa abordagem uml/redes de petri com prevençao de bloqueio mortal em sistemas de tempo real. Universidade Federal de Uberlândia, 2003.

KLAI, K.; OCHI, H.; TATA, S. Formal abstraction and compatibility checking of web services. In: **2013 IEEE 20th International Conference on Web Services**. [s.n.], 2013. p. 163–170. Disponível em: <<https://doi.org/10.1109/ICWS.2013.31>>.

MACIEL, P. R.; LINS, R. D.; CUNHA, P. R. **Introdução às redes de Petri e aplicações**. [S.l.]: UNICAMP - Instituto de Computação, 1996.

MARTENS, A. Analyzing web service based business processes. In: _____. **Fundamental Approaches to Software Engineering: 8th International Conference, FASE 2005, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2005, Edinburgh, UK, April 4-8, 2005. Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 19–33. ISBN 978-3-540-31984-9. Disponível em: <https://doi.org/10.1007/978-3-540-31984-9_3>.

MARUTA, T. et al. A deadlock detection algorithm for business processes workflow models. In: **Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on**. [S.l.: s.n.], 1998. v. 1, p. 611–616 vol.1. ISSN 1062-922X.

MERLIN, P. M. **A Study of the Recoverability of Computing Systems**. Tese (Doutorado) — University of California, 1974.

MOLLOY, M. K. **On the Integration of Delay and Throughput Measures in Distributed Processing Models**. Tese (Doutorado) — University of California, 1981.

MUGARZA, J. et al. Réductions fondées sur la logique linéaire pour l'ordonnancement des réseaux de petri. In: **MOSIM'01: conference francophone de modélisation et simulation**. [S.l.: s.n.], 2001.

MURATA, T. Petri nets: Properties, analysis and applications. **Proceedings of the IEEE**, v. 77, n. 4, p. 541–580, Apr 1989. ISSN 0018-9219.

OLIVEIRA, V. F. de et al. A linear logic based synchronization rule for deadlock prevention in web service composition. In: INSTICC. **Proceedings of the 19th International Conference on Enterprise Information Systems - Volume 2: ICEIS**,. ScitePress, 2017. p. 316–323. ISBN 978-989-758-248-6. Disponível em: <<https://doi.org/10.5220/0006308303160323>>.

_____. A synchronization rule based on linear logic for deadlock prevention in interorganizational workflow nets. In: _____. **Information Technology - New Generations: 14th International Conference on Information Technology**. Cham: Springer International Publishing, 2018. p. 929–934. ISBN 978-3-319-54978-1.

PASSOS, L. M. S. **Formalização de workflow nets utilizando lógica linear: análise qualitativa e quantitativa**. Dissertação (Mestrado) — Universidade Federal de Uberlândia, 2009.

PASSOS, L. M. S. **Uma Metodologia baseada na Lógica Linear para Análise de Processos de Workflow Interorganizacionais**. Tese (Doutorado) — Universidade Federal de Uberlândia, 2016.

PASSOS, L. M. S.; JULIA, S. Qualitative analysis of workflow nets using linear logic: Soundness verification. In: **2009 IEEE International Conference on Systems, Man and Cybernetics**. [S.l.: s.n.], 2009. p. 2843–2847. ISSN 1062-922X.

_____. Qualitative analysis of interorganizational workflow nets using linear logic: Soundness verification. In: **2013 IEEE 25th International Conference on Tools with Artificial Intelligence**. [S.l.: s.n.], 2013. p. 667–673. ISSN 1082-3409.

_____. Linear logic as a tool for deadlock-freeness scenarios detection in interorganizational workflow processes. In: **2014 IEEE 26th International Conference on Tools with Artificial Intelligence**. [S.l.: s.n.], 2014. p. 316–320. ISSN 1082-3409.

_____. Deadlock-freeness scenarios detection in web service composition. In: **2015 12th International Conference on Information Technology - New Generations**. [s.n.], 2015. p. 780–783. Disponível em: <<https://doi.org/10.1109/ITNG.2015.141>>.

PETRI, C. A. **Kommunikation mit Automaten**. Tese (Doutorado) — Institut für instrumentelle Mathematik, 1962.

PRADIN-CHÉZALVIEL, B. et al. Calculating duration of concurrent scenarios in time petri nets. **APII-Journal Européen des Systèmes Automatisés**, v. 33, n. 8-9, p. 943–958, 1999.

PRADIN-CHÉZALVIEL, B.; VALETTE, R.; KUNZLE, L. A. Scenario durations characterization of t-timed petri nets using linear logic. In: **Proceedings 8th International Workshop on Petri Nets and Performance Models (Cat. No.PR00331)**. [S.l.: s.n.], 1999. p. 208–217. ISSN 1063-6714.

RAMCHANDANI, C. **Analysis of asynchronous concurrent systems by timed Petri nets**. Tese (Doutorado) — Massachusetts Institute of Technology, 1974.

RATZER, A. V. et al. Cpn tools for editing, simulating, and analysing coloured petri nets. In: _____. **Applications and Theory of Petri Nets 2003: 24th International Conference, ICATPN 2003 Eindhoven, The Netherlands, June 23–27, 2003 Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. p. 450–462. ISBN 978-3-540-44919-5. Disponível em: <https://doi.org/10.1007/3-540-44919-1_28>.

REZENDE, L.; JULIA, S. Deadlock avoidance in interorganizational business processes using a possibilistic workflow net. In: INSTICC. **Proceedings of the 17th International Conference on Enterprise Information Systems - Volume 1: ICEIS**,. SciTePress, 2015. p. 429–439. ISBN 978-989-758-096-3. Disponível em: <<https://doi.org/10.5220/0005347004290439>>.

REZENDE, L. P. de; JULIA, S. Possibilistic workflow net for deadlock avoidance in interorganizational business processes. In: _____. **Enterprise Information Systems: 17th International Conference, ICEIS 2015, Barcelona, Spain, April 27-30, 2015, Revised Selected Papers**. Cham: Springer International Publishing, 2015. p. 168–191. ISBN 978-3-319-29133-8. Disponível em: <https://doi.org/10.1007/978-3-319-29133-8_9>.

RIVIERE, N.; PRADIN-CHÉZALVIEL, B.; VALETTE, R. Reachability and temporal conflicts in t-time petri nets. In: **Proceedings 9th International Workshop on Petri Nets and Performance Models**. [S.l.: s.n.], 2001. p. 229–238. ISSN 1063-6714.

ROZINAT, A. et al. Discovering colored petri nets from event logs. **International Journal on Software Tools for Technology Transfer**, v. 10, n. 1, p. 57–74, Jan 2008. ISSN 1433-2787.

SHIM, J.; HAN, D.; KIM, H. Communication deadlock detection of inter-organizational workflow definition. In: _____. **Databases in Networked Information Systems: Second International Workshop, DNIS 2002 Aizu, Japan, December 16–18, 2002 Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. p. 43–57. ISBN 978-3-540-36233-3. Disponível em: <https://doi.org/10.1007/3-540-36233-9_4>.

SIFAKIS, J. Use of petri nets for performance evaluation. In: **Proceedings of the Third International Symposium on Measuring, Modelling and Evaluating Computer Systems**. Amsterdam, The Netherlands, The Netherlands: North-Holland Publishing Co., 1977. p. 75–93. ISBN 0-444-85058-9.

SILVA, L. d. F. **Detecção e correção de situações de deadlock em workflow nets interorganizacionais**. Dissertação (Mestrado) — Universidade Federal de Uberlândia, 2014.

SILVA, L. de F. et al. Siphon-based deadlock prevention policy for interorganizational workflow net design. In: **2013 IEEE 14th International Conference on Information Reuse Integration (IRI)**. [S.l.: s.n.], 2013. p. 293–300.

SOARES, M. d. S. Uma abordagem baseada num jogador de redes de Petri p-temporal e no cálculo de sequentes da Lógica Linear para a verificação de cenários de Sistemas Tempo Real especificados através de diagramas

dinâmicos da UML. Dissertação (Mestrado) — Universidade Federal de Uberlândia, 2004.

WHITE, S.; MIERS, D. **BPMN Modeling and Reference Guide: Understanding and Using BPMN.** [S.l.]: Future Strategies Incorporated, 2008.

XIONG, P.; FAN, Y.; ZHOU, M. A petri net approach to analysis and composition of web services. **IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans**, v. 40, n. 2, p. 376–387, March 2010. ISSN 1083-4427.

XIONG, P.; ZHOU, M.; PU, C. A petri net siphon based solution to protocol-level service composition mismatches. In: **2009 IEEE International Conference on Web Services**. [s.n.], 2009. p. 952–958. Disponível em: <<https://doi.org/10.1109/ICWS.2009.108>>.

Apêndices

Árvores de Prova da Lógica Linear

Neste apêndice são apresentadas as árvores de prova da Lógica Linear construídas para os exemplos utilizados nesta pesquisa. Além disso, são apresentadas as tabelas com os intervalos de datas simbólicas de execução para as tarefas dos exemplos apresentados.

O Apêndice A.1 apresenta a árvore de prova e a árvore de prova canônica da Lógica Linear com cálculo de datas do cenário Sc_2 , apresentado na seção 2.5, para apresentar a relação das redes de Petri com a Lógica Linear.

O Apêndice A.2 apresenta as árvores de prova, antes da aplicação da regra de sincronização, para os cenários apresentados na seção 4.1, além da árvore de prova do cenário alterado após a aplicação da regra de sincronização.

O Apêndice A.3 apresenta as árvores de prova canônica da Lógica Linear com cálculo de datas e a tabela com intervalos de datas simbólicas de execução para as tarefas do cenário Sc_4 da seção 4.3, antes e depois da aplicação da regra de sincronização.

O Apêndice A.4 apresenta as árvores de prova antes da aplicação da regra de sincronização para os cenários apresentados no estudo de caso da seção 5.2, além das árvores de prova dos cenários alterados após a aplicação da regra de sincronização.

O Apêndice A.5 apresenta as árvores de prova canônica da Lógica Linear com cálculo de datas e as tabelas com intervalos de datas simbólicas de execução para as tarefas dos cenários Sc_1 e Sc_2 apresentados no estudo de caso da seção 5.4, antes e depois da aplicação da regra de sincronização.

A.1 Árvore de Prova da Lógica Linear

A árvore de prova do cenário Sc_2 é mostrada na sequência.

$$\begin{array}{c}
\frac{p6 \vdash p6 \quad o \vdash o}{\multimap_L} \\
\frac{p5 \vdash p5 \quad p6, p6 \multimap o \vdash o}{\multimap_L} \\
\frac{\frac{p3 \vdash p3 \quad p4 \vdash p4}{p3, p4 \vdash p3 \otimes p4} \otimes_R \quad p5, p5 \multimap p6, t_7 \vdash o}{\multimap_L} \\
\frac{p2 \vdash p2 \quad p3, p4, p3 \otimes p4 \multimap p5, t_6, t_7 \vdash o}{\multimap_L} \\
\frac{p1 \vdash p1 \quad p3, p4, p2 \multimap p4, t_4, t_6, t_7 \vdash o}{\multimap_L} \\
\frac{p1, p2, p1 \multimap p3, t_3, t_4, t_6, t_7 \vdash o}{\otimes_L} \\
\frac{i \vdash i \quad p1 \otimes p2, p1 \multimap p3, t_3, t_4, t_6, t_7 \vdash o}{\multimap_L} \\
i, i \multimap p1 \otimes p2, t_2, t_3, t_4, t_6, t_7 \vdash o
\end{array}$$

A árvore de prova canônica da Lógica Linear com cálculo de datas construída para a prova do sequente do cenário Sc_2 é mostrada na sequência, sendo i sua marcação inicial e $Seq = D_i + d_1 + \max(d_2, d_3) + d_4$.

$$\begin{array}{c}
\frac{p6(Seq + \max(d_5, d_6), Seq + \max(d_5, d_6) + d_7) \vdash p6 \quad o(Seq + \max(d_5, d_6) + d_7, \cdot) \vdash o}{\multimap_L} \\
\frac{p5(Seq, Seq + \max(d_5, d_6)) \vdash p5 \quad p6(Seq + \max(d_5, d_6), \cdot), p6 \multimap o \vdash o}{\multimap_L} \\
\frac{\frac{p3(Di + d_1 + d_2, Seq) \vdash p3 \quad p4(Di + d_1 + d_3, Seq) \vdash p4}{p3(Di + d_1 + d_2, Seq), p4(Di + d_1 + d_3, Seq) \vdash p3 \otimes p4} \otimes_R \quad p5(Seq, \cdot), p5 \multimap p6, t_7 \vdash o}{\multimap_L} \\
\frac{p2(Di + d_1 +, Di + d_1 + d_3) \vdash p2 \quad p3(Di + d_1 + d_2, \cdot), p4(Di + d_1 + d_3, \cdot), p3 \otimes p4 \multimap p5, t_5, t_7 \vdash o}{\multimap_L} \\
\frac{p1(Di + d_1 +, Di + d_1 + d_2) \vdash p1 \quad p2(Di + d_1, \cdot), p3(Di + d_1 + d_2, \cdot), p2 \multimap p4, t_4, t_5, t_7 \vdash o}{\multimap_L} \\
\frac{p1(Di + d_1, \cdot), p2(Di + d_1, \cdot), p1 \multimap p3, t_3, t_4, t_5, t_7 \vdash o}{\otimes_L} \\
\frac{i(Di, Di + d_1) \vdash i \quad p1(Di + d_1, \cdot) \otimes p2(Di + d_1, \cdot), p1 \multimap p3, t_3, t_4, t_5, t_7 \vdash o}{\multimap_L} \\
i(Di, \cdot), i \multimap p1 \otimes p2, t_2, t_3, t_4, t_5, t_7 \vdash o
\end{array}$$

A.2 Árvore de Prova para o exemplo apresentado na Proposta do Capítulo 4

A árvore de prova do cenário Sc_1 é a seguinte:

$$\begin{array}{c}
\frac{\frac{efa \vdash efa \quad efp \vdash efp}{efa, efp \vdash efa \otimes efp} \otimes_R \quad o \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{a2 \vdash a2 \quad r \vdash r}{a2, r \vdash a2 \otimes r} \otimes_R \quad efp, efa, efa \otimes efp \multimap o \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{a1 \vdash a1 \quad ad \vdash ad}{a1, ad \vdash a1 \otimes ad} \otimes_R \quad r, efp, a2, a2 \otimes r \multimap efa, t_o \vdash o}{\vdash o} \multimap_L \\
\frac{a1, ad, r, efp, a1 \otimes ad \multimap a2, t_4, o \vdash o}{\vdash o} \otimes_L \\
\frac{p3 \vdash p3 \quad a1, ad, r \otimes efp, t_2, 4, o \vdash o}{\vdash o} \multimap_L \\
\frac{p2 \vdash p2 \quad a1, ad, p3, p3 \multimap r \otimes efp, t_2, 4, o \vdash o}{\vdash o} \multimap_L \\
\frac{a1, ad, p2, p2 \multimap p3, t_2, 4, 13, o \vdash o}{\vdash o} \otimes_L \\
\frac{p1 \vdash p1 \quad a1, ad \otimes p2, t_2, 4, 12, 13, o \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{sfpc \vdash sfpc \quad d \vdash d}{sfpc, d \vdash sfpc \otimes d} \otimes_R \quad a1, p1, p1 \multimap ad \otimes p2, t_2, 4, 12, 13, o \vdash o}{\vdash o} \multimap_L \\
\frac{sfpc, a1, d, sfpc \otimes d \multimap p1, t_2, 4, 11, 12, 13, o \vdash o}{\vdash o} \otimes_L \\
\frac{sfa \vdash sfa \quad sfpc, a1 \otimes d, t_2, 4, 10, 11, 12, 13, o \vdash o}{\vdash o} \multimap_L \\
\frac{sfa, sfpc, sfa \multimap a1 \otimes d, t_2, 4, t_{10}, t_{11}, t_{12}, t_{13}, t_o \vdash o}{\vdash o} \otimes_L \\
\frac{i \vdash i \quad sfa \otimes sfpc, t_1, t_2, t_4, t_{10}, t_{11}, t_{12}, t_{13}, t_o \vdash o}{\vdash o} \multimap_L \\
i, t_i, t_1, t_2, t_4, t_{10}, t_{11}, t_{12}, t_{13}, t_o \vdash o
\end{array}$$

A árvore de prova para o cenário Sc_2 é apresentada a seguir:

$$\begin{array}{c}
\frac{\frac{efa \vdash efa \quad efpc \vdash efpc}{efa, efpc \vdash efa \otimes efpc} \otimes_R \quad o \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{\frac{a3 \vdash a3 \quad tl \vdash tl}{a3, tl \vdash a3 \otimes tl} \otimes_R \quad efpc, efa, efa \otimes efpc \multimap o \vdash o}{\vdash o} \multimap_L \\
\frac{a3, tl, efpc, a3 \otimes tl \multimap efa, to \vdash o}{\vdash o} \otimes_L \\
\frac{p4 \vdash p4 \quad a3, tl \otimes efpc, t6, to \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{\frac{a2 \vdash a2 \quad a \vdash a}{a2, a \vdash a2 \otimes a} \otimes_R \quad p4, a3, t6, p4 \multimap tl \otimes efpc, to \vdash o}{\vdash o} \multimap_L \\
\frac{a2, a, p4, a2 \otimes a \multimap a3, t6, t_{15}, to \vdash o}{\vdash o} \otimes_L \\
\frac{p3 \vdash p3 \quad a2, a \otimes p4, t3, t6, t_{15}, to \vdash o}{\vdash o} \multimap_L \\
\frac{p2 \vdash p2 \quad a2, p3, t3, t6, p3 \multimap a \otimes p4, t_{15}, to \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{\frac{a1 \vdash a1 \quad ad \vdash ad}{a1, ad \vdash a1 \otimes ad} \otimes_R \quad p2, a2, t3, t6, p2 \multimap p3, t_{14}, t_{15}, to \vdash o}{\vdash o} \multimap_L \\
\frac{a1, ad, p2, a1 \otimes ad \multimap a2, t3, t6, t_{12}, t_{14}, t_{15}, to \vdash o}{\vdash o} \otimes_L \\
\frac{p1 \vdash p1 \quad a1, ad \otimes p2, t2, t3, t6, t_{12}, t_{14}, t_{15}, to \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{\frac{sfpc \vdash sfpc \quad d \vdash d}{sfpc, d \vdash sfpc \otimes d} \otimes_R \quad a1, p1, t2, 3, 6, p1 \multimap ad \otimes p2, t_{12}, t_{14}, t_{15}, to \vdash o}{\vdash o} \multimap_L \\
\frac{sfpc, a1, d, t2, t3, t6, sfpc \otimes d \multimap p1, t_{11}, t_{12}, t_{14}, t_{15}, to \vdash o}{\vdash o} \otimes_L \\
\frac{sfa \vdash sfa \quad sfpc, a1 \otimes d, t2, t3, t6, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, to \vdash o}{\vdash o} \multimap_L \\
\frac{sfa, sfpc, sfa \multimap a1 \otimes d, t2, t3, t6, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, to \vdash o}{\vdash o} \otimes_L \\
\frac{i \vdash i \quad sfa \otimes sfpc, t1, t2, t3, t6, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, to \vdash o}{\vdash o} \multimap_L \\
i, t_i, t_1, t_2, t_3, t_6, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, to \vdash o
\end{array}$$

A árvore de prova para o cenário Sc_3 é dada por:

$$\begin{array}{c}
\frac{\frac{efa \vdash efa \quad efpc \vdash efpc}{efa, efpc \vdash efa \otimes efpc} \otimes_R \quad o \vdash o}{\quad} \multimap_L \\
\frac{\frac{a4 \vdash a4 \quad tl \vdash tl}{a4, tl \vdash a4 \otimes tl} \otimes_R \quad efpc, efa, efa \otimes efpc \multimap o \vdash o}{\quad} \multimap_L \\
\frac{a3 \vdash a3 \quad tl, efpc, a4, a4 \otimes tl \multimap efa, to \vdash o}{\quad} \multimap_L \\
\frac{a3, tl, efpc, a3 \multimap a4, t8, to \vdash o}{\quad} \otimes_L \\
\frac{p4 \vdash p4 \quad a3, tl \otimes efpc, t5, t8, to \vdash o}{\quad} \multimap_L \\
\frac{\frac{a2 \vdash a2 \quad a \vdash a}{a2, a \vdash a2 \otimes a} \otimes_R \quad p4, a3, t5, t8, p4 \multimap tl \otimes efpc, to \vdash o}{\quad} \multimap_L \\
\frac{a2, a, p4, a2 \otimes a \multimap a3, t5, t8, t15, to \vdash o}{\quad} \otimes_L \\
\frac{p3 \vdash p3 \quad a2, a \otimes p4, t3, t5, t8, t15, to \vdash o}{\quad} \multimap_L \\
\frac{p2 \vdash p2 \quad a2, p3, t3, t5, t8, p3 \multimap a \otimes p4, t15, to \vdash o}{\quad} \multimap_L \\
\frac{\frac{a1 \vdash a1 \quad ad \vdash ad}{a1, ad \vdash a1 \otimes ad} \otimes_R \quad p2, a2, t3, t5, t8, p2 \multimap p3, t14, t15, to \vdash o}{\quad} \multimap_L \\
\frac{a1, ad, p2, a1 \otimes ad \multimap a2, t3, t5, t8, t12, t14, t15, to \vdash o}{\quad} \otimes_L \\
\frac{p1 \vdash p1 \quad a1, ad \otimes p2, t2, t3, t5, t8, t12, t14, t15, to \vdash o}{\quad} \multimap_L \\
\frac{\frac{sfpc \vdash sfpc \quad d \vdash d}{sfpc, d \vdash sfpc \otimes d} \otimes_R \quad a1, p1, t2, 3, 5, 8, p1 \multimap ad \otimes p2, t12, t14, t15, to \vdash o}{\quad} \multimap_L \\
\frac{sfpc, a1, d, t2, t3, t5, t8, sfpc \otimes d \multimap p1, t11, t12, t14, t15, to \vdash o}{\quad} \otimes_L \\
\frac{sfa \vdash sfa \quad sfpc, a1 \otimes d, t2, t3, t5, t8, t10, t11, t12, t14, t15, to \vdash o}{\quad} \multimap_L \\
\frac{sfa, sfpc, sfa \multimap a1 \otimes d, t2, t3, t5, t8, t10, t11, t12, t14, t15, to \vdash o}{\quad} \otimes_L \\
\frac{i \vdash i \quad sfa \otimes sfpc, t1, t2, t3, t5, t8, t10, t11, t12, t14, t15, to \vdash o}{\quad} \multimap_L \\
i, t_i, t_1, t_2, t_3, t_5, t_8, t_{10}, t_{11}, t_{12}, t_{14}, t_{15}, to \vdash o
\end{array}$$

A árvore de prova do cenário Sc_4 antes da aplicação da regra de sincronização é apresentada na sequência:

$$\begin{array}{c}
\frac{\frac{efa \vdash efa \quad efp \vdash efp}{efa, efp \vdash efa \otimes efp} \otimes_R \quad o \vdash o}{\rightarrow_L} \\
\frac{\frac{a5 \vdash a5 \quad af \vdash af}{a5, af \vdash a5 \otimes af} \otimes_R \quad efp, efa, efa \otimes efp \multimap o \vdash o}{\rightarrow_L} \\
\frac{a5, af, efp, a5 \otimes af \multimap efa, t_o \vdash o}{\otimes_L} \\
\frac{p5b \vdash p5b \quad a5, af \otimes efp, a5 \otimes af \multimap efa, t_o \vdash o}{\rightarrow_L} \\
\frac{p5a \vdash p5a \quad a5, p5b, t_9, t_o \vdash o}{\rightarrow_L} \\
\frac{a5, p5a, p5a \multimap p5b, t_9, t_{17}, t_o \vdash o}{\otimes_L} \\
\frac{\frac{a4 \vdash a4 \quad p4 \vdash p4}{a4, p4 \vdash a4 \otimes p4} \otimes_R \quad a5 \otimes p5a, t_9, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{a3 \vdash a3 \quad p4, a4, a4 \otimes p4 \multimap a5 \otimes p5a, t_9, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{\frac{a2 \vdash a2 \quad a \vdash a}{a2, a \vdash a2 \otimes a} \otimes_R \quad p4, a3, a3 \multimap a4, t_7, t_9, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{a2, a, p4, a2 \otimes a \multimap a3, t_5, t_7, t_9, t_{16}, t_{17}, t_o \vdash o}{\otimes_L} \\
\frac{p3 \vdash p3 \quad a2, a \otimes p4, t_3, t_5, t_7, t_9, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{p2 \vdash p2 \quad a2, p3, t_3, t_5, t_7, t_9, p3 \multimap a \otimes p4, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{\frac{a1 \vdash a1 \quad ad \vdash ad}{a1, ad \vdash a1 \otimes ad} \otimes_R \quad p2, a2, t_3, t_5, t_7, t_9, p2 \multimap p3, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{a1, ad, p2, a1 \otimes ad \multimap a2, t_3, t_5, t_7, t_9, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\otimes_L} \\
\frac{p1 \vdash p1 \quad a1, ad \otimes p2, t_2, t_3, t_5, t_7, t_9, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{\frac{sfpc \vdash sfpc \quad d \vdash d}{sfpc, d \vdash sfpc \otimes d} \otimes_R \quad a1, p1, t_2, t_3, t_5, t_7, t_9, p1 \multimap ad \otimes p2, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{sfpc, a1, d, t_2, t_3, t_5, t_7, t_9, sfpc \otimes d \multimap p1, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\otimes_L} \\
\frac{sfa \vdash sfa \quad sfpc, a1 \otimes d, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
\frac{sfa, sfpc, sfa \multimap a1 \otimes d, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\otimes_L} \\
\frac{i \vdash i \quad sfa \otimes sfpc, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o}{\rightarrow_L} \\
i, t_i, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, t_o \vdash o
\end{array}$$

A árvore de prova para o cenário Sc_5 é apresentada a seguir:

$$\begin{array}{c}
\frac{tl,efpc,a5,fv,t9,t_o \vdash o}{\otimes_L} \\
\frac{a4 \vdash a4 \quad tl,efpc,a5 \otimes fv,t9,t_o \vdash o}{\multimap_L} \\
\frac{a3 \vdash a3 \quad tl,efpc,a4,a4 \multimap a5 \otimes fv,t9,t_o \vdash o}{\multimap_L} \\
\frac{a3,tl,efpc,a3 \multimap a4,t_7,t9,t_o \vdash o}{\otimes_L} \\
\frac{p4 \vdash p4 \quad a3,tl \otimes efpc,t_5,t_7,t9,t_o \vdash o}{\multimap_L} \\
\frac{\frac{a2 \vdash a2}{a2,a \vdash a2 \otimes a} \quad \frac{a \vdash a}{\otimes_R} \quad p4,a3,t_5,t_7,t9,p4 \multimap tl \otimes efpc,t_o \vdash o}{\multimap_L} \\
\frac{a2,a,p4,a2 \otimes a \multimap a3,t_5,t_7,t9,t_{15},t_o \vdash o}{\otimes_L} \\
\frac{p3 \vdash p3 \quad a2,a \otimes p4,t_3,t_5,t_7,t9,t_{15},t_o \vdash o}{\multimap_L} \\
\frac{p2 \vdash p2 \quad a2,p3,t_3,t_5,t_7,t9,p3 \multimap a \otimes p4,t_{15},t_o \vdash o}{\multimap_L} \\
\frac{\frac{a1 \vdash a1}{a1,ad \vdash a1 \otimes ad} \quad \frac{ad \vdash ad}{\otimes_R} \quad p2,a2,t_3,t_5,t_7,t9,p2 \multimap p3,t_{14},t_{15},t_o \vdash o}{\multimap_L} \\
\frac{a1,ad,p2,a1 \otimes ad \multimap a2,t_3,t_5,t_7,t9,t_{12},t_{14},t_{15},t_o \vdash o}{\otimes_L} \\
\frac{p1 \vdash p1 \quad a1,ad \otimes p2,t_2,t_3,t_5,t_7,t9,t_{12},t_{14},t_{15},t_o \vdash o}{\multimap_L} \\
\frac{\frac{sfpc \vdash sfpc}{sfpc,d \vdash sfpc \otimes d} \quad \frac{d \vdash d}{\otimes_R} \quad a1,p1,t_2,3,5,7,9,p1 \multimap ad \otimes p2,t_{12},t_{14},t_{15},t_o \vdash o}{\multimap_L} \\
\frac{sfpc,a1,d,t_2,t_3,t_5,t_7,t9,sfpc \otimes d \multimap p1,t_{11},t_{12},t_{14},t_{15},t_o \vdash o}{\otimes_L} \\
\frac{sfa \vdash sfa \quad sfpc,a1 \otimes d,t_2,t_3,t_5,t_7,t9,t_{10},t_{11},t_{12},t_{14},t_{15},t_o \vdash o}{\multimap_L} \\
\frac{sfa,sfpc,sfa \multimap a1 \otimes d,t_2,t_3,t_5,t_7,t9,t_{10},t_{11},t_{12},t_{14},t_{15},t_o \vdash o}{\otimes_L} \\
\frac{i \vdash i \quad sfa \otimes sfpc,t_1,t_2,t_3,t_5,t_7,t9,t_{10},t_{11},t_{12},t_{14},t_{15},t_o \vdash o}{\multimap_L} \\
i,t_i,t_1,t_2,t_3,t_5,t_7,t9,t_{10},t_{11},t_{12},t_{14},t_{15},t_o \vdash o
\end{array}$$

A árvore de prova para o cenário Sc_4 alterado após a aplicação da regra de sincronização é a seguinte:

$$\begin{array}{c}
\frac{\frac{efa \vdash efa \quad efp \vdash efp}{efa, efp \vdash efa \otimes efp} \otimes_R \quad o \vdash o}{\neg o} \neg_L \\
\frac{\frac{\frac{a5 \vdash a5 \quad af \vdash af}{a5, af \vdash a5 \otimes af} \otimes_R \quad efp, efa, efa \otimes efp \neg o \vdash o}{a5, af, efp, a5 \otimes af \neg efa, t_o \vdash o} \otimes_L \\
\frac{p5 \vdash p5 \quad a5, af \otimes efp, t_9, t_o \vdash o}{fv \vdash fv \quad p5, p5 \neg af \otimes efp, t_9, t_o \vdash o} \neg_L \\
\frac{a5, fv, fv \neg p5, t_9, t_{17}, t_o \vdash o}{\neg o} \neg_L \\
\frac{\frac{\frac{a4 \vdash a4 \quad p4 \vdash p4}{a4, p4 \vdash a4 \otimes p4} \otimes_R \quad a5 \otimes fv, t_9, t_{16}, t_{17}, t_o \vdash o}{a3 \vdash a3 \quad a4, p4, a4 \otimes p4 \neg a5 \otimes fv, t_9, t_{16}, t_{17}, t_o \vdash o} \neg_L \\
\frac{\frac{\frac{a2 \vdash a2 \quad a \vdash a}{a2, a \vdash a2 \otimes a} \otimes_R \quad p4, a3, a3 \neg a4, t_7, t_9, t_{16}, t_{17}, o \vdash o}{a2, a, p4, a2 \otimes a \neg a3, t_5, t_7, t_9, t_{16}, t_{17}, o \vdash o} \otimes_L \\
\frac{p3 \vdash p3 \quad a2, a \otimes p4, t_3, t_5, t_7, t_9, t_{16}, t_{17}, o \vdash o}{p2 \vdash p2 \quad a2, p3, t_3, t_5, t_7, t_9, p3 \neg a \otimes p4, t_{16}, t_{17}, o \vdash o} \neg_L \\
\frac{\frac{\frac{a1 \vdash a1 \quad ad \vdash ad}{a1, ad \vdash a1 \otimes ad} \otimes_R \quad p2, a2, t_3, t_5, t_7, t_9, p2 \neg p3, t_{14}, t_{16}, t_{17}, o \vdash o}{a1, ad, p2, a1 \otimes ad \neg a2, t_3, t_5, t_7, t_9, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o} \otimes_L \\
\frac{p1 \vdash p1 \quad a1, ad \otimes p2, t_2, t_3, t_5, t_7, t_9, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{\neg o} \neg_L \\
\frac{\frac{\frac{sfpc \vdash sfpc \quad d \vdash d}{sfpc, d \vdash sfpc \otimes d} \otimes_R \quad a1, p1, t_2, t_3, t_5, t_7, t_9, p1 \neg ad \otimes p2, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{sfpc, a1, d, t_2, t_3, t_5, t_7, t_9, sfpc \otimes d \neg p1, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o} \otimes_L \\
\frac{sfa \vdash sfa \quad sfpc, a1 \otimes d, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{sfa, sfpc, sfa \neg a1 \otimes d, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o} \otimes_L \\
\frac{i \vdash i \quad sfa \otimes sfpc, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o}{i, t_i, t_1, t_2, t_3, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{14}, t_{16}, t_{17}, o \vdash o} \neg_L
\end{array}$$

A.3 Árvore de Prova Canônica para o exemplo apresentado na Proposta do Capítulo 4

Para a árvore de prova da Lógica Linear com cálculo de datas construída para a prova do sequente do cenário Sc_4 antes da aplicação da regra de sincronização, apresentada em Passos (2016), considerar $Seq = d_{10} + d_{11} + d_{12} + d_{14}$ e $Seq1 = D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3$.

Para a árvore de prova da Lógica Linear com cálculo de datas construída para a prova do sequente do novo cenário Sc_4 depois da aplicação da regra de sincronização, considerar $Seq = d_{10} + d_{11} + d_{12} + d_{14}$, $Seq1 = D_i + d_i + d_1 + d_{10} + d_{11} + \max\{d_2, d_{12} + d_{14}\} + d_3$ e $Seq2 = \max\{Seq1 + d_5, D_i + d_i + d_1 + Seq\} + d_7$.

Para a Tabela 14 apresentada nesta seção considerar $Seq_{min} = d_{10min} + d_{11min} + d_{12min} + d_{14min}$, $Seq_{max} = d_{10max} + d_{11max} + d_{12max} + d_{14max}$, $Seq1_{min} = D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + \max\{d_{2min}, d_{12min} + d_{14min}\} + d_{3min}$, $Seq1_{max} = D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + \max\{d_{2max}, d_{12max} + d_{14max}\} + d_{3max}$, $Seq2_{min} = \max\{Seq1_{min} + d_{5min}, D_i + d_{imin} + d_{1min} + Seq_{min}\} + d_{7min}$ e $Seq2_{max} = \max\{Seq1_{max} + d_{5max}, D_i + d_{imax} + d_{1max} + Seq_{max}\} + d_{7max}$.

Tabela 14 – Intervalos de datas simbólicas de execução para as tarefas do cenário Sc_4 .

Átomo	Antes da aplicação da regra de sincronização	Depois da aplicação da regra de sincronização
t_i	$[D_i,$ $D_i + d_{imax} + d_{1max} + d_{10max}]$	$[D_i,$ $D_i + d_{imax} + d_{1max} + d_{10max}]$
$send_draft$ (t_1)	$[D_i + d_{imin},$ $D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + d_{2max}]$	$[D_i + d_{imin},$ $D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + d_{2max}]$
$receive_draft$ (t_{10})	$[D_i + d_{imin},$ $D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max}]$	$[D_i + d_{imin},$ $D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max}]$
$send_ack_draft$ (t_{11})	$[D_i + d_{imin} + d_{1min} + d_{10min},$ $max\{D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + d_{2max},$ $D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + d_{12max}\}]$	$[D_i + d_{imin} + d_{1min} + d_{10min},$ $max\{D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + d_{2max},$ $D_i + d_{imax} + d_{1max} + d_{10max} + d_{11max} + d_{12max}\}]$
$evaluate$ (t_{12})	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min},$ $Seq_{1max}]$	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min},$ $Seq_{1max}]$
$send_accept$ (t_{14})	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + d_{12min},$ $Seq_{1max} + d_{5max}]$	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + d_{12min},$ $Seq_{1max} + d_{5max}]$
$receive_ack_draft$ (t_2)	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min},$ $D_i + d_{imax} + d_{1max} + Seq_{max}]$	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min},$ $D_i + d_{imax} + d_{1max} + Seq_{max}]$
$receive_accept$ (t_3)	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + d_{2min},$ $Seq_{1max} + d_{5max}]$	$[D_i + d_{imin} + d_{1min} + d_{10min} + d_{11min} + d_{2min},$ $Seq_{1max} + d_{5max}]$
$prepare_final_version$ (t_5)	$[Seq_{1min},$ $Seq_{1max} + d_{5max} + d_{7max} + d_{16max}]$	$[Seq_{1min},$ $Seq_{2max}]$
$send_final_version$ (t_7)	$[Seq_{1min} + d_{5min},$ $Seq_{1max} + d_{5max} + d_{7max} + d_{16max} + d_{17max}]$	$[max\{Seq_{1min} + d_{5min}, D_i + d_{imin} + d_{1min} + Seq_{min}\},$ $Seq_{2max} + d_{16max}]$
$receive_final_version$ (t_{16})	$[max\{D_i + d_{imin} + d_{1min} + Seq_{min},$ $Seq_{1min} + d_{5min} + d_{7min}\},$ $Seq_{1max} + d_{5max} + d_{7max} + d_{16max} + d_{17max} + d_{9max}]$	$[Seq_{2min},$ $Seq_{2max} + d_{16max} + d_{17max}]$
$send_ack_final$ (t_{17})	$[Seq_{1min} + d_{5min} + d_{7min} + d_{16min},$ $Seq_{1max} + d_{5max} + d_{7max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$	$[Seq_{2min} + d_{16min},$ $Seq_{2max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$
$receive_ack_final$ (t_9)	$[Seq_{1min} + d_{5min} + d_{7min} + d_{16min} + d_{17min},$ $Seq_{1max} + d_{5max} + d_{7max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$	$[Seq_{2min} + d_{16min} + d_{17min},$ $Seq_{2max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$
t_o	$[Seq_{1min} + d_{5min} + d_{7min} + d_{16min} + d_{17min} + d_{9min},$ $Seq_{1max} + d_{5max} + d_{7max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$	$[Seq_{2min} + d_{16min} + d_{17min} + d_{9min},$ $Seq_{2max} + d_{16max} + d_{17max} + d_{9max} + d_{omax}]$

A.4 Árvore de Prova para o exemplo apresentado no Estudo de Caso do Capítulo 5

A árvore de prova para o cenário Sc_1 é a seguinte:

$$\begin{array}{c}
\frac{\frac{q2 \vdash q2 \quad p3 \vdash p3}{q2, p3 \vdash q2 \otimes p3} \otimes_R \quad o \vdash o}{\neg_L} \\
\frac{\frac{p1 \vdash p1 \quad eCash \vdash eCash}{p1, eCash \vdash p1 \otimes eCash} \otimes_R \quad q2, p3, q2 \otimes p3 \neg o \vdash o}{\neg_L} \\
\frac{}{p1, eCash, q2, p1 \otimes eCash \neg p3, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p1, eCash \otimes q2, t_6, t_o \vdash o}{\neg_L} \\
\frac{\frac{q0 \vdash q0 \quad ticket \vdash ticket}{q0, ticket \vdash q0 \otimes ticket} \otimes_R \quad p1, q1, q1 \neg eCash \otimes q2, t_6, t_o \vdash o}{\neg_L} \\
\frac{}{q0, ticket, p1, q0 \otimes ticket \neg q1, t_3, t_6, t_o \vdash o} \otimes_L \\
\frac{p0 \vdash p0 \quad q0, ticket \otimes p1, t_1, t_3, t_6, t_o \vdash o}{\neg_L} \\
\frac{}{q0, p0, t_1, t_3, p0 \neg ticket \otimes p1, t_6, t_o \vdash o} \otimes_L \\
\frac{i \vdash i \quad q0 \otimes p0, t_1, t_3, t_4, t_6, t_o \vdash o}{\neg_L} \\
i, i \neg q0 \otimes p0, t_1, t_3, t_4, t_6, t_o \vdash o
\end{array}$$

A árvore de prova para o cenário Sc_2 é apresentada a seguir:

$$\begin{array}{c}
\frac{\frac{q2 \vdash q2 \quad p3 \vdash p3}{q2, p3 \vdash q2 \otimes p3} \otimes_R \quad o \vdash o}{\neg_L} \\
\frac{\frac{p2 \vdash p2 \quad VISA \vdash VISA}{p2, VISA \vdash p2 \otimes VISA} \otimes_R \quad q2, p3, q2 \otimes p3 \neg o \vdash o}{\neg_L} \\
\frac{}{p2, VISA, q2, p2 \otimes VISA \neg p3, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p2, VISA \otimes q2, t_7, t_o \vdash o}{\neg_L} \\
\frac{\frac{q0 \vdash q0 \quad ticket \vdash ticket}{q0, ticket \vdash q0 \otimes ticket} \otimes_R \quad p2, q1, q1 \neg VISA \otimes q2, t_7, t_o \vdash o}{\neg_L} \\
\frac{}{q0, ticket, p2, q0 \otimes ticket \neg q1, t_2, t_7, t_o \vdash o} \otimes_L \\
\frac{p0 \vdash p0 \quad q0, ticket \otimes p2, t_1, t_2, t_7, t_o \vdash o}{\neg_L} \\
\frac{}{q0, p0, t_1, t_2, p0 \neg ticket \otimes p2, t_7, t_o \vdash o} \otimes_L \\
\frac{i \vdash i \quad q0 \otimes p0, t_1, t_2, t_5, t_7, t_o \vdash o}{\neg_L} \\
i, i \neg q0 \otimes p0, t_1, t_2, t_5, t_7, t_o \vdash o
\end{array}$$

A árvore de prova para o cenário Sc_3 é dada por:

$$\begin{array}{c}
\frac{}{p1, VISA, q2, t_6, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p1, VISA \otimes q2, t_6, t_o \vdash o}{\quad} \multimap_L \\
\frac{\frac{q0 \vdash q0 \quad ticket \vdash ticket}{q0, ticket \vdash q0 \otimes ticket} \otimes_R \quad p1, q1, q1 \multimap VISA \otimes q2, t_6, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, ticket, p1, q0 \otimes ticket \multimap q1, t_2, t_6, t_o \vdash o} \otimes_L \\
\frac{p0 \vdash p0 \quad q0, ticket \otimes p1, t_1, t_2, t_6, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, p0, t_1, t_2, p0 \multimap ticket \otimes p1, t_6, t_o \vdash o} \otimes_L \\
\frac{i \vdash i \quad q0 \otimes p0, t_1, t_2, t_4, t_6, t_o \vdash o}{\quad} \multimap_L \\
i, i \multimap q0 \otimes p0, t_1, t_2, t_4, t_6, t_o \vdash o
\end{array}$$

E finalmente a árvore de prova para o cenário Sc_4 é a seguinte:

$$\begin{array}{c}
\frac{}{p2, eCash, q2, t_7, t_o \vdash o} \otimes_L \\
\frac{q1 \vdash q1 \quad p2, eCash \otimes q2, t_7, t_o \vdash o}{\quad} \multimap_L \\
\frac{\frac{q0 \vdash q0 \quad ticket \vdash ticket}{q0, ticket \vdash q0 \otimes ticket} \otimes_R \quad p2, q1, q1 \multimap eCash \otimes q2, t_7, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, ticket, p2, q0 \otimes ticket \multimap q1, t_3, t_7, t_o \vdash o} \otimes_L \\
\frac{p0 \vdash p0 \quad q0, ticket \otimes p2, t_1, t_3, t_7, t_o \vdash o}{\quad} \multimap_L \\
\frac{}{q0, p0, t_1, t_3, p0 \multimap ticket \otimes p2, t_7, t_o \vdash o} \otimes_L \\
\frac{i \vdash i \quad q0 \otimes p0, t_1, t_3, t_5, t_7, t_o \vdash o}{\quad} \multimap_L \\
i, i \multimap q0 \otimes p0, t_1, t_3, t_5, t_7, t_o \vdash o
\end{array}$$

A.5 Árvore de Prova Canônica para o exemplo apresentado no Estudo de Caso do Capítulo 5

Considerando que $Seq = D_i + d_i + d_4 + d_1$, a árvore de prova da Lógica Linear com cálculo de datas construída para a prova do sequente do cenário Sc_1 antes da aplicação da regra de sincronização é apresentada a seguir:

$$\begin{array}{c}
\frac{\frac{q2(Seq+d_3, Seq+d_3+d_6+d_o) \vdash q2 \quad p3(Seq+d_3+d_6, Seq+d_3+d_6+d_o) \vdash p3}{q2(Seq+d_3, Seq+d_3+d_6+d_o), p3(Seq+d_3+d_6, Seq+d_3+d_6+d_o) \vdash q2 \otimes p3} \otimes_R \quad o(Seq+d_3+d_6+d_o, \cdot) \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{p1(D_i+d_i+d_4, Seq+d_3+d_6) \vdash p1 \quad eCash(Seq+d_3, Seq+d_3+d_6) \vdash eCash}{p1(D_i+d_i+d_4, Seq+d_3+d_6), eCash(Seq+d_3, Seq+d_3+d_6) \vdash p1 \otimes eCash} \otimes_R \quad q2(Seq+d_3, \cdot), p3(Seq+d_3+d_6, \cdot), q2 \otimes p3 \multimap o \vdash o}{\vdash o} \multimap_L \\
\frac{\vdash o}{p1(D_i+d_i+d_4, \cdot), eCash(Seq+d_3, \cdot), q2(Seq+d_3, \cdot), p1 \otimes eCash \multimap p3, t_o \vdash o} \otimes_L \\
\frac{\vdash o}{q1(Seq, Seq+d_3) \vdash q1 \quad p1(D_i+d_i+d_4, \cdot), eCash(Seq+d_3, \cdot) \otimes q2(Seq+d_3, \cdot), t_6, t_o \vdash o} \multimap_L \\
\frac{\frac{q0(D_i+d_i, Seq) \vdash q0 \quad Ticket(D_i+d_i+d_4, Seq) \vdash Ticket}{q0(D_i+d_i, Seq), Ticket(D_i+d_i+d_4, Seq) \vdash q0 \otimes Ticket} \otimes_R \quad p1(D_i+d_i+d_4, \cdot), q1(Seq, \cdot), q1 \multimap eCash \otimes q2, t_6, t_o \vdash o}{\vdash o} \multimap_L \\
\frac{\vdash o}{q0(D_i+d_i, \cdot), Ticket(D_i+d_i+d_4, \cdot), p1(D_i+d_i+d_4, \cdot), q0 \otimes Ticket \multimap q1, t_3, t_6, t_o \vdash o} \otimes_L \\
\frac{\vdash o}{p0(D_i+d_i, D_i+d_i+d_4) \vdash p0 \quad q0(D_i+d_i, \cdot), Ticket(D_i+d_i+d_4, \cdot) \otimes p1(D_i+d_i+d_4, \cdot), t_1, t_3, t_6, t_o \vdash o} \multimap_L \\
\frac{\vdash o}{q0(D_i+d_i, \cdot), p0(D_i+d_i, \cdot), t_1, t_3, p0 \multimap Ticket \otimes p1, t_6, t_o \vdash o} \otimes_L \\
\frac{\vdash o}{i(D_i, D_i+d_i) \vdash i \quad q0(D_i+d_i, \cdot) \otimes p0(D_i+d_i, \cdot), t_1, t_3, t_4, t_6, t_o \vdash o} \multimap_L \\
\frac{\vdash o}{i(D_i, \cdot), i \multimap q0 \otimes p0, t_1, t_3, t_4, t_6, t_o \vdash o} \multimap_L
\end{array}$$

Considerando que $Seq = D_i + d_i + d_5 + d_1$, a árvore de prova da Lógica Linear com cálculo de datas construída para a prova do sequente do cenário Sc_2 antes da aplicação da regra de sincronização é dada por:

$$\begin{array}{c}
\frac{\frac{q2(Seq+d_2, Seq+d_2+d_7+d_o) \vdash q2 \quad p3(Seq+d_2+d_7, Seq+d_2+d_7+d_o) \vdash p3}{q2(Seq+d_2, Seq+d_2+d_7+d_o), p3(Seq+d_2+d_7, Seq+d_2+d_7+d_o) \vdash q2 \otimes p3} \otimes_R \quad o(Seq+d_2+d_7+d_o, \cdot) \vdash o}{\vdash o} \multimap_L \\
\frac{\frac{p2(D_i+d_i+d_5, Seq+d_2+d_7) \vdash p2 \quad VISA(Seq+d_2, Seq+d_2+d_7) \vdash VISA}{p2(D_i+d_i+d_5, Seq+d_2+d_7), VISA(Seq+d_2, Seq+d_2+d_7) \vdash p2 \otimes VISA} \otimes_R \quad q2(Seq+d_2, \cdot), p3(Seq+d_2+d_7, \cdot), q2 \otimes p3 \multimap o \vdash o}{\vdash o} \multimap_L \\
\frac{\vdash o}{p2(D_i+d_i+d_5, \cdot), VISA(Seq+d_2, \cdot), q2(Seq+d_2, \cdot), p2 \otimes VISA \multimap p3, t_o \vdash o} \otimes_L \\
\frac{\vdash o}{q1(Seq, Seq+d_2) \vdash q1 \quad p2(D_i+d_i+d_5, \cdot), VISA(Seq+d_2, \cdot) \otimes q2(Seq+d_2, \cdot), t_7, t_o \vdash o} \multimap_L \\
\frac{\frac{q0(D_i+d_i, Seq) \vdash q0 \quad Ticket(D_i+d_i+d_5, Seq) \vdash Ticket}{q0(D_i+d_i, Seq), Ticket(D_i+d_i+d_5, Seq) \vdash q0 \otimes Ticket} \otimes_R \quad p2(D_i+d_i+d_5, \cdot), q1(Seq, \cdot), q1 \multimap VISA \otimes q2, t_7, t_o \vdash o}{\vdash o} \multimap_L \\
\frac{\vdash o}{q0(D_i+d_i, \cdot), Ticket(D_i+d_i+d_5, \cdot), p2(D_i+d_i+d_5, \cdot), q0 \otimes Ticket \multimap q1, t_2, t_7, t_o \vdash o} \otimes_L \\
\frac{\vdash o}{p0(D_i+d_i, D_i+d_i+d_5) \vdash p0 \quad q0(D_i+d_i, \cdot), Ticket(D_i+d_i+d_5, \cdot) \otimes p2(D_i+d_i+d_5, \cdot), t_1, t_2, t_7, t_o \vdash o} \multimap_L \\
\frac{\vdash o}{q0(D_i+d_i, \cdot), p0(D_i+d_i, \cdot), t_1, t_2, p0 \multimap Ticket \otimes p2, t_7, t_o \vdash o} \otimes_L \\
\frac{\vdash o}{i(D_i, D_i+d_i) \vdash i \quad q0(D_i+d_i, \cdot) \otimes p0(D_i+d_i, \cdot), t_1, t_2, t_5, t_7, t_o \vdash o} \multimap_L \\
\frac{\vdash o}{i(D_i, \cdot), i \multimap q0 \otimes p0, t_1, t_2, t_5, t_7, t_o \vdash o} \multimap_L
\end{array}$$

Considerando que $Seq = D_i + d_i + d_4 + d_1$ e $Seq1 = \max\{D_i + d_i + d_4, D_i + d_i + d_4 + d_1\} + d_3$, a árvore de prova da Lógica Linear com cálculo de datas construída para a prova do sequente do cenário Sc_1 depois da aplicação da regra de sincronização é a seguinte:

$$\begin{array}{c}
\frac{\frac{q2(Seq1, Seq1+d_6+d_o) \vdash q2 \quad p3(Seq1+d_6, Seq1+d_6+d_o) \vdash p3}{q2(Seq1, Seq1+d_6+d_o), p3(Seq1+d_6, Seq1+d_6+d_o) \vdash q2 \otimes p3} \otimes_R \quad o(Seq1+d_6+d_o, \cdot) \vdash o}{\frac{eCash(Seq1, Seq1+d_6) \vdash eCash \quad q2(Seq1, \cdot), p3(Seq1+d_6, \cdot), q2 \otimes p3 \multimap o \vdash o}{eCash(Seq1, \cdot), q2(Seq1, \cdot), eCash \multimap p3, t_o \vdash o} \otimes_L} \multimap_L \\
\frac{\frac{q1(Seq, Seq1) \vdash q1 \quad p1(D_i+d_i+d_4, Seq1) \vdash p1}{q1(Seq, Seq1), p1(D_i+d_i+d_4, Seq1) \vdash q1 \otimes p1} \otimes_R \quad eCash(Seq1, \cdot) \otimes q2(Seq1, \cdot), t_6, t_o \vdash o}{\frac{q0(D_i+d_i, Seq) \vdash q0 \quad Ticket(D_i+d_i+d_4, Seq) \vdash Ticket}{q0(D_i+d_i, Seq), Ticket(D_i+d_i+d_4, Seq) \vdash q0 \otimes Ticket} \otimes_R \quad p1(D_i+d_i+d_4, \cdot), q1(Seq, \cdot), q1 \otimes p1 \multimap eCash \otimes q2, t_6, t_o \vdash o} \multimap_L \\
\frac{p0(D_i+d_i, D_i+d_i+d_4) \vdash p0 \quad q0(D_i+d_i, \cdot), Ticket(D_i+d_i+d_4, \cdot) \otimes p1(D_i+d_i+d_4, \cdot), t_1, t_3, t_6, t_o \vdash o}{q0(D_i+d_i, \cdot), p0(D_i+d_i, \cdot), t_1, t_3, p0 \multimap Ticket \otimes p1, t_6, t_o \vdash o} \otimes_L \\
\frac{i(D_i, D_i+d_i) \vdash i \quad q0(D_i+d_i, \cdot) \otimes p0(D_i+d_i, \cdot), t_1, t_3, t_4, t_6, t_o \vdash o}{i(D_i, \cdot), i \multimap q0 \otimes p0, t_1, t_3, t_4, t_6, t_o \vdash o} \multimap_L
\end{array}$$

Considerando que $Seq = D_i + d_i + d_5 + d_1$ e $Seq1 = \max\{D_i + d_i + d_5, D_i + d_i + d_5 + d_1\} + d_2$, a árvore de prova da Lógica Linear com cálculo de datas construída para a prova do sequente do cenário Sc_2 depois da aplicação da regra de sincronização é apresentada a seguir:

$$\begin{array}{c}
\frac{\frac{q2(Seq1, Seq1+d_7+d_o) \vdash q2 \quad p3(Seq1+d_7, Seq1+d_7+d_o) \vdash p3}{q2(Seq1, Seq1+d_7+d_o), p3(Seq1+d_7, Seq1+d_7+d_o) \vdash q2 \otimes p3} \otimes_R \quad o(Seq1+d_7+d_o, \cdot) \vdash o}{\frac{VISA(Seq1, Seq1+d_7) \vdash VISA \quad q2(Seq1, \cdot), p3(Seq1+d_7, \cdot), q2 \otimes p3 \multimap o \vdash o}{VISA(Seq1, \cdot), q2(Seq1, \cdot), VISA \multimap p3, t_o \vdash o} \otimes_L} \multimap_L \\
\frac{\frac{q1(Seq, Seq1) \vdash q1 \quad p2(D_i+d_i+d_5, Seq1) \vdash p2}{q1(Seq, Seq1), p2(D_i+d_i+d_5, Seq1) \vdash q1 \otimes p2} \otimes_R \quad VISA(Seq1, \cdot) \otimes q2(Seq1, \cdot), t_7, t_o \vdash o}{\frac{q0(D_i+d_i, Seq) \vdash q0 \quad Ticket(D_i+d_i+d_5, Seq) \vdash Ticket}{q0(D_i+d_i, Seq), Ticket(D_i+d_i+d_5, Seq) \vdash q0 \otimes Ticket} \otimes_R \quad p2(D_i+d_i+d_5, \cdot), q1(Seq, \cdot), q1 \otimes p2 \multimap VISA \otimes q2, t_7, t_o \vdash o} \multimap_L \\
\frac{q0(D_i+d_i, \cdot), Ticket(D_i+d_i+d_5, \cdot), p2(D_i+d_i+d_5, \cdot), q0 \otimes Ticket \multimap q1, t_2, t_7, t_o \vdash o}{p0(D_i+d_i, D_i+d_i+d_5) \vdash p0 \quad q0(D_i+d_i, \cdot), Ticket(D_i+d_i+d_5, \cdot) \otimes p2(D_i+d_i+d_5, \cdot), t_1, t_2, t_7, t_o \vdash o} \otimes_L \\
\frac{q0(D_i+d_i, \cdot), p0(D_i+d_i, \cdot), t_1, t_2, p0 \multimap Ticket \otimes p2, t_7, t_o \vdash o}{i(D_i, D_i+d_i) \vdash i \quad q0(D_i+d_i, \cdot) \otimes p0(D_i+d_i, \cdot), t_1, t_2, t_5, t_7, t_o \vdash o} \multimap_L \\
i(D_i, \cdot), i \multimap q0 \otimes p0, t_1, t_2, t_5, t_7, t_o \vdash o
\end{array}$$

Tabela 15 – Intervalos de datas simbólicas de execução para as tarefas do cenário SC_1 .

Átomo	Antes da regra de sincronização	Depois da regra de sincronização
t_i	$[D_i,$ $D_i + d_{imax} + d_{4max} + d_{1max}]$	$[D_i,$ $D_i + d_{imax} + d_{4max} + d_{1max}]$
t_4	$[D_i + d_{imin},$ $D_i + d_{imax} + d_{4max} + d_{1max} + d_{3max} + d_{6max}]$	$[D_i + d_{imin},$ $\max\{D_i + d_{imax} + d_{4max}, D_i + d_{imax} + d_{4max} + d_{1max}\} + d_{3max}]$
t_1	$[D_i + d_{imin} + d_{4min},$ $D_i + d_{imax} + d_{4max} + d_{1max} + d_{3max}]$	$[D_i + d_{imin} + d_{4min},$ $\max\{D_i + d_{imax} + d_{4max}, D_i + d_{imax} + d_{4max} + d_{1max}\} + d_{3max}]$
t_3	$[D_i + d_{imin} + d_{4min} + d_{1min},$ $D_i + d_{imax} + d_{4max} + d_{1max} + d_{3max} + d_{6max} + d_{omax}]$	$[D_i + d_{imin} + d_{4min} + d_{1min},$ $\max\{D_i + d_{imax} + d_{4max}, D_i + d_{imax} + d_{4max} + d_{1max}\} + d_{3max} + d_{6max} + d_{omax}]$
t_6	$[D_i + d_{imin} + d_{4min} + d_{1min} + d_{3min},$ $D_i + d_{imax} + d_{4max} + d_{1max} + d_{3max} + d_{6max} + d_{omax}]$	$[\max\{D_i + d_{imin} + d_{4min}, D_i + d_{imin} + d_{4min}\} + d_{3min},$ $\max\{D_i + d_{imax} + d_{4max}, D_i + d_{imax} + d_{4max} + d_{1max}\} + d_{3max} + d_{6max} + d_{omax}]$

Tabela 16 – Intervalos de datas simbólicas de execução para as tarefas do cenário SC_2 .

Átomo	Antes da regra de sincronização	Depois da regra de sincronização
t_i	$[D_i,$ $D_i + d_{imax} + d_{5max} + d_{1max}]$	$[D_i,$ $D_i + d_{imax} + d_{5max} + d_{1max}]$
t_5	$[D_i + d_{imin},$ $D_i + d_{imax} + d_{5max} + d_{1max} + d_{2max} + d_{7max}]$	$[D_i + d_{imin},$ $\max\{D_i + d_{imax} + d_{5max}, D_i + d_{imax} + d_{5max} + d_{1max}\} + d_{2max}]$
t_1	$[D_i + d_{imin} + d_{5min},$ $D_i + d_{imax} + d_{5max} + d_{1max} + d_{2max}]$	$[D_i + d_{imin} + d_{5min},$ $\max\{D_i + d_{imax} + d_{5max}, D_i + d_{imax} + d_{5max} + d_{1max}\} + d_{2max}]$
t_2	$[D_i + d_{imin} + d_{5min} + d_{1min},$ $D_i + d_{imax} + d_{5max} + d_{1max} + d_{2max} + d_{7max} + d_{omax}]$	$[D_i + d_{imin} + d_{5min} + d_{1min},$ $\max\{D_i + d_{imax} + d_{5max}, D_i + d_{imax} + d_{5max} + d_{1max}\} + d_{2max} + d_{7max} + d_{omax}]$
t_7	$[D_i + d_{imin} + d_{5min} + d_{1min} + d_{2min},$ $D_i + d_{imax} + d_{5max} + d_{1max} + d_{2max} + d_{7max} + d_{omax}]$	$[\max\{D_i + d_{imin} + d_{5min}, D_i + d_{imin} + d_{5min}\} + d_{2min},$ $\max\{D_i + d_{imax} + d_{5max}, D_i + d_{imax} + d_{5max} + d_{1max}\} + d_{2max} + d_{7max} + d_{omax}]$