VERIFICAÇÃO DO NÍVEL DE ENLACE DO PROTOCOLO X–25

Célio C. Guimarães
Edmundo R. M. Madeira

RELATÓRIO TÉCNICO Nº 08/88

SUMÁRIO. Neste trabalho analisamos o nível 2 do protocolo de comunicações X–25 definido pela recomendação do CCITT. Este trabalho fez parte de uma implementação realizada e testada num microprocessador 8088 da Intel na Unicamp. Para esta análise foi utilizada a técnica de P. Zafiropulo (6) para deteção de erros em protocolos, tais como: deadlocks, recepções não especificadas, interações não executáveis e ambiguidades de estados.

ABSTRACT. This work presents a formal analysys of the CCITT X–25 level 2 protocol using a technique introduced by P. Zafiropulo (6) for error detection in protocols such as deadlocks, unspecified receptions, non executable interactions and state ambiguity. It has been part of a real implementation using an Intel 8088 microcomputer at Unicamp.

Universidade Estadual de Campinas
Instituto de Matemática, Estatística e Ciência da Computação
IMECC – UNICAMP
Caixa Postal 6065
13.081 – Campinas – SP
BRASIL

Maio – 1988

VERIFICAÇÃO DO NÍVEL DE ENLACE DO PROTOCOLO X-25

Célio C. Guimarães (.)

Edmundo R. M. Madeira (..)

Departamento de Ciência da Computação - IMECC - UNICAMP

SUMÁRIO

Neste trabalho analisamos o nível 2 do protocolo de comunicações X-25 definido pela Recomendação do CCITT. Este trabalho fez parte de uma implementação realizada e testada num microprocessador 8088 da Intel na Unicamp. Para esta análise foi utilizada a técnica de P. Zafiropulo [6] para deteção de erros em protocolos, tais como: deadlocks, recepções não especificadas, interações não executáveis e ambiguidades de estados.

ABSTRACT

This work presents a formal analysys of the CCITT X-25 level 2 protocol using a technique introduced by P. Zafiropulo [6] for error detection in protocols such as deadlocks, unspecified receptions, non executable interactions and state ambiguity. It has been part of a real implementation using an Intel 8088 microcomputer at Unicamp.

1. ÁRVORE DE PERTUBAÇÃO

Alguns trabalhos [5,1] comentam que os protocolos existentes (CCITT X-21 e X-25, SNA da IBM) são corretos à nível funcional e de semântica, cumprindo objetivamente os propósitos aos aos quais foram criados. Porém, estes protocolos não tem previsto todas as possibilidades sintáticas possíveis, podendo ser incompletos ou logicamente inconsistentes.

Um exemplo deste fato é a colisão de um pacote DCE-CLEAR INDICATION proveniente da rede com o pácote DTE-CALL-RESQUEST proveniente de um DTE no nível 3 do protocolo X-25. De acordo com a especificação, esta colisão seria identificada pela rede

---

(.) PhD em Ciência da Computação (Case Western Reserve University - 73); Sistemas Operacionais, Sistemas Distribuídos.

(..) Mestre em Ciência da Computação (Unicamp-85); Redes de Computadores, Sistemas Operacionais.

como um "local procedure error", apesar dessa colisão ser permitida pela especificação. Portanto , "local procedure error" torna-se ambíguo, pois está sendo utilizado para identificar colisões naturais e também para identificar violações reais do protocolo. Este exemplo é analisado também em [6] por P. Zafiropulo.

Isto nos leva a especificar e analisar formalmente este protocolo de comunicação. Para especificar o protocolo usamos diagramas de estado . Para analisar o protocolo construímos uma "árvore de perturbação". A árvore de perturbação simula a execução do protocolo, analisando todas as transições possíveis. Essa árvore é sugerida por P. Zafiropulo em [6] e detecta situações de erros no protocolo.

Para comunicação entre duas máquinas ditas computador 1 e computador 2, os nós da árvore possuem o seguinte formato:

EST1	CAN1
CAN2	EST2

onde: EST1: é o estado do computador 1.
EST2: é o estado do computador 2.
CAN1: é o canal que contém a seqüência de mensagens enviadas pelo computador 1 e ainda não recebidas pelo computador 2.
CAN2: é o canal que contém a seqüência de mensagens enviadas pelo computador 2 e ainda não recebidas pelo computador 1.

A árvore de perturbação é construída da seguinte maneira:

O nó inicial (ou estado inicial do sistema) possui os dois canais vazios e os estados iniciais de cada computador.

Dado um nó, analisamos as próximas transições possíveis, ou seja, as transmissões a partir dos estados das duas partes e as recepções a partir do conteúdo dos dois canais. Quantas transições forem possíveis, quantos filhos este nó terá. Para cada filho, os estados e os canais serão obtidos em função do pai e da transição.

Se a transição é de recepção, a mensagem é retirada do canal e o estado do receptor é alterado (em função do diagrama de estados que define o protocolo).

Se a transição é de transmissão, a nova mensagem é inserida no canal correspondente (em função de qual computador enviou a mensagem) e o estado do transmissor também é alterado (de acordo com o diagrama de estados).

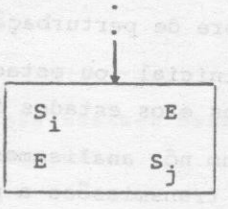
A árvore de perturbação vai sendo construída até se obter em suas folhas estados do sistema já analisados.

Esta técnica para a análise de protocolos pode ser aplicada para outras áreas de processos cooperantes. O argumento utilizado para demonstrar este fato é de Merlin em [4] : "Dado um sistema de processos cooperantes tal que a cooperação é feita através de troca de mensagens, um protocolo é um conjunto de regras que governa esta troca".

A partir da construção da árvore de perturbação podemos detectar quatro erros em potencial de projeto: deadlock, recepção não especificada, interações não executáveis e ambiguidade de estados.

Deadlock ocorre quando os dois computadores permanecem indefinidamente no mesmo estado em virtude de não terem o que computar. Na árvore de perturbação isto pode ser identificado pela existência de um nó que possui ambos os canais vazios (nenhuma transição de recepção a realizar) e nenhuma transição de transmissão possível a realizar.

EX.: O computador 1 em S_i e o computador 2 em S_j não podem enviar mensagens, ou seja na árvore de perturbação :



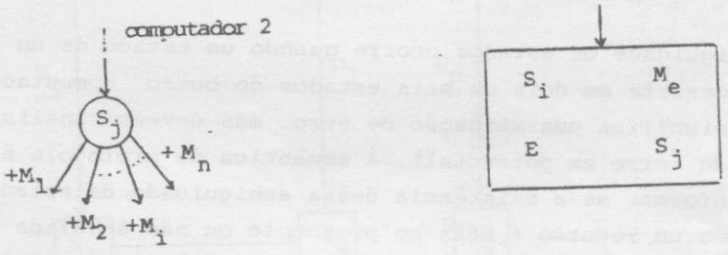
No diagrama de estados:

Computador 1 Computador 2

onde $\xrightarrow{+}$ significa recepção de mensagem.

Recepção não especificada ocorre quando um dos computadores recebe uma mensagem M, num determinado estado S_i, e que no diagrama de blocos não possui transição capaz de absorver esta mensagem M. Na árvore de perturbação isto pode ser identificado pela existência de um nó que não possua transições (eventos) para absorver mensagens que estavam em um dos dois canais.

EX.: no diagrama de estados: na árvore de perturbação :



onde $\forall i , i \neq e$

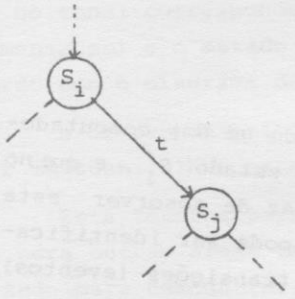
Interação não executável ocorre quando um evento de transmissão ou recepção nunca acontece em função de como foi projetado o protocolo (diagrama de estados). É análogo ao código não atingível em programas.

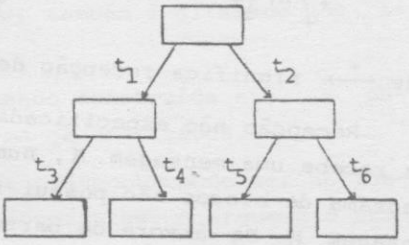
Este fato não é um erro, se o projetista colocou este evento apenas para ser executado em condições anormais (por exemplo, para se recuperar de uma condição de erro).

Identificamos uma interação não executável, pela não exis

tência de uma transição (transmissão ou recepção) do diagrama de estados na árvore de perturbação. A árvore de perturbação "cobre" todas as possibilidades de eventos).

EX.: No diagrama de estados: na árvore de perturbação:



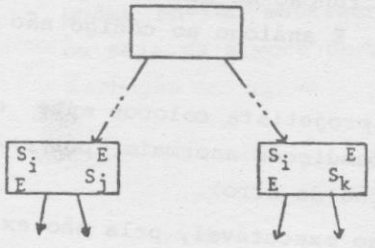


onde $t \neq t_i$ $\quad i = 1,\ldots,6$

Ambiguidade de estados ocorre quando um estado de um computador coexiste em dois ou mais estados do outro computador . Isto não significa uma situação de erro, mas devemos analisá-la com cuidado (erro em potencial). A semântica do protocolo é que nos vai informar se a existência dessa ambiguidade de estados é válida como um recurso a mais no protocolo ou não é válida por se tratar de um erro. Na árvore de perturbação essa ambiguidade é identificada pela existência de dois ou mais nós do sistema que possuam os dois canais vazios e um estado comum na diagonal.

4. Ambiguidade de estados:

EX.: Na árvore de perturbação:



onde $j \neq k$

Construímos a seguir a árvore de perturbação para o protocolo do bit alternado como um exemplo. (figura 2).

O diagrama de estados deste protocolo é apresentado na figura 1, onde M significa transmisão de M, M significa recepção de M, D é uma mensagem de dados (D_o c/bit zero e D_1 com bit um) e A é uma mensagem de confirmação (A_o da mensagem de bit zero e A_1 da mensagem de bit 1).

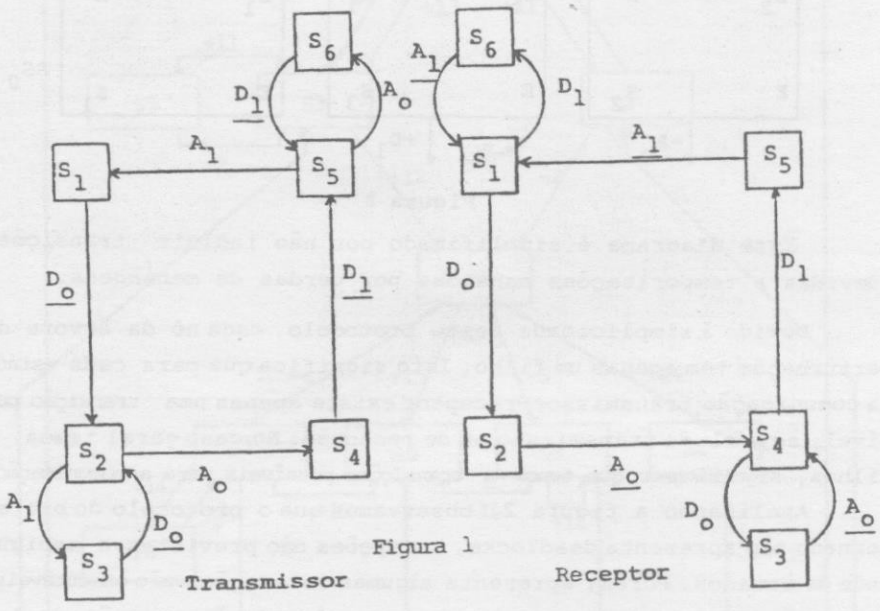


Figura 1

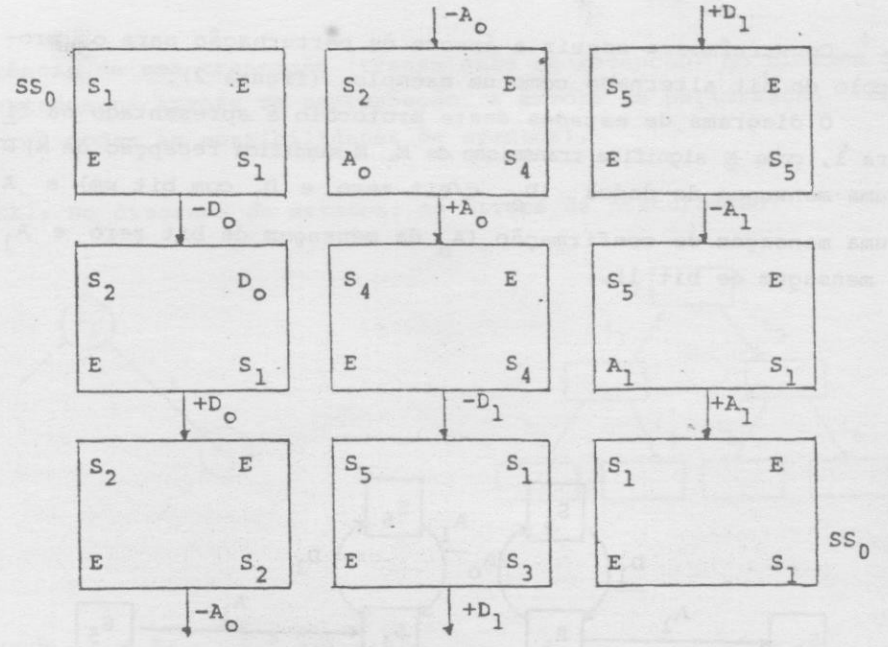


Figura 2

Este diagrama é simplificado por não incluir transições devidas a temporizações cansadas por perdas de mensagens.

Devido à simplicidade deste protocolo, cada nó da árvore de perturbação tem apenas um filho. Isto significa que para cada estado da comunicação transmissor-receptor existe apenas uma transição possível, seja ela de transmissão ou de recepção. No caso geral temos n filhos, significando que temos n transições possíveis para a comunicação.

Analisando a figura 2, observamos que o protocolo do bit alternado não apresenta deadlocks, recepções não previstas e ambiguidade de estados. Porém, apresenta algumas interações não executáveis: transição de S_2 para S_3 no transmissor devido à recepção de A_1, transição de S_1 para S_6 no receptor devido à recepção de D_1, etc.

Isto não é um erro no protocolo, pois essas transições foram projetadas para serem realizadas apenas em condições de erro, como por exemplo, troca não detetada do conteúdo de mensagem: o receptor no estado S_2 envia o quadro A_o que é recebido pelo transmissor como um quadro A_1. Observe que neste caso tanto o transmissor quanto o receptor entram no estado S_3 se recuperando posteriormente.

2. ANÁLISE DO NÍVEL DE ENLACE (NÍVEL 2) DO PROTOCOLO X-25

O diagrama de estado construído para este nível de acordo com a especificação do CCITT [7] é o seguinte:

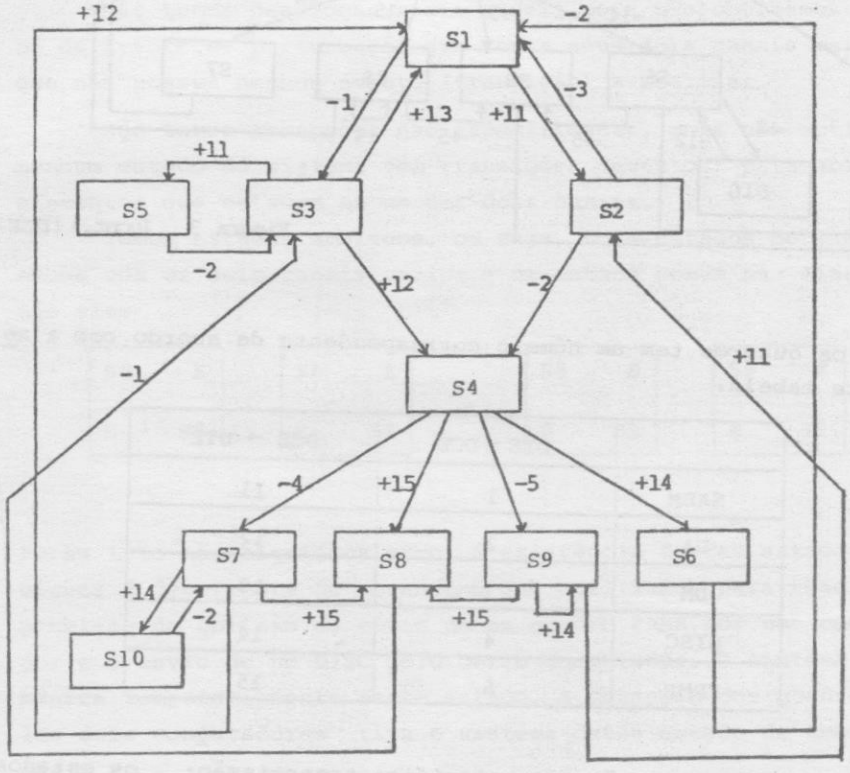


Figura 3 (DTE)

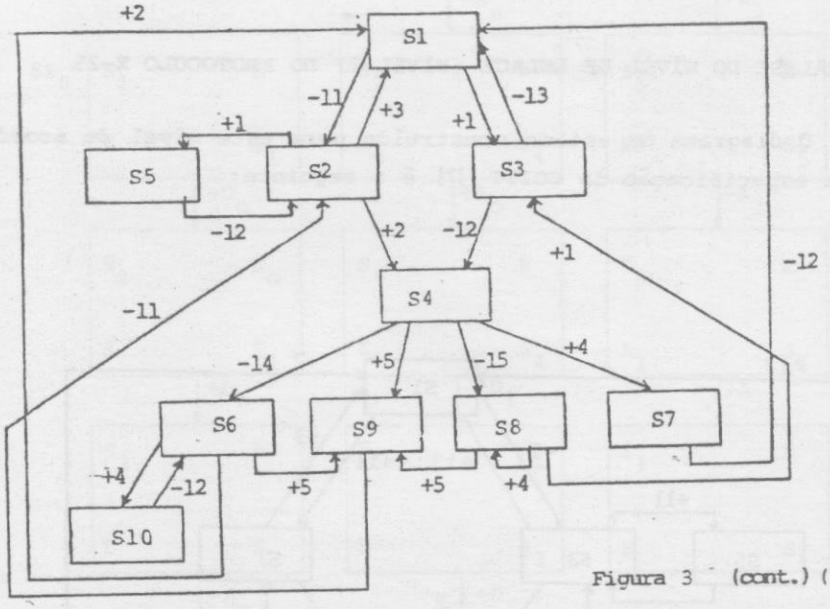


Figura 3 (cont.) (DCE)

onde os quadros tem um número correspondente de acordo com a seguinte tabela:

	DTE → DCE	DCE → DTE
SABM	1	11
UA	2	12
DM	3	13
DISC	4	14
FRMR	5	15

+ significa recepção, - significa transmissão; os estados são os seguintes:

S1 - Desconectado
S2 - DCE esperando por conexão
S3 - DTE esperando por conexão
S5 - Colisão de pedido de conexão
S4 - Conectado - Transferência de informação

S6 - DCE esperando por desconexão
S7 - DTE esperando por desconexão
S10 - Colisão de pedido de desconexão
S8 - DCE - Condição de exceção
S9 - DTE - Condição de exceção.

A figura 4 mostra a árvore de perturbação construída para análise do protocolo. Analisando cuidadosamente a árvore pode-se observar:

Não temos deadlocks neste nível, pois não obtivemos nenhum nó da árvore de perturbação que tenha seus dois canais vazios e que não possua nenhum evento (transição) a realizar.

Não temos recepções não especificadas, pois não obtivemos nenhum estado do sistema sem transições (eventos) para absorver elementos que estavam em um dos dois canais.

Temos estados ambíguos, ou seja, dois estados do sistema, ambos com os dois canais vazios e um estado comum na diagonal. São eles

S8	E
E	S8

S8	E
E	S9

e

S8	E
E	S9

S9	E
E	S9

.

Porém isto não significa erro. A existência destes estados ambíguos é decorrente do mecanismo que utilizamos para resolver o problema da colisão do envio de um quadro FRMR por um computador e o envio de um DISC pelo outro computador. O sistema permanece temporariamente neste estado. A recepção dos quadros pelos dois computadores tira o sistema desse estado de ambiguidade.

Não temos interações não executadas, pois todas as transições do diagrama de estado aparecem na árvore de perturbação construída.

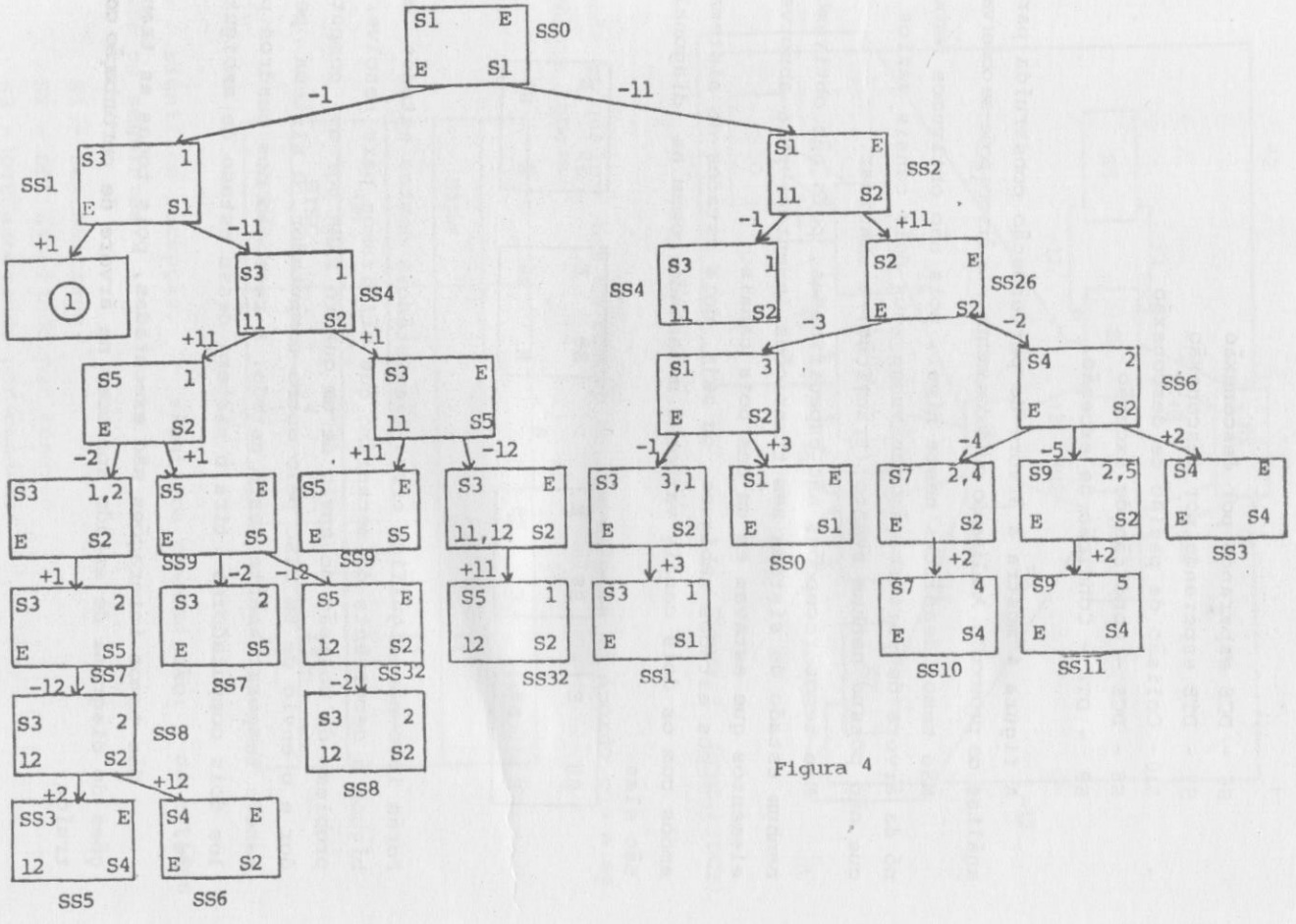

S1 E
E S1
SS0
-1
-11
S3 1
E S1
SS1
S1 E
11 S2
SS2
+1
-11
-1
+11
1
S3 1
11 S2
SS4
S3 1
11 S2
SS4
S2 E
E S2
SS26
+11
+1
-3
-2
S5 1
E S2
S3 E
11 S5
S1 3
E S2
S4 2
E S2
SS6
-2
+1
+11
-12
-1
+3
-4
-5
+2
S3 1,2
E S2
S5 E
E S5
SS9
S5 E
E S5
SS9
S3 E
11,12 S2
S3 3,1
E S2
S1 E
E S1
SS0
S7 2,4
E S2
S9 2,5
E S2
S4 E
E S4
SS3
+1
-2
-12
+11
+3
+2
+2
S3 2
E S5
SS7
S3 2
E S5
SS7
S5 E
12 S2
SS32
S5 1
12 S2
SS32
S3 1
E S1
SS1
S7 4
E S4
SS10
S9 5
E S4
SS11
-12
-2
S3 2
12 S2
SS8
S3 2
12 S2
SS8
+2
+12
SS3 E
12 S4
SS5
S4 E
E S2
SS6

Figura 4

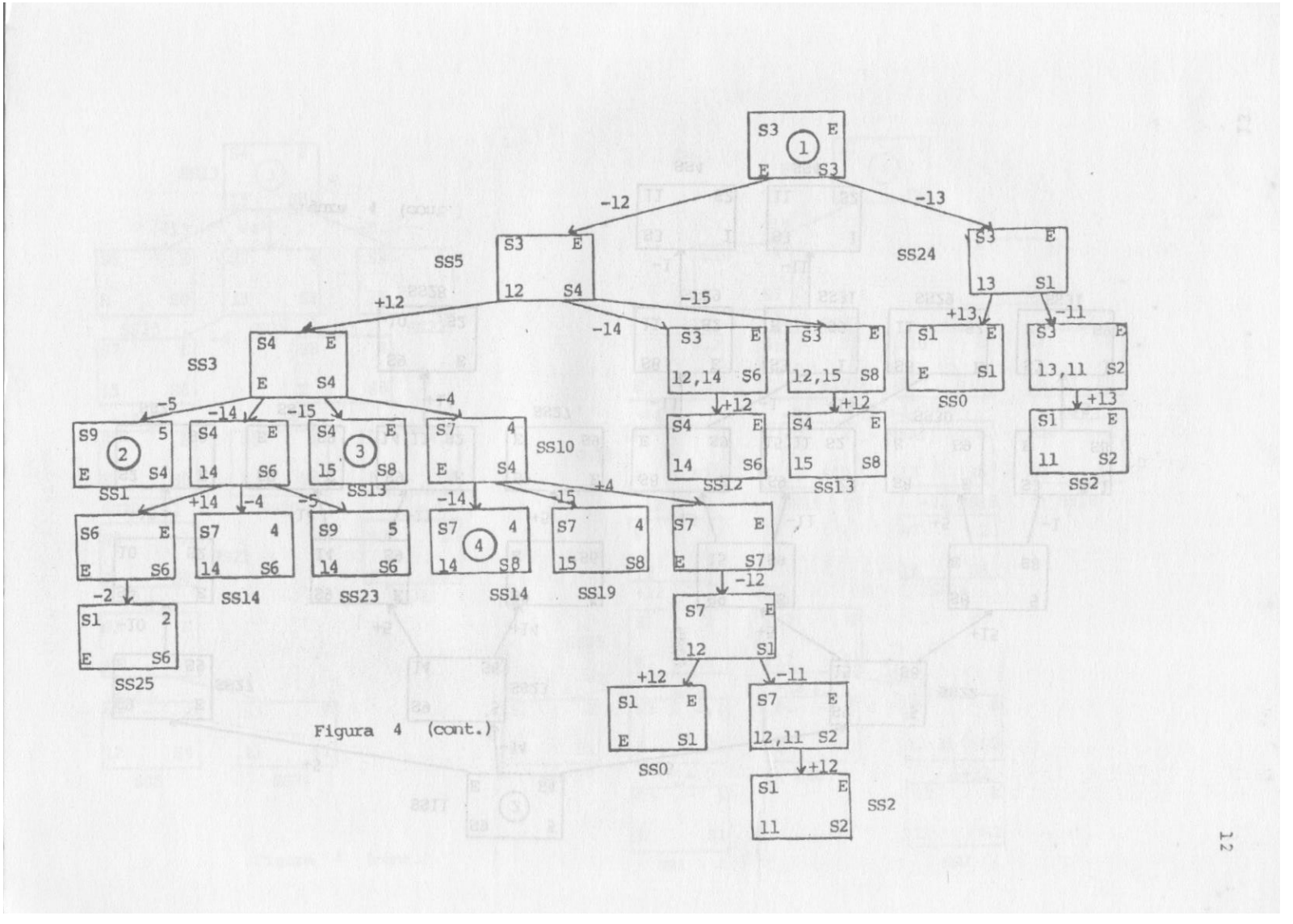


Figura 4 (cont.)

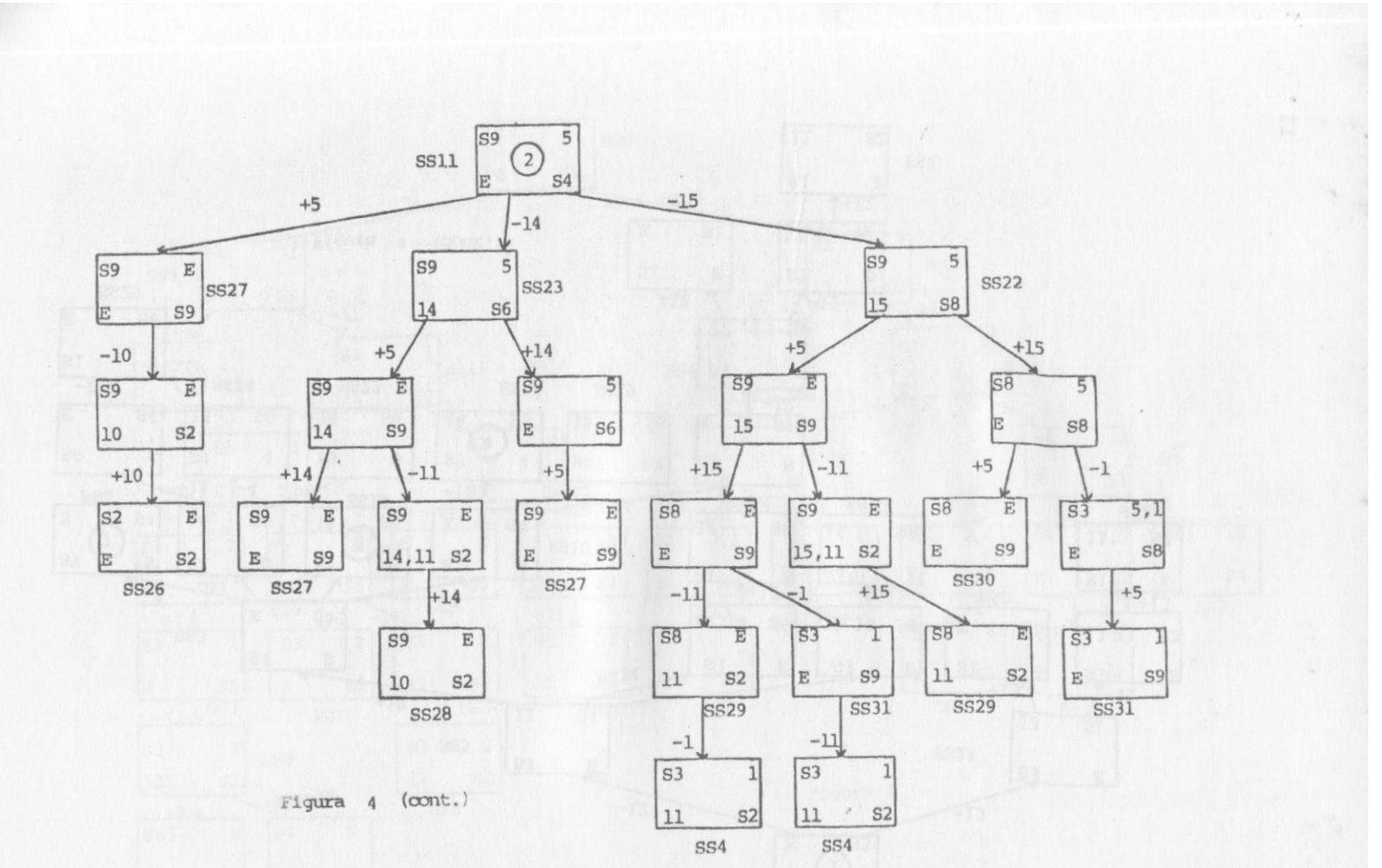


Figura 4 (cont.)

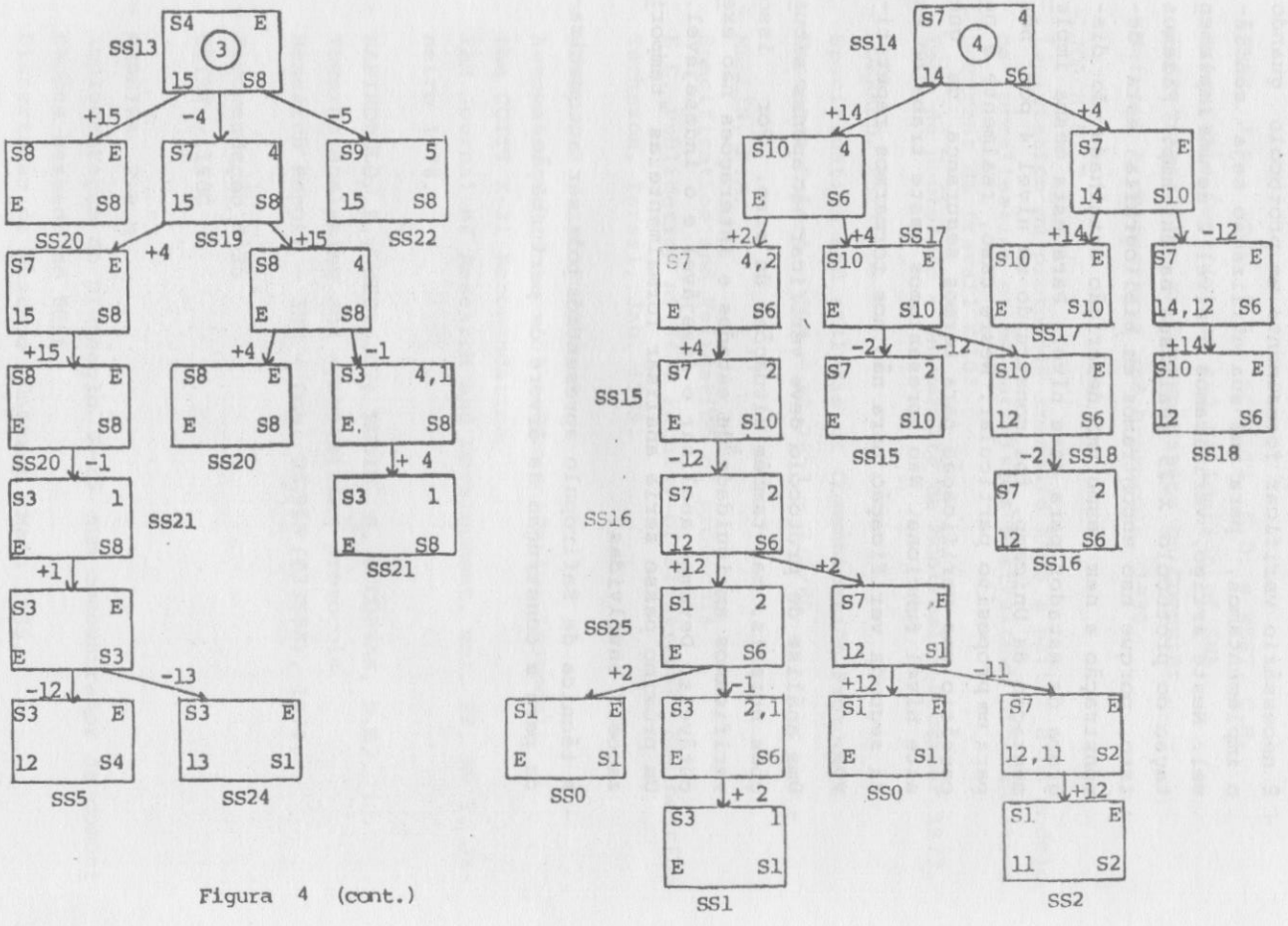


Figura 4 (cont.)

3. CONCLUSÕES

- É necessário verificar formalmente um protocolo quando o implementamos, para que sua utilização seja confiável. Neste artigo, verificamos o nível 2 de uma implementação do protocolo X-25 realizada na Unicamp. Fizemos isto, porque não encontramos em bibliografia esta demonstração e nem mesmo uma descrição detalhada do diagrama de estados para este nível. Para esta mesma implementação da Unicamp, foi construído um nível 4 por nós para um proposito particular. Neste caso, realmente é necessário uma verificação para termos segurança de que este nível funciona. Não apresentamos neste trabalho esta segunda verificação para não nos tormarmos repetitivos.
- Uma análise de protocolo deve verificar não apenas situações normais, mas também situações de erro. Por isso verificamos ambiguidade de estados e interações não executáveis. Devemos analisar o desejável e o indesejável. Um próximo passo seria analisar formalmente as "temporizações" envolvidas.
- A técnica de Zafiropulo apresentada pode ser automatizada para a construção da árvore de perturbação.

BIBLIOGRAFIA:

1 - GRAY, J.P.; ROSE, D.B.; SCHULTZ, G.D. e WEST, C.H.
Executable Description and Validation of SNA
IEEE Transaction on Communication on Computer
Network Architectures and Protocols, 1980.

2 - JOACHIM, T.
Implatation du protocole standard X-25 a partir d'un modele de formalisation et de mecanismes abstraits de programmation
Document de travail # 103
Tese de doutorado - Universite de Montreal - December 1977.

3 - JORRAND, Philippe
Specification and analysis of Communication Protocols
Research Report - IBM - RJ 2853 - 7/2/80.

4 - MERLIN, P.M.
Specification and Validation of protocols
E.E. Publication nº 343, Faculty of Electrical Emgineering
Technion, Israel, Jan. 1979.

5 - WEST, C.H.; ZAFIROPULO, P.
Automated Validation of a Communication Protocol:
the CCITT X-21 Recomendation
IBM Journal of Research and Development, vol. 22, nº 1, Janeiro 1978.

6 - ZAFIROPULO, P.; WEST, C.; RUDIN, H. e CORVAN, D.D.
Towards analysing and synthesizing protocols
Research Report - IBM - Com. RZ 963 (#33588), 1979.

7 - Recomendação X-25
CCITT - 1980

8 - MADEIRA, E.R.M.
Implementação do protocolo X-25 num concentrador de comunicações baseado no 8088.
Dissertação de Mestado - Unicamp - Maio 1985.

RELATÓRIOS TÉCNICOS — 1988

01/88 – A Linear Continuous Transportation Problem – *Enrique D. Andjel, Tarcisio L. Lopes* and *José Mario Martínez.*

02/88 – A Splitting Theorem for Complete Manifolds With Non-Negative Curvature Operator – *Maria Helena Noronha.*

03/88 – Mathematical Physics of the Generalized Monopole without String – *W. A. Rodrigues Jr., M. A. Faria-Rosa, A. Maia Jr.* and *E. Recami.*

04/88 – A Family of Quasi-Newton Methods with Direct Secant Updates of Matrix Factorizations – *José Mario Martínez.*

05/88 – Rotation Numbers of Differential Equations. A Framework in the Linear Case – *Luis San Martin.*

06/88 – A Geometrical Theory of Non Topological Magnetic Monopoles – *Marcio A. Faria-Rosa* and *Waldyr A. Rodrigues Jr.*

07/88 – Cosmic Walls and Axially Symmetric Sigma Models – *Patricio S. Letelier* and *Enric Verdaguer.*