INTERFACE PARA USUÁRIO CASUAL - IUC
LINGUAGEM DE COMUNICAÇÃO PARA USUÁRIO CASUAL DE BANCO DE DADOS, BASEADA EM LINGUAGEM NATURAL

Antônio Cláudio Gomez de Sousa

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO EM ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.).

Aprovada por:

Sueli Mendes dos Santos
Presidente

José Lucas Mourão Rangel Netto

Antônio Cláudio de Carvalho M. da Silva

RIO DE JANEIRO, RJ - BRASIL

MARÇO DE 1983

SOUSA, ANTÔNIO CLÁUDIO GOMES DE

Interface para Usuário Casual - IUC - Linguagem de Comunicação para Usuário Casual de Banco de Dados, Baseada em Linguagem Natural |Rio de Janeiro| 1983.

VIII, 168p. 29,7 cm (COPPE-UFRJ, M.Sc., Engenharia de Sistemas e Computação, 1983)

Tese - Universidade Federal do Rio de Janeiro, COPPE.

1. Linguística Computacional. I. COPPE/UFRJ II. Título (série).

RESUMO

Este trabalho apresenta a especificação funcional e a especificação para implementação de uma interface em linguagem natural, que permite a comunicação entre um "usuário casual" e um Sistema de Banco de Dados.

A linguagem da interface IUC é um subconjunto do Portugues, e a linguagem do Sistema de Banco de Dados é a Linguagem de Operação de Banco de Dados - LOBAN.

A interface IUC permite perguntas ao Sistema de Banco de Dados, que responde então com informações não ambíguas na forma de tabela.

A tradução de IUC para LOBAN é realizada por uma Rede de Transição Aumentada, que utiliza a semântica da base de dados.

ABSTRACT

Presented here are the functional specification and the specification for implementation of a natural language interface which allows communication between a "casual" user and a Database system.

Interface language IUC is a subset of Portuguese; the Database system is accessed through the LOBAN Database Operation Language.

The IUC user is able to place queries to the Database system, which then answers with non-ambiguous information in table form.

The translation from IUC to LOBAN is described by means of an Augmented Transition Network, and follows the semantics of the Database.

ÍNDICE

I. INTRODUÇÃO

I.1. APRESENTAÇÃO

Este trabalho faz a especificação funcional de uma interface para a comunicação de um "usuário casual" com um Sistema de Banco de Dados (SBD), assim como a especificação para a implementação dessa interface.

A interface a ser especificada é a "Interface para um Usuário Casual - IUC". Ela é um subconjunto restrito do Português, portanto uma interface em "linguagem natural".

Consideramos neste trabalho como "usuário casual" todo usuário de Banco de Dados que necessariamente deve conhecer a natureza das informações armazenadas no Banco de Dados que ele deseja acessar, mas que não necessita ter conhecimentos em computação em geral, ou especificamente no Banco de Dados em questão. O usuário casual não necessita saber como as informações estão armazenadas no Banco de Dados, nem como acessá-las. Ele necessita saber quais informações estão no Banco de Dados, e ter uma noção da estrutura dessas informações.

A linguagem do SBD para a qual estamos realizando a tradução é a LOBAN. Esta linguagem é a especificação de uma interface para SBD desenvolvida na COPPE/UFRJ e na UFRGS, em convênio com a GMD/Alemanha, através do CNPq.

Está em desenvolvimento na COPPE o "Projeto MICRO-LOBAN" que trata de uma implementação de LOBAN em um microcomputador nacional (COBRA-305). Inicialmente o projeto era para

ser implementado em computador nacional de pequeno porte, com a finalidade de tornar-se "software" de computador nacional de amplo uso. Com as dificuldades encontradas para dispor de um equipamento desses, condição necessária ao desenvolvimento do SBD, optou-se por alterar o projeto para um microprocessador nacional de amplo uso. O trabalho da tese seria utilizado com o SBD desenvolvido na COPPE. Com a mudança no projeto tornou-se impossível esse uso imediato, pois uma interface como a IUC exige recursos não disponíveis em microprocessadores. Como consequência o trabalho da tese passou a ser de especificação, para ser implementado no futuro em outro projeto para LOBAN, mas em computador de pequeno porte.

A interface IUC será sobreposta a LOBAN, tem portanto, segundo LOKEMANN[29], hierarquia superior a LOBAN. Ela é um nível de linguagem mais próximo do usuário.

Na PUC/RJ, FARIAS[15], ROSA[38] e ZIVIANI[52] realizaram interfaces similares à IUC. Na UNICAMP/SP, FRAGA[17,18,19] realizou vários trabalhos em análise morfológica do Português, e tradução do Português para o Inglês. Dentro do possível utilizamos a terminologia em Português adotada nesses trabalhos, assim como os resultados obtidos.

I.2. A NATUREZA DO PROBLEMA

Os Sistemas de Banco de Dados são uma ferramenta poderosa para a manipulação de informações. No entanto para ter acesso às informações armazenadas em um Banco de Dados, um usuário deve ter conhecimentos gerais em programação, e específicos na linguagem de operação do Banco de Dados. A maioria dos usuários não tem esses conhecimentos, e fica na dependência de especialistas para, através destes, ter acesso às informações desejadas. Este problema limita o uso dos Bancos de Dados, e tende a se agravar na medida em que os Bancos de Dados são cada vez mais utilizados em sistemas de informações.

Há várias linhas de trabalho para a solução deste problema, todas com suas vantagens e desvantagens. As mais significativas são os sistemas gráficos, os sistemas com "menus", os "formulários", os sistemas "query by example", e a linguística computacional.

À exceção da linguística computacional, os demais sistemas, de uma maneira geral, tem como vantagem a facilidade relativa para implementar e usar. Como desvantagem todos tem dificuldade em permitir perguntas mais complexas, ou perguntas simples mas que exigem "navegação" em mais de uma tabela (ou arquivo) para a obtenção da resposta.

A linguística computacional tem como desvantagem a complexidade de implementação, mas por outro lado permite perguntas complexas, perguntas com referência ao contexto, perguntas com referências a outras perguntas, e perguntas com "embutimento" de condições limitado apenas pelo desempenho do usuá

rio.

Nossa opção pela linguística computacional derivou de suas vantagens. Procuramos amenizar suas desvantagens limitando o subconjunto de Português, mas de maneira a manter a interface IUC como uma interface em linguagem natural. Este aspecto é fundamental, pois para que uma linguagem seja natural, não basta que as palavras utilizadas pertençam ao léxico de uma linguagem natural. É necessário que sua gramática seja um subconjunto da gramática da linguagem natural, e de fácil aprendizado pelo usuário casual. De outra forma não estaríamos criando nenhuma facilidade para o usuário.

I.3. O OBJETIVO DO TRABALHO

Mais adiante será visto um problema básico dos sistemas baseados em linguística computacional e utilizados em Banco de Dados: sua dependência ao universo do discurso da aplicação. Este problema não permite que um sistema desenvolvido para uma aplicação de um Banco de Dados, seja utilizado em outra aplicação, quando muda o universo do discurso. Assim uma interface desenvolvida para uma aplicação em controle acadêmica, não poderá ser utilizada, por exemplo, em uma aplicação para controle de estoque.

Neste trabalho tivemos como objetivo especificar uma interface independente do universo do discurso, portanto de uso geral em Banco de Dados. Isto foi obtido impondo certas limitações à interface, e obtendo as informações "semânticas" a

partir da definição da "coerência" da Base de Dados, e a partir do léxico (que é o elemento variável e particular para cada aplicação).

I.4. O ESCOPO DO TRABALHO

Este trabalho pretende ser em primeiro lugar uma especificação funcional da interface IUC. Esta primeira parte, apresentada no capítulo IV da tese, especifica a linguagem aceita pela interface IUC, através da definição da gramática dessa linguagem. Nesta parte são expostas além da gramática, que define a comunicação homem-IUC, os tipos de resposta, que definem a comunicação IUC-homem, as limitações impostas e os objetivos dessas limitações.

A segunda parte deste trabalho trata da especificação para a implementação da interface IUC. Como não está definido o ambiente onde será posta a interface IUC, esta especificação é independente de máquina ou programação. Ela portanto apresenta os algoritmos e estruturas de dados, mas não trata da otimização dos recursos exigidos para esses algoritmos e estruturas de dados. Esta segunda parte é desenvolvida nos capítulos V a IX da tese.

Para facilitar o entendimento das especificações, no capítulo II fazemos uma apresentação da linguística computacional, orientada para a presente aplicação. É portanto uma apresentação parcial. Com o mesmo objetivo, e mesma limitação, no capítulo III fazemos uma apresentação da interface LOBAN.

Os exemplos de todos capítulos são tirados de uma possível aplicação em controle acadêmico. A Base de Dados dessa aplicação está representada no Apêndice 1.

II. A LINGUÍSTICA COMPUTACIONAL

II.1. INTRODUÇÃO

A Linguística Computacional estuda a comunicação entre o homem e o computador, em linguagem natural. Seu desenvolvimento data da década de 40, quando começaram os primeiros esforços para a tradução automática.

Desde o início no entanto um problema surgiu para os especialistas: como reconhecer ou gerar uma linguagem natural. Isto porque as linguagens naturais apresentam irregularidades que dificultam o projeto de algoritmos para esse fim.

As tentativas com grafos de estado finito não deram resultado, porque as linguagens naturais podem ter constituintes embutidos dentro de outros. Isto exige que durante a computação ela seja interrompida, e iniciada outra que analise aquele componente embutido, para depois voltar à computação anterior. Os grafos de estado finito reconhecem apenas linguagens regulares, e as linguagens naturais não são regulares.

A utilização dos grafos recursivos de estado finito resolveu o problema acima, mas não resolveu outra característica das linguagens naturais: a dependência contextual. Os constituintes podem ser modificados pelo contexto. Um exemplo disto é a concordância, onde a forma de um constituinte depende de outro. Os grafos recursivos de estado finito reconhecem linguagens livres de contexto, e as linguagens naturais não são livres de contexto.

II.2. A GRAMÁTICA TRANSFORMACIONAL

Em 1975 em seu livro "Syntactic Structures", Chomsky desenvolveu a teoria da Gramática Transformacional. Segundo esta teoria, a gramática é composta por três componentes: um léxico, uma gramática de "estrutura profunda" livre de contexto, e "regras de transformações" que passam as frases da "estrutura superficial" para a "estrutura profunda" através de "transformações".

Além de resolver os problemas das irregularidades das linguagens naturais, a gramática transformacional apresentou outra vantagem: permitiu identificar frases com mesmo sentido, apesar de terem estruturas superficiais distintas. O exemplo clássico é o caso das frases ativas e passivas. Por exemplo, as frases "João comeu a maçã", e "A maçã foi comida por João" tem estruturas superficiais distintas, apesar de terem o mesmo significado. Na estrutura profunda da teoria transformacional ambas tem a mesma estrutura, refletindo o fato de terem o mesmo sentido. A frase passiva é transformada, e fica com a mesma estrutura da frase ativa.

A Gramática Transformacional é até hoje o melhor modelo para descrever a competência de um falante de uma língua natural. Infelizmente porém não é adequada como algoritmo para o reconhecimento de linguagens naturais, pois não há como selecionar quais transformações aplicar, e em que ordem, a partir da estrutura superficial das frases. Há tentativas neste sentido, como em PLATH[35] e mais recentemente em GUIDA[23], mas nenhuma resolveu o problema acima citado.

II.3. A REDE DE TRANSIÇÃO AUMENTADA

Em 1970 WOODS[51] desenvolveu um analisador sintático, que continua sendo até hoje o analisador mais eficiente para linguagens naturais. Trata-se do "Augmented Transition Network -ATN", ou "Rede de Transição Aumentada-RTA".

A RTA é um grafo recursivo de estado finito. Trata-se portando de uma rede de nós, com arestas dirigidas ligando os nós. Cada nó corresponde a um estado em uma máquina de estado finito, e cada aresta uma transição entre estados. Há um nó inicial e nós finais. Uma sentença pertence à língua definida pela RTA, se iniciando pelo estado inicial se chegar a um estado final com toda sentença "consumida" nas transições de estados.

Em cada aresta podemos ter:

a) Um símbolo, cuja ocorrência na sentença de entrada, causa uma transição para o estado no fim da aresta;

b) Um nome de estado, que causa um desvio para esse estado, após empilhar o estado no fim do arco. É um mecanismo recursivo, que permite continuar a computação interrompida, após a conclusão da computação de mais baixo nível;

c) Testes que permitem seguir ou não a aresta;

d) Ações que alteram o resultado da computação já realizada em qualquer nível, ou que alteram computações a serem realizadas.

Essas características da RTA dão a ela o poder de uma máquina de Turing.

II.4. A COMPONENTE SEMÂNTICA

A análise sintática nos informa se uma frase é gramatical ou não, e pode fornecer uma descrição estrutural da frase, mas não nos informa se a frase tem sentido ou não. Para isso é necessário um componente semântico, que trate do significado das frases.

Há muitos analisadores semânticos, alguns muito poderosos, mas todos com um problema crucial para aplicação em Banco de Dados: sua dependência do universo do discurso. O analisador semântico é realizado para um universo do discurso de determinada aplicação, e se ao mudar de aplicação também mudar o universo do discurso, será necessário refazer o analisador semântico.

Devido a este problema não vamos detalhar nenhum analisador semântico, e voltaremos a este ponto superficialmente no próximo ítem, ao analisar alguns sistemas existentes.

II.5. ALGUNS SISTEMAS PARA BANCO DE DADOS

Vamos discorrer brevemente sobre alguns sistemas desenvolvidos para Bancos de Dados, procurando salientar em cada um a contribuição mais importante obtida pelo sistema.

II.5.1. LSNLIS

Em 1971 WOODS[50], Kaplan e Nash-Webber desenvolveram o "Lunar Sciences Natural Language Information System-LSNLIS" ou LUNAR, com a finalidade de permitir o acesso fácil às informações geológicas sobre as rochas lunares trazidas pela Apolo 11.

O analisador sintático é uma RTA que mapeia as perguntas em Inglês para árvores. Essas árvores são traduzidas por um processador semântico para uma representação semântica da pergunta, na forma de expressões (programas). Essas expressões são então executadas diretamente pelo módulo de recuperação, obtendo-se assim a reposta desejada.

É um sistema dirigido para recuperar informações técnicas, e completamente dependente do universo do discurso. Apesar de ser um bom exemplo de utilização da RTA, tem deficiências no sistema heurístico de recuperação. Para responder a uma pergunta que questiona se todas as rochas tem determinado elemento, ele busca todas rochas, e verifica se em todas há esse elemento. Um sistema do porte desse poderia estar organizado de forma a responder algumas perguntas sem percorrer a base de dados, ou pelo menos sem percorrer toda base de dados. No exemplo bastaria uma informação geral dizendo que todas rochas tem esse elemento, ou uma série de elementos.

II.5.2. RENDEZVOUS

Este sistema foi desenvolvido por CODD[10] em 1974, para traduzir perguntas em Inglês para a linguagem Alpha de um Banco de Dados relacional. A tradução é feita através de transformações sucessivas, aplicadas conforme os tipos de estrutura das perguntas.

Apesar de ser uma tentativa de fazer um sistema independente do universo da aplicação, a técnica de tipos (estruturas) não apresentou bons resultados, pois ficou difícil mapear todos os tipos de perguntas, e fazer distinções entre perguntas do mesmo tipo mas de significados diferentes.

A grande contribuição desse sistema foi no "diálogo esclarecedor" para resolver ambiguidades, e na formalização das "sete etapas" para o desenvolvimento de sistemas similares.

II.5.3. TORUS

O projeto TORUS - Toronto Understanding Systems, foi desenvolvido em 1975 na Universidade de Toronto, Canadá, por MYLOPOULOS[30-31-32], BORGIDA[4], Cohen, Roussopoulos, Tsotsos, Wong e Sugar.

O analisador sintático do TORUS é uma RTA, e o analisador semântico utiliza uma base de conhecimento em rede semântica. As perguntas são analisadas pelo analisador sintático e transformadas em árvores, que por sua vez são aplicadas

pelo analisador semântico sobre a rede semântica para resolver ambiguidades e fazer inferências. No caso de dúvidas na análise semântica, TORUS produz um "diálogo esclarecedor". Dirimidas todas ambiguidades, TORUS produz uma pergunta ao SBD. Com a resposta obtida do SBD, um gerador de respostas gera a resposta em linguagem natural.

TORUS mantêm o contexto das perguntas, o que permite em uma pergunta fazer referências a perguntas anteriores, ou utilizar o contexto das perguntas anteriores.

É um dos sistemas mais bem sucedidos no uso de rede semântica, mas também mostra a complexidade que atingem esses sistemas. Para uma aplicação de controle acadêmico com 400 palavras no dicionário, a rede semântica chegou a 100 nós, e o sistema completo exigiu quase 1M bytes para ser executado.

Atualmente TORUS está sendo revisto em alguns aspectos que apresentam problemas: a representação do conhecimento, a comunicação com o SBD, e a manutenção do contexto.

É um sistema que não pretende ser geral, isto é, a base de conhecimento deve ser refeita quando muda o universo de discurso. Dentro desse objetivo, é um sistema que já apresentou bons resultados, e espera-se que apresente resultados melhores ainda com as alterações que estão em estudo.

II.5.4. SLN

O Subconjunto de Linguagem Natural - SLN foi desenvolvido por ZIVIANI[52] na PUC/RJ em 1976. Esse sistema define um subconjunto restrito do Português. As perguntas a um SBD relacional nesse subconjunto são analisadas por uma RTA (a mesma do projeto TORUS), que produz estruturas sintáticas em árvore. Essas estruturas são apresentadas a um módulo chamado GCR (gerador de expressões em cálculo relacional), que a partir delas produz uma pergunta ao SBD na linguagem deste (Alpha).

O SLN procurou ser geral, isto é, independende do universo do discurso. Para isso foi eliminado o nível semântico, às custas de limitações impostas no subconjunto do Português. Exatamente essas limitações são o principal problema da SLN, pois exigem que o usuário as estude para ter acesso ao SBD.

A presente tese é uma continuidade no trabalho de Ziviani, pois procuramos atingir o mesmo objetivo. Só que para diminuir as limitações impostas ao subconjunto da linguagem natural, utilizamos a semântica derivada da base de dados.

II.5.5. PLANES

O "Programmed Language-based Enquiry System - PLANES" foi desenvolvido por WALTZ[47] na Universidade de Illinois em 1977. É uma interface para um SBD relacional que contém informações sobre manutenção e vôo de aviões.

Também utiliza a RTA, mas tem para cada objeto semântico diferente de seu universo de discurso, uma sub-rede em ATN. Essa estrutura torna todo o sistema dependente do universo do discurso.

Este sistema devolve ao usuário um parafraseado da pergunta, para confirmação, e permite que o usuário defina termos ou siglas, além de auxiliar na correção de erros de datilografia.

II.5.6. USL

O "User Speciality Languages - USL", foi desenvolvido por LEHMANN[28] para a IBM em 1978, como uma interface para um SBD.

A análise sintática do USL é feita por um parser que tem 800 regras em BNF modificado, e que fornece uma árvore intermediária para o analisador semântico. Este é composto por 70 regras de interpretação, que são aplicadas conforme a estrutura da árvore sintática, gerando por sua vez novas estruturas em árvore, que já não refletem a estrutura superficial. Esse nível intermediário permite a resolução da coordenação, quantificação e pronomes possessivos. Nesse novo nível as árvores tem nós rotulados com "objetivos", "relações", "domínios" e "atributos". Essas estruturas podem agora ser mapeadas para a linguagem do Banco de Dados.

Apesar de existir no projeto uma preocupação com a separação entre o que é e o que não é dependente da aplicação, o

sistema é dependente da aplicação.

II.5.7. DONAU

O projeto "Domain Oriented Natural Language Understanding - DONAU" foi desenvolvido por GUIDA[23] e Somalvico no Istituto di Elettrotecnica ed Elettronica Politecnico di Milano, Milão, em 1979, como uma interface para um SBD relacional com informações sobre recursos em petróleo.

A análise sintática é realizada por uma gramática de dependência, desenvolvida em Grenoble. Trata-se de uma gramática transformacional que tem como vocabulário o conjunto dos tipos léxicos, e regras de dependência onde sempre há um "governor" e um "dependent", e uma série opcional de "pesos". Esses pesos indicam a "distância" que o "dependent" pode ficar do "governor". A aplicação de regras sucessivas vai criando uma ou várias estruturas sintáticas em árvore.

A análise semântica escolhe entre as árvores construídas pelo analisador sintático, a que é mais consistente com o modelo semântico do sistema. O modelo semântico foi desenvolvido especialmente para o projeto DONAU, e se baseia em uma hierarquia de "model-list". Cada "model-list" é uma regra transformacional que exige certas regras de concordância estrutural com a árvore sintática. O resultado do analisador semântico deve ser uma só árvore semântica. Se no fim ele produzir mais de uma árvore, há ambiguidade semântica.

As árvores semânticas passam posteriormente por outros módulos do sistema, até se chegar à pergunta na linguagem do SBD.

DONAU exigiu apenas 25 Kbytes de memória para ser executada, e seu projeto foi estruturado em níveis horizontais e dois níveis verticais. Os níveis horizontais são os módulos que compõem o projeto, como o analisador léxico, o parser, etc. Cada módulo desses foi projetado em dois níveis verticais: o nível superior independente da aplicação e do sistema de suporte, e o nível inferior dependente. Essa estrutura é talvez a grande contribuição do projeto DONAU, pois tratou de definir formalismos e algoritmos com possibilidade de serem utilizados em outros sistemas, inclusive servindo como regras para projeto de sistemas similares.

II.6. CONCLUSÃO

No nível atual de pesquisa, todos sistemas que pretendem ter uma comunicação poderosa em linguagem natural entre homem e computador, permitindo diálogo esclarecedor, inferências ao contexto das perguntas, etc, necessitam ter um módulo semântico dependente do universo do discurso. Essa é uma restrição que provavelmente nos próximos anos não será superada, pois uma base semântica geral, com os meios que dispomos, é ainda algo muito complexo. É discutível inclusive se essa restrição será superada algum dia. A tendência parece ser no momento criar duas bases semânticas: uma dependente do contex

to, e outra que permita a solução de problemas mais ou menos constantes em Banco de Dados, como referências temporais, referências ao contexto das perguntas, quantificadores, coordenação, e resolução de pronomes e elipses.

A procura de independência do universo do discurso, objetivo da tese, leva ao enfraquecimento da comunicação entre o homem e o computador. É claro que este objetivo tem também suas vantagens, mas esta limitação deve ficar clara, para permitir posteriormente um julgamento adequado da interface IUC.

III. LOBAN

III.1. INTRODUÇÃO

A "Linguagem de Operações de Banco de Dados - LOBAN" é uma interface autocontida que contêm todas instruções necessárias para a definição dos dados, a manipulação dos dados na base de dados, a entrada e saída de dados, e para todas as funções de administração e manutenção da base de dados. Ela tem tabelas relacionais similares às tabelas dos modelos relacionais, mas tem também tabelas ligacionais que se aproximam das redes dos modelos de redes. Por isso LOBAN não se enquadra em nenhum dos três modelos de Banco de Dados.

Em SANTOS[40] há uma especificação detalhada de LOBAN. Aqui vamos nos referir apenas às caractetísticas que influem na compreensão deste trabalho, e muitas vezes de forma parcial.

III.2. AS CONSTRUÇÕES NA BASE DE DADOS

A interface LOBAN tem "pretipos" primitivos, a partir dos quais pode-se definir "tipos" de construções..

O pretipo mais elementar é o "átomo", e este pretipo contêm o pretipo "sigla", átomo de natureza não numérica, e os átomos de natureza numérica "natural", "inteiro" e "real". Na figura (III.1.a) está a representação gráfica dos átomos.

A "tupla" é uma nominação de nomes sobre átomos ou tuplas, e contêm os pretipos "numeração de caracteres", "inteiro decimal", "real decimal", "data" e "hora". A figura (III.1.b) é uma representação gráfica de uma tupla.

As tabelas são coleções. No caso de tabela relacional, elas são coleções de nominações, isto é, coleções de tuplas. No apêndice (1) há a representação de duas tabelas relacionais, uma no arquivo "Alunos" e a outra no arquivo "Disciplinas".

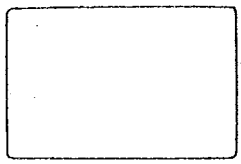
Uma ligação é uma nominação sobre uma tupla (o ligante) e uma tabela relacional (o ligado). O nome sobre a tupla é "L", e o nome sobre a tabela é T. A figura (III.1.c) é a representação gráfica de uma ligação.

A tabela ligacional é uma coleção de ligações. No anexo (1) há a representação de quatro tabelas ligacionais. São as tabelas nos arquivos "Cursos", "Requisitos", "Turmas" e "Históricos".

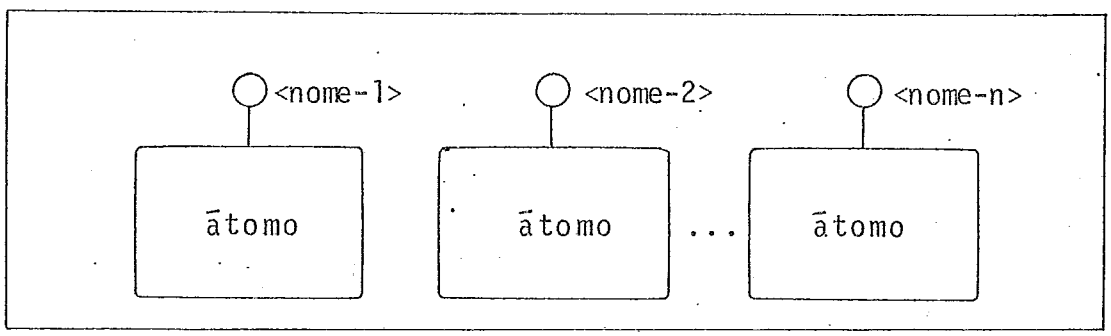
Há dois tipos de arquivos: o relacional e o ligacional. O arquivo relacional contêm uma tabela relacional, e o arquivo ligacional contêm uma tabela ligacional. No anexo (1) está a representação gráfica de dois arquivos relacionais (Alunos e Disciplinas), e a representação gráfica de quatro arquivos ligacionais (Cursos, Requisitos, Turmas e Históricos).

A "Folha" é uma construção na Base de Dados que contêm a descrição dessa Base de Dados através de "verbetes". Estes verbetes especificam todas as construções que podem ocorrer na Base de Dados por eles descrita.

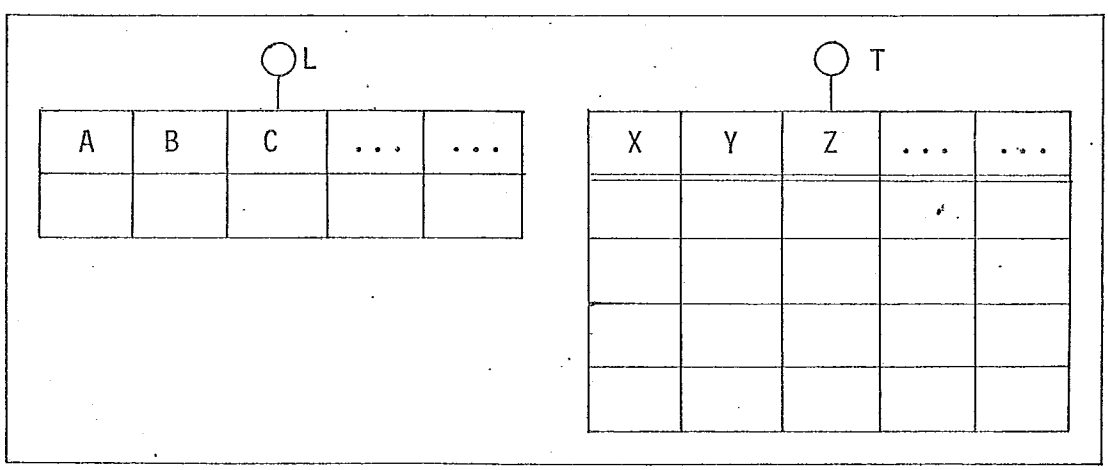
O "Acervo de Trabalho" é uma nominação, na qual constam a "Folha", sob o nome "Folha", e os arquivos relacionais ou ligacionais sob os nomes escolhidos pelo usuário.



a) Átomo



b) Tupla



c) Ligação

Figura - III.1.

III.3. ENDEREÇO DE PONTO

Um ponto identifica a posição relativa de uma construção dentro do contexto de referência, através de uma sequência de pares (nome, construção sob esse nome).

O endereço de ponto é a forma pela qual a interface possibilita ao usuário identificar os pontos. É um sistema de cima para baixo que parte de um ponto de referência (contexto), e busca o ponto imediatamente abaixo do ponto de referência, continuando assim até o nível de endereçamento desejado.

Vejamos um exemplo: no acervo do apêndice (1) queremos endereçar as tuplas de todos alunos do curso de Química. O endereço de ponto para isso será:

```
ACTRAB.ALUNOS.TR.(C CURSO = "QUIMICA")
```

Neste endereço de ponto ACTRAB é um identificador de ponto que identifica o contexto do acervo de trabalho. A cada ponto gráfico descemos um nível na avaliação do endereço de ponto. Assim, após ACTRAB.ALUNOS o contexto selecionado é o do arquivo ALUNOS. Após o endereço ACTRAB.ALUNOS.TR o contexto selecionado é o da tabela relacional, e após a avaliação completa do endereço de ponto são selecionadas (apontadas) todas tuplas que tem a construção "QUIMICA" na construção sob o nome curso.

Como foi dito, o endereço de ponto apenas seleciona os pontos. Se queremos agora recuperar as construções nesses pontos selecionados, devemos utilizar uma instrução que faça isso.

Para facilitar referências futuras a pontos endereçados, podemos marcá-los com "marcas", e depois através dessas marcas tornar a selecionar esses pontos sem ter de avaliar novamente o endereço de ponto dos pontos marcados.

III.4. INSTRUÇÕES

LOBAN tem instruções de controle de fluxo, instruções de operação sobre a base de dados, instruções de gerência, instruções de proteção, instruções de reconstrução e instruções de entrada e saída. Todas essas instruções operam sobre construções selecionadas por endereço de ponto, endereço de campo (no caso de entrada e saída) ou então sobre construção obtida através de outra instrução embutida.

Vejamos um exemplo: queremos representar as tuplas das disciplinas com carga horária igual a seis, em um relatório. Para facilitar a compreensão do exemplo, vamos deixar o endereço de campo no relatório apenas como FITA.REL( ). A instrução é:

```
FAZER PARA CADA PONTO
   ACTRAB.DISCIPLINAS.TR.(C CARGA-HORARIA.TEORICA = 6)
     E PARA CADA CAMPO
   FITA.REL( )
     (REPRESENTAR PC EM CC)
```

A instrução FAZER é uma instrução de controle de fluxo que permite executar a instrução REPRESENTAR, "navegando" por todos pontos selecionados nos endereços de ponto e campo.

III.5. COERÊNCIA

Para definir os acervos permitidos na base de dados, bem como as transições permitidas, utiliza-se a coerência.

A coerência subdivide-se em consistência e persistência. A consistência define a composição das construções, e as conexões entre construções. A persistência define as transições permitidas na base de dados.

O conteúdo da base de dados, o acervo total, deve estar completamente definido através dos verbetes de coerência, em todos seus níveis.

Já vimos que os tipos de construções em LOBAN são definidos através dos tipos de seus componentes imediatos, seguindo assim até o nível mais terminal (átomos). Portanto cada tipo de construção composta é definida por composição, isto é, por seus componentes imediatos.

No apêndice (2) está a coerência da base de dados apresentada graficamente no apêndice (1). Definimos nessa coerência apenas a composição das construções, por ser este o único aspecto da coerência que interessa para o presente trabalho.

IV. A INTERFACE IUC

IV.1. INTRODUÇÃO

Este capítulo apresenta uma especificação da linguagem, independente de implementação.

O objetivo deste trabalho é projetar uma interface para a comunicação entre usuários causais e um Sistema de Banco de Dados intependente do universo do discurso das aplicações. Evidentemente deve haver uma relação entre a interface e o universo do discurso das aplicações. Essa relação é estabelecida pelo dicionário, que contêm às palavras particulares às aplicações, com suas características.

Esse objetivo, como já foi dito no capítulo (III), implica em limitações a serem impostas à linguagem. A escolha dessas limitações obedeceu a dois critérios básicos:

1º) Não limitar demasiadamente o poder da IUC.

2º) Manter a IUC como uma interface em linguagem natural.

O primeiro critério é óbvio. O segundo critério significa impor regras fáceis de um usuário casual assimilar rapidamente, e que não desfigurem o Português. Fugindo desse segundo critério estaremos criando uma nova linguagem artificial, mesmo que utilize as palavras e certas construções do Português.

IV.2. AS LIMITAÇÕES

A primeira limitação permite ao usuário casual fazer apenas perguntas ao SBD através da IUC. Essa limitação é forte, mas não tira o poder da IUC. Isto porque as linguagens naturais permitem muitas ambiguidades, e em muitos casos não precisam o alcance de uma assertiva. Para a qualidade dos dados em um Banco de Dados, é necessário o maior controle possível sobre os dados de entrada, e esse controle só pode ser realizado através de linguagem não ambígua e com instruções perfeitamente definidas quanto ao alcance. Por isso não é interessante permitir alterar a base de dados através de instruções em linguagem natural.

A segunda limitação permite apenas o entendimento, e não a produção de frases. Essa limitação torna a comunicação em um só sentido: do usuário para a IUC. A comunicação no sentido contrário será realizada pelo SBD. A IUC não tem diálogo esclarecedor, e a implementação de perguntas ao usuário para resolver ambiguidades ou palavras desconhecidas, será realizada em linguagem não natural. O fato da resposta ser dada pelo SBD no formato do SBD e não em linguagem natural, tem a vantagem de não introduzir ambiguidades, o que poderia ocorrer no caso da resposta em linguagem natural.

A terceira limitação permite realizar as perguntas apenas por escrito, via teclado de terminal. Essa limitação elimina a necessidade de um componente fonológico na IUC.

A quarta limitação impõe a análise das perguntas independentes uma das outras. A IUC não mantem um contexto de

perguntas, o que permitiria em uma pergunta referir-se a outra. Essa limitação é forte, mas dentro do objetivo da IUC é quase impossível resolver os problemas de referência a contexto de conversação.

A quinta limitação impõe um formato rígido para o início das perguntas. Essa limitação é fácil de ser aprendida pelo usuário, e tem a finalidade de facilitar a análise das perguntas. Ela exige que todas perguntas iniciem por "ACHE", o que elimina as formas interrogativas, e orienta para uma forma imperativa, onde o usuário se refere às informações na terceira pessoa, mesmo que sejam informações referentes a ele mesmo. Essa limitação permite também exigir a ordem direta, o que não seria possível em perguntas na forma interrogativa.

IV.3. O FORMATO DAS PERGUNTAS E RESPOSTAS

As perguntas tem quatro formatos possíveis, a saber:

ACHE SE ...

ACHE A QUANTIDADE ...

ACHE A SOMA ...

ACHE ...

O primeiro formato é utilizado para saber se uma condição, ou uma série de condições coordenadas, são verdadeiras na base de dados. A resposta para esse formato de pergunta é "SIM" ou "NÃO".

Exemplo:

Pergunta: ACHE SE ALGUM ALUNO TEM TODAS NOTAS MAIORES QUE 5.
Resposta: SIM

O segundo formato é utilizado para saber quantas vezes ocorre determinada informação na base de dados. A resposta para esse formato é um número.

Exemplo:

Pergunta: ACHE A QUANTIDADE DE DISCIPLINAS QUE TEM PRÉ-REQUISITOS.
Resposta: 6

O terceiro formato é utilizado para saber a soma numérica de informações na base de dados, que obedecem a determinadas condições. A resposta para esse formato de pergunta é um número.

Exemplo:

Pergunta: ACHE A SOMA DOS CREDITOS DO ALUNO DE MATRICULA 3.
Resposta: 22

A escolha de dois formatos para respostas numéricas deve-se à ambiguidade inerente às palavras "quantidade","quantia", etc, que podem significar contagem ou soma. Por exemplo a pergunta "Ache quantos créditos o aluno de matrícula 6 tem" é ambígua, pois pode estar significando a soma dos créditos, ou a ocorrência (a contagem) de créditos. Para resolver essa ambiguidade foram estabelecidos dois formatos distintos, cada um com um significado.

O quarto formato é utilizado para recuperar informações da base de dados. Nesse formato o usuário deve primeiro enumerar os atributos (colunas de tabelas) que deseja recuperar, e a seguir pode coordenar quantas condições quiser a serem verificadas para os atributos a recuperar. Se o usuário não especificar os atributos, o sistema assume que são todos atributos da tabela referida na pergunta. Os atributos não necessitam pertencer à mesma tabela, mas tem de se referir à mesma entidade, por exemplo, alunos.

A resposta para o quarto formato é uma tabela relacional.

Exemplo:

Pergunta: ACHE O NOME E AS NOTAS DOS ALUNOS DE MATRICULAS 3 OU 4.

Resposta:

NOME	NOTA
LUIZ	5
LUIZ	7
LUIZ	9
ROSA	7,8
ROSA	8
ROSA	9
ROSA	9,2
ROSA	10

Pergunta: ACHE OS ALUNOS DO CURSO DE QUIMICA E DO SEXO FEMININO

Resposta:

MATRICULA	NOME	SEXO	DATA-NASC	CURSO	DATA-INGRESSO
4	ROSA	FEM	26/9/60	QUIMICA	1/3/79
7	BETTY	FEM	12/7/59	QUIMICA	1/3/80
14	MARIA	FEM	4/6/54	QUIMICA	1/3/80

IV.4. A GRAMÁTICA

Vamos especificar a linguagem IUC através de sua gramática, e discutir alguns aspectos importantes.

A gramática da IUC está apresentada a seguir em uma forma próxima à BNF. Não utilizamos uma forma rigorosa para apresentar a gramática, porque devido às irregularidades das linguagens naturais fica impossível fazê-lo, ou ficaria uma gramática tão extensa, que não auxiliaria na compreensão da linguagem. Assim, a ordem dos constituintes pode ser alterada em alguns casos.

Os constituintes em minúsculas são terminais ou categorias gramaticais. No caso das categorias, elas serão discriminadas apenas no dicionário.

PERGUNTA::= ache { se ORAÇÃO | a quantidade prep SN | a soma prep SN | SN }

ORAÇÃO::= {[adv] [SN] SV} [coord ORAÇÃO] ...

SN ::= { GS [GADJ] | ocor } [coord SN] ...

SV ::= GV { [[prep] SN] | GADJ } [GADV]

GV ::= [adv] [aux [prep]] verbo

GS ::= { [DET] { subst | pr.pess | pr.indef | pr.dem | num } | pr.relat }

GADJ ::= { adjet | LADJ | ORADJ | prep SN | ocor } [[coord] GADJ] ...

GADV ::= { adv | LADV | ORADV | prep SN } [[coord] GADV] ...

ORADJ ::= [prep] pr.relat ORAÇÃO [coord ORAÇÃO] ..

ORADV ::= subord ORAÇÃO [coord ORAÇÃO] ...

LADJ ::= { adj [coord adj] [prep] SN
         { prep SN [adv] { coord / prep } SN [adv]
         { conj.comp SN

LADV ::= prep SN [adv] [{ prep / coord } SN [adv]]

DET ::= { [pr.indef]
        { [art.def.] [pr.poss / num / pr.indef]
        { pr.dem [num]
        { art.indef

Na ORAÇÃO o sintagma nominal (SN) é opcional. Ele pode ser omitido quanto há várias orações ligadas por coordenação, e o SN aparece explicitamente apenas na primeira oração. É omitido também no caso de sujeito indeterminado (ache se existe ...). O SN pode vir após o sintagma verbal (SV) no caso de verbos intransitivos, ou no caso de orações adjetivas com verbo transitivo, e onde o objeto está no início da oração (caso de pronome relativo). Nas orações adjetivas o SN pode ser substituido pelo pronome relativo.

No sintagma nominal (SN) o grupo adjetival (GADJ) é opcional quando há uma repetição de grupos substantivais (GS) por coordenação. No caso de adjetivos de comparação (maior, menor, etc), o grupo adjetival pode vir antes do grupo substantival. Todo SN pode corresponder a uma ocorrência na base de dados.

No sintagma verbal (SV) é permitida a próclise de pronomes pessoais oblíquos. No caso de verbo intransitivo não há SN, nem GADJ. No caso de verbo de ligação há GADJ. No caso de verbo transitivo há SN ou prep SN.

Nas gramáticas das linguagens naturais é comum existir o sintagma preposicional, composto por uma preposição e um sintagma nominal. Na IUC esse sintagma existe como "prep SN", existe na locução adjetiva (LADJ) e existe na locução adverbial (LADV). Na locução adjetiva o sintagma preposicional está se referindo a um grupo substantivo (GS). Na locução adverbial (LADV) o sintagma preposicional está se referindo ao grupo verbal. Como "prep SN" pode estar se referindo tanto ao GS

quanto ao GV. Essa ambiguidade é difícil de eliminar, por isso ela foi mantida, mas sem afetar o funcionamento da interface.

O grupo substantival (GS) é composto por pronome relativo, ou por determinante (opcional) e substantivo ou seus substituidores. Quando o substantivo é substituido, a gramática, como está descrita, pode permitir a repetição incorreta de pronomes ou numerais. Por exemplo, um pronome indefinido determinante pode ocorrer novamente como substituidor do substantivo. Na realidade isso não é permitido, mas esse fato não está explícito na gramática para não complicá-la.

As orações adverbiais permitidas na IUC são apenas as condicionais e as temporais. As demais não são necessárias para os tipos de pergunta da interface.

As orações subordinadas comparativas normalmente tem o predicado oculto. Por isso na IUC as conjunções subordinadas comparativas não introduzem orações subordinadas, e sim comparações na locução adjetiva (LADJ).

A IUC não permite orações reduzidas. Essa talvez seja uma limitação a reestudar no futuro.

Essa gramática permite o embutimento ilimitado de frases. Por exemplo, ela aceita as perguntas:

ACHE O NOME DOS ALUNOS COM TODAS NOTAS MAIORES QUE 5.
ACHE O NOME DOS ALUNOS COM TODAS NOTAS MAIORES QUE AS NOTAS DOS ALUNOS QUE NASCERAM EM 1/1/60.

A primeira pergunta é uma pergunta simples. A segunda pergunta tem um embutimento.

A interface resolve os problemas de conjunções coorde

nadas, por "níveis". Cada vez que aparece uma conjunção coordenada, a interface tenta a repetição da última categoria que apareceu, no nível mais baixo. Se falhar, tenta a categoria no nível seguinte, e assim sucessivamente até o nível mais elevado. Vejamos um exemplo para ficar mais claro.

Seja a pergunta que aparece analisada na figura (IV.1). Na avaliação do primeiro "ou" dessa pergunta a interface tenta primeiro um SN à direita da conjunção. Como falha, tenta sucessivamente GADJ, SN e SV, acertando para SV. A avaliação do segunto "ou" é imediata, na primeira tentativa para GADJ.

IV.5. A GRAMÁTICA FUNCIONAL

Para permitir uma comparação com a Gramática Funcional do Português, apresentamos os termos da oração que a IUC admite e a estrutura do período, nas figuras (IV.2) e (IV.3).

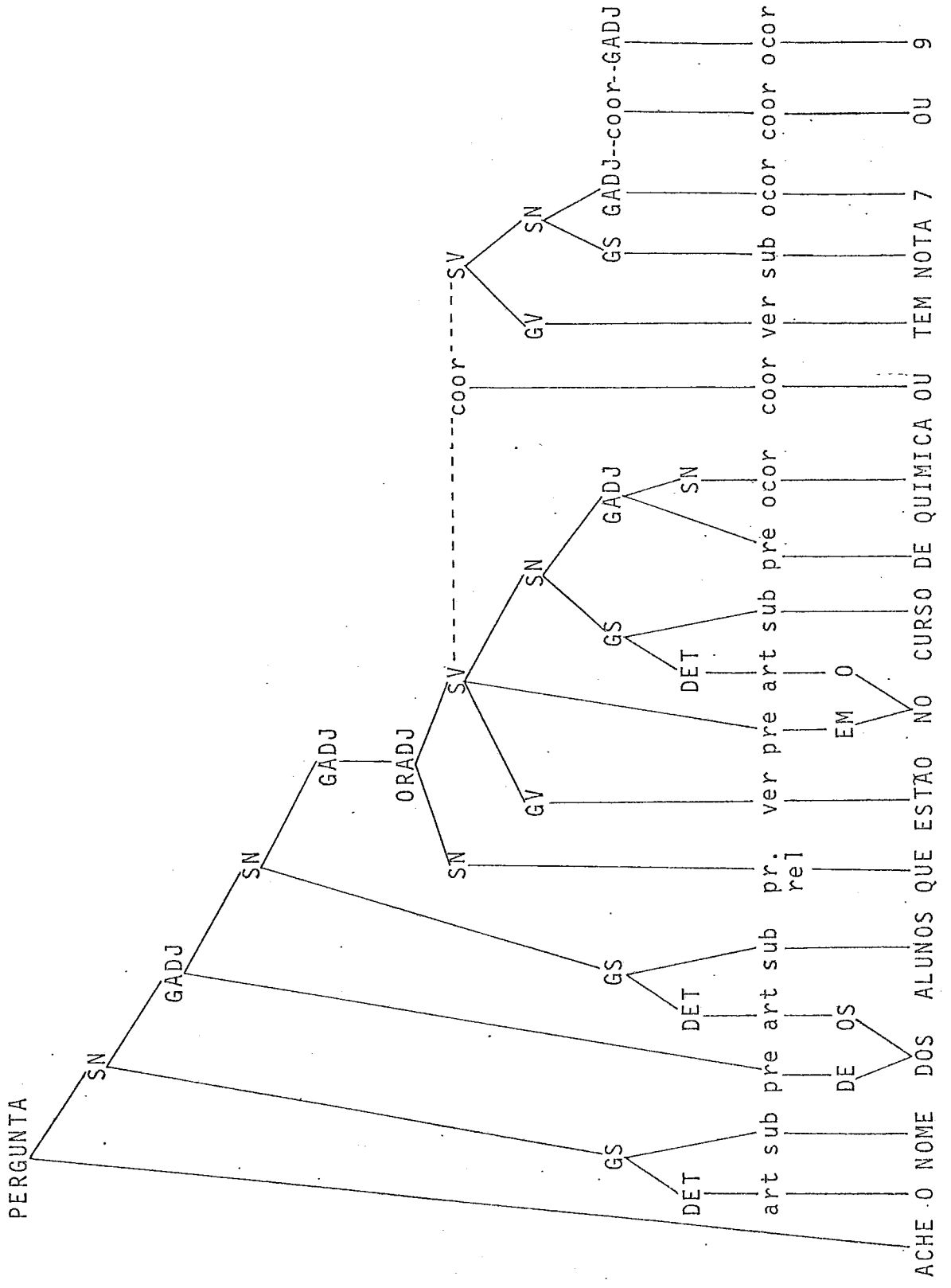

PERGUNTA
SN
GADJ
SN
GADJ
ORADJ
SV
coor
SV
SN
GADJ
SN
GADJ--coor--GADJ
GS
DET
art
sub
pre
DE
OS
dos
alunos
que
estão
no
curso
de
química
ou
GS
DET
art
sub
pr.
rel
GV
ver
pre
EM
O
GS
DET
art
sub
pre
ocor
GV
coor
ver
sub
ocor
coor
ocor
GS
GADJ--coor--GADJ
ACHE O NOME
TEM NOTA 7
OU
9

Figura - IV.1

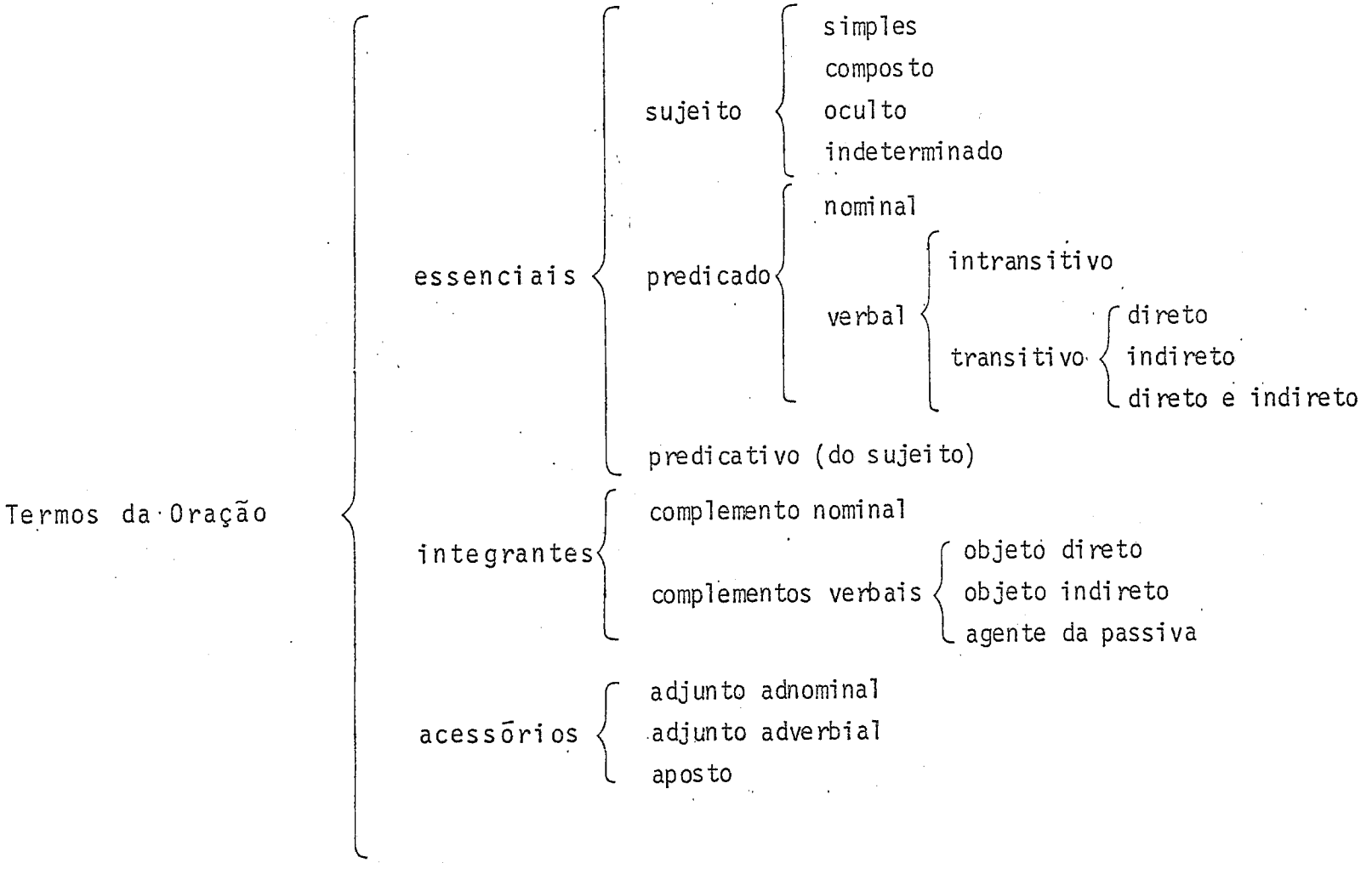

Termos da Oração
essenciais
sujeito
simples
composto
oculto
indeterminado
predicado
nominal
verbal
intransitivo
transitivo
direto
indireto
direto e indireto
predicativo (do sujeito)
integrantes
complemento nominal
complementos verbais
objeto direto
objeto indireto
agente da passiva
acessórios
adjunto adnominal
adjunto adverbial
aposto

Figura - IV.2

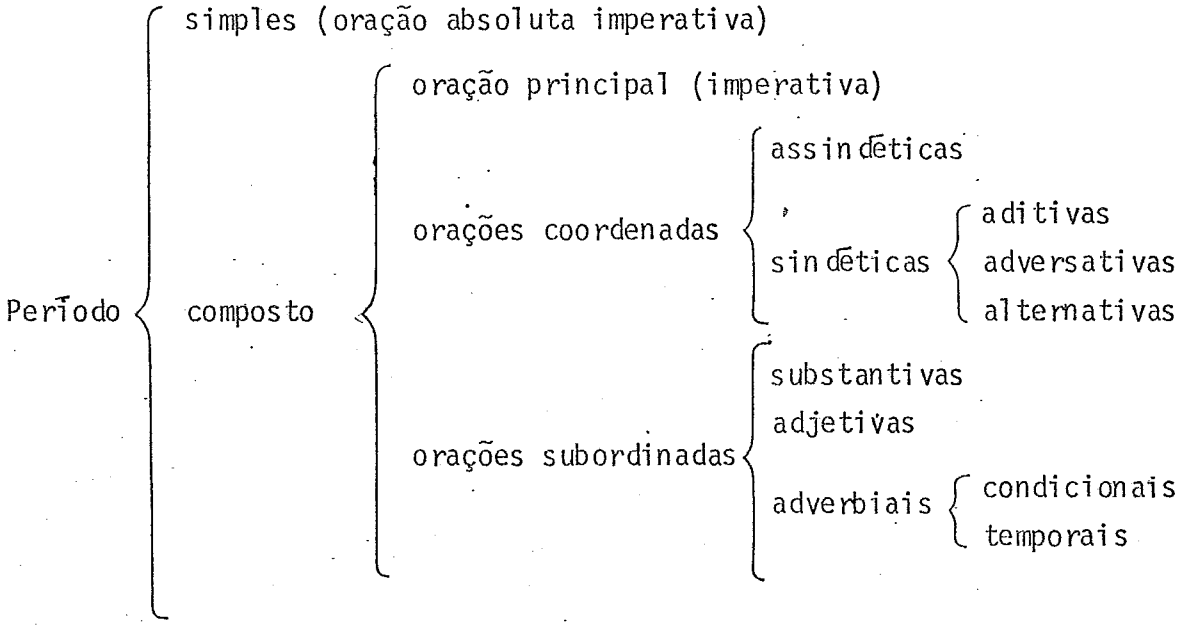

Período
simples (oração absoluta imperativa)
composto
oração principal (imperativa)
orações coordenadas
assindéticas
sindéticas
aditivas
adversativas
alternativas
orações subordinadas
substantivas
adjetivas
adverbiais
condicionais
temporais

Figura - IV.3

V. A SEMÂNTICA DA BASE DE DADOS

V.1. INTRODUÇÃO

Na base de dados temos "construções", que são a representação das informações do mundo real. Quando usamos um Banco de Dados em um sistema de informações, modelamos uma parte do mundo real de nosso interesse, segundo o modelo do Banco de Dados. No caso de LOBAN nós modelamos as "entidades" do mundo real através de seus "atributos", estruturas bem definidas. Exatamente essa estrutura, o modelo, nos dá informações semânticas sobre os componentes da base de dados e suas relações, que serão utilizadas no analisador sintático e semântico.

No apêndice (3) está representada a "estrutura semântica" da base de dados, assim como a estrutura de endereços da base de dados. Pode-se ver por esse desenho que os arquivos receberam um número que os identifica, assim como os atributos. Essa sequência de números é o "endereço", que posteriormente será convertido no endereço em LOBAN. Por exemplo, ALUNOS tem endereço 1, e NOME tem endereço 1.2.

V.2. OS TIPOS

Cada construção na base de dados pertence necessariamente a um tipo definido na "coerência" da base de dados, e um tipo é um conjunto de construções que possuem os mesmos predicados.

No apêndice (2) está definida a coerência do acervo que nos serve de exemplo. Dessa coerência nós podemos obter os tipos existentes na base de dados, e os atributos (representados pelos endereços numéricos) que possuem esses tipos, como está mostrado na tabela (V.1).

Essa tabela mantida pela IUC auxilia na resolução de elipses ou ambiguidade. Por exemplo, se em uma pergunta aparecer a expressão " ... ALUNOS COM NOTA 8 ...", sabemos que 8 é uma ocorrência de nota. No entanto se aparecer a expressão "... ALUNOS COM 8 ...", pode ficar difícil de identificar 8 como ocorrência de nota. Nesse caso a interface verifica a "pertinência" de 8 a todos tipos da base de dados. Neste exemplo a pertinência se verifica para real, inteiro (> 0 e < 999999), inteiro (< 9) e inteiro (> 5 e < 31). Portanto 8 só pode ser ocorrência dos atributos nos endereços obtidos da tabela para as pertinências verificadas, a saber: 6.5, 1.1, 5.4, 6.1, 2.4, 6.3 e 5.3.

Essa análise de pertinência a tipos é realizada na análise léxica.

V.3. AS ENTIDADES, ATRIBUTOS E OCORRENCIAS

Há uma relação entre as entidades da realidade que queremos modelar e os arquivos na base de dados. Por exemplo a entidade "alunos" é descrita pelos arquivos "ALUNOS", "TURMAS" e "HISTÓRICOS". Este exemplo mostra que uma entidade pode corresponder a um ou mais arquivos. Se corresponder a mais de um arquivo, necessariamente deve haver um elo de ligação entre eles, e este elo deve ser "chave" em pelo menos um arquivo. No exemplo o elo da ligação é a matrícula, conforme pode ser visto na figura do apêndice (3). Os elos são representados pelas linhas na parte inferior da figura do apêndice (3).

As entidades são descritas por seus atributos (colunas das tabelas), e cada ocorrência da entidade é identificada pelas ocorrências de seus atributos.

Esse modelo nos permite as seguintes relações semânticas, que serão utilizadas na análise das frases:

a) Só posso recuperar em uma pergunta atributos de uma mesma entidade. Se esses atributos não pertencem à mesma tabela, devem pertencer a tabelas com elo de ligação.

Exemplo: ACHE O NOME E A NOTA DOS ALUNOS ...

b) Uma condição para a seleção da ocorrência de entidade só pode ser referir a atributos dessa entidade.

Exemplo: ACHE O NOME DOS ALUNOS DO CURSO ...

c) Uma ocorrência de atributo deve pertencer ao tipo do atributo.

Exemplo: ACHE O NOME DOS ALUNOS DO CURSO DE QUIMICA E ..

V.4. A DOMINÂNCIA

No modelo LOBAN há uma hierarquia entre entidades, atributos e ocorrências de atributos. Para expressar essa hierarquia, dizemos que entidade "domina" atributo, e atributo domina ocorrência de atributo.

Essa relação de dominância deve ser mantida nas perguntas. Assim em um sintagma preposicional "prep GS GADJ" se o GADJ é uma ocorrência (de atributo) na base de dado, então necessariamente o GS deve ser o atributo que domina essa ocorrência.

Outro exemplo: se temos em uma pergunta o formato "GS prep GS prep ocor", onde ocor é uma ocorrência de atributo, então o primeiro GS é uma entidade que domina o atributo referido pelo segundo GS, que por sua vez domina ocor.

V.5. AS CHAVES

Em LOBAN chave é um atributo ou um grupo de atributos, que identificam uma e apenas uma tupla ou ligação nas tabelas relacionais e ligacionais, respectivamente. Ela portanto identifica a ocorrência da entidade. Na figura do anexo (3) as chaves das tabelas são os atributos com os circulos pretos, ou pretos pela metade (chaves substitutas).

Esse significado das chaves, deve ser verificado nas perguntas. Na pergunta "... ALUNO COM MATRICULA 8 ...", fica claro que 8 é uma ocorrência de matrícula, portanto uma ocor

rência de chave. Se no entanto agora a pergunta for " ... ALUNO 8 ...", a elipse da referência ao atributo cria ambiguidade, pois 8 pode se referir a qualquer atributo de aluno. Como no entanto ALUNO está no singular, portanto identificado por 8, então é uma chave, logo 8 é resolvido como a matrícula do aluno. Na pergunta "... ALUNOS COM 8 ... ",8 não pode ser ocorrên de chave porque aluno está no plural.

TIPOS EXISTENTES NA BASE DE DADOS

PRÉ-TIPO	CRITÉRIO P/ AVALIAÇÃO DA PERTINÊNCIA	ENDEREÇO
NUMCAR	POR 1:30 DE LETRA ∪ BRANCO	1.2
	POR 6 DE CARACTERE	2.1,3.3,4.1,4.2,5.1,6.2
	POR 8:15 DE LETRA ∪ BRANCO	2.2
LETRA		5.2
DATA		1.4,1.6
REAL	$\leq$ 10	6.5
INTEIRO	> 0 E < 999999	1.1,5.4,6.1
	< 7	2.3.1,2.3.2
	$\leq$ 9	2.4,6.3
	= 1 OU = 2	2.5,6.4.2
	> 50	3.2
	> 5 E < 31	5.3
	> 78 E < 99	6.4.1
NUMCAR	DEFINIÇÃO POR EXTENSÃO: AS POSSÍVEIS OCORRÊNCIAS ECONTRAM-SE NO DICIONÁRIO, A SABER:	
	MASC	1.3
	FEM	1.3
	QUIMICA	1.5,3.1
	FISICA	1.5,3.1
	OBR	3.4
	ELET	3.4
	PRE	4.3
	CO	4.3

TABELA V.1

VI.1. O DICIONÁRIO

VI.1. INTRODUÇÃO

O dicionário contem todas as palavras aceitas pelo sistema, excetuadas as ocorrências na base de dados, que podem constar ou não do dicionário. Para cada palavra o dicionário contem uma série de informações sintáticas e semânticas.

Ele contem palavras permanentes e independentes da aplicação do Banco de Dados (conjunções, preposições, advérbios, numerais, pronomes e artigos) e palavras que dependem do universo do discurso da aplicação (substantivos, adjetivos e verbos). Desta forma, o dicionário é o módulo que deve ser refeito a cada nova aplicação do Banco de Dados.

Para construir o dicionário deve-se introduzir inicialmente as entidades, os atributos, as ocorrências não numerosas, os verbos comuns ao universo do discurso da aplicação, e os sinônimos dessas palavras.

No apêndice (4) está o dicionário para a aplicação que tem servido de exemplo.

VI.2. AS ENTRADAS

As palavras no Português tem muitas flexões. Se para cada flexão diferente as palavras constassem do dicionário, ele ficaria enorme, mesmo com poucas palavras. Para evitar esse problema, o analisador léxico ao recuperar as palavras da pergunta, realiza uma análise da flexão da palavra, devolvendo uma "base" e a flexão da palavra. A base será a entrada no dicionário.

A fim de não repetir as características sintáticas e semânticas de palavras sinônimas, essas características constam apenas de uma palavra, e nos sinônimos há um apontador para essa palavra.

O dicionário tem palavras isoladas e lexias. Para isso na base da palavra inicial de lexia há um apontador que aponta para o complemento da lexia. Neste caso o analisador léxico verifica se existe o complemento na pergunta, e se existir, então agrupa as palavras formando a lexia, como um único conjunto de características sintáticas e semânticas.

O dicionário resolve as contrações. Por exemplo, ele transforma "dos" em "de os", como se fossem sinônimos.

As palavras com flexões irregulares aparecem no dicionário a cada flexão irregular. É o caso, por exemplo, dos verbos irregulares.

VI.3. O FORMATO

Para cada base o dicionário tem todas as flexões possíveis, à exceção dos verbos, que tem um analisador de flexões que devolve a informação de tempo e pessoa a que corresponde a flexão.

As flexões podem ser completamente independentes, ou então podem ser flexões de uma mesma palavra. No segundo caso elas introduzem uma informação particular (por exemplo, o gênero e o número nos substantivos), e apontam todas para o mesmo conjunto de características da palavra.

Cada "base + flexão" ou então "base + flexões" pode ter um apontador para complementos (para fornecer as lexias) ou um apontador para sinônimo. Além disso tem a categoria gramatical da entrada e as características sintáticas e semânticas.

No caso de homógrafos o dicionário devolve todas as interpretações dessas palavras, na ordem em que estão no dicionário.

As características sintáticas e semânticas podem se referir à própria palavra, a seu "antecedente" na pergunta, ou a seu "consequente". Por exemplo, o verbo pode ser transitivo, mas também pode exigir que seu antecedente (o sujeito) seja humano, e que seu consequente (o objeto) seja um curso ou uma disciplina.

Intencionalmente não utilizamos aqui a terminologia da gramática de casos (FILMORE[16], BRUCE[6]) nem a terminologia da gramática funcional (CUNHA[11,12]). Procuramos com isso evitar confusões.

Entre as características sintáticas e semânticas podemos ter traços (humano, animado, inanimado, concreto, abstrato), marcas (tempo, tempo passado, tempo presente), condições (nota maior que 5), especificações (particulariza um antecedente ou consequente), e o significado da palavra.

VII. A ANÁLISE LÉXICA

VII.1. INTRODUÇÃO

O analisador léxico é o módulo que faz a primeira análise da pergunta. Ele separa as palavras, concatena palavras para formar lexias ou ocorrências, verifica a classe das palavras, a categoria gramatical, os endereços da base de dados, e recupera todas informações sintáticas e semânticas do dicionário.

No fim da análise léxica o analisador chama o analisador sintático e semântico, e passa para este a pergunta decomposta em unidades léxicas. Cada unidade léxica está em um registro que contem a lexia, e todos as informações sobre a lexia.

VII.2. A ORTOGRAFIA

Na especificação funcional da interface IUC não fizemos o detalhamento da ortografia, porque ela depende do conjunto de caracteres existentes no terminal do sistema portador, e da flexibilidade que se quer dar ao usuário. Agora vamos apresentar uma ortografia "mínima", para a partir dela discutir o analisador léxico.

A interface IUC necessita como mínimo, os seguintes caracteres:

a) As letras do alfabeto Português, mais a letras "k", "w" e "y".

b) O hífen para ligar os elementos de palavras compostas, para ligar ao verbo pronomes enclíticos, e para separar dia, mês e ano nas datas.

c) Os sinais de pontuação, vírgula e ponto e vírgula.

d) As aspas para iniciar e terminar "literais".

e) A barra e o ponto, além do hífen, para separar dia, mês e ano nas datas.

A utilização desses caracteres define as seguintes classes léxicas de lexias na interface IUC:

a) A classe literal-alfabet, formada pelas lexias iniciadas e terminadas por aspas, e com os caracteres letras, espaço e hífen entre as aspas.

b) A classe literal-alfanum, formada pelas lexias iniciadas e terminadas por aspas, e com qualquer caractere aceito pela IUC entre as aspas.

c) A classe alfanum, formada pelas lexias escritas com os caracteres letra, número, hífen, barra, ponto, vírgula e ponto e vírgula.

d) A classe alfabet, formada pelas lexias escritas com os caracteres letra, hífen e espaço.

e) A classe data, formada pelas lexias escritas com dia, mês e ano separados por ponto, hífen ou barra.

f) A classe real, formada pelas lexias escritas com dígitos seguidos de uma vírgula e mais dígitos.

g) A classe inteiro, formada pelas lexias escritas com dígitos.

h) A classe pontuação, formada pelas lexias escritas com vírgula ou ponto e vírgula.

Vamos ver como é a sintaxe em BNF, das lexias "mínimas" aceitas pela IUC:

```
<lexia>              ::=  <literal-alfabet>
                        |<literal-alfanum>
                        |<alfanum>
                        |<alfabet>
                        |<data>
                        |<real>
                        |<inteiro>
                        |<pontuação>

<literal-alfabet>::= <aspas> <alfabet> <aspas>
<literal-alfanum>::= <aspas> <alfanum> <aspas>
<alfanum>          ::= <carac>|<alfanum> <carac>
<alfabet>          ::= <alfa> |<alfabet> <alfa>
<data>             ::= <dm> <sep> <dm> <sep> <ano>
<real>             ::= <inteiro> <virg> <inteiro>
<inteiro>          ::= <digito> |<inteiro> <digito>
<pontuação>        ::= , | ;
<aspas>            ::= "
<carac>            ::= <alfa>|<digito>
<alfa>             ::= <letra>| - |Ø
<dm>               ::= <digito>|<digito> <digito>
```

```
<sep>          ::= .|/|-
<ano>          ::= <digito> <digito>|19<digito> <digito>
<virg>         ::= ,
<digito>       ::= 0|1|2|...|9
<letra>        ::= a|b|c|...|z
```

A respeito dos literais cabe uma explicação. Às vezes é difícil encontrar onde termina uma lexia composta por mais de uma palavra, como é o caso dos nomes próprios. Para diminuir essa fonte de ambiguidades o usuário pode escrever as ocorrência na base de dados entre aspas, como é feito nos literais de linguagens artificiais. Como no entanto essa é uma forma artificial, portanto não natural, a IUC aceita que se escreva as numerações de caracteres na base de dados das duas formas.

VII.3. A ESTRUTURA DE DADOS

O analisador recebe como entrada uma pergunta escrita pelo usuário, e fornece como saída uma sequência de registros. Cada registro contem uma lexia e uma lista de significados, na mesma ordem em que eles estavam no dicionário. Há um significado para cada homógrafo. Por exemplo, a palavra "e" pode ser uma conjunção ou o verbo ser (isto se não existir acentos no terminal), portanto tem dois significados.

Cada significado tem uma categoria sintática, e pode ter ou não outros componentes. Se a categoria é pontuação, conjunção, preposição ou advérbio, o significado só tem a catego

ria. Se a categoria é pronome ou artigo, o significado tem também o gênero e número. Se a categoria é numeral e a lexia tem flexão de gênero femimino, então o significado tem categoria numeral, senão há dois significados: um com a categoria numeral, e outro com a categoria inteiro. Se a categoria é inteiro, data, numeração de caracteres (numcar) ou real, o significado tem a categoria e os endereços (obtidos da tabela de tipos da base de dados). Finalmente se a categoria é adjetivo, substantivo ou verbo, além da categoria, o significado pode ter uma lista de características, como endereço, gênero, número, tempo, pessoa, traço, antecedente, consequente, e todas outras características que constem no dicionário.

VII.4. AS FLEXÕES DAS PALAVRAS

Para diminuir as entradas no dicionário as palavras são decompostas em "base" e "flexão", como já foi explicado no capítulo (VI). Desta forma haverá no dicionário uma entrada por base, mesmo que existam muitas flexões.

No projeto dessa decomposição não nos preocupamos em definir um analisador morfológico para o Português, mas apenas em separar da forma mais eficiente as flexões mais comuns que ocorrem na IUC, e dentro do possível aproveitar essa decomposição para obter informações sobre modo verbal, tempo, pessoa, gênero, número, e ocorrência de pronomes pessoais oblíquos enclíticos.

As informações obtidas das flexões não são completas, e são equivocadas para alguns casos específicos. Nesses casos repetimos o mesmo tipo de informação no dicionário, e quando houver discrepância entre o que é obtido da flexão e o que é obtido do dicionário, dá-se preferência às informações do dicionário.

Na tabela (VII.1) estão apresentadas as flexões, e suas características. A coluna "E" dessa tabela informa quando a flexão corresponde a um pronome pessoal oblíquo átono (*), ou quando a flexão é "vazia" (Ø). A coluna "MTP" informa o modo, tempo e pessoa das flexões verbais, nessa ordem, e segundo a codificação:

a) Modo: 1 para indicativo ou subjuntivo; 2 para infinitivo; 3 para gerúndio; 4 para particípio.

b) Tempo: 1 para passado; 2 para presente.

c) Pessoa: 3 para a 3ª do singular; 6 para a 3ª do plural.

A coluna "GN" informa o gênero e o número nessa ordem, e segundo a codificação:

a) Gênero: 1 para masculino; 2 para feminino.

b) Número: 1 para singular; 2 para plural.

VII.5. O ALGORITMO

O algoritmo do Analisador Léxico - AL está no apêndice (5). Ele está bem detalhado para mostrar como tentamos resolver os problemas de lexias, palavras compostas como nomes próprios, sinônimos, ênclise de pronomes e homógrafos.

A linguagem para definir o algoritmo é uma linguagem estruturada simples, que utiliza os seguintes símbolos: procedimento, se, então, senão, nada, enquanto, faça, repita, até que, início, fim, execute, devolva, recupere, para cada.

TABELA DE FLEXÕES

FLEXÃO	E	MTP	GN	FLEXÃO	E	MTP	GN
A		123	21	S			-2
OA			21	AS			22
ONA			21	OAS			22
-LA	*		21	ONAS			22
-NA	*		21	-LAS	*		22
-A	*		21	-NAS	*		22
ADA		4--	21	-AS	*		22
IDA		4--	21	ES			
-LHE	*		-1	AES			
E		123		OES			
ASSE		113		ARES			
ESSE		113		ERES			
ISSE		113		IRES			
I	ø			-LHES	*		
O			11	IS			
AO				OS			12
-LO	*		11	-LOS	*		12
-NO	*		11	-NOS	*		12
-O	*		11	-OS	*		12
ADO		4--	11	EUS			
IDO		4--	11	NS			
ANDO		3--	11	ENS			
ENDO		3--	11	ADAS		4--	22
INDO		3--	11	IDAS		4--	22
U	ø			ADOS		4--	12
EU		113		IDOS		4--	12
IU		113		X	ø		
OU		113		Z	ø		
L							
M							
AM		126					
EM		126					
ARAM		116					
ERAM		116					
IRAM		116					
ASSEM		116					
ESSEM		116					
ISSEM		116					
N	ø						
R	ø						
AR		2--					
ER		2--					
IR		2--					

TABELA VII.1

VIII. A ANÁLISE SINTÁTICA E SEMÂNTICA

VIII.1. INTRODUÇÃO

As análises sintática e semântica são realizadas por uma "rede de transição aumentada". Na maioria dos sistemas já implementados essas duas análises são feitas por algoritmos em módulos separados. Essa escolha tem o objetivo de facilitar a programação, ao estabelecer módulos distintos para funções distintas.

No nosso caso optamos por colocar as duas análises em um só módulo para diminuir o "retorno" ("backtracking"). Esse retorno se dá quando caminhamos na rede por uma ou mais arestas até chegar a um estado no qual nenhuma aresta pode ser seguida, seja pela lexia de entrada, ou seja por condições exigidas nessas arestas. Neste caso devemos retornar o caminho andado na rede, até um estado onde seja possível tomar outra aresta diferente. Nesse retorno devemos desfazer a análise feita a cada transição, obrigando portanto a uma grande perda de tempo. Juntando as duas análises, a análise semântica orienta o caminho na rede, diminuindo com isso o retorno.

Enquanto caminha na rede, o analisador vai montando em registros a "árvore sintática", e em outros registros vai montando uma estrutura em listas para o "Gerador de Perguntas em LOBAN". No caso de retorno essas estruturas deverão voltar ao estado anterior. Para facilitar esse procedimento, todos registros são listas de valores, e toda vez que atribuimos um

novo valor a um registro, agregamos um novo elemento à lista desse registro, não destruindo os valores anteriores. Essa solução facilita o retorno.

VIII.2. A REDE DE TRANSIÇÃO AUMENTADA

A RTA foi apresentada no capítulo (II). Vamos agora mostrar a sintaxe e a semântica de uma RTA similar à desenvolvida por BORGIDA[4] e MYLOPOULOS[30-31-32], e mais simples que a rede desenvolvida por WOODS[51]. Essa RTA está especificada para um interpretador a ser programado em SNOBOL.

<rta> ::= (<arco>)$^{+}$

<arco> ::= <estado$_1$> (T<teste>) <tipo> (<ação>) <estado$_2$>

<tipo> ::= CAT <nome categoria>

| LEX <lexia>

| LL <lista lexia>

| EMPIL (<pré-ação>)

| DESPIL

| VIR <arg>

| DESVIO

<estado$_1$> ::= <identificador>

<estado$_2$> ::= <identificador>

<teste> ::= <identificador>

<ação> ::= <identificador>

<nome categoria> ::= inteiro|data|numcar|real|substantivo
|verbo|adjetivo|pronome|artigo|conjunção
|preposição|numeral|advérbio

<lexia> ::= <cadeia de caracteres>

<lista lexia> ::= (<lexia>)$^{+}$

<pré-ação> ::= <identificador>

<arg> ::= <lexia>|<lista-lexia>|<nome categoria>

Vejamos agora a semântica dessa rta. Ela é composta por arcos. Cada arco tem um <estado$_1$> (estado inicial) e um <estado$_2$> (estado final), para o qual segue o processamento após a transição.

Em todos arcos podem existir testes sobre os registros com estruturas, ou sobre registros com condições (bandeiras "flags"). O arco só será seguido se o teste for verdadeiro.

Nos arcos podem existir também ações que operam sobre os registros no nível de computação, ou em registros em nível inferior de computação.

Há vários "tipos" de arcos. No arco CAT o argumento é um nome de categoria, e ele é seguido se a categoria da lexia de entrada corresponde à categoria do argumento.

No arco LEX o argumento é uma lexia, e ele pode ser seguido se a lexia de entrada é igual à lexia do argumento.

No arco LL o argumento é uma lista de lexias, e ele pode ser seguido se a lexia de entrada pertence a essa lista de lexias.

O arco EMPIL para o processamento corrente, empilha o próximo estado, e desvia para um nível inferior de processamento. Esse arco e o arco DESPIL permitem a recursão na rede. Antes de seguir para o nível inferior de processamento, são realizadas as "pré-ações" do arco.

O arco DESPIL faz o processamento retornar ao nível superior, no ponto em que havia sido interrompido. Uma das condições para ele ser seguido, é estar vazio o registro GUARDA. Esse registro tem a função de guardar temporariamente es

truturas de entrada, "deslocadas" de sua posição verdadeira por inversões permitidas pelas linguagens naturais. O <estado$_2$> desse arco é vazio.

O arco VIR tem como argumento uma categoria, uma lexia ou uma lista de lexias, e será seguido se a lexia no registro GUARDA do nível de processamento, ou de nível superior, coincidir com o argumento.

O arco DESVIO causa um desvio para o estado com nome em <estado$_2$>, sem avançar para a próxima lexia de entrada.

Os identificadores das ações e pré-ações apontam rotinas na linguagem do interpretador, com pelo menos uma instrução de retorno no fim. Essas rotinas utilizam funções que operam sobre os registros.

O identificador dos testes aponta para uma expressão a ser avaliada, que retorna "verdadeiro" ou "falso".

São três as funções da RTA que operam sobre os registros, e tem o formato:

ATR <registro> <valor>
ENV <registro> <valor>
REC <registro> <valor>

A função ATR realiza a atribuição de um valor a um registro, no nível atual de processamento.

A função ENV é similar à função ATR, mas a atribuição é para registro em nível inferior de processamento. Essa função é utilizada nas pré-ações dos arcos tipo EMPIL.

A função REC recupera o conteúdo de um registro.

O registro EST é muito usado na rede, e seu valor é atualizado da seguinte forma:

a) Nos arcos CAT, LEX e LL ele é atualizado com o conteúdo do registro de entrada, e o registro de entrada passa a ser o próximo registro na lista de registro de entrada.

b) Nos arcos VIR ele é atualizado com o conteúdo do registro GUARDA, e o registro GUARDA perde esse conteúdo.

c) Nos arcos DESPIL ele é atualizado com a estrutura obtida no processamento desse nível, através das ações nesses arcos.

Paralelamente ao registro EST é atualizado o registro PROX, que contém sempre o registro com a próxima lexia de entrada. Esse registro permite testes sobre a próxima lexia.

O registro GUARDA é uma pilha, e seu uso já foi descrito.

A execução da RTA é controlada pela função EXEC, que guia o analisador passo a passo a partir do estado inicial, e que é chamada recursivamente no fim de cada aresta. A função EXEC tem o seguinte algoritmo:

1 - Encontra a primeira aresta saindo do nó corrente que ainda não foi experimentada. Se não há mais arestas a experimentar a função EXEC falha, e o controle retorna para a função EXEC que controla o nó anterior, após desfazer todas ações executadas na última transição de estado.

Cabe explicar aqui que as arestas que saem de um nó são ordenadas, por isso é possível uma ordem de pesquisa. Es

sa ordem é estabelecida de maneira a por no início as arestas mais utilizadas.

2 - Se há algum teste na aresta ele é avaliado, e em caso de falha volta ao passo 1.

3 - A função "tipo" que aparece na aresta é chamada, avaliada e executada. Se houver falha, voltar ao passo 1.

Se o tipo de função é EMP e há pré-ações, as mesmas são processadas antes do processamento da função.

4 - As ações especificadas na aresta são executadas. Em caso de falha, voltar ao passo 1.

5 - Chamar EXEC. Em caso de falha voltar ao passo 1.

VIII.3. AS ESTRUTURAS DE DADOS

O Analisador Sintático e Semântico (ASS) recebe uma estrutura de dados do Analisador Léxico (AL), e fornece outra estrutura para o Gerador de Perguntas em LOBAN (GPL). A estrutura que ele recebe é uma sequência de registros de entrada, cada um com uma lexia da pergunta, e todas informações obtidas pelo Analisador Léxico (AL). Essa estrutura tem o formato:

Pergunta = ((Reg-en) (Reg-en)...)
Reg-en = lexia (.) ((significado) (significado)...)

Cada lexia pode ter mais de um significado, devido à ambiguidade introduzida pelos homógrafos. A estrutura do significado depende da categoria da lexia. Poderemos ter as seguintes estruturas do significado:

a) Para as categorias pontuação, conjunção, preposição e advérbio:

   Significado = categoria ( )

b) Para as categorias pronome e artigo:

   Significado = categoria ( ) gênero ( ) número ( )

c) Para as categorias inteiro, numcar, real ou data:

   Significado = categoria ( ) endereços ( )

d) Para a categoria substantivo:

   Significado = categoria ( ) gênero ( ) número ( )
   endereço ( ) traço ( ) marca ( )

e) Para a categoria adjetivo:

   Significado = categoria ( ) gênero ( ) número ( )
   endereço ( ) antecedente ( )
   consequente ( )

f) Para a categoria verbo:

   Significado = categoria ( ) modo ( ) tempo ( )
   pessoa ( ) antecedente ( )
   consequente ( ) condição ( )
   característica ( ) ambiguidade ( )

Após a análise sintática e semântica o ASS fornece para o Gerador de Perguntas em LOBAN, uma estrutura em lista com todas informações necessárias.

A interface IUC permite quatro tipos das perguntas, e a estrutura gerada pelo ASS depende do tipo de pergunta. Para as perguntas "ACHE SE ...", "ACHE A QUANTIDADE ... " e "ACHE A SOMA ...", a estrutura gerada tem a forma:

```
Pergunta = (Tipo pergunta ( ) Entidade/atributo ( )
            Negação ( ) Quantificação ( )
           ((Condição) (Condição) ... ))
```

Para as perguntas "ACHE ..." a estrutura gerada tem a forma:

```
Pergunta = (Tipo pergunta ( ) Entidade ( )
           ((list-atrib ( ) (condição) (condição) ...) E
            (list-atrib ( ) (condição) (condição) ...) ...))
```

O tipo de pergunta corresponde a 1 para pergunta "se", 2 para pergunta "quantidade", 3 para pergunta "soma" e 4 para pergunta "tabela".

Entidade/atributo é a entidade ou atributo ao qual a pergunta se refere.

A negação informa se há uma negação na pergunta ou na condição.

A quantificação indica se a pergunta ou condição se refere a todos, a algum, a nenhum, a um número determinado ou

não tem quantificação.

As condições tem a estrutura:

Condição = Conetivo ( ) Negação ( ) Quantificação ( )
Especificação ( )

E a especificação tem as estruturas:

Especificação = Endereço ( ) Comparador ( ) Valor ( )
OU Endereço ( ) = Endereço ( ) Elo ( ) = Valor ( )
OU Endereço ( ) Comparador ( ) condição ( )

"List-atrib" é a lista de atributos a recuperar na pergunta. Pode ser vazia, no caso de se recuperar todos atributos da tabela.

O conetivo pode ser "E" ou "OU".

"Endereço" é o endereço codificado na base de dados, conforme está mostrado no Apêndice (3).

Comparador pode ser um dos comparadores $<$, $\leq$, $>$, $\geq$, $=$, $\in$, $\subset$ e $\supset$

Elo é um elemento de ligação entre duas tabelas, que permite selecionar ocorrências de atributos de uma entidade em uma tabela, a partir de condições em outra ligada pelo elo. Na figura do Apêndice (3) os elos estão representados pelos traços inferiores.

Na última forma de especificação há um "embutimento" de condição dentro de condição, permitida na IUC. Exemplo:

ACHE O NOME DOS ALUNOS QUE NASCERAM NA DATA DE NASCIMENTO DO ALUNO 5.

Neste exemplo, "do aluno 5" é uma condição embutida que deve ser avaliada. Nós podemos ter vários níveis de embutimento.

VIII.4. O ANALISADOR SINTÁTICO E SEMÂNTICO - ASS

No capítulo (IV) foi descrita a gramática da linguagem IUC. Neste capítulo não vamos repetir a especificação da linguagem aceita pelo ASS, mas sim apresentar a análise, e como ela produz a estrutura para o Gerador de Perguntas em LOBAN (GPL).

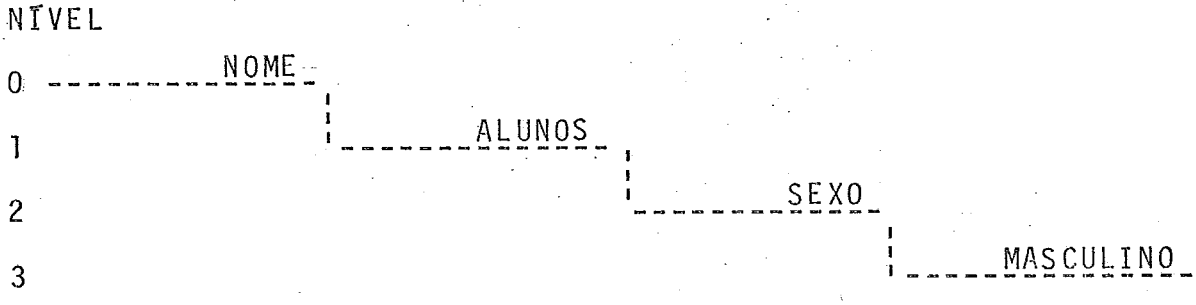
O ASS está detalhado nos gráficos do Apêndice (6). Esses gráficos foram montados utilizando a sintaxe da RTA, para se obter diretamente deles a programação. Cada aresta deve pertencer a um tipo de aresta definido na sintaxe. Os testes de condições assim como as ações e pré-ações, são mostrados em cada gráfico para facilitar a compreensão do gráfico.

Na medida em que o ASS analisa uma pergunta, ele monta uma árvore sintática da sentença, monta a pergunta para o GPL, e monta uma estrutura para manter o contexto de referência e resolver ambiguidades.

A árvore é montada no "passeio" do ASS. Como este é um procedimento similar ao de outros analisadores, não vamos detalhá-lo.

A estrutura para manter o contexto é construida em níveis a partir do nível zero. Cada vez que o ASS entra em um GADJ, em um GADV, em uma "ocorrência" após substantivo, em uma ocorrência que substitui substantivo (neste caso em dois níveis), em um objeto de verbo transitivo com sujeito expresso, o nível desce de 1 (um). Cada vez que o ASS sai dessas situações, o nível aumenta de 1 (um). Em cada um desses níveis há apenas um substantivo, ou seu substituidor, ou uma comparação. Vejamos um exemplo:

ACHE O NOME DOS ALUNOS DO SEXO MASCULINO ...



Essa estrutura é usada para resolver elipse de SN quando há a repetição de condições, e é a base para obter a estrutura de saída. Todas palavras dessa estrutura tem endereços na base de dados, e são o elemento central das perguntas em LOBAN.

O AAS coloca nessa estrutura as palavras com seus endereços, e com base nas características semânticas da base de dados e na sintaxe do Português, seleciona os endereços que serão passados ao GPL. No exemplo acima a entidade da pergunta é ALUNOS, portanto os endereços só podem se referir às tabelas 1, 5 e 6 (ver Apêndice 3). NOME tem no dicionário o endereço 1.2, e é o atributo a recuperar. ALUNOS é a entidade do arquivo 1,

e é atributo nos endereços 1.1, 1.2, 5.4 e 6.1. SEXO é atributo no endereço 1.3. MASCULINO é ocorrência de atributo no endereço 1.3. Nessa pergunta está verificada a semântica da base de dados. No apêndice 8 há exemplos completos de tradução de frases.

Para gerar a estrutura para o GPL, o AAS além da estrutura acima referida, utiliza os seguintes procedimentos:

a) Tipo de pergunta: é obtido no início da análise sintática.

b) A entidade ou atributo: é obtido no início da análise sintática.

c) A lista de atributos: é obtida do nível zero da estrutura em níveis.

d) A negação: é obtida da análise sintática pela presença na oração de advérbio de negação, ou por determinante que indica inexistência (nenhum, ninguém). A pergunta ou condição é negativa também no caso de existirem as duas formas de negação.

e) A quantificação: é obtida do determinante, pela análise de seu significado. Ele pode estar determinando "todos", "algum", "o", ou uma determinada quantidade.

f) O conectivo: é obtido diretamente das conjunções coordenadas "E" ou "OU", das adversativas transformadas em "E", das subordinadas transformadas em "E" (se não existir uma aditiva ou alternativa expressa), e no caso de coordenação assindética, pela substituição da vírgula ou ponto e vírgula por "E", ou por "OU" no caso de aparecer explicitamente uma alternativa no fim da sequência de condições separadas por pontuação.

g) O endereço: é obtido do dicionário, e selecionado utilizando-se as estrutura de níveis e a semântica da base de dados.

h) O comparador: é obtido diretamente do verbo (conter, existir), da locução adjetiva de comparação(>, ≥, <, ≤, =, NÃO =), e da verificação do que aponta o endereço, se uma construção ou um conjunto de construçcoes (∈ , ⊂ , ⊃).

i) O elo: é obtido com o endereço.

j) O valor: é obtido da análise sintática e da estrutura de níveis.

Finalmente vamos detalhar aqui a avaliação de substituidor do substantivo, isto é, pronome substantivo ou numeral.

O substituidor é avaliado no contexto segundo as seguintes regras:

a) Pronome relativo: refere-se ao substantivo que o precede em nível superior.

b) Numeral: refere-se a um conjunto de entidades ou atributos com cardinalidade igual ao valor do numeral. Esse conjunto pode estar no próprio contexto de avaliação ou em contexto superior. É escolhido o primeiro que coincide (também em gênero) a partir do próprio contexto, e subindo para os contextos superiores.

c) Pronome indefinido: refere-se a entidade ou atributo que ocorre no mesmo contexto ou contexto superior, e concorda em gênero e número com o substituido. É avaliado a partir do nível do contexto atual.

d) Pronome pessoal: refere-se a entidade ou atributo que o antecede, e com o qual concorda em gênero e número. A avaliação é iniciada também a partir do nível atual do contexto.

IX. A GERAÇÃO DE PERGUNTAS EM LOBAN

IX.1. INTRODUÇÃO

No capítulo (VIII) vimos como o Analisador Sintático e Semântico - ASS produz a estrutura, a partir da qual o Gerador de Perguntas em LOBAN-GPL gera a pergunta em LOBAN. Neste capítulo vamos ver como o GPL funciona.

Considera-se que todas ambiguidades e elipses foram resolvidas, portanto o GPL tem todas informações necessárias à tradução final.

A pergunta em LOBAN deve ser apresentada dentro de instrução autônoma, e deve endereçar os pontos das construções a representar na resposta, assim como os campos onde serão inseridas as representações dessas construções. Para facilidade de apresentação não vamos incluir o endereço de campo completo, mas apenas uma referência a CC (campo corrente). Para detalhar o endereço de campo seria necessário detalhar as estruturas permitidas pela interface LOBAN para a entrada e saída, tal como foi feito para as estruturas na base de dados, além da estrutura do endereço de campo. Como isto foge muito do escopo do trabalho e sua omissão não prejudica a compreensão do GPL, decidimos não detalhar o endereço de campo, e como já foi dito, representá-lo por CC.

Algumas perguntas podem ser respondidas pela interface exclusivamente com as informações que constam na base de dados. Outras no entanto necessitam uma "área de trabalho" pa

ra acumular nessa área de trabalho as informações obtidas na base de dados, e posteriormente representar a construção da área de trabalho na saída. As áreas de trabalho necessárias são uma tabela, e um ítem numérico. Essas áreas são criadas pela IUC no início de cada sessão de trabalho, e destruidas no fim.

As instruções para criar essas áres são:

```
INCLUIR TAB:= VAZIO EM AUX
INCLUIR ITEM:= 0    EM AUX
```

As instruções para destruir são:

```
EXCLUIR DE AUX.TAB
EXCLUIR DE AUX.ITEM
```

Toda área de trabalho do usuário é mantida pela interface LOBAN em um acervo auxiliar, identificado por AUX.

IX.2. A SINTAXE DAS PERGUNTAS EM LOBAN

Para compreender o processo de tradução final para LOBAN, vamos apresentar a sintaxe das perguntas em LOBAN.

Há quatro tipos de perguntas, e cada qual tem uma estrutura diferente em LOBAN. Vejamos essas estruturas:

a) Pergunta do tipo "ACHE SE ..."

Em LOBAN essa pergunta tem o formato:

```
FAZER EM CASO
     <obter valbool>   REPRESENTAR 'SIM' EM CC
      CONTRARIO        REPRESENTAR 'NÃO' EM CC
```

b) Pergunta do tipo "ACHE A QUANTIDADE ..."

Em LOBAN tem o formato

```
REPRESENTAR CONT <end ponto> EM CC
```

c) Pergunta do tipo "ACHE A SOMA ..."

Essa pergunta em LOBAN tem o formato:

```
SUBSTITUIR EM AUX.ITEM POR 0
FAZER PARA CADA PONTO <end ponto>
     (SUBSTITUIR EM AUX.ITEM POR
           C AUX.ITEM + V PC)
REPRESENTAR C AUX.ITEM EM CC
```

d) Pergunta do tipo "ACHE ..."

Essa pergunta é a mais complexa. Por problemas de eficiência na busca na base de dados, ela foi subdividida em sete perguntas conforme o tipo da tabela a recuperar, a quantidade de tabelas, a existência ou não de condições para seleção de tuplas ou ligações, e a quantidade de atributos. Vejamos então para cada caso a estrutura da resposta:

d1) Recuperar tabela relacional:

```
REPRESENTAR C <end tarel> EM CC
```

d2) Recuperar tabela ligacional:

```
REPRESENTAR DESAGRUP C <end talig> EM CC
```

d3) Recuperar tuplas de uma tabela relacional:

```
REPRESENTAR COLEC <end tupla> EM CC
```

d4) Recuperar ligações de uma tabela ligacional:

```
REPRESENTAR DESAGRUP COLEC <end ligação> EM CC
```

d5) Recuperar atributos de uma tabela relacional:

```
REPRESENTAR ESTREIT COLEC <end tupla>
     PARA <atributo>,,,  EM CC
```

d6) Recuperar atributos de uma talig, com seleção sobre as ligações:

```
REPRESENTAR ESTREIT DESAGRUP COLEC <end ligação>
     PARA <atributo>,,, EM CC
```

d7) Demais casos:

```
SUBSTITUIR EM AUX.TAB POR VAZIO
FAZER PARA CADA <end ponto>
    (FAZER PARA CADA <end ponto>
       (  :
          (INCLUIR COMPOR
               (<obter nome>:=  C <end atrib>
                          :
                                                )
           EM AUX.TAB) ... ))
REPRESENTAR AUX.TAB EM CC
```

Vamos ver agora o restante da sintaxe:

<end ponto>::= { <id ponto> | [<id ponto>]<termo avaliação end ponto>... }

[((=<obter marca>))]

<id ponto>::= ACTRAB|AUX|PC [/<obter marca>]
|PX[/<obter marca>]

<termo avaliação end ponto>::= [((=<obter marca>))]<cri ponto imed>.

<cri ponto imed>::= { <obter valbool> | <lit nome> }

<obter valbool>::= <termo elementar de valbool>
|NÃO <obter valbool>
|<obter valbool> {E|OU}<obter valbool>
|PARA TODO <end ponto>((<obter valbool>))
|EXIST <end ponto>

<termo elementar de valbool>::=
C <end ponto>= C <end ponto>

| { COLEC <end ponto> | {<literal>;;; } } {⊆ | ⊇ }COLEC <end ponto>

| C <end ponto> ∈ COLEC <end ponto>

| C <end ponto> {=|<|≤|>|≥} <end ponto>
[EM ORDEM DATA]

| CONT <end ponto>= { CONT <end ponto> | <literal> }

IX.3. O ALGORITMO DO GERADOR DE PERGUNTAS EM LOBAN-GPL

O algoritmo do GPL é apresentado no Apêndice (7). Ele inicialmente se subdivide em quatro, um para cada tipo de pergunta. O quarto algoritmo se subdivide em sete, conforme as estruturas de resposta mostradas no ítem anterior. Essa subdivisão permite estabelecer para cada pergunta um formato bem definido, restando avaliar os endereços, os obter valor booleano, e os atributos no caso de seleção de atributos.

Na avaliação dos endereços e obter valor booleano o algoritmo do GPL verifica se a expressão exige um quantificador existencial, um quantificador universal, uma comparação ou uma contagem sobre pontos.

A interface LOBAN tem estruturas de endereço que permitem critérios de seleção no nível de ocorrência de coleções. Nas tabelas relacionais esse nível é no nível das tuplas. Nas tabelas ligacionais há dois níveis de seleção: um para as ligações, e outro para as tuplas nos ligados. A figura no Apêndice (3) mostra essa estrutura de endereços.

X. CONCLUSÕES

X.1. REVISÃO

O trabalho de especificação está concluido, e definiu uma interface de fácil uso para um usuário casual. Há limitações impostas ao usuário, e há limitações na linguagem natural.

A interface permite condições embutidas. Sob o ponto de vista funcional não há limite de nível de embutimento. Sob o ponto de vista de uso, deverão ser feitas limitações devido ao desempenho do usuário e do sistema.

Uma limitação importante da interface e só permitir condições comparativas sobre dados diretamente existentes na base de dados. O sistema não realiza inferencias nem deduções, pois isto exige um nível semântico que foge do objetivo da tese.

Um problema que ainda deve ser considerado é o escopo das conjunções. O sistema resolve linearmente as conjunções, mas não cria nenhum mecanismo de precedência na avaliação das conjunções. Desta forma, na avaliação de expressões com conjunções mantêm-se a precedência da linguagem natural, isto é, primeiro as conjunções aditivas depois as conjunções alternativas. Se o usuário em sua pergunta expressou outra precedência, seja por vírgulas, seja pelo uso de conjunções coordenadas e subordinadas, seja por especificações nos determinantes das orações, o sistema não considera essa precedência e a resposta pode não corresponder ao que o usuário deseja.

A análise sintática funciona bem, mas há uma inadequação de terminologia no sintagma preposicional, no grupo adjetival e no grupo adverbial. Essa questão já foi levantada no corpo da tese, ao discutirmos o sintagma preposicional. Exige no entanto mais estudo, para evitar que o nome dos componentes leve a confusão na análise sintática.

Sob o ponto de vista de adequação da interface a um usuário casual, poder e desempenho, não podemos fazer uma avaliação maior. Esta avaliação depende de implementar e usar a interface. Esperamos no entanto que ela tenha um índice próximo a 80% de perguntas corretamente respondidas.

Como conclusão deste trabalho, temos uma solução para o problema de projetos de interfaces para SBD em linguagem natural, independentes do contexto da aplicação. É necessário agora implementar a interface, e verificar a adequação da solução e suas limitações práticas.

X.2. PROPOSTAS FUTURAS

Já estou trabalhando em um mecanismo para definir o escopo das conjunções. Se ele se mostrar adequado, será introduzido na interface.

Para implementar a interface temos vários problemas. O mais sério é não existir previsão para uma implementação de LOBAN em curto prazo, que suporte a interface IUC. Se isto realmente não ocorrer, uma solução será redesenhar a interface para o Sistema de Banco de Dados COPPEREL. Esta tarefa é simples,

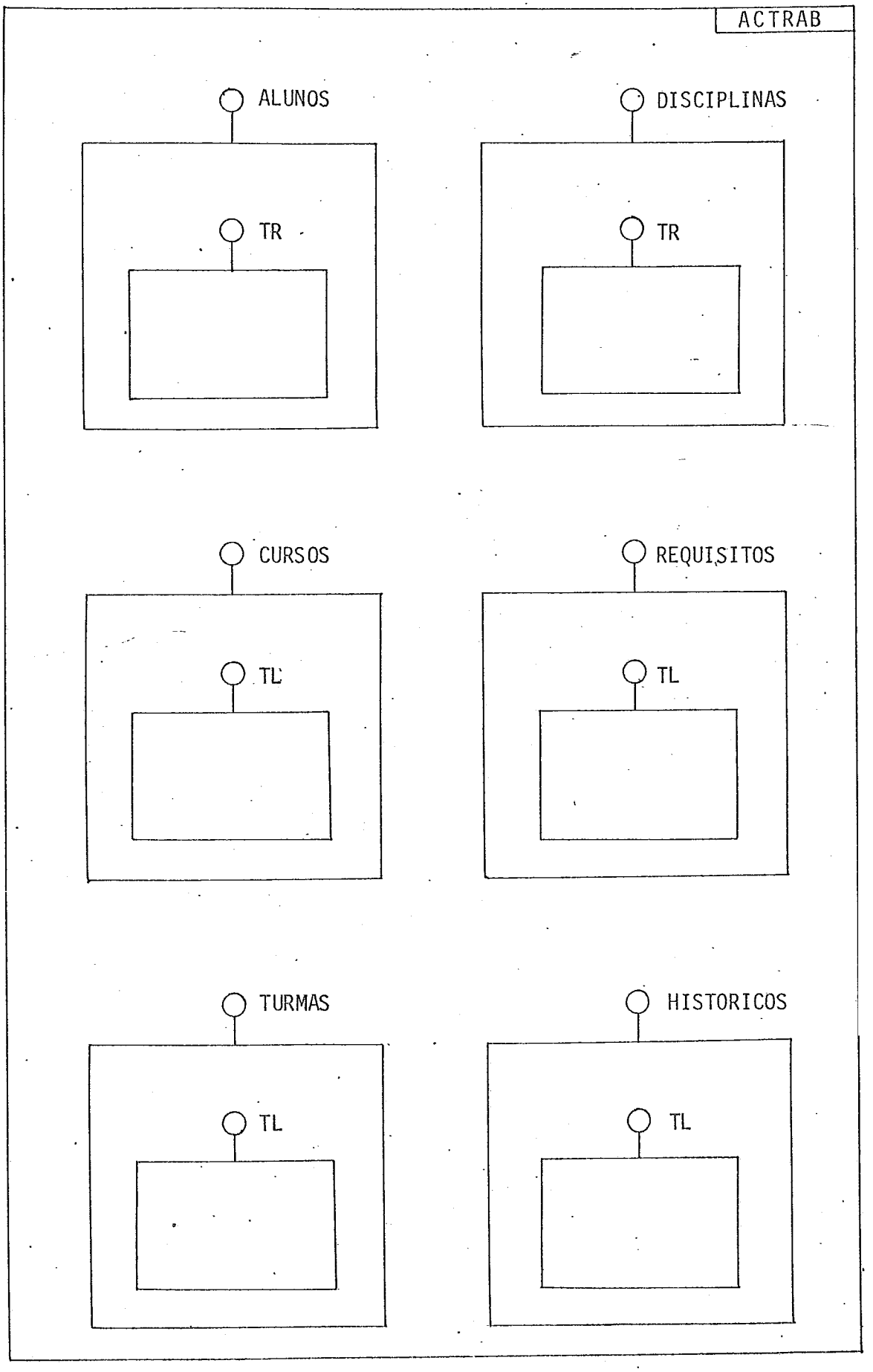
pois LOBAN é muito mais complexo que COPPEREL, portanto a adaptação significará simplificações na interface. A grosso modo, a adaptação exige apenas eliminar as estruturas e algoritmos utilizados para as tabelas ligacionais e manter as estruturas e algoritmos para as tabelas relacionais.

Uma proposta interessante para trabalhos futuros refere-se à semântica da base de dados de SBD, e como utilizar essa semântica em interfaces como a IUC. Na tese desenvolvemos a semântica da estrutura e das chaves em LOBAN. Não utilizamos no entanto as "conexões", que talvez permitam conclusões práticas interessantes. Além disso o trabalho da tese não esgotou a semântica da estrutura e chaves. Pelo contrário, há muito o que fazer nesse sentido.

Outra proposta refere-se à análise sintática do Portugues para a Linguística Computacional. Este campo está muito incipiente, e no desenvolvimento da tese fui obrigado a utilizar resultados obtidos para o Ingles, Espanhol e Frances. Já foram realizados trabalhos interessantes nessa área, mas o maior problema é a descontinuidade desses trabalhos.

APÊNDICE 1

A BASE DE DADOS


ACTRAB
ALUNOS
TR
DISCIPLINAS
TR
CURSOS
TL
REQUISITOS
TL
TURMAS
TL
HISTORICOS
TL

ALUNOS

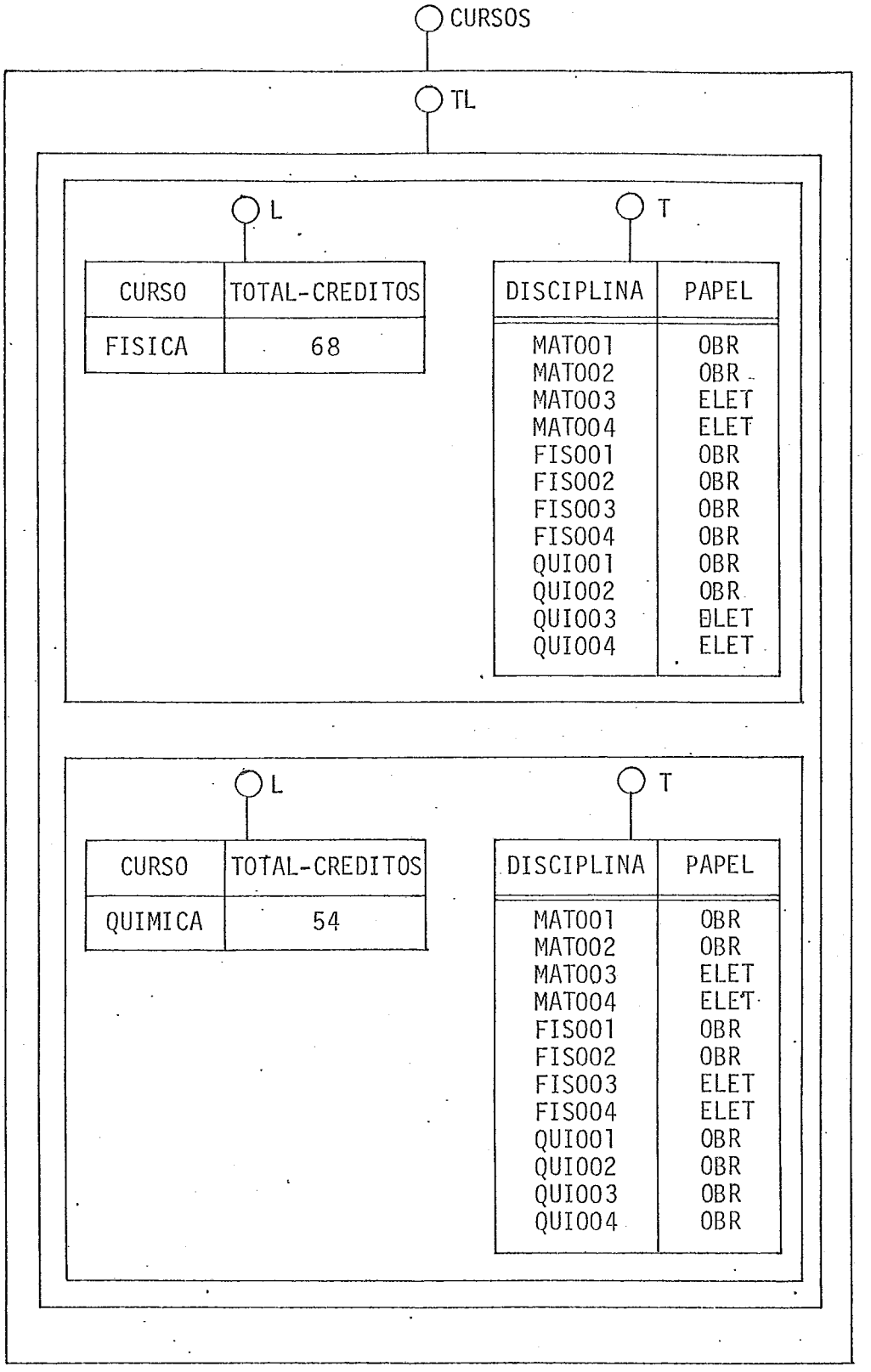
TR

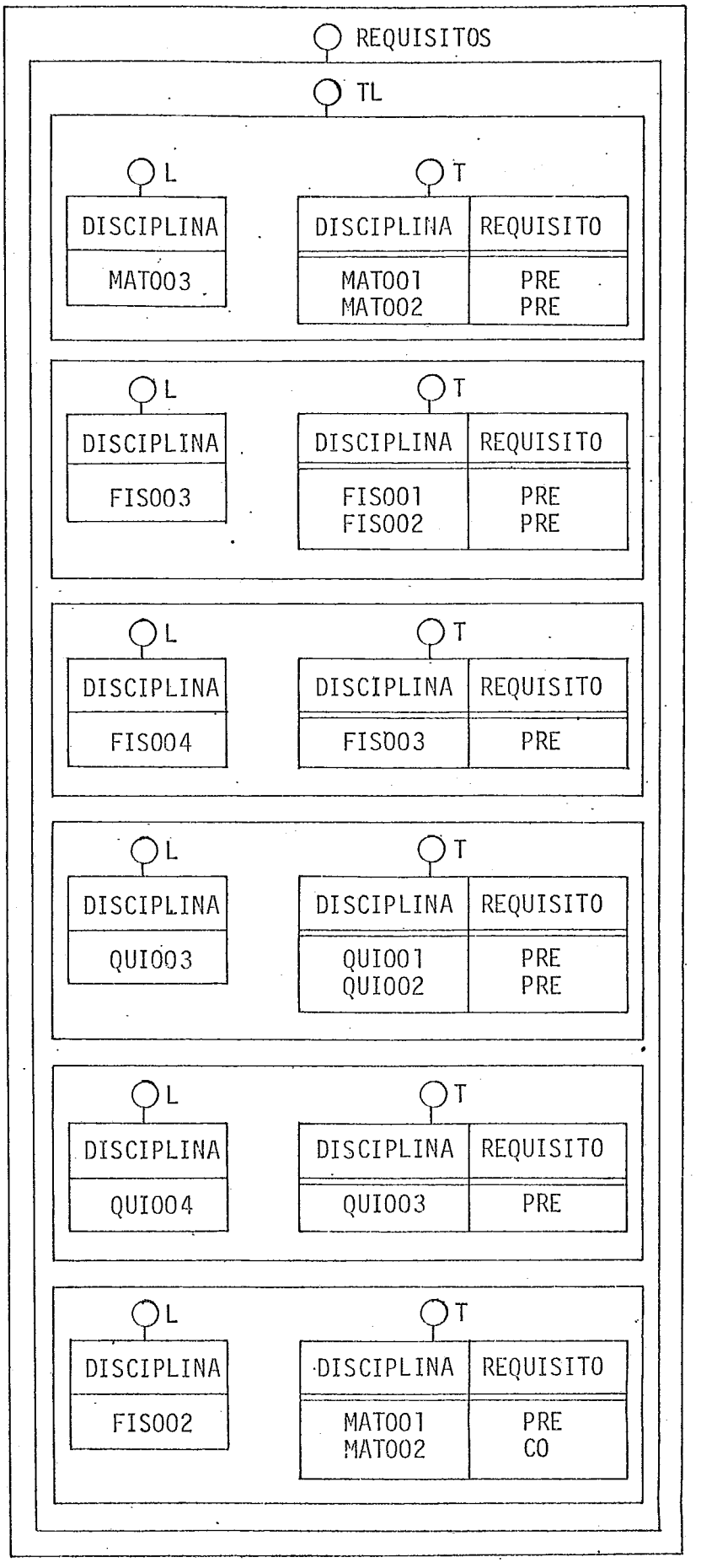
MATRICULA	NOME	SEXO	DATA-NASC	CURSO	DATA-INGRESSO
1	JOSE	MASC	1-1-60	QUIMICA	1-3-79
2	ALCIR	MASC	3-5-59	FISICA	1-3-79
3	LUIZ	MASC	7-11-60	FISICA	1-3-79
4	ROSA	FEM	26-9-60	QUIMICA	1-3-79
5	CIRA	FEM	25-12-60	FISICA	1-3-79
6	CLAUDIO	MASC	29-10-60	QUIMICA	1-3-79
7	BETTY	FEM	12-7-59	QUIMICA	1-3-80
8	VERA	FEM	3-8-60	FISICA	1-3-80
9	JANO	MASC	7-5-58	FISICA	1-3-80
10	ROMÃO	MASC	2-1-50	FISICA	1-3-80
11	ROMUALDO	MASC	7-8-55	QUIMICA	1-3-80
12	ESTER	FEM	9-2-57.	FISICA	1-3-80
13	FRANCISCO	MASC	10-4-56	QUIMICA	1-3-80
14	MARIA	FEM	4-6-54	QUIMICA	1-3-80

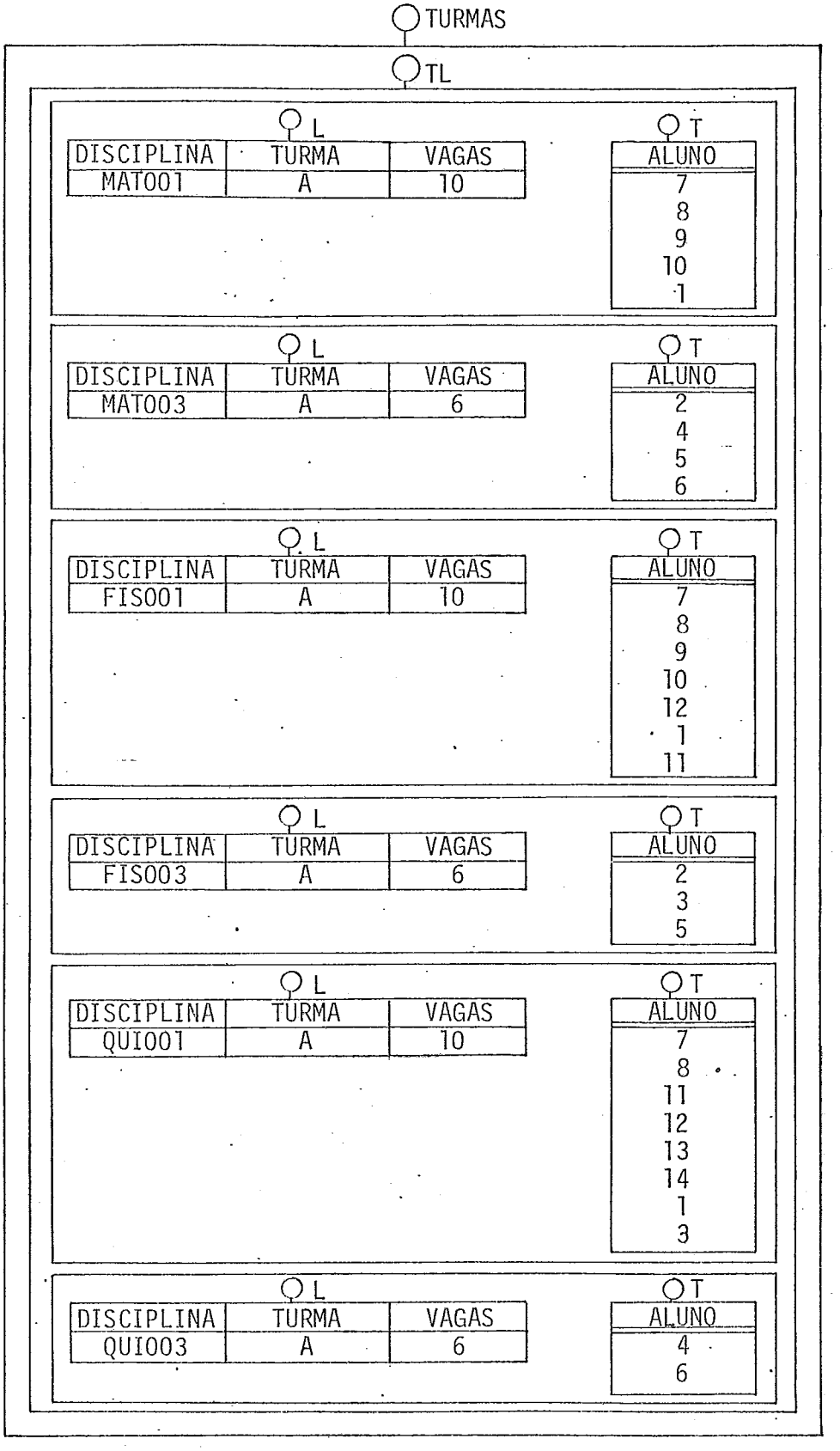
DISCIPLINAS

TR

CODIGO	TITULO	CARGA-HORARIA		CREDITOS	PERIODO DE APRESENTAÇÃO
		TEORICA	PRATICA		
MAT001	MATEMATICA I	6	0	6	1
MAT002	MATEMATICA II	6	0	6	2
MAT003	MATEMATICA III	6	0	6	1
MAT004	MATEMATICA IV	6	0	6	2
FIS001	FISICA I	4	4	8	1
FIS002	FISICA II	4	4	8	2
FIS003	FISICA III	4	4	8	1
FIS004	FISICA IV	4	4	8	2
QUI001	QUIMICA I	3	3	6	1
QUI002	QUIMICA II	3	3	6	2
QUI003	QUIMICA III	3	3	6	1
QUI004	QUIMICA IV	3	3	6	2


CURSOS
TL
L
T
CURSO
TOTAL-CREDITOS
FISICA
68
DISCIPLINA
PAPEL
MAT001 OBR
MAT002 OBR
MAT003 ELET
MAT004 ELET
FIS001 OBR
FIS002 OBR
FIS003 OBR
FIS004 OBR
QUI001 OBR
QUI002 OBR
QUI003 ELET
QUI004 ELET
L
T
CURSO
TOTAL-CREDITOS
QUIMICA
54
DISCIPLINA
PAPEL
MAT001 OBR
MAT002 OBR
MAT003 ELET
MAT004 ELET
FIS001 OBR
FIS002 OBR
FIS003 ELET
FIS004 ELET
QUI001 OBR
QUI002 OBR
QUI003 OBR
QUI004 OBR


REQUISITOS
TL
L
DISCIPLINA
MAT003
T
DISCIPLINA
REQUISITO
MAT001
MAT002
PRE
PRE
L
DISCIPLINA
FIS003
T
DISCIPLINA
REQUISITO
FIS001
FIS002
PRE
PRE
L
DISCIPLINA
FIS004
T
DISCIPLINA
REQUISITO
FIS003
PRE
L
DISCIPLINA
QUI003
T
DISCIPLINA
REQUISITO
QUI001
QUI002
PRE
PRE
L
DISCIPLINA
QUI004
T
DISCIPLINA
REQUISITO
QUI003
PRE
L
DISCIPLINA
FIS002
T
DISCIPLINA
REQUISITO
MAT001
MAT002
PRE
CO


TURMAS
TL
L
DISCIPLINA | TURMA | VAGAS
MAT001 | A | 10
T
ALUNO
7
8
9
10
1
L
DISCIPLINA | TURMA | VAGAS
MAT003 | A | 6
T
ALUNO
2
4
5
6
L
DISCIPLINA | TURMA | VAGAS
FIS001 | A | 10
T
ALUNO
7
8
9
10
12
1
11
L
DISCIPLINA | TURMA | VAGAS
FIS003 | A | 6
T
ALUNO
2
3
5
L
DISCIPLINA | TURMA | VAGAS
QUI001 | A | 10
T
ALUNO
7
8
11
12
13
14
1
3
L
DISCIPLINA | TURMA | VAGAS
QUI003 | A | 6
T
ALUNO
4
6

HISTORICOS

TL

L

ALUNO
1

T

DISCIPLINA	CREDITOS	EPOCA		NOTA
		ANO	PERIODO	
FIS001	0	79	1	3
FIS002	0	79	2	1
MAT001	0	79	1	2
MAT002	0	79	2	0,5
QUI001	0	79	1	4,1
QUI002	0	79	2	2

L

ALUNO
2

T

DISCIPLINA	CREDITOS	EPOCA		NOTA
		ANO	PERIODO	
FIS001	8	79	1	7,5
FIS002	8	79	2	9
MAT001	6	79	1	10
MAT002	6	79	2	8,2
QUI001	6	79	1	7,9
QUI002	6	79	2	8

L

ALUNO
3

T

DISCIPLINA	CREDITOS	EPOCA		NOTA
		ANO	PERIODO	
FIS001	8	79	1	5
MAT001	6	79	1	7
FIS002	8	79	2	9

L

ALUNO
4

T

DISCIPLINA	CREDITOS	EPOCA		NOTA
		ANO	PERIODO	
FIS001	8	79	1	7,8
FIS002	8	79	2	9,2
MAT001	6	79	1	8
MAT002	6	79	2	8
QUI001	6	79	1	9
QUI002	6	79	2	10

L

ALUNO
5

T

DISCIPLINA	CREDITOS	EPOCA		NOTA
		ANO	PERIODO	
FIS001	8	79	1	7
MAT001	6	79	1	9
QUI001	6	79	1	8
FIS002	8	79	2	7
MAT002	6	79	2	8
QUI002	6	79	2	10

L

ALUNO
6

T

DISCIPLINA	CREDITOS	EPOCA		NOTA
		ANO	PERIODO	
FIS001	8	79	1	7
MAT001	6	79	1	8
QUI001	6	79	1	9

APÊNDICE 2

A COERÊNCIA DA BASE DE DADOS

TIPOS DE ÁTOMOS

```
AT-MAT: TIPO DE INTEIRO TAL QUE
            C OCOR > 0
        E C OCOR  < 999999
AT-SEX: TIPO DE SIGLA TAL QUE
        C OCOR ∈ {"MASCULINO", "FEMININO"}
AT-TIT-CURSO: TIPO DE SIGLA TAL QUE
        C OCOR ∈ {"QUIMICA", "FISICA"}
AT-CARGA-HOR: TIPO DE INTEIRO TAL QUE
        C OCOR < 7
AT-CRED: TIPO DE INTEIRO TAL QUE
        C OCOR < 9
AT-APRES: TIPO DE INTEIRO TAL QUE
        C OCOR = 1
        OU C OCOR = 2.
AT-TOT-CRED: TIPO DE INTEIRO TAL QUE
        C OCOR > 50
AT-PAPEL: TIPO DE SIGLA TAL QUE
        C OCOR ∈ {"OBRIGATORIA", "ELETIVA"}
AT-REQUIS: TIPO DE SIGLA TAL QUE
        C OCOR ∈ {"PRE-REQUISITO", "CO-REQUISITO"}
AT-TURMA: TIPO DE LETRA
AT-VAGAS: TIPO DE INTEIRO TAL QUE
        C OCOR > 5
        E C OCOR < 31
```

```
AT-ANO: TIPO DE INTEIRO TAL QUE
        C OCOR > 78
        E C OCOR < 99
AT-NOTA: TIPO DE REAL TAL QUE
        C OCOR ≤ 10
```

TIPOS DE TUPLAS

```
TUP-NOME-AL: TIPO DE NUMCAR TAL QUE
        COMPOS
        POR 1: 30
        DE LETRA ∪ BRANCO
TUP-DATA-NASC: TIPO DE DATA TAL QUE
        C OCOR.ANO > 40
TUP-DATA-INGRESS: TIPO DE DATA TAL QUE
        C OCOR.ANO > 78
TUP-EPOCA: TIPO DE TUPLA TAL QUE
        COMPOS
        ANO → AT-ANO
        PERIODO → AT-APRES
TUP-COD-DIS: TIPO DE NUMCAR TAL QUE
        COMPOS
        POR 6
        DE CARACTERE
```

```
TUP-TIT-DIS: TIPO DE NUMCAR TAL QUE
        COMPOS
        POR 8:15
        DE LETRA ∪ BRANCO
TUP-CARGA-HOR: TIPO DE TUPLA TAL QUE
        COMPOS
        TEORICA → AT-CARGA-HOR
        PRATICA → AT-CARGA-HOR
TUP-ALUNO: TIPO DE TUPLA TAL QUE
        COMPOS
        MATRICULA → AT-MAT
        NOME → TUP-NOME-AL
        SEXO → AT-SEX
        DATA-NASC → TUP-DATA-NASC
        CURSO → AT-TIT-CURSO
        DATA - INGRESSO → TUP-DATA-INGRES
TUP-DISCIPLINA: TIPO DE TUPLA TAL QUE
        COMPOS
        CODIGO → TUP-COD-DIS
        TITULO → TUP-TIT-DIS
        CARGA-HORARIA → TUP-CARGA-HOR
        CREDITOS → AT-CRED
        PERIODO-DE-APRESENTAÇÃO → AT-APRES
TUP-LI-CUR: TIPO DE TUPLA TAL QUE
        COMPOS
        CURSO → AT-TIT-CURSO
        TOTAL-CREDITOS → AT-TOT-CRED
```

```
TUP-ADO-CUR: TIPO DE TUPLA TAL QUE
        COMPOS
        DISCIPLINA → TUP-COD-DIS
        PAPEL → AT-PAPEL
TUP-LI-PRE: TIPO DE TUPLA TAL QUE
        COMPOS
        DISCIPLINA → TUP-COD-DIS
TUP-ADO-PRE: TIPO DE TUPLA TAL QUE
        COMPOS
        DISICPLINA → TUP-COD-DIS
        REQUISITO → AT-REQUIS
TUP-LI-TURMA: TIPO DE TUPLA TAL QUE
        COMPOS
        DISCIPLINA → TUP-COD-DIS
        TURMA → AT-TURMA
        VAGAS → AT-VAGAS
TUP-AL: TIPO DE TUPLA TAL QUE
        COMPOS
        ALUNO → AT-MAT
TUP-ADO-HIST: TIPO DE TUPLA TAL QUE
        COMPOS
        DISCIPLINA → TUP-COD-DIS
        CREDITOS → AT-CRED
        EPOCA → TUP-EPOCA
        NOTA → AT-NOTA
```

TIPOS DE TABELAS RELACIONAIS

```
TAR-AL: TIPO DE TAREL TAL QUE
        COMPOS
        TUP-ALUNO
TAR-DISC: TIPO DE TAREL TAL QUE
        COMPOS
        TUP-DISCIPLINA
TAR-CURSO:  TIPO DE TAREL TAL QUE
        COMPOS
        TUP-ADO-CUR
TAR-REQUIS: TIPO DE TAREL TAL QUE
        COMPOS
        TUP-ADO-PRE
TAR-TURMA: TIPO DE TAREL TAL QUE
        COMPOS
        TUP-AL
TAR-HIST: TIPO DE TAREL TAL QUE
        COMPOS
        TUP-ADO-HIST
```

TIPOS DE LIGAÇÕES

LIG-CURSO: TIPO DE LIGAÇÃO TAL QUE

COMPOS

L → TUP-LI-CUR

T → TAR-CURSO

LIG-REQUIS: TIPO DE LIGAÇÃO TAL QUE

COMPOS

L → TUP-LI-PRE

T → TAR-REQUIS

LIG-TURMA: TIPO DE LIGAÇÃO TAL QUE

COMPOS

L → TUP-LI-TURMA

T → TAR-TURMA

LIG-HIST: TIPO DE LIGAÇÃO TAL QUE

COMPOS

L → TUP-AL

T → TAR-HIST

TIPOS DE TABELAS LIGACIONAIS

```
TALI-CURSO: TIPO DE TALIG TAL QUE
        COMPOS
        LIG-CURSO
TALI-REQUIS: TIPO DE TALIG TAL QUE
        COMPOS
        LIG-REQUIS
TALI-TURMA: TIPO DE TALIG TAL QUE
        COMPOS
        LIG-TURMA
TALIG-HIST: TIPO DE TALIG TAL QUE
        COMPOS
        LIG-HIST
```

TIPOS DE ARQUIVOS

```
AR-ALUNOS: TIPO DE AREL TAL QUE
        COMPOS
        TR → TAR-AL
AR-DISCIPLINAS: TIPO DE AREL TAL QUE
        COMPOS
        TR → TAR-DISC
AL-CURSOS: TIPO DE ALIG TAL QUE
        COMPOS
        TL → TALI - CURSO
```

AL-REQUIS: TIPO DE ALIG TAL QUE

COMPOS

TL → TALI-REQUIS

AL-TURMAS: TIPO DE ALIG TAL QUE

COMPOS

TL → TALI-TURMA

AL-HIST: TIPO DE ALIG TAL QUE

COMPOS

TL → TALI-HIST

TIPO DE ACERVO

CONTROLE-ACADÊMICO: TIPO DE ACSET TAL QUE

COMPOS

ALUNOS → AR-ALUNOS

DISCIPLINAS → AR-DISCIPLINAS

CURSOS → AL-CURSOS

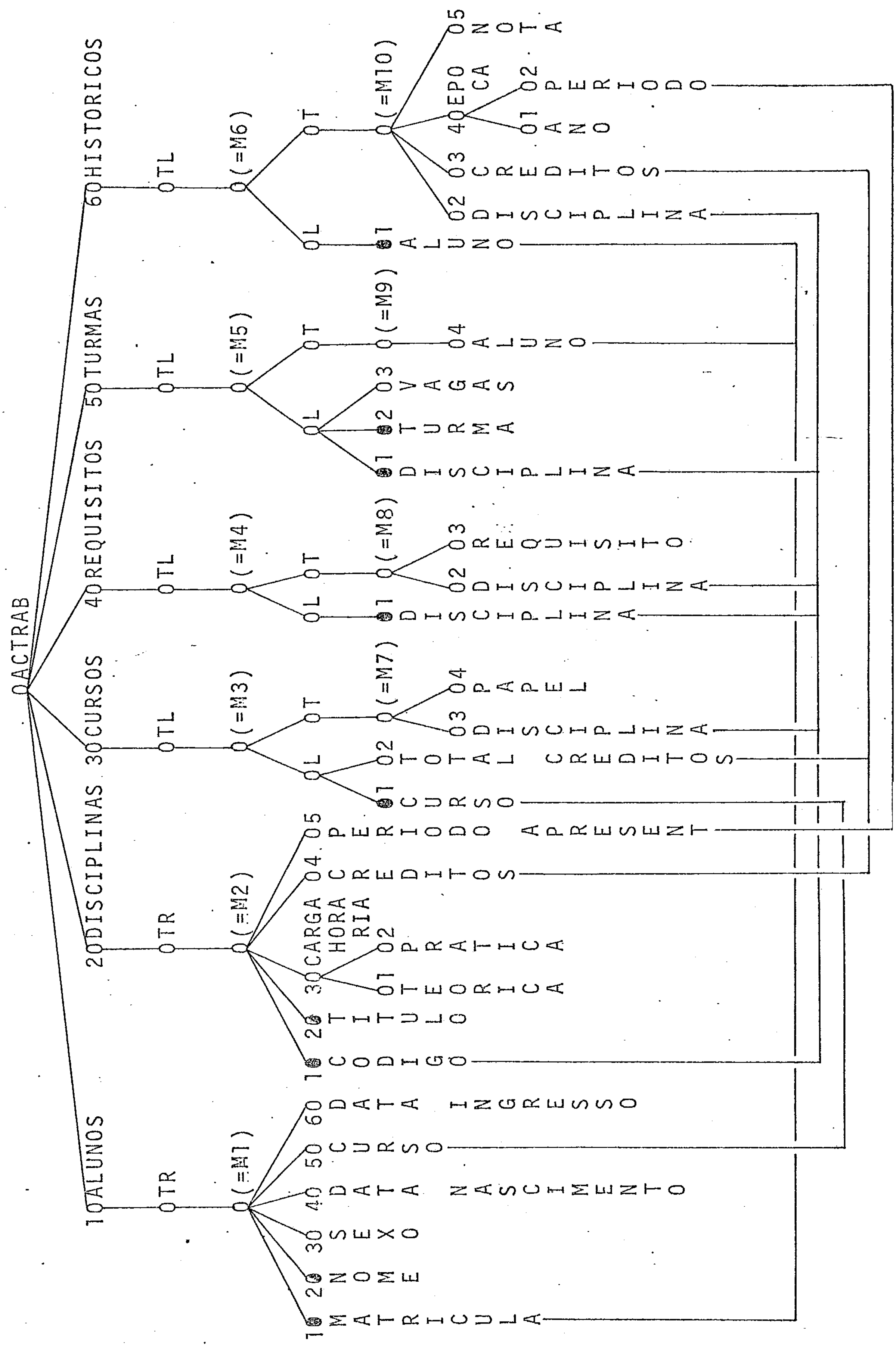
PRE-REQUISITOS → AL-REQUIS

TURMAS → AL-TURMAS

HISTORICOS → AL-HIST

APÊNDICE 3

ENDEREÇOS E TIPOS DE ENTIDADES, ATRIBUTOS E OCORRÊNCIAS


0 ACTRAB
10 ALUNOS
20 DISCIPLINAS
30 CURSOS
40 REQUISITOS
50 TURMAS
60 HISTORICOS
TR
TL
(=M1)
(=M2)
(=M3)
(=M4)
(=M5)
(=M6)
(=M7)
(=M8)
(=M9)
(=M10)
MATRICULA
NOME
SEXO
DATA NASCIMENTO
CURSO
DATA INGRESSO
CODIGO
TITULO
CARGA HORARIA
TEORICA
PRATICA
CREDITOS
PERIODO APRESENT
TOTAL CREDITOS
DISCIPLINA
PAPEL
REQUISITO
TURMAS
VAGAS
ALUNO
EPOCA
ANO
PERIODO
NOTA

APÊNDICE 4

O DICIONÁRIO DA IUC PARA O CONTROLE ACADÊMICO

LEXIAS			SINÔNIMO	CATEGORIA	HOMÓGRAFOS
INICIO		COMPLEMENTO			
BASE	FLEXÕES				
Ø	o			art.def	pr.pess,ocor
Ø	a			art.def	pr.pess,ocor,prep+art.def
Ø	os			art.def	pr.pess
Ø	as			art.def	pr.pess,prep,art.def
U	m			art.indef	card
UM	a			art.indef	card
U	ns			art.indef	
UM	as			art.indef	

LEXIAS			SINÔNIMO	CATEGORIA	HOMÓGRAFOS
INICIO		COMPLEMENTO			
BASE	FLEXÕES				
el	e-1,a-2,es-3,as-4			pr.pess	
Ø	o-1,			pr.pess	art.def,ocor
Ø	a-2			pr.pess	art.def,ocor,prep+art.def
Ø	os-3			pr.pess	art.def,
Ø	as-4			pr.pess	art.def,prep+art.def
Ø	-o,-a,-os,-as			pr.pess	
Ø	-lo,-la,-los,-las		-o,a,-os,-as		
Ø	-no,-na,-nos,-nas		-o,-a,-os,-as		
Ø	e,es			pr.pess	
Ø	-lhe,-lhes			pr.pess	
s	e			pr.pess	subord
est,	e,a,es,as			pr.dem	
ess	e,a,es,as			pr.dem	
aquel	e,a,es,as			pr.dem	prep.+pr.dem
algu	em			pr.indef	
ningu	em			pr.indef	
outr	em		alguem		
algu	m,ns			pr.indef	
algum	a,as			pr.indef	
nenhu	m,ns			pr.indef	
nenhum	a,as			pr.indef	
tod	o,a,os,as			pr.indef	
outr	o,a,os,as			pr.indef	
muit	o,a,os,as		algum		
pouc	o,a,os,as		algum		
vari	os,as		algum		
dema	is			pr.indef	
qualquer	Ø			pr.indef	
qualquer	Ø	um ,uma	todos,todas		
cad	a			pr.indef	
cad	a	um ,uma	todos,todas		
cad	a	qual	todos,todas		
qu	em		alguem que		

qu	e			pr.relat	
Ø	o	qual	que		
cuj	o			pr.relat	
quant	o		que		
ond	e			pr.relat	
aond	e		onde		
dond	e		onde		

LEXIAS			SINÔNIMO	CATEGORIA	HOMÓGRAFOS
INICIO		COMPLEMENTO			
BASE	FLEXÕES				
u	m-1,1			card	art.indef
um	a			card	art.indef
u	ns 1,2			card	
um	as			card	
do	is 1			card	
du	as 2			card	
tr	es			card	
quatr	o			card	
cinc	o			card	
se	is			card	
set	e			card	
oit	o			card	
nov	e			card	
dez	Ø			card	
amb	os	[os,as]	dois		
primeir	o,a,os,as			ord	
segund	o,as,os,as			ord	

LEXIAS			SINÔNIMO	CATEGORIA	HOMÓGRAFOS
INICIO		COMPLEMENTO			
BASE	FLEXÕES				
si	m			adv	
certament	e		Ø		
efetivament	e		Ø		
realment	e		Ø		
co	m	certeza	Ø		
acas	o		Ø		
por	Ø	ventura	Ø		
por	Ø	acaso	Ø		
bastant	e		Ø		
b	em		Ø		
dema	is		Ø		
ma	is			adv	
men	os				
muit	o		Ø		
pouc	o		Ø		
quas	e		Ø		
t	ao		Ø		
qu	ao		Ø		
abaix	o			adv	
acim	a			adv	
al	em		acima, depois		
aqu	em		abaixo, antes		
dentr	o			adv	
for	a			adv	
assi	m		Ø		
b	em		Ø		
ma	l		Ø		
n	ao			adv	
d	e	forma alguma	não		
d	e	modo nenhum	não		
agor	a			adv	
aind	a			adv	
ant	es			adv	

depo	is			adv	
ent	ao		ø		
j	a		agora		
jama	is		nunca		
nunc	a			adv	
outror	a			adv	
sempr	e			adv	
hoj	e			adv	
inclusiv	e			adv	
exclusiv	e			adv	
tamb	em		ø		
respectivament	e			adv	
s	o			adv	
apen	as		so		

LEXIAS			SINONIMO	CATEGORIA	HOMÓGRAFOS
INICIO		COMPLEMENTO			
BASE	FLEXÕES				
Ø	a			prep	art.def,pr. pess.,ocor,
ap	ós			prep	
at	e			prep	
co	m			prep	
d	e			prep	
desd	e			prep	
Ø	em			prep	
entr	e			prep	
par	a			prep	
por	Ø			prep	
s	em			prep	
sobr	e		em		
durant	e		em		
abaix	o	de	menor que		
acim	a	de	maior que		
ant	es	de		prep	
depo	is	de	apos		
Ø	a	partir de	apos		
dentr	o	de	em		
for	a	de	não em		
d	o,a,os,as		de+arf.def		
n	o,a,os,as		em+art.def		
pel	o,a,os,as		por+art.def		
u	m,ns		em+art.indef		
num	a,as		em+art.indef		
du	m,ns		de+art.indef		
dum	a,as		de+art.indef		
del	e,a,es,as		de+pr.pess		
nel	e,a,es,as		em+pr.dem		
dest	e,a,es,as		em+pr.dem		
nest	e,a,es,as		de+pr.dem		
dess	e,a,es,as		em+pr.dem		

ness	e,a,es,as		em+pr.dem		
daquel	e,a,es,as		de+pr.dem		
naquel	e,a,es,as		em+pr.dem		
aquel	e,a,es,as		a +pr.dem		pr.dem
Ø	a,o		a +ar.def		
a	o,s		a +ar.def		
Ø	a,s		a +ar.def		art.def

LEXIAS			SINÔNIMO	CATEGORIA	HOMÓGRAFOS
INICIO		COMPLEMENTO			
BASE	FLEXÕES				
Ø	e			coord	verbo ser
n	em			coord	
assi	m	como	e		
b	em	como	e		
m	as			coord	
por	em		mas		
contud	o		mas		
todavi	a		mas		
n	o	entanto	mas		
entretant	o		mas		
Ø	ou			coord	
or	a		ou		
,			*avaliar	coord	
;			*avaliar	coord	
s	e			subord	
cas	o			subord	
contant	o	que	caso		
salv	o	se	se não		
desd	e	que	caso		
Ø	a	menos que	caso não		
Ø	a	não ser que	caso não		
quand	o			subord	
ant	es	que		subord	
depo	is	que		subord	
at	e	que		subord	
maior	Ø	que, do que		compar	
menor	Ø	que, do que		compar	
melhor	Ø	que, do que	maior que		
pior	Ø	que, do que	menor que		
ta	l	qual	igual a		
com	o			compar	
ma	is	[do que]		compar	
men	os	[do que]		compar	

LEXIAS			SINÔNIMO	CATEGORIA	HOMÓGRAFOS	ANTECEDENTE	CONSEQUENTE
BASE	FLEXÕES	COMPLEMENTO					
MAIOR	Ø,ES			adj		numérico	numérico
MAIOR	Ø,ES	QUE		compar		numérico	numérico
MENOR	Ø,ES			adj		numérico	numérico
MENOR	Ø,ES	QUE		compar		numérico	numérico
MELHOR	Ø,ES		MAIOR				
PIOR	Ø,ES		MENOR				
SUPERIOR	Ø,ES	A	MAIOR QUE				
INFERIOR	Ø,ES	A	MENOR QUE				
ANTERIOR	Ø,ES	A		adj		tempo	tempo
POSTERIOR	Ø,ES	A		adj		tempo	tempo
ULTERIOR	Ø,ES		ANTERIOR				
IGUA	L,IS	[A]		adj			
DIFERENT	E,ES	DE		adj			
MESM	O,A,OS,AS	QUE		adj			
COM	O			compar			
MA	IS	[DO] QUE		compar			
MEN	OS	[DO] QUE		compar			
DESIGUA	L,IS	A	DIFERENTE DE				
ANALOG	O,OS	A	IGUAL				
SEMELHANT	E,ES	A	IGUAL				
HOM	EM,ENS		DE SEXO MASCULINO				
MULHER	Ø,ES		DE SEXO FEMININO				
MA	IS	ALTO	MAIOR				
MA	IS	BAIXO	MENOR				
MA	IS	VELHO[DO]QUE	ANTERIOR A				
MA	IS	MOÇO[DO]QUE	PORTERIOR A				
MA	IS	ANTIGO[DO]QUE	ANTERIOR A				

FLEXÕES			SINÔNIMO	CATEGORIA	HOMÓGRAFOS	ANTECEDENTE	CONSEQUENTE
INICIO		COMPLEMENTO					
BASE	FLEXOES						
MA	IS	NOVO [DO] QUE	POSTERIOR A				
MA	IS	RECENTE [DO] QUE	POSTERIOR A				
MA	IS	ATUAL [DO] QUE	POSTERIOR A				
VELH	O,A,OS,AS			adj			
MOÇ	O,A,OS,AS			adj			
ALT	O,A,OS,AS			adj			
BAIX	O,A,OS,AS			adj			
ANTIG	O,A,OS,AS			adj			
NOV	O,A,OS,AS			adj			
ATUA	L,IS			adj			
RECENT	E,ES			adj			

LEXIAS			SINÔNIMO	CATEGORIA	HOMOGRAFOS	ENDEREÇOS	TRAÇOS E MARCAS
INICIO		COMPLEMENTO					
BASE	FLEXÕES						
ALUN	O,A,OS,AS			entidade		1	traço humano
				atributo		1.1,1.2,5.4,6.1	traço humano
DISCIPLIN	A,AS			entidade		2	
				atributo		2.1,2.2,3.3,4.1	
						4.2,5.1,6.2	
CURS	O,OS			entidade	verbo cursar	3	
				atributo		1.5,3.1	
REQUISIT	O,OS			entidade		4	
				atributo		4.3	
TURM	A,AS			entidade		5	marca tempo presente
				atributo		5.2	
HISTORIC	O,AS	[ESCOLAR]		entidade		6	marca tempo passado
MATRICUL	A,AS	[DE] [COM] NUMERO		atributo	verbo matricular	1.1,5.4,6.1	
NOM	E,ES			atributo		1.2	
SEX	O,OS			atributo		1.3	
NASCIMENT	O,OS			atributo		1.4	marca tempo
INGRESS	O			atributo		1.6	marca tempo
CODIG	O,OS			atributo		2.1,3.3,4.1,4.2,	
						5.1,6.2	
TITUL	O,OS			atributo		2.2,3.1	
TEORIC	A,AS			atributo		2.3.1	
PRATIC	A,AS			atributo		2.3.2	
CREDIT	O,OS			atributo		2.4,3.2,6.3	
PERIOD	O,OS	[DE APRESENTACAO]		atributo		2.5,6.4.2	marca tempo
PAPE	L,IS			atributo		3.4	
VAG	A,AS			atributo		5.3	
AN	O,OS			atributo		6.4.1	marca tempo
NOT	A,AS			atributo		6.5	

LEXIAS			SINÔNIMO	CATEGORIA	HOMOGRAFOS	ENDEREÇOS	TRAÇOS E MARCAS
INICIO		COMPLEMENTO					
BASE	FLEXOES						
EPOC	A		DATA	atributo		6.4	marca tempo
CARG	A,AS	HORARIA		atributo		2.3	
ESTUDANT	E,ES		ALUNO				
CADEIR	A,AS		DISCIPLINA				
MATERI	A,AS		DISCIPLINA				
DI	A,AS		DATA				
DAT	A,AS	DE NASCIMENTO	NASCIMENTO				
		EM QUE NASCER	NASCIMENTO				
		DE INSCRIÇÃO	INGRESSO(ANO E PERIODO)				
		EM QUE INSCREVER	INGRESSO,(ANO E PERIODO)				
		EM QUE CURSAR	(DATA E PERIODO)				
		EM QUE CONCLUIR	(ANO E PERIODO)				
		DE INICIO	INGRESSO				
		EM QUE INICIOU	INGRESSO				
NUMER	O,OS	DE MATRICULA	MATRICULA				
		DE DISCIPLINA	CODIGO				
CARG	A,AS	HORARIA	(TEORICA E PRATICA)				
		HORARIA TEORICA	TEORICA				
		HORARIA PRATICA	PRATICA				
		TEORICA	TEORICA				
		PRATICA	PRATICA				
CLASS	E,ES		TURMA				
GRA	U		NOTA				
GRAU	S		NOTA				

LEXIAS			SINÔNIMO	CATEGORIA	HOMOGRAFOS	ENDEREÇOS	TRAÇOS E MARCAS
INICIO		COMPLEMENTO					
BASE	FLEXÕE						
VID APROVEITAMENT SEMESTR	A,AS O,OS E,ES		HISTORICO HISTORICO PERIODO				

LEXIA			SINÔNIMO	CATEGORIA	HOMOGRAFOS	ANTECEDENTE	CONSEQUENTE	TIPO	
INICIO		COMP.							
BASE	FLEX.								
s	er			verbo,aux	conj. E			lig,tr	PASSIVO
est	ar			verbo,aux				lig,tr	
t	er			verbo,aux				tr	
continu	ar			verbo,aux		animado		lig,tr	
inici	ar			verbo,aux		animado		tr	ambiguidade: passado
termin	ar			verbo,aux		animado		tr	
hav	er			verbo,aux					= existir
fic	ar		continuar						
permanec	er		continuar						
segu	ir		continuar						
começ	ar		iniciar						
acab	ar		terminar						
find	ar		terminar						
conclu	ir		terminar						
encerr	ar		terminar						
complet	ar		terminar						
possu	ir		ter			animado			
cont	er			verbo					
obt	er			verbo		animado			
consegu	ir		obter						
alcanç	ar		obter						
conquist	ar		obter						
tir	ar		obter						
abrang	er		conter						
cadastr	ar			verbo		humano	aluno, disciplina curso	tr	= estar:fazer
inclu	ir			verbo		animado		tr	
faz	er			verbo		animado		tr	
curs	ar			verbo		aluno	curso, disciplina	tr	= fazer
realiz	ar		fazer						
estud	ar			verbo		aluno	curso, disciplina	tr	= fazer
matricul	ar			verbo		humano		tr,ref	= estar,fazer

val	er			verbo				tr,int	= ser
vari	ar			verbo				tr,int	= ser
pertenc	er			verbo				tr	
reprov	ar			verbo		aluno	curso,disciplina	tr	condição: ter nota < 5
aprov	ar			verbo		aluno	curso,disciplina	tr	condição: ter nota ≥ 5
rod	ar		reprovar						
pass	ar		aprovar						
inscrev	er			verbo		humano	aluno,curso, disciplina	tr,ref	= estar ambiguidade: passado
nasc	er			verbo		animado		inf	relativo: data nascimento, local nascimento
exist	ir			verbo				int	
ach	ar			verbo				tr	
consider	ar			verbo				tr	

APÊNDICE 5

O ALGORITMO DO ANALISADOR LÉXICO-AL

```
procedimento analisador-léxico
início
   enquanto houver palavras a analisar faça
   início
      execute procedimento ler-palavra-identificar-classe
      se      classe pontuação
      então execute procedimento pro-pontuação
      senão
            se    classe inteiro ou data ou real
            então execute procedimento pro-int-dat-real
            senão
                  se    classe literal
                  então execute procedimento pro-numcar
                  senão execute procedimento pro-alfabetica
   fim
fim
```

```
procedimento ler-palavra-identificar-classe
início
   enquanto próximo-caratere = espaço
      faça recupere proximo caractere
   se    proximo-caractere = aspas
   então execute procedimento  pro-lit
   senão
         se proximo-caractere = letra
         então execute procedimento  pro-letra
         senão
               se    proximo-caractere = pontuação
               então execute procedimento pro-pont
               senão palavra recebe espaço
                     execute procedimento pro-digito
fim
```

```
procedimento pro-lit
início
    recupere próximo caractere
    palavra recebe espaço
    comprimento recebe zero
    classe recebe literal-alfabet
    enquanto próximo-caractere ≠ aspas
    faça
    início
        concatene próximo-caractere na palavra
        comprimento recebe comprimento + 1
        se    próximo-caractere = letra ou hífen ou espaço
        então nada
        senão classe recebe literal-alfanum
        recupere próximo caractere
    fim
    recupere próximo caractere
fim
```

```
procedimento pro-letra
início
   palavra recebe espaço
   comprimento recebe zero
   classe recebe alfabet
   enquanto próximo-caractere ≠ espaço ou pontuação
   faça
   início
      concatene próximo-caractere na palavra
      comprimento recebe comprimento + 1
      se    próximo-caractere = letra ou hífen
      então nada
      senão classe recebe alfanum
      recupere próximo-caractere
   fim
fim


procedimento pro-pont
início
   classe recebe pontuação
   palavra recebe próximo-caractere
   recupere próximo-caractere
fim
```

```
procedimento pro-digito
início
    concatene próximo-caractere na palavra
    recupere próximo-caractere
    se    próximo-caractere = espaço ou ponto e vírgula
    então classe recebe inteiro
    senão
       se próximo-caractere = hífen
       então execute procedimento pro-data
       senão
             se proximo-caractere = vírgula
             então
                 se caractere seguinte = dígito
                 então execute procedimento pro-real
                 senão classe recebe inteiro
             senão execute procedimento pro-digito
fim
```

```
procedimento pro-data
início
   enquanto próximo-caractere = dígito ou hífen
   faça
   início
      concatene próximo caractere na palavra
      recupere próximo-caractere
   fim
   se    palavra tem formato de data
   então classe recebe data
   senão classe recebe alfanum
fim


procedimento pro-real
início
   classe recebe real
   concatene próximo-caractere na palavra
   recupere proximo-caractere
   enquanto próximo-caractere = dígito
   faça
   início
      concatene próximo-caractere na palavra
      recupere  próximo-caractere
   fim
fim
```

```
procedimento pro-pontuação
início
    se    categoria registro anterior = conjunção
          ou categoria da próxima palavra = conjunção
    então nada
    senão categoria recebe classe
          lexia recebe palavra
          significado recebe categoria
          registro recebe lexia e significado
          devolva registro
fim
```

```
procedimento pro-int-dat-real
início
   categoria recebe classe
   lexia recebe palavra
   se classe = data
      então endereço recebe os endereços dos tipos data na base
            de dados que contem essa ocorrência
      senão se    classe = real
            então endereço recebe os endereços dos tipos real
                  na base de dados que contem essa ocorrência
            senão endereço recebe os endereços dos tipos real
                  e inteiro na base de dados que contem essa
                  ocorrência
   significado recebe categoria e endereço
   registro recebe lexia e significado
   devolva registro
fim
```

```
procedimento pro-alfabetica
início
   separe a base e a flexão da palavra
   se     a flexão é pronome pessoal enclítico
   então execute procedimento pro-enclise
   senão recupere base no dicionário
         se     base não existe ou a flexão não existe para
                essa base
         então execute procedimento pro-numcar
         senão
               se     a palavra tem complemento e as próximas
                      palavras coincidem com um complemento
               então lexia recebe palavra + complemento
               senão lexia recebe palavra
               se     lexia tem sinônimo
               então recupere sinônimo
                     lexia recebe sinônimo
               senão nada
               para cada significado faça
               início
                  categoria recebe a categoria do dicionário
                  se  . categoria = conjunção ou preposição
                        ou advérbio
                  então significado recebe categoria
                  senão
                        se    categoria = pronome ou artigo
                        então execute procedimento pro-pro-art
                        senão
                              se    categoria = numeral
                              então execute procedimento pro-num
                              senão
                                    se    categoria = adjetivo ou
                                          substantivo
                                    então execute procedimento pro-ad-sub
                                    senão execute procedimento pro-verbo
               fim
               registro recebe lexia e todos significados
               devolva registro
fim
```

```
procedimento pro-numcar
início
    se      categoria registro anterior = numcar
    então   concatene lexia do registro anterior com um espaço
            e com a palavra e destrua registro anterior
    senão   nada
    lexia recebe palavra
    categoria recebe numcar
    recupere a palavra no dicionário
    se      existe e é numcar
    então   endereço recebe endereços do dicionário
    senão   endereço recebe endereços dos tipos numcar na base
            de dados que contem numcar com o comprimento dessa
            numcar
    significado recebe categoria e endereços
    registro recebe lexia e significado
    devolva registro
fim
```

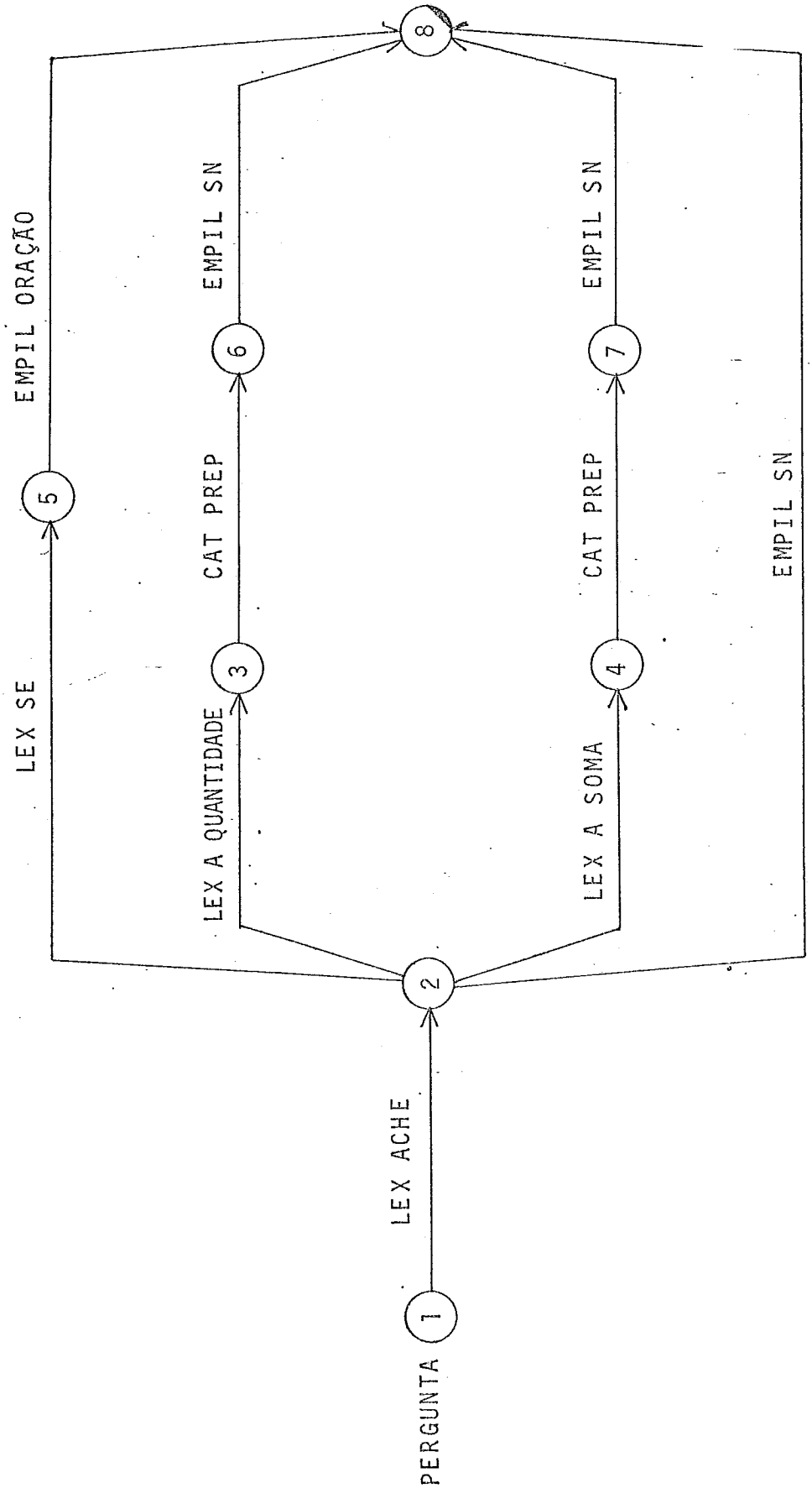
```
procedimento pro-enclise
início
    variável recebe flexão
    separe a base e a flexão da base
    recupere base no dicionário
    se      base não existe ou a flexão não existe para essa base
            ou não há significado com a categoria verbo
    então execute procedimento pro-numcar
    senão lexia recebe base + flexão
            categoria recebe verbo
            execute procedimento pro-verbo
            registro recebe lexia e significado
            devolva registro
            lexia recebe variável
            categoria recebe pr. pess
            genero recebe genero da flexão
            número recebe número de flexão
            função recebe objeto
            significado recebe categoria, gênero, número e função
            registro recebe lexia e significado
            devolva registro
fim
```

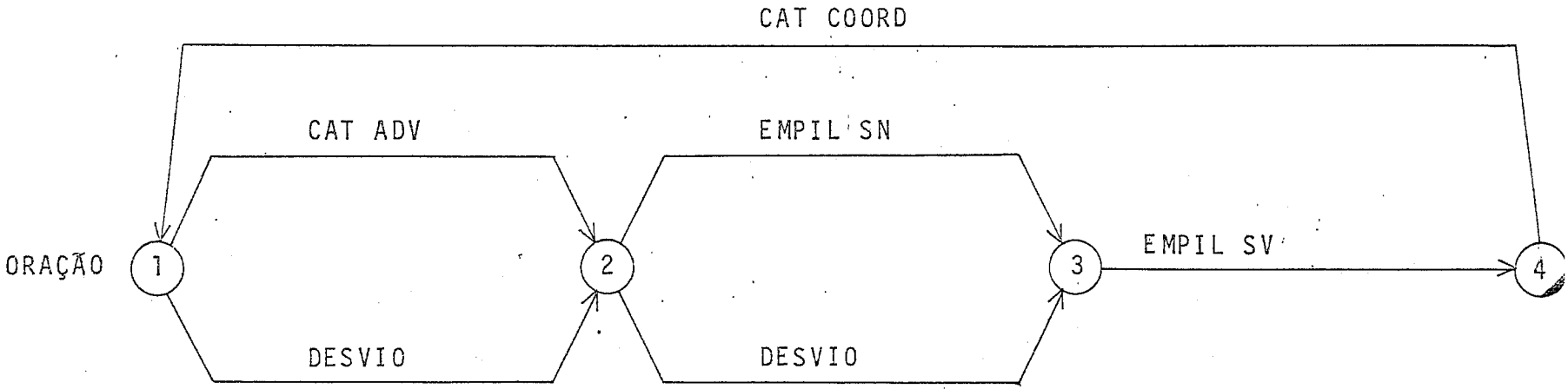
```
procedimento pro-pro-art
início
    gênero recebe o gênero da flexão
    número recebe o número da flexão
    significado recebe categoria, gênero e número
fim


procedimento pro-num
início
    gênero recebe o gênero da flexão
    número recebe o número da flexão
    significado recebe categoria, gênero e número
    se    gênero masculino
    então categoria recebe inteiro
          gênero recebe masculino
          número recebe o número da flexão
          outro significado recebe categoria, gênero e número
fim
```

```
procedimento pro-ad-sub
início
    gênero recebe gênero da flexão
    número recebe número da flexão
    significado recebe categoria, gênero, número e
    todas características do dicionário
fim


procedimento pro-verbo
início
    recupere modo, tempo e pessoa da flexão
    se    modo = particípio e categoria do registro
          anterior = verbo
    então tempo do verbo do registro anterior recebe passado
    senão nada
    significado recebe categoria, modo, tempo, pessoa e todas
    características do dicionário
fim
```

APÊNDICE 6

O ALGORITMO DO ANALISADOR SINTÁTICO E SEMÂNTICO - ASS


PERGUNTA
1
LEX ACHE
2
LEX A QUANTIDADE
3
CAT PREP
EMPIL SN
6
LEX A SOMA
4
CAT PREP
EMPIL SN
7
LEX SE
5
EMPIL ORAÇÃO
EMPIL SN
8

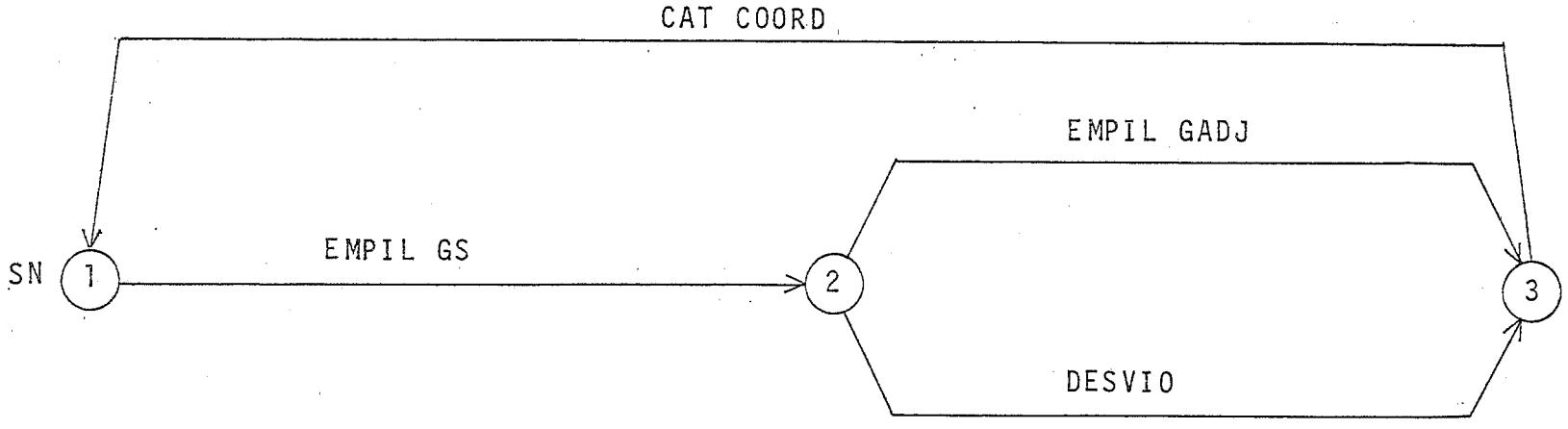


2 → 3 A omissão do SN é permitida nos seguintes casos:

a) Verbo haver no sentido de existir, com sujeito indeterminado

b) Verbo existir com SN posposto

c) Em uma sequência de condições que se referem ao mesmo SN, pode-se omitir a repetição do SN; testar a "flag" de sequência de condições; repetir SN.

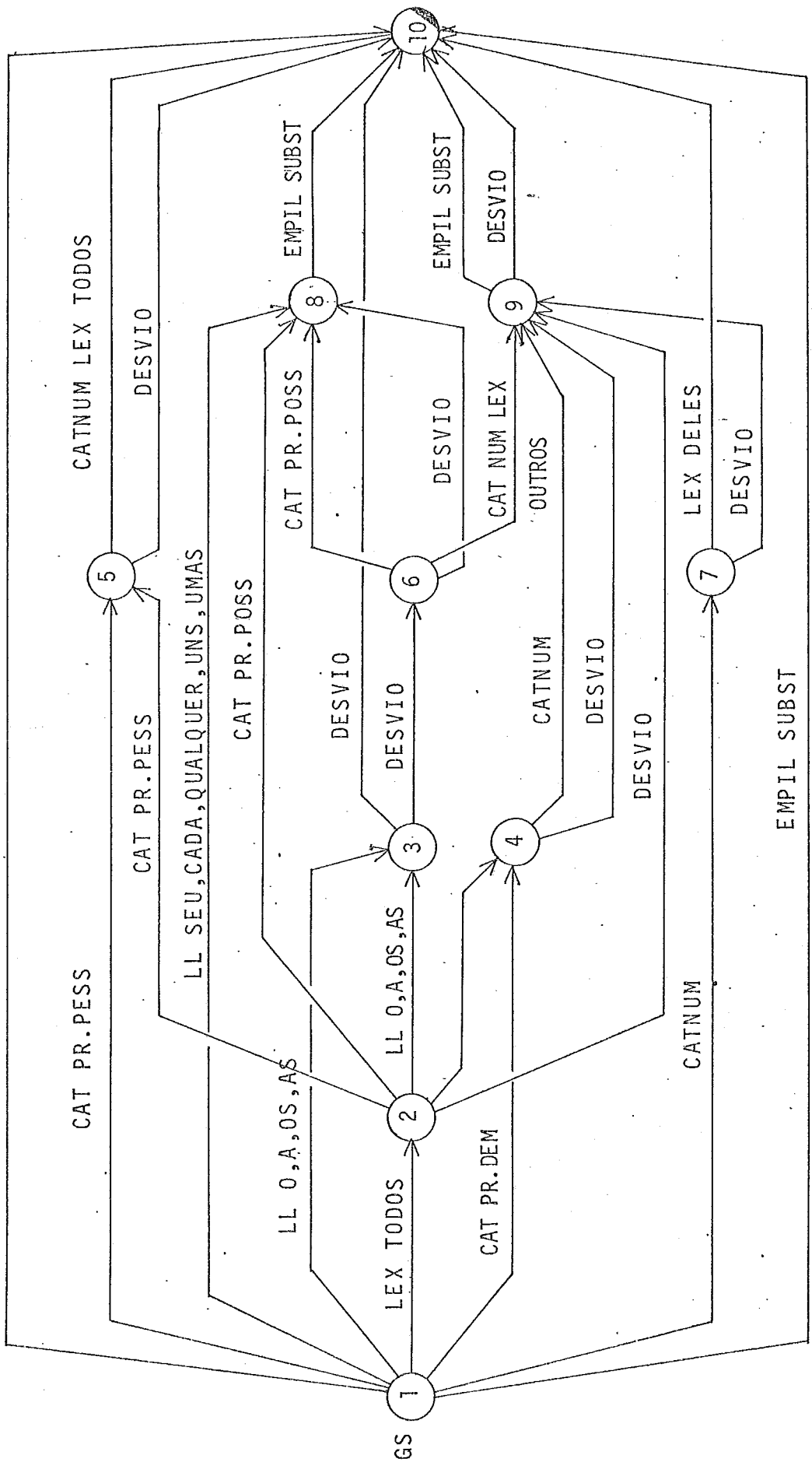


2 → 3 A omissão do GADJ é permitida em dois casos: .

a) Na lista de atributos, caso da pergunta tipo 4

b) No caso de existir junto ao substantivo ou substituindo o substantivo, uma ocorrência.

Nos demais casos em que o GADJ for omitido, é criada a condição "VERDADEIRO", para indicar a não qualificação.

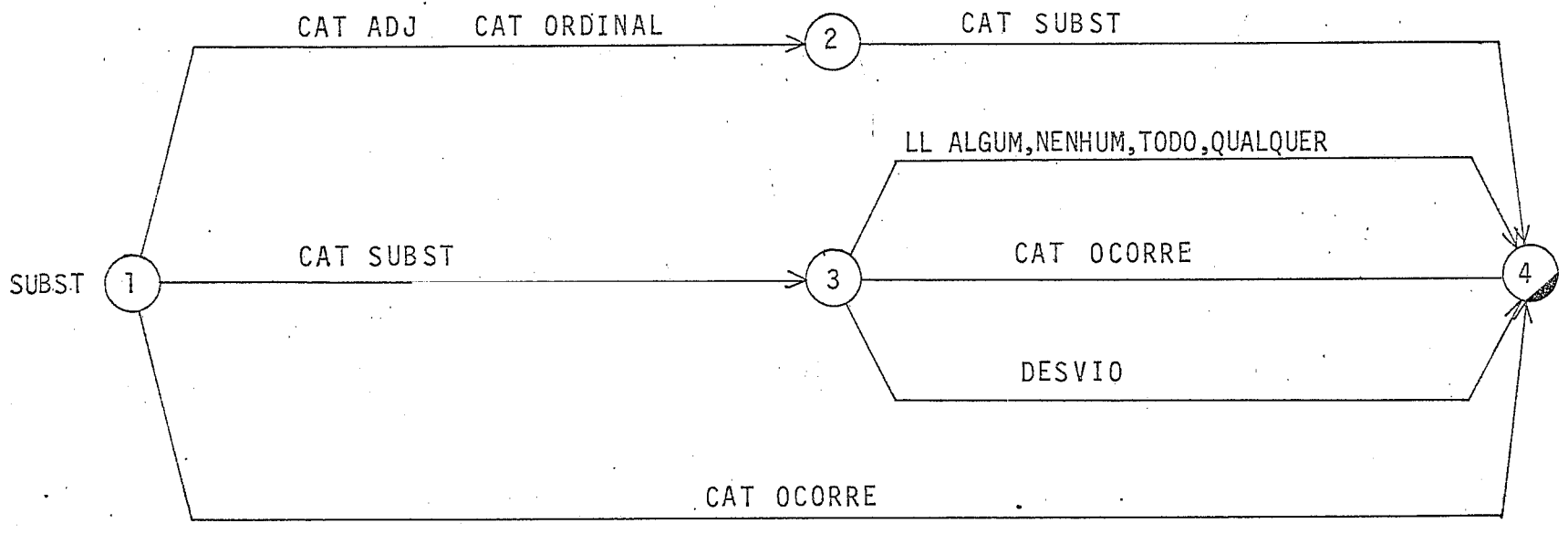

GS
1
LEX TODOS
CAT PR.DEM
2
LL 0,A,OS,AS
LL 0,A,OS,AS
3
DESVIO
DESVIO
4
CATNUM
DESVIO
5
CAT PR.PESS
CAT PR.PESS
6
DESVIO
DESVIO
7
CATNUM
DESVIO
8
CAT PR.POSS
CAT PR.POSS
9
CAT NUM LEX
OUTROS
10
EMPIL SUBST
EMPIL SUBST
DESVIO
LEX DELES
DESVIO
EMPIL SUBST
CATNUM LEX TODOS
DESVIO
LL ALGUEM,NINGUEM
CAR PR.RELAT
LL SEU,CADA,QUALQUER,UNS,UMAS

3 → 10 Se categoria da próxima palavra é verbo, então O,AS,OS,AS é pronome pessoal oblíquo.

3 → 6 Nos demais casos.

9 → 10 Se não substantivo, então o substantivo está sendo substituido por pronome ou numeral.

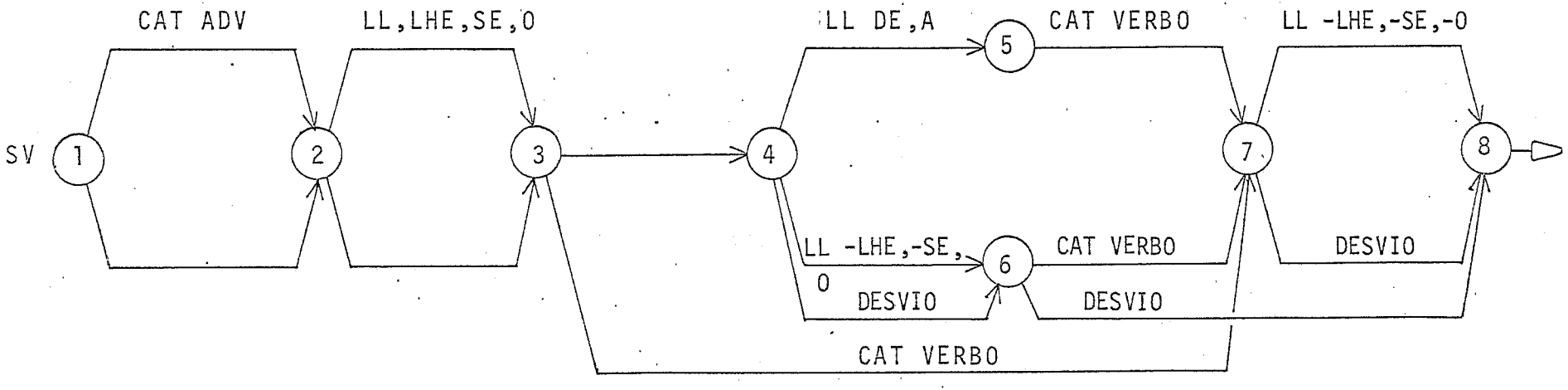
5 → 10 Se existir numeral ou pronome indefinido, os mesmos são incluidos no determinante.



1 → 2 Adjetivo ou ordinal anteposto ao substantivo é colocado em GUARDA.

3 → 4 Pronome indefinido após o substantivo é transferido para o determinante.
No caso de ocorrência, isto é, numcar, data, inteiro ou real, nível desce 1 e após a ocorrência sobe 1.

1 → 4 É possível a omissão do substantivo, e só constar a ocorrência. Neste caso o nível desce 2 antes da ocorrência, e sobe 2 após, e é necessário avaliar a que atributo se refere a ocorrência.



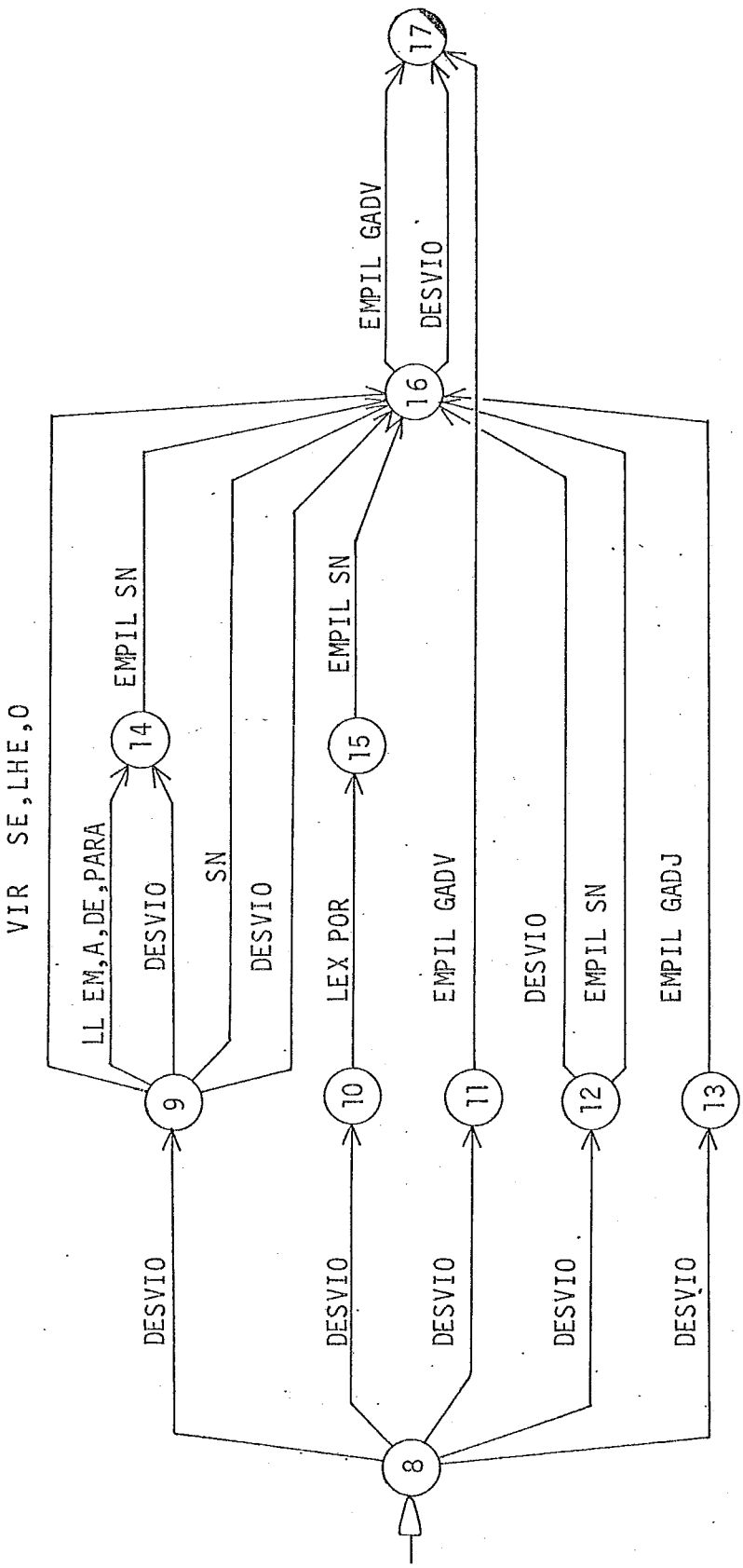
Início do SV

2 → 3, 4 → 6, 7 → 8 Os pronomes pessoais oblíquos são colocados em GUARDA.

3 → 4 Duas condições: verbo auxiliar, e modo indicativo ou subjuntivo.

5 → 6 Condição: forma nominal.

6 → 7 Condição: forma nominal.


VIR SE,LHE,0
LL EM,A,DE,PARA
DESVIO
EMPIL SN
SN
DESVIO
LEX POR
EMPIL SN
EMPIL GADV
DESVIO
EMPIL SN
EMPIL GADJ
DESVIO
EMPIL GADV
DESVIO
8
9
10
11
12
13
14
15
16
17

8 → 9 Condição: verbo transitivo, forma ativa.

8 →10 Condição: verbo transitivo, forma passiva.

8 →11 Condição: verbo transitivo e reflexivo, forma passiva.

8 →12 Condição; verbo transitivo.

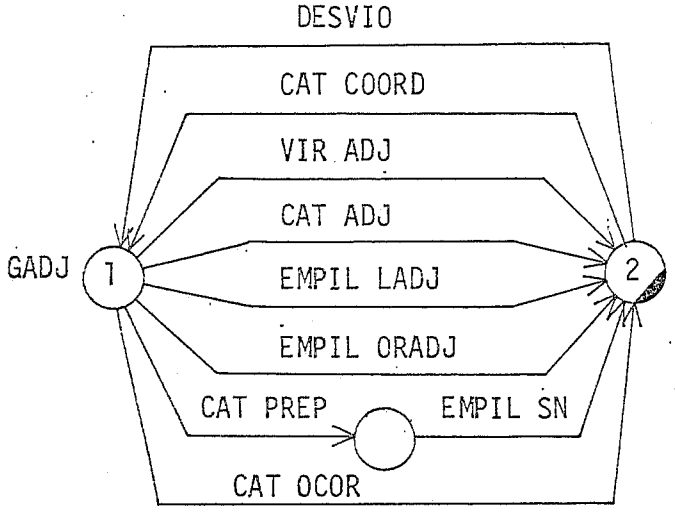
8 →13 Condição: verbo de ligação.

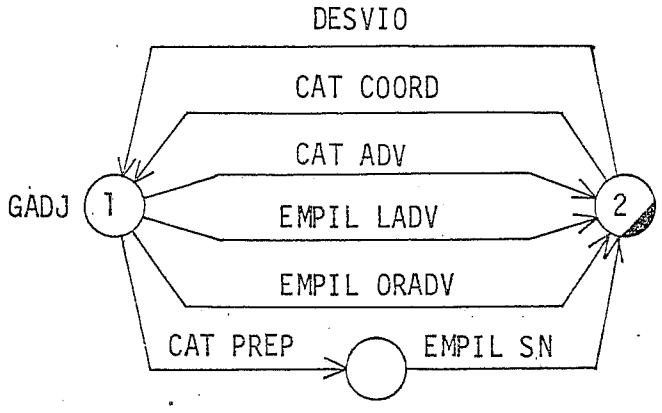
9 →16' Condição: sujeito posposto em oração relativa em que o objeto é pronome relativo.

9 →16'' Condição: objeto antecede o verbo (pronome relativo).

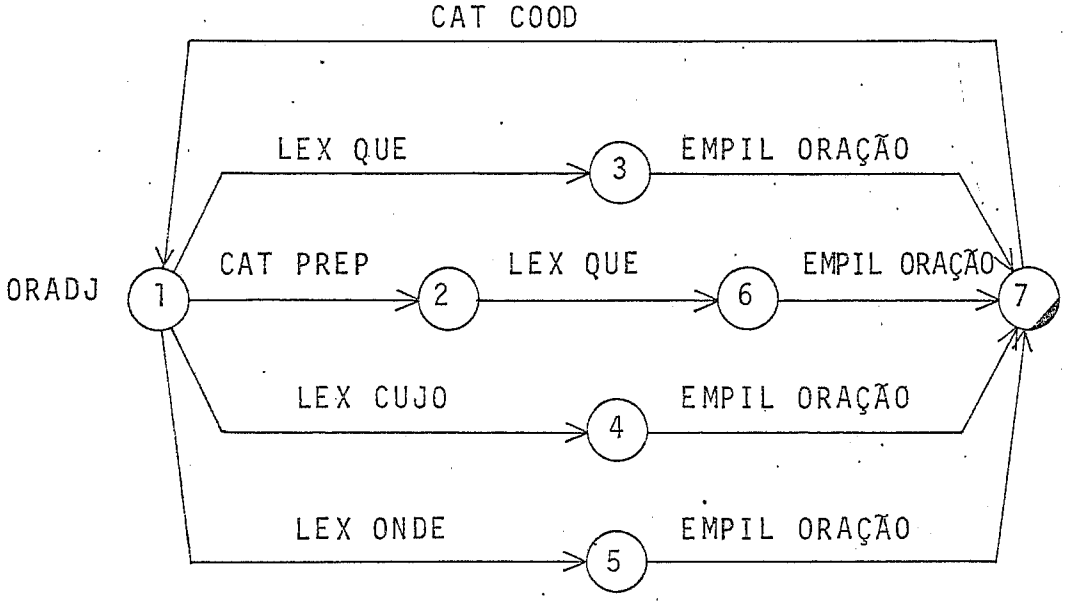
12 →16'' Condição: SN posposto ao verbo.

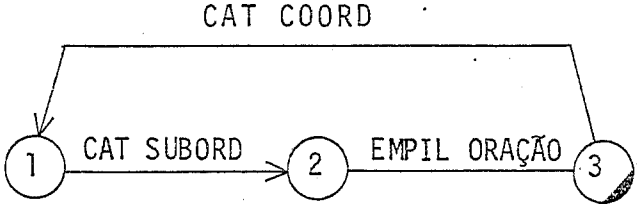
No SV é verificado o antecedente e o consequente do verbo, comparando-os com o SN que o antecede ou segue. Na voz passiva há inversão do antecedente e consequente.





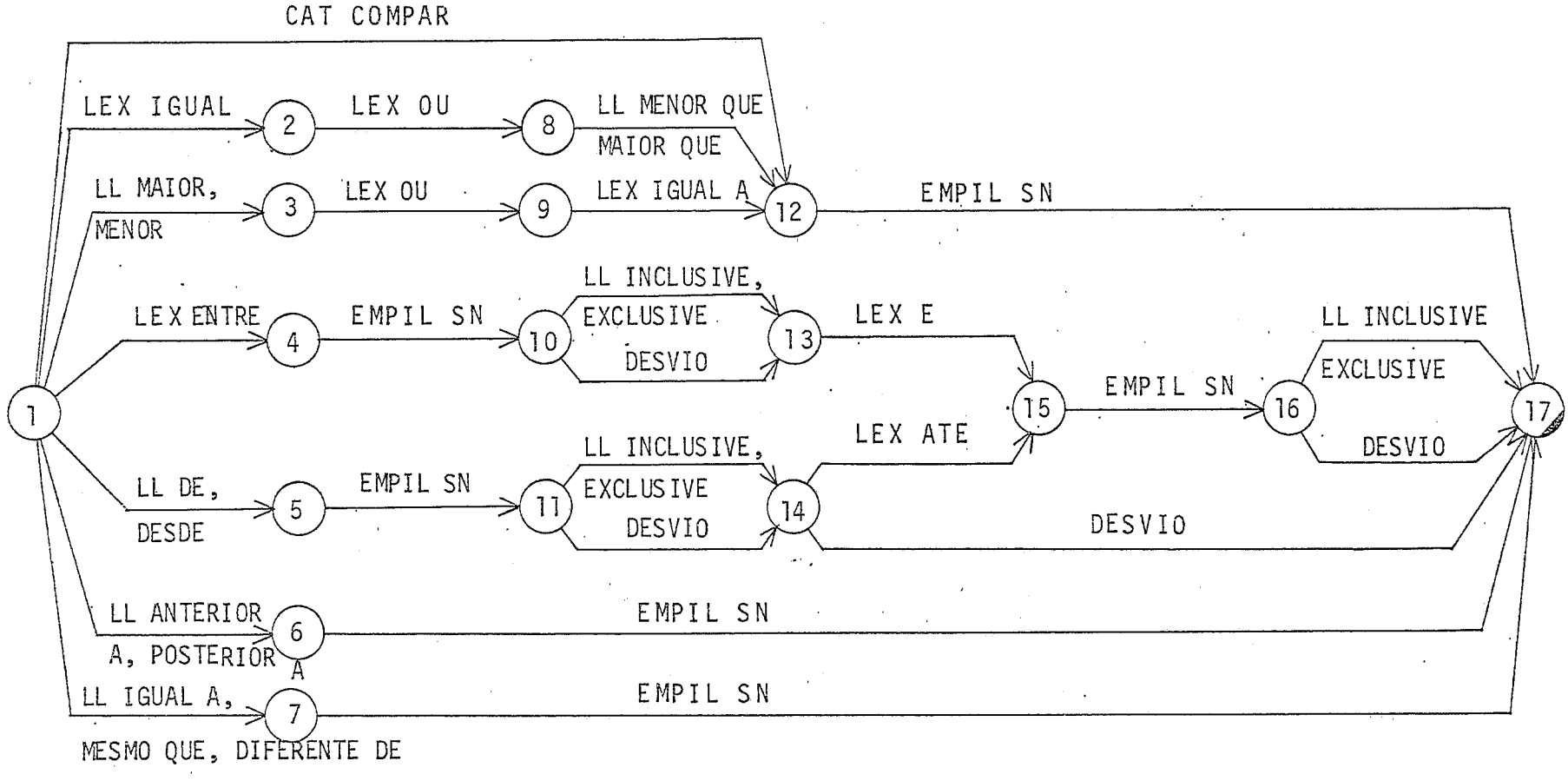
Os grupos GADJ e GADV podem ser repetidos com ou sem coordenação entre as ocorrências. No início dos grupos GADJ e GADV o nível desce de um , e no fim sobe de um, mas os retornos por "CAT COORD" e "DESVIO" não alteram o nível.





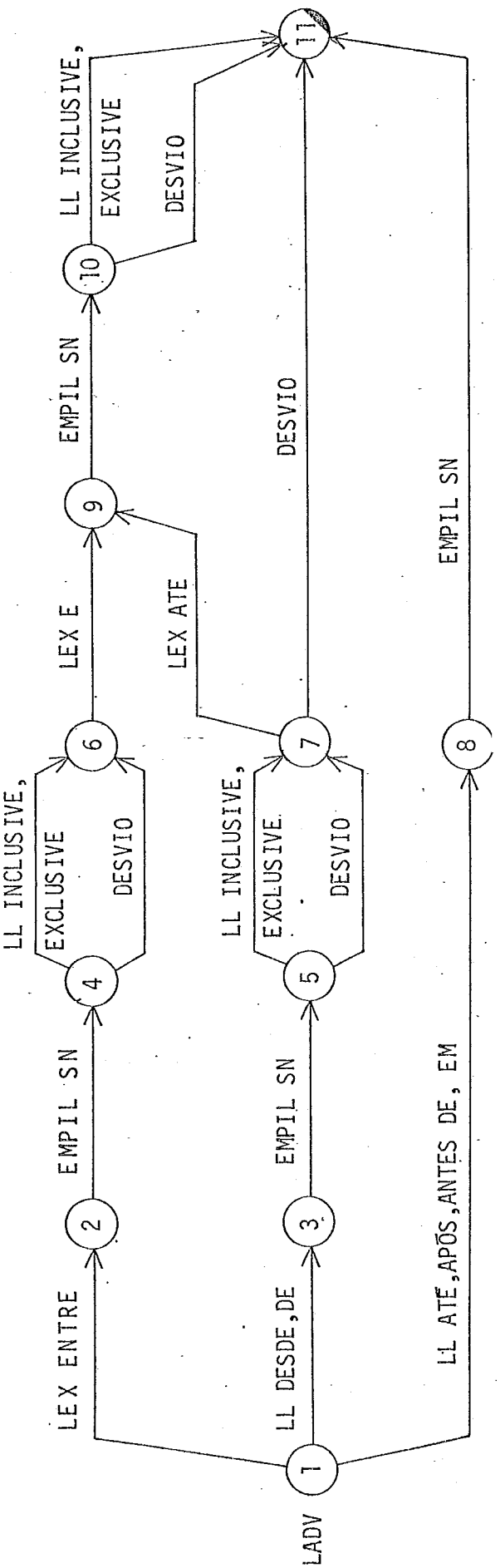
3 → 7 Marcar flag condição:"que"pode ser sujeito ou objeto.

6 → 7 Marcar flag condição:"que"pode ser objeto.



4 → 10 e 5 → 11 e 12 → 17 e 15 → 16 Condição: na avaliação do SN obter inteiro ou real.

6 → 17 Condição: na avaliação do SN obter data.



2 → 4 e 3 → 5 e 10 → 11 e 8 → 11 Condição: na avaliação do SN obter data.

APÊNDICE 7

O ALGORITMO DO GERADOR DE PERGUNTAS EM LOBAN - GPL

```
procedimento gerador-de-perguntas
início
   se tipo pergunta = 1
   então execute procedimento pergunta - 1
   senão
      se tipo pergunta = 2
      então execute procedimento pergunta - 2
      senão
         se tipo pergunta = 3
         então execute procedimento pergunta - 3
         senão execute procedimento pergunta - 4
fim


procedimento pergunta - 1
início
   pergunta tem a estrutura
      "FAZER EM CASO
         <obter valbool>  REPRESENTAR 'SIM' EM CC
          CONTRARIO REPRESENTAR 'NAO' EM CC"
   execute procedimento obter-valbool
fim
```

```
procedimento pergunta - 2
início
   pergunta tem a estrutura
        "REPRESENTAR CONT <end ponto> EM CC
   execute procedimento end-ponto
fim


procedimento pergunta - 3
início
   pergunta tem a estrutura
        "SUBSTITUIR EM AUX.ITEM POR 0
         FAZER PARA CADA PONTO <end ponto>
         (SUBSTITUIT EM AUX.ITEM POR
          C AUX.ITEM + V PC)
         REPRESENTAR C AUX.ITEM EM CC
   execute procedimento end ponto
fim
```

```
procedimento pergunta - 4
início
   se lista-atributos = vazia
      e condição = vazia
   então execute procedimento pergunta - 4.12
   senão
       se lista-atributos = vazia
       então execute procedimento pergunta - 4.34
       senão se atributos de lista-atributos pertence
             a uma mesma tabela
             então execute procedimento pergunta - 4.56
             senão execute procedimento pergunta - 4.7
fim
```

```
procedimento pergunta - 4.12
início
   se endereço de tarel
   então pergunta tem a estrutura .
         "REPRESENTAR C <end tarel> EM CC"
         obter end-tarel diretamente do endereço da entidade
   senão pergunta tem a estrutura
         "REPRESENTAR DESAGRUP C <end talig> EM CC"
         obter end-talig  diretamente do endereço da entidade
fim
```

```
procedimento pergunta - 4.34
início
   se endereço de tarel
   então pergunta tem a estrutura
         "REPRESENTAR COLEC <end tupla> EM CC"
   senão pergunta tem a estrutura
         "REPRESENTAR DESAGRUP COLEC <end ligação> EM CC"
   end-tupla ou end-ligação tem a estrutura
   "ACTRAB.<nome arquivo>.TR ou TL.(=<obter marca>).
    (<obter valbool>)"
   recuperar o nome do arquivo do endereço de entidade
   selecione TR para tabela relacional ou TL para tabela
        ligacional
   recuperar nome da marca do endereço de entidade
   execute procedimento obter valbool
fim
```

```
procedimento pergunta 4.56
início
   se endereço de tarel
   então pergunta tem a estrutura
         "REPRESENTAR ESTREIT COLEC <end tupla>
          PARA <atributo>,,, EM CC"
   senão pergunta tem a estrutura
         "REPRESENTAR ESTREIT DESAGRUP COLEC <end ligação>
          PARA <atributo>,,, EM CC"
   end-tupla ou end-ligação tem a estrutura
   "ACTRAB.<nome arquivo>.TR ou TL.(=<obter marca>)
    (<obter valbool>)"
    recuperar o nome do atributo, TR ou TL e o nome da marca
    do endereço da entidade
    execute procedimento obter-valbool
    recuperar atributos de lista de atributos
fim
```

```
procedimento pergunta 4.7
início
   pergunta tem a estrutura
   "SUBSTITUIR EM AUX.TAB POR VAZIO
    FAZER PARA CADA <end ponto>
       (FAZER PARA CADA <end ponto>
           (.
            :
              (INCLUIR COMPOR
                  (<obter nome>:= C <end atrib>
                                  .
                                  :
                                                  )
               EM AUX.TAB) ...)
    REPRESENTAR AUX.TAB EM CC"
    cada "fazer" e seu "<end ponto>" corresponde à lista de atri
    butos e condições sobre uma tabela, conforme especificado na
    pergunta
    se a tabela é relacional
    então <end ponto> tem a estrutura
          "ACTRAB.<nome arquivo>.TR.(=<obter marca>).
           (<obter valbool>)"
    senão
       se o atributo pertence ao ligante
       então o <end ponto> tem a estrutura
             "ACTRAB.<nome arquivo>.TL(=<obter marca).
              (<obter valbool>)"
       senão o <end ponto> tem a estrutura
             "ACTRAB.<nome arquivo>.TL(=<obter marca)
              (<obter valbool>) .T.(=<obter marca>)(<obter
               valbool>))"
       enquanto existir condição sobre tabela diferente faça
       início
           execute procedimento obter valbool
       fim
```

recuperar <obter nome> da lista de atributos,
sabendo-se que C <end atrib> tem o formato
"C PC /<obter marca>.atributo", e o nome da marca é
obtido do endereço, e atributo é igual ao <obter nome>

fim

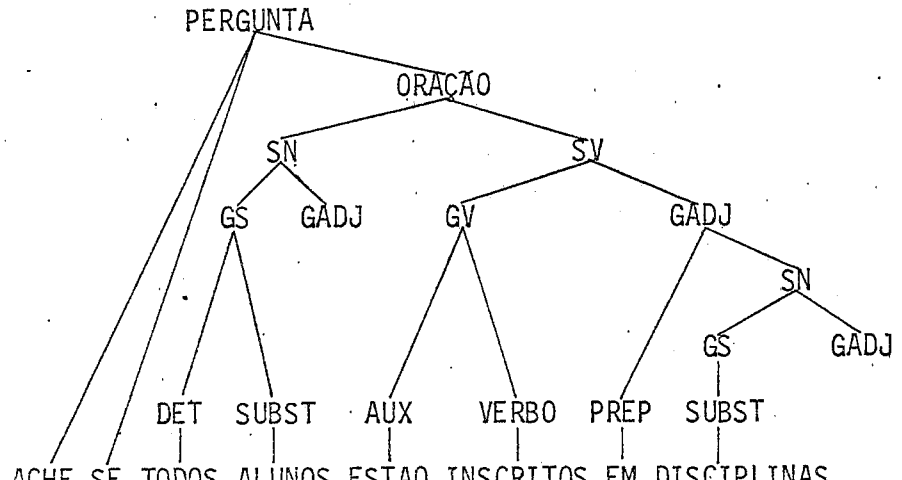
```
procedimento obter-valbool
início
   enquanto existir condição faça
   início
      se primeira vez
      então não há conetor
      senão avalia conetor "E" ou "OU"
      se negação
      então comece obter-valbool com "NÃO"
      senão nada
      se  quantificação = numeral
      então obter-valbool tem o formato "CONT <end ponto> =
            = <literal>"
            execute procedimento end-ponto
      senão se quantificação = ALGUM
            então obter-valbool tem o formato "EXIST
                <end ponto>
                execute procedimento end-ponto
            senão
               se  quantificação = TODOS
               então obter-valbool tem o formato
                     "PARA TODO <end ponto>((<obter valbool>))"
                     execute procedimento end-ponto
                        para a condição até predicado
                     execute procedimento obter-valbool
               senão execute procedimento termo-elementar
   fim
fim
```

```
procedimento termo-elementar
início
    obter valbool é avaliado sobre condição com comparação,
    onde o primeiro termo da comparação é end ponto e o segundo
    é end ponto ou literal; o contexto do primeiro termo é  o
    contexto já atingido pela avaliação da pergunta ou ACTRAB;
    o contexto do segundo termo é ACTRAB.
    execute procedimento end-ponto para o primeiro termo da
        comparação
    se segundo termo = literal
    então avalie literal
    senão execute procedimento end-ponto
    se primeiro termo aponta uma construção
    então
        se segundo termo aponta uma construção
        então obter valbool tem o formato
              "C <end ponto>{=|<|≤|>|≥}{<literal>|
               C <end ponto>} [EM ORDEM DATA]"
        senão obter valbool tem o formato
              "C <end ponto> ∈ COLEC <end ponto>"
    senão obter valbool tem o formato
          "COLEC <end ponto> { ⊂|⊃ } COLEC <end ponto>"
fim
```

```
procedimento end-ponto
início
   se contexto não está definido
   então contexto recebe ACTRAB
   senão
      se contexto é conjunto de pontos marcados a nível de
         tupla ou ligação
      então contexto recebe "PC/<obter marca>", onde a mar
            ca é obtida da marca do contexto
      senão nada
   enquanto não chegar ao nível terminal faça
   início
      desça um nível no endereço
      se nível de arquivo
      então obtenha nome-arquivo do endereço na pergunta
      senão
         se nível tabela
         então obtenha TR para tarel ou TL para talig
         senão
            se nível tupla ou ligação
            então obtenha marca
                  execute procedimento obter-valbool
                  com condições da pergunta
            senão
               se nível de ligante ou ligado
               então obtenha L para ligante ou TL para ligado
               senão obtenha nome do atributo do endereço na
                     pergunta
   fim
fim
```

APÊNDICE 8

EXEMPLOS



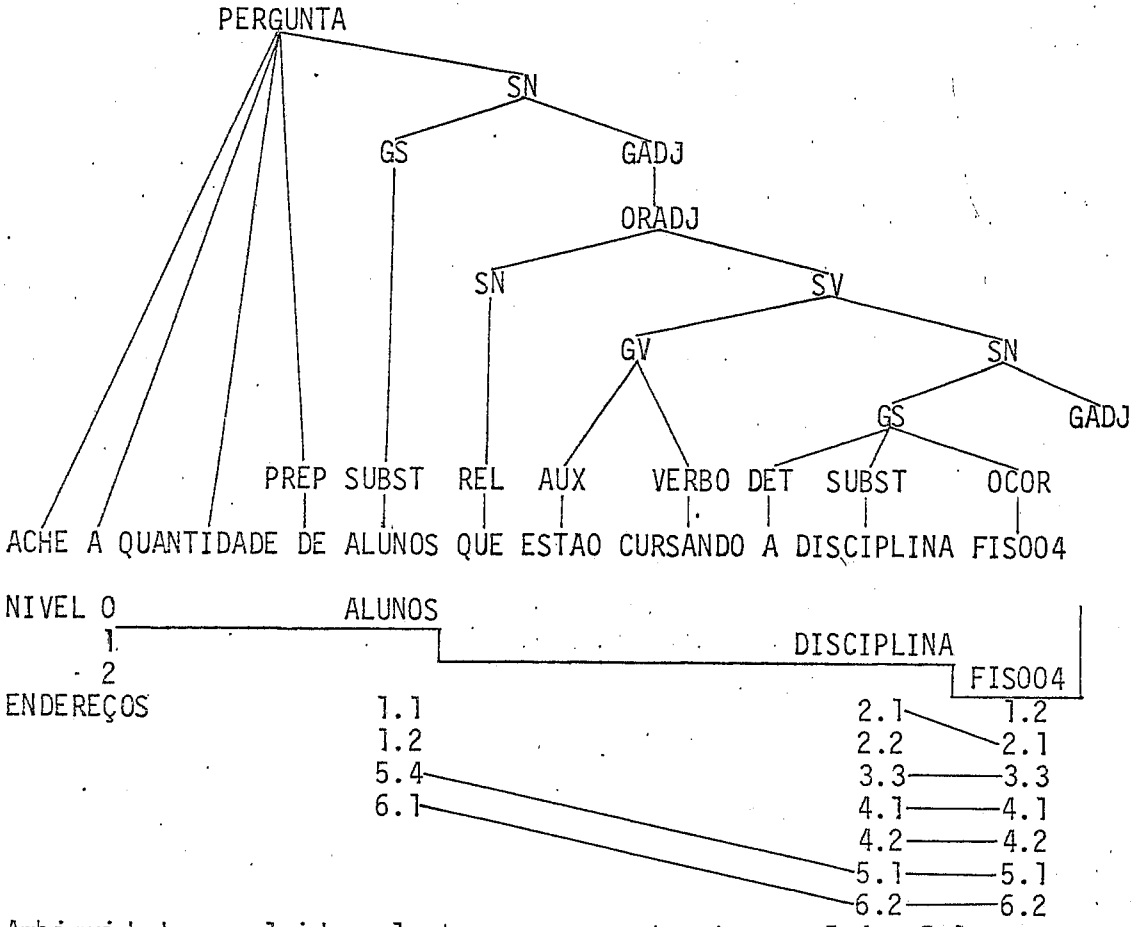
NIVEL 0 ALUNOS DISCIPLINAS

ENDEREÇOS				
	1.1	2.1	2.2	3.3
	1.2	4.1	4.2	
	5.4		5.1	
	6.1		6.2	

Ambiguidade resolvida pelo tempo: presente, turmas 5.4 e 5.1

(TIPO(1) ENTIDADE/ATRIBUTO(1.1) NEGAÇÃO( ) QUANTIFICAÇÃO(TODOS) (CONDIÇÃO(CONETIVO( ) NEGAÇÃO( ) QUANTIF( ) ESPECIFICAÇÃO(ENDEREÇO(1.1) = ENDEREÇO(5.4) ELO(5.1) = VALOR(VERDADEIRO)))))

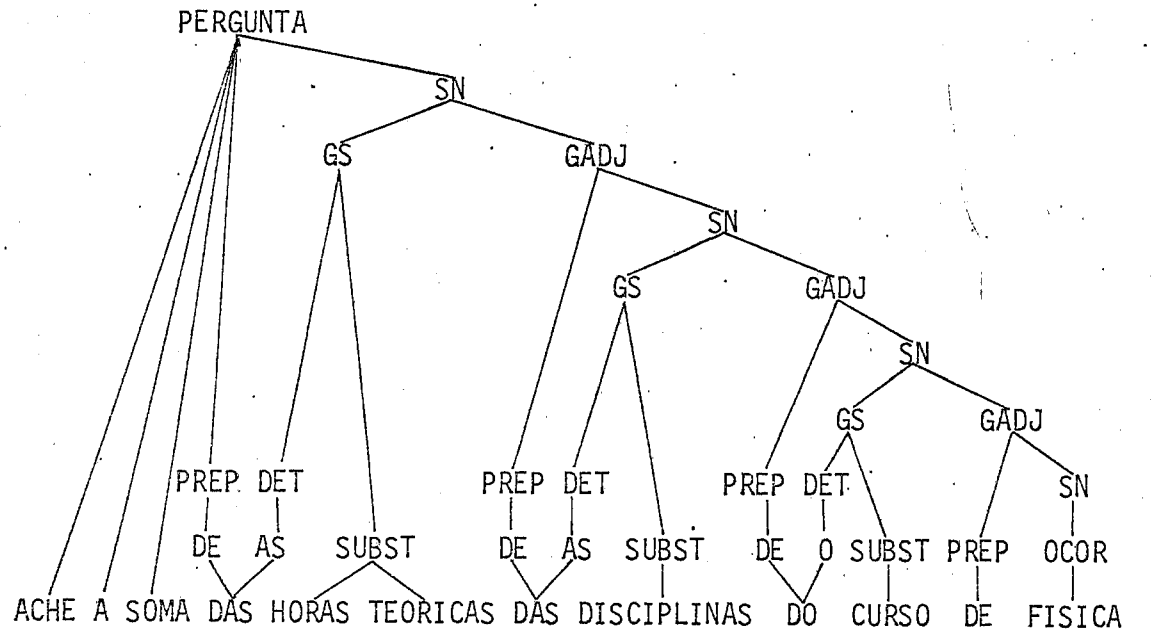
```
FAZER EM CASO
    PARA TODO ACTRAB.ALUNOS.TR.(=M1)(VERDADEIRO)
        (C PC/M1.MATRICULA ∈ COLEC ACTRAB.TURMAS.TL.(=M5)(VERDADEIRO).T.(=M9)(VERDADEIRO).ALUNO)
            REPRESENTAR 'SIM' EM CC
    CONTRARIO REPRESENTAR 'NAO' EM CC
```

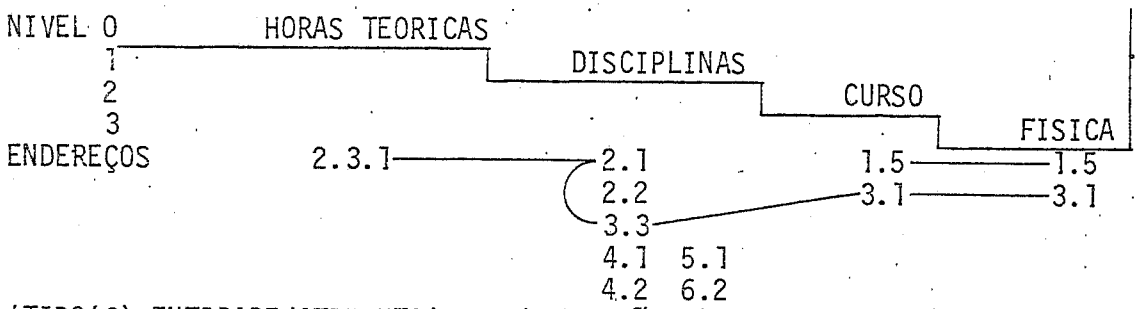


Ambiguidade resolvida pelo tempo: presente, turmas 5.4 e 5.1

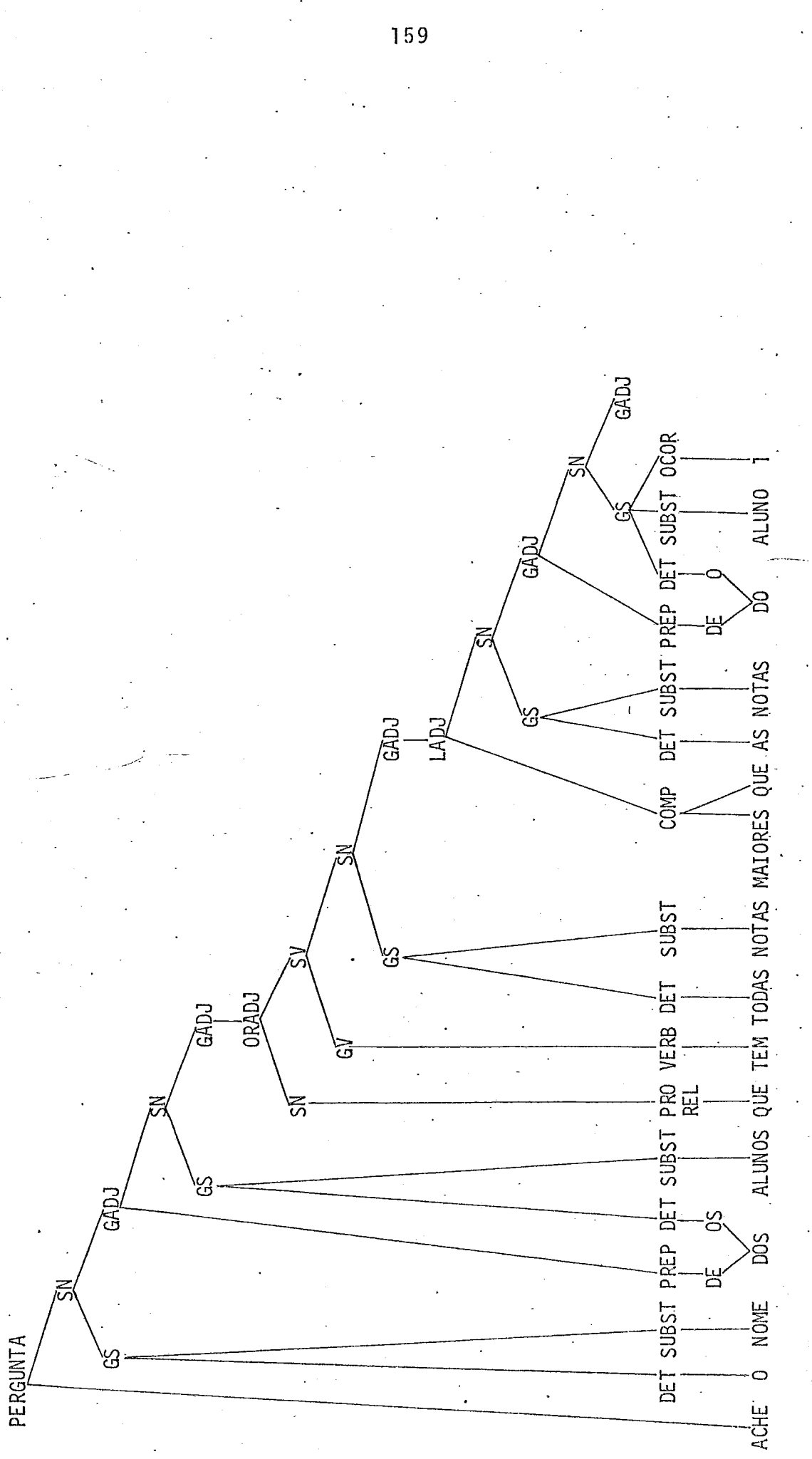
(TIPO(2) ENTIDADE/ATRIBUTO(1.1) NEGAÇÃO( ) QUANTIFICAÇÃO( )(CONDIÇÃO(CONETIVO( ) NEGAÇÃO( ) QUANTIFICAÇÃO( ) ESPECIFICAÇÃO(ENDEREÇO(1.1) = ENDEREÇO(5.4) ELO (5.1) = VALOR(FIS004)

REPRESENTAR CONT ACTRAB.TURMAS.TL.(=M5)(C L.DISCIPLINA = 'FIS004').T.(=M9)(VERDADEIRO).ALUNO EM CC

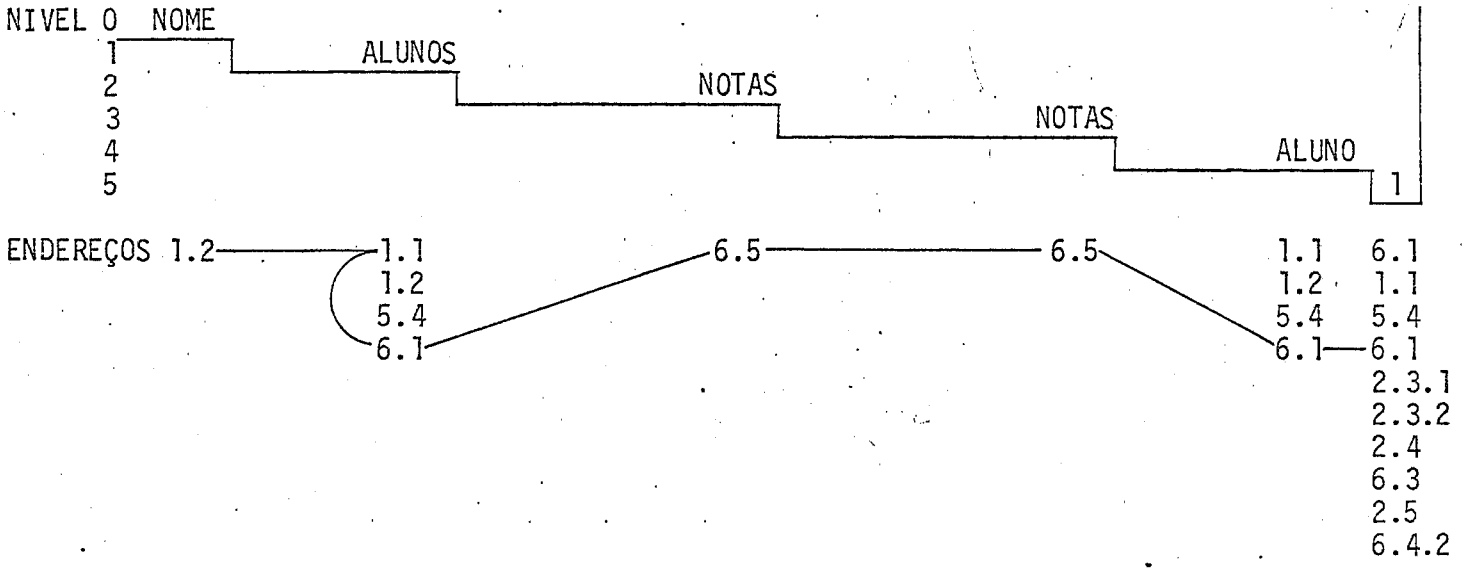




(TIPO(3) ENTIDADE/ATRIBUTO(2.3.1) NEGAÇÃO( ) (QUANTIFICAÇÃO( )(CONDIÇÃO(CONETIVO( ) NEGAÇÃO( ) QUANTIFICAÇÃO( ) ESPECIFICAÇÃO(ENDEREÇO(2.1) = ENDEREÇO(3.3) ELO (3.1) = VALOR(FISICA)))))
SUBSTITUIR EM AUX.ITEM POR 0
FAZER PARA CADA PONTO ACTRAB.DISCIPLINAS.TR.(=M2)(C CODIGO ∈ COLEC ACTRAB.CURSOS.TL.(=M3)(C L.CURSO = 'FISICA').T
.(=M7)(VERDADEIRO)).CARGA-HORARIA.TEORICA
(SUBSTITUIR EM AUX.ITEM POR C AUX.ITEM + V PC)
REPRESENTAR C AUX.ITEM EM CC


PERGUNTA
SN
GS
DET SUBST
ACHE O
GADJ
SN
GS
DET SUBST PREP DET SUBST
O NOME DE OS
GADJ
SN
GS
DET SUBST PREP DET SUBST OCOR
DO ALUNO 1
GADJ
SN
GS
DET SUBST PRO VERB DET SUBST
DOS ALUNOS QUE TEM TODAS NOTAS
ORADJ
SV
SN
GS
DET SUBST
REL
GV
TEM TODAS NOTAS
SN
PRO
QUE
GADJ
LADJ
SN
GS
DET SUBST PREP DET SUBST
AS NOTAS DE OS MAIORES
COMP
QUE

ACHE O NOME DOS ALUNOS QUE TEM TODAS NOTAS MAIORES QUE AS NOTAS DO ALUNO 1



(TIPO(4) ENTIDADE(1.2) ((LIST-ATRIB(1.2)(CONDIÇÃO(CONETIVO( ) NEGAÇÃO( ) QUANTIFICAÇÃO(TODOS) ESPECIFICAÇÃO(ENDEREÇO(1.1) = ENDEREÇO(6.1) ELO(6.5) > VALOR(ESPECIFICAÇÃO(ENDEREÇO(6.5) COMPARADOR( ) CONDIÇÃO(CONETIVO( ) NEGAÇÃO( ) QUANTIFICAÇÃO( ) ESPECIFICAÇÃO(ENDEREÇO(6.1) COMPARAÇÃO(=) VALOR(1))))))))))))

```
REPRESENTAR
   ESTREIT
      COLEC ACTRAB.ALUNOS.TR.(=M1)(PARA TODO ACTRAB.HISTORICOS.TL.(C L.ALUNO = C PX/M1.MATRICULA).T.(VERDADEIRO)
                                          .NOTA(=MX)
                                          (PARA TODO ACTRAB.HISTORICOS.TL.(C L.ALUNO = 1).T.(VERDADEIRO).NOTA(=MY)
                                                     (C PC/MX    C PC/MY)))
   PARA NOME
EM CC
```

BIBLIOGRAFIA

|1| AZEVEDO, Milton M. - O Subjuntivo em Português, Vozes, Petrópolis, 1976, 56 p.

|2| BACH, Emmnon - Teoria Sintática, Zahar, Rio, 1981, 305 p.

|3| BOBROW, Daniel G., KAPLAN, Ronald M., KAY, Martin, NORMAN, Donald A., THOMSON, Henry, WINOGRAD, Terry - GUS, A Frame -Driven Dialog System, Palo Alto, A.I.8, 1977, 155-173.

|4| BORGIDA, Alexander T. - Topics in the Understanding of English Sentences by Computer, University of Toronto, Technical Report nº 78, 1975, Department of Computer Science

|5| BRUCE, Bertram C. - A Model for Temporal References and Its Application in a Question Answering Program, University of Texas, A.I.3, 1972, 1-25.

|6| BRUCE, Bertram C. - Case Systems for Natural Language, BBN, A.I.6, 1975, 327-360.

|7| CAMARA, JR, Joaquim M. - Estrutura da Língua Portuguesa, Vozes, Petrópolis, 1977, 114 p.

|8| CHOMSKY, Noam - Aspectos da Teoria da Sintaxe, Armênio Amado, Coimbra, 1978, 372 p.

|9| CHOMSKY, Noam - Linguagem e Pensamento, Vozes, Petrópolis, 1977, 127 p.

|10| CODD, E.F. - Seven Steps to Rendezvous with the Casual User, North-Holland Publishing Company, 1974, 179-200.

|11| CUNHA, Celso - Gramática do Português Contemporâneo, Editora Bernardo Álvares, Belo Horizonte, 1978, 509 p.

|12| CUNHA, Celso - Gramática da Língua Portuguesa, Fename, Rio, 1980, 655 p.

|13| DAHL, Verônica - Logical Design of Deductive Natural Language Consultable Data Bases, Proc. 5ª International Conference on VLDB, Rio, 1979, 24-31.

|14| DATE, C.J. - An Introduction to Database Systems, Addison-Wesley, 1977, 536 p.

|15| FARIAS, Oscar L.M. de - Proposta de um Analisador Conceitual para Aplicação no Sistema de Ensino P Básico, Tese de Mestrado, PUC, Rio, 1977, 140 p.

|16| FILMORE, Charles J. - Toward a Modern Theory of Case, Reibel e Schane (eds), Modern Studies in English, New Jersey, Prentice-Hall, 1969, 361-375.

|17| FRAGA, Paltonio D. - Analises Morfológicas e Sintática da Língua Portuguesa num Projeto de Tradução Automática, Resumos 28ª Reunião Anual SBPC, Brasília, 1976.

|18| FRAGA, Paltonio D. - Análise e Síntese de Frases pelo Computador (Sistemas-Q), Relatório Interno nº 129, Unicamp/DCC/IMESS, Campinas, 1978, 29 p.

|19| FRAGA, Paltonio D. - Traduction Automatique Portugais - Anglais, Relatório Interno nº 130, Unicamp/DCC/IMECC, Campinas, 1979, 65 p.

|20| GREENE, Judith - Pensamento e Linguagem, Zahar, Rio, 1976, 172 p.

|21| GRIES, David - Compiler Construction for Digital Computer, Wiley, Toronto, 1971, 493 p.

|22| GROSS, Maurice - Modelos Matemáticos em Linguística, Zahar, Rio, 1972, 192 p.

|23| GUIDA, Giovanni, SOMALVICO, Marco - Interacting in Natural Language with Artificial Systems: The Donau Project, Information Systems, Vol. 5, London, 1980, 333-344.

|24| KAPLAN, Ronald M. - Augmented Transition Network as Psychological Models of Sentence Comprehension, A.I.3, North-Holland, 1972, 77-100.

|25| KATO, Mary A. - A Semântica Gerativa e o Artigo Definido, Ática, São Paulo, 1974, 186 p.

|26| KIMBAL, John P. - Teoria Formal da Gramática, Zahar, Rio, 1976, 158 p.

|27| KRAUSE, Jurgen - Natural Language Access to Information Systems. An Evaluation Study of Its Acceptance By End Users, Information Systems, London, Vol. 5, 297-318.

|28| LEHMANN, H. - Interpretation of Natural Language in an Information System, IBM Journal Res. Develop. Vol. 22, nº 5, 1978, 560-572.

|29| LOCKEMANN, Peter C. - Data Base User Languages for the Non -Programmer, Groos e Hartmanis, Lectures Notes in Computer Science, 1975, 183-212.

|30| MYLOPOULOS, John, BORGIDA, Alexander, COHEN, Philip, ROUSSOPOULOS, Nicholas, TSOTSOS, John, WONG, Harry - TORUS - A Natural Language Understanding System for Data Management, University of Toronto, Department of Computer Science, Toronto, 1975, 8 p.

|31| MYLOPOULOS, John, COHEN, Philip, BORGIDA, Alexander, SUGAR, Laszlo - Semantic Network and the Generation of Context, Department of Computer Science, University of Toronto, Toronto, 9 p.

|32| MYLOPOULOS, John, BORGIDA, Alexander, COHEN, Philip, ROUSSOPOULOS, Nicholas, TSOTSOS, John, WONG, Harry - Torus: A Step Towards Bridging the Gap Between Data Bases and the Casual User, Toronto, Information Systems, Vol, 2, 1976, 49-64.

|33| PARKISON, Roger C., COLBY, Kenneth M., FAUGHT, William S. - Conversational Language Comprehension Using Integrated Pattern-Matching and Parsing, A.I., 9, North-Holland, 1977, 111-134.

|34| PERINI, Mário A. - A Gramática Gerativa: Introdução ao Estudo da Sintaxe Portuguesa, Vigília, Belo Horizonte, 1976, 254 p.

|35| PLATH, W.J. - Request: A Natural Language Question - Answering System, IBM J.Res. Develop., July 1976, 326-335.

|36| PONTES, Eunice - Verbos Auxiliares em Portugues, Vozes, Petrópolis, 1973, 145 p.

|37| POTTIER, Bernard, AUDUBERT, Albert, PAIS, Cidmar T. - Estruturas Linguísticas do Portugues, Difel, São Paulo, 1975, 138 p.

|38| ROSA FILHO, Pedro L.da - Comunicação em Linguagem Natural com um Sistema de Ensino: Base de Conhecimento e Compreensão, Tese de Mestrado, PUC, Rio, 1976.

|39| SANDEWALL, Erik - Formal Methods in the Design of Question -Answering Systems, A.I., Nort-Holland, 2, 1971, 129-145.

|40| SANTOS, Antônio C. dos, SOUSA, Antônio C.G. de, MIYASATO, Beatriz Z., RICHTER, Gernot, SOUZA, Jano M. de, DANTAS, Jorge S., PINTO, Paulo R.B., D'ALBUQUERQUE, Vera L. - Projeto MINIBAN/COPPE - Especificação da Interface LOBAN, Relatório Técnico, vèrsão 2, COPPE/UFRJ, Rio, 1981, 650 p.

|41| SCHANCK, Roger C., RIEGER, Charles J. - Inference and the Computer Understanding of Natural Language, A.I. North-Holland, 5, 1974, 373-412.

|42| SCHUBERT, L.K. - Extending the Expressive Power of Semantic Network, A.I., North-Holland, 7, 1976, 163-198.

|43| SCHWARCZ, Robert M., BURGER, John F., SIMMONS, Robert F. - A Deductive Question - Answerer for Natural Language Inference, CACM, Vol. 3, nº 3, 1970, 167-183.

|44| SIBUYA, M., FUJISAKI, T., TAKAO, Y. - Noun-Phrase Model and Natural Query Language, IBM J.Res. Develop. Vol. 22, nº 5, 1978.

|45| SIMMONS, Robert F. - Natural Language Question-Answering Systems: 1969, CACM, Vol. 13, nº 1, 1970, 15-30.

|46| SOUSA, Antônio C.G. de, MIYASATO, Beatriz Z., SOUZA, Jano M. de, DANTAS, Jorge S., MEDEIROS, Sérgio P. da M, D'ALBUQUERQUE, Vera L., O Banco de Dados MICROBAN, 2º Congresso da SBC, 1982, Ouro Preto, 55-74.

|47| WALTZ, David L. - An English Language Question Answering System for a Large Relational Database, CACM, Vol. 21, nº 7, 1978, 526-539.

|48| WILKS, Yorich - A Preferential, Pattern-Seeking, Semantics for Natural Language Interface, A.I., North-Holland, 6, 1975, 53-74.

|49| WINOGRAD, Terry - Understanding Natural Language, Cognitive Psychology, Vol. 3, nº 1, 1972, 1-191.

|50| WOODS, W.A., KAPLAN, M., NASH-WEBBER, B. - The Lunar Sciences Natural Language System, Report nº 2378B, BBN, Cambridge, 1972.

|51| WOODS, W.A. - Transition Network Grammars fo Natural Language Analysis, CACM, Vol, 13, Nº 10, 1970, 591-606.

|52| ZIVIANI, Nivio - Subconjunto de Linguagem Natural para Consulta a Banco de Dados, Tese de Mestrado, PUC, Rio, 1976, 86 p.