UNIVERSIDADE NOVA DE LISBOA

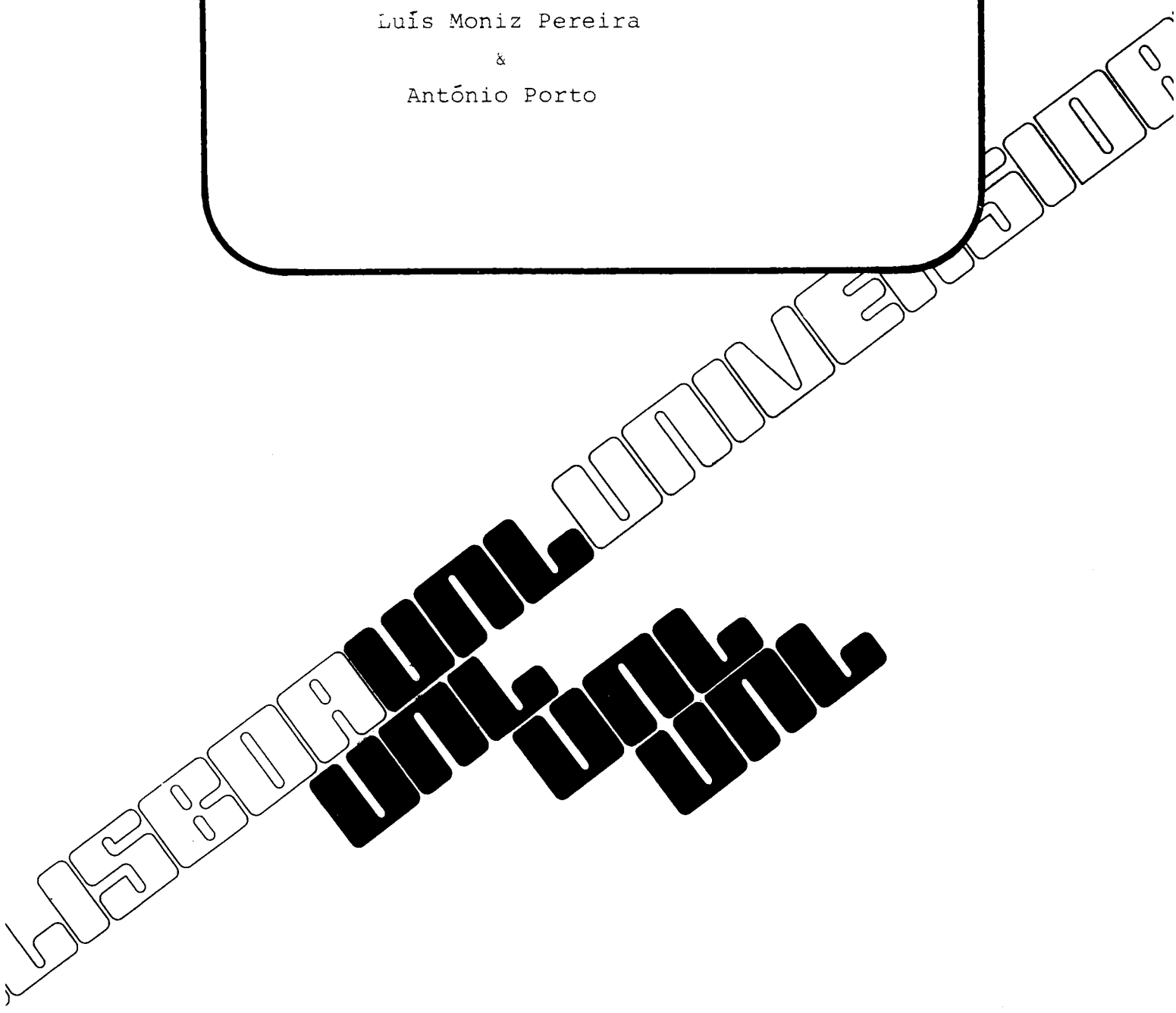
DEPARTAMENTO DE INFORMÁTICA

Interrogação em Português de um sistema de apoio ao planeamento

Luís Moniz Pereira

&

António Porto



SEMINÁRIO DOS OLIVAIS
QUINTA DO CABEÇO
OLIVAIS - LISBOA

Interrogação em Português de um sistema de apoio ao planeamento

Luís Moniz Pereira & António Porto

Departamento de Informática
Universidade Nova de Lisboa
Quinta da Torre
2825 Monte da Caparica
Portugal

Resumo

Descreve-se um sistema interrogável em linguagem natural (Português) para apoio ao planeamento do investimento em investigaçao, que foi completamente escrito em Prolog (incluindo a base de dados) e que corre num pequeno microcomputador.

São especialmente focadas as técnicas usadas para conseguir implementar um tal sistema numa máquina tão pequena.

Introdução

O sistema aqui descrito é um sistema interrogável em linguagem natural (Português) para servir de apoio ao planeamento do investimento em investigação em Portugal. O sistema tem conhecimento de interacções entre disciplinas científicas e objectivos de desenvolvimento, interacções apenas entre ciências e interacções entre objectivos. Os dados foram recolhidos em reuniões de âmbito nacional, seguindo um processo recomendado pela UNESCO [1][5].

Este sistema foi implementado usando uma versão ligeiramente modificada do RT-11 Prolog de Edinburgh (ver [2]) numa pequena máquina com um processador LSI-11/03, 64K *bytes* de memória central e um *drive* duplo de *diskettes* de densidade simples. Estas restrições do *hardware* infuenciaram consideràvelmente o projecto, mas uma versão razoàvelmente funcional foi no entanto conseguida.

O sistema é composto por dois módulos principais: o **interpretador de linguagem natural** e o **módulo de avaliação.**

O interpretador de linguagem natural subdivide-se ainda em dois sub-módulos: o **analisador lexical** e o **analisador sintáctico/semântico**. O analisador lexical aceita caracteres introduzidos a partir de um terminal e produz uma lista de elementos morfológicos, que são usados pelo analisador sintáctico/semântico para construir uma expressão de *goals* Prolog que corresponde à semântica da pergunta em linguagem natural.

O módulo de avaliação inclui as cláusulas usadas para avaliar a expressão Prolog que sai do interpretador de linguagem natural. Qualquer *subgoal* que involva accesso a disco é executado em dois passos: primeira há uma fase de **planeamento** em que um novo *subgoal* é produzido e depois esse novo *subgoal* é executado — só faz acesso às filas relevantes para o problema em causa, e a sua sequência de operações foi optimizada.

§1 O domínio

Este sistema trata de **ciências e objectivos de desenvolvimento.**

As ciências estão divididas em ramos (p.e. *ciências aplicadas*), cada um destes em grupos (p.e. *física*) e cada grupo em disciplinas (p.e. *óptica*).

Os objectivos de desenvolvimento dividem-se em grupos (p.e. *agricultura*), que por sua vez se dividem em objectivos básicos (p.e. *cereais*).

Há 110 disciplinas científicas e 78 objectivos básicos de desenvolvimento.

O sistema é suposto saber acerca de três tipos de correlações: entre ciências e objectivos, entre ciências e outras ciências, e entre objectivos e outros objectivos.[1] Qualquer categoria hierárquica pode ser referida, p.e. a *pertinência da física na agricultura* ou a *dependência do desenvolvimento na óptica.* As correlações são expressas em percentagens.

§2 O interpretador de linguagem natural

A competência linguística do sistema é obtida através de análises lexical e sintáctica/semântica, transformando uma frase em linguagem natural numa expressão de *goals* Prolog.

---

[1] As correlações são normalmente expressas como *dependência de A em B* ou *pertinência de B em A* (que representam a mesma coisa).

Os módulos que realizam esta transformação foram adaptados de uma gramática genérica de Português inicialmente desenvolvida para outra aplicação. Não vamos aqui entrar em pormenores; o leitor interessado é remetido para [3].[2]

2.1 Análise lexical

O nosso analisador lexical substitui palavras na frase de entrada pelas categorias lexicais correspondentes (nome, verbo, preposição, ...) com anotações sintácticas e semânticas. Isto é obtido fazendo cada palavra aceder ao **dicionário**. Se uma palavra não está no dicionário é assinalada ao utilizador como desconhecida.

Dado que o RT-11 Prolog apenas nos fornece indexação nos nomes dos predicados, o dicionário consiste num conjunto de cláusulas únicas para predicados cujos nomes são as próprias palavras que o sistema reconhece. Isto permite acessos rápidos.

A cada palavra válida, então, é associada uma cláusula do dicionário contendo toda a informação eventualmente necessária que diz respeito a essa palavra — categoria lexical, morfologia profunda e de superfície, género, número, etc.

Em geral a cada palavra corresponde uma entidade lexical, mas por vezes várias palavras são agrupadas para formarem uma única entidade, como por exemplo *ciências aplicadas*, que vai constituir um simples nome. Então, para além das mencionadas cláusulas para palavras simples temos também cláusulas que verificam se uma dada palavra é seguida por outras com as quais forma um nome único. Estas cláusulas, no entanto, são para um único predicado, a palavra entrando como argumento; o problema é que são necessários mais dois argumentos — a lista contendo o resto das palavras e a lista que é devolvida (com menos algumas palavras se serviram para formar um nome) — e se fôssemos uniformizar o dicionário todas as cláusulas teriam de ter os dois argumentos extra, o que seria poibitivo nesta máquina dado o número de cláusulas do dicionário e o tamanho da memória central disponível. (É indispensável ter **todo** o dicionário simultâneamente acessível.)

O vocabulário do sistema pode ser dividido em duas partes: um vocabulário **nuclear** e um **específico**.

O dicionário nuclear é independente do domínio particular do sistema, e subdivide-se numa parte **linguística** e numa parte **metalinguística**. A parte linguística contém determinadores (*o*, *uma*, ...), preposições (*de*, *para*, ...), as respectivas contracções (*aos*, *num*, ...), verbos comuns (*ser*, *ter*, ...), pronomes relativos e interrogativos (*que*, *quais*, ...), ligadores (*e*, *ou*) e expressões prelocutórias (*será que*, *é que*, ...). O vocabulário metalinguístico contém palavras usadas para inquirir o sistema acerca das suas capacidades linguísticas (*palavra*, *artigo*, *conjunção*, ...).

O vocabulário específico contém os nomes das disciplinas científicas e dos objectivos de desenvolvimento, assim como nomes, verbos e adjectivos usados para expressar as relações entre eles (*pertinência*, *depende*, *fundamental*, ...).

2.2 Análise sintáctica/semântica

A análise sintáctica/semântica é efectuada através de uma gramática nuclear que contém regras livres de contexto e regras sensitivas ao contexto (usando o formalismo das **gramáticas de cláusulas definidas**) com mecanismos de controle sintáctico e semântico. Estas regras manipulam as estruturas fundamentais de perguntas em língua Portuguesa, nomeadamente:

---

[2] Uma comunicação sobre esse sistema aparece também neste volume.

- perguntas sim-não;
- perguntas do tipo qu-;
- frases imperativas;
- cláusulas afirmativa, negativa, relativa, proposicional, coordenada, extraposta e elíptica;
- complementação e adjunção de nomes complexos;
- determinadores universal, existencial, cardinal, definido e indefinido;
- verbos comuns;
- nomes e verbos referentes à terminologia linguística.
  (permitindo perguntas acerca da competência linguística do sistema.)

Um grande esforço foi dedicado à incorporação de estruturas elípticas e extrapostas visto serem essenciais para que a interacção possa ser considerada natural, como por exemplo na seguinte sequência de perguntas:

- *Em relação à agricultura, qual a pertinência da física?*
- *E da química?*

Esta gramática nuclear é independente da sua aplicação e pode ser transportada para qualquer domínio, onde é completada como neste sistema por uma gramática específica que contém estruturas (essencialmente nomes e frases verbais) e controles semânticos referentes ao domínio.

Controles sintácticos verificam acordos de número e género, apontando quaisquer falhas ao utilizador.

As análises sintáctica e semântica não estão separadas mas entremeadas. Esta solução é melhor para poder parar imediatamente a análise mal um erro semântico é detectado, como acontece, por exemplo, quando aparece um complemento errado para um verbo; nesta eventualidade, o sistema informa sempre o utilizador de qual foi o problema. A junção das análises sintáctica e semântica por outro lado também produz uma gramática mais compacta, o que é bastante relevante na situação concreta deste sistema.

§3 A base de dados

A base de dados deste sistema tem duas partes distintas. Uma contém informação sobre as divisões hierárquicas das ciencias e dos objectivos, e a outra contém informação sobre correlações.

3.1 Descrição hierárquica

Cada definição hierárquica é uma cláusula da forma

$$<código>(<número>,[<nome1>,<nome2>,\ldots]).$$

onde $<código>$ é o código interno de algum grupo de ciências ou objectivos, $<número>$ é o número de elementos nesse grupo, e $<nome1>$, $<nome2>$, ... são os nomes desses elementos.

$<código>$ é usado como um nome dum predicado em vez de como argumento de um predicado genérico por razões de eficiência, como foi explicado na discussão do dicionário.

A escolha de códigos internos foi a seguinte: c é o código para *ciências* (como um todo), c2 é o código para o 2º ramo principal das ciências (*ciências aplicadas*), e c21 é o código para o 1º grupo dentro das ciências aplicadas (*física*); d é o código para o *desenvolvimento* (como um todo) e d1 é o código para o 1º grupo de objectivos de desenvolvimento (*agricultura*).

Uma cláusula de descrição hierárquica serve dois propósitos: pode ser usada na fase de avaliação para obter, através de $<número>$, os códigos dos elementos do grupo, ou pode ser usada na fase de resposta para obter os nomes através dos respectivos códigos.

A fila que contém a descrição hierárquica é consultada para memória central quando vai começar a avaliação, e aí permanece até a resposta ser produzida.

3.2 Correlações

A organização da forma de manipular informação sobre correlações era crítica em termos de exequibilidade e *performance* do sistema, envolvendo considerações de espaço e de tempo.

A informação de base relativa a correlações foi-nos fornecida sob a forma de três matrizes CD, CC e DD, contendo respectivamente as correlações entre disciplinas Científicas e objectivos básicos de Desenvolvimento, entre disciplinas Científicas e outras disciplinas Científicas, e entre objectivos básicos de Desenvolvimento e outros objectivos básicos de Desenvolvimento. Cada valor de correlação foi dado como um inteiro no conjunto {0,1,2,4}(de *irrelevante* a *fundamental*).

Dado o tamanho das matrizes (110×78, 110×110 and 78×78) e cada um dos seus elementos indo ser representado por uma cláusula, estava fora de causa ter num dado instante toda essa informação acessível em memória central. Foi portanto necessário subdividir as matrizes em sub-matrizes que podem ser individualmente consultadas; a escolha natural foi partir as matrizes ao longo das linhas de separação entre grupos, e é assim que temos, por exemplo, uma fila que contém as correlações entre as disciplinas da *física* e os objectivos básicos da *agricultura*.

Para que uma tal fila possa ser consultada quando é necessária, o seu nome deve estar relacionado com os grupos a que a fila diz respeito. De facto, escolhemos usar o nome que se obtém justapondo os códigos desses dois grupos — `c21d1` é o nome da fila contendo as correlações entre os elementos da física (`c21`) e da *agricultura* (`d1`).

O que é que cada uma destas filas contém exactamente?

Primeiro há uma cláusula definindo as dimensões da sub-matriz:

```
dim(c21d1,7,9).
```

Depois há uma cláusula que relaciona um predicado geral de correlação `cor` com um predicado particular de correlação que só é usado nesta fila (e que tem o mesmo nome que a fila):

```
cor(c21d1,X,Y,N/4) :- c21d1(X,Y,N).
```

Quando esta cláusula é efectivamente usada `X` recebe um inteiro que representa um elemento de `c21`, e `Y` recebe um inteiro que representa um elemento de `d1`. Note-se que o valor obtido para a correlação é dado na forma `N/4`, em que `N` é um inteiro no conjunto {0,1,2,4}; de facto todos os valores de correlações são representados internamente por um termo `N/D`, embora `D` nem sempre seja 4, e só na escrita da resposta é que um tal termo é convertido para uma percentagem.

A seguir na fila vêm cláusulas `c21d1` para valores *não-nulos* de correlações particulares, como

```
c21d1(4,7,2) :- !.
```

O *cut* é necessário por causa da última cláusula da fila, que é

```
c21d1(_,_,0) :- !.
```

Aqui o *cut* não é necessário, mas fornece um padrão uniforme para retirar cláusulas `c21d1` da memória central.

Este arranjo permite uma poupança considerável de espaço em disco, pois o valor de correlação mais vulgar nas matrizes é *zero*. A principal vantagem de ligar `cor` com `c21d1`, em vez de usar simplesmente `cor`, é que as cláusulas para `cor` têm quatro argumentos em vez de três, e portanto estamos a ganhar espaço em disco e em memória central quando a fila é consultada.

Esta técnica de subdividir as matrizes resolve o problema de espaço para as correlações, mas não resolve o problema do tempo de execução. Para ver qual o problema, debrucemo-nos um pouco sobre o modo como as correlações são calculadas.

Uma correlação entre dois elementos básicos (p.e. *óptica* e *cereais*) obtém-se por simples acesso a uma cláusula depois de ter consultado a fila apropriada.

Para achar uma correlação entre um elemento básico e um grupo (p.e. *óptica* e *agricultura*) é preciso consultar a fila correspondente, achar as correlações entre o elemento básico e **cada** elemento do grupo, e então calcular o valor médio. Isto implica uma chamada ao predicado `all`, que calcula **todas** as soluções que satisfazem um dado *goal* [4].

Agora para achar a correlação entre dois grupos (p.e. *física* e *agricultura*), tendo consultado as filas relevantes teríamos de achar, para **cada** elemento de um grupo, a correlação entre esse elemento e o outro **grupo**, e então calcular o valor final. Isto implica chamar `all` dentro de `all`, que pode levar algum tempo. Pior é achar a correlação entre um elemento básico e um super-grupo (p.e. *cereais* e *ciências aplicadas*), porque então dentro do `all` externo vamos ter, antes do `all` interno, que consultar uma fila diferente de cada vez. Casos ainda piores são fàcilmente imagináveis.

A solução é fazer algum pré-processamento de correlações de uma vez por todas, e guardar esses resultados em filas a que o sistema saiba aceder.

Pré-computar cada possível correlação estava absolutamente fora de causa, portanto havia que decidir que correlações pré-computar. Felizmente a natureza hierárquica e regular do domínio presta-se para uma solução elegante e eficaz.

Dizemos que o **nível** de um elemento é 0 se se trata de um elemento básico, 1 se é um grupo de elementos básicos, etc. Então a solução é ter uma fila para cada dois elementos não-básicos cuja diferença de níveis é **par**. Isto garante que para o cálculo de qualquer correlação simples apenas **uma** fila precisa de ser consultada e não mais que **uma** chamada a `all` precisa de ser executada. (É claro que perguntas complexas podem envolver várias correlações simples.)

As filas iniciais já estão de acordo com esta definição, e assim nenhuma fila necessita de receber tratamento especial por parte do sistema. Apenas temos de gerar as filas restantes a partir das iniciais. Os nomes destas filas continuam a ser obtidos por justaposição dos códigos dos grupos envolvidos. Dois "grupos" especiais têm no entanto de ser considerados, um tendo *ciências* como único elemento e o outro tendo *desenvolvimento* como único elemento. Demos-lhes os nomes, respectivamente, de `x` e `y`. Por exemplo, a fila `cy` contém as correlações entre os *ramos das ciências* e o *desenvolvimento*.

§4 Avaliação

A expressão que sai do interpretador de linguagem natural contém eventualmente chamadas para avaliar correlações. Tais chamadas têm todas a forma

```
correlacao(X,Y,V)
```

onde `X` e `Y` definem quais são os elementos cuja correlação se pretende e `V` espera receber o valor dessa correlação (na forma `N/D`).

Como foi dito antes, a avaliação de `correlacao` segue primeiro uma fase de **planeamento** antes de realmente consultar as filas relevantes e calcular o valor da correlação (usando `cor`).

A ideia genérica aqui é calcular em primeiro lugar tudo o que é determinístico, exceptuando chamadas que precisam de esperar por instanciações nos seus parâmetros (p.e. cálculos aritméticos), e então prosseguir com a avaliação dos *goals* adiados, tendo-os colocado na melhor ordem de execução.

Três tarefas principais são levadas a cabo na fase de planeamento:

- Obter os nomes das filas a consultar — isto é bàsicamente feito olhando para os códigos dos elementos cuja correlação se quer calcular: estes permitem calcular os respectivos níveis, a diferença de níveis mostra que tipo de fila é necessária e, consequentemente, os códigos que se têm de justapôr para obter o nome da fila são ou os dos elementos ou os dos grupos de que fazem parte, que são fàcilmente calculados a partir dos próprios.
- Obter *goals* para gerar os elementos de um grupo — se o predicado **all** vai ser usado isto torna-se necessário; a partir da cláusula de descrição hierárquica para o grupo, ou a partir da cláusula **dim** de uma fila respeitante ao grupo, pode-se aceder ao número **N** de elementos do grupo, e esta é a única informação necessária para construir um *goal* que gera inteiros entre 1 e **N**, que representam os elementos do grupo dentro de **cor**.
- Colocar *goals* na ordem correcta de execução — de entre os *goals* a serem adiados o sistema sabe quais é que produzem instanciações para serem usadas pelos outros, e assim pode optimizar a futura execução, colocando os *goals* na expressão correcta para tal.

Estas três tarefas são executadas concorrentemente. Longe de se usar um método geral de planeamento, escreveram-se cláusulas para **correlacao** que efectuam o planeamento de uma maneira altamente optimizada para as tarefas em questão.

§5 Performance do sistema

Para conseguir correr este sistema com a pequena quantidade de memória central disponível teve de se recorrer à técnica de encadear vários módulos que são portanto programas separados), usando uma fila em disco para passar informação entre eles. Isto foi conseguido tornando acessível dentro do Prolog a rotina do sistema RT-11 para encadear programas — o *goal* **chain**(<*savfile*>, <*consultfile*>) lança <*savfile*> que é um programa Prolog que inicialmente consulta <*consultfile*>.

Três módulos são encadeados: o **analisador lexical** (por causa do tamanho do dicionário), o **analisador sintáctico/semântico** e o **módulo de avaliação**, que encadeia novamente no analisador lexical para a próxima pergunta.

A principal consequência é que uma grande parte do tempo que o sistema leva a responder a uma pergunta é perdida no processo de carregar módulos do disco para a memória central.

Uma pergunta típica de 100 caracteres levas cerca de 16 segundos a ser respondida, assim distribuidos:

Análise lexical	3
Encadeamento	4
Análise sintáctica/semântica	2
Encadeamento	4
Avaliação (consultando 1 fila)	4

O sistema vai ser instalado num PDP-11/23, onde a extensão de memória, processador mais rápido e melhor acesso a disco vão certamente produzir um salto qualitativo na *performance* global. Em particular, os módulos a encadear poderão residir permanentemente em memória, tornando a operação de encadeamento virtualmente instantânea.

Agradecimentos

Este trabalho foi feito sob contrato com a Junta Nacional de Investigação Científica e Tecnológica (JNICT).

Agradecemos a John McCarthy por ter proporcionado a ocasião e condições para preparar este manuscrito na Universidade de Stanford.

Referências

[1] Caraça, J.M.G. ; Pinheiro, J.D.R.S. — *Prioridades em ciência e tecnologia — Identificação de áreas prioritárias para I&D*

Junta Nacional de Investigação Científica e Tecnológica 1981

[2] Clocksin, W. F. ; Mellish, C. S. — *Programming in Prolog*

Springer-Verlag 1981

[3] Pereira, L.M. ; Oliveira, E. ; Sabatier, P. — *An expert system for environmental resource evaluation through natural language*

submitted to First International Conference on Logic Programming, Marseille, 1982

[4] Pereira, L. M. ; Porto, A. — *All Solutions*

Logic Programming Newsletter, n. 2, Autumn 1981

[5] UNESCO — *Méthode de détermination des priorités dans le domaine de la science et de la technologie*

Études et documents de politique scientifique n. 40, UNESCO 1977