PARALELISMO NO TRATAMENTO DE LINGUAS NATURAIS

Paltonio Daun Fraga
DCEs - UFSCar - São Carlos, SP

E' justificado o paralelismo na aplicação de regras linguisticas (lexicais e gramaticais) por um transdutor de arborescências.

Estruturas Básicas de Dados

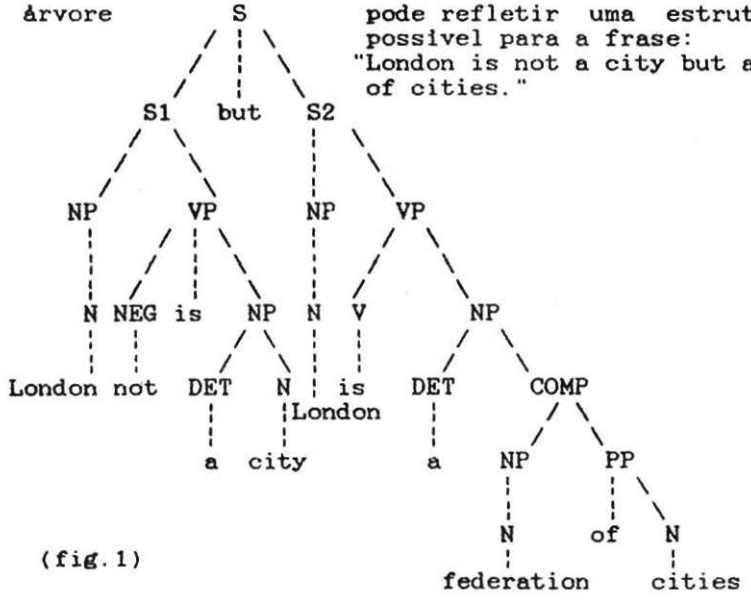
Um texto é uma sequência de caracteres ou de simbolos (palavras, abreviações, números, marcadores de fim de frase, de fim de parágrafo, pontuação forte e fraca, simbolos de prè-edição, etc)

A cadeia de caracteres é a estrutura sequencial mais simples tratada quase que diretamente pela maioria das linguagens de baixo e de alto nivel como um vetor de caracteres, ou por modelos ("patterns") em SNOBOL

A cadeia de simbolos ou lista é uma sequência de simbolos de tamanho e tipos heterogêneos e são melhor tratadas por linguagens especializadas como LISP, PROLOG e SNOBOL, podendo ser tratadas por PASCAL e com mais dificuldade em FORTRAN e BASIC.

A estrutura de árvore é uma estrutura bastante geral e adequada ao tratamento de linguas naturais, mas como apresentada classicamente ela só tem a geometria hierárquica e uma etiqueta de identificação em cada nó. Estas estruturas podem ser tratadas com certa facilidade em LISP e diretamente em PROLOG, confundindo-as com funtores.

A árvore pode refletir uma estrutura formal possivel para a frase:
"London is not a city but a federation of cities."



(fig.1)

A estrutura arborescente, utilizada no projeto GETA [BOITET, 1980], tem a geometria de árvore mas apresenta nós complexos decorados com uma lista de etiquetas (ou máscara), não podendo conter apontadores, mas sim referências a outros nós da árvore, o que permite seu tratamento como grafo do ponto de vista lógico, tomando-se o cuidado de manter a coerência da estrutura, evitando-se operações de apagamento e desdobramento de nós referenciados. Alguns exemplos serão mostrados nas figuras 4 e 5.

Nenhuma linguagem de programação de uso geral apresenta uma solução elegante e geral para o tratamento de arborescências, podendo ser simulados em PASCAL, LISP, PROLOG e suas sucessoras, mas em geral um software especializado é criado por cada grupo de pesquisa, com soluções "ad-hoc".

A estrutura de grafos é a uma das mais gerais, e apropriadas ao tratamento paralelo, no entanto não existem algoritmos economicos para o seu tratamento.

Estruturas de Dados e Algoritmos Paralelos

Poucas das linguagens de programação acima citadas apresentam modelos paralelos de processamento, casos especiais são as linguagens PROLOG [Clark, 1983] [Shapiro, 1983] e PASCAL concorrente.

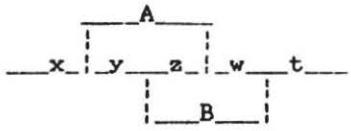
As estruturas de dados, em geral, podem ser tratadas por algoritmos paralelos, em especial as estruturas em árvore e grafos, pois a hierarquia intrínseca na estrutura fornece um critério homogêneo para o paralelismo.

Contrariamente, seja o reconhecimento de uma sub-cadeia (substring) numa cadeia de caracteres (string), é possivel a aplicação do paralelismo, mas onde começar um segundo "parsing", qualquer ponto adotado pode estar truncando uma sub-cadeia buscada, o que pode ser contornado, permitindo a superposição de dominios entre o primeiro e o segundo automato reconhecedor, se houver uma sub-cadeia prefixada da sub-cadeia procurada que está nos limites de busca.

Tais critérios podem ser perigosos, pois se houverem transformações a serem feitas nas sub-cadeias encontradas, o critério de transformação da sub-cadeia mais a esquerda é quebrado, o que impediria a aplicação segura de algoritmos paralelos, portanto numa estrutura tipicamente sequencial, com operações em que a sequência é importante, há complicações na aplicação de algoritmos paralelos. Neste caso é evidente que a busca paralela é possivel, mas sem transformação "in loco" da estrutura original.

A solução simples para um sistema de transformações implicaria numa cópia das sub-estruturas a serem tratadas em paralelo, e neste caso, a associação das suas transformadas pode implicar em dificuldades, possivelmente com encavalamento de sub-estruturas.

Tal situação se apresenta no SISTEMA-Q que é um transdutor de florestas de árvores [Colmerauer, 197Ø], [Fraga, 1978]



A -> y z

B -> z w

(fig.2)

Reconhecedores de árvores (por "pattern matching") programados eficientemente como uma composição de automatos reconhecedores de sub-cadeias [Chauché 1974] podem explorar o paralelismo.

Automatos analisadores e transdutores

Os analisadores são basicamente automata reconhecedores como os "parsers", bastante estudados na década de 7Ø e atualmente em cursos regulares de Linguagens Formais e Automata, nos currículos de pos-graduação em Ciência da Computação.

Os analisadores se caracterizam basicamente pelo fato de aceitar ou recusar uma cadeia de entrada num alfabeto, satisfazendo rigorosamente uma gramática expressa por meio de regras. Como sub-produto, pode resultar uma estrutura que retrata o encadeamento das regras que foram sendo aplicadas com sucesso no reconhecimento da cadeia de entrada.

Os transdutores são mecanismos que alteram a estrutura de entrada ou produzem estas transformações numa cópia da entrada, e se caracterizam pela modularidade e a possibilidade de criação , teste e manutensão dos módulos sem efeitos colaterais que são muito frequentes em analisadores, em especial, nos analisadores de gramáticas livres-de-contexto que em vista de um grande número de regras (da ordem de centenas e mesmo milhares para um modelo realista de lingua natural), novas regras acarretariam efeitos colaterais indesejáveis, dificeis de detectar e muito mais ainda de contornar, porisso esta metodologia tem sido abandonada, e foi bem aproveitada pelos ATN [Woods, 1972]. Os transdutores não são ensinados oficialmente em curriculos universitários.

Transdução

Uma transdução é a transformação de uma estrutura de entrada noutra estrutura de saida, em geral mudando algumas etiquetas e a geometria da estrutura, mas sem alterar seu tipo.

Um modelo de transdução de arborescências, do ponto de vista computacional, pode ser considerado como generalização de um modelo transformacional (tipo Chomskyano ou Melchukiano), que ao invés de fazer uma transformação num nivel (ou corte) da árvore, pode faze^-lo a qualquer nivel.

Do ponto de vista linguistico, não é necessário se fixar numa teoria linguistica, sendo extremamente fácil mostrar as transformações da passiva, translações, apagamento e geração de

novos elementos (que é feita, se necessário, no caso de elipse) [Chomsky 1964].

Uma sub-gramática G/P(E,S) pode aplicar em paralelo todas as suas regras na entrada E e obter S como saida, o automato sendo guiado pelo jogo P de parametros. Um parametro adicional de G seria o esquema de validação de nós na recursão.

Para as situações de conflito de aplicação de regras, como superposição de contextos e mesmo de elementos principais de uma transformação, certos critérios de solução de conflitos devem ser adotados, tais como: ordem na sequência, o contexto mais longo, e certos parâmetros de seleção de regras, como a mais a esquerda, a mais a direita, a mais abaixo, a mais acima, em geral critérios geométricos na estrutura suporte.

Um exemplo de sub-gramática é uma das que tratam Grupos Nominais Simples, que mostramos em forma BNF como gramática livre-de-contexto:

```
r1: GNS -> D Q N      determinador + qualificador
r2: GNS -> D N        determinador + nome
r3: GNS -> Q N        qualificador + nome
r4: GNS -> N          nome

r5: GNS2 -> S GNS     grupo nominal preposicionado
r6: GNS2 -> GNS       grupo nominal não-preposicionado
```

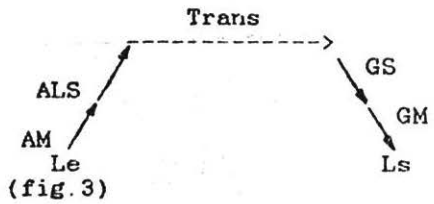
obs: um qualificador Q já é um grupo adjetival

Verifica-se facilmente a hieraquia das sub-gramáticas, e apesar das regras serem independentes, sugerindo paralelismo completo, vemos que há uma superposição de contextos, o que caracteriza uma situação de conflito, e tais situações podem ocorrer frequentemente, devido as ambiguidades,

Tradução Automatizada

Um sistema de TA apresenta as seguintes fases de processamento:

análise morfológica
análise lógico-semântica
tradução para uma forma canônica
geração sintática
geração morfológica



(fig.3)

Um texto de entrada após a análise morfológica é transformado numa arborescência por um automato de estados finitos não-determinístico. [Hopcroft, 1968], [Fraga, 1978, 1980]

Esta arborescência podendo ser enriquecida com informações para a próxima fase de análise sintática-lógico-semântica é submetida

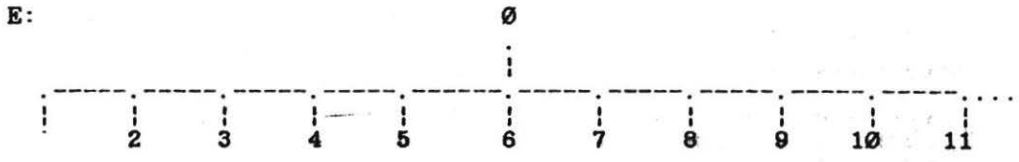
a uma gramática que é composta de sub-gramáticas organizadas em grafo, aqui já é possivel um nivel de paralelismo, desde que a estrutura a ser transformada satisfaça os critérios de entrada de várias sub-gramáticas subsequentes

Cada sub-gramática é uma sequência ordenada de regras que podem ser aplicadas em paralelo, fornecendo a cada passo da recursão uma arborescência de saida que é a entrada da próxima recursão ou de uma das próximas gramáticas dependendo de condições de entrada a serem satisfeitos

Um exemplo de transdução da gramática GNS acima sobre um parágrafo da língua inglêsa será mostrada a seguir:

"London is not a city but a federation of cities. It has grown from a paltry village where the Romans, found a convenient crossing place of the river Thames, into an octopus-like conurbation, embracing the greater part of Middlesex, Hertfordshire, Surrey, Essex, Kent, Buckinghamshire, and taking in Brighton, Reading and Lutton."

Após a análise morfológica, a estrutura será uma arborescência pouco profunda, como a seguir:



(fig.4)

```
Ø: [ul(TEXTO)]
   -- texto ou parágrafo
1: [occ(London), ul(London), cat(n), subn(p), num(s)]
   -- substantivo ou nome proprio singular
2: [occ(is), ul(be), cat(v), subv(vb), pess(3), num(s), temp(pres),
    modo(ind)] -- verbo conjugado, 3.a pessoa do singular do
    presente do indicativo do verbo "to be"
3: [occ(not), ul(not), cat(adjunto), suba(adv)]
   -- adverbio
4: [occ(a), ul(a), cat(deit), subd(artd), num(sin)]
   -- artigo definido singular
5: [occ(city), ul(city), cat(n), subn(c), num(sin)]
   -- nome comum singular
6: [occ(but), ul(but), cat(sub), subs(conj)]
   -- conjunção
7: [=(4)]                                    -- identico ao nó 4
8: [occ(federation), ul(federation), cat(n), subn(c), num(sin)]
   -- nome comum singular (derivado)
9: [occ(of), ul(of), cat(sub), subs(prep)]    -- preposição
10:[occ(cities), ul(city), cat(n), subn(c), num(plu)]
   -- nome comum plural
11:[occ(.), ul(.), cat(pont), subp(forte)]    -- pontuação
 ... e assim por diante ...
```

Cada nó, aqui numerado para facilitar sua referência apresenta uma decoração contendo informações pertinentes a esta fase do tratamento.

occ	ocorrência ou forma analisada
ul	unidade lèxica
cat	categoria gramatical (nome, adjunto, subordinante, pontuação, representante, etc)
sub	subcategoria gramatical subn(p,c), suba(adj,adv,...)
num	número (sin,plu)
pess	pessoa (1,2,3)

Outras informações sintáticas, semânticas e táticas podem ser enriquecidas em etapa posterior, a maioria dessas dependendo da UL e não da forma.

No texto em inglês, acima pode-se verificar a existência de vários grupos nominais simples. As regras r1..r6 podem ser aplicadas em paralelo, sem conflitos:

r1: GNS -> D Q N (the greater part) (a paltry village)

r2: GNS -> D N (a city) (a federation) (the Romans)

r4: GNS -> N (London) (Hertfordshire) (Surrey) (Essex) (Kent) (Buckinghamshire) (Reading) (Lutton)

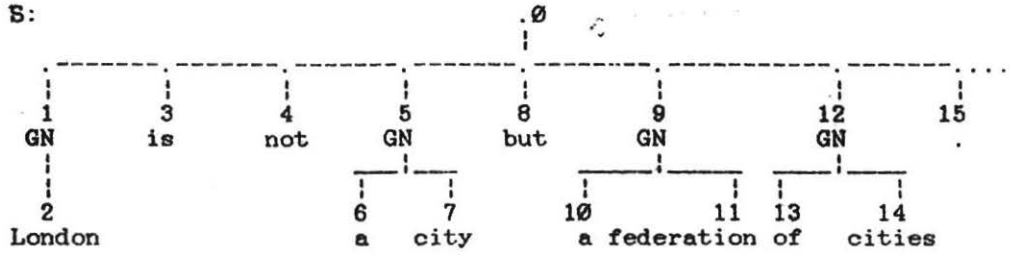
esta regra pode ser generalizada para captar pronomes pessoais.

outras regras deveriam tratar os casos seguintes:
GNS -> D Q PAPR N (a convenient crossing place)
GNS -> D Q N N (of the river Thames)

As regras da sub-gramática GNS2 são aplicadas na sequência, ou numa recursão sobre a sub-gramática GNS.

r5: GNS2 -> S GNS (of (cities)) (of (Middlesex)) (in (Brighton)) (into (an octopus-like conurbation))

Obtendo-se da aplicação das gramáticas GNS e GNS2:



(fig.5)

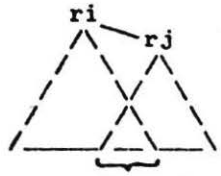
com novas etiquetas sintáticas, por exemplo:

k grupo sintagmático:
(grupo nominal, adjetival, frase verbal, relativa, etc)
fs função sintática:
(determinador, qualificador, governante, modificador, etc)

1: [k(gn), cat(n), subn(p), num(sin)]
2: [fs(gov), occ(London), ul(London), cat(n), subn(p), num(s)]
5: [k(gn), cat(n), subn(c), num(sin)]
6: [fs(det), occ(a), ul(a), cat(deit), subd(artd), num(sin)]
7: [fs(gov), occ(city), ul(city), cat(n), subn(c), num(sin)]
... e assim por diante ...

Os possiveis conflitos de aplicação das regras r1..r4 foram resolvidos adotando-se prioridade decrescente na sequência.

Este exemplo simples mostra a possibilidade de aplicação paralela de regras livres-de-contexto, mas o mesmo princípio pode ser aplicado em regras contextuais, desde que o contexto comum não seja alterado, ie, seja usado apenas para consulta. [BOITET 1980]



(fig.6)

Um exemplo mais realista de gramática é a de análise da língua portuguêsa, efetuada em Grenoble entre 1972 e 1978.(vide Anexo)

A gramática se apresenta como um grafo de sub-gramáticas, podendo ser testadas condições de entrada e saida de cada uma delas. Maiores detalhes na publicação [Fraga, 1978]

Geração sintática e morfológica

A fase de geração sintática, é efetuada pelo mesmo "software" transdutor de arborescência da análise e da transferência e pode utilizar de todo paralelismo possivel, mas ele se caracteriza pelo fato de se fazer por um modelo gerador descendente, pois se todo o processo estiver correto até aqui, a etiquetagem a nivel lógico-semântico está completa ou parcialmente correto. Será gerada a estrutura sintática de saida: ordem dos elementos, acordo em gênero, número e pessoa, tempos e modos verbais, etc.

A geração morfológica finalmente constroi as formas de saida e a geração de formas verbais, femininos e plurais. Esta fase é efetuada por um Automato de Estados Finitos Deterministico, como este mecanismo extremamente eficiente em tempo e em espaço, se caracterizando por algoritmos quasi-lineares, a busca de

paralelismo nestes algoritmos se torna desnecessária.

Conclusões

O paralelismo pode ser uma arma de aceleração do processo de análise com ferramentas de programação (linguagens, estruturas de dados e algoritmos) adequados, certamente no inicio da análise da arborescência pouco profunda, tanto em situações sem ambiguidades quanto no tratamento das ambiguidades que dependem do contexto para sua solução.

Quando todos os sintagmas elementares estiverem analisados, a árvore e' mais profunda e as situações em que o paralelismo pode atuar, fatalmente diminuem, exceto para regras atuando localmente, mas a construção de frases imbricadas (relativas, subordinadas, circunstantes) e finalmente da frase principal a partir dos sintagmas elementares, numa análise ascendente, depende cada vez mais do contexto, pois fenômenos como elipse, inversões, expressões idiomáticas complexas exigem o exame dos vários esquemas possiveis e o número de condições inter-nós a ser testadas è tambem maior. As ambiguidades estruturais reduzem a possibilidade de paralelismo.

Este trabalho pioneiro na América Latina, deve trazer resultados razoáveis a curto e medio prazos e extremamente favoráveis a longo prazo, podendo ser muito eficaz se fôr executado em cooperação com o projeto francês de Tradução Automatizada. E evidente a necessidade de se formalizar os modelos linguisticos e tratá-los matemática e computacionalmente, devido à grande massa de bancos de informações acessiveis democraticamente a todos os povos desenvolvidos, a barreira de linguas deve ser ultrapassada.

Referência Bibliográfica

Chomsky,N. - Aspects of the Theory of Syntax.
Cambridge, Mass. MIT Press, 1964.

Colmerauer,A. - Les Systèmes-Q ou un formalisme pour Analyser et synthetiser des phrases sur ordinateur.
Publ.43 Dept d'Informatique, Un. de Montreal, Montreal, 1970

Woods, W.A; Kaplan,R.M.; Webber,B. - The LUNAR Sciences Natural Language Information System. Final Report # 2378.
Cambridge Mass. BBN, 1972.

Aho, A.V.; Ullman,J.D. - The Theory of Parsing. Translation and Compiling. Prentice Hall, 1972.

Chauché,J. - Transducteurs & Arborescences - Etudes et réalisation de système appliquées aux grammaires transformationnelles. Thèse d'Etat GETA, USMG, jan. 1974.

Roussel,P. - Manuel de Reference et d'Utilisation

GIA, Université d'Aix Marseille, 1975.

Vauquois, B. - La Traduction Automatique à Grenoble
DUNOD, Paris, 1975

Fraga,P.D. - Análise Morfológica da Lingua Portuguêsa usando um Automato de Estados Finitos.
IMECC-UNICAMP, r 1.78, Campinas, Nov.1977

Fraga, P.D. - Análise Sintática da Lingua Portuguêsa
IMECC-UNICAMP, Campinas, marco 1978 - relatório não publicado

Fraga,P.D. - Análise e Síntese de frases pelo computador: Sistemas-Q, IMECC-UNICAMP, rel.129, dez.1978

Fraga,P.D. - Heuristiques en Traduction Automatique: Application à la Traduction du Portugais vers d'autres langues.
GETA, Grenoble, 1980

Boitet,Ch., Chatelin,P., Fraga,P.D. - Present and future Paradigms in the Automatized Translation of Natural Languages
COLING 80, Tokyo, 1980

Boitet,Ch. - Manipulation d'Arborescences et Parallelisme: Système ROBRA. GETA, USMG, Grenoble, 1980

Boitet, Ch. Traduction Automatisée au GETA: Principes, Applications, Evaluations et Exemples.
GETA, USMG, Sept. 1982

Clark,K. & Gregory,S. - PARLOG: A parallel Logic Programming Language, Research Report DOC 83/5, Dept of Computing, Imperial College, London, 1983

Shapiro,E.Y. - A Subset of Concurrent PROLOG and its Interpreter, Tech.Rep. TR-003, ICOT, Tokyo, 1983.

Shapiro,E.Y. & Takeuchi,A. - Object-Oriented Programming in Concurrent Prolog, New Generation Computing 1,1(1983)

B.Vauquois, Boitet,Ch. - Automated Translation at GETA
GETA, Grenoble, 1984

Vauquois,B. - The Organization of an Automated Translation System for Multilingual Translations at GETA. IBM Europe Institute, 1984

CRISS - PROLOG CRISS, une extension du langage PROLOG (v.4.0)
CRISS-Université II Grenoble, juillet 1985.

Anexo: Grafo de encadeamento de sub-gramaticas na Analise Logico-Semantica da Lingua Portuguesa