属性文法における属性の並列評価法に関する研究

2N－3

芳山和弘　原田賢一
慶應義塾大学理工学部

1　背景

近年、多くの分野において要求されてるソフトウェアは、年々巨大化、複雑化してきている。そのため、プログラムの長大化も進み、それに伴い、そのプログラムをコンパイルするのにも膨大な時間を要するようになってきた。そのような背景を経て、コンパイル時間を短縮できないかという研究が進んでいる。

2　目的

一般に、コンパイラでの処理を考える時、字句解析、構文解析、意味解析、コード生成の各フェーズがある。これらのフェーズの解析速度が上がればコンパイル時間そのものも短縮されるはずである。

そこで、関数的で並列化の可能性が高いと言われる属性文法 [1] を用い、構文解析フェーズと意味解析フェーズについて並列化を考える。

3　属性文法

属性文法は、文脈自由文法で表すのが難しい性質の静的意味を定式化する代表的な文法である。これは、文脈自由文法と静的意味を統合した定式化である。属性文法では、文脈自由文法の各文法記号に意味の情報を表すために属性というものを付加し、その属性に対する値の決めかたを、意味規則として生成規則に付随させる。

属性の値を計算することを、属性評価、属性評価をおこなうプログラムを属性評価器という。

また属性は、ただ一回だけ意味規則によって値が決められ、その後、その値は不変となる。これより属性文法は宣言的、関数的、あるいは単一代入的である性質を持つ。

4　属性の種類

属性は一般に**相続属性** (inherited attribute) か、**合成属性** (synthesizedattribute) のどちらかである。相続属性は、属性の値が、解析木の親または兄弟の節の属性の値から決まる属性をいう。合成属性は、属性の値が、解析木の子供の節の属性の値から決まる属性をいう。

Parallel Attribute Evaluation: structure of paser and evaluator.
Kazuhiro YOSHIYAMA, Ken'ichi HARADA
Keio University
3-14-1 Hiyoshi, Kouhoku-ku, Yokohama 223, JAPAN

5　属性文法の部分クラス

属性文法をコンパイラに適用する場合、属性文法に制限を加えて、効率良く属性評価を行うことが考えられ、様々な属性評価法と、それに対応した属性文法の部分クラスが提案されてきた。

5.1　一パス型属性文法

属性文法の部分クラスの内の一つに、一パス型の属性文法があり、そのクラスの属性文法としてＬ属性文法 [3, 2] や、ＬＲ属性文法がある。

5.1.1　Ｌ属性文法

Ｌ属性文法は、解析木を上から下、左から右へと一回訪問するだけで属性評価できる属性文法のクラスである。属性の依存関係に「右→左」のものがない。今回はこの属性文法のクラスを用いる。

Ｌ属性文法の特性として、相続属性と合成属性には、それぞれ、各部分解析木の入力と出力のパラメータとしての役割がある。この意味は、部分解析木の根の相続属性は部分解析木の中を見なくても評価できる事と、相続属性の値が一度評価されれば、その部分解析木の周りを調べなくても、完全に意味評価ができ、部分解析木の根の合成属性に、その結果が表れる事である。この部分解析木の処理は、プロセス間通信をなるべく最小にするという観点からプロセスを分割する有望な候補となる。

6　構文解析

ソースになる文法は通常逐次的である、そのため、コンパイラは、あるレベルでは逐次的に動作しなくてはならない。逐次的な部分はソースになる文法を並列な部分に切り分ける必要がある。逐次的な部分と並列な部分のやりとりはパイプラインで処理する事を考える。すなわち、逐次的下向き構文解析器と並列Ｌ属性評価器間をパイプライン処理する。

構文解析は先ず根の節から解析を始める、子の節を作り最も左の節の解析を行い。葉の節になるまで続ける。そして、解析木の深さの半分の所にある節 (図１の

斜線の節) を Next としてマークし、この節を根とする部分解析木を作り、それを意味解析器に渡し属性評価を行う。(図1の点線で囲まれた部分解析木1である。) この部分解析木を意味解析している間も、構文解析器では解析を行う。次の解析は先程マークした節 Next の兄弟の節または、最も近い親の兄弟の節から続け、この節を根として先程と同様に部分解析木を作り (図1の部分解析木1である。)、意味解析器に渡し、属性を評価する。この解析を全体の解析木が出来るまで続ける。

この構文解析の方法は、逐次的に解析木を上から下、左から右へと下向き構文解析をしている。

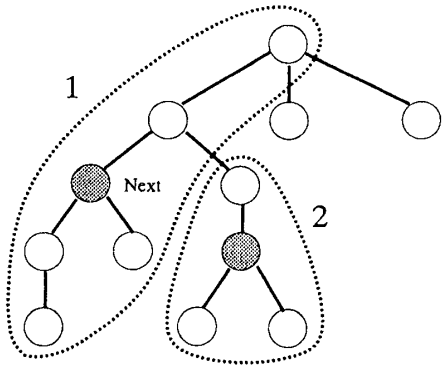

図 1: 解析木

7 意味解析

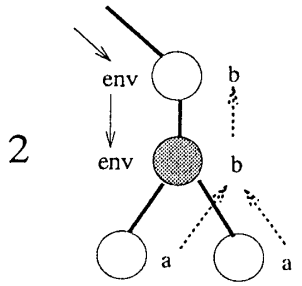

図 2: 部分解析木

構文解析器から渡された部分解析木それぞれについて、L属性の評価を行う。属性は木を上から下、左から右に節を訪問し評価する。また、他の部分解析木の属性から依存関係がある属性は、その属性値が与えられるまで待つ。つまり、属性文法は関数的であるので、その関数の引数が全てそろった時に属性を評価する。

例として、図2を見ると、属性 a,b は他の部分解析木からの依存関係がないので評価を行うことができるが、属性 env の評価については他の部分解析木の属性値を待たなければならない。

8 まとめと今後の課題

L属性文法の特性を利用することにより、並列に属性を評価でき、このことにより時間効率が向上したと考えられる。

今後の課題としては前述の下向き構文解析に、並列な意味解析を統合する際、更なる効率の向上のため、以下のことを考える事が重要である。

8.1 部分解析木

今回、構文解析で、Next と言うラベルを節に付け、その節より下の部分の木を完成させ、部分解析木として意味解析器に渡した。この部分解析木の大きさは、大きすぎても小さすぎても効率が悪くなると予想されるので、今回は木の深さの半分としたが、より効率を向上させるため、Next ラベルを付ける節の位置を解析的、または実験的に求める必要がある。

8.2 属性値の通信

意味解析において、部分解析木同士は属性値をやりとりしているので、この通信をなるべく減らす必要がある。この問題も Next をどの節に付けるかという事に深く関わって来る。

8.3 属性評価

与えられた属性文法で、属性の依存関係にサイクルがあるかどうかを調べる際用いられる属性の依存関係を図にした、属性依存グラフを使い、属性が逐次的に評価されなければいけない部分、また依存関係が独立で並列に評価できる部分を静的に求めることができれば、時間的効率の向上が得られると考えられる。

参考文献

[1] Hans-Juergen Boehm and Willy Zwaenepoel: Parallel Attribute Grammar Evaluation, in R. Popescu-Zeletin, G. LeLann, and K. H. Kim, editors, The 7th International Conference on Distributed Computing Systems, pages 347-354. IEEE, September 1987.

[2] Engelfriet J., File G.: The Formal Power of One-Visit Attribute Grammars, Acta Informatica 16,3 (November 1981), 275-302.

[3] Lewis P.M., Rosenkrantz D.J., Stearns R.E.: Attributed Translation, Journal of Computer and System Sciences 9 (1974), 279-307.