

依存文法と並列関係*

トーマス・ミヒヤエル・グロース

Abstract

Coordinate structures are among the most difficult structures to describe in any syntactic theory—although these structures are very common in every language. While early approaches in both the dependency and the constituency tradition have basically repeated Leonard Bloomfield's analysis of endocentric and exocentric structures, newer approaches in the 90's have emphasized the idea that coordinate structures are a procedural phenomenon. According to this insight, coordination is the repeated application of formation rules to further constituents. This approach technically works best within in the dependency framework. However, the proponents of the procedural approach also claim that as a procedural phenomenon, coordinate structuring cannot be represented for instance in dependency stemmata. In this paper, I will first show how the procedural approach works, and then, how dependency stemmata can be expanded into a third dimension to depict coordinate structuring.

「並列（関係）」というのは、二つ以上の成分をつなぐ道具で、コミュニケーションにとって、とても便利な道具である。例えば、太郎君が走っていて、花子さんが走っているという場面があれば、それ述べたいとき、二つの文

- (1) 太郎君が走っている。
- (2) 花子さんが走っている。

より、一つだけの文で表せればいだろう。そうすると、まず上の二つの文の共有した部分——その動詞——を一回削除できる。そして、名詞——「太郎君」、「花子さん」——の間に特別なことばを入れたら、一つの文が作れる：

* 安藤良太先生には全文に目を通していただき、日本語の表現を中心に適切なコメントと複雑な概念と述語に関する貴重な忠告をいただいた。ここに記して感謝にかえたい。

(3) 太郎君と花子さんが走っている。

文(1)と(2)これで作った新しい文(3)はどう違っているだろうか？ 意味的には、相違はないだろう。(1)と(2)を発言する代わりに、(3)を発言できるし、(3)を発言する代わりに、(1)と(2)を発言できるのである。

しかし、この言い方こそが重要なポイントを示すのである。並列関係の現象を説明するときにも、並列関係を利用しなければならない。私が使わなければならない並列関係は「(1)と(2)」という構造であった。これは一体なぜ使わなければならないだろう？——(3)という文を発言する者の言うのは、その者はまず(1)の意味、そして(2)の意味を発言して、または、まず(2)の意味、そして(1)の意味を発言するのである。その順序はあまり重要ではないが、(3)を発言する者は是非(1)のも(2)の意味も発言しなければならない。

それと簡単に確認できる方法がある。(3)を発言するときに、まず

(4) 太郎君が走っていますか？

を聞いてみて、それが正しければ、次に

(5) 花子さんが走っていますか？

を聞いてみる。それも正しければ、(3)も正しいが、(4)か(5)に対して「イエ」いう返事があれば、(3)も正しくない。

これが、「と」の論理的な意味なのである。勿論、日本語には、「と」だけでなく、ほかの並列関係を表すことばもある。例えば、「と」を代わりに「か」が出れば、確認方法が違う。

(6) 彼はコーヒーかお茶を飲みました。

(6)の場合には、まず

(7) 彼はコーヒーを飲みましたか？

を聞いてみて、正しければ、(6)が正しい。「イエ」いう返事があつたら、次に

(8) 彼はお茶を飲みましたか？

を聞いてみて、正しければ、(6)が正しい。しかし、(8)に対しても、「イエ」いう返事があれば、(6)は正しくない。これが、「か」の論理的な意味なのである。

しかし、日本語を使う人たちは、(3)という文を聞くと、(4)か(5)を聞くことがあまりない。それは、話し手はもう自分で上の確認を暗黙のうちに行い、文が正しいことを確認するのである。聞き手は自分で(3)のような文を聞くとときに、確認しなくてもいいという言語習慣

ができてからである。だから、便利な道具なのである。(1)と(2)と比べれば、(3)の便利さはその発言速度である。(3)は(1)と(2)より速く発言できるのである。しかし、この便利さには難点がある。(3)は(1)も(2)よりも複雑な構造になってしまう。

日本語なら、そして日本語の名詞なら、名詞間に助詞「と」を入れるのは一つの可能性である。品詞が変わると、別な並列道具を使うので、問題があまりない。その上、日本語の並列道具は接尾辞か副詞なので、並列構造は記述しやすい。(3)の場合には、並列している品詞は名詞なので、「と」が使われる。「と」は助詞なので、前の名詞につけなければならない。そのため、つける名詞の一つの部分となる。(3)を言語学的に形式してみると、

(9) Taroo.kun = to Hanako.san = ga hashit.te i.ru.

という形になる。そうすると、(3)も(9)も4つの成分から成り立っていることが分かる。そして、並列関係を指す「と」は最初の語の一部であることも分かる。

しかし、ヨーロッパ語では、品詞別の並列道具はないので、難しい並列構造もある。(3)か(9)を英語に訳したら、次の文になる：

(10) Taroo *and* Hanako are running.

そうすると、その確認方法は——「and」は「と」と同様な意味があるから——(3)とも同様である。まず、

(11) Is Taroo running?

を聞いてみて、その返事は「ハイ」なら、

(12) Taroo is running.

が正しい。「イエ」なら、(10)は正しくない。次に、

(13) Is Hanako running?

を聞いてみて、その返事は「ハイ」なら、

(14) Hanako is running.

が正しく、(10)も正しい。(13)に対する返事は「イエ」なら、(10)も正しくない。

これから、(1)と(2)または(12)と(14)、更に(3)と(10)を比較してみたら、重要な相違があることが分かる。

英語には、主語と述語の間に一致関係 (agreement) があるので、主語が単数なら、述語も単数形を取らなければならない。そのため、(12)と(14)の英語述語は「is」であった。しか

し、(10)には、(12)と(14)の主語が並列されたので、複数の主語になり、述語も複数形を取らなければならない。そのため、(10)の述語は「are」であった。不思議なのは、どの人間にも(10)ぐらいの構造は説明しなくても分かるはずだろう。しかし、文法的にはこの構造は分析にくいのである。なぜか？ まず、(1)と(12)の構造を見てみよう。両者は3つの成分から成り立っている。区別は、英語の述語は主語と継続形動詞の間にあり、日本語のは文末にあることである。依存文法の樹型図として示してみたら、次のようになる：

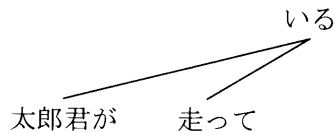


図1：文(1)

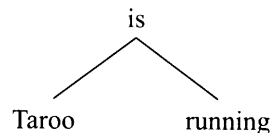


図2：文(12)

並列の(3)の構造は——「と」が助詞で前の名詞につけなければならないので——主語句の拡大だけである：

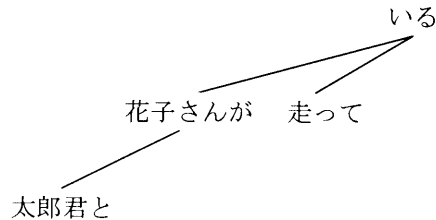


図3：文(3)

しかし、(10)の並列構造は一体どうなるだろう？ その並列主語は複数だが、主語が成り立っている名詞はそれぞれ単数なので、「Taro」と「Hanako」のどれを頭成分にするだろう？

伝統からみれば、どれも頭成分にされず、それより並列接続詞「and」を頭成分にすることになっている。そうすると、Bloomfieldの構造主義的な立場を取って見たら、(10)の構造は次のようになる：

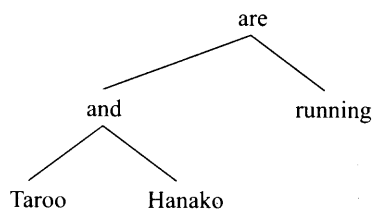


図4：文(10)

そのためには、「Taro」も「Hanako」も単数なので、述語「are」と一致できず、述語と一致するのは複数を表す並列接続詞「and」のみである。この記述の不満は、どの品詞を並列してもいつも接続句になってしまうことである。

もっと難しい問題は、名詞だけでなく、文を並列するときに出る。例えば、(10)の場合には、(12)と(14)の主語を並列して、複数の主語を作ったが、別な構造もある。そうすると、(12)と(14)を並列して、次の文となる：

(15) Taroo is running, and Hanako is running.

さらに、Bloomfield の立場を取ると、(15)は図4のような構造があるだろう：

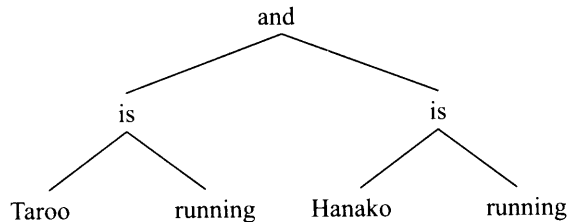


図5：文(15)

(10)の主語と同様に、(15)は文ではなく、接続句である。そして、(15)の后者の文——(14)と同様——の動詞を削除し、その代わりに「too」を置換えることもある：

(16) Taroo is running, and Hanako, too.

(16)のような構造は「gapping」といわれ、最後の文で動詞が削除されるからである。Gappingは多分自然言語の一番記述しにくい現象である。(16)の前者の文は(12)と同様な構造であり、図2と同様である。しかし、「and Hanako, too」の構造はどうだろう？もし、図4に示した構造なら、「Taroo」は「and Hanako, too」の並列接続詞に依存し、非連続性を起してしまうのである。それは好ましくないだろう。更に、図5に示した文の並列なら、前者の述語「is」は並列接続詞に依存し、ほかの述語はないので、並列構造は不完全となってしまうのである。

Gappingの記述から、新しい考え方が90年代に生まれた。特にLobin(1993)は次の概念を提案し、そのときまでの概念をうまくまとめた。並列という現象を構造的で静的な現象を見るよりも、これを動的な過程を見るようになってきている。動的な過程なので、この立場を取っている言語学者は並列関係を樹型図などで代表できないことを強調している。私は、前者の概念が正しく思っているが、後者の結果が少し速すぎてしまったのではないかと思っている。

並列関係を動的な過程を見るとは、一体どういう意味だろう？この考え方は特に自然言語の計算機用言語処理を中心した言語学者の中で生まれたので、言語をまず規則通りの制度的な産出過程を見る。(12)という文を作るために、様々な規則に従わなければならない。例えば、主語と述語の一致関係はそれらの規則の一つである。並列関係はある文を作るための規則を、次の文を作るために、もう一度繰り返すことである。ある規則をもう一度繰り返して別な構造に適用するのは、「再実行」と名づける。この立場から見ると、並列接続詞の過程

的な機能は、再実行の開始場所を示すだけである。例えば、(10)をみると、「Taroo and Hanako」の主語では、「Taroo」だけでも主語で、主語の産出に必要な規則通りに作られ、その規則をもう一度「Hanako」に適用する。並列接続詞「and」の機能は、「これから：今まで適用した規則を前の名詞と同様なことばにもう一度適用せよ！」というのである。同じ規則を少なくとも二度適用するので、主語は複数形となり、従って述語も複数形を取らなければならない。しかし、動的な過程の立場から見ると、一致関係をやり直すのは、並列過程と全然関係がなく、一般的な文法規則なのである。

さて、並列過程は同じ情報をもっと速く通じさせることで省略のための道具なので、省略は要らない成分の削除となる。文(10)の場合には、削除できる成分はなく、再実行した過程は最初のことばから開始するからである。

Gapping の場合には、削除が必要になる。(16)は元に(12)と(14)の意味を含めており、(12)という前者の構造が産出された後、(12)を作るための規則もう一度(14)に適用しなければならないのは、並列接続詞の合図なのである。しかし、本当にそうすると、動詞句も再び産出されてしまい、これは——発言時間を縮めるため——好ましくないので、前者と後者の文のまったく同じ成分を作るための規則は二度目の過程で省略され、というのはその規則で作られる成分は削除され副詞「too」と置き換えられる。

Gross (1999) では、文を作る規則を紹介し、これを今の問題に適用してみよう。まず、(12)は次の規則で作れる：

- (17.1) S {文} ⇒ NP V AP
 (17.2) NP {名詞句} ⇒ Taroo
 (17.3) AP {形容詞句} ⇒ running
 (17.4) V {動詞} ⇒ is

文(14)を規則(17)により産出された(12)につなぐときには、二度目の cycle を産出するわけなので、そのときに同様な成分を削除し、これを「too」と置き換える。簡単に言えば、(16)を産出するために、まず(17)を全部(12)に適用し、そして、(17.2)だけに「Taroo」を代わりに「Hanako」に、(17.3)を「too」に置き換え、(17.4)を削除して(14)に適用するのである。それが並列関係の便利さで、並列関係の情報伝達の速さの理由である。

ほかの例文を見ながら、並列関係を動的な過程と見なす方法も少し深めてみよう。そのために、Gross (1999) で紹介した「帯 (zone)」という概念も利用する。

- (18) Peter gave Mary two cookies, and Joan three.

文(18)の前者の文の帯構造は

(19) [Peter_⊗] gave [[Mary_{VAL1}] [two cookies_{VAL2}] _⊙]

「Peter」は述語「gave」と一致しているので、遠帯 (remote zone) に置き、「Mary」と「two cookies」は一致しないので、近帯 (close zone) に置く。「Mary」は間接目的語なので、述語に直接目的語より近く置かなければならぬ。「VAL1」という場所に置き、直説目的語「two cookies」は「VAL2」に置く。英語の名詞句にも遠帯と近帯があり、冠詞は遠帯に、形容詞は近帯に置く。そうなら、(20)の VAL2の構造は

(20) [_⊗] [two_⊙] cookies

「two」は形容詞なので、近帯に置き、遠帯にことばはないので、空となる。

規則(17)を帯構造として形式すると、(20)を作る規則は次のようとなる：

- (21.1) S ⇒ [_⊗] V [[VAL1] [val2] _⊙]
 (21.2) _⊗S {遠帯} ⇒ NP_⊗
 (21.3) _⊙S {近帯} ⇒ VAL1 VAL2
 (21.4) VAL1 ⇒ NP_{VAL1}
 (21.5) VAL2 ⇒ NP_{VAL2}
 (21.6) NP_⊗ ⇒ N_⊗
 (21.7) NP_{VAL1} ⇒ N_{VAL1}
 (21.8) NP_{VAL2} ⇒ [_⊙] N_{VAL2}
 (21.9) _⊙N-VAL2 ⇒ A
 (21.10) V ⇒ gave
 (21.11) N_⊗ ⇒ Peter
 (21.12) N_{VAL1} ⇒ Mary
 (21.13) N_{VAL2} ⇒ cookies
 (21.14) A {形容詞} ⇒ two

重要な規則は語彙的な規則(21.10–14)なので、今構造的な規則(21.1–9)を無視してもよい。そうすると、(18)の後者の文を作るためには、(21.12)に「Mary」を代わりに「Joan」に、(21.14)に「two」を代わりに「three」に置き換え、ほかの規則を無視しその成分を削除するだけである。そうすると、次のようとなる：

- (21'.10) V ⇒ gave |=(21.10) → 削除
 (21'.11) N ⇒ Peter |=(21.11) → 削除
 (21'.12) N_{VAL1} ⇒ Mary |≠(21.12) → Joan

(21'.13) N_{VAL2} ⇒ *cookies* | = (21.13) → 削除

(21'.14) A ⇒ *two* | ≠ (21.14) → *three*

→ Peter gave Mary two cookies, and Peter gave Joan three cookies.

次の文には、gapping 以外「left-deletion」という現象も出る。

(22) Peter gave Mary two, and Joan three cookies.

Left-deletion というのは、後の文を作る前には少なくとも一つの前の文を作る規則を変更し、そして後の文を産出する。(22)には、前の文を作るときに、規則(21.13)ぐらいの規則を実行しなければならないが、後の文を産出する前には、この規則を取り消し、名詞を削除する。しかし、そうすると、後者の文をつなぐときに、(21.13)を実行しなければならない。

(23.1) S ⇒ [®] V [[VAL1] [val2] ©]

(23.2) ®_S {遠帯} ⇒ NP_®

(23.3) ©_S {近帯} ⇒ VAL1 VAL2

(23.4) VAL1 ⇒ NP_{VAL1}

(23.5) VAL2 ⇒ NP_{VAL2}

(23.6) NP_® ⇒ N_®

(23.7) NP_{VAL1} ⇒ N_{VAL1}

(23.8) NP_{VAL2} ⇒ [©] N_{VAL2}

(23.9) ©_{N-VAL2} ⇒ A

(23.10) V ⇒ *gave*

(23.11) N_® ⇒ *Peter*

(23.12) N_{VAL1} ⇒ *Mary*

(23.13) N_{VAL2} ⇒ *cookies*

(23.14) A ⇒ *two*

まず、前の文には規則(23.13)を取り消す：

(23.13') N_{VAL2} ⇒ *cookies* | ≠ (23.13) → 削除

そして、再実行の規則は次のようにある：

(23'.10) V ⇒ *gave* | = (23.10) → 削除

(23'.11) N ⇒ *Peter* | = (23.11) → 削除

(23'.12) N_{VAL1} ⇒ *Mary* | ≠ (23.12) → *Joan*

(23'.13) N_{VAL2} ⇒ — | ≠ (23.13) → *cookies*

(23'.14) A ⇒ *two* |≠ (23.14) → *three*
 → Peter gave Mary ~~two cookies~~, and Peter gave Joan three cookies.

つまりどのぐらい複雑な文を産出しても、文末に届くまでは、産出規則を完全に実行しなければならない。

次の文はまた単純な *gapping* であるが、この場合には、動詞以外どの名詞句も置き換える必要があるが、これから構造的な規則を無視してはいけなくなる。

(24) Peter read a book in his room, and Mary a letter in the barn.

- (25.1) S ⇒ [⊗] V [[VAL] [ADJ] ©]
 (25.2) ⊗ ⇒ NP_⊗
 (25.3) © ⇒ VAL ADJ
 (25.4) VAL ⇒ NP_{VAL}
 (25.5) ADJ {自由成分} ⇒ PP
 (25.6) NP_⊗ ⇒ N_⊗
 (25.7) NP_{VAL} ⇒ [DET_⊗] [©] N_{VAL}
 (25.8) PP_{ADJ} {前置詞句} ⇒ P NP_{ADJ}
 (25.9) NP_{ADJ} ⇒ [PRO_⊗] [©] N_{ADJ}
 (25.10) V ⇒ *read*
 (25.11) N_⊗ ⇒ *Peter*
 (25.12) N_{VAL} ⇒ *book*
 (25.13) DET {定冠詞} ⇒ *a*
 (25.14) P {前置詞} ⇒ *in*
 (25.15) N_{ADJ} ⇒ *room*
 (25.16) PRO {代名詞} ⇒ *his*

文(24)を見てみると、動詞だけを削除してじゅうぶんであることが分かる。そうすると、規則(25.10,11)を代わりに次の規則を利用するのが適当であろう：

- (25'.10) V ⇒ *read* |= (25.10) → 削除
 (25'.11) N_⊗ ⇒ *Peter* |≠ (25.11) → *Mary*

しかし、(21'.10-14)や(23'.10-14)のように次の規則は実行してはいけない：

- (25'.12) N_{VAL} ⇒ *book* |≠ (25.12) → *letter*
 (25'.13) DET ⇒ *a* |= (25.13) → 削除

- (25'.14) P ⇒ *in* |= (25.14) → 削除
 (25'.15) N_{ADJ} ⇒ *room* |≠ (25.15) → barn
 (25'.16) PRO ⇒ *his* |≠ (25.16) → the

→ * Peter read a book in his room, and Mary a letter in the barn.

つまり、そうすると、次の正しくない文になってしまうのである：

(24') * Peter read a book in his room, and Mary letter the barn.

今まで利用した語彙的な規則の再実行は便利であったのに、一体なぜ(24)に適用できないだろう？ まず、定義を上げよう：もし、ある構造には遠帯があれば、この構造を「サイクル (cycle)」と名づける。この定義によると、まず、どの文でもサイクルである。そして、名詞句でもサイクルである。サイクルという概念は特に複雑な文の成分の移動に関して重要な術語である。特に名詞句的なサイクルから成分を移動させることに対して厳しい制限がある。例えば、次の文では、

(26) 彼は今日買ったばかりの本をすでに読み終わった。

目的語を前に移動させてもよいが、そうするとき、目的語から成り立っている成分を全部移動させなければならない：

(27) [今日買ったばかりの本は] 彼がすでに読み終わった。

しかし、部分的な移動は禁止される：

(28)* [本は] 彼が [今日買ったばかりの] 読み終わった。

(29)* [今日買ったばかりの] 彼が [本を] 読み終わった。

つまり、サイクルというのは様々な現象に非常に弱いものである。この中には、並列現象は一つである。

さて、(24)の並列関係は二つの文の間に働いているが、その文の成分の間には働いていない。つまり、「Peter」は「a book」だけを読むので、「a letter」は読まないのである。そして、どの名詞句でもサイクルであるので、複雑な名詞句なら頭成分が同様であれば、規則再実行時には削除してもよい。しかし、ほかの場合には、削除してはいけない。そのためには、次の規則は実行してはいけない：

- (25'.12) N_{VAL} ⇒ *book* |≠ (25.10) → letter
 (25'.13) DET ⇒ *a* |≠ (25.10) → 削除

そうするよりも、次の規則を再実行すればよい：

(25'.4) VAL ⇒ NP_{VAL} = [*a book*] ≠ (25.10) → [*a letter*]

PP の場合には、前置詞は名詞句に格を与えるので、そのものを削除してしまうと残る名詞句は格がなくなってしまう。そのため、次の規則を再実行すればよい。

(25'.5) ADJ ⇒ PP = [*in his room*] ≠ (25.10) → [*in the barn*]

しかし、もっと一般的な規則があるので、それを(25'.10,11)とともに再実行すればよい：

(25'.3)© ⇒ VAL ADJ = [[*a book*] [*in his room*]] ≠ (25'.3) → [[*a letter*] [*in the barn*]]

(25'.10) V ⇒ *read* | = (25.10) → 削除

(25'.11) N_® ⇒ *Peter* | ≠ (25.10) → *Mary*

→ *Peter read a book in his room, and Mary read a letter in the barn.*

まとめると、次のメタ規則となる：

- i. サイクルを直接支配する成分は、——並列関係がそのサイクル間に働かないのなら——削除してはいけない。
- ii. 並列しないサイクルの頭成分については産出規則を取り消してもよいが、制限としてその頭成分は文に少なくとも一回産出しなければならないし、削除しようとする成分と同様でなければならない。
- iii. サイクルの頭成分以外の成分は必ずしも削除してはいけない。

i によると、(24)の前置詞句の頭成分——その前置詞——は削除してはいけないが、(24)の動詞は削除してもよい。つまり、動詞は並列する文の頭成分なのであるが、前置詞句は並列しないのである。

ii によると、(18)と(22)には規則を再実行するときに、「cookies」を削除してもよい。

iii によると、(24)の同様な冠詞は削除してはいけない。

さて、ドイツ語のあいまいな文を見てみよう：

(30) *Peter hörte einen Schuss in seinem Zimmer, und Maria im Hof.*

“Peter heard a shot in his room, and Mary in the yard.”

文(30)には、意味が二つある。まず、「Peter は自分の部屋で、Maria は中庭で銃声を聞いた」という意味がある。一方、「Peter は自分の部屋で銃声と Maria が庭の中にいたと聞いた」

という意味もある。前者の意味では、(30)を *gapping* の構造として解釈し、後者の意味では、近帯の並列関係という解釈をする。

前者の意味は、(24)と同じである：

- (31.1) S ⇒ [⊗] V [[VAL] [ADJ] ©]
 (31.2) ⊗ ⇒ NP_⊗
 (31.3) © ⇒ VAL ADJ
 (31.4) VAL ⇒ NP_{VAL}
 (31.5) ADJ ⇒ PP
 (31.6) NP_⊗ ⇒ N_⊗
 (31.7) NP_{VAL} ⇒ [DET_⊗] [©] N_{VAL}
 (31.8) PP_{ADJ} ⇒ P NP_{ADJ}
 (31.9) NP_{ADJ} ⇒ [PRO_⊗] [©] N_{ADJ}
 (31.10) V ⇒ *hörte*
 (31.11) N_⊗ ⇒ *Peter*
 (31.12) N_{VAL} ⇒ *Schuss*
 (31.13) DET ⇒ *einen*
 (31.14) P ⇒ *in*
 (31.15) N_{ADJ} ⇒ *Zimmer*
 (31.16) PRO ⇒ *seinem*

そして、再実行の規則は次である：

- (31'.2) ⊗ ⇒ NP_⊗ = [*Peter*] |≠ (31.2) → [*Maria*]
 (31'.4) VAL ⇒ NP_{VAL} = [*einen Schuss*] |= (31.4) → 削除
 (31'.5) ADJ ⇒ PP_{ADJ} = [*in seinem Zimmer*] |≠ (31.5) → [*im Hof*]
 (31'.10) V ⇒ *hörte* |= (31.10) → 削除

→ Peter hörte einen Schuss in seinem Zimmer, und Maria hörte einen Schuss im Hof.

後者の意味でも、(31.1-16)は有効だが、再実行の規則には(31''.5)が同じだが、(31''.2)と(31''.4)は(31'.2)と(31'.4)と逆となる：

- (31''.2) ⊗ ⇒ NP_⊗ = [*Peter*] |= (31.2) → 削除
 (31''.4) VAL ⇒ NP_{VAL} = [*einen Schuss*] |≠ (31.4) → [*Maria*]
 (31''.5) ADJ ⇒ PP_{ADJ} = [*in seinem Zimmer*] |≠ (31.5) → [*im Hof*]
 (31''.10) V ⇒ *hörte* |= (31.10) → 削除

→ Peter hörte einen Schuss in seinem Zimmer, *und Peter hörte Maria im Hof.*

ほかのあいまいなドイツ語の文を見てみよう：

(32) Liegt das Buch auf oder unter dem Tisch?
 “Is the book on or under the table?”

文(32)は Lobin (1993: 101) が上げたもので、意味が二つあるのである。発音により、二者択一疑問文になったり決定疑問文になったりするという意味が変わる。例えば、前置詞「auf」にアクセントを置き、その後短いポーズを入れれば、決定疑問文になり、返事として

(33.1) (Das Buch liegt) auf dem Tisch.
 “(The book is) on the table.”

または

(33.2) (Das Buch liegt) unter dem Tisch.
 “(The book is) under the table.”

があるだけである。アクセントもポーズもないと、二者択一疑問文になり、返事として

(34.1) Ja, (das Buch liegt) auf dem Tisch.
 “Yes, (the book is) on the table.”

(34.2) Ja, (das Buch liegt) unter dem Tisch.
 “Yes, (the book is) under the table.”

または

(34.3) Nein, (das Buch liegt) auf dem Stuhl.
 “No, (the book is) on the chair.”

などが可能である。決定疑問文——いわゆる(33)の解釈——の場合には、まず(33.1)のような完全な文の規則は次のようになる：

- (35.1) S ⇒ V [⊗] [c]
 (35.2) ⊗ ⇒ NP_⊗
 (35.3) © ⇒ VAL
 (35.4) NP_⊗ ⇒ [DET_⊗] [c] N_⊗
 (35.5) VAL ⇒ PP
 (35.6) PP ⇒ P NP_{VAL}

- (35.7) NP_{VAL} ⇒ [DET_⊗] [c] N_{VAL}
 (35.8) V ⇒ *liegt*
 (35.9) N_⊗ ⇒ *Buch*
 (35.10) DET_⊗ ⇒ *das*
 (35.11) P ⇒ *auf*
 (35.12) N_{VAL} ⇒ *Tisch*
 (35.13) DET ⇒ *dem*

決定疑問文は二つの文の並列構造なので、gapping があるが、left-deletion もある。そのときに、(35.12)と(35.13)を変更することになる：

- (35.12') N_{VAL} ⇒ *Tisch* |=(35.12) → 削除
 (35.13') DET ⇒ *dem* |=(35.13) → 削除

並列するときの再実行は次の規則に適用する：

- (35'.8) V ⇒ *liegt* |≠(35.8) → 削除
 (35'.9) N_⊗ ⇒ *Buch* |≠(35.9) → 削除
 (35'.10) DET_⊗ ⇒ *das* |≠(35.10) → 削除
 (35'.11) P ⇒ *auf* |≠(35.11) → *unter*
 (35'.12) N_{VAL} ⇒ — |≠(35.12') → *Tisch*
 (35'.13) DET ⇒ — |≠(35.13') → *dem*

→ Liegt das Buch auf dem Tisch oder liegt das Buch unter dem Tisch?

さて、二者択一疑問文の解釈——いわゆる(34)の意味——とどう違うだろうか？ 殆ど同じだが、(35'.8-10)は再実行せず、(35.12', 13')と(35'.11-13)だけを再実行することである：

- (36.12') N_{VAL} ⇒ *Tisch* |=(36.12) → 削除
 (36.13') DET ⇒ *dem* |=(36.13) → 削除
 (36'.11) P ⇒ *auf* |≠(35.11) → *unter*
 (36'.12) N_{VAL} ⇒ — |≠(35.12') → *Tisch*
 (36'.13) DET ⇒ — |≠(35.13') → *dem*

→ Liegt das Buch auf dem Tisch oder unter dem Tisch?

最後に、並列関係が動的な過程なので、樹型図などで代表できないという主張に戻ってみよう。この主張は極端なので、こんなに極端な主張にとって適当な証拠を求めてもよいではないかと思っている。証拠を上げるのは議論説という哲学的な分野の一つであり、難しく思

われている。しかし、よく忘れられてしまう方法は解決を作ってみせるということである。これから、並列関係を樹型図で代表する方法を作ってみせたい。

Gross (1998) では、依存文法の樹型図の作り方を述べた。樹型図とは一体何だろう？ 樹型図の機能は文の構造を図形として明らかにする手段である。しかし、言語が1次元の現象なのに、樹型図は2次元のものである。そのためには、樹型図の第1次元には、発言時間を表し、第2次元には、発言の階級的な構造を表す。後者の次元を正確に表すためには、前もって依存関係という概念を定義しなければならない。もう一度、図1を見てみよう：

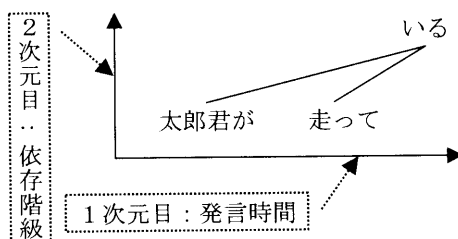


図6：文(1)の樹型図を2次元図として

発言時間を表す1次元目は発言の語を順序により並び示し、依存階級の2次元目は依存する語をそれを支配する語の下に並び示す。

尚、自然言語には、2次元目という次元はなく、それは言語学者が作ったことである。さて、なぜ樹型図を2次元に限るべきであろうか？ 人間は3次元目までも分かるので、これを利用すればよいのではないかと思ひ、これからそれはどうやって利用できるかを示してみたい。

まず、今までの2次元図を「層」と名づける。この層には文を作るための規則を初めて実行することによる構造を記入してみよう。これを「過程第1層」と名づける。二度目の再実行による構造を「過程第2層」に記入する。過程第1層と過程第2層はそれぞれ2次元目であり、これを3次元目につないでみよう。3次元目を「過程時間」と名づけ、最初の過程は次の過程より速く実行しなければならないので、2次元図の語の順序と同様な特徴なのである。抽象的な図として下のようになる：

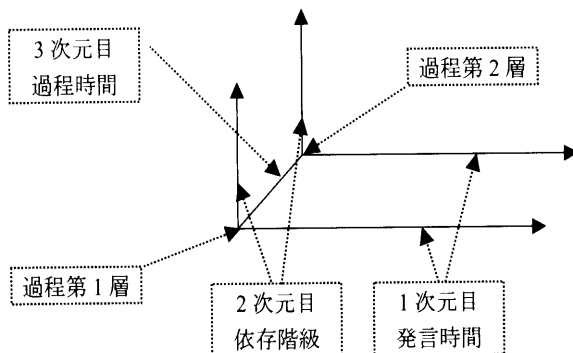


図7：3次元の樹型図

こういうタイプの樹型図の読み方は、まず過程第1層を左から右へ読み、そして、過程第2層に飛び、それをまた左から右へ読むのである。

3次元樹型図を生産する前には、重要な問題を解決しなければならない。並列接続詞は一体どの依存関係を起すだろうか？ 上では、並列接続詞につき、二つのことを主張した：各並列接続詞には論理的な意味があることを述べ、その機能は再実行過程開始を指すものであると提案した。並列接続詞の論理的な意味は並列接続詞が出る発言の樹立に最も重要な影響と及ぼし、過程開始を指す機能は発言の産出に影響を持つ。しかし、依存関係にはどのような影響があるのであろうか？ 英語でもドイツ語でも、並列接続詞は語なので、依存関係に参加すべきものと扱わなければならないと思われている。上記の(4)と(5)の確認方法のような論理的な樹立の場合には、疑問の順序は重要ではないが、統語論的な依存関係にとっては、そうでもない。これよりも、並列接続詞を二つ以上の過程層をつなぐ道具を考え、それも依存関係ぐらいの段級的な関係を見れば適当ではないかと思っている。その考え方によると、過程第 $n+1$ 層の成分を並列接続詞に、その並列接続詞を過程第 n 層の成分に依存することと見る。しかし、上に述べたように、サイクルである成分の並列とサイクルでない成分の並列を区別すべきである。サイクルの並列の場合には、過程第 $n+1$ 層の成分を並列接続詞に、その並列接続詞を過程第 n 層の支配者に依存することと見る。しかし、サイクルでない成分の並列の場合には、過程第 $n+1$ 層の成分を並列接続詞に、その並列接続詞を過程第 n 層の同じ規則で産出される成分に依存することと見る。しかし、並列接続詞がないこともあるので、そのときに成分を直接依存する語とつながらなければならない。

まず、Gappingの並列構造としては、文(16)にしてみよう。その樹型図は次のようになる：

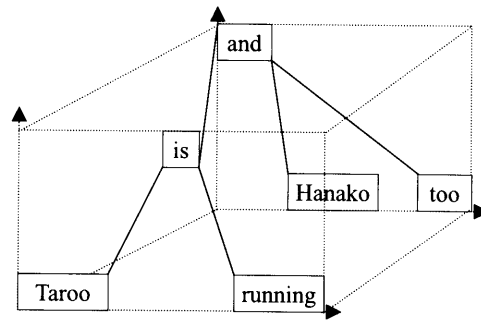


図8：文(16)の3次元樹型図

文(16)の場合には、「and」はサイクルをつなぐので、再実行過程層の成分が「and」に、そして「and」が動詞「is」に依存するように示すことにした。

図8を読むと、まず「Taroo is running」の過程第1層を読んでから、過程第2層「and Hanako, too」を読むことができる。

図8は文の並列構造を示すので、過程第1層は第2層が始まる前に完全に産出される。こ

れと違う場合もある。文(10)には、過程第1層「Taroo is running」が終わる前に過程第2層「and Hanako」を入れなければならないのである。再実行の過程はそれが始まる前までの前者の過程の成分を作る規則をコピーし、文(10)の場合には、一つだけの語なので、規則をコピーしてから過程第1層に戻る。このような構造を「非連続過程層」と名づける。Left-deletionという並列関係は常に非連続過程層になってしまう。さて、文(10)の3次元樹型図は下のようになる：

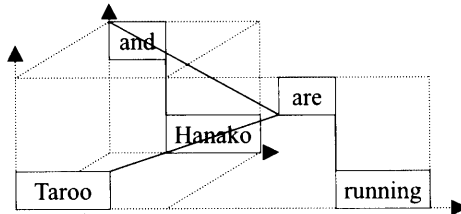


図9：文(10)の3次元樹型図

文(10)の場合には、「and」は二つの名詞——いわゆるサイクル——をつなぐので、過程第2層の名詞が「and」に、そして「and」が「is」に依存しているように示した。

結局、述べたあいまいな文の樹型図を上げてみよう。文(32)は決定疑問文による意味で次のようになる：

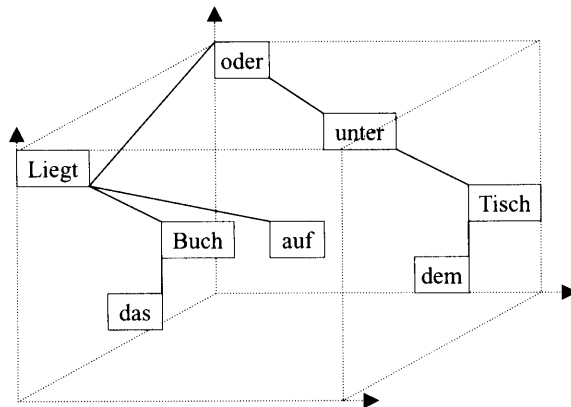


図10：文(32)を決定疑問文としての3次元樹型図

文(32)の決定疑問文の解釈では、サイクル並列なので、再実行過程の成分を「oder」に、「oder」を動詞「liegt」に依存するように示した。

文(32)の二者択一疑問文の解釈では、サイクルでない成分の並列なので、再実行過程の成分を並列接続詞に、これを前置詞「auf」に依存するようになる（図11）：

しかし、並列関係のあいまいな文では、ときどき3次元樹型図でも区別できない樹型図もある。文(30)はそういう例である（図12）。

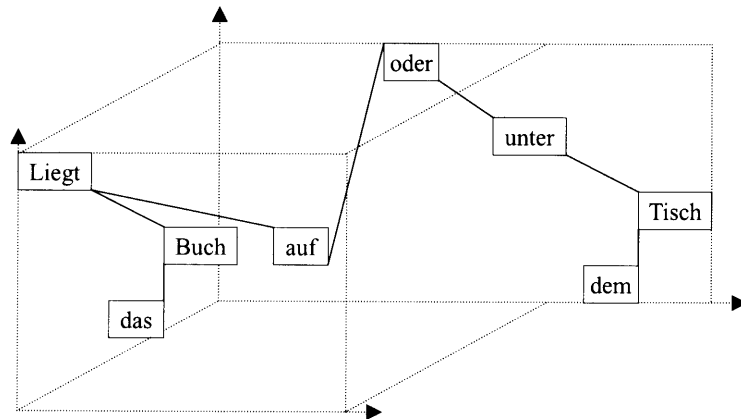


図11：文(32)を二者択一疑問文としての3次元樹型図

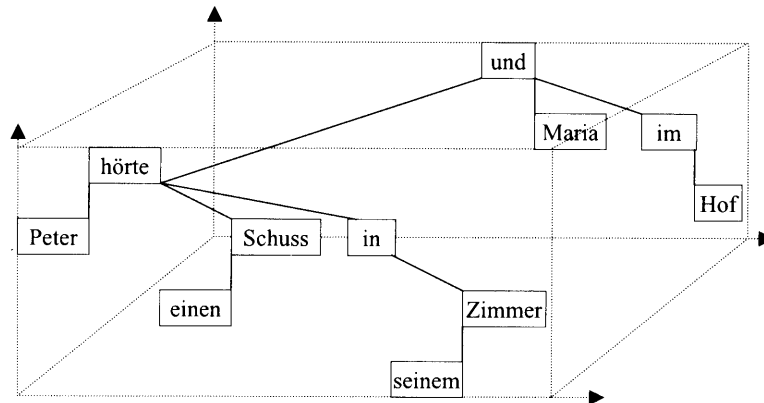


図12：文(30)の3次元樹型図

しかし、これはこの文章で提案した並列関係論にも3次元樹型図論にも全然関係ないのである。そのためには、依存文法論の樹型図は依存関係を詳しく形式化せず、その依存関係だけを枝として示すものなのである。

並列関係というのは、自然言語の日常利用に非常によく出るものなのに、文法説上には非常に記述しにくいものである。上には、その並列関係はまず、どういうふうに記述できるか、これをどういうふうに樹型図で代表できるかを述べた。

文献

- Gross, Thomas (1998) : 「„Weise am Weisen ist die Haltung“ —その依存文法的な分析」『文明21』創刊号：愛知大学国際コミュニケーション学会：1998年11月
 Gross, Thomas (1999) : 「依存文法と語順」『文明21』2号：愛知大学国際コミュニケーション学会：

1999年3月

Lobin, Henning (1993) : Koordinationssyntax als prodezurales Phanomen. Studien zur deutschen Grammatik
46. Gunter Narr Verlag, Tübingen.