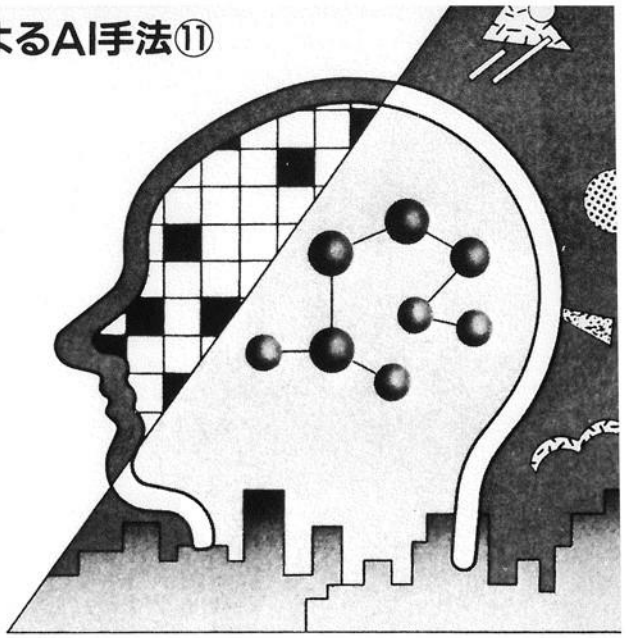

Prologによる機械翻訳システム

その3

高野　真

機械翻訳システムは，英文の形態素解析，構文解析と中間表現の生成，中間表現からの日本語文の生成といった3段階から構成される。今回は，前回ご紹介した日本語の述語部分の生成を受け継いで，埋め込み文の処理を考慮した日本語の文全体の生成について具体的なインプリメントに取りかかる。

機械翻訳システムをご紹介するにあたっては，8回の連載を予定している。2～3回は日本語の文生成，4回は英文の形態素解析，5～7回は英文の構文解析と中間表現の生成について考え，最後に全体のまとめを行なう予定である。

日本語の文生成については，機械翻訳システムの一部としてだけでなく，任意のシステムへ組み込む際に生かせるような汎用的なシステムをご紹介したい。機械翻訳には興味はないが，自然言語インターフェイスの構築などに日本語の文生成を組み込みたいという方にも参考になるであろう。次回以降の連載についても，単独でも他のシステムに組み込むことのできる一般性の高いものをとり上げていく予定である。

さて，文章を生成するときに考慮しなければならない要素としては，以下の2つのものが挙げられよう。

①主語，述語，修飾語といった文を構成する要素

②時制，話者の判断といった文全体の属性

文を構成する要素には，名詞や形容詞，動詞などの具体的な語が入ってくる。これらの語は，文章を生成したとき，実際に文面に出てくる。

文全体の属性としては，以下のようなものが挙げられる。

①時制(過去・現在・未来)

②アスペクト(継続・完了・結果)

③受身，使役

④話者の判断(否定，推定，希望)

⑤話者の働きかけ(命令，疑問，禁止)

日本語の場合，これらの属性は属性値に応じた付属語(助詞，助動詞)で表現される。時制とアスペクトに限れば，英語でも動詞の語尾変化と付属語(助動詞)を組み合わせて処理をする。しかし，受身や使役については，英語では語順で処理するのに対し，日本語では助動詞や助詞といった付属語で対処し，語順を変えることはない。

このように，日本語の文生成の手続きのなかで，文全体にかかる属性の処理は，動詞を中心とした述語の部分にかかっている。述語の生成さえできれば，あとは名詞句や副詞，埋め込み文といった文を構成する要素を，文法にかなった順に並べるだけで，意味の通じる文を生成することが可能である。

意味ネットワークによる中間表現

図1，図2は，今回の日本語の文生成に用いる中間表現を意味ネットワークで表現したものである。

意味ネットワークで枠で囲ってある部分は「ノード」，ノードを結んでいる線は「リンク」と呼ばれている。意味ネットワークは，

「左側のノード」の「リンクで示される属性」の値は「右側のノード」

「着る」の「object」の値は「セーター」

「男」の「adj 」の値は「背の低い」

「人間」の「isa 」の値は「生物」

のように読むことができる。各リンクがどのような属性を示しているかの例を以下に示そう。

A isa B

AはBの1種である

A agent B

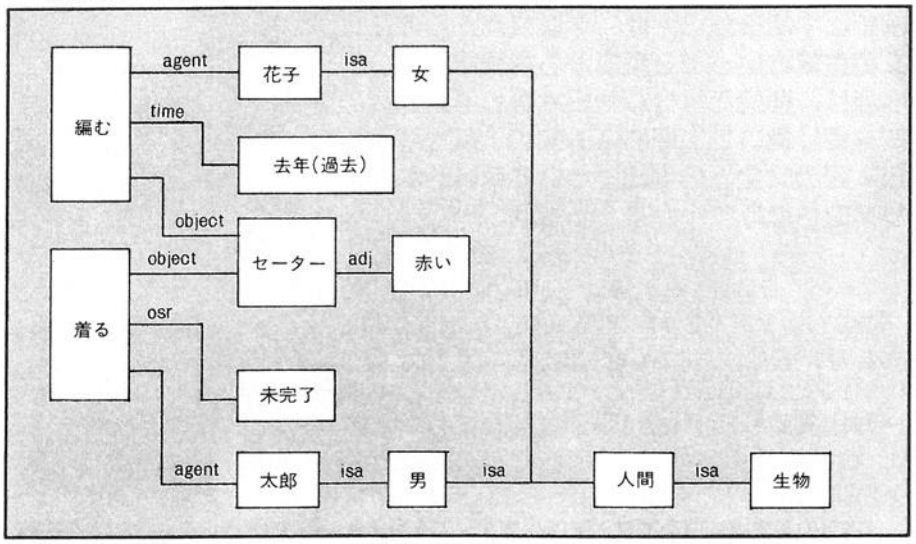

図1　花子は太郎が来ているセーターを去年編んだ

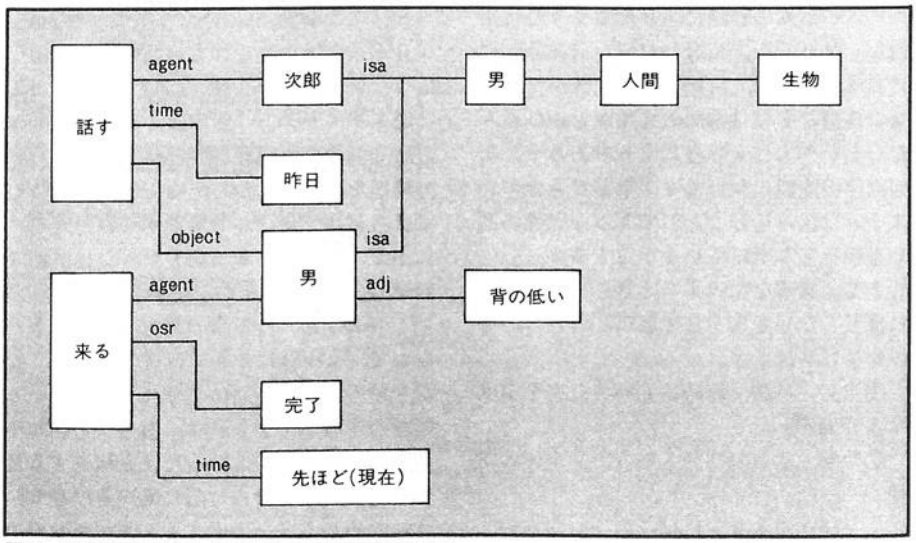

図2　次郎が昨日話した背の低い男が先ほどきた

Aの主格はBである

A object B

Aの目的格はBである

A adj B

Aを修飾する形容詞はBである

A osr B

Aのアスペクトは B である

格に関する属性は，動詞辞書と対応していさえすれば，任意のものをもって来ることができる。図1，図2の例に出ていない主なリンクには，

adv　述語を修飾する副詞

owner　名詞の所有者を示す

などがある。意味ネットワークの先頭(図の左側)に来ているのは，その文章の動詞である。動詞は，要素を4つもった動詞辞書を参照するために用いられる。第1要素は，

[[・・・|T]，T]，

であり，動詞の語幹部分を重リストで表わしたものがくる。・・・の部分は動詞語幹をシフトJISコードに変換したものが入る。たとえば「話す」の場合は，「話」のシフトJISコードである"-26526"を入れて，

[[-26526|X]，X]

を語幹とする。第2要素は，

[行，段]

という書式で動詞の活用形が入る。たとえば「話す」はさ行5段活用なので，

[さ，5]

を活用形とする。第3要素は，動詞の必要とする格に関する情報を入れる。たとえば「話す」の取り得る格は，以下のような制約をもっている。

話す主体(agent)：人間

話す相手(object)：生物

話す話題(topic)：どんな要素が入ってきてもかまわない

この制約を辞書のなかでは，

[[agent，が，人間，_]，object，に，生物，_]，[topic，について，_，_]]

```
話す([
        [[-26526|X],X],
        [さ,5],
        [[agent,が,人間,_],[object,に,生物,_],[topic,について,_,_]],
        [-,+]
        ]).
来る([
        [X,X],
        [か,変],
        [[agent,が,生物,_]],
        [-,-]
        ]).
編む([
        [[-271821X],X],
        [ま,5],
        [
                [agent,が,人間,_],
                [obj,を,衣服,_],
                [instr,によって,具体物,_],
                [material,で,具体物,_]
        ],
        [-,+]
        ]).
着る([
        [X,X],
        [か,上一],
        [
                [agent,が,人間,_],
                [obj,を,衣服,_]
        ],
        [-,+]
        ]).
```

図3　意味ネットワーク中に出てくる動詞の辞書

```
link(agent,[述語,編む],花子).
link(isa,花子,女).
link(isa,女,人間).
link(adj,セーター,[adj,赤い]).
link(osr,[述語,着る],未完了).
link(agent,[述語,着る],太郎).
link(obj,[述語,着る],セーター).
link(isa,太郎,男).
link(isa,セーター,衣服).
link(isa,男,人間).
link(isa,人間,生物).
```

図4　図1のProlog表現

```
link(time,[述語,話す],[過去,昨日]).
link(topic,[述語,話す],男).
link(agent,[述語,話す],次郎).
link(isa,次郎,男).
link(adj,男,[tall,背の低い]).
link(time,[述語,来る],[現在,先ほど]).
link(osr,[述語,来る],完了).
link(agent,[述語,来る],男).
link(isa,男,男).
```

図5　図2のProlog表現

と表現する。第4要素はアスペクト素性を表わす。アスペクト素性は現在は以下に挙げる4通りのものがある。

[+, +]　「～ている」がつかない動詞

[+, -]　「前，間，後，ながら」がつかない動詞

[-, -]　「～し終える」がつかない動詞

[-, +]　その他大半の動詞はここに入る

たとえば「話す」は，「～ている」「前」「～し終える」のいずれもつけることができるので，

[-, +]

となる。また，来るは「来終える」という表現を取れないので，

[-, -]

となる。図1，図2に出て来る4つの動詞の辞書を図3に示す。

図1，図2の意味ネットワークをPrologで記述したものを，それぞれ図4，図5に示す。

述語ノードはそれが述語であることを示すために，

['述語', Node]

と表記する。副詞を表わすノードには，

['adv', Node] Nodeには副詞を表わすアトム

['time', Node] Nodeには副詞を表わすアトム

['locate', Node] Nodeには副詞を表わすアトム

の3種類のものがある。形容詞を表わすノードは，

['adj', Node] Nodeには形容詞を表わすアトム

という書式で書くことになっている。

次に，時制などの文に関する属性についての書き方を説明しよう。「れる，られる」「せる，させる」で表現される受身，使役，可能といった属性は，「命題」というリンク名で以下のように表現する。

link('命題', ['述語', Node], X).

X には「受身」「使役」「可能」のいずれかが入る

テンス，アスペクトについては，

link('osr', [' 述語', Node], X).

X には「完了」「未完了」のいずれかが入る

link('time', ['述語', Node], X).

X には「['過去', Adv]」「['現在', Adv]」「['未来', Adv]」のいずれかが入る。Adv には時を表わす副詞が入る。

と記述する。話者の判断については――

link('話者の判断', ['述語', Node], X).

X は[Mode, Time]という書式でMode には，

打ち消し・・・・(ない)

希望・・・・・・(たい)

推定・・・・・・(らしい)

様態・・・・・・(そうだ)

伝聞・・・・・・(そうだ)

不確定・・・・・(ようだ)

丁寧・・・・・・(ます)

意志・・・・・・(う・よう)

打ち消し意志・・(まい)

のいずれかが入り，Timeには過去，現在，未来のいずれかが入る。

――と記述する。話者の働きかけについては，次のように記述する。

link('話者の働きかけ', ['述語', Node], X).

Xには連言、選言、例示、程度、帰着点、類推、限定、仮定、原因、順接、逆接、並列、命令、疑問、禁止、確認、平叙のいずれかの記号が入る。

こうした文の属性は別に指定しなくてもかまわない。指定がない場合は，デフォルト値として以下のような値が取られる。

命題 ： [] (能動文)
time ： [] (非過去)
osr ： [] (中立)
話者の判断：[[ ],[ ]](中立,非過去)
話者の働きかけ：[](平叙)

日本語文生成システムのインプリメント

述語の生成については，前回インプリメントしたプログラムを，そのまま用いることにする。今回は以下の2点に焦点を当てて，日本語文生成を考察していきたい。

①動詞，名詞，形容詞，副詞といった文を構成する要素をどういった語順で表現すか。

②埋め込み文を処理するにはどうしたらよいか。

図6が，意味ネットワークから日本語を生成するためのプログラムである。このプログラムと図3の動詞辞書，図4，図5の意味ネットワークおよび11月号の述語生成の辞書(リスト1)をコンサルトして，

?-ex(_ 述語ノード).

と入力する。述語ノードとしては「着る」「編む」「話す」「来る」を入れることができる。たとえば，

?-ex(着る).

と入力すると，

太郎が花子が去年編んだ赤いセーターをきている

という文が返り，

?-ex(編む).

と入力すると，

花子が太郎がきている赤いセーターを去年編んだ

という文が返る。

文を構成する動詞・名詞・形容詞・副詞といったさまざまな要素は，前節で見たように意味ネットワークによって表現している。日本語生成プログラムはこの意味ネットワークを参照しつつ，日本語の文法に従って各要素をつなげていく。プログラムの先頭は，

日本語文(S1, St, Node, SubNode)

となっているが，それぞれの引き数は以下のような役割をもっている。

S1：生成された日本語文が入る
St：空リスト [] が入る
Node：生成を始める述語ノードを指定する
SubNode ：最初は空リスト [] が入る。埋め込み文の処理を行なうときは，ここに主文の述語ノードが埋め込み文へ渡される

上の述語のなかでは，

手続き _ 日本語文(Node, SubNode, _ 辞書),
述部(S3, St, Node, _ 辞書),
副詞 _ 時(S2, S3, Node, []),
主部(S1, S2, Node, _ 辞書).

といった一連の処理が行なわれる。手続き _ 日本語文は，動詞辞書の第3引き数にある格の制約に関する情報と実際の意味ネットワークのマッチングをとり，意味ネットワークの内容が格に関する制約を満たしていれば，実際の内容を束縛する。このマッチングは以下のような手順で行なわれる。

①格の制約に関する情報を動詞辞書から取り出す。たとえば「話す」の「agent」は「人間」である。

②意味ネットワーク中の「話す」の「agent」を参照すると「次郎」になっている。

③「次郎」が「人間」であれば，「話す」の「agent」を次郎として束縛する。「次郎」が「人間」であるかどうかは意味ネットワーク中の「isa」リンクをたどっていけばわかる。

その後は文の語尾の方向から，述部，副詞，主部の順に文章を構成していく。述部の構成は意味ネットワークから，テンス，アスペクトなどの文の属性に関する情報を取ってきて述語生成ルーチンに渡すことで行なう。述語生成については，11月号でご紹介したルーチンをそのまま用いている。

文の属性に関する情報は，

手続き_述部(Node, _命題, Osr, T1, Mode, T2, Func),

というProlog述語で行なう。主部を構成する名詞句の生成は，以下の手順で行なわれる。

①その名詞句が他の動詞との共通の格になっている場合は，埋め込み文の処理を行なう。

②owner リンクがあれば，名詞句の所有者としてその内容を加える。

③adj リンクがあれば，名詞句の修飾語としてその内容を加える。形容詞にかかる副詞の処理は，形容詞の処理のときに行なわれる。

④格に応じた格助詞をつけて名詞句を構成し，つけ加える。

このうち埋め込み文の処理は，最初の日本語文/4を再帰的に呼び出すことにより処理を行なう。このときは，2つの述語によって共有されている名詞句が，生成された文章に重複しないように埋め込み文のほうからは取り除く処理を行なう。そうしないと，

花子が赤いセーターを太郎がきている赤いセーターを去年編んだ

といったおかしな文章が生成されてしまう。埋め込み文の処理のときは，すでに言及された動詞と，埋め込み文の動詞の区別をつけるために，marked/1という述語で処理の終わった動詞にマークをつけるようにしている。

☆　☆　☆

述語生成システムさえしっかりしたものになっていれば，文生成システムのインプリメントはさほどむずかしくはない。今回のシステムでは，5w1h形式の疑問文や従位接続詞を含んだ複文には対応できないが，拡張は2～3述語を追加するだけで可能であろう。興味のある読者の方は試みられると理解を深めることができるであろう。

また，この意味ネットワークからの文生成システムは機械翻訳システムの一部としてだけでなく，コンサルテーションシステムやCAI(コンピューター補助教育)システムの一部としても十分実用になる。そのほかにも意味ネットワークという知識表現は，推論システムの記述も十分可能であるためにいろいろと応用がきく。さらに，文を構成する要素や，文の属性の記述をフレームで記述しなおすことも十分可能である。

次回は，英文の形態素解析および辞書引きを行なうシステムをご紹介する。

```
%************************%
%                        %
%        平叙文          %
%                        %
%************************%
日本語文(S1,St,Node,SubNode) :-
        手続き_日本語文(Node,SubNode,_辞書),
        述部(S3,St,Node,_辞書),
        副詞_時(S2,S3,Node,[]),
        主部(S1,S2,Node,_辞書).
%*************************************%
%       手続き_日本語文(Node,_辞書)     %
%*************************************%
手続き_日本語文(Node,SubNode,[X,Y,Z,W]) :-
        M =.. [Node,[X,Y,Z1,W]],
        M,
        格束縛(Node,Z1,Z2),
        remove(SubNode,Z2,Z).
格束縛(_,[],[]).
格束縛(Node,[[_格,X,_格条件,_格内容]|T],[[_格,X,_格条件,_格内容]|Ret]) :-
        link(_格,[述語,Node],_格内容),
        !,
        generic(_格内容,_格条件),
        格束縛(Node,T,Ret).
格束縛(Node,[_|T],Ret) :-
        格束縛(Node,T,Ret).
generic(X,X).
generic(_格内容,_格条件) :-
        link(isa,_格内容,X).
```

図6　意味ネットワークから日本語を生成するためのプログラム

```
        generic(X,_格条件).
mark(X) :-
        marked(X).
        !,fail.
mark(X) :-
            asserta(marked(X)),
        !.
remove(_,[],[]).
remove(SubNode,[[_,_,_,SubNode]|T],T).
remove(SubNode,[X|T1],[X|T2]) :-
        remove(SubNode,T1,T2).
%******************************%
%       副詞(S1,S2,Node,[])     %
%******************************%
副詞_時([Time|S1],S1,Node,[]) :-
        link(time,[述語,Node],[_,Time]),
        副詞_場所(S1,St,Node,[]).
副詞_時(S1,St,Node,[]) :-
        副詞_場所(S1,St,Node,[]).
副詞_場所([Locate|S1],S1,Node,[]) :-
        link(locate,[述語,Node],[_,Locate]),
        副詞(S1,St,Node,[]).
副詞_場所(S1,St,Node,[]) :-
        副詞(S1,St,Node,[]).
副詞([Adv|St],St,Node,[]) :-
        link(adv,[述語,Node],[_,Adv]).
副詞(St,St,Node,[]).
%***********************%
%                       %
%       述部            %
%                       %
%***********************%
述部(S1,St,Node,_辞書) :-
        手続き_述部(Node,_命題,Osr,T1,Mode,T2,Func),
        命題生成(S1,St,_辞書,_命題,Osr,T1,Mode,T2,Func).
%*********************************************************%
%       手続き_述部(Node,_命題,Osr,T1,Mode,T2,Func)     %
%*********************************************************%
手続き_述部(Node,_命題,Osr,T1,Mode,T2,Func) :-
        (link(命題,[述語,Node],_命題);_命題 = []),
        (link(osr,[述語,Node],Osr);Osr = []),
        (link(time,[述語,Node],[T1,_]);T1 = []).
        (link(話者の判断,[述語,Node],[Mode,T2]);(Mode = [],T2 = [])),
        (link(話者の働きかけ,[述語,Node],Func);Func = []).
%***********************%
%                       %
%       主部            %
%                       %
%***********************%
主部(S1,St,Node,[_,_,_格構造,_]) :-
        主部1(S1,St,Node,_格構造).
主部1(St,St,Node,[]).
主部1(S1,St,Node,[[_格,_格助詞,_,_格内容]|T]) :-
        名詞句1(S1,S2,_格内容,_格助詞),
        主部1(S2,St,Node,T).
%***********************%
%                       %
%       名詞句          %
%                       %
%***********************%
%***********************%
%       名詞句1         %
%***********************%
名詞句1(S1,St,Node,_格助詞) :-
        link(_,[述語,Next_Node],Node),
        mark(Next_Node),
        日本語文(S1,S2,Next_Node,Node),
        名詞句2(S2,St,Node,_格助詞).
名詞句1(S1,St,Node,_格助詞) :-
        名詞句2(S1,St,Node,_格助詞).
%***********************%
%       名詞句2         %
%***********************%
名詞句2(S1,St,Node,_格助詞) :-
        link(owner,Node,Owner),
        名詞句(S1,S2,Owner,[]),
        名詞句3(S2,[の|St],Node,_格助詞).
```

(次ページに続く)

```
名詞句2(S1,St,Node,_格助詞) :-
        名詞句3(S1,St,Node,_格助詞).
%*********************%
%       名詞句3       %
%*********************%
名詞句3(S1,St,Node,_格助詞) :-
        link(adj,Node,[_,Adj]),
        形容詞(S1,S2,Adj,[]),
        名詞句4(S2,St,Node,_格助詞).
名詞句3(S1,St,Node,_格助詞) :-
        名詞句4(S1,St,Node,_格助詞).
%***************%
%       名詞    %
%***************%
名詞句4([Node,_格助詞|St],St,Node,_格助詞).
%***************%
%               %
%       形容詞  %
%               %
%***************%
形容詞([Adv|S1],S1,Node,[]) :-
        link(adv,Node,[_,Adv]),
        形容詞1(S1,St,Node,[]).
形容詞(S1,St,Node,[]) :-
        形容詞1(S1,St,Node,[]).
形容詞1([Node|St],St,Node,[]).
ex(Node) :-
        abolish(marked,1),
        link(_,[述語,Node],_),
        mark(Node),
        日本語文(X,[],Node,[]),
        !,
        jwrite(X),nl.
%*********************%
%                     %
%       生成時        %
%       動詞命題の処理 %
%                     %
%*********************%
述語生成([[S0,S1],[_行,_段],_,Asp],_命題,Osr,T1,Mode,T2,Func) :-
        命題生成(S0,[],[[S0,S1],[_行,_段],_,Asp],_命題,Osr,T1,Mode,T2,Func),
        knjprint(S0).
命題生成(S0,St,[[S0,S1],[_行,_段],_,Asp],_命題,Osr,T1,Mode,T2,Func) :-
        M6 =.. [話者の働きかけ,S6,St,Func,Kt5],
        M6,
        M5 =.. [判断時制,S5,S6,T2,Kt4,Kt5],
        M5,
        M4 =.. [話者の判断,S4,S5,Mode,Kt3,Kt4],
        M4,
        M3 =.. [アスペクト,S3,S4,Asp,Osr,T1,Kt2,Kt3],
        M3,
        M2 =.. [命題変換,S2,S3,_命題,[_行,_段,Kt1],Kt2],
        M2,
        活用形(_行,_段,_活用),
        M1 =.. [_活用,S1,S2,Kt1],
        M1.
活用形(_行,_段,_活用) :-
        name(_行,N1),
        name(_段,N2),
        append(N1,N2,N3),
        name(_活用,N3),
        !.
append([A|X],Y,[A|Z]) :-
        append(X,Y,Z).
append([],X,X).
jwrite([]) :-
        !.
jwrite([X|Y]) :-
        integer(X),
        knjput(X),
        jwrite(Y).
jwrite([X|Y]) :-
        write(X),
        jwrite(Y).
knjput(X) :-
        A is X / 256 + 255,
        B is X mod 256 + 256,
        put(A),put(B).
```

図6 (続き)