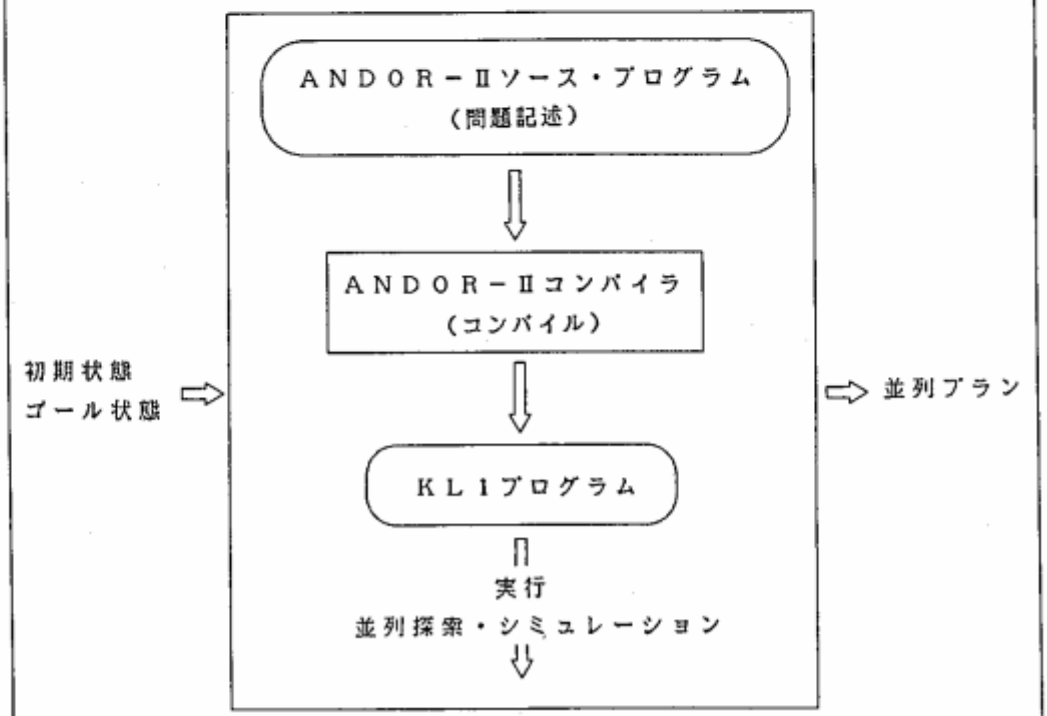
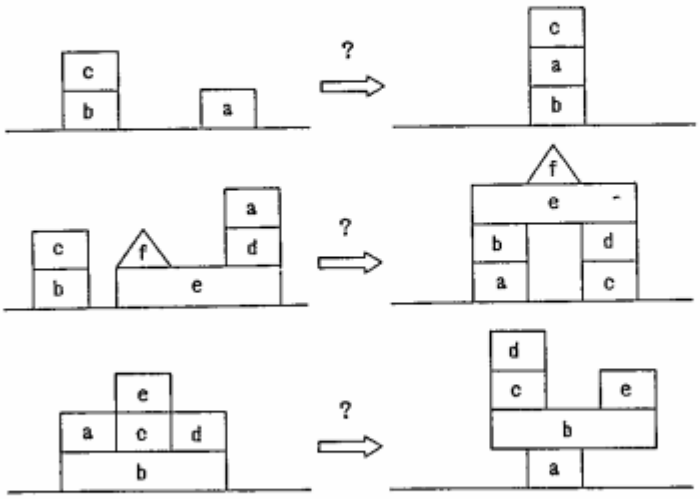
題　名	並列動作系推論実験システム
目　的	並列事象の簡明な記述言語ＡＮＤＯＲ－Ⅱとその表現に基づいた高度な推論メカニズムを核言語第１版（ＫＬ１）でインプリメントすることにより、効率のよい並列推論システムの実現を目指しています。
概要及び特徴	①　ＡＮＤＯＲ－ⅡはＡＮＤ、ＯＲ両並列性を備えた言語です。 ②　論理積の関係にあるゴールは並列に実行されます。（ＡＮＤ並列） ③　複数の動作の可能性がある時は、各々の推論は分岐した世界で独立に実行されます。（ＯＲ並列） ④　コンパイラはＫＬ１で記述され、ＡＮＤＯＲ－Ⅱの記述はＫＬ１のプログラムに変換されます。 ⑤　並列故障診断や並列プラン生成などに応用できます。 並列プラン生成への応用例（ブロックの積み替え） ・　各ブロックは能動的なオブジェクトで複数のブロックが並列に動作することが可能な世界を想定します。 ・　入力として初期状態とゴール状態を与えると、自分のゴールを達成するプランを生成することによって出力として並列プランが生成されます。
構　成	

デ　　モ　　内　　容　（1／3）

問題　並列プラニング　（ブロック積み替え問題）

初期状態　　ゴール状態



ブロック及びそのアクションに関する仮定

① 各ブロックは自律的なオブジェクト。
② 各ブロックは自分の上下の状況しか知らない。
③ 全体の状況を把握したオブジェクトは存在しない。
④ 上に何もないブロックだけが移動可能
⑤ ブロックの移動には０以上の有限の時間（一定ではない）がかかる。

並列性

ＡＮＤ：　ブロックの並列動作

ＯＲ：　(a) ブロックが自分のゴールを達成するためのアクションと、他のブロックからの要求を満たすためのアクションのどちらを選択するか。
　　(b) あるブロックが複数のブロックから要求を受けるときどういう順序で処理するか。

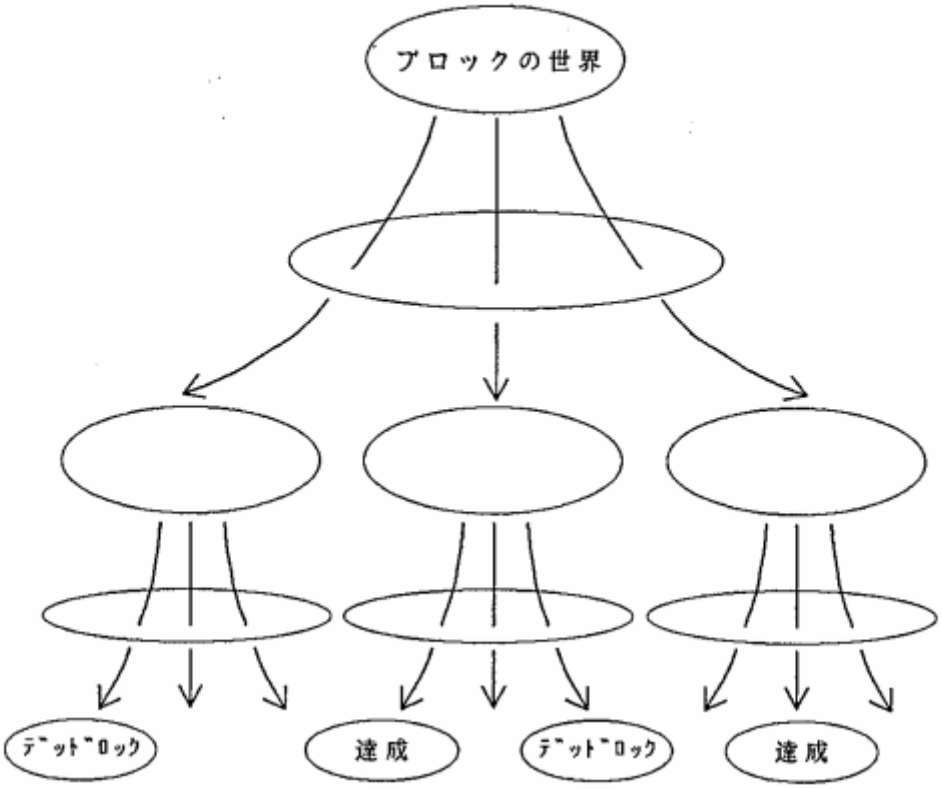
本システムでは、ブロックの世界をＡＮＤとＯＲの両並列性を備えた言語ＡＮＤＯＲ－Ⅱで記述し、その記述に基づきＯＲのもたらす可能世界の各々について推論を進め、ゴールを達成する可能世界を見つけ出し、そこからプランを抽出します。

ＡＮＤＯＲ－Ⅱによる記述の例

```
pf1(Plan) :- true |
   block(a, AB,AC, BA,CA, c(clear,table),f(c,b),     [start],Pa),
   block(b, BC,BA, CB,AB, c(c,table),    f(a,table), [start],Pb),
   block(c, CA,CB, AC,BC, c(clear,b),    f(clear,a), [start],Pc),
   Plan=[Pa,Pb,Pc].

action(Id,S1,S2,In,Current,Final,H,P) :-
   act_action(Id,S1,S2,In,Current,Final,H,P).
action(Id,S1,S2,In,Current,Final,H,P) :-
   pas_action(Id,S1,S2,In,Current,Final,H,P).
```

デ　　モ　　内　　容　(2／3)



ＡＮＤＯＲ－Ⅱの記述はＡＮＤＯＲ－ⅡコンパイラによりＫＬ１に変換され、実行はＫＬ１処理系で行われます。ＡＮＤＯＲ－ⅡのＡＮＤ／ＯＲ並列性は、各可能世界のデータを固有の色で色づけすると言う手法でＫＬ１のＡＮＤ並列に変換されます。

変換例

```
pf1_Core(Plan,Cs) :- true |
   Cs={CsA,CsB,CsC},
   block(a,AB,AC,BA,CA,c(clear,table),f(c,b),    [start],Pa,CsA),
   block(b,BC,BA,CB,AB,c(c,table),    f(a,table),[start],Pb,CsB),
   block(c,CA,CB,AC,BC,c(clear,b),    f(clear,a),[start],Pc,CsC),
   unify_N4_NShell_1_1(Plan,Pa,Pb,Pc).

action_AShell(Id,S1,S2,In,Current,Final,H,P,w(C),Cs) :- true |
   Cs=[get(Cnt)|Css],
   Css={CsA,CsP},
   andor:append(C,[(d2,Cnt)],C2),
   pas_action_Check(Id,S1p,S2p,In,Current,Final,H,P2,w(C2),CsP),
   andor:append(C,[(d1,Cnt)],C1),
   act_action(Id,S1a,S2a,In,Current,Final,H,P1,w(C1),CsA),
   andor:merge_BLT(S1a,S1p,S1),
   andor:merge_BLT(S2a,S2p,S2),
   andor:merge_BLT(P1,P2,P).
```

デ　　モ　　内　　容　（3／3）

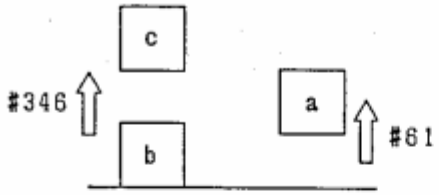
出力はゴールを達成する並列プランの列（通常無限列）です。並列プランは各ブロックのイベント付きアクション列の集合です。１つのブロックに関するアクションは全順序づけされていますが、異なるブロックのアクション間に順序制約を導入するのがイベントidです。

並列プランの例

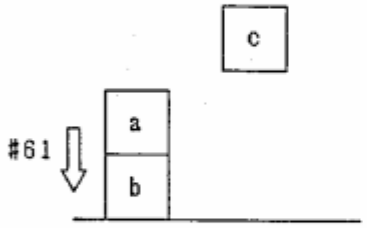
```
a: [end,(accept(c),#346),(clear,#61),(move_to(b),#61),start]
b: [end,(accept(a),#61),(cleared,#346),start]
c: [end,(clear,#346),(move_to(a),#346),start]
```

この並列プランをイベントidによる順序制約を考慮して図で表現すると、以下のようになります。

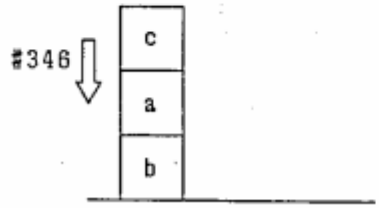
初期状態



・aがbの上へ行こうとする
・cがaの上へ行こうとする



bがaを受け取る



aがcを受け取る

ゴール状態