

並列プログラムの プロトタイピング支援環境 MENDELS

ICOT第1研究室

(株)東 芝

本 位 田 真 一

目的：並列プログラムを対象としたプロトタイピング支援環境の構築

- ソフトウェア生産性の向上
- ソフトウェアの高品質化

| 手法 | 目的 ソフトウェア 生産性の向上 | ソフトウェア の高品質化 |
|--------|------------------------|-----------------|
| ①定理証明 | × | ○ |
| ②部品再利用 | ○ | △ |

①+②：部品の再利用をベースとして、部品間のインタフェース部分の生成に定理証明手法を用いる。

概 要

- 並列プログラム部品の再利用
部品検索, 結合, 同期部生成, 実行
- 時制命題論理による仕様から同期部生成
タブロー法によるモデルグラフ生成
- 意味ネットワーク表現による部品仕様
柔軟な検索, 結合
- 並列プログラムはMENDEL/87で記述
Prologベースの並列型オブジェクト指向言語
- PSI上で試作・評価

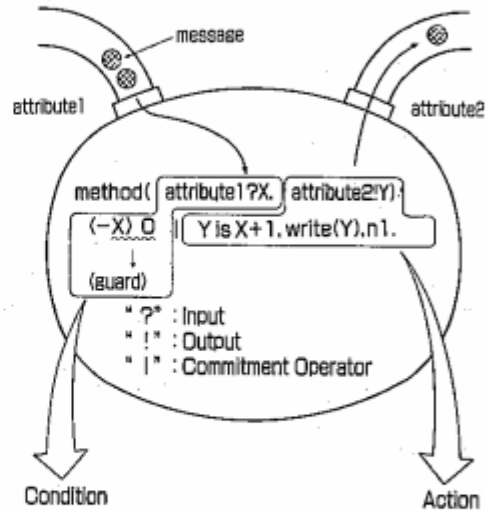
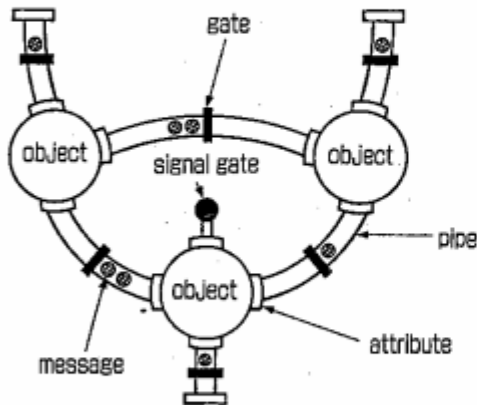
並列プログラムとMENDEL/87

| 並列プログラム | 並列プログラム 部 品 | MENDEL/87 |
|---------|------------------|--------------------------|
| 機 能 部 分 | 部 品 本 体 | オブジェクト |
| 同 期 部 分 | 部 品 間 インタフェース | メッセージ・パッシング の 順 序 付 け |

MENDEL/87

Method Part

Method=Production Rule



Concurrent Program Synthesis System V 2.2
Mendel's Zone

I/O specification

```
stream!
keyword!
summary_report?
```

demo:keycheck.mdl

```
atomic object keycheck {
  dec: {
    input (keyword,word) ;
    output (check_data) ;
    data (keylist![])
  } ;
  meth: {
    method (keyword?K,keylist?KL,keylist![]K:KL) ;
    method (word?W,keylist?KL,check_data!W) <-
      member (W,KL) ;
      write('<< check word >>,write (K),n1 ;
    method (word?W)
  } ;
  junc: {
    member (L,[]) :- !,fail ;
    member (E,[E:L]) ;
    member (E,[L:L]) :-
      !,member (E,L)
  } ;
}
```

Object library

| | | |
|----------|----------|----------|
| wordcut | keycheck | sunrep |
| sundump | calendar | schedule |
| g_member | swp_room | swp_job |
| adjust | b_node | c_node |
| d_node | e_node | cpu |
| disk | naneger | node1 |

#: Please select window
 screen editor_buffer_window/20
 USER : zone88
 SIMPOS Version 3.2-11

19-Apr-88 Tuesday 17:00:46

Concurrent Program Synthesis System V 2.2
Mendels Zone

I/O specification

```
stream!
keyword!
summary_report?
```

Object library

| | | | |
|----------|----------|----------|---|
| wordcut | keycheck | sunrep | * |
| sundump | calendar | schedule | * |
| g_member | swp_room | swp_job | * |
| adjust | b_node | c_node | * |
| d_node | e_node | cpu | * |
| disk | manager | node1 | * |

PTL Result Window

```
#####
# MODEL GRAPH #
#####
START
n(1)
|-n(1) g2 *
|-n(1) g1 *
|-n(5) eog(g1)
|   |-n(5) g2 *
|   |-n(9) eog(g2)
|       |-n(16) g4
|           |-n(16) g5 *
|           |-n(16) g4 *
|           |-n(22) g3
|               |-n(24) halt
|                   |-n(24) halt *
|-n(15) g3
|   |-n(20) halt
|       |-n(20) halt *
|-n(4) eog(g2)
|   |-n(4) g1 *
|   |-n(10) g4
|       |-n(10) g5 *
|       |-n(10) g4 *
|       |-n(10) g1 *
|       |-n(16) eog(g1) *
|-n(9) eog(g1) *
```

PTL specification

```
>eog (g1)
>eog (g2)
>halt
□(eog (g1) => □□(-eog (g1) ^
□(eog (g2) => □□(-eog (g2) ^
□(halt => □□halt))
~g4 U eog (g2)
>g3
□(g3 => >halt)
~g5 U g4
```

Command window

```
file name>kywd.att
filename>kywd.spc
```

screen

USER : zone88
SIMPOS Version 3.2-11

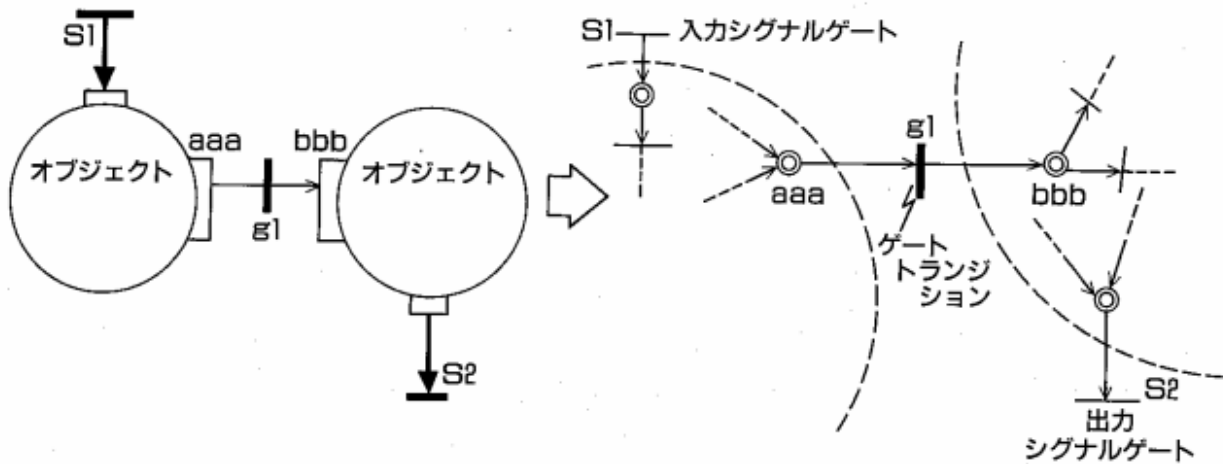
19-Apr-88 Tuesday 17:08:36

MENDEL ネット (1)

- MENDEL ネットは限定ペトリネットである。
- MENDEL プログラムの骨格は、MENDEL ネットで表わされる。
- MENDEL ネットと時制命題論理を用いて MENDEL プログラムの検証ができる。

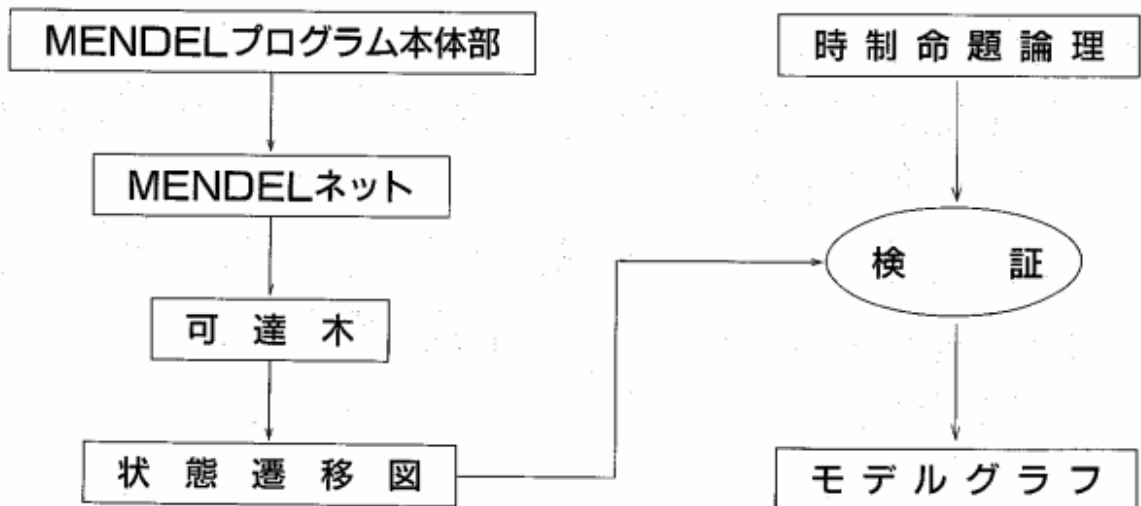
| MENDEL プログラム | MENDEL ネット |
|--------------|------------|
| 外部入出力ポート | ◎ |
| 内部変数 | ○ |
| メソッド | |
| ゲート (メッセージ) | |
| ゲート (シグナル) | |

配管図とMENDELネット



MENDELネット (2)

● プログラム検証



モデルグラフが空ならばMENDELプログラム本体部は時制命題論理で記述された同期仕様を満たしえない。

PTS(Practical Temporal Specification Language)

時制命題論理(PTL)を拡張した仕様記述言語

対象, その状態および状態を変化させる作用が明示的に記述可能

- [例] switch : 対象, {on, off} : switchのドメイン,
 ↑ : 作用オペレータ
 switch (on) \equiv switchの状態がonである.
 ↑ switch (off) \equiv switchの状態をoffにする.
 ↑ switch (ϵ) \equiv switchに対する作用は何もない.

作用がない場合, オブジェクトの状態は次の時点でも保持される.

switch(on) \wedge ↑ switch(ϵ) \supset ○ switch(on) は恒真

CPTL(Conditional Propositional Temporal Logic)

- PTLとConditional expressionを融合
- タブロー図(状態遷移図形式)のエッジにConditional expressionを付加
- Conditional expressionを評価しながら状態遷移
- Conditional expression

#P (これまでに命題記号Pが成立した回数)

ブール式

優先順位

[例]

- (req1 ((#make1 - #req1) > 0) \vee req2 ((#make2 - #req2) > 0))
- (req1 (priority(0)) \vee req2 (priority(1)))

今 後 の 予 定

- MENDEL/GHCの実装
- リアルタイムSA(Structured Analysis)との融合により要求定義からプログラミングまでの一貫支援システム
- 時制命題論理の拡張とその適用
 - 状 態
 - 動的条件
 - 構 造
- 実アプリケーションへの適用