制約論理プログラミングによる
ハイブリッドシステムのパラメータ設計

平石 邦彦 ・ 石川 礼

北陸先端科学技術大学院大学・情報科学研究科

高信頼ハイブリッドシステム構築のために

- 数値シミュレーション
- 定性シミュレーション
- モデル検査
- パラメータ設計
- 記号的シミュレーション
- 近似的シミュレーション

→ 制約論理プログラミング

制約論理プログラミング

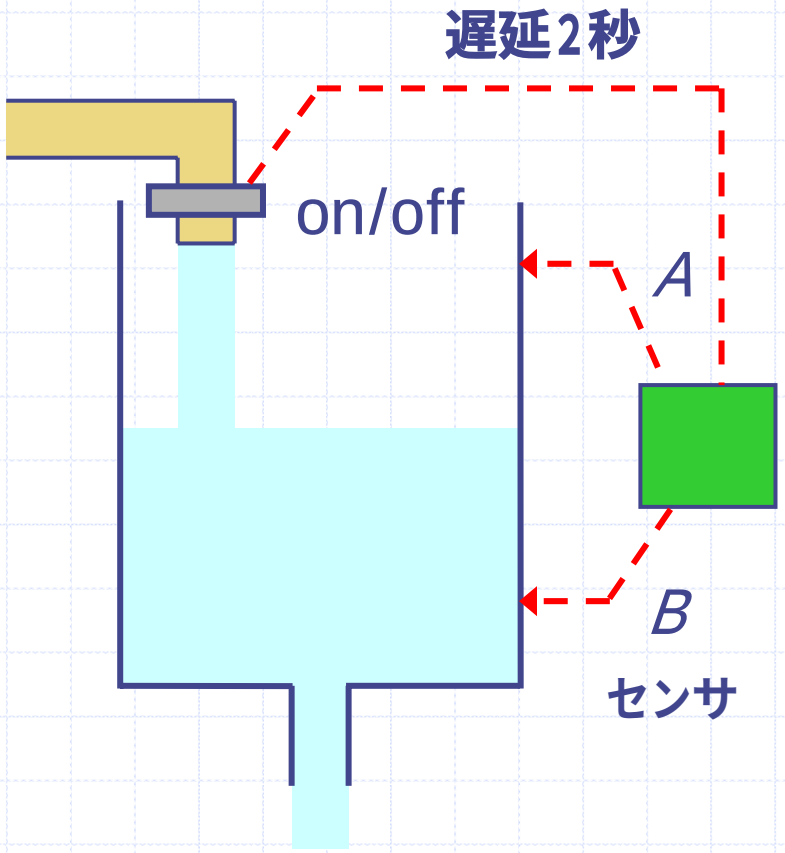
- **制約論理プログラミング**(CLP: Constraint Logic Programming)

  CLP = Prolog + Constraint Solver.
- CLP**は**ハイブリッド言語.
  - Logic Programming (**離散制約**)
  - Constraint Solver (**連続制約**)
- Keyed CLP for HS =Prolog Interpreter
  - + Linear Constraint Solver/Optimiser
  - + Quadratic Optimizer
  - + Projection(a subset of Quantifier Elimination)
  - + Global Variables

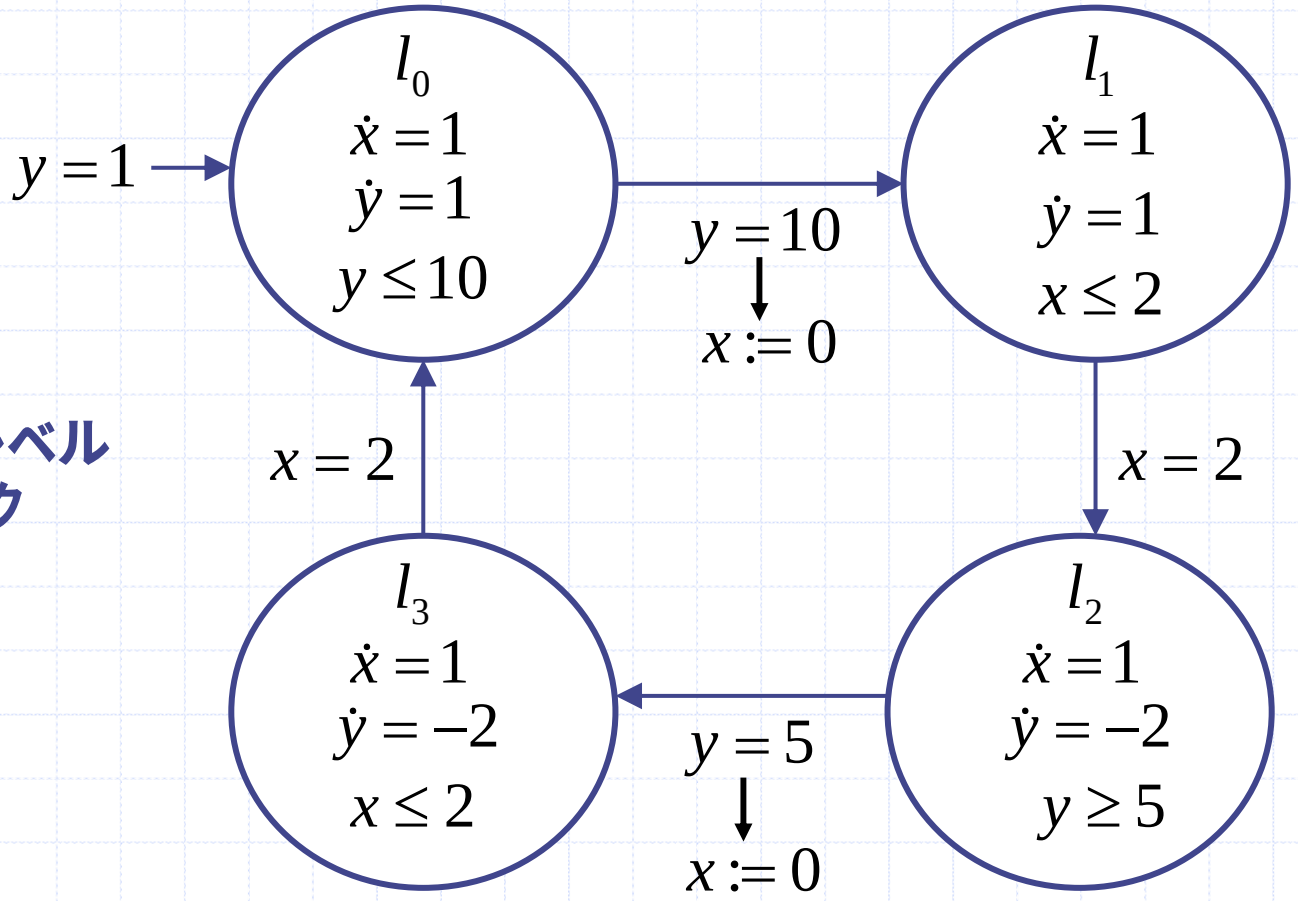
Keyed CLP for HSの目的

- ハイブリッドシステムの設計，検証に必要なプリミティブを実装．
- 問題に応じて，それを解くCLPコードを**自動生成**．

例題：水面レベルモニタ

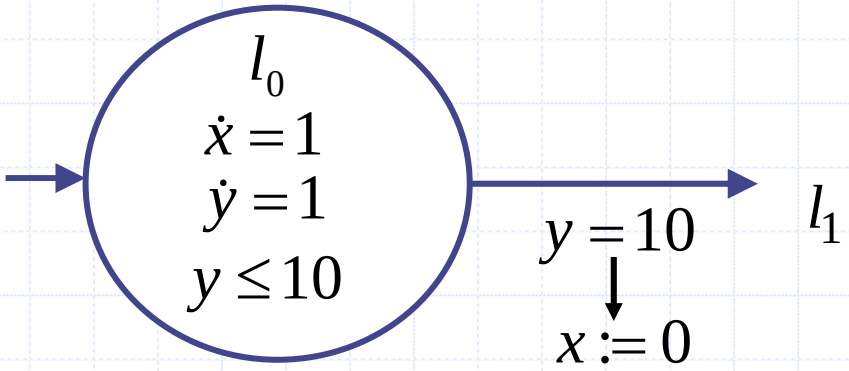


例題：水面レベルモニタ



y: **水面レベル**
x: **クロック**

単純なモデル化 (1)




```
l0(X, Y, Time):-
        X1 = X + D, Y1 = Y + D, D >= 0,
        Y1 = 10,
        l1(0, Y1, Time + D).
```

単純なモデル化 (2)

```
I0(X, Y, Time):-
        X1 = X + D, Y1 = Y + D, D >= 0,
        Y1 = 10,
        I1(0, Y1, Time + D).
I1(X, Y, Time):-
        X1 = X + D, Y1 = Y + D, D >= 0,
        X1 = 2,
        I2(X1, Y1, Time + D).
I2(X, Y, Time):-
        X1 = X + D, Y1 = Y -2 * D, D >= 0,
        Y1 = 5,
        I3(0, Y1, Time + D).
I3(X, Y, Time):-
        X1 = X + D, Y1 = Y -2 * D, D >= 0,
        X1 = 2,
        I0(X, Y, Time + D).
```

実行トレース

```
(trace)| ?- I0(X, 1, 0).
1-0) CALL : I0(X, 1, 0) ?
1-0) TRY  : I0(X, 1, 0) :- _5 = X + _6, _8 = 1 + _6, _6 >= 0, _8 = 10, I1(0, _8, 0 + _6)
1-0) SUC  : I0(X, 1, 0) :- _5 = X + _6, _8 = 1 + _6, _6 >= 0, _8 = 10, I1(0, _8, 0 + _6)
1-1) CONSTRAINT : _5 = X + _6
1-1) SUC  : _5 = X + _6
1-1) CONSTRAINT : _8 = 1 + _6
1-1) SUC  : _8 = 1 + _6
1-1) CONSTRAINT : _6 >= 0
1-1) SUC  : _6 >= 0
1-1) CONSTRAINT : _8 = 10
1-1) SUC  : 10 = 10
1-1) CALL : I1(0, 10, 0 + _6) ?
1-1) TRY  : I1(0, 10, 9) :- _15 = 0 + _16, _18 = 10 + _16, _16 >= 0, _15 = 2, I2(_15, _18, 9 + _16)
1-1) SUC  : I1(0, 10, 9) :- _15 = 0 + _16, _18 = 10 + _16, _16 >= 0, _15 = 2, I2(_15, _18, 9 + _16)
1-2) CONSTRAINT : _15 = 0 + _16
1-2) SUC  : _15 = 0 + _16
1-2) CONSTRAINT : _18 = 10 + _16
1-2) SUC  : _18 = 10 + _16
1-2) CONSTRAINT : _16 >= 0
1-2) SUC  : _16 >= 0
```

CLPの動作

Goal

```
l0(X, 1, 0)
```

CLPの動作

l0(X, 1, 0)

unification

l0(X, Y, Time):-

X1 = X + D, Y1 = Y + D, D >= 0,

Y1 = 10,

l1(0, Y1, Time + D).

CLPの動作

l0(X, 1, 0)

unification

l0(X, 1, 0):-

X1 = X + D, Y1 = 1 + D, D >= 0,

Y1 = 10,

l1(0, Y1, 0 + D).

CLPの動作

Goal

X1 = X + D, Y1 = 1 + D, D >= 0, Y1 = 10, l1(0, Y1, D)

CLPの動作

Goal

X1 = X + D, Y1 = 1 + D, D >= 0, Y1 = 10, l1(0, Y1, D)

CLPの動作

I1(0, Y1, D)

Constraints

X1 = X + D,
Y1 = 1 + D,
D ≥ 0,
Y1 = 10

CLPの動作

I1(0, Y1, D)

Constraints

X1 = X + D,
Y1 = 1 + D,
D ≥ 0,
Y1 = 10

Constraint Solver

TRUE

CLPの動作

Goal

Constraints

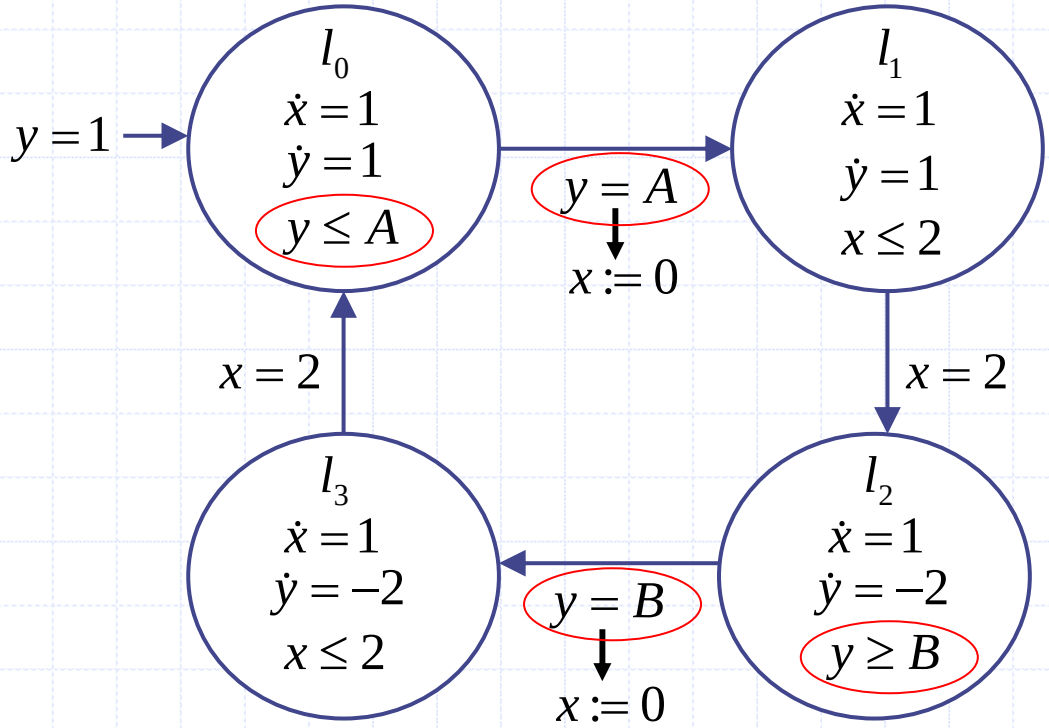
l1(0, 10, D)	with	X1 = X + D, Y1 = 1 + D, D ≥ 0

単純なモデル化では

- ハイブリッドオートマトンの記述とCLPの記述は1対1に対応する.
- CLPの実行過程はハイブリッドオートマトンにおけるロケーション間の遷移に対応する.
- ハイブリッドオートマトンの各変数はその値が時刻とともに変化する時間の関数である. 論理プログラミングでは変数の値は1つの実行パスに対して一意なので, 各時刻ごとに別の変数を用意しなければならない.
- ハイブリッドオートマトンが停止しないならば, CLPの実行も停止しない.

パラメータ設計

- 問題：水面レベルモニタにおいて，水面の高さ y を $1 \le y \le 12$ に保つようにセンサの位置 A, B を定めよ．



パラメータ設計

- 3回目以降のふるまいは2回目と同じであり調べる必要はない.
- ロケーション1に入るときの各変数の値を見ると,
  - 変数xは最初は未定, 2回目以降は $x = 2$ であるが, ロケーション1から2への遷移のガードには x は出現せず, また $x := 0$ に初期化されるので, x の値は任意で構わない.
  - 変数 y については, 最初は1であり, 2回目以降はパラメータBに依存する値($y = B - 4$)である.

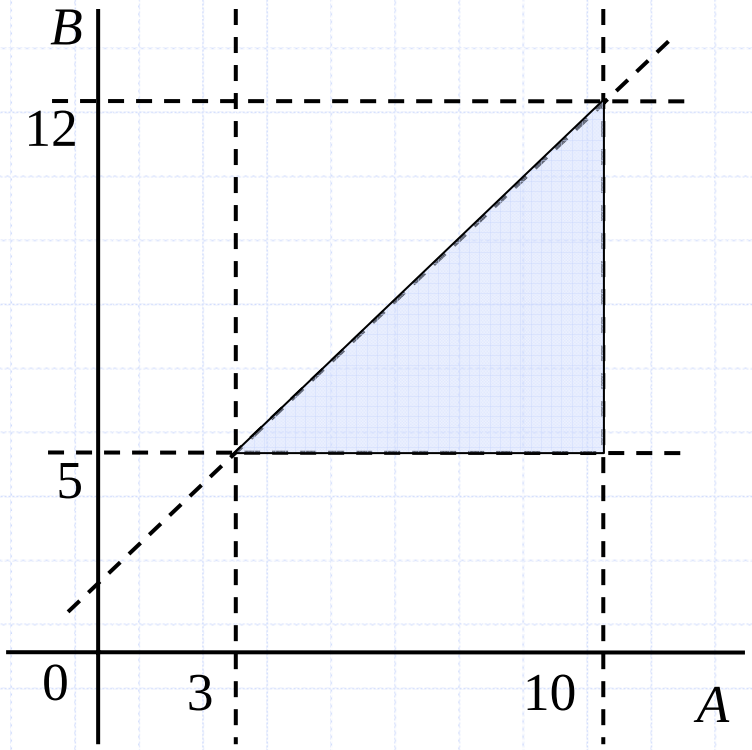
パラメータ設計

```
l0(X, Y, T, TT, [A, B]):-
        X1 = X + D, Y1 = Y + D, D >= 0,
        Y1 = A,
        spec(Y1),
        l1(0, Y1, T + D, TT, [A, B]).
l1(X, Y, T, TT, [A, B]):-
・・・

l3(X, Y, T, TT, [A, B]):-
        X1 = X + D, Y1 = Y -2 * D, D >=
        0,
        X1 = 2,
        spec(Y1),
        l0_2(X1, Y1, T + D, TT, [A, B]).
```

```
l0_2(X, Y, T, TT, [A, B]):-
        X1 = X + D, Y1 = Y + D, D >= 0,
        Y1 = A,
        spec(Y1),
        l1_2(0, Y1, T + D, TT, [A, B]).
l1_2(X, Y, T, TT, [A, B]):-
・・・

l3_2(X, Y, T, T + D, [A, B]):-
        X1 = X + D, Y1 = Y -2 * D, D >=
        0,
        X1 = 2,
        spec(Y1).

spec(Y) :- 1 <= Y, Y <= 12.
```

パラメータ設計

```
| ?-  l0(X, 1, 0, TT, [A, B]), project([A, B], Z).
X = ?
TT = 8  + _84 + _18 + _51 + _117
A = 1  + _18
B = 3  + _18 -2 * _51
Z = [1 * B >= 5, (-1) * B >= -12, (-0.5) * B + 0.5
       * A >= -1, (-1) * A >= -10, (-1) * A >= -10]
0 =  _18 - _20
-11 =  - _18 - _21
-2 =  _18 - _36
-9 =  - _18 - _37
-2 =  _18 -2 * _51 - _53
-9 =  - _18 + 2 * _51 - _54
2 =  _18 -2 * _51 - _70
-13 =  - _18 + 2 * _51 - _71
-2 =  - _84 + 2 * _51
2 =  _84 + _18 -2 * _51 - _86
-13 =  - _84 - _18 + 2 * _51 - _87
```



パラメータ設計

- 制約条件を満たすパラメータA, Bの値は図の領域であり無限に存在する.
- その中で, ポンプのon/off回数の最小化(= サイクル1回りの時間最大化)を目的関数にしたときの最適解はつぎのように求められる.

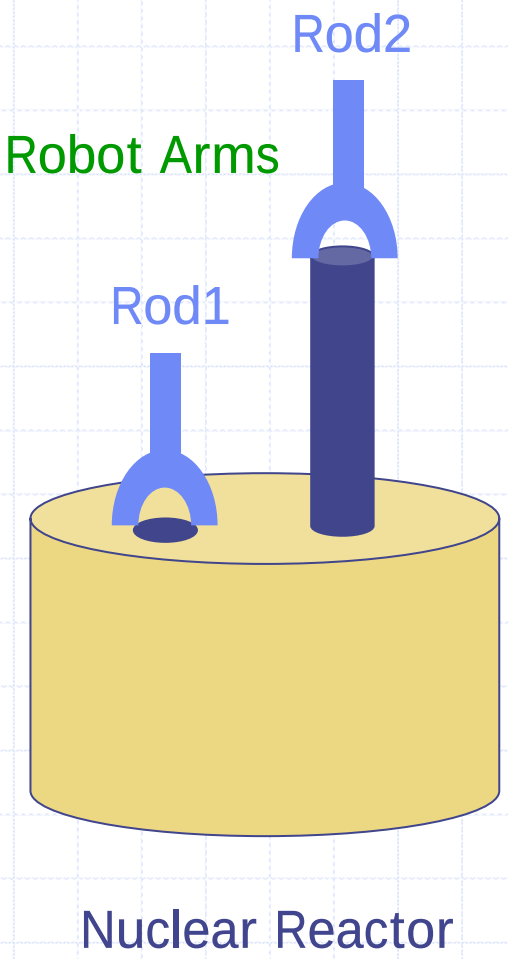
```
| ?- max(TT, l0(X, 1, 0, TT, [A, B])).
TT = 33
X = ?
A = 10
B = 5

*** yes ***
```

パラメータ設計問題：定式化

ハイブリッドシステムのパラメータ設計問題：

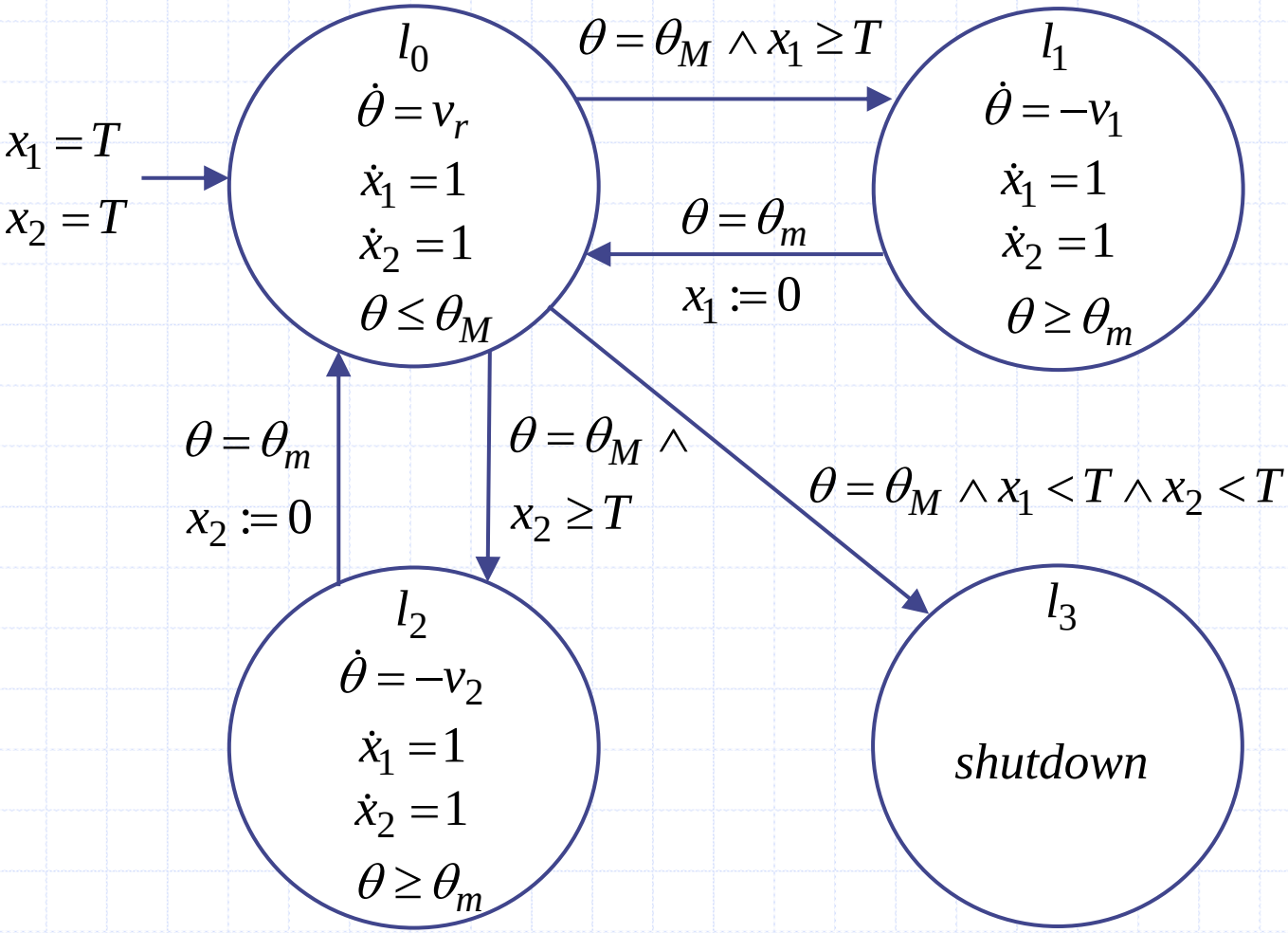
- Given: **パラメータ付きハイブリッドオートマトン，実時間CTLにより記述された動作仕様．**
- Find: **動作仕様を満足するパラメータ値の集合．**

例題



- 反応炉の温度を2つの冷却棒で制御する.
- 冷却棒は1度使用してから一定時間(T)は使用できない.
- 制御不能な場合はシャットダウンする.

例題


l_0
$\dot{\theta} = v_r$
$\dot{x}_1 = 1$
$\dot{x}_2 = 1$
$\theta \le \theta_M$
l_1
$\dot{\theta} = -v_1$
$\dot{x}_1 = 1$
$\dot{x}_2 = 1$
$\theta \ge \theta_m$
l_2
$\dot{\theta} = -v_2$
$\dot{x}_1 = 1$
$\dot{x}_2 = 1$
$\theta \ge \theta_m$
l_3
shutdown
$x_1 = T$
$x_2 = T$
$\theta = \theta_M \wedge x_1 \ge T$
$\theta = \theta_m$
$x_1 := 0$
$\theta = \theta_m$
$x_2 := 0$
$\theta = \theta_M \wedge$
$x_2 \ge T$
$\theta = \theta_M \wedge x_1 < T \wedge x_2 < T$

例題

- 動作仕様：$\mathrm{AG}_{\leq L}\,[l = l_1 \rightarrow \mathrm{EF}_{\leq 5}\, l = l_2]$

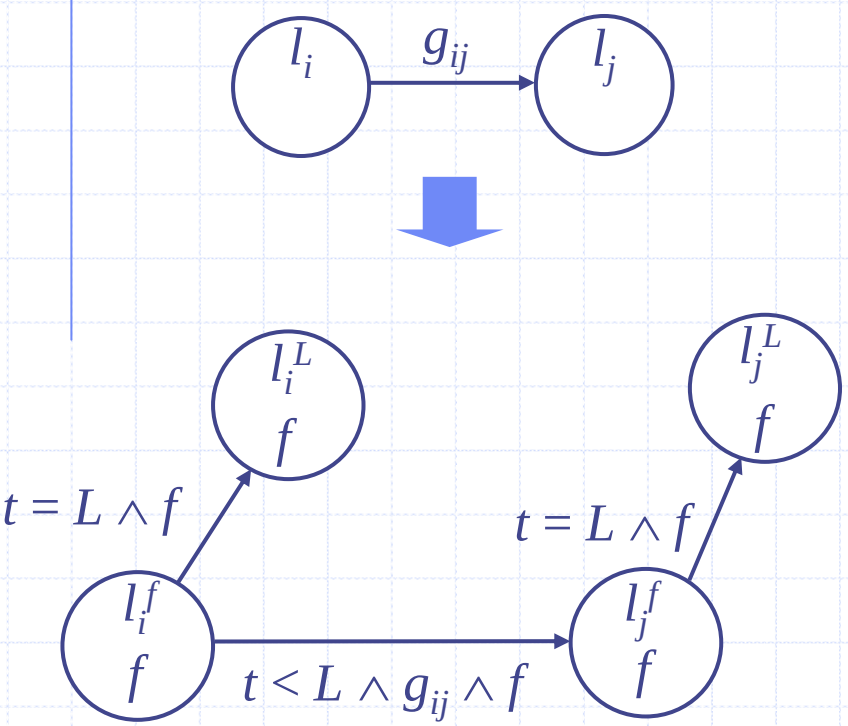
  時刻 L 以前において，ロケーションがl_1の任意の状態から，5単位時間内にl_2に遷移することが可能である．
- 問題：動作仕様を満たす**パラメータ** T の値を求めよ．
- 解法：ハイブリッドオートマトンおよびCTL論理式から，パラメータの値を計算するCLPコードを生成する．

例題

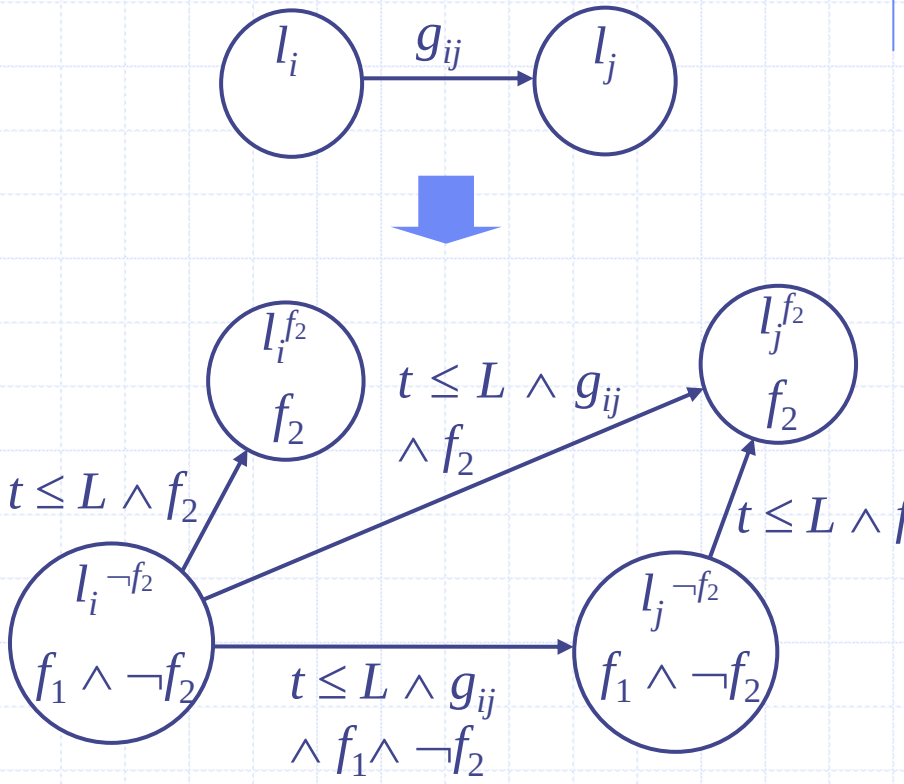
```
EF_l0(X1, X2, Temp, TT) :-
  D >= 0, vr(: Vr),
  EF_l0_next(X1 + D, X2 + D, Temp + Vr * D, TT + D).
EF_l0_next(X1, X2, Temp, TT) :-
  L(: L), TT <= L, TM(: Temp), T(: T), X1 >= T,
  EF_l1(X1, X2, Temp, TT).
EF_l0_next(X1, X2, Temp, TT) :-
  L(: L), TT <= L, TM(: Temp), T(: T), X2 >= T,
  EF_l2(X1, X2, Temp, TT).
EF_l1(X1, X2, Temp, TT) :-
  D >= 0, v1(: V1),
  EF_l1_next(X1 + D, X2 + D, Temp - V1 * D, TT + D).
EF_l1_next(X1, X2, Temp, TT) :-
  L(: L), TT <= L,
  EF2(X1, X2, Temp).
EF_l1_next(X1, X2, Temp, TT) :-
  L(: L), TT <= L, Tm(: Temp),
  EF_l0(0, X2, Temp, TT).
EF_l2(X1, X2, Temp, TT) :-
  D >= 0, v2(: V2),
  EF_l2_next(X1 + D, X2 + D, Temp - V2 * D, TT + D).
EF_l2_next(X1, X2, Temp, TT) :-
  L(: L), TT <= L,
  Tm(: Temp),
  EF_l0(X1, 0, Temp, TT).
```

```
EF2(X1, X2, Temp):-
  current_state(: [X1, X2, Temp]),
  range([T],
    (T(: T), current_state(: [X1, X2, Temp]),
      EF2_l1(X1, X2, Temp, 0)),
  [TMin], [TMax]),
  T(: T), negate(T, TMin, TMax).
…
negate(T, infeasible, _):- !.
negate(T, _, infeasible):- !.
negate(T, unbounded, unbounded):- !, fail.
negate(T, TMin, unbounded):- !, T < TMin.
negate(T, unbounded, TMax):- !, T > TMax.
negate(T, TMin, TMax):- !, (T > TMax; T < TMin).

go(T, L):-
  L(: L), T(: T), T >= 0,
  vr(: 6), v1(: 4), v2(: 3), TM(: 15), Tm(: 3), Temp(: 9),
  range ([T],
    (T(: T), Temp(: Temp), EF_l0(T, T, Temp, 0)),
  [TMin], [TMax]),
  negate(T, TMin, TMax).
```

時相オペレータの扱い



$EG_{\leq L}\ f$ の計算



$E[f_1\ U\ f_2]_{\leq L}$ の計算

時相オペレータの扱い

$\mathsf{AG}_{\leq L}\,[l = l_1 \rightarrow \mathsf{EF}_{\leq 5}\, l = l_2]$

$\neg\mathsf{EF}_{\leq L}\,[l = l_1 \wedge \neg\mathsf{EF}_{\leq 5}\, l = l_2]$

$\neg\mathsf{EF}_{\leq L}\,[l = l_1 \wedge \neg R_1(T)]$

$R_1(T)$は $\mathsf{EF}_{\leq 5}\, l = l_2$ を真にする T を表す制約

$\neg\mathsf{EF}_{\leq L}\,[l = l_1 \wedge R_1{}^c(T)]$

$\neg R_2(T)$

$R_2(T)$は $\mathsf{EF}_{\leq L}\,[l = l_1 \wedge T \in R_1{}^c(T)]$ を真にする T を表す制約

$R_2{}^c(T)$

実行例

```
| ?- go(T, 30), project([T], Z).
T =  _16
Z = [(-1) * T > -7, 1 * T >= 0]
-7 =  - _16 - _19
0 <= _16
0 < _19

*** yes ***
```

$L = 30$ **のとき,** $0 \leq T < 7$**が可能な** T **の範囲.**

今後の課題

- **領域（凸多面体）の否定の計算**：現状では領域を包含する矩形領域を計算し，その否定により近似している．これを，厳密な計算を行えるようにする．
- **最大時間指定の制限**：領域の包含判定アルゴリズムを実装する．
- 他の問題への適用．