定性推論における比較解析

九州工業大学 情報工学部
知能情報工学科
89231202 白井達也

昭和66年 2月 19日

目 次

Chapter 1

序論

本研究で我々は，物理系を対象とした，定性推論についての研究と，比較解析を応用した知的教授システム (ITS:Intelligent Tutoring System) を試作し，その特徴を評価した．

まず定性推論について述べる．人工知能の研究は，電子計算機に人間の頭脳活動が代行しえるのか，といった一連の議論から始まった．それを実現するには人間の知識，推論活動をいかにモデル化し，計算機上に実現するかが問題であった．

エキスパートシステムは，熟練者の知識を計算機上に構築したものである．そこで扱う情報は，ある事柄の真偽を表わす事実的知識と，事実関係を表わす推論の規則である．しかし人間の頭脳活動の表現は難しく，全ての情報を知識として洩れなく，矛盾することなく表現することは，まず不可能だろう．これは観測される事実が，ごく表面的な知識に過ぎないからである．

定性推論には大きな特徴がある．その一つは，扱う値が定性的であり，人間の物の考え方に良く似ていることである．例えばボールを投げる時，人は投げたボールが，いつか最高点に達し，そのうち地面に落ちて来ることを知っている．強く投げれば遠く高く飛び，重たいボールほど投げにくいことも知っている．これはニュートンの運動方程式を知っているからではなく，経験に基づいた，ボールの振舞いの因果関係に関する知識があるからである．

また対象とする系を記述するのに，微分方程式が用いられる．微分方程式は対象とする系の，時間変化に伴う状態の変化を陽に表わす関係式を記述しやすい．例えば，バケツに水を入れる場合を考える．さらにバケツに穴が空いていた場合も考え

る．流量や穴の大きさ，バケツの容積などの関係から，このバケツと水道の系の振舞いは定性的に予測できる．水を入れれば水面は上がるし，水を抜けば水面は下がる．入ってくる流量と出て行く流量の関係から，溢れるか溢れないか，平衡するか，が推論できる．

対象領域での系の振舞いが分析し尽くされ，知識ベース化されている場合，一般的なエキスパートシステムは定性推論よりも高い性能を持つかも知れない．しかし動的に変化するような系を扱う場合は，深い知識をもつ定性推論の方が有利である．定性推論は，必要な方程式が足りなくても，いまある制約式から考えられる，可能な振舞いを推論できるので，実際の系で取り得ない振舞いを答えることもあるが，少なくとも正しい振舞いも返す．

次に比較解析である．比較解析は，求まっている系の振舞いに対して推論を行う．その系のある定性状態において，所定の関数が今よりも大きかったら，少なかったら，変わらなかったらと云う知識，変動を与えた際に，その振舞いがどう変化するかを推論する．また，この変動に対する振舞いの反応が生じた理由を説明することもできる．

最も古典的なCAIにおいては，学習者が入力した答えが正しいか，否かだけを扱っていた．しかしITSにおいては，問題を解くための知識をも持ち，学習者が解答を導く途中で，その問題に必要な状況を見落としている時は，その見落としを指摘し，間違った時はどこを間違ったかを同定し，それに応じた教授を行う必要がある．

しかしその為の知識を条件毎に列挙するプロダクション・システムではなく，定性推論を用いれば，深い知識に基づくので，記述が少なくて済む．また，人の頭脳活動に近いなどの定性推論，比較解析の考えをITSの各所に応用させれば，少ない記述で，より深い理解に基づいた教授が行える．

本稿の構成を述べる．

2章では，定性推論の紹介をする．まず2.1節では，Kuipersの定性シミュレーション *Qualitative Simulation* について述べる．次に2.2節では，Weldの比較解析

Comparative Analysis [2] について述べる．最後に 2.3 節では，今回試作した ITS で用いた比較解析について述べる．

3 章では，本研究で試作した ITS について述べる．はじめに 3.1 節では，本 ITS を作成した意図と対象とする領域を述べる．次に 3.2 節では，本 ITS の仕様を述べる．次に 3.4 節では，本 ITS で利用する教材知識の構造を述べる．更に 3.5 節では，本 ITS のプログラム構造を述べる．最後に 3.6 節では，ITS に比較解析を応用した特徴を表す実行例を挙げる．

4 章では，本研究で得たことと，これからの課題について述べる．

付録 A に ITS の全プログラムリスト，付録 B に実行例を添付してある．

Chapter 2

定性推論

本章では定性推論についての説明を行う.

定性推論は，動的に変化する系のモデル化，予測，説明の過程における人間の定性的な思考プロセスをモデル化することを目的とし，対象の系を定性的にモデル化し，パラメータを関係付ける制約式に対して推論を行い，対象の系の振舞いを求める.

定性推論は定性モデルを利用するため抽象度が高く，人の理解と推論の過程によく似ている．多くのエキスパートシステムが系の振舞いを列挙して知識を記述するのに対し，定性推論はその系を特徴付ける制約式を知識として持つので記述量が少なくて済む.

定性推論の研究は Hayes の**素朴物理学**に代表されるように，1979 年あたりが始まりであると言われる． そして数年の内に，J.de Kleer , J.S.Brown , K.Forbus , B.Kuipers, , B.Williams らが中心となって幾つかの基本的な研究が行われ，それらの研究成果が 1984 年の Artificial Intelligence 第 24 巻で特集号にまとめられ，大々的に扱われるようになった．近頃は，各パラメータ間の因果関係や，不連続変化の推論，推論結果から□昧性を減らす研究などが行われている.

今回は定性推論の内，定性シミュレーションと比較解析についてを取り上げた.

2.1 定性シミュレーション

本節では，B.Kuipers の論文「 Qualitative Simulation 」[1] をもとに，定性シミュレーションについて述べる.

QSIM は入力として，関数の初期値と系の制約式の集合を受け取り，この系の取り得る可能な振舞いを出力する．

系を構成する関数と，系の構造を記述する制約式と，系の振舞いについて順に説明する．

まず関数 (function) とは，時間とともに値が変化する正当な関数であり，系 (system) の状態を表わすパラメータ，例えば力 f や加速度 a のようなもの，である．系は一つ以上の関数 f_j で表現され，系 F は関数 $\{f_1, \ldots, f_j, \ldots, f_n\}$ の集合である．正当な関数とはつぎのように定義される．

定義 2.1.1 (正当な関数)　$[a, b] \subseteq R^*$ において，以下のような場合，関数 $f : [a, b] \rightarrow R^*$ を正当な関数 *(reasonable function)* という．

(1) f が区間 $[a, b]$ で連続である．

(2) f が区間 (a, b) で連続微分可能である．

(3) f が全有界区間 *(bounded interval)* において有限個の臨界点 *(critical point)* しか持たない．

(4) $\lim_{t \downarrow a} f'(t)$ と $\lim_{t \uparrow b} f'(t)$ が，R^* 中に存在する．つまり $f'(a)$ と $f'(b)$ はそれら極値と等しい．

定性推論において関数の値とは，有限個の境界標で区切られた数直線上の境界標，もしくは隣り合った境界標間の区間である．関数の境界標と関数の distinguished time-point はつぎのように定義される．

定義 2.1.2 全ての正当な関数 $f : [a, b] \rightarrow R^*$ は，有限個の境界標の集合を持つ．境界標値は $0, f(a), f(b)$ と，各臨界点における $f(t)$ の値を含む．それと沢山の追加された値を含むだろう．

定義 2.1.3　f が正当な関数であり，t が集合 $t \in [a, b] \mid f(t) = x, x$ は関数 f の境界標値の境界要素であるなら，$t \in [a, b]$ は関数 f の *distinguished time-point* である．

すなわち distinguished time-point とは，関数 fが境界標を越したり，極限に達するような，何か重要なことが起こる瞬間である．

以上の定義をもとに，関数 fについてまとめると，つぎのようになる．

定義 2.1.4 正当な関数 f :[a,b]→ R^*は有限個の *distinguished time-point:*

$$a = t_0 < t_1 < \cdots < t_n = b$$

の集合と，有限個の境界標値

$$l_1 < l_2 < \cdots < l_k$$

の集合を持つ．

次に制約式 (constraint) とは，系の構造をモデル化して記述した知識である．制約式は，系の定性的振舞いの可能な組み合せを制限するのに用いられる．

QSIM における系の構造のモデル化とは，系において関連のある関数間の因果関係を記述することである．QSIM では微分方程式を制約式として用いる．使用できる制約式は，ADD(f,g,h), MULT(f,g,h), MINUS(f,g), M^+(f,g), M^-(f,g), DERIVE(f,g) の 6 つである．f,g,h は正当な関数である．

ADD(f,g,h), MULT(f,g,h), MINUS(f,g) はそれぞれ以下のような定性算術を表わしている．

定義 2.1.5 各制約式は，引数として正当な関数 $f, g, h : [a, b] \rightarrow R^*$を持ち，全ての $t \in [a, b]$ において右に表わす関係を表わす．

$$\begin{aligned} ADD(f,g,h) &\quad : f(t) + g(t) = h(t) \\ MULT(f,g,h) &\quad : f(t) \times g(t) = h(t) \\ MINUS(f,g) &\quad : f(t) = -g(t) \\ DERIV(f,g) &\quad : f'(t) = g(t) \end{aligned}$$

また，加算と乗算は可換であるので，

$$\begin{aligned} ADD(f,g,h) &\Leftrightarrow : ADD(g,f,h) \\ MULT(f,g,h) &\Leftrightarrow : MULT(g,f,h) \\ MINUS(f,g) &\Leftrightarrow : MINUS(g,f) \end{aligned}$$

である．

残りの $\mathrm{M}^{+}(f,g)$,$\mathrm{M}^{-}(f,g)$ は，境界標を通りかつ，至るところで全く単調に増加 ($\mathrm{M}^{-}(f,g)$ は減少) する 2 つの関数の関係を表わす，定性的関数制約 (Qualitative function constraints) である．この制約式は，定義域の両端点以外では導関数が 0 であってはならない．$\mathrm{M}^{+}(f,g)$,$\mathrm{M}^{-}(f,g)$ は，つぎのように定義される．

定義 2.1.6 $M^{+}(f,g)$ は，2 つの正当な関数 $f,g : [a,b] \rightarrow R^{*}$を引数としてもつ．$M^{+}(f,g)$ が真であるのは，全ての $t \in [a,b]$ において $f(t) = H(g(t))$ であるのと同値である．ここで Hは定義域 $g([a,b])$ と値域 $f([a,b])$ をもち，微分可能であり，定義域中の全ての x で $H'(x) > 0$ な関数である．$M^{-}(f,g)$ は $H'(x) < 0$ 以外は同様に定義される．

$\mathrm{M}^{+}(f,g)$,$\mathrm{M}^{-}(f,g)$ も同様に，

$$\mathrm{M}^{+}(f,g) \Leftrightarrow \mathrm{M}^{+}(g,f)$$
$$\mathrm{M}^{-}(f,g) \Leftrightarrow \mathrm{M}^{-}(g,f)$$

である．

最後に系の振舞いについて説明する．系の振舞いとは，時系列にそった系の定性状態の列である．定性状態の列は，ただ一列の連なりとは限らず，繰り返しループを形成したり，幾つかに分岐したりする状態遷移図を描いたりする．それぞれが可能な振舞いであり，QSIM はその全集合を出力する．系の定性状態とは，系を構成する全関数の定性状態の集合である．

系の振舞いを説明する前に，単一の関数 fの定性状態 (qualitative state) と定性的振舞い (qualitative behavior) を定義する．

定義 2.1.7 (単一関数 fの定性状態) $l_1 < \cdots < l_k$を関数 fの境界標値とする．瞬間 t_iにおける関数 fの定性状態 $QS(f,t)$ は，**関数の定性値** $qval$ とその**変化の方向** $qdir$ の対，$\langle qval, qdir \rangle$ ，である．　時区間 (t_i, t_{i+1}) における関数 fの定性状態 $QS(f,t_i,t_{i+1})$ も $\langle qval, qdir \rangle$ ，である．

(1)

$$qval = \begin{cases} l_j, & \text{if } f(t) = l_j, \text{境界標値} \\ (l_j, l_{j+1}), & \text{if } f(t) \in (l_j, l_{j+1}) \end{cases}$$

(2)

$$qdir = \begin{cases} inc, & if\ f'(t) > 0, \\ std, & if\ f'(t) = 0 \\ dec, & if\ f'(t) < 0, \end{cases}$$

定義 2.1.8 (単一関数 fの定性的振舞い)　関数 fの定性的振舞いは，fの定性状態の連なりである．

$$QS(f,t_0), QS(f,t_0,t_1), QS(f,t_1), \ldots, QS(f,t_{n-1},t_n), QS(f,t_n)$$

それでは，系の定性状態と系の定性的振舞いについて以下に定義する．

定義 2.1.9 (系の定性状態) 系 Fは関数 f_iの集合 $F = f_1, \ldots, f_m$ であり，境界標値と *distinguished time-point* の集合をもつとする．系 Fの *distinguished time-point* は，各関数 f_iの *distinguished time-point* の集合の和集合である．m 関数の系 Fの定性状態は，各関数の定性状態 m 個の組である．

$$QS(F,t_i) = [QS(f_1,t_i), \ldots, QS(f_m,t_i)],$$
$$QS(F,t_i,t_{i+1}) = [QS(f_1,t_i,t_{i+1}), \ldots, QS(f_m,t_i,t_{i+1})].$$

もし t_iと t_{i+1}が関数 f_jの distinguished time-point でないなら，t_iと区間 (t_i,t_{i+1}) は f_jの二つの distinguished time-point，t_kと t_{k+1}の間にあるはずである．　そのとき，定性状態 QS(f_j,t_i) と QS(f_j,t_i,t_{i+1}) は，QS(f_j,t_k,t_{k+1}) と同じである．

定義 2.1.10 (系の定性的振舞い)　系 Fの定性的振舞いは，Fの定性状態の連なりである．

$$QS(F,t_0), QS(F,t_0,t_1), QS(F,t_1), \ldots, QS(F,t_n).$$

系や関数の定性状態の移り変わり，遷移 (transition)，について説明する．系を構成する関数のどれかが臨界点に達した瞬間を distinguished time-point と言う．定性推論では時間軸を，この瞬間と瞬間と瞬間の間の時区間が交互に繰り返す連続した列と考える．この瞬間を境にした関数の定性状態の変化を，遷移と言う．遷移 (transition) には 2 種類あり，瞬間から時区間への遷移である P-遷移と，時区間から瞬間への遷移である I-遷移である．これらはつぎのように定義される．

定義 2.1.11 (遷移) *distinguished time-point* の t_iにおいて，fの *P*-遷移は fの隣り合った定性状態の組であり，表 *2.1*の左側にあるものだけである．

$$QS(f, t_i) \Rightarrow QS(f, t_i, t_{i+1}),$$

その遷移前の状態は *distinguished time-point* における定性状態である．*I*-遷移は，fの隣り合った定性状態の組であり，表 *2.1*の右側にあるものだけである．

$$QS(f, t_{i-1}, t_i) \Rightarrow QS(f, t_i),$$

その遷移前の状態は *distinguished time-point* 間の区間における定性状態である．

表 2.1に P-**遷移**および I-**遷移**の可能な遷移を全て挙げる．これらは，関数 f_jが連続微分可能なので，中間値の定理および平均値の定理から，求められる．

Table 2.1: 可能な遷移の一覧表

Ptran-sitions	QS(f, t_i)
P1	$\langle l_j$,std$\rangle$
P2	$\langle l_j$,std$\rangle$
P3	$\langle l_j$,std$\rangle$
P4	$\langle l_j$,inc$\rangle$
P5	$\langle(l_j, l_{j+1})$,inc
P6	$\langle l_j$,dec$\rangle$
P7	$\langle(l_j, l_{j+1})$,dec

I8,I9 で，fは l^*で std である．l^*は，$l_j < l^* < l_{j+1}$であるような，新しい境界標値である．

以上の定義に基づいて，定性シミュレーター QSIM の入出力と動作アルゴリズムを説明する．入力は制約式の集合と，各関数の初期状態である．準備として，関数記号 $\{f_1, \ldots, f_n\}$，関数間の因果関係を束縛する制約式の集合，各関数の境界標値の全順序集合 (最初は $\{-\infty, 0, \infty\}$ のみ) を用意する．時間を t_0にセットし，t_0時の各関数の定性状態 (初期状態) をセットし，現在の系の状態を ACTIVE-STATE キューに入れる．QSIM は ACTIVE-STATE キューが空になるまで，繰り返し ACTIVE な定性状態を選び出しては以下の処理を繰り返す．ACTIVE な定性状態とは，求まった可能な定性状態の内，以下に定義する停止条件に合わないものを言う．

定義 2.1.12 (停止条件) 1.No Change: シミュレーション中の関数の変化の方向 *qdir* が全て *std* になったとき．つまり全ての関数の遷移が *I1,I4,I7* のとき．こ

れ以上シミュレートしても系の振舞いに変化は現れない.

2.Cycle シミュレーション結果の履歴中に，いま処理中の系の定性状態とまったく同じものがあったとき． これ以上シミュレートしても同じ振舞いの繰り返しにしかならない.

3.Divergence: どれか一つの関数の定性値が ∞ もしくは $-\infty$ になったとき．これ以上シミュレートしても系の振舞いに変化は現れない.

1. ACTIVE-STATE キューから定性状態を一つ取り出す.
2. その定性状態が次に取り得る全ての可能な遷移を，表 2.1から求める.
3. その可能な遷移の集合に対して制限を加えるためのヒューリスティックスな処理 (constraint consystency, pairwise consistency など) を行い，(可能ではあるが) 正しくない遷移をできるだけ filtering out する．残った各関数の遷移の集合が，系が次に取り得る定性状態集合 (successor) である.
4. 残った定性状態に対して停止条件のチェックを行い，停止しない定性状態は ACTIVE-STATE キューに付け加える.
5. ACTIVE な定性状態が ACTIVE-STATE キューに残っているなら 1 へ.

Qualitative Simulation は不完全である．QSIM が出力するのは，表??から求めた可能な遷移のうち，制約式に矛盾せず，ヒューリスティックなフィルタリングにかからなかった振舞いを集めただけなので，モデル化する対象の実際の系では取り得ない振舞いを含むことも考えられる.

QSIM 以降の定性シミュレーターでは，以下の問題点についての研究が行われている.

- あいまい性について：対象系に対する定量的な情報が足りないときでも，分かっている情報を組み合せることで推論できるのが，定性推論の利点である．しかし，対象系に対して十分な定量的情報があっても，それを有効に利用でき

ていないのも事実である．定量的情報を有効に利用することは，可能な振舞いの中のにせ物を排除するのに役立つ．

- 不連続な変化について：QSIM では，全ての関数は連続微分可能である必要があった．つまりその変化の方向が一瞬にして反転する様な変化を許さなかった．例えば，ボールが壁に当たった場合である．ボールは壁に当たった途端に，その運動の方向を変えてしまう．勿論，微視的に見れば，不連続な変化は速い連続的な変化として捉えるられるかも知れないが，例外処理に外ならず，根本的な解決にはならない．

2.2 比較解析

本節では，D.Weld の論文「Comparative Analysis」を参考に，比較解析 (Comparative Analysis) については紹介する．

比較解析は系を構成するパラメータに，ある変動を与えたときに系の振舞いがどのような影響を受けるかを定性的に推論する．比較解析はこの推論過程を元にして，変動に対する系の振舞いがなぜ変化するかを説明できる．比較解析はこれらの特徴をいかして，以下に挙げるような領域での応用が考えられる．

- 設計:自動設計の一つの方法は，今ある設計の部分的変更である．規則の追加・変更・合成・削除である．この変更の効果を確かめるのに有効である．新しい設計に変動を加えた時の反応を解析するすることで，設計の部分的変更が妥当であったかどうかを判断する．同時にその反応の理由の解析は，望まない効果を打ち消すのに有用である．

- 診断:診断は対象とする実在の系の異常を発見するのが役目である．　異常が発生した原因を予想する．そして対象の系の定性モデルに　予想した異常なパラメータを変動として与えて，その変化が実際の　振舞いと一致したとき，予想した原因が答えだと思われる．

- ITS[†]比較解析は複雑な系の振舞いを説明するのに有効である．

[†]Intelligent Tutoring System:知的教授システム

比較解析を行うための手段としてD.Weldは，定性差分解析(DQanalysis)[††]と**誇張解析**(Exaggeration)を発表した．DQanalysisが比較的小さな範囲で系のパラメータを変化させたときの振舞いの変化を扱うのに対し，誇張方はパラメータの値を極端に変化させたときの振舞いの変化を推論する．今回我々はDQanalysisのみを研究した．

DQanalysisについての説明

DQanalysisは2.1で述べた定性シミュレーターQSIMの出力結果として得られる状態遷移の列と，系に加える変動を入力とする．

DQanalysisは以上の二つを入力とし，QSIMで使われたのとまったく同じ構造記述を利用して状態遷移の列の変化を推論し，出力する．

ここから先を説明するのに，いくつか定義が必要である．まず比較解析を行うためには，固有の時間を比較できるように抽象化する必要が有る．二つの系が似通っているが，実時間では違う時間に遷移をするかも知れない．以下では，変動を加えない系のパラメータFに対し，変動を加えられた系のパラメータを$\hat{F}$と書いて区別する．

定義 2.2.1 パラメータはその$QVAL$が境界標へ，もしくは境界標から変化したとき，遷移に達すると言われる．系はどれかパラメータが遷移したとき，遷移に達すると言われる．つまり遷移は*distinguished time-point*でのみ起こり，全ての*distinguished time-point*は遷移を持つ．

遷移の列を集合γ_iで示す．そして時間関数$\mathcal{T}$を導入する．時間の関数$\mathcal{T}$は引数γ_iをとり，$\mathcal{T}(\gamma_i)$である．これで二つの振舞いの遷移の瞬間が区別できる．

さきから述べられている，変動とは，具体的に云うと，”質量mを増やす”，”初期位相xを大きくする”などのような増(⇑)，減(⇓)，不変(||)を表わすRC(relative change)情報である．定義2.2.2にRCを定義する．

定義 2.2.2 (RC) パラメータFと遷移γ_iが与えられた時，γ_iにおけるFの*RC(relative change)*は以下のように定義される，

[††]Diffrencial Qualitative Analysis

$$F \Uparrow_i, \quad if\ |\hat{F}(\hat{\mathcal{T}}(\gamma_i))| > |F(\mathcal{T}(\gamma_i))|\ ,$$
$$F \|_i, \quad if\ |\hat{F}(\hat{\mathcal{T}}(\gamma_i))| = |F(\mathcal{T}(\gamma_i))|\ ,$$
$$F \Downarrow_i, \quad if\ |\hat{F}(\hat{\mathcal{T}}(\gamma_i))| < |F(\mathcal{T}(\gamma_i))|$$

以上の定義により，二つの系の各パラメータ・時間関数の，変動に対する反応をRC で表現し，比較できるようになった．

DQanalysis は沢山の推論規則といくつかのテクニックを用いる．DQanalysys は，1 ダースの例でテストされ，健全であると証明されている [2]．ただし DQanalysis のテクニックは健全ではあるが不完全である．常に終了するが，常に答えを返すとは限らない．

D.Weld はこの DQanalysis の理論をテストするプログラム CA を Symbolic LISP マシン上に構築した．CA は利用者が選んだ問題の定性的振舞いの集合を生成するために QSIM を実行する．そして求まった定性的振舞いの集合の中から，利用者が選んだ状態遷移の列一つに対し，与えられた変動を加えて推論を行う．この際に用いられる**推論規則** ARK が約 60 ある．これは，たとえ一つの推論規則でも命題が選言もしくは否定を含んでいる場合は複数の ARK を要するので，実際の推論規則の数はもっと少ない．

具体的には以下に挙げるような規則がある．

1. 継続区間規則 (duration rule)
2. 区間導関数規則 (interval derivative rule)
3. 遷移導関数規則 (transition derivative rule)
4. 自己参照規則 (self-reference rule)
5. 視点 flipping 規則 (perspective flipping rule)
6. 遷移, 区間定数規則 (transition and interval constant rule)
7. 時間終点規則 (end of time rule)
8. 自己導関数規則 (one's own derivative rule)
9. 定性算術規則 (rules from qualitative arithmetic)

全てを説明する訳にはいかないので，1の継続区間規則について述べる．

定義 2.2.3 (継続区間規則) この規則は，”距離＝割合×継続時間”を形式化したものである．もし変動が加えられた系の**割合**の方が，元の系の**割合**よりも小さいなら，同じ距離を移動するのにより長い時間を要することを表わす．

第二の振舞いの方が第一の振舞い *(QSIM* の出力*)* に比べ，

if 移動距離が短くない *and* 割合が小さい

$$then\ \hat{T}(\gamma_{i+1}) - \hat{T}(\gamma_i) > T(\gamma_{+\infty}) - T(\gamma_i)$$

即ち (γ_i, γ_{i+1}) の継続時間は長くなる

if 移動距離が長くない *and* 割合が大きい

$$then\ \hat{T}(\gamma_{i+1}) - \hat{T}(\gamma_i) < T(\gamma_{+\infty}) - T(\gamma_i)$$

即ち (γ_i, γ_{i+1}) の継続時間は短くなる

定義 2.2.4 (定性算術)

		Y		
		$\Uparrow_i$	$\|\|_i$	$\Downarrow_i$
X	$\Uparrow_i$	$\Uparrow_i$	$\Uparrow_i$	$?$
	$\|\|_i$	$\Uparrow_i$	$\|\|_i$	$\Downarrow_i$
	$\Downarrow_i$	$?$	$\Downarrow_i$	$\Downarrow_i$

問題は？の解釈である．？を何でもよい，とするか，何だか分からない，とするかである．我々はこれを何だか分からない，と解釈して規則を作成した．

例えば `add(up,X,?)` の X は表からは単純に$\Downarrow$となるが，これは？である．それに対して `add(up,down,X)` の X は表通りに？である．

比較解析はこのような推論規則を用いて，分かっているRC値から分かってないRC値を求める続け，最後には目的のパラメータのRC値を返す．

最もDQanalysisを特徴づけるテクニックが**多重視点** (multiple perspective) である．これを説明するために，図2.1 の,「理想のバネ†で壁に繋がれた 摩擦のない台上のブロック」を考える．

この系は6つのパラメータで表わされる．バネ定数$-K$, 質量 M, 位置 X, 速度 V, 加速度 A, そして力 Fである．これらがニュートンの第二法則 $(F = MA)$ とフッ

†(力)=-(バネ定数)×(変位) の関係を常に保つバネ

クの法則 ($F = -KX$) によって関係付けられる．質量とバネ定数は時間に関係無く一定値であり続ける．初期条件は $M(0) > 0, -K(0) < 0, V(0) = 0, X(0) = x_0 < 0$ である．これらを元にしてQSIMはいくつかの振舞いを生成する．その内の一つ，安定振動についての比較解析の例を示す．

もしブロックの質量を増やしたら振動の周期はどうなるか？

> 答えは”周期はのびる”である．力は位置と反対方向に比例するので，ブロックの質量が増加してもブロックに働く力は同じままだろう．しかしブロックが重くなるとそれは速く加速しないだろうし，善周期（同じ距離を移動すると仮定）を終了するのに長くかかる様になるだろう．

さて，ここで注意すべきなのは，”ブロックに働く力は同じままだろう”の実際の意味は，Fと$\hat{F}$[†] がXの全ての値で同じである，と云う点である．時間の関数Fと$\hat{F}$ は等しくないが，このFと$\hat{F}$の違いは定性的には区別できない．そこで”Xの全ての値で...”と云う，変化の比較のための基準とする軸を与える必要がある．それが**視点** (perspective) である．先の一行は，”もし質量が増加しても，力は位置の**視点**からから見て変化しない”と言い直せる．

D.WeldのDQanalysisは，**位相幾何学的に等しい** (topologically equal) 場合について述べたものを，位相幾何学的に等しくないものへと拡張されている．位相幾何学的に等しいとはどういうことか，は定義2.2.5にある通りである．

定義 2.2.5 二つの系の振舞い，Sと$\hat{S}$は，もしそれらが同じ遷移の列，$\gamma_0, \ldots, \gamma_k$, を持ち，$0 \leq i \leq k$であるような全ての$i$において，

$$QS(S, \mathcal{T}(\gamma_i)) = QS(\hat{S}, \hat{\mathcal{T}}(\gamma_i)),$$

であり，$0 \leq i \leq k$であるような全てのiにおいて，

$$QS(S, \mathcal{T}(\gamma_i), \mathcal{T}(\gamma_{i+1})) = QS(\hat{S}, \hat{\mathcal{T}}(\gamma_i), \hat{\mathcal{T}}(\gamma_{i+1}))$$

であるなら位相幾何学的に等しい．

二つの振舞いが位相幾何学的に等しいのなら，それらの各々の境界標の集合は同じ順序関係を持つ．しかし境界標の潜在的な実際の値は違っても構わない．

[†] 変動を加えられた場合のF

2.3 微分制約を用いない比較解析

本節では，今回作成した比較解析について，2.2 節の D.Weld の比較解析 DQanalysis と比較しながら説明する．

今回作成した比較解析は，D.Weld の物とは微分制約 DERIV(f,g) が使えない点で異なる．微分制約とは，正当な関数 $f, g : [a, b] \rightarrow R^*$が，全ての $t \in [a, b]$ において，$f'(t) = g(t)$ であるような関係を表わす制約式である．

比較解析の制約式として DERIV(f,g) を用いると，推論規則が沢山必要になり，実現が困難である．微分制約を使わない場合は，対象として扱える系が，微分方程式で表わせる系から，多変数方程式だけで記述できる，静的な系に限られてしまう．静的な系とは，時間の経過に伴う系の状態遷移がない系のことを指す．しかし，物理の系には静的な系の問題も多く存在するので (e.g. 電気回路，等速回転運動の角速度と遠心力の関係など)，今回は制約式 DERIV(f,g) なしの比較解析を試作してみた．

この比較解析は入力として，制約式の集合と，変動の集合と，どの関数の RC 値を求めたいか，の 3 つを受け取る．そして推論結果として，目的の関数の RC 値と，その反応が生じた理由を返す．

D.Weld の DQanalysis は，QSIM の出力結果である系の振舞いを入力として受け取ったが，今回の比較解析では系の振舞いを利用しない．定性シミュレーター QSIM で扱う系は，時間の経過に伴う系の状態遷移がある系であるので，今回作成した比較解析では扱えないからである．

系を構成するパラメータと，系の構造を記述する制約式について説明する前に，系を構成するパラメータ $\{f_1, \ldots, f_j, \ldots, f_n\}$ について以下のように定義する．

定義 2.3.1　パラメータ f_j は，制約式で関係付けられた値の関係を保ち続けることのみを意味するものであり，パラメータ f_j 自体の値は意味を持たない．

まず変動について説明する．今回の比較解析で用いる変動は，D.Weld の DQanalysis と同じく，各パラメータに与える RC 値である．ただし，RC 値は D.Weld の扱っている物のサブセットであり，以下の 2 つの点で異なる．視点がないことと，遷移がないことである．よって以下のように定義される．

定義 2.3.2 (RC 値) パラメータ fの値は，第一のシミュレーション結果の値を想定する．それに対して推論規則を用いて求まった第二のシミュレーション結果を$\hat{f}$ とした時，以下のように定義される．

$$
\begin{aligned}
&f \Uparrow, \quad if\ |\hat{f}| > |f|\ ,\\
&f \Vert, \quad if\ |\hat{f}| = |f|\ ,\\
&f \Downarrow, \quad if\ |\hat{f}| < |f|
\end{aligned}
$$

そして制約式について論じる．制約式には，ADD(X,Y,Z), MULT(X,Y,Z), MINUS(X,Y), M^{+}(X,Y), M^{-}(X,Y), CONSTANT(X) の 6 つがあり，各制約式の規則を以下に定義する．

定義 2.3.3 (ADD(X,Y,Z)) もしパラメータ X,Y,Zが，制約式 $ADD(X,Y,Z)$ で関係付けられているなら，Zの可能な RC値は以下のように表わされる．

		Y		
		$\Uparrow$	$\Vert$	$\Downarrow$
X	$\Uparrow$	$\Uparrow$	$\Uparrow$	$?$
	$\Vert$	$\Uparrow$	$\Vert$	$\Downarrow$
	$\Downarrow$	$?$	$\Downarrow$	$\Downarrow$

定義 2.3.4 (MULT(X,Y,Z)) もしパラメータ X,Y,Zが，制約式 $ADD(X,Y,Z)$ で関係付けられているなら，Zの可能な RC値は $ADD(X,Y,Z)$ と同じ表から求めることができる．

定義 2.3.5 (MINUS(X,Y)) もしパラメータ X,Yが，制約式 $MINUS(X,Y)$ で関係付けられているなら，X,Yの可能な RC値は以下の表の横の組で表わされる．

X	Y
$\Uparrow$	$\Downarrow$
$\Vert$	$\Vert$
$\Downarrow$	$\Uparrow$

定義 2.3.6 (M^{+}(X,Y)) もしパラメータ X,Yが，制約式 $M^{+}(X,Y)$ で関係付けられているなら，X,Yの可能な RC値は以下の表の横の組で表わされる．

X	Y
$\Uparrow$	$\Uparrow$
$\|\|$	$\|\|$
$\Downarrow$	$\Downarrow$

定義 2.3.7 ($M^{-}(X,Y)$)　もしパラメータ X,Yが，制約式 $M^{-}(X,Y)$ で関係付けられているなら，X,Yの可能な RC値は以下の表の横の組で表わされる．

X	Y
$\Uparrow$	$\Downarrow$
$\|\|$	$\|\|$
$\Downarrow$	$\Uparrow$

定義 2.3.8 (CONSTANT(X))　もしパラメータ Xが，制約式$CONSTANT(X)$で関係付けられているなら，Xの可能な RC値は $\|\|$ のみである．

今回作成した比較解析のアルゴリズムを説明する．

推論規則は前述の，制約式に関する規則だけである．これらの規則は，n 引数の制約式の場合，n-1 個の引数の RC 値が分かれば，残りの一つの引数の RC 値が分かることを使っている．

準備は，入力された制約式の全集合を用意し，各パラメータに与えられた変動を加え，反応を求める目的のパラメータを用意することである．

1. 目的のパラメータを含む制約式を，制約式の集合から探す．
2. 残りの引数の RC 値が求まっていないなら，それらを目的のパラメータとし，制約式の集合から今の制約式を除いた集合を制約式の集合として 1. 比較解析を行う．
3. 残りの引数の RC 値が求まったら，推論規則から目的のパラメータの RC 値を求める．
4. 求まった RC 値を，再参照できるように登録する．
5. 目的のパラメータの求まった推論過程を保存する．この推論過程は，後で理由の説明に用いられる．

推論過程(理由)は具体的には，目的のパラメータが，どの推論規則より，どのパラメータと，どのパラメータをもとにして求められたかを表わす．そしてそのパラメータが求まった理由が別にそれぞれある．これを辿れば目的のパラメータのRC値が求まった理由を説明できる．

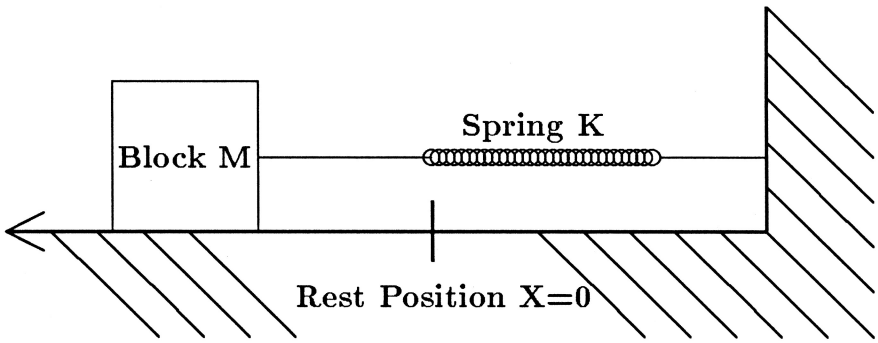

Figure 2.1: バネ／ブロックの系

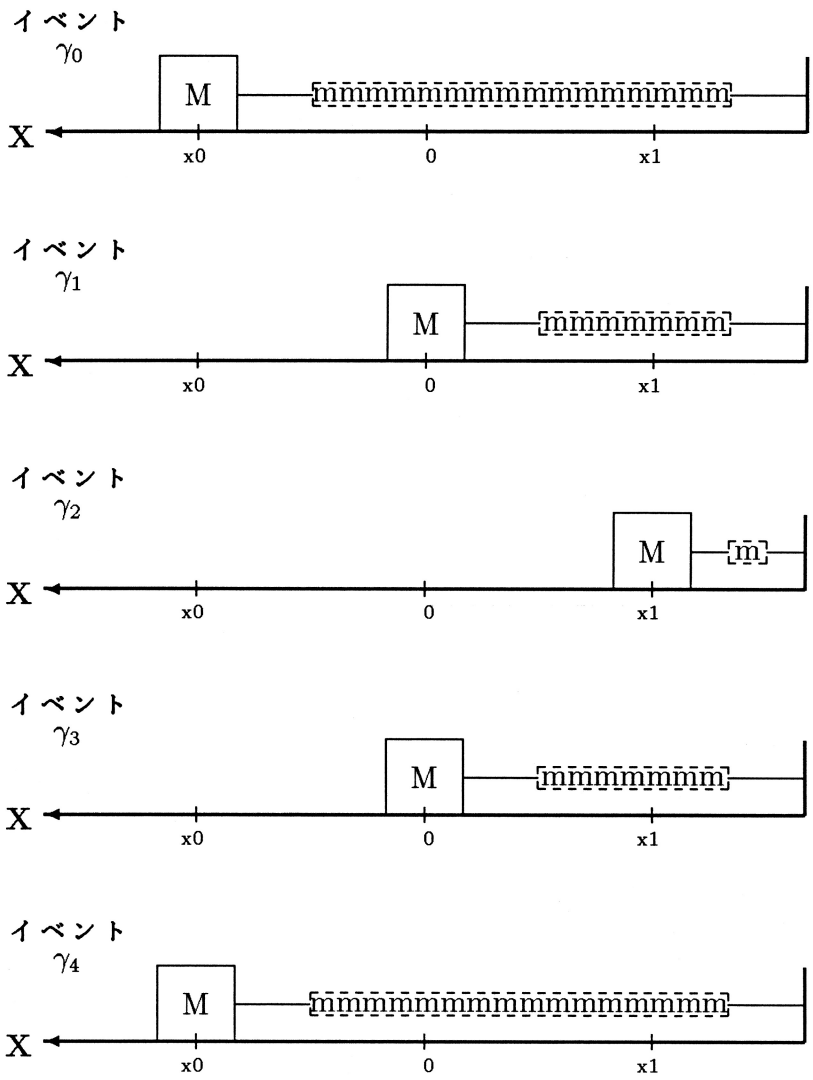

Figure 2.2: 安定振動の振舞い

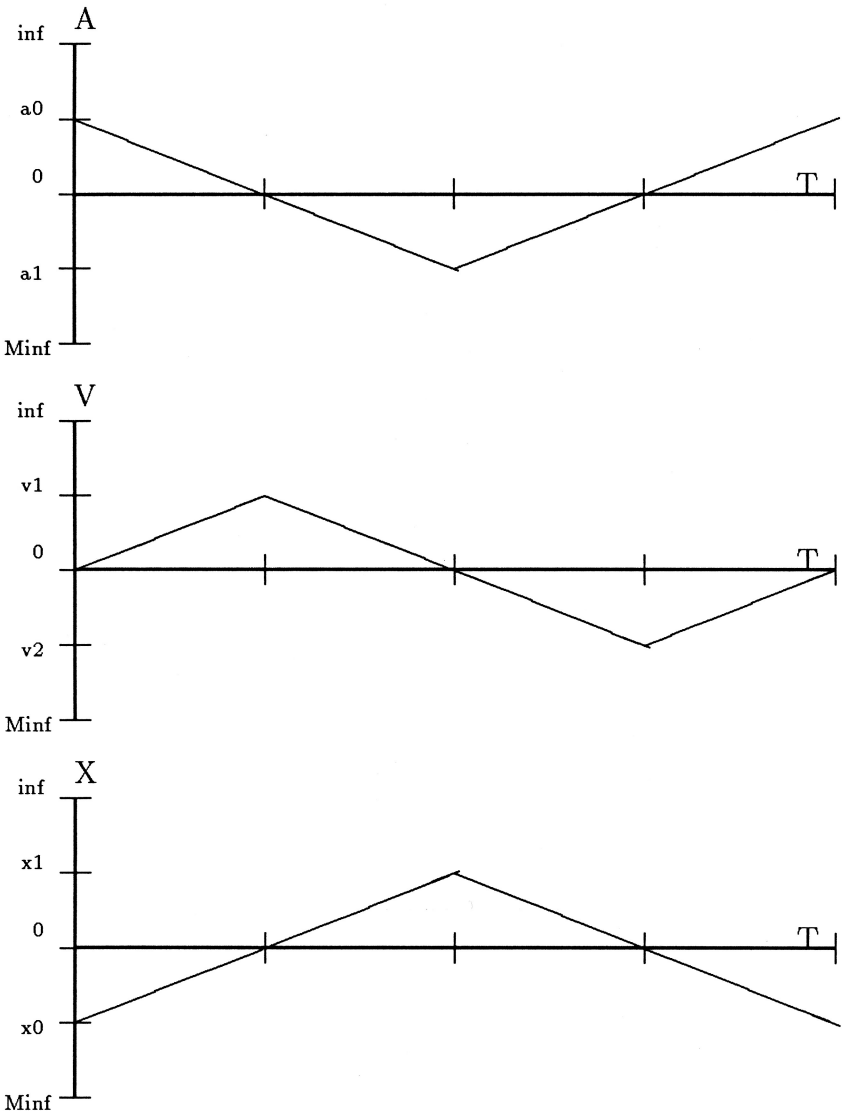

Figure 2.3: QSIM による安定振動の振舞い

Chapter 3

比較解析を応用した物理系ITSの試作

この章では今回試作したITSの説明を行う．その目的，仕様，教授知識の構造，プログラムの構成，そして実行例とその評価である．　これは実際に作成したプログラムに基づいており，プログラムリストは付録A，実行結果は付録Bを参照するように．

3.1　作成目的

定性推論を用いる対象としては，方程式で構造記述を表わせる物理系が効果的である．対象とする系を定性的に捉えることで，系の特徴を人が考えるのに近い感覚で捉えることができるからである．

さて，ITSに定性推論がどの様に応用できるか．まずは2.1で説明した定性シミュレーションであろう．定性シミュレーターを用いれば，定量的に無限に近い振舞いを持つ複雑な系でも，代表的ないくつかのパターンに分類して扱える．前述のバネの系で云えば，大きく分けて3つのパターン†に分類できる．この様に定量的に考えたらパラメータの組み合せの数だけ振舞いがありそうな複雑な系でも，定性的に考えることで随分とその振舞いを理解し易くなる．

そして第二に，2.2で説明した比較解析である．比較解析は，系に変動を与えた場合の**反応の予測**，そしてその**反応の理由の説明**ができる．今回はこの比較解析による系の変化の予測と説明をITSに応用した．

比較解析を用いた教授手順として，以下の手順を考えた．目的は対象とする系の

†振幅が発散，収束するか，臨界振動するか

構造記述を理論的に理解させ，その振舞いを直感的に予測できる程度に理解させることである．

A. 系に変動を与えた場合の反応を学習者に予測させる問題．系の構造記述は学習者に与えず，事前にその系の概要だけを説明する．もちろん間違う可能性が高いが，それは問題ではない．軽い問答を繰り返すことで，系の構造を頭にイメージさせるのが目的である．

B. あるパラメータを特徴づける構造記述を求める問題．例えば,「以下の 電流 i を求める式のうち、正しいものを答えよ。」など．本ITSでは事前に用意された構造記述の中から正しい構造記述を選択させる方式を取っている．間違った場合は，反例を示すことで間違いであることを理解させ，正解を導く手助けをする．

C. A. 同様，系に変動を与えた場合の反応を学習者に予測させる問題．しかしA. と違って，正解するまで部分問題を出したり，要求に応じてヒントを与え，正解した場合はその反応が現れる理由を説明する．この説明を見て学習者は，系の構造記述に対する自分の考えが正しかったかを知る．このタイプの問題は，系のパラメータ間の因果関係を理解するための問題である．

このA. からB. を段階的に行うことで，学習者は系の構造記述（方程式）を理解し，その方程式が表わしている意味を直感的に理解できるだろう．本ITSではこの内，B. とC. を実現している．

B. については，教材作成者が事前に用意した構造記述の中からの選択だけでなく，学習者に一から構造記述を考えさせることも考えた．そうすれば，教授戦略にもっと幅が出るし，教材作成者の労力も大幅に減少する．しかし，入力された式を制約式に変換する手順と，学習者の入力した間違った構造記述に対する適切な反例を生成する手順のプログラム化が困難であったので，今回は見送った．

2.3 で述べた様に，本ITS は Weld の DQanalysis のサブセットであるので微分制約が使えない．この為，時間の変化によって系の振舞いが変化するような系には適用できない．例えば前述のバネ／ブロックの系や電子回路の過渡応答など．逆に

云えばこの制限が克服されれば，本ITSの問題に加えてより巾広い領域の問題を扱えるようになるだろう．

3.2 仕様

本ITSでは，3.1で述べた学習手順のうち，

1. **制約式の選択**

2. **変動の反応の予測**

が実現されている．

1.は，制作者の作成した系の構造記述と，意図して作成された間違いの構造記述を元にして問題を作る． 学習者に与える情報は，系の名称，図，コメントなどであり，構造記述は与えない． 学習者は箇条書になった構造記述の中から正しいと思う番号を入力する．

対話型のシェルである． 入力できるのは，

- 答え：番号
- ヒントの要求(help)：正しい構造記述に変動を与えた場合の反応をヒントとして返す．
- 対象系の説明(world)：構造記述以外の 系の情報の再表示．
- 現在の問題(proble)：現在の問題の再表示．
- 質疑応答(question)：正しい構造記述に対する比較解析が行える． ヒントでは制作者の用意した変動に対する反応しか観察できないので，対話形式で，どのパラメータをどうするか[†]の変動，そしてどのパラメータの変化が知りたいかを入力すると，比較解析を行い，目的のパラメータの反応を返す．
- 問題を飛ばす(skip)：問題を**正解した**ことにして先へ進む． その問題をやりたく無いときに有用である．

[†]増加(↑)，減少(↓)，不変(‖)，不明(?)

である．　括弧内は実際に入力する場合のコマンドである．　頭文字が太字のものは，頭1文字だけで識別しているので，problem ならp とだけ打てばよい．

2. は，制作者が事前に作成した変動と，系の構造記述のみを用いる．　ITS は問題を出す前に一度だけ比較解析を行う．　その後の部分問題，答え合わせや説明は，先に行った比較解析時に生成された証明木を元に行う．　1. と異なり，系の情報として構造記述を見ることができる．

これも 1. と同様に対話型のシェルである．

- 答え：変化—up(↑),down(↓),eq(‖),?(？)
- ヒントの要求 (help)：いま求めようとしているパラメータがどの制約式から求まるかを，証明木を元にして提示してくる．　自発的に部分問題へ下りることもできる．
- 対象系の説明 (world)：構造記述も含めた 系の情報の再表示．
- 現在の問題 (proble)：現在の問題の再表示．　部分問題に移っている場合に，いま解こうとしている対象が分からなくなることがある．
- 質疑応答 (question)：正しい構造記述に対する比較解析が行える．　対話形式で，どのパラメータをどうするかの変動，そしてどのパラメータの変化が知りたいかを入力すると，比較解析を行い，目的のパラメータの反応を返す．
- 問題を飛ばす (skip)：問題を**正解した**ことにして先へ進む．　その問題をやりたく無いときや，間違って部分問題に入ったときや，部分問題を解いている途中で主問題の答えが分かったときなどに有用である．

である．　Prolog であるので，各コマンドや答えの後に「.」を付け忘れないよう注意すること．

教材知識と ITS プログラムが独立しているので，教材を作成する際に ITS プログラムに関する知識が必要無い．　また，別の教材知識を読み直すことで，即座に別の教授領域に移れる．

学習者が間違えた場合，比較解析を用いた反例を提示ができる．　反例は，**制約式の選択**の問題の時にのみ利用される．　いくつかある制約式群の中から学習者が選んだ制約式が，正しいものでなかった場合，ITS は予め与えられた変動を正解の制約式に与え，その反応を学習者に提示する．　そして，学習者が選んだ（間違った）制約式にも同じ変動を与えてみる．　そして，その反応を先に求めた正しい制約式の反応とが異なることを学習者に示す．　これが本 ITS で用いられている反例である．

本 ITS は，学習者の理解度によって問題の難度を変えたり，間違い原因を同定したりは出来ない．本 ITS では決められた問題を決められた順番に解いて行くようにしかできていない．　しかし部分問題を自動的に提案してきたり，解きたくない問題を飛ばすことができる点で，ドリル方式の ITS とは異なる．

本 ITS は，系の振舞いを制約する構造記述を表わす制約式として $add(A,B,C)$, $mult(A,B,C)$, $minus(A,B)$, $constant(K)$, $mP(A,B)$[†], $mM(A,B)$[††]が用意されている．　各制約式は以下に挙げる定性算術を意味する．

$$
\begin{aligned}
add(A,B,C) &\quad : A+B=C \\
mult(A,B,C) &\quad : A\times B=C \\
minus(A,B) &\quad : A=-B \\
mP(A,B) &\quad : A=h(B), h(X)'>0 \\
mM(A,B) &\quad : A=h(B), h(X)'<0 \\
constant(K) &\quad : K\text{は定数}
\end{aligned}
$$

割り算 $A=B\div C$ は $mult(A,C,B)$ と記述すれば良い．　また，引算 $A-B=C$ は add と $minus$ を組み合せるか，3.3 節で説明するマクロを使う．

3.3　マクロの説明

本節は，制約式のマクロ表記の意味を，利用面と動作面から説明する．

[†]Kuipers の論文 [1] では $M^{+}(A,B)$ と表記
[††]同論文では $M^{-}(A,B)$ と表記

本 ITS で利用した比較解析には，制約式を組み合せてマクロを定義する機能がある．マクロとは，複数の制約式を組み合せて，一つの制約式として扱うことである．例えば $equal(A,B) : A = B$ ならば，制約式 $minus(X,Y)$ を用いて，次のように定義できる．

$$minus(A,F1), minus(F1,B)$$

$F1$ はローカルなパラメータであり，マクロ内でのみ有効であるので，$equal(A,B)$ と $equal(C,D)$ の $F1$ はまったく無関係であるので，制約式を記述する際に，余分なパラメータ $F1$ について，注意を向ける必要はない．同様に，並列に接続された 2 抵抗 R1 と R2 の合成抵抗 R を求める式 $1/(1/R1+1/R2)$ のマクロ paraR(R1,R2,R) は以下の様になる．

$$mult(F1,R1,1), mult(F2,R2,1), add(F1,F2,F3), mult(R,F3,1), constant(1)$$

そしてもう一つの工夫は，マクロの定義ができることである．同じ形の関係式を複数の関数の組が用いている場合に，このマクロを用いて制約式を記述すると，記述するのが楽なうえに，見やすくなる．また，比較解析で作成される，変動の反応の理由も見やすくなる．

マクロの利点は 3 つある． 一つは，同じ関係式を複数のパラメータの組み合せで用いたい場合． 上記の合成抵抗を求める公式は，複雑な電気回路系を扱う場合，必ず必要になる． 例えば，

$$RA = \frac{1}{(\frac{1}{R1}+\frac{1}{R2})}, RB = \frac{1}{(\frac{1}{R3}+\frac{1}{R4})}, RC = \frac{1}{(\frac{1}{R5}+\frac{1}{R6})}, RR = \frac{1}{(\frac{1}{RA}+\frac{1}{RB})}, R = \frac{1}{(\frac{1}{RC}+\frac{1}{RR})}$$

の様な場合，制約式は mult が 15 個，add が 5 つ必要である． paraR(A,B,C) が定義してあれば，

$$paraR(R1,R2,RA), paraR(R3,R4,RB), paraR(R5,R6,RC), paraR(RA,RB,RC), paraR(RC,RR,R)$$

の 5 つで済む． もう一つは，先の例での $F1$ のような使い捨てのパラメータの推論を見えないように隠蔽するためである． 上記の $paraR(A,B,C)$ を使わなかった場合，$3 \times 5 = 15$ 個[†]のパラメータが余分に現れ，比較解析はこれらの変化に関する証明木までも生成してしまう． 明らかに無駄である． マクロを用いればマクロ中の推論は外からは見えず，影響も与えない． そして最後が，公式に名前を付けたい場合である． オームの法則 $V = I \times R$ はただの $mult(I,R,V)$ であるが，これでは説得力がない． $ohm(A,B,C)$ を「オームの法則 $C = A \times B$」とテンプレートともども定義すれば説明時に，

「オームの法則 $V = I \times R$ において $\cdots$」

と表示される．

[†]各 $paraR(A,B,C)$ 毎に F1,F2,F3 を要するので 15 個

3.4 教材知識の構造

本節では，教材知識の構造と，その特徴を述べる．

我々は，教材を手軽に作れるように以下の２つの工夫をした．

まず第一に，データをプログラムとは別のファイルに分離した．所定のフォーマットさえ守れば，プログラム本体に関する知識は必要ない．また，問題を教材毎にファイル化しておけば，ファイルを変えるだけで即座に別の教材領域に切り替えられる．

もう一つの工夫は，問題の種類をパターン化して用意したことである．教材知識には問題の種類の番号と，その種類の問題に必要な情報を与えるだけで教材が作成できる． 各問題形式に固有の制御構造をプログラム本体内で持つので，ヒント・反例・部分問題の提案・説明などの処理の制御命令を教材中に持ち込まないで済む．具体的には

1. 制約式の選択
2. 変動の反応の予測

である．詳しくは 3.2 に譲る．ここでは，それぞれの問題の型の区別を番号 1,2 とすることだけを明確にしておく．

教材制作者が作成する必要のある知識は，

1. 教材知識 `teaching_material/2`
2. マクロフォーム `macro/2`
3. マクロ表示用テンプレート `explain_template/1`

である．

１の教材知識から説明を始める．ここでは，パラメータのことを関数と呼ぶ．

一番大きな分類はボリューム単位の分類である．ドリル一冊に相当する単位であり，ファイル別に分けて管理する．そしてその次が `teaching_material/2` である．各 `teaching_material/2` は同じ構造記述をもつ物理系単位でまとめられる．

```
teaching_material(World,Problems).
```

- **World:** 系に関する情報を保持
- **Problems:** 問題 (Problem) をリストで保持．先頭にある問題から順に出題される

```
world(message(MSGs),VisibleFunctions,Constraints).
```

・**MSGs:** 系の説明を保持．`title(Strings)` で この系の題名を，`figure(Fig)` で図形を，それ以外は文字列として扱う

・**VisibleFunctions:** 関数記号と関数名を対応させるテンプレートをデータベースに登録するための `visible(_,_)` をリストで保持
例) `visible(v,'速度ｖ'),visible(r,'回転半径ｒ')`

・**Constraints:** この系の構造記述を表わす制約式をリストで保持．

Problem は問題の型によって異なる．
`problem(1,Words,Entries,Answer)`.

・**1:** 問題の型番号１　制約式の選択

・**Words:** 問題の文章

・**Entries:** 制約式の候補 `entry(No,Words,Constraints,QCEs`[†]`)` をリストで保持

  ・**No:** この候補の番号

  ・**Words:** この候補の構造記述の式を表現する文字列

  ・**Constraints:** この候補の構造記述をリストで保持するが，正解のエントリーの場合は World 中の Constraints を利用するので，空 ([]) で構わない

  ・**QCEs:** この候補の反例のため変動 `qce(TargetFunction,Perbutations)` をリストで保持

    ・**TargetFunction:** 反応を求める関数の記号

    ・**Perbutations:** 変動 `inference_rule/2` をリストで保持

・**Answer:** 正しい答えの番号

`problem(2,Perbutations,rc(TargetFunction,RCvalues`[†]`)`.

・**2:** 問題の型番号２　変動の反応の予測

・**Prebutations:** 変動 `inference_rule/2` をリストで保持

・**TargetFunction:** World 中の正しい Constraints に対して Perbutations の変動を与えた時の，TargetFunction の値を求める

・**RCvalue:** その結果．この値が返って来ても，いまのシステムでは役に立たないので，_にしておく

[†]Qualitative Counter Example:定性的反例
[†]Relative Change value

2. のマクロフォーム macro/2 の説明を始める．マクロは，一つ以上の引数をもち，1つ以上の制約式で構成される制約式の集合である．

マクロの記述方法は，macro([マクロ名，引数 1，引数 2,...，引数 n],[制約式群]) である．例えば 3.2 で取り上げた合成抵抗のマクロは以下のように定義される．

```
macro([paraR,A,B,C],[ constraint(mult,f1,A,1),
                      constraint(mult,f2,B,1),
                      constraint(add,f1,f2,f3),
                      constraint(mult,C,f3,1) ]).
```

引数に対応する関数だけ大文字†，他の関数は各々適当な小文字の名前を付ける．

そして最後に 3. のマクロ表示用テンプレート explain_template/1 の説明である．これまで説明した explain_template/1 は teaching_material/2 や macro/2 と異なり述語である．マクロ表示用テンプレートは，比較解析の理由の説明や，ヒントなどでマクロを画面表示する時に見やすくするための述語である．具体的には以下の様なものである．

```
explain_template(ohm(A,B,C))   :-
        writel(['オームの法則： ',C,' ＝ ',A,' × ',B]).
```

こうすれば，

ohm(i,r,v) において...

と表示される代わりに，

「オームの法則 V=I\times R において...」

と表示される．

3.5 プログラムの構成と動作

この節では今回試作した ITS の構成と動作原理を，付録 A のプログラムリスト中の述語名を用いて説明する．このプログラムはパーソナルコンピュータ PC9801 上の RUN-Prolog と，Sun ワークステーション上の SICStus-Prolog での動作を確認済みである．

本システムは，ca.pl と ca_ex.pl と writeln.pl からなる．プログラム ca.pl (比較解析+説明機構ユニット) と，ca_ex.pl (ITS システム) と，writeln.pl (出力制御ユニット) は常にロードしたままにする．ここに各種教材，例えば data.pl，を読み込む．各教材は複数の teaching_material で構成されている．教材を読み込んだら，its_start で ITS は起動，教材を読み込んだ順に学習者へ教授してゆく．

最初に比較解析及び証明木説明プログラムの説明をした後，ITS についての説明を行う．

†Prolog で先頭一文字が大文字のアトムは変数

3.5.1 比較解析及び説明プログラム

今回作成した比較解析はD.Weldの作成したDQanalysisの推論規則の内の，定性算術に関する推論規則のサブセットである．

比較解析ルーチン `inference_rule/4` は，制約式と変動を与えると，指定した関数[†]についての反応 (RC:relative change) を後ろ向きに求め，その推論過程を記録する．この推論過程は，説明プログラム `explain_tree/2` を用いて説明できる．

`inference_rule/4` は呼び出されると，目的の関数を含む制約式を構造記述の中から探し出す．その制約式が3引数なら残りの2引数の反応 (RC) を目的の関数とした `inference_rule/4` をそれぞれ再帰的に呼び出す．この際に気を付けるのは，一つ下のレベルの `inference_rule/4` には，今回探し出した制約式を取り除いた[††]構造記述を渡すこと．そうしないと無限ループに陥る．

残りの引数の反応 (RC) が求まったなら，それをもとにして目的の関数の反応 (RC) を求める．具体的には，`arithmetic_table` を用いる．

この推論結果を `inference_rule(目的の関数,RC\footnote[2]{与えられた変動に対するパラメータの反応:Relative Change})` と，データベースに登録する．こうしておけば次に他の制約式で，この関数の反応が必要になった時に余分な推論をしないで済む．そして最後に証明木を作成して終了する．

これらが再帰的に行われるので，証明木には比較解析の推論過程が，木構造になって生成される．以下に証明木の定義と例を示す．`:- because(above)` とあるのは，その関数がデータベース `inference_rule/2` をもとにしたことを意味する．

『証明木の定義』

Tree = rc(TF,RCv) :- because(Formula,(Tree1),(Tree2),...)

TF(Target Function)	: この関数が
RCv(Related Change value)	: こうなるには
Formula(制約式)	: この制約式で
Tree1,Tree2,...	: 各引数がこの様な反応をするからである．

『証明木の例』

```
rc(r,down) :- because(ohm(i,r,v),
 (rc(v,eq) :- because(above)),
 (rc(i,up) :- because(add(i1,i2,i),
   (rc(i1,up) :- because(ohm(i1,r1,v),
     (rc(v,eq) :- because(above)),
     (rc(r1,down) :- because(above)))),
   (rc(i2,eq) :- because(ohm(i2,r2,v),
     (rc(v,eq) :- because(above)),
```

[†]いわゆるパラメータ，力fや電流iなど，のこと．

[††]`select/3` を用いれば探し出すのと，削除の両方が同じに行える．

```
(rc(r2,eq) :- because(above))))))
```

マクロを使った制約式を推論する inference_rule/4 は，制約式の引数の数が不定なのでリスト処理を要する．またマクロ内の隠蔽性を高める為に，マクロ内の推論結果を残さない様に工夫をした．マクロを推論中の inference_rule/4 は，inference_rule(_,_,_,[]) と，証明木の出力として使っている第4引数 Tree==[] で区別している．

定性算術の推論規則はテーブルである．以下に例として add の場合のテーブル[†]を示す．

		Y		
		$\Uparrow_i$	$\Vert_i$	$\Downarrow_i$
X	$\Uparrow_i$	$\Uparrow_i$	$\Uparrow_i$	?
	$\Vert_i$	$\Uparrow_i$	$\Vert_i$	$\Downarrow_i$
	$\Downarrow_i$	?	$\Downarrow_i$	$\Downarrow_i$

これらのテーブルをもとにして，我々は以下の規則を作成した．

```
arithmetic_table(add,_,_,_) , arithmetic_table(mult,_,_,_),
arithmetic_table(minus,_,_) , arithmetic_table(mP,_,_),
arithmetic_table(mM,_,_)
```

ちなみに，制約式 constant(K) は常に K が $\Vert$ であると云う意味なので，特にテーブルを作る必要はない．

証明木説明ルーチン explain_tree(TF,Tree) は，与えられた関数 TF が求まった理由を，証明木 Tree を元にして説明する．これは，上記の証明木を:- because の前と後に分解して，テンプレートに当てはめて表示する再帰プログラムである．

3.5.2 ITS 本体

```
its_start :-
    teaching_material(World,Problems),   %% 問題の取り出し
    set_visible_function_table(World),   %% 関数記号名テーブル作成
    present_world(World,[],true),        %% 対象世界の説明
    cai(World,Problems,1),               %% ITS メインルーチンへ
    its_end.                             %% 別の教材に進みますか？
```

its_start は失敗駆動ループを用いて，読み込まれた問題の出題を繰り返す．teaching_material(_,_) で教材を取りだし，set_visible_function_table(_) で画面表示に用いるテンプレートを作成し，present_world(_,_,true) で教材領

[†]D.Weld の論文 [2] より．

域を提示し，cai(_,_,_) で，この教材領域の問題群を教授する．そして全ての問題を出し尽くしたら its_end で先へ進むかを問い，進まないなら true ，つまり本 ITS を終了し，進むなら fail ，つまりバックトラックを起こして，新しく別の teaching_material(_,_) とのマッチングを行う．つまり次の教授領域の問題群へ移る．以上が最も大まかな流れである．次は ITS の本体である cai/3 について説明する．

cai/3 は，リストで渡された大問を小問ごとに分解するのであるが，その小問の問題形式†によって，別々の cai/3 がある．問題形式 1 の cai/3 なら問題を提示し，解答用対話型シェル discussion/2 へと移る．問題形式 2 ならば問題を提示した後，比較解析パッケージ ca_exe/4 を起動して証明木を求めた後，discussion/4 へと移る．次は discussion/2 /3 の説明をする．

discussion/2 /3 も失敗駆動ループを用いている．discussion/2 /4 の先頭の述語 ask/3 中の read/1 の repeat/0 まで戻って延々と繰り返すので，繰り返しループとも云う [3]．discussion/2 と discussion/4 との違いは，各問題が答えを得るのに必要とする情報が違うだけである．問題形式 1 では比較解析の結果である証明木は必要無い．discussion/2 /4 共に ask/3 で学習者から受けた指示に基づき分岐する，いわば switch case 文のような処理を行う．学習者からの指示が**問題への解答**であった場合は，それぞれ教授ルーチン teacher/3 /4 へと分岐する．question なら質疑応答ルーチン question/2 へ，world なら教授領域説明ルーチン present_world/2 へ，problem なら問題説明ルーチン present_problem/3 へと分岐し，fail でバックトラックすることで戻ってくる．問題に正解した場合と問題を飛ばす skip の時だけは true で終了する．次は教授ルーチン teacher/3 /4 について説明する．

teacher/3 は問題形式 1 の為の教授ルーチンである．引数として渡された解答と正解を比較し，正解なら true ，不正解なら解答の番号と一致するエントリーを problem(1,_,Entries,_) から求め，その qce(Target,InitRCs) より定性的反例ルーチン qualitative_counter_example/2 を起動する．引数で渡されたのが解答ではなく，help であった場合は定性的例題ルーチン qualitative_example/2 を起動する．

teacher/4 は問題形式 2 の為の教授ルーチンである．引数として渡された解答が正解ならば，証明木を説明する．不正解ならば部分問題移行ルーチン goto_subgoal/4 へ処理を移す．解答ではなく help であった場合は，比較解析時に生成した証明木を元に，ヒントを与えた後に goto_subgoal/4 へと処理を移す．

3.6 実行例とその評価

比較解析を応用した特徴をよく表わす，以下の機能についての例を示す．一連の実行例は付録 Bに添付してある．

『制約式の選択』

1. 定性的反例解時に，正しい答えの場合の反応と比較することで理解を促す．

†制約式の選択なら 1，変動の反応の予測なら 2 である．

2. ヒント[定性的例]ときに，正しい答えの場合の反応を教えることで直感的に正答へ『変動の反応の予測』

3. 単純な説明時に，答えを求めた過程を確認する助けになる．

4. ヒント答えを求めるための前提条件を提示し，問題にを解く際の着眼点を教授する

5. 部分問題解析作成した証明木をもとにする．

6. 質疑応答作者が用意したもの以外の変動を学習者が設定し，その反応を確認できる

1. 定性的反例

違います。 各関数を

抵抗ｒ１　　を ||
抵抗ｒ２　　を ↑

とした時、実際の系では 合成抵抗ｒ が ↑ になるはずですが、
あなたが選んだ １ では 合成抵抗ｒ が ↓ になってしまいます。

2. ヒント [定性的例]

ヒントです。
例えば 各関数を、

移動速度ｖ　　を ↑
磁束密度Ｂ　　を ||
長さｌ　　を ||

とした時、この系では 電位差ｅ が ↑ になるはずです。

同様に 各関数を、

移動速度ｖ　　を ||
磁束密度Ｂ　　を ||
長さｌ　　を ↑

とした時、この系では 電位差ｅ が ↑ になるはずです。

3. 単純な説明

正解です。
説明をしますか？(no) : yes.

抵抗ｒ は ↑ である（前述）
電圧ｖ は ‖ である（前述）
電流ｉ × 抵抗ｒ ＝ 電圧ｖ において、抵抗ｒ が ↑ かつ 電圧ｖ が ‖ なので、電流ｉ
は ↓ である。

4. ヒント

ヒントです。遠心力ｆ は 質量ｍ × 加速度ａ ＝ 遠心力ｆ より求まります。
部分問題に移りますか？ (no) :

5. 部分問題への移行

部分問題に移りますか？ (no) : yes.

速度ｖ の変化について考えてみましょう。

6. 質疑応答

データなし

Chapter 4

まとめ

定性推論の一手法として，**定性シミュレーション**と**比較解析**について勉強をした．定性シミュレーションについてはQSIMの一部を，比較解析についてはDQanalysisのサブセットを作成した．どちらにしても，定性的に対象とする系のパラメータや時間を捉えることの利点と弱点を実感した．少なくとも**定性推論**の研究はこれからの人工知能の研究になくてはならない研究になるだろう．ひいては人の頭脳活動の解明に役立つかも知れない．

今回作成した程度の比較解析でも，非常に興味深い効果が得られた．少ない記述で教材知識が記述でき，推論過程で作成された証明木が部分問題やヒントを出すのに有効であることも分かった．D.WeldのDQanalysisのように状態遷移をも比較できるような比較解析を用いれば，定性シミュレーターQSIMと組み合せることで，より幅広い問題領域に対して，より幅広い問題形式の教授が行える．

今回，定性推論はITS中で，対象とする物理系を解析することにのみ用いられたが，比較解析の特徴として2.2で述べた**診断**は，ITSでの**誤り原因同定**にも応用できるだろう．

Bibliography

[1] Kuipers,B.,Qualitative Simulation, *Artificial Intelligence 29* (1986) 289-338

[2] Weld,D.,Comparative Analysis, *Artificial Intelligence 36* (1988) 333-373

[3] Sterling,L.,Shapiro,E.,The Art of Prolog,MIT press,1986. (邦訳) 松田利夫, Prolog の技芸, 共立出版, 1988.

[4] 西田豊明, 川村正, 堂下修司:動的因果関係解析法による電子回路の定性的解析, 情報処理学会論文誌, Vol.28,No.2 , pp.177-188(1987)

[5] 西田:定性推論に関する最近の研究動向 (1) 基礎技術の進歩, 情報処理, Vol.29 , No.9 , pp.1009-1022(1988).

[6] 西田:定性推論に関する最近の研究動向 (2) 新しい研究分野・応用, 情報処理, Vol.29 , No.11 , pp.1322-1333(1988).

[7] 長尾真:知識と推論 (岩波講座ソフトウェア科学 14) , 岩波書店, 1988.

[8] 溝口文雄, 古川康一, 安西祐一郎,(監修) 淵一博:定性推論 (知識情報処理シリーズ別冊 1), 共立出版,1989

Appendix A

プログラムリスト

Appendix B

実行例