＜論　説＞

主体を含む動的複雑系の定式化

——動的複雑システムシミュレーションの基礎として——

高　井　徹　雄

はじめに

経営学など社会科学と呼ばれる諸分野が，科学であるか否かは議論の分かれるところである。確かに，科学としてのフォーマルな記述言語を持つか，再現性のある実験が設計できるかといった面で，社会科学の知識は自然科学と同等の公的知識とは言い難い面がある。しかし，複雑で理解しにくい対象を捉え，その構造や行動に関する法則性を解明して行くことが科学の本質であるとすれば，社会科学も科学を指向していることは間違いない。人間という不可解な主体を含み，かつ大規模なシステムを扱う社会科学の諸分野は，自然科学以上に複雑で理解しにくい対象を扱っているとえ言るかもしれない。

複雑で理解しにくい対象とは，その対象の行動を客観的に記述，予測することが構造的に難しいものを指す。それは，次のような性質の一部または全部を持つ対象である。

(1) 大規模で，要素と考えられるものの数が多い。また時間の経過に伴って，要素が新たに発生したり，消滅するような場合もある。

(2) 各要素は他の要素からの影響を受けながら，ある程度自律的に行動する。このような自律的システムの行動を記述，予測する言葉は，単純な状態遷移システムの場合と違って，他要素との関係，行動の目標などを加味したものになり，必然的に複雑になる。

(3) 要素間の関係にはいろいろな種類，様々なレベルのものがあり，また，

関係自体，時間経過に伴って動的に変化する場合もある。

(4) 階層各レベルの行動を記述する言葉は，その階層における創発特性を記述するために，下位レベルの行動を記述する言葉とは別に用意されなければならない。

数学的一般システム理論（以下 MGST と略記）では，入出力間の因果関係 $S\subset X\times Y$ として認識できる対象を，広く一般システムと呼ぶ。さらに，複数の要素システム $S_i\subset X_i\times Y_i$ $i=1,2,\cdots,n$ を含むシステム $S\subset\Pi X_i\times\Pi Y_i$ を複雑システムと呼ぶ。MGST では，複雑で，理解しにくいシステムの本質を「階層的認識を前提として理解されるべきシステム」と捉えている。経営学，社会学など社会科学の諸分野が扱う対象は，ほとんどがこの種の複雑システムと言えよう。MGST 複雑システム理論の存在意義は，意思決定主体を含む複雑な対象に関わる問題に対し，科学的アプローチを試みる際の指針を与えることにある。すなわち，上述のような複雑な対象に関しても，行動のフォーマルな記述，それに基づくモデル化，法則性に関する仮説の設定，シミュレーションによる仮説の検証，といった自然科学的扱いを可能にするためのガイドラインを示すことである。従来の自然科学は，複雑な問題に対処するために，分析主義的方法を用いてきた。つまり現実世界に存在する複雑で大きな問題を，実験室に運び込めるような単純で小さな問題に分解して扱う，分解による還元化の方法である。複雑システムの問題は，サブシステム間あるいは要素間の関係にこそ問題の本質が内在している場合が多く，分解による還元化の方法は必ずしも適当とは言えない。しかし，全体を一括して捉えきれない以上，やはり何等かの単純化は必要である。MGST では，複雑な対象をシステムの階層として捉え，階層ごとに適切な観点と表現を用意し，レベル間の関係に注目する，階層化による単純化を図る。すなわち，全体システム，サブシステム，全体とサブシステム間の関係，サブシステム間関係，サブシステムの要素，サブシステムと要素間の関係，……，要素間関係，というように問題を重層的に設定することで，各レベルの問題に適した言葉を用いることが可能となる。また相互関係についても，同一階層内，および異なる階層間という形で，整理して捉える

ことができる。

従来，MGST における複雑システムあるいは階層システムの定式化は，静的なものに限られており，動的表現のレベルでの定式化はほとんどなされていない。本小論では，上述の基本線に沿って，動的な階層システムの定式化を試みる。定式化の目標は，シミュレーションモデル作成のガイドラインを与えることである。

§1. 複雑システムの含まれる問題状況

本論で問題とする複雑系は，次のような特徴をもった時間システムである。

(1) 主体（特に意思を持った人間）及び，主体に制御される制御対象が，複数含まれている。

(2) 各主体の制御に関する意思決定は相互に連関しあっている。
つまり，複雑な意思決定の問題を含んでいる。

(3) 各主体の意思決定を調整するメカニズムが組み込まれている。
（但し，調整システムが実体として存在するとは限らない。）

このようなシステムは，主としては社会科学の諸分野が対象としてきた複雑システムである。この種のシステムは，行動の記述自体大変難しい。実際，社会科学では対象の行動に関する客観的，網羅的かつ動的な記述は初めから放棄され，分析者の主観により選択された，ある部分の断片的観察に基づく記述と評論が行われることの方が多い。こうした記述や評論は，必ず反論の余地が残り，決して公的な科学的知見とならない。社会科学も，科学性を追求するなら，少なくとも，自らの記述や評論の正当性を証明しうる実験装置を持つ必要があるのではないだろうか。我々は，システム論の観念に立ったシミュレーションこそが，社会科学における有力な実験装置としての可能性を秘めていると考える。

システム論の立場に立って，社会科学の対象を複雑な時間システムと見なせば，少なくとも概念上は，構成要素の行動の時系列をすべて列挙することで行動が記述できるはずである。つまり，

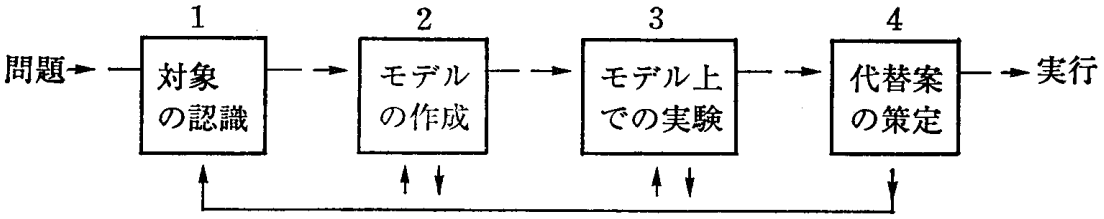
(a)　各 S_i=[制御対象 A_i+主体 B_i] の行動
(b)　調整システムCが存在するなら，その行動

の時系列としての記述である。

仮に，すべての可能性を列挙できたとすれば，行動の完全な外延的記述が得られたと言える。しかし，それは特殊なケースを除いて実現不可能である。また，実際に観察できる行動の時系列は，通常，無限にある可能性のなかから出現した，1つの結果に過ぎないから，その記述自体に予測力はない。さらに，それは，全体システムのなかで何故 S_i がそのように行動したかについては何も説明してくれない。ただし，問題の存在，あるいは，問題が改善されたといった認識は，この時系列の（一部分の）観察結果として現れる。

行動の時系列の観察に基づいて，問題の存在が認識されると，一般には，下図のような問題解決のプロセスが開始される。

問題解決のプロセス



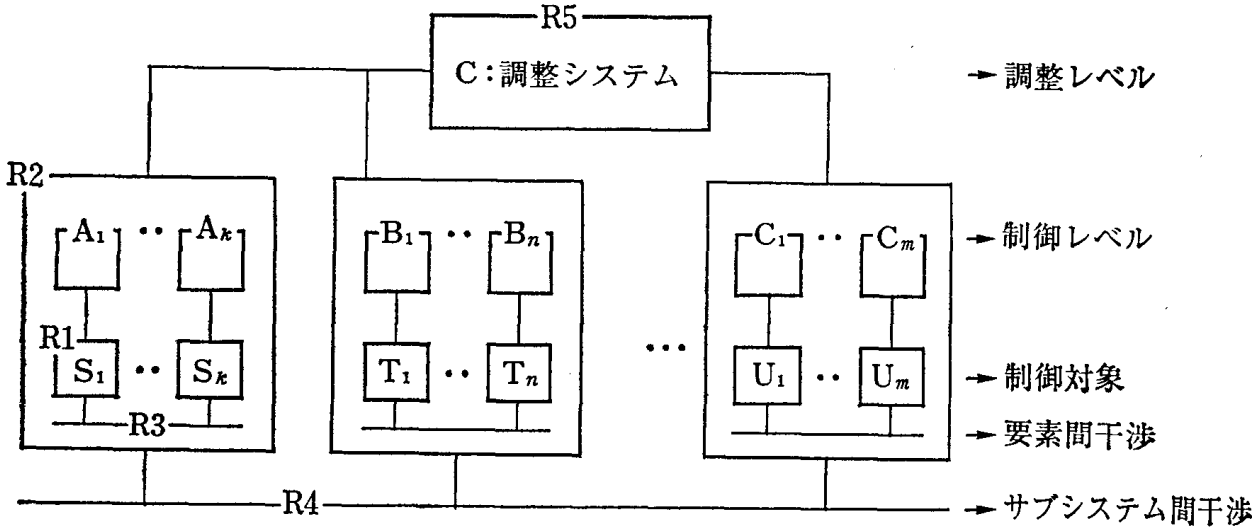
このなかで，予測力豊かなモデル=[システム行動に関する内包的表現]を創出できるか否かが，科学的アプローチに成功するキーとなる。具体的には，現実をよく近似し，予測力のあるシミュレーションモデルが設定できるかどうかということだ。以下では，階層システムの見方に沿って，動的複雑システムモデルの構築に関して考察を加えたい。

§2. 対象の構造とルールに関する階層的認識

我々が問題としている複雑システムは，初めから全体として捉えることが不可能な対象である。しかし，分析的思考に慣れている我々は，このような対象についても，その要素は何か？要素群のひとまとまりとしてのサブシステムは

何か？またそれらの行動原理は何か？といった質問には比較的容易に答えられる場合が多い。そこで，問題を単純化する手順としては，まず，システム境界の物理的包含関係においてグループ分け，あるいはレベル分けするのが一般的であろう。この視点に基づいて，複雑システムの構造を整理すると，例えば下図のような構造図が得られる（二階層システムモデル）。より複雑なシステムに対しては，3階層以上の多階層システムモデルも考え得るが，それらの多くは，二階層システムの重層したものと見ることができる。以下では概念図に示した階層システムモデルに沿って考察を進める。

二階層システムモデルの概念図



対象の構造が階層的に認識されると，各階層の行動を制約する条件ないしはルールが問題となる。予測力のある内包的モデルを造るには，主体の意思決定を制約するものは何か？を整理して捉えることがポイントとなる。これを階層的に類別すると，以下のようになる。

(1) R1. 基本制約条件：制御対象そのもののもつ制約条件

(2) R2. 場のルール　：主体の存在する「場」において適用されるルール

(3) R3. 要素間干渉のルール：要素システムと他の要素システムの状態との関係に関するルール

(4) R4. サブシステム間干渉のルール：要素システムと他の要素システムの状態との関係に関するルール

(5) R5.　調整のルール：調整機構の課す制約条件

(6) R6.　環境のルール：全体システムと外部環境との間の入出力関係

以上，システム内の主体の意思決定は，階層的ルール R1 & R2 & R3 & R4 & R5 に制約されながら行われる。これら系内のルールとは別に，外部環境との関わりに関するルール R6 が規定されることで，システム全体の挙動の内包的記述が可能となる。

さて，ルール R1～R6 はどのような言語で記述されるのだろうか。実際のモデルでは，モデル作成の目的に応じて，日常言語による記述で充分な場合もあるし，数式や計算機言語に記述が要求される場合もある。以下では，動的複雑系のフォーマルな定式化の基礎として，一階の述語論理で記述されるルールに限定して議論を進める。

§3. ルールの記述と記述言語

複雑システムでは，システムの行動を制約するルールが，複雑に絡み合っている。対象を階層化して捉えるひとつの意義は，階層別に類型化してルールを捉えられることにある。これにより，ルールの記述が簡素になる。あとは，階層間にまたがるルール間関係を規定してやればよい。

ルール設定において，まず，モデル上のルールは必ずしも現実のルールとは同じでないことに注意する必要がある。モデル上のルールは，問題解決の目標に応じて，モデルを造る者の視点が，システム行動のどの側面を本質と捉えるかによって変わってくる。設定されたルールが正当か否かは，分析者が考える「システム行動の本質」をどの程度近似できるか，つまり同様の状況下で現実と類似した結果がシミュレートできるかに依存して決まる。なお，モデルの近似性に関するテーマは興味深いが，Topological な議論を前提とし，議論の枠組みとして，より詳細な数学的構造が要求される。これは，本論の趣旨からは，はずれるので，これ以上触れない。ここでは，ルールの概念を記述するための言語Lと動的複雑系の表現枠として，一群のLの文を規定するに止める。

以下では，参考文献 [3] に準じて，まず，ある type をもった relational structure を定義し，次にこの type の well formed formulas を定義するという手順で，言語Lを規定して行く。

(i) 動的複雑系の性質を記述するための枠

定義 1：frame structure

以下の symbols を持つ relational structure

$\mathscr{A}=\langle \mathbf{A};\ \in,\ =,\ \mathbf{I},\ \mathbf{E},\ \mathbf{T},\ \mathbf{K},\ \mathbf{C},\ \mathbf{M},\ \mathbf{D},\ \varphi,\ \mu,\ \mathbf{R}_1,\ \mathbf{R}_2\cdots\rangle$を

ここでは frame structure と呼ぶことにする。

$\in$：binary relational symbol... 集合間の'属す'関係を表す

$=$：binary relational symbol... equality symbol

$\mathbf{I}$：unary relational symbol... index symbol

$\mathbf{E}$：binary relational symbol... elements index symbol

$\mathbf{T}$：unary relational symbol... time symbol

$\mathbf{K}$：binary relational symbol... time span symbol

$\mathbf{C}$：unary relational symbol... state symbol

$\mathbf{M}$：unary relational symbol... manipulation symbol

$\mathbf{D}$：unary relation symbol... manipulation change state symbol

φ：five-ary relational symbol... state transition symbol

μ：four-ary relational symbol... manipulation transition symbol

$\mathbf{R}_1,\ \mathbf{R}_2\cdots$ arbitrary relational symbols

ただし，$\mathbf{R}_k$ は m_k-ary relational symbol とする。

(ii) 言語L：

frame structure $\mathscr{A}$に対応して，次の記号を含む一階の述語論理を定義 2,3 によって定める。ここでは，これを言語Lと呼ぶ。

(1) 変数記号：$x_0,\ x_1,\ \cdots,\ x_n\cdots$ nは任意の自然数とする

(2) 述語記号：$\in,\ =,\ \mathbf{I},\ \mathbf{E},\ \mathbf{T},\ \mathbf{K},\ \mathbf{C},\ \mathbf{M},\ \mathbf{D},\ \varphi,\ \mu,\ \mathbf{R}_1,\ \mathbf{R}_2\cdots$

(3) 論理記号：$\wedge,\ \vee,\ \neg,\ \exists,\ \forall$

(4) カッコとコンマ：(,), ,

※ なお，見やすくするため，変数記号は，アルファベット $a, b, c, \cdots, x, y, z$ などで代用する場合がある。

定義 2：言語Lの atomic formulas

以下のものが，言語Lの atomic formula である。

(1) $x \in y$　(2) $x=y$　(3) $\mathbf{I}(x)$　(4) $\mathbf{E}(x,y)$　(5) $\mathbf{T}(x)$　(6) $\mathbf{K}(x,y)$　(7) $\mathbf{C}(x)$

(8) $\mathbf{M}(x, y)$　(9) $\varphi(x, y, u, v, w)$　(10) $\mu(x, y, z, w)$　(11) $\mathbf{D}(x)$

(12) $\mathbf{R}_k(y_1, y_2, \cdots, y_{mk})$

※ ただし，$x, y, y_1, y_2, \cdots$は，変数記号とする。

定義 3：言語Lの formulas

以下の6つのルールで規定されたもののみが，言語Lの formula である。

(1) 任意の atomic formula は formula である。

(2) ψ が formula であるなら，$\neg\psi$ も formula である。

(3) ψ_1 と ψ_2 がともに formula であるなら，$\psi_1 \wedge \psi_2$ も formula である。

(4) ψ_1 と ψ_2 がともに formula であるなら，$\psi_1 \vee \psi_2$ も formula である。

(5) ψ が formula であるなら，$(\exists x)\psi$ も formula である。

(6) ψ が formula であるなら，$(\forall x)\psi$ も formula である。

(iii) 時間集合とその上の順序に関する公理 τ：

(1) $(\forall x)(\forall y)(\mathbf{K}(x, y) \rightarrow \mathbf{T}(x) \wedge \mathbf{T}(y))$

(2) $(\forall x)(\mathbf{T}(x) \rightarrow \mathbf{K}(x, x))$

(3) $(\forall x)(\forall y)(\forall z)(\mathbf{K}(x, y) \wedge \mathbf{K}(y, z) \rightarrow \mathbf{K}(x, z))$

(4) $(\forall x)(\forall y)(\mathbf{K}(x, y) \wedge \mathbf{K}(y, x) \rightarrow x=y)$

(5) $(\exists x)(\mathbf{T}(x) \wedge (\forall y)(\mathbf{T}(y) \rightarrow \mathbf{K}(x, y))$

(6) $(\forall x)(\forall y)(\mathbf{T}(x) \wedge \mathbf{T}(y) \rightarrow \mathbf{K}(x, y) \vee \mathbf{K}(y, x))$

(iv) 複雑システムの基本構造に関する公理 α：

言語Lで記述される以下の文の集合を，複雑システムの基本構造に関する公理 α と呼ぶことにする。

(1) $(\forall x)(\forall y)(\mathbf{E}(x, y) \rightarrow \mathbf{I}(x))$

(2) $(\forall x)(\forall y)(\forall z)(\mathbf{E}(x, y)\wedge\mathbf{E}(x, z)\rightarrow y=z)$

(3) $(\forall x)(\forall y)(\mathbf{C}(x)\wedge y\in x\rightarrow(\exists z)(\exists w)(\mathbf{I}(z)\wedge y=(z, w)))$

(4) $(\forall x)(\forall y)(\forall z)(\mathbf{C}(x)\wedge y\in x\wedge\mathbf{I}(z)\rightarrow(\exists!w)(y=(z, w)))$

(5) $(\forall x)(\forall y)(\forall z)(\forall p)(\mathbf{I}(x)\wedge(\exists y)(\mathbf{C}(y)\wedge(x, z)\in y\wedge p\in z)$
$\rightarrow(\exists q)(\exists r)(\exists s)(\mathbf{E}(x, q)\wedge r\in q\wedge p=(r, s)))$

(6) $(\forall x)(\forall z)(\forall p)(\forall q)(\forall r)$
$((\exists y)(\mathbf{C}(y)\wedge(x, z)\in y\wedge p\in z)\wedge\mathbf{E}(x, q)\wedge r\in q\rightarrow(\exists!s)((r, s)\in p))$

(7) $(\forall x)(\forall y)(\mathbf{M}(x)\wedge y\in x\rightarrow(\exists z)(\exists w)(\mathbf{I}(z)\wedge y=(z, w)))$

(8) $(\forall x)(\forall y)(\forall z)(\mathbf{M}(x)\wedge y\in x\wedge\mathbf{I}(z)\rightarrow(\exists!w)(y=(z, w)))$

(9) $(\forall x)(\forall y)(\forall u)(\forall v)$
$(\mathbf{M}(x)\wedge y\in x\wedge\mathbf{I}(u)\wedge(u, v)=y\rightarrow(\exists t)(\exists z)(\mathbf{T}(t)\wedge(t, z)=v))$

(10) $(\forall x)(\forall y)(\forall u)(\forall v)(\forall t)$
$(\mathbf{M}(x)\wedge y\in x\wedge\mathbf{I}(u)\wedge(u, v)=y\wedge\mathbf{T}(t)\rightarrow(\exists z!)((t, z)=v))$

(11) $(\forall x)(\forall y)(\forall p)(\forall q)(\forall w)$
$(\varphi(x, y, p, q, w)\rightarrow\mathbf{C}(x)\wedge\mathbf{M}(y)\wedge\mathbf{K}(p, q)\wedge\mathbf{C}(u))$

(12) $(\forall x)(\mathrm{A}y)(\forall p)(\forall q)$
$(\mathbf{C}(x)\wedge\mathbf{M}(y)\wedge\mathbf{K}(p, q)\rightarrow(\exists!w)(\varphi(x, y, p, q, w)))$

(13) $(\forall x)(\forall y)(\forall z)(\forall w)(\mu(x, y, z, w)\rightarrow\mathbf{M}(x)\wedge\mathbf{C}(y)\wedge\mathbf{T}(z)\wedge\mathbf{C}(w))$

(14) $(\forall x)(\forall y)(\forall z)(\mathbf{M}(x)\wedge\mathbf{C}(y)\wedge\mathbf{T}(z)\rightarrow(\exists!w)(\mu(x, y, z, w)))$

(15) $(\forall x)(\forall y)(\forall z)(\forall w)(\mu(x, y, z, w)\rightarrow$
$(\forall t)(\forall p)(\forall q)(\mathbf{K}(t, z)\wedge\neg t=z\wedge(t, p)\in x\wedge(t, q)\in w\rightarrow p=q)$

(16) $(\forall y)(\mathbf{D}(y)\leftrightarrow(\exists x)(\exists z)(\exists w)(\mu(x, y, z, w)\wedge\neg x=w))$

(17) $(\forall z)(\forall u)(\forall v)(\forall r)(\forall s)(\forall p)(\forall q)(\forall a)(\forall b)(\forall w)(\forall h)(\forall x)(\forall y)$
$(\mathbf{I}(z)\wedge(z, w)\in u\wedge(z, w)\in v\wedge(z, h)\in r\wedge(z, h)\in s\wedge$
$\varphi(u, r, p, q, a)\wedge\varphi(v, s, p, q, b)\wedge(z, x)\in a\wedge(z, y)\in b\rightarrow x=y)$

(18) $(\forall p)(\forall u)(\forall v)(\forall r)(\varphi(u, r, p, q, v)\rightarrow u=v)$

(19) $(\forall z)(\forall u)(\forall r)(\forall s)(\forall p)(\forall a)(\forall b)(\forall h)(\forall x)(\forall y)$
$(\mathbf{I}(z)\wedge(z, h)\in r\wedge(z, h)\in s\wedge\mu(r, u, p, a)\wedge\mu(s, u, p, b)\wedge$

$(z, x)\in a\wedge(z, y)\in b\rightarrow x=y)$

⑳ $(\forall p)(\forall q)(\forall r)(\forall x)(\forall y)(\forall z)(\forall w)(\forall u)$

$(\varphi(x, u, p, r, z)\wedge\varphi(x, u, p, q, y)\wedge\varphi(y, u, q, r, w)\rightarrow z=w)$

※ $(\exists!x)\varphi(x)$ なる文は $(\exists x)\varphi(x)\wedge(\forall y)(\forall z)(\varphi(y)\wedge\varphi(z)\rightarrow y=z)$ の簡略形。

(v) model $\mathscr{A}$ の性質：

構造 $\mathscr{A}=\langle A;\ \in,\ =,\ I,\ E,\ T,\ K,\ C,\ M,\ D,\ \varphi,\ \mu,\ R_1,\ R_2\cdots\rangle$ が，公理 τ および α のモデル $\mathscr{A}\models\alpha\cup\tau$ であるとき，$\mathscr{A}$ を動的複雑システムの実現と呼ぶ。以下では，公理 α が規定する基本的要件を整理しておく。

まず，$I(i)$ なる各 $i\in A$ に対し，集合 C_i, M_i を

$C_i=\{a\in A\mid(\exists c)(C(c)\wedge(i, a)\in c\}$

$M_i\triangleq\{a\in A\mid(\exists m)(M(m)\wedge(i, a)\in m\}$ と定義する。

このとき，次が成り立つ。

補題 1：$\mathscr{A}\models\alpha$ なる $\mathscr{A}$ においては，次が成り立つ。

(a) $C\subset\Pi(C_i\mid i\in I)$　　(b) $M\subset\Pi(M_i\mid i\in I)$

[証明] 補遺 1 参照。 □

補題 1 より，$\mathscr{A}\models\alpha$ なる $\mathscr{A}$ においては，$C(c)$ すなわち，$c\in C$ なる c 及び，$M(m)$ すなわち，$m\in M$ なる m は，それぞれ，直積集合 $\Pi(C_i\mid i\in I)$，$\Pi(M_i\mid i\in I)$ の要素であり，それぞれ，関数 $c: I\rightarrow\cup(C_i\mid i\in I)$，$m: I\rightarrow\cup(M_i\mid i\in I)$ とみなせる。よって，以下では，$I(i)$, $C(c)$, $M(m)$ なる i, c, m に対しては，$(i, a)\in c$ なる a を $c(i)$，$(i, b)\in m$ なる b を $m(i)$ と書くことにする。

$\alpha-(1)$, (2) より，E は，関数 $E: I\rightarrow A$ とみなせる。そこで，各 $i\in I$ に対して，$J_i\triangleq E(i)$ と書き，各 $j\in J_i$ に対し，集合 B_{ij} を $B_{ij}\triangleq\{b\in A\mid(\exists c)(C(c)\wedge(j, b)\in c(i)\}$ とする。このとき，

補題 2：$\mathscr{A}\models\alpha$ なる $\mathscr{A}$ においては，$I(i)$ なる各 i に対して，

$C_i\subset\Pi(B_{ij}\mid j\in J_i)$ が成り立つ。

[証明] 補遺 2 参照。

次に，i 要素の操作集合 M_i の構造を特徴づけておこう。

$P_i\triangleq\{a\in A\mid(\exists m)(\exists t)(M(m)\wedge T(t)\wedge m(i)=(t, a))\}$ とおくと，次が成り立つ。

補題 3 ：$\mathscr{A}\models\alpha$ なる$\mathscr{A}$において，$I(i)$ なる各 i に対して，

$M_i\subset P_i{}^T=\{x: T\to P_i \mid x$ は mapping$\}$ が成り立つ。

[証明]　補遺3参照。　□

これは，要素の操作が時間関数で表されることを示している。次に φ, μ の基本的性質を整理しておこう。

補題 4 ：$\mathscr{A}\models\alpha$ なる$\mathscr{A}$において，2つの4項関係 φ, μ は，それぞれ，関数

$\varphi: C\times M\times K\to C$, $\mu: M\times C\times T\to M$ とみなせる。

[証明]　補遺4参照。　□

φ を状態遷移関数，μ を操作変更関数と呼ぶことにする。

補題 5 ：$\mathscr{A}\models\alpha\cup\tau$ なる$\mathscr{A}$において，補題4の関数 $\mu: M\times C\times T\to M$ は，

次の性質をもつ。

(i)　任意の $m\in M$, $c\in C$, $t\in T$ に対し，$c\notin D$ ならば，$\mu(m, c, t)=m$

(ii)　任意の $i\in I$, $m, m'\in M$, $c\in C$, $t\in T$ $s.t.$ $\mu(m, c, t)=m'$

に対し，$(\forall t')(t'<t\to m(i)(t')=m'(i)(t'))$ が成り立つ。

[証明]　補遺5参照。　□

(i)は，複雑システムの状態 c が D に入らない限り，各要素の操作は変更されないことを示している。(ii)は，操作が変更される場合でも，過去の操作の履歴は変更されないことを示している。

複雑システムにおける要素は，他の要素からの影響を受けながらも，自律的に行動する存在である。次の補題6,7は，公理 α のモデルにおいては，状態遷移が要素ごとに完全に分離できること，操作変更は，他要素の状態に影響されながら行われることを示唆している。

補題 6 ：$\mathscr{A}\models\alpha\cup\tau$ なる$\mathscr{A}$において，補題4の関数 $\varphi: C\times M\times K\to C$ は，

次の意味で分離可能である。

(i)　各 $i\in I$ に対して，

$\varphi_i\triangleq\{(c(i), m(i), k, \varphi(c, m, k)(i)) \mid C(c)\wedge M(m)\}$

とおくと，φ_i は関数 $C_i\times M_i\times K\to C_i$ とみなせる。

(ii)　任意の $c\in C$, $m\in M$, $k\in K$ に対し，

$\varphi(c, m, k)=(\varphi_i(c(i), m(i), k)\,|\,i\in I)$ が成り立つ。

[証明]　補遺6参照。　□

補題7：$\mathscr{A}\models\alpha\cup\tau$ なる$\mathscr{A}$において，補題4の関数 $\mu: C\times M\times T\to C$ は，次の意味で分離可能である。

各 $i\in I$ に対して，

$\mu_i\triangleq\{m(i), c, t, \mu(m, c, t)(i))\,|\,M(m)\wedge C(c)\}$

とおくと，μ_i は関数 $M_i\times C\times T\to M_i$ とみなせる。

[証明]　補遺7参照。　□

補題8：$\mathscr{A}\models\alpha\cup\tau$ なる$\mathscr{A}$において，補題4の関数 $\varphi: C\times M\times K\to C$ 及び，補題6の $\varphi_i: C_i\times M_i\times K\to C_i$ は，

(1)　任意の $t, t', t''\in T$ $s.t.$ $t<t'<t''$，$c\in C$，$v\in M$ に対し，

$\varphi(c, v, t, t'')=\varphi(\varphi(c, v, t, t'), v, t', t'')$

(2)　さらに，任意の $i\in I$ に対し，

$\varphi_i(c(i), v(i), t, t'')=\varphi_i(\varphi_i(c(i), v(i), t, t'), v(i), t', t'')$

[証明]　補遺8参照。　□

(1)，(2)は状態遷移関数に求められる半群性の条件と呼ばれる。

なお，公理 α には含めなかったが，次に示される過去依存性条件も通常のシステムでは仮定される。

$(\forall u)(\forall v)(\forall c)(\forall t)(\forall t')$

$(c\in C\ \&\ u, v\in M\ \&\ t<t'\ \&\ u|_t^{t'}=v|_t^{t'}\to\varphi(c, u, t, t')=\varphi(c, v, t, t'))$

以上，$\mathscr{A}\models\alpha\cup\tau$ なる$\mathscr{A}$において得られる集合および関数を次のように呼ぶことにする。

(1)　時間集合：T

(2)　要素インデックス集合：I

(3)　時間順序関係：$K\subset T\times T$

(4)　全体状態：$C\subset\Pi(C_i\,|\,i\in I)$

(5)　要素状態：$C_i\subset\Pi(B_{ij}\,|\,j\in E(i))$ for $i\in I$

(6)　全体操作変数：$M\subset\Pi(M_i\,|\,i\in I)$

(7) 要素操作変数：$M_i \subset P_i^T$

(8) 状態遷移関数：$\varphi : C \times M \times K \to C$

(9) 要素状態遷移関数：$\varphi_i : C_i \times M_i \times K \to C_i$ for $i \in I$

(10) 操作変更関数：$\mu : M \times C \times T \to M$

(11) 要素操作変更関数：$\mu_i : M_i \times C \times T \to M_i$ for $i \in I$

(12) 操作変更状態：$D \subset C$

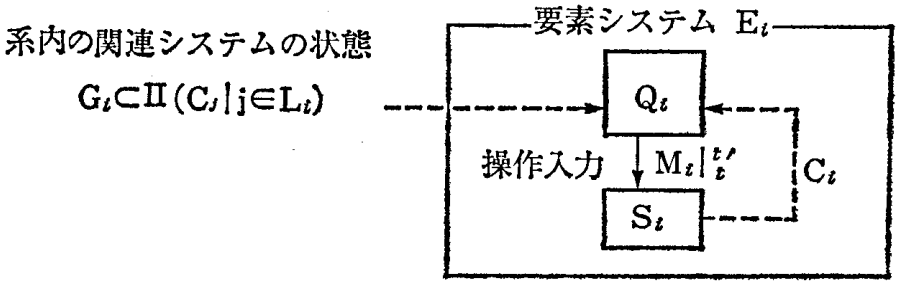
§4. 動的複雑システムの定式化

複雑システムの問題解決には，様々なモデル化の方法論が用いられる。このなかで，操作性のあるものに限定すれば，大別して，数理モデル，コンピュータシミュレーションモデル，あるいはこの混合型になる。さらに，対象が複雑になればなるほど，表現の自由度が大きく，システム行動の本質を近似しやすいシミュレーションモデルが有効と思われる。ただし，シミュレーションモデルが問題解決において真に有効なモデルになるかどうかは，モデル作成者のセンスや経験に大きく依存していることは言うまでもない。それは，未熟なモデル作成者にとって，確かなガイドラインとなる一般理論が不足していることも一因であろう。§3のモデル論的表現枠の設定は，こうした状況を改善しようとするひとつの試みである。以下では，シミュレーションモデル作成のガイドラインとして，公理αの意味付けを行なう。

要素システム E_i を次のような，制御システム Q_i と被制御システム S_i の結合したシステムと捉える。

次に動的複雑系を，動的に干渉しあって行動する動的な要素システムの集合

要素システム概念図



体と捉える。各 S_i は，Q_i からの操作入力を受けて，状態遷移する時間システム。Q_i は，S_i の状態のみでなく，関連する他のシステムの状態に関する情報を見ながら，制御対象への操作入力を変更して行くシステムとする。

まず，§3の定式化枠に準じながら，一般的な離散系シミュレーションモデル作成のガイドとして，動的複雑系のモデルを定式化しておこう。

[定式化1]

(1) 被制御システム S_i の状態遷移関係：$\varphi_i : C_i \times M_i \times K \to C_i$

(2) 制御システム Q_i の操作遷移関数：$\mu_i : M_i \times C \times T \to M_i$

(3) 操作変更規則：$c \in D_i \to \mu_i(m, c, t) = m|^t \cdot \eta_i(m, c, t)$

但し
$\eta_i : M_i \times C \times T \to M_i|_t$
C_i ：第 i サブシステムの状態集合
M_i：第 i サブシステムの操作の軌跡を表す時間関数集合
$K = \{(t, t') | t, t' \in T \ \& \ t \leqq t'\}$

ここで注意すべきは，操作入力は，μ_iによって，遠い未来まで決定されている訳ではないという点である。μ_i は，あくまで暫定的な操作入力を規定しているだけであって，ΠC_j の状態が各要素システムの状態遷移関数によって変化し，ある組み合わせの状態に達したとき，操作入力の再計算をする。つまり，次のような状態操作適合性は，一般にはある短い時間間隔の間でしか成り立たない。

定義4：状態—操作適合性

complex system $(\varphi_i, \mu_i)_{i=1, \cdots, n}$ における状態 $c = (c_i | i \in I)$ と操作 $m = (m_i | i \in I)$ が時間区間 $[t, t']$ で状態—操作適合性を満たすとは，

任意の $i \in I$，任意の $s \in [t, t']$ において，

$\mu_i(m, \varphi(c, m, (t, s)), s) = \mu_i(m, c, t) \cdots$(a)　□

(a) 式は，制御システムによる，時刻 t における操作入力の決定が，時刻 τ でも生きていることを示す。公理 α を満たす複雑系では，状態—操作適合性の十分条件は，次のものである。

補題9：$\mathscr{A} \models \alpha \cup \tau$ なる $\mathscr{A}$ において，次の条件(b)は，状態 $c = (c_i | i \in I)$ と操作 $m = (m_i | i \in I)$ が時間区間 $[t, t']$ で状態—操作適合性を満たすための十

分条件である。

$$(\forall s)(t\leqq s\leqq t'\rightarrow\varphi(c, m, t, s)\notin D)\cdots(\mathrm{b})$$

[証明]　補遺9参照。　□

条件 (b) が不成立のとき，すなわち，

$$(\exists s)(t\leqq s\leqq t' \ \&\ \varphi(c, m, t, s)\in D)\cdots(\mathrm{c})$$

となるときのみ，状態—操作適合性が不成立なる可能性がある。これは，いずれかの要素システムで操作の変更が行われたことを意味する。

例えば，交通システムという動的複雑系において，道路を走っている車1台を要素システム E_i と捉えるとき，制御システム Q_i はドライバー，被制御システム S_i は機械としての自動車ということになる。ドライバー Q_i は，各時刻 $t\in T$ において，例えば5秒後にブレーキをかけようとか，2秒後にアクセルを踏もうといった，t 以降の操作入力の予定 $m\in M|_t$ をもっているが，関連システム，すなわち近隣の車や信号の状態 $c\in G_i$ が，ある領域 G_i に入ったとき，操作予定 m を変更する。時刻 t に領域 G_i に入ったとき，どのように操作を変更するかは，ドライバーの判断に基づく変更規則 $G_i\times M_i\rightarrow M_i|_t$ があって，ルール化されている。このような動的複雑システムの定式化に際しては，要素システムに関連するシステムは何か，という点が加味される。

ここで，操作変更の判断を決する集合 D の性質について考察しておこう。まず，$\alpha-(18)$ より，

$$D=\{c\in C|(\exists t)(\exists m)(t\in T \ \&\ m\in M \ \&\ \mu(m, c, t)\neq m)\}$$

である。各 $i\in I$ について，

$$D_i\triangleq\{c\in C|(\exists t)(\exists m)(t\in T \ \&\ m\in M_i \ \&\ \mu_i(m, c, t)\neq m)\}\subset C$$

とおけば，$D=\cup(D_i|i\in I)$ となり，$c\in D_i$ ならば，要素 i の操作変更が要請される。

[定式化1] に基づくシミュレーション構成は，およそ次のようになる。

(1)　初期条件の設定：最初の時刻 $t\in T$，終了時刻 $t_{\max}$，初期状態 $c\in C$，初期操作 $m\in M$ を定める。

(2)　$t<t_{\max}$ の限り，以下を繰り返す。

［1］　各 $i\in I$ について，予定される操作 $m(i)\in M_i$ を維持したと仮定して，次に $\varphi(c, m, t, t_i)\in D_i\cdots①$　となる時刻 t_i を計算する。

［2］　最も早く $\varphi(c,m,t,t_k)\in D_k$ となる要素 k とその時刻 $t':=t_k$ を求める。

［3］　μ_k を適用して，操作予定 $m(k)$ を変更し，新たな m を得る。

［4］　状態を遷移させ，$c:=\varphi(c, m, t, t')$。時刻を進め，$t:=t'$ とする。

ところで，D_i の要素であるか否かは，$D_i\subset D\subset C\subset\Pi(C_i|i\in I)$ であるから，一般には I のすべての要素の状態を観て判定されることになる。しかし，実際のシステムでは，要素システム i と直接関連のある $L\subset I$ の要素の状態を観れば判定が可能であるケースも多い。これは，次の意味で，D_i が D から分離可能の場合である。

定義５：ある $L_i\subset I$ があって，

(1)　$D_i\cong G_i\times\Pi(C_j|j\in I-L_i)$

(2)　$(\forall i)(\forall t)(\forall c)(\forall c')(\forall m)(i\in I\ \&\ t\in T\ \&\ c\in C\ \&\ c'\in C\ \&\ m\in M_i\ \&\ (\forall j)(j\in L_i\rightarrow c(j)=c'(j))\rightarrow\mu_i(m, c, t)=\mu_i(m, c', t))$

となるとき，D_i は，L_i によって，分離可能と呼ぶ。

ただし，$G_i\triangleq\{(d(j)|j\in L_i)|d\in D\}$ とする。　□

L_i を要素 i の関連システムの添字集合，G_i を要素 i の操作状態と呼ぶことにする。(2)より，D_i が L_i によって分離可能な場合，$\mu_i: M_i\times C\times T\rightarrow M_i$ の代わりに，μ_i から自然に定義される $\hat{\mu}_i: M_i\times H_i\times T\rightarrow M_i$ を用いることができる。ただし，$H_i\triangleq\Pi(C_j|j\in L_i)$ とする。

D_i の分離可能性を前提にした定式化は次のようになる。

［**定式化２**］：各 D_i が分離可能な場合

(1)　被制御システム E_i の状態遷移関係：$\varphi_i: C_i\times M_i\times K\rightarrow C$

(2)　制御システム Q_i の操作遷移関数：$\hat{\mu}_i: M_i\times H_i\times T\rightarrow M_i$

(3)　操作変更規則：

(a)　変更状態：$G_i\subset H_i\triangleq\Pi(C_j|j\in L_i)$

L_i は，サブシステム i と関係を持つサブシステムズの添字集合を表す。

(b)　変更関数：$c\in G_i\rightarrow\hat{\mu}_i(m,\ c,\ t)=m|^t\cdot\eta_i(m,\ c,\ t)$

$c\notin G_i\rightarrow\hat{\mu}_i(m,\ c,\ t)=m$

但し，$\eta_i: M_i\times G_i\times T\rightarrow M_i|_t$

この場合，シミュレーションモデルの構成は，およそ次のようになる。

(1)　条件設定：最初の時刻 $t\in T$，終了時刻 t_{max}，初期状態 $c\in C$，

初期操作予定 $m\in M$，各要素の操作変更時刻 t_i for $i\in I$

初期操作要素 $k\in I$ $s.t.$ $t_k=\min\{t_i|i\in I\}$，操作時刻 $t':=t_k$

(2)　$t<t_{max}$ の限り，以下を繰り返す。

[1]　$c:=\varphi(c,m,t,t')$：全体の状態を遷移させ，$t:=t'$：時刻を進める。

[2]　$m(k):=\hat{\mu}_k(m(k),\ (c(j)|j\in L_k),\ t)$：$k$ 要素のみ操作の再計算。

[3]　$k\in L_i$ なる各 $i\in I$ のみについて，

次に　$(\varphi_i(c(j),\ m(j),\ t,\ t_i)|j\in L_i)\in G_i$…②

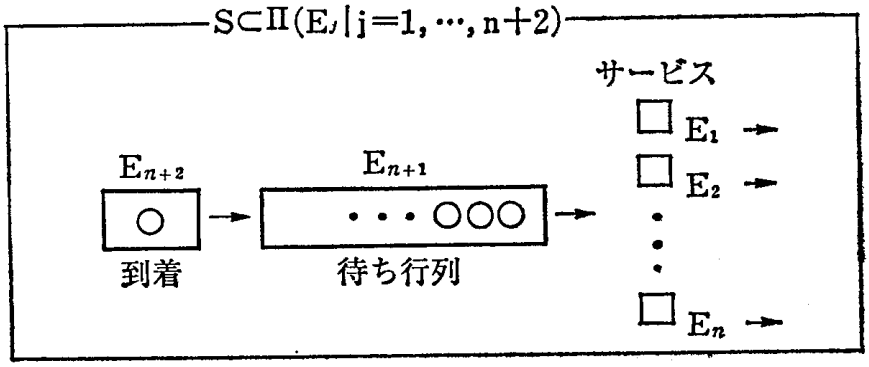
となる時刻 t_i を再計算する。なお，②を満たす t_i が存在しないときは，$t_i:=t_{max}$ とする。

[4]　$k\in I$，$t':=t_k$ $s.t.$ $t_k=\min\{t_i|i\in I\}$ を求める。

§5.　定式化枠による記述例

本節では，前節までに述べた定式化枠と，シミュレーションモデルとの関係を示すため，2つの動的複雑システムの例を挙げながら，定式化枠による記述の例を示す。まず，離散型シミュレーションの最も典型的な例題として，複数窓口の待ち行列システムに関する記述例を示しておく。

例1：複数並列窓口の待ち行列



図のような要素からなる複雑システムを考える。

全体システムの構成：$\&=\{E_1, \cdots, E_n, E_{n+1}, E_{n+2}\}$

for $1\leqq i\leqq n$

(1) 状態集合：$C_i=\{0, 1\}$

(2) 操作変数：$M_i=\{m : T\to\{-1, 0, 1\} \mid |m^{-1}(1)| \text{ と } |m^{-1}(-1)| \text{は有限}\}$

(3) 関連システム：

添字集合：$L_i=\{1, \cdots, n, n+1\}$

状態空間：$H_i=\Pi(C_j \mid j\in L_i)$

(4) 状態遷移：$\varphi_i : C_i\times M_i|_t^{t'}\times K\to C_i$

$$;(c, m, (t, t'))\mapsto c+|m^{-1}(1)|-|m^{-1}(-1)|$$

(5) 操作遷移：$\mu_i : M_i\times H_i\times T\to M_i$

$$;(m, c, t)\mapsto\begin{cases} m|^t\cdot\eta_i(m, c, t) & \text{if } c\in G_i \\ m & \text{otherwize}\end{cases}$$

(6) 操作生成規則：(G_i, η_i)

(a) 操作状態：$G_i=\{c\in H_i \mid c(i)=0 \ \&\ (\forall j<i)(c(j)=1) \ \&\ c(n+1)>0\}$

(b) 操作関数：$\eta_i : M_i\times T\to M_i|_t;\ (m, t)\mapsto\eta_i(m, t) : T|_t\to\{-1, 0, 1\}$

$$\eta_i(m, t)(\tau)=\begin{cases} 1 & \text{if } \tau=t \\ -1 & \text{if } \tau=t-\ln(\text{rand})/\text{サービス率} \\ m(\tau) & \text{otherwize}\end{cases}$$

$i=n+1$

(1) 状態集合：$C_{n+1}=\{0, 1, 2, \cdots\}$

(2) 操作変数：$M_{n+1}=\{m : T\to Z \mid (\forall z)(z\in Z \ \&\ z\neq 0\to |m^{-1}(z)| \text{ は有限})\}$

但し，Zは整数集合。

(3) 関連システム：

添字集合：$L_{n+1}=\{1, \cdots, n+2\}$

状態空間：$H_{n+1}=\Pi(C_j \mid j\in L_{n+1})$

(4) 状態遷移：$\varphi_{n+1} : C_{n+1}\times M_{n+1}\times K\to C_{n+1}$

$$;(c, m, (t, t'))\mapsto c+\sum_{z\in Z} z|m^{-1}(z)|$$

(5) 操作遷移：$\mu_{n+1}: M_{n+1}\times H_{n+1}\times T\to M_{n+1}$

$$;(m,\ c,\ t)\mapsto\begin{cases} m|^{t}\cdot\eta_{n+1}(m,\ c,\ t) & \text{if } c\in G_{n+1}\\ m & \text{otherwize}\end{cases}$$

(6) 操作規則：$G(_{n+1},\eta_{n+1})$

(a) 操作状態：$G_{n+1}=G_A\cup G_B$

$$G_A=\{c\in H_{n+1}|c(n+1)>0\ \&\ (\exists j)\,(1\leqq j\leqq n\ \&\ c(j)=0)\}$$

$$G_B=\{c\in H_{n+1}|c(n+2)=1\}$$

(b) 操作関数：$\eta_{n+1}: M_{n+1}\times H_{n+1}\times T\to M_{n+1}|_t$

$$;(m,\ c,\ t)\mapsto\eta_{n+1}(m,\ c,\ t): T|_t\to Z$$

$$\eta_{n+1}(m,\ c,\ t)\,(\tau)=\begin{cases} m(\tau)-1 & \text{if } \tau=t\ \&\ c\in G_A-G_B\\ m(\tau)+1 & \text{if } \tau=t\ \&\ c\in G_B-G_A\\ m(\tau) & \text{otherwize}\end{cases}$$

$\underline{i=n+2}$

(1) 状態集合：$C_{n+2}=\{0,\ 1\}$

(2) 操作変数：$M_{n+2}=\{m: T\to\{0,\ 1\}\,|\,m^{-1}(1)\ \text{は有限}\}$

(3) 関連システム：

添字集合：$L_{n+2}=\{n+2\}$

状態空間：$H_{u+2}=C_{u+2}$

(4) 状態推移：$\varphi_{u+2}: C_{n+2}\times M_{n+2}\times K\to C_{n+2};\ (c,\ m,\ (t,\ t'))\mapsto m(t')$

(5) 操作推移：$\mu_{n+2}: M_{n+2}\times C_{n+2}\times T\to M_{n+2}$

$$;(m,\ c,\ t)\mapsto\begin{cases} m|^{t}\cdot\eta_{n+2}[t] & \text{if } c\in G_{n+2}\\ m & \text{otherwize}\end{cases}$$

(6) 操作生成規則：$(G_{n+2},\ \eta_{n+2})$

(a) 操作状態：$G_{n+2}=\{1\}\subset C_{n+2}$

(b) 操作関数：$\eta_{n+2}: M_{n+2}\times C_{n+2}\times T\to M_{n+2}|_t$

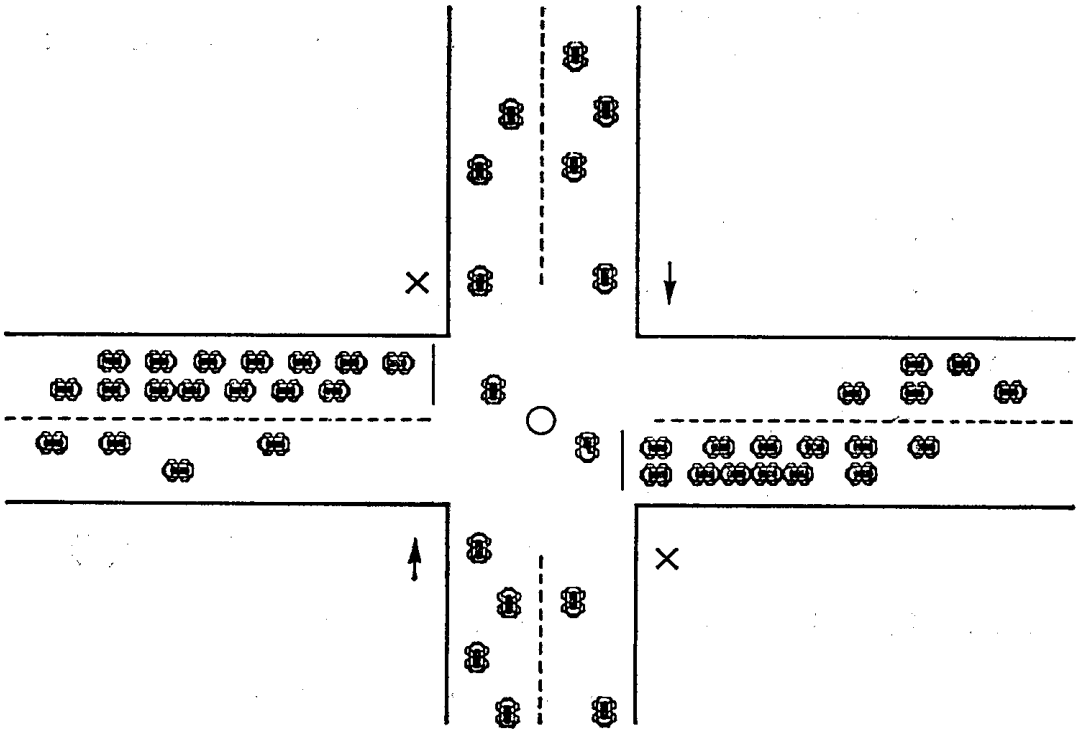
$$;(m,\ c,\ t)\mapsto\eta_{n+2}(m,\ c,\ t): T|_t\to\{0,\ 1\}$$

$$\eta_{n+2}(m,\ c,\ t)=\begin{cases} 1 & \text{if } \tau=t-\ln(\text{rand})/\text{到着率}\\ m(\tau) & \text{otherwize}\end{cases}$$

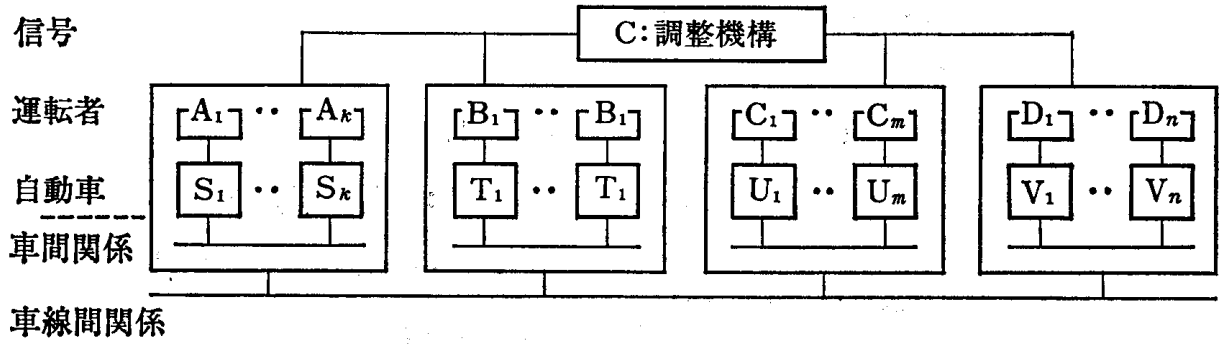
複数窓口の待行列シミュレーションは，離散型シミュレーションの典型的な例題として，様々な教科書のなかで紹介されている。この程度の複雑性象ならば，敢えて上述のような定式化を行わなくても，少し慣れた分析者ならば，容易にプログラム化が可能である。しかし，より複雑な対象を扱う場合は，プログラム化の前段階として，システムをどう観るか，またその見方を表現する厳密な定式化を行っておく方が，結果として効率が良い。次に，より複雑な動的システムの例として，交差点を含む道路交通システムの例を取り挙げる。

例2：交差点交通システム

次図のような，1交差点を含む道路交通システムを対象として，我々の定式化枠を適用してみよう。



まず，交差点システムの物理的構造を階層図として表現してみる。



ここで，物理的な階層に双対な，ルール階層が存在する。また，各階層のルールの中に，調整メカニズムが内在する。以下では，各階層のルールを見て行く。

(i) 要素システムに関するルール

R0. 制御対象＝車のもつ制約：

(1) 力学の法則：$v=at$; $x=vt$; $x=at^2/2$	…定加速度運動の場合
(2) 自動車の性能：$\min \leqq a \leqq \max$	…加速度には上下限がある

R1. 場＝道路の全域における制約条件：

(3) 制限速度　$v \leqq$ 制限速度	…制限速度に達したら定速走行に入る
(4) 右左折の場合，安全な速度に減速ないし停止	…交差点手前で適切に減速，停止

※ R0, R1は，［車＋運転車］という要素単体の行動を制約する。

(ii) 要素システム間に存在するルール

R2. 干渉＝車間関係のルール：

(5) 後車は，追突しないように走行する：前車が減速したら，適当なタイムラグの後，適当な減速をする。

(6) 後車は，無駄に車間を空けない：前車が加速したら，適当なタイムラグの後適当な加速を行う。

(7) 右折時，右折中に対向車がこないこと：対向車が来る限り，右折車は発進してはならない。

※ R2には，要素システム間の調整メカニズムが組み込まれている。
このレベルの調整により，全体としての秩序ある動態が生まれる。

(iii) 要素システム間を調整するルール

R3. 調整機構＝信号機による制約：

(8) 黄：交差点前にあって，停止可能な車は停車　…信号停止
先頭の右折待車は発進。　…信号発進

(9) 赤：右折発進を禁止する。　…信号停止

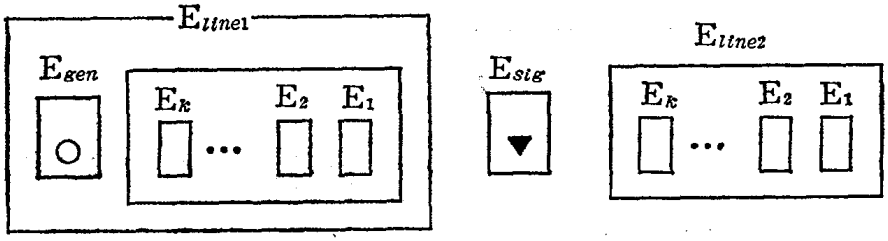
(10)　青：直進，左折の信号待車を発進。　…信号発進

右折車は，対向車なければ発進

※ R3は，車線間の関係を調整するメカニズムである。

※ さらに，車線間の結合．外部環境からの入力（到着），外部環境への出力（消滅）に関するルールの規定が必要となる。

以上を盛り込んで，2つの車線と信号を含む部分（下図）について，我々の定式化枠による表現を試みてみよう。



全体システムの構成：$\&=\{\{E_i|i=1, \cdots k\}, E_{\text{gen}}, E_{\text{line}}, E_{\text{sig}}\}$

for $2\leqq i\leqq c_{\text{line}}$

(1)　状態集合：$C_i=A\times V\times P$；Aは加速度，Vは速度，Pは位置の値を表す集合。

(2)　操作変数：$M_i=A^T$；Aは加速度を表す集合。

(3)　関連システム：

$$\text{添字集合：}L_i=\begin{cases}\{i, i-1, \text{ sig}\} & \text{if } i>1\\ \{1, \text{sig}\} & \text{if } i=1\end{cases}$$

$$\text{状態空間：}H_i=\Pi(C_j|j\in L_i)$$

(4)　状態遷移：$\varphi_i: C_i\times M_i\times K\to C_i$

$$;((a, v, p), m, (t, t'))$$
$$\mapsto(m(t'), v+\int_t^{t'} m(\tau)\,dt, p+(t'-t)v+\int_t^{t'}\left[\int_0^s m(\tau)\,d\tau\right]ds)$$

(5)　操作遷移：$\mu_i: M_i\times H_i\times T\to M_i$

$$;(m, c, t)\mapsto\begin{cases}m|^t\cdot\eta_i[t](m, c) & \text{if } c\in G_t\\ m & \text{otherwize}\end{cases}$$

(6)　操作生成規則：(G_i, η_t)

(a) 操作状態：$G_i=G_{iA}\cup G_{iB}$

$c=((a_i, v_i, p_i), (a_{i-1}, v_{i-i}, p_{i-1}), (c_{sig})\in H_i$ とする。

減速１：$G_{iA}=\{c\in H_i | p_{i-1}-p_i<10 \ \& \ v_{i-1}<v_i\}$

減速２：$G_{iB}=\{c\in H_i | v_{i-1}-v_i>5\}$

⋮

※ 結局，全体システムの状態 c が G に入る時，操作を変更する。次に，G に入るのがいつかが計算できればシミュレーションは可能である。

※ 詳細モデルは，紙面の都合で割愛する。ここではもっとも単純な仮定によって，操作関数を設定する。

(b) 操作関数：$\eta_i : M_i\times T\to M_i|_t$

$$;(mt)\mapsto\eta_i(m, t) : T|_t\to\{-1, 0, 1\}$$

$$\eta_i(m, t)(\tau)=\begin{cases} 1 & \text{if } \tau=t \\ -1 & \text{if } \tau=t-\ln(\text{rand})/\text{サービス率} \\ m(\tau) & \text{otherwize}\end{cases}$$

$\underline{i=\text{line}}$

(1) 状態集合：$C_{\text{line}}=\{0, 1, 2, \cdots\}$；車線内の自動車数を表す集合

(2) 操作変数：$M_{\text{line}}=\{m : T\to Z | (\forall z)(z\in Z \ \& \ z\neq 0\to |m^{-1}(z)| \text{ は有限}\}$

但し，Z は整数集合。

(3) 関連システム：

添字集合：$L_{\text{line}}=\{\text{line, gen, 1}\}$

状態空間：$H_{n+1}=C_{\text{line}}\times C_{\text{gen}}\times C_1$

(4) 状態遷移：$\varphi_{\text{line}} : C_{\text{line}}\times M_{\text{line}}\times K\to C_{\text{line}}$

$$;(c, m)\mapsto c+\sum_{z\in Z} z|m^{-1}(z)|$$

(5) 操作遷移：$\mu_{\text{line}} : M_{\text{line}}\times H_{\text{line}}\times T\to M_{\text{line}}$

$$;(m, c, t)\mapsto\begin{cases} m|^t\cdot\eta_{\text{line}}(m, c, t) & \text{if } c\in G_{\text{line}} \\ m & \text{otherwize}\end{cases}$$

(6) 操作生成規則：$(G_{\text{line}}, \eta_{\text{line}})$

(a) 操作状態：$G_{\text{line}}=G_A\cup G_B$

$c=(c_{\text{line}}, c_{\text{gen}}, (a_1, v_1, p_1))\in H_{\text{line}}$ とする。

$G_A=\{c\in H_{\text{line}} \mid c_{\text{line}}>0 \ \&\ p_1>\text{交差点}\}$

$G_B=\{c\in H_{\text{line}} \mid c_{\text{gen}}=1\}$

$\eta_{\text{line}}: M_{\text{line}}\times H_{\text{line}}\times T\to M_{\text{line}}|_t$

$;(m, c, t)\mapsto\eta_{\text{line}}(m, c, t): T|_t\to Z$

$$\eta_{\text{line}}(m, c, t)(\tau)=\begin{cases} m(\tau)-1 & \text{if } \tau=t \ \&\ c\in G_A-G_B \\ m(\tau)+1 & \text{if } \tau=t \ \&\ c\in G_B-G_A \\ m(\tau) & \text{otherwize} \end{cases}$$

$i=$gen

(1) 状態集合：$C_{\text{gen}}=\{0, 1\}$

(2) 操作変数：$M_{\text{gen}}=\{m: T\to\{0, 1\} \mid m^{-1}(1)\ \text{は有限}\}$

(3) 関連システム：

添字集合：$L_{\text{gen}}=\{\text{gen}\}$

状態空間：$H_{\text{gen}}=C_{\text{gen}}$

(4) 状態推移：$\varphi_{\text{gen}}: C_{\text{gen}}\times M_{\text{gen}}\times K\to C_{\text{gen}}$; $(c, m, (t, t'))\mapsto m(t')$

(5) 操作推移：$\mu_{\text{gen}}: M_{\text{gen}}\times C_{\text{gen}}\times T\to M_{\text{gen}}$

$$;(m, c, t)\mapsto\begin{cases} m|^t\cdot\eta_{\text{gen}}(m, c, t) & \text{if } c\in G_{\text{gen}} \\ m & \text{otherwize} \end{cases}$$

(6) 操作生成規則：$(G_{\text{gen}}, \eta_{\text{gen}})$

(a) 操作状態：$G_{\text{gen}}=\{1\}\subset C_{\text{gen}}$

(b) 操作関数：$\eta_{\text{gen}}: M_{\text{gen}}\times C_{\text{gen}}\times T\to M_{\text{gen}}|_t$

$;(m, c, t)\mapsto\eta_{\text{gen}}(m, c, t): T|_t\to\{0, 1\}$

$$\eta_{\text{gen}}(m, c, t)=\begin{cases} 1 & \text{it } \tau=t-\ln(\text{rand})/\text{到着率} \\ m(\tau) & \text{otherwize} \end{cases}$$

最後に，上述の定式化枠による記述のシミュレーションプログラムへの変換の概略を示しておく。

(1) **離散型シミュレーションモデルの構造：要素システムの状態表現**

構成要素の状態表現：状態量は，C×F で表現される。

	構成要素	状態量C	予約事象F
要素システム	運転者 ＋ 車	加 速 度：kasoku：real 速　　度：sokudo：real 位　　置：place：real; 回転方向：turn：0..2;	予約時刻：jikoku：real; 加 減 速：accel_break：real;
調整機構	信号機	色　　：byr：1..3;	予約時刻：jikoku：real;

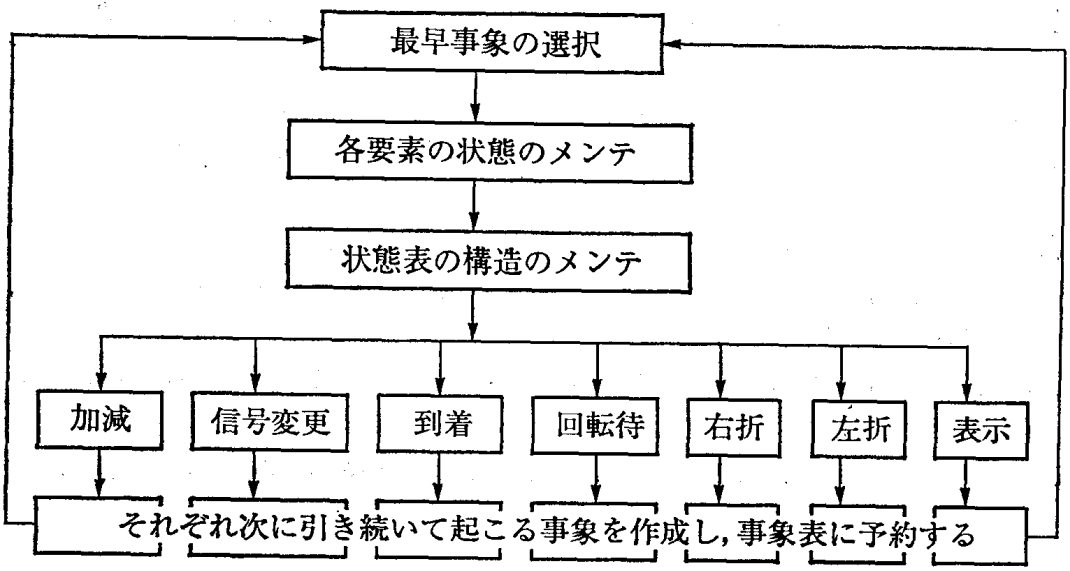
※ 要素システムの状態遷移：$\phi_{tt'}$：$(C\times F)\times X_{tt'}\rightarrow(C\times F)$

$X_{tt'}$：加速度の時間関数

※ 自らあるいは系内の他の要素が事象を発生しないかぎり，あるパラメータ（この場合加速度）の $\phi_{tt'}$ が適用される。

(2) シミュレーションプログラムの流れ：

(1)で示した状態遷移の動態を表現するために，以下のフローで表される離散型シミュレーションプログラムを作成する。



おわりに

本論文では，階層システム論の立場から，動的な複雑システムのモデル理論的定式化を試みた。これは，動的複雑システのシミュレーションの理論的基礎を与えるとともに，シミュレーションモデル作成のガイドラインとして使うた

めである。定式化は，モデル理論のフレームワークを用いて行われた。つまり，あるタイプを持つ一階の述語論理，すなわち言語Lと，その上で表される一連の公理群が，複雑システムの行動を記述するための枠として用いられることを示した。定式化と具体的なシミュレーションプログラムとの関係については，定式化枠による記述の具体例を示すにとどめ，詳述は割愛した。今後は，本論の定式化枠を基礎とするシミュレーションプログラム開発を通じて，動的階層システムの諸法則を整理して行く必要がある。

参考文献

[1] M.D. Mesarovic, D. Macko and V. Takahara, "Theory of Hierarchical, Multilevel System", Academic Press, 1971.

[2] M.D. Mesarovic, V.Takahara, "General Systems Theory: Mathematical Foundations", Academic Press, 1975.

[3] G.Gratzer, "Universal Algebra, Springer-Verlag, 1976.

[4] J. Bridge, "Beginning Model Theory", Oxford Logic Guides, 1977.

[5] M.G. Sign, "Dynamical Hierarchical Control", north-holland, 1977.

[6] Y. Takahara and T. Takai, "The Category Thegory of Time Systems", Int. J. General Systems, Vol 12, pp.71-105, 1986.

[7] P. Checkland, "System Thinking, Systems practice", John Wiley & Sons Ltd., 1981.

[8] M.D. Mesarovic and Y. Takahara "Lecture Notes in Control and Information Science", Springer-Verlag, 1989

[9] 高橋安人，「システムと制御（上下巻）」，岩波書店，1978.

[10] 高原康彦，高井徹雄，「complex goal seeking system の Coordination について」，第8回システムシンポジウム，1982.

[11] 高井徹雄，「マネジメントシステム研究への適用を考慮した階層システムモデルの定式化」，熊本商大論集，第32巻第1号，1985.

[12] P. Checkland 著，高原康彦監訳，高井他訳「新しいシステムアプローチ」，オーム社，1985.

補　遺

1. 補題1の証明：

(a) 任意の $c\in C$ に対して，$c\in\Pi(C_i|i\in I)$ を示せばよい。

$C(c)$ なる $c\in A$ を任意にとる。このとき，$\alpha-(3)$ より $c\subset I\times A$ である。また，$\alpha-(4)$ より，c は関数 $I\to A$ とみなせる。さらに，C_i の定義より，$c(I)=\{c(i)|i\in I\}$

$=\cup(C_i|i\in I)$ が成り立つ。

よって，c は関数

$c: I\to\cup(C_i|i\in I)$ とみなせる。

また，任意の $i\in I$ に対して，$c(i)\in C_i$ も定義より明らかである。

したがって，$c\in\Pi(C_i|i\in I)$ が成り立つ。

(b) $\alpha-(7)$，(8) より，(a) と全く同様に証明できる。 Q.E.D.

2. 補題2の証明：

任意に $a\in C_i$ をとる。これが $a\in\Pi(B_{ij}|j\in J_i)$ を示せばよい。まず，C_i の定義より，ある $c\in C$ があって，$(i, a)\in c$ となる。任意に，$p\in a$ なとると，$\alpha-(5)$ より，ある J, j, b が存在して，$J=E(i)=J_i \,\&\, j\in J_i \,\&\, p=(j,b)\in a$ となる。さらに，B_{ij} の定義より，$b\in B_{ij}$ である。

以上より，$a\subset J_i\times\cup(B_{ij}|j\in J_i)$ である。さらに，$\alpha-(6)$ より，a は関数 $a: J_i\to\cup(B_{ij}|j\in J_i)$ とみなせる。しかも，上の検討より，任意の $j\in J_i$ に対して，$b=a(j)\in B_{ii}$ となる。以上で，$a\in\Pi(B_{ij}|j\in J_i)$ が示せた。 Q.E.D.

3. 補題3の証明：

任意に $v\in M_i$ をとる。$\alpha-(9)$ および P_i の定義より，$v\subset T\times P_i$ である。また，$\alpha-(10)$ より，$v: T\to P_i$ とみなせる。よって，$v\in P_i^T$ Q.E.D.

4. 補題4の証明：

φ については $\alpha-(11)$，(12) より，また μ については $\alpha-(13)$，(14) より明らか。

Q.E.D.

5. 補題5の証明：

(i) $\alpha-(16)$ は，モデル上における

$$(\forall c)(c\notin D\to(\forall m)(\forall t)(\forall w)(\mu(m, c, t, w)\to m=w))$$

を表す。また，補題4より，μ は関数 $M\times C\times T\to M$ とみなせる。

よって，(i)が成り立つ。

(ii) $\alpha-(15)$ より明らか。 Q.E.D.

6. 補題6の証明：

(i) まず，関係 $\varphi_i\subset(C_i\times M_i\times K)\times C_i$ について，

$\mathcal{D}(\varphi_i)\triangleq\{(c,m,k)\in C_i\times M_i\times K|(\exists d)((c,m,k,d)\in\varphi_i))\}$

とおくと $\mathcal{D}(\varphi_i)=C_i\times M_i\times K$ であることを示す。

$\mathcal{D}(\varphi_i)\subset C_i\times M_i\times K$ は，明らかなので，$C_i\times M_i\times K\subset\mathcal{D}(\varphi_i)$

のみ示せばよい。

$(c,m,k)\in C_i\times M_i\times K$ を任意にとれば，C_i, M_i の定義より，ある $x\in C$，$y\in M$ が存在して，$c=x(i)$，$m=y(i)$ となっている。

補題4より，φ は関数なので，$z=\varphi(x,y,k)\in C$ が一意に存在。

このとき，$d=z(i)\in C_i$ とおけば，$(c,m,k,d)\in\varphi_i$ である。

よって，$(c,m,k)\in\mathcal{D}(\varphi_i)$ となる。

つぎに，任意の (c, m, k, d)，$(c, m, k, e)\in\varphi_i$ なる $c, d, e\in C_i$，$m\in M_i$，$k\in K$ に対して，$d=e$ が示せれば，証明は終わる。

φ_i の定義より，ある $x, y\in C$，$u, v\in M$ があって，

$x(i)=y(i)=c$ & $u(i)=v(i)=m$ & $\varphi(x, u, k)(i)=d$ & $\varphi(y, v, k)(i)=e$

となっている。このとき，$\alpha-(17)$ より，$d=e$ となる。

(ii)　φ_i が関数ならば，定義より明らか。　Q.E.D.

7.　補題7の証明

$\alpha-(19)$ を用いて，補題6(i)と同様に証明できる。　Q.E.D.

8.　補題8の証明

(1)　φ が関数であることと，$\alpha-(20)$ を用いれば，結果を得る。

(2)　(1)の結果と，補題6の分解可能性から導かれる。　Q.E.D.

9.　補題9の証明

まず，$\alpha-(18)$ より，$\varphi(c, m, t, t)=c$ である。よって，条件(b)より，$c\notin D$ となる。

補題5(i)より，$\mu(m, c, t)=m$ となる。さらに，条件(b)と補題5(i)より，

$\mu(m, \varphi(c, m, (t, s)), s)=m$ ゆえ，状態—操作適合性が成り立つ。　Q.E.D.