跡見学園女子大学マネジメント学部紀要　第6号　(2008年3月15日)

# 並列動的計画法複合アルゴリズムとスペースプレーン経路最適化問題への応用

A Parallel Hybrid Dynamic Programming Algorithm
and Its Application to Aerospace Plane Trajectory Optimization Problems

花　岡　照　明

**要　旨**

本論文では，複雑な非線形最適制御問題に対し，統一的に適用できる算法として，動的計画法と分枝限定法を併用した動的計画法複合アルゴリズムを考察している．今回の複合アルゴリズムでは，分枝限定と優先順位計算をとり入れた従来の複合アルゴリズムに対し，計算上の工夫として，コストの下界値を精度よく推定する繰り返し論理と，局所解を避ける並列処理を組み入れた算法を提案する．それらによって数値計算上の負荷（計算数とサイズ）を大幅に削減し，局所解を回避することができる．繰り返し論理では，計算数を左右する下界値の推定精度を向上させている．一方，並列処理は，複数の最適経路候補を効率的に取り扱い，局所解へ陥ることを避けている．本論文では，この拡張した複合アルゴリズムをスペースプレーンの上昇経路最適化問題に適用した結果が示されている．

キーワード：動的計画法，ブランチ・アンド・バウンド，スペースプレーン，上昇経路最適化

## 1　はじめに

今日，航空宇宙をはじめ，多くの分野で，システムに対する要求は複雑かつ多様化してきている．一方，これらの要求を含んだ複雑な問題を容易に扱うことのできる計算法の開発が強く望まれている．本稿では，複雑な非線形最適制御問題を統一的に解く計算法とそのスペースプレーン経路最適化問題への応用について考察する．

動的計画法[(1)]は，基本的には，どのような複雑なシステムに対しても数値的に取り扱うことのできる手段としてよく知られている．さらに，動的計画法を用いると，大域的最適解（以

下，大域解と表記）やフィードバック構造をもつ解が得られること，制約条件や非線形の取扱いが容易であること，状態の遷移やコスト関数の表現が表関数であってもよいなど，数多くの望ましい性質を利用できる．しかしながら，動的計画法は，「Bellmanの次元の呪い」の問題[1]，すなわち，状態変数の次元数の増加に伴って計算数が指数関数的に増加し，特別な問題を除き，計算実行上，求解が困難，という致命的な欠陥を持つため，次元の高い問題への適用は，きわめて難しいとされてきた．本論文では，動的計画法の応用で最も障害となっている「過大な計算数」を低減する手法について考察する．

動的計画法の計算数を削減するのに，分枝限定法を併用する方法がある[2~5]．この方法が効果を発揮するかどうかの最大要因は，分枝限定法の限定操作で使用する強力な下界値の作り方にある．Hanaoka and Tanabe[4] およびHanaoka[5] では，再突入飛行体や航空機の経路最適化などの実際的な連続系に対して，動的計画法と分枝限定法を併用した，動的計画法複合アルゴリズム（基本形複合アルゴリズム）を適用し，計算数を大幅に削減させることに成功している．その中では，アルゴリズムの実行で必要となるコストの下界値を，システムの特殊性を利用して計算する方法が提案されている．

今回の複合アルゴリズムでは，この基本形複合アルゴリズムに対し，計算上の工夫として，コストの下界値を精度よく推定する繰り返し論理と，局所解を避ける並列処理を組み入れた算法（並列繰り返し型複合アルゴリズム）を提案する．それらによって数値計算上の負荷（計算数とサイズ）を大幅に削減し，局所解を回避することができる．

この動的計画法複合アルゴリズムは，ダーウィン進化，すなわち，淘汰，増殖，突然変異の3つの繰り返し過程，の一部をまねたような算法であり，従来型動的計画法[1] とは異なった構成法に基づいている．淘汰過程は，本法の「ブロックを使った優先順位計算」に，増殖過程は，量子化した制御の適用による「経路群の繰り返し生成」に対応する．突然変異の組入れは，本稿では扱わない．本稿で述べる複合アルゴリズムの計算メカニズムは，分散的に，ばらばらに存在する経路群を，最適な経路へと集約させるもので，その理論的根拠を前向き動的計画法[4,5] に置いている．

## 2　最適制御問題と動的計画法

いま，制御システムは，

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}_k(\boldsymbol{x}_k, \boldsymbol{u}_k) \qquad (k = 0, 1, 2, \ldots, N) \tag{1}$$

で記述されるものとする．ここで，$\boldsymbol{x}_k$ と $\boldsymbol{u}_k$ を，それぞれ，$n$ 次元状態ベクトルおよび $m$ 次元制御ベクトルとする．また，$k$ を段変数の添え字，$\boldsymbol{g}_k$ を $n$ 次元ベクトル値関数とする．このとき，最小化したい評価関数を

$$J = \sum_{i=0}^{N-1} L_i(\boldsymbol{x}_i, \boldsymbol{u}_i) + \Phi_N(\boldsymbol{x}_N) \tag{2}$$

とすれば，最適制御問題は

$$\text{Minimize } J = \sum_{i=0}^{N-1} L_i(\boldsymbol{x}_i, \boldsymbol{u}_i) + \Phi_N(\boldsymbol{x}_N) \tag{3}$$

$$\text{Subject to } \boldsymbol{x}_{k+1} = \boldsymbol{g}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)(k = 0, \ldots, N-1) \tag{4}$$

$$\boldsymbol{x}_0 = \boldsymbol{c}_0 \tag{5}$$

$$\boldsymbol{x}_N \in \Omega_F \tag{6}$$

$$\boldsymbol{x}_k \in X_k \quad (k = 1, \ldots, N-1) \tag{7}$$

$$\boldsymbol{u}_k \in U(\boldsymbol{x}_k) \quad (k = 0, \ldots, N-1) \tag{8}$$

となる．ただし，$L_k(\boldsymbol{x}_k, \boldsymbol{u}_k)(k = 0, 1, \ldots, N-1)$ を 1 段当りのコスト関数，$\Phi_N(\boldsymbol{x}_N)$ を最終段のコスト関数とする．また，$X_k$ と $U(\boldsymbol{x}_k)$ を，それぞれ $k$ 段での許容状態集合と許容制御集合とする．(5) と (6) 式は，それぞれ，初期条件と終端条件であり，$\Omega_F(\Omega_F \subset X_N)$ を終端条件を満たす集合とする．

上記の問題を従来型（後向き）動的計画法 [1] を用いて解くには，各段 $k$ $(k = 0, 1, \ldots, N)$ でのすべての許容状態 $\boldsymbol{x}_k \in X_k$ に対して動的計画法の再帰関数方程式を適用し，それらの状態 $\boldsymbol{x}_k$ の最適制御 $\boldsymbol{u}_k^*$ を計算する．この再帰関数方程式は，$k$ 段以後の最小コスト関数を

$$J_k(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k, \boldsymbol{u}_{k+1}, \ldots, \boldsymbol{u}_{N-1}} \{ \sum_{i=k}^{N-1} L_i(\boldsymbol{x}_i, \boldsymbol{u}_i) + \Phi_N(\boldsymbol{x}_N) \}$$

と定義すると，Bellman の最適性の原理 [1] から，

$$J_N(\boldsymbol{x}_N) = \Phi_N(\boldsymbol{x}_N)$$

$$J_k(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k \in U(\boldsymbol{x}_k)} \{ L_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + J_{k+1}(\boldsymbol{g}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)) \}$$
$$(k = 0, \ldots, N-1) \tag{9}$$

となる．この (9) 式から最適方策を解析的に求めることは多くの場合難しいので，通常，数値的に解くことになる．

(9) 式を解くには，通常，各状態変数 $x_k^i(i = 1, 2, \ldots, n)$ を $l_x$ レベルに，また，各制御変数 $u_k^j(j = 1, 2, \ldots, m)$ を $l_u$ レベルに量子化する．そして，状態ベクトルの $n$ 次元格子の各点において量子化した制御を適用し，(9) 式を用いて格子点での $J_k(\boldsymbol{x}_k)$ と最適制御 $\boldsymbol{u}_k^*$ を求める．もし，この計算で，次段の状態 $\boldsymbol{g}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)$ が格子点上に無い場合は，格子点での $J_{k+1}(\boldsymbol{x}_{k+1})$

を用い，内挿によって $J_{k+1}(\boldsymbol{g}_k(\boldsymbol{x}_k, \boldsymbol{u}_k))$ を計算する．状態変数の次元数 $n$ が大きいとき，この内挿計算は相当煩わしい．(9) 式の計算は $k = N-1$ から始め，後向きに，$k = 0$ まで再帰的に行う．

従来型動的計画法を適用する上で最も難しいことは，状態変数 $\boldsymbol{x_k}$ の次元数の増加に伴ってメモリーサイズ（データの格納領域）と計算数が飛躍的に増大することである．これは，状態ベルトルを量子化することに起因する．たとえば，問題の定義域の次元数が $n = 5$ で，段数が $N = 10$ の問題に対し，$\boldsymbol{x_k}$ の各要素当り $l_x = 20$ レベルに分割したとき，(9) 式の右辺を評価するために計算機メモリーに格納する必要がある $J_{k+1}(\boldsymbol{x}_{k+1})$ の格子点の数は $l_x^n = 20^5 = 3,200,000$ 個である．また，$J_k(\boldsymbol{x}_k)$ と $\boldsymbol{u}_k^*$ の計算を必要とする格子点の総数（計算領域）は $l_x^n \cdot N = 20^5 \times 10 = 32,000,000$ 個となる．量子化レベルをさらに増やしたとき，従来型動的計画法によるアプローチでは事実上，計算が実行不可能に陥る．

## 3　基本形複合アルゴリズム

従来型動的計画法は，状態の遷移方向（通常は，時間の増加方向）に関し，後向きに定式化している．これは，「後向き」最適性の原理 (1) を用いたことによる．一方，基本形複合アルゴリズムで用いる前向き動的計画法は，この原理を，時間に関して前後を逆にした「前向き」最適性の原理に基づき，初期点から前向きに定式化する．

いま，量子化した許容制御 $\boldsymbol{u}_k \in U(\boldsymbol{x}_k)$ を初期点 $\boldsymbol{x}_0$ から，各段 $k$ で繰り返し適用してできる，初期点からの実行可能経路集合を考え，$\{X_k^\circ\} = \bigcup_{i=0}^{k} X_i^\circ$ とおく．ただし，$X_k^\circ$ は，再帰的に定義され，$X_0^\circ = \{\boldsymbol{x}_0\}, X_{k+1}^\circ = \{\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_{k+1} = \boldsymbol{g}_k(\boldsymbol{x}_k, \boldsymbol{u}_k), \boldsymbol{x}_k \in X_k^\circ, \boldsymbol{u}_k \in U(\boldsymbol{x}_k)\} (k = 0, \ldots, N-1)$ とする．また，$\boldsymbol{g}_k$ を状態遷移関数とする．この経路群は初期点からの枝構造をもつ．このような，状態点の決め方，すなわち，$k$ 段の状態点 $\boldsymbol{x}_k \in X_k^\circ$ を押し出すことによって，$k+1$ 段の状態点 $\boldsymbol{x}_{k+1} \in X_{k+1}^\circ$ を確定させる計算を，「押し出し計算」と呼ぶことにする．ここで，計算上の工夫として，前向き動的計画法でのすべての計算点を，押し出し計算によって，この実行可能経路 $\{X_k^\circ\}$ 上にとり，かつ，コスト関数の計算と比較を，正確にこの経路上のコスト値を用いて行うことを考える．このような処理では，コスト関数の内挿計算を含まない．しかし，このままでは，段数 $N$ が大きいとき，実行可能経路数の巨大化を招く．この巨大化を避けるため，次節では，量子化について考察する．

### 3．1　量子化手続きと前向き動的計画法

基本形複合アルゴリズムでは，前向き動的計画法を適用するのに，許容状態集合 $X_k$ を，$X_k = \bigcup_{i=1}^{n_k} X_{ki}, X_{ki} \cap X_{kj} = \emptyset (i \neq j)$ となるように，適当な部分集合 $X_{k1}, X_{k2}, \ldots, X_{kn_k}$ に量子化する．以後，この部分集合 $X_{ki}$ を，「ブロック」と呼ぶ．各ブロック $X_{ki}$ に対し，初期点 $\boldsymbol{x}_0$ か

らの経路が存在する場合のみ，一個ずつ代表点 $\boldsymbol{x}_{ki}(\in X_{ki})$ とそのコスト関数の値を以下のように定義する．すなわち，

$$f_k(\boldsymbol{x}_{ki}) = \min_{\boldsymbol{x}_k}\{T(\boldsymbol{x}_k) \mid \boldsymbol{x}_k \in X_{ki}\}$$
$$(i = 1, \ldots, m_k,\ k = 0, \ldots, N) \tag{10}$$

とする．ここで，$T(\boldsymbol{x}_k)$ は，前段の代表点 $\boldsymbol{x}_{k-1i}(\boldsymbol{x}_{k-1i} \in X_{k-1i})$ に，量子化した各 $\boldsymbol{u}_{k-1} \in U(\boldsymbol{x}_{k-1i})$ を適用し，式，

$$\begin{aligned} &T(\boldsymbol{x}_0) = 0 \\ &T(\boldsymbol{x}_k) = f_{k-1}(\boldsymbol{x}_{k-1i}) + L_{k-1}(\boldsymbol{x}_{k-1i}, \boldsymbol{u}_{k-1}) \\ &\boldsymbol{x}_k = \boldsymbol{g}_{k-1}(\boldsymbol{x}_{k-1i}, \boldsymbol{u}_{k-1}) \\ &\qquad (i = 1, \ldots, m_k, k = 1, \ldots, N) \end{aligned} \tag{11}$$

を用い，押し出し計算によって計算する．ここで，$L_k$ を 1 段当りのコスト関数とする．ただし，便利さのため，最終段 $N$ でのコスト $\Phi_N(\boldsymbol{x}_N)$ を，$N-1$ 段でのコスト $L_{N-1}$ に含めて考える．また，$m_k(m_k \leq n_k)$ を，$k$ 段の代表点の数とする．この代表点 $\boldsymbol{x}_{ki}$ が，初期点 $\boldsymbol{x}_0(\boldsymbol{x}_0 = \boldsymbol{x}_{01})$ から，$k$ 段の各ブロック上での到達点までの経路の中で，対応するコスト関数の値が（各ブロックの中で）最小となる点であり，初期点から再帰的に計算できる．このような代表点におけるコスト関数の最小値を最小コスト関数と呼び，代表点 $\boldsymbol{x}_k$ の関数として $f_k(\boldsymbol{x}_k)$ とおく．このとき，有限個の代表点 $X_k^\circ = \{\boldsymbol{x}_{k1}, \boldsymbol{x}_{k2}, \ldots, \boldsymbol{x}_{km_k}\}(k = 0, \ldots, N)$ 上で考慮した前向き動的計画法が成立する．すなわち，

$$\begin{aligned} &f_0(\boldsymbol{x}_{01}) = 0 \\ &f_{k+1}(\boldsymbol{x}_{k+1i}) = \min_{\boldsymbol{x}_{ki}, \boldsymbol{u}_k}\{f_k(\boldsymbol{x}_{ki}) + L_k(\boldsymbol{x}_{ki}, \boldsymbol{u}_k) \mid \\ &\qquad \boldsymbol{x}_{ki} \in X_k^\circ, \boldsymbol{u}_k \in U(\boldsymbol{x}_{ki})\} \quad (i = 1, \ldots m_k) \\ &\boldsymbol{x}_{k+1i} = \boldsymbol{g}_k(\boldsymbol{x}_{ki}, \boldsymbol{u}_k) \qquad (k = 0, \ldots, N-1) \end{aligned} \tag{12}$$

とする．

一方，代表点以外の各ブロック内の状態点の最小コスト関数の値は，代表点の値で近似し，

$$\text{もし}\boldsymbol{x}_k \in X_{ki}\text{ならば，}\ f_k(\boldsymbol{x}_k) = f_k(\boldsymbol{x}_{ki})$$
$$(i = 1, 2, \ldots, m_k,\ k = 0, 1, \ldots, N-1)$$

とする．

### 3．2　優先順位計算

基本形複合アルゴリズムの計算は，初期点から前向きに進行する．初期点および各段の代表点では，量子化した制御を適用し，枝構造をもった最適経路候補群を繰り返し生成させる．この際，(11) 式によって計算されるコスト関数値 $T(\boldsymbol{x}_k)$ の小さい順に，各経路がブロックに到着するように処理する．以後，この処理を優先順位計算と呼ぶことにする．このようにすると，代表点の定義により，ブロックに最初に到着した経路がそのブロックの代表点となる．それ以後に到着するいかなる経路も代表点となることはできないから，それらを削除できる．このアルゴリズムは，初期点からの経路のいずれかが，終端条件を満たす状態集合 $\Omega_F$ に最初に到着したとき終了する．このとき，この先着経路が最適経路となる．この先着経路に対応する最適コストを

$$f^*_{0,N} = \min_{\boldsymbol{x}_N}\{f_N(\boldsymbol{x}_N) \mid \boldsymbol{x}_N \in \Omega_F\} \tag{13}$$

と定義することにする．

### 3．3　限定操作とクリアランス

基本形複合アルゴリズムでは，前向き動的計画法の計算数を削減するために，分枝限定法の限定操作を使う．

いま，(9) 式の最小コスト関数 $J_k(\boldsymbol{x}_k)$ の下界値を $M_k(\boldsymbol{x}_k)$ とし，また，原問題 (3)～(8) 式下での最終最適解 $f^*_{0,N}$(13) 式の上界値を $I$（暫定解）とする．これらの下界値と上界値が満たすべき条件は，それぞれ

$$\begin{aligned} M_k(\boldsymbol{x}_k) &\le J_k(\boldsymbol{x}_k), \qquad \boldsymbol{x}_k \in X_k \\ &= \min_{\boldsymbol{u}_k,\boldsymbol{u}_{k+1},\ldots,\boldsymbol{u}_{N-1}}\{\sum_{i=k}^{N-1} L_i(\boldsymbol{x}_i,\boldsymbol{u}_i) + \Phi_N(\boldsymbol{x}_N)\} \\ &\qquad\qquad (k = 0,\ldots,N-1) \end{aligned} \tag{14}$$

$$I \ge f^*_{0,N} \tag{15}$$

で与えられる．よく知られているように，もし，

$$f_k(\boldsymbol{x}_k) + M_k(\boldsymbol{x}_k) > I \qquad (k = 0,\ldots,N) \tag{16}$$

ならば，状態 $\boldsymbol{x}_k$ を通る経路について，

$$f_k(\boldsymbol{x}_k) + J_k(\boldsymbol{x}_k) \ge f_k(\boldsymbol{x}_k) + M_k(\boldsymbol{x}_k) > I \ge f^*_{0,N}$$

が成り立つ．よって，代表点 $\boldsymbol{x}_k$ は最適経路の一部になれないから削除してよい．ただし，$f_0(\boldsymbol{x}_0)$ は，(11) 式より $f_0(\boldsymbol{x}_0) = T(\boldsymbol{x}_0) = 0$ である．また，ここでは，最終段 $N$ でのコスト $\Phi_N(\boldsymbol{x}_N)$ を，$N-1$ 段でのコスト $L_{N-1}$ に含めて考えたから，形式的に，$J_N(\boldsymbol{x}_N) = 0$ とする．したがって，$M_N(\boldsymbol{x}_N) = 0$ とおく．削除できる代表点 $\boldsymbol{x}_k$ の数を増やすには，(16) 式から明らかなように，上界値 $I$ をより小さな値へ，一方，下界値 $M_k(\boldsymbol{x}_k)$ をより大きな値へと強化すればよい．ここで，上界値と下界値の強さを表わす量として，それぞれ，$\Delta a$ および $\Delta b_k$ を導入し，

$$\Delta a = I - f^*_{0,N}, \quad \Delta b_k = J_k(\boldsymbol{x}_k) - M_k(\boldsymbol{x}_k) \tag{17}$$

とおく．(17) 式の $I$ と $M_k(\boldsymbol{x}_k)$ を (16) 式に代入すると

$$f_k(\boldsymbol{x}_k) + J_k(\boldsymbol{x}_k) > f^*_{0,N} + \Delta a + \Delta b_k \tag{18}$$

となる．(18) 式より，基本形複合アルゴリズムの計算数は，$\Delta a$ や $\Delta b_k$ の個別の値というよりも，それらの和 $\Delta\epsilon(\Delta\epsilon = \Delta a + \Delta b_k)$ に依存する．以後，この和 $\Delta\epsilon$ をクリアランスと呼ぶことにする．基本形複合アルゴリズムを用いて大域解を得るには，(14), (15) 式の制約を満たす有効な下界値と上界値を計算できなくても，より緩和された，大域解を得るためのクリアランスの条件，すなわち，

$$\Delta\epsilon = \Delta a + \Delta b_k \geq 0 \tag{19}$$

を満足すればよい．ここで，この複合アルゴリズムの計算数が最小となるのは，$\Delta\epsilon = \Delta a + \Delta b_k = 0$ のときである．

### 3．4　基本形複合アルゴリズム

基本形複合アルゴリズムは，つぎの 10 ステップに要約できる．

1. (境界条件設定): 初期条件 $\boldsymbol{x}_0 = \boldsymbol{c}_0$ と終端条件 $\boldsymbol{x}_N \in \Omega_F$ を設定する．

2. (状態集合の量子化): 各段の状態集合 $X_k$ を，$n_k$ 個のブロック $X_{k1}, X_{k2}, \ldots, X_{kn_k}$ に分割する (ただし，$X_k = \bigcup_{i=1}^{n_k} X_{ki}, X_{ki} \cap X_{kj} = \emptyset\ (i \neq j)$)．

3. (クリアランスの設定): 各 $k = 0, \ldots, N$ に対し，適当な $\boldsymbol{x}_k \in X_k$ に対する下界値 $M_k(\boldsymbol{x}_k)$ を計算する．また，一つの上界値 I を設定する．(ただし，有効な下界値を計算できないときは，$M_k(\boldsymbol{x}_k) = 0$ とし，$I$ を $\Delta\epsilon \geq 0$ となるように設定する)．

4. (初期化): $\Omega \leftarrow \{\boldsymbol{x}_0\}$, $T(\boldsymbol{x}_0) = 0$, $\Re \leftarrow \emptyset$ を行う ($\Re$ は，そこまでの最小コスト経路が確定したブロックの集合を，また，$\leftarrow$ は代入操作を表わす)．

5. (停止): もし，$\Omega = \emptyset$ なら停止せよ．

6. (前向き動的計画法): $\Omega$ の中から $T(\boldsymbol{x}^*)$ の値が最小である $\boldsymbol{x}^*$ を選び，$\Omega \leftarrow \Omega \backslash \{\boldsymbol{x}^*\}$ とせよ．そして，$\boldsymbol{x}^*$ が属する段の値を $k$ にセットし，$\boldsymbol{x}_k^* \leftarrow \boldsymbol{x}^*$ とせよ（$\backslash$ は差集合の演算子）．

7. (終端テスト): もし $\boldsymbol{x}_k^*$ が終端条件 $\boldsymbol{x}_N \in \Omega_F$ を満たすなら，停止せよ (この段階で最適解が得られる)．

8. (代表点テスト): もし $[x_k^*] \in \Re$ ならばステップ5へ行け．それ以外は $\Re \leftarrow \Re \cup \{[\boldsymbol{x}_k^*]\}$ とし，ステップ9へ行け ([y] は状態 y を含むブロックを示す．この段階で，$\boldsymbol{x}_k^*$ に到達させる最適制御 $\boldsymbol{u}_{k-1}^*$ が確定する．また，最小コスト関数の候補値 $T(\boldsymbol{x}_k^*)$ は，真の最小コスト関数値 $f_k(\boldsymbol{x}_k^*)$ となる.)

9. (分枝限定操作): 量子化した各 $\boldsymbol{u}_k \in U(\boldsymbol{x}_k^*)$ に対して，次段の状態 $\boldsymbol{x}_{k+1}$ および最小コスト関数の候補値 $T(\boldsymbol{x}_{k+1})$ を，$\boldsymbol{x}_{k+1} = \boldsymbol{g}_k(\boldsymbol{x}_k^*, \boldsymbol{u}_k)$, $T(\boldsymbol{x}_{k+1}) = T(\boldsymbol{x}_k^*) + L_k(\boldsymbol{x}_k^*, \boldsymbol{u}_k)$ とする．そして，もし各 $\boldsymbol{x}_{k+1}$ に対し，$T(\boldsymbol{x}_{k+1}) + M_{k+1}(\boldsymbol{x}_{k+1}) \leq I$ ならば，$\Omega \leftarrow \Omega \cup \{\boldsymbol{x}_{k+1}\}$ とせよ．

10. ステップ5へ行け．

ステップ6で $\boldsymbol{x}^*$ を選択するとき，$\boldsymbol{x}$ をそれらに対応する最小コスト関数の候補値のサイズの順序に整列しておくと，探索の手間を削減できる．

## 4　経路群の繰り返し生成による，コスト上下界値の推定精度の向上

4.1節と4.2節で述べる複合アルゴリズムは，3章で述べた基本形複合アルゴリズムを改良したものである．4章以降での複合アルゴリズムは，この改良型の複合アルゴリズムである．

### 4．1　繰り返し複合アルゴリズム

(16) 式による限定操作を強化し，より多くの計算数を削減するため，下界値を精度良く推定する繰り返し論理を考える．この目的のため，複合アルゴリズムの一連の計算を，より細かく量子化した状態集合と許容制御の下で繰り返す．すなわち，$i(i = 2, 3, \ldots)$ 回目の逐次計算では，状態集合 $X_k$ を，前回 $(i-1)$ のブロックよりも，より小さなブロックに分割する．たとえば，各状態変数に対し，量子化幅を前回の半分とし，したがって，量子化レベルを2倍に増やす．一方，許容制御 $u_k^{l,i}$ の範囲を

$$u_{kmin}^{l,i} \leq u_k^{l,i} \leq u_{kmax}^{l,i} \tag{20}$$

ここで,

$$u_{kmin}^{l,i} = u_k^{*l,i-1} - \Delta u_k^i, u_{kmax}^{l,i} = u_k^{*l,i-1} + \Delta u_k^i$$
$$(k = 0, 1, \ldots, N-1\ , l = 1, \ldots m\ ,\ i = 1, 2, \ldots)$$

によって計算する．ここで，$u_k^{*l,i-1}$ は $i-1$ 回目の繰り返し計算で得た最適制御であり，$l$ を制御変数のベクトル成分の添え字とする．また，$\Delta u_k^i$ は，通常，$\Delta u_k^i \leq \Delta u_k^{i-1}$ となるように設定し，制御入力の分割数を変えずに量子化幅を前回よりも細かくする．たとえば，大域解を得る保証を放棄し，$\Delta u_k^i = 0.5\Delta u_k^{i-1}$ とする．また，$i$ 回目の許容制御 $u_k^{l,i}$ をつくる際，$u_k^{*l,i-1}$ を含めるように量子化し，かつ，前回の最適経路上の代表点を $i$ 回目の代表点に加えると，最悪でも，前回の最適コスト値を保証できる．

しかし，状態変数や許容制御のこのような量子化レベルの増加の下では，複合アルゴリズムの計算数を指数関数的に増大させる可能性がある．この増大を抑制するため，上界値 $I^i$ と下界値 $M_k^i(\boldsymbol{x}_k)(\boldsymbol{x}_k \in X_k)$ を，それぞれ，前回 $(i-1)$ の繰り返し計算で得た，最終最適コスト $f*_{0,N}^{i-1}$ と最適経路上のコスト値を用い，それぞれ，

$$I^i = f*_{0,N}^{i-1} \quad (i = 2, 3, \ldots) \tag{21}$$

$$\begin{aligned} M_k^i(\boldsymbol{x}_k) &= J_k^{i-1}(\boldsymbol{x}_k^{*i-1}) \\ &= \sum_{l=k}^{N-1} L_l(\boldsymbol{x}_l^{*i-1}, \boldsymbol{u}_l^{*i-1}) + \Phi_N(\boldsymbol{x}_N^{*i-1}) \\ &\quad (k = 0, 1, \ldots, N-1,\ i = 2, 3, \ldots) \end{aligned} \tag{22}$$

によって強化する．ここで，$\boldsymbol{x}_l^{*i-1}$ と $\boldsymbol{u}_l^{*i-1}$ は，それぞれ，$i-1$ 回目の繰り返し計算で得た最適経路上の状態量と制御量である．(21) 式による $I^i$ は良好な上界値を与える．また，(22) 式による $M_k^i$ は，前回の最適経路の近傍で，ぴったりとした下界値を与える．しかしながら，最適経路の近傍以外では，(22) 式は大域解を得るための下界の必要条件，すなわち，(14) 式の $M_k^i(\boldsymbol{x}_k) \leq J_k(\boldsymbol{x}_k)(k = 0, \ldots, N-1)$ を常に満たすとは限らない．また，この判定には $J_k(\boldsymbol{x}_k)$ の評価を必要とするが，正確な $J_k(\boldsymbol{x}_k)$ の値を知ることは難しい．そこで，次節では，(19) 式の，大域解を得るためのクリアランスの条件 $\Delta\epsilon \geq 0$ を用いて，大域解を得ることを考える．

一方，この繰り返し複合アルゴリズムの現実的な停止条件を

$$|f*_{0,N}^{i} - f*_{0,N}^{i-1}| \ll \epsilon_1 \tag{23}$$

で与える．ただし，$\epsilon_1 \ll 1$ とする．

一般に，状態量や制御量の量子化単位が粗い時点では，経路群が終端条件を満たせなくなって，計算が停止してしまう可能性がある．この現象に対処するため，終端条件が満たされない

ほど終端段のコスト値が大きくなるようなペナルティー関数 $P(\boldsymbol{x}_N)$ を導入する．そして，元の評価関数にこのペナルティー関数項を加えた新しい評価関数

$$J = \sum_{i=0}^{N-1} L_i(\boldsymbol{x}_i, \boldsymbol{u}_i) + \mu P(\boldsymbol{x}_N) \tag{24}$$

$$\text{ここで,} P(\boldsymbol{x}_N) = \Psi(\boldsymbol{x}_N)^T \Psi(\boldsymbol{x}_N) \tag{25}$$

の下で，終端条件のない最小化問題を解く．ここで，$\Psi(\boldsymbol{x}_N) = \mathbf{0}$ は原問題の終端条件であり，$\mu$ は十分大きな正定数とする．このとき，量子化単位を細かくするほど，したがって，イテレーション回数の増加とともに，最適経路は終端条件を満たすように改善される．

このように，繰り返し複合アルゴリズムでは，上界値と下界値を逐次改良する過程で，同時に，初期点からの最小コストを，より精密な値に逐次近似している．

### 4．2　並列複合アルゴリズム

大域解を得るためには，状態変数の定義域全体に対して，$\Delta\epsilon \geq 0$（クリアランスの条件 (19) 式）となる必要がある．そのために，もし，$\Delta\epsilon < 0$ の領域を含む場合は，(22) 式の下界値を下方修正し，

$$M_k^i(\boldsymbol{x}_k) = J_k^{i-1}(\boldsymbol{x}_k^{*i-1}) - |\Delta\,\tilde{\epsilon}_{min}| \qquad (k = 0, 1, \ldots, N-1, i = 2, 3, \ldots) \tag{26}$$

と置く．ただし，$\Delta\epsilon$ の値が明示的でないため，$\Delta\,\tilde{\epsilon}_{min}$ を $\Delta\epsilon(<0)$ の最小値の推定値とする．もし，$\Delta\epsilon \geq 0$ ならば，$\Delta\,\tilde{\epsilon}_{min}=0$ と置く．しかし，コスト関数が多峰性である場合には，(26) 式のように，一つのパラメータ $\Delta\,\tilde{\epsilon}_{min}$ によって，全定義域に渡る下界値を設定することは，クリアランスの増加，したがって，計算数とメモリーサイズの増加を招く．

並列複合アルゴリズムは，このような場合に，大域解を求める実際的な算法である．最初に，並列処理のために，粗い量子化の下で経路候補群を生成する．これらの経路での値を，並列計算の初期値とする．これは原問題のコスト構造を調べることに相当する．すなわち，並列複合アルゴリズムでは，繰り返し複合アルゴリズムの初回 $(i = 1)$ の計算で得た，1番目から $j_{max}$ 番目までのコスト値をもつ最適経路候補群に対し，次回以降 $(i \geq 2)$，それぞれ個別に，繰り返し複合アルゴリズムを適用し，各々の逐次近似解を求める．したがって，並列複合アルゴリズムの $i$ 回目 $(i \geq 2)$ の繰り返し計算が終了したとき，新たな最適経路候補群が生成される．並列複合アルゴリズムに対しても，$i$ 回目のイテレーションでの状態集合と許容制御の量子化のしかた，および，第 $j$ 番目の経路に関する上下界値の更新のしかたは，繰り返し複合アルゴリズムと同様である．すなわち，

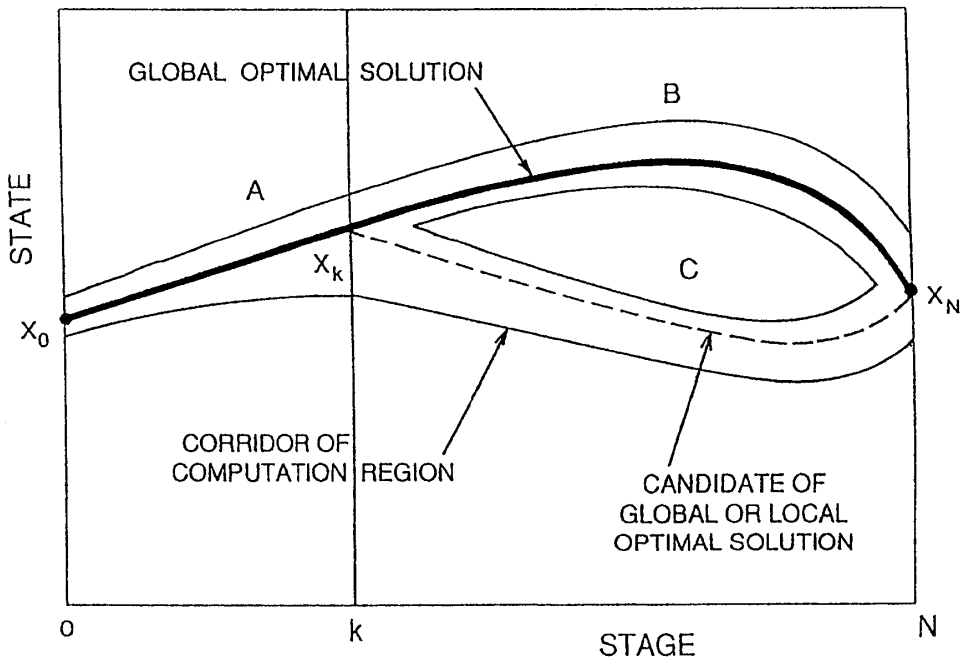


図 1　Parallel HDP の計算回廊

$$I^{j,i} = f*_{0,N}^{j,i-1} \quad (j = 1, 2, \ldots, j_{max},\ i = 2, 3, \ldots) \tag{27}$$

$$\begin{aligned} M_k^{j,i}(\boldsymbol{x}_k) &= J_k^{j,i-1}(\boldsymbol{x}_k^{*j,i-1}) - |\Delta \overset{\sim}{\epsilon}{}^{j}_{min}| \\ &= \sum_{l=k}^{N-1} L_l(\boldsymbol{x}_l^{*j,i-1}, \boldsymbol{u}_l^{*j,i-1}) + \Phi_N(\boldsymbol{x}_N^{*j,i-1}) - |\Delta \overset{\sim}{\epsilon}{}^{j}_{min}| \\ &(k = 0, 1, \ldots, N-1,\ j = 1, 2, \ldots, j_{max},\ i = 2, 3, \ldots) \end{aligned} \tag{28}$$

によって強化する．ここで，$\Delta \overset{\sim}{\epsilon}{}^{j}_{min}$ は，クリアランスの条件 $\Delta\epsilon \geq 0$ (19) 式を満たす範囲を，全定義域から第 $j$ 番目の最適経路周りの回廊と呼ばれる領域（図 1 参照）に緩和（縮小）したときの推定値とする．

さらに，この上界値 $I^{j,i}$ は，並列処理によってすでに計算された，より小さな上界値，すなわち，

$$\begin{aligned} \overset{\sim}{I}{}^{j,i} &= \min_{j_1 \in \{j, j+1, \ldots, j_{max}\}, j_2 \in \{1, 2, \ldots, j-1\}} \{f*_{0,N}^{j_1,i-1}, f*_{0,N}^{j_2,i}\} \\ &(j = 1, \ldots, j_{max},\ i = 2, 3, \ldots) \end{aligned} \tag{29}$$

によって強化することが可能である．このとき，第 $j$ 経路の繰り返し計算で，全ての経路が限定操作により削除され，したがって，第 $j$ 局所解が消去される可能性を期待できる．また，$i$ 回目の繰り返し計算では，より小さな上界値 $I^{j,i}$ をもつ経路を先に処理すれば，早い時点で，より強力な上界値を得る可能性が高くなる．

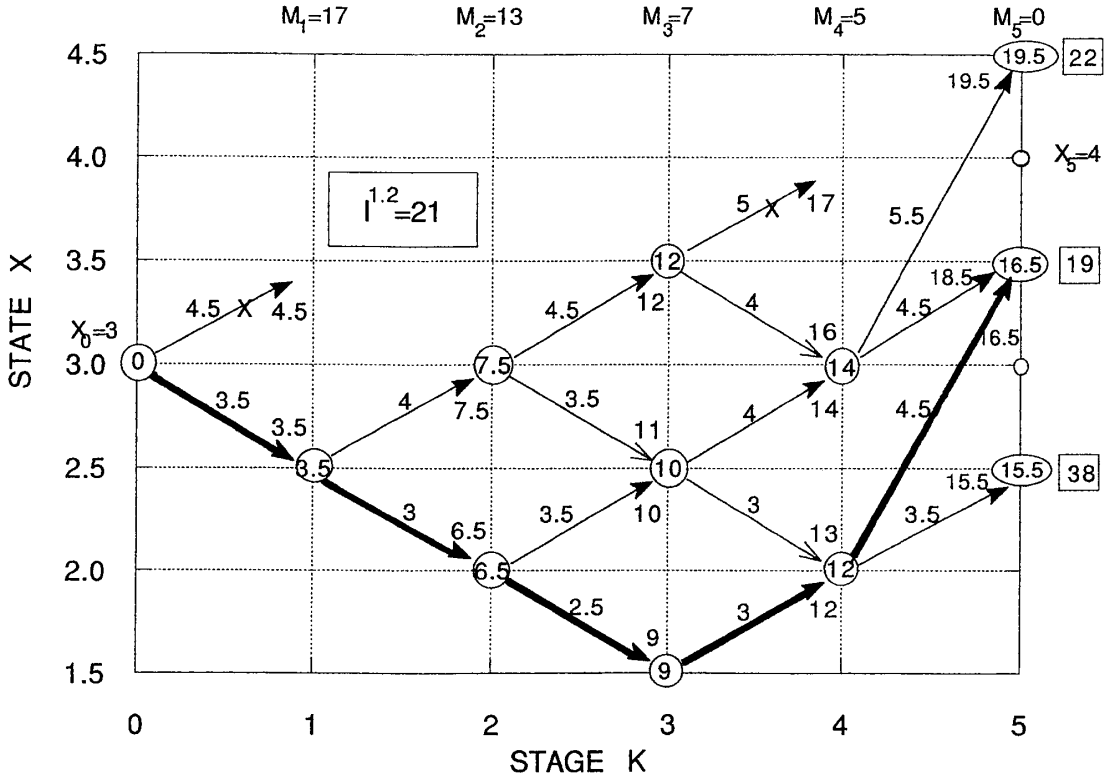


図 2　例題の iteration$i = 1$ における Parallel HDP のスケッチ

前述の並列計算では，固定した各 $i(i = 2, 3, \ldots)$ に対して，$j$ を変化させた．一方，固定した各 $j(j = 1, \ldots, j_{max})$ に対して，$i$ を変化させれば，もう 1 つの並列計算を構成できる．

並列複合アルゴリズムの実行は，前回 $(i = 1, 2, \ldots)$ の量子化単位下で計算した大域解と代表的な局所解の周りに，最適経路候補群からなる計算領域の回廊を逐次生成することに対応している．並列複合アルゴリズム (Parallel Hybrid Dynamic Programming Algorithm, Parallel HDP と略記) の計算領域の回廊を図 1 に示す．この回廊幅のサイズは，クリアランスのサイズ $\Delta\epsilon$ によって制御され，イテレーション回数 $i$ の増加とともに，上下界値の逐次改良によって，徐々に絞られる．

ここで，並列複合アルゴリズムの例題として，システム方程式 $x_{k+1} = x_k + u_k (k = 0, \ldots, 4)$ の下で，評価関数 $J = \sum_{i=0}^{4} \{ \mid x_i \mid + \mid u_i + 1 \mid \}$ を最小にする最適制御列 $\{u_k^*, k = 0, \ldots, 4\}$ を求める問題を考える．初期値，終端値，制御変数を，それぞれ $x_0 = 3, x_5 = 4, u_k \in [-0.5, 1.5]$ としたときの計算手順を以下に示す．

1. 基本形複合アルゴリズムによる計算を第 1 回目のイテレーションとみる．状態量 $x_k$ と制御量 $u_k$ の量子化単位を，それぞれ 1 とし，$u_k \in \{0, 1\}$ で代表させる．また，上下界値 $I^1 = 24$, $M_k^1 = 0$ $(k = 0, 1, \ldots, 4)$ で出発する．この結果は図 2 のとおりで，つぎの 2 つの経路候補が得られる．

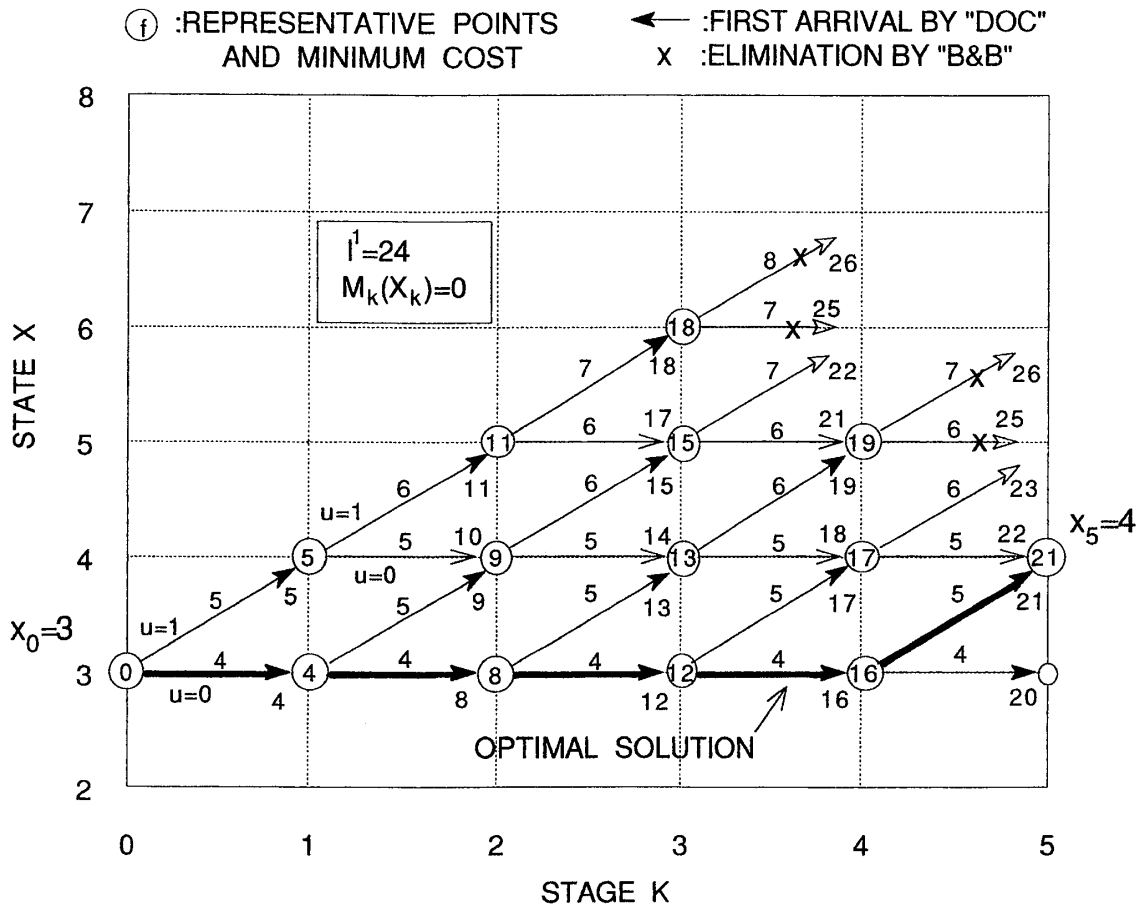


図 3　経路候補 $-1$ に対する iteration $i = 2$ での Parallel HDP のスケッチ

経路候補 $-1$;　$0 \to 0 \to 0 \to 0 \to 1$ （制御方策）

経路候補 $-2$;　$0 \to 0 \to 0 \to 1 \to 0$

2. 第 1 回目のイテレーションの結果，経路候補 $-1$ より $I^{1,2} = f*^{1,1}_{0,N} = 21$（最小コスト）で $M_0^{1,2} = 21$, $M_1^{1,2} = 17$, $M_2^{1,2} = 13$, $M_3^{1,2} = 9$, $M_4^{1,2} = 5$ が，経路候補 $-2$ より $I^{2,2} = f*^{2,1}_{0,N} = 22$（第 2 番目コスト）で $M_0^{2,2} = 22$, $M_1^{2,2} = 18$, $M_2^{2,2} = 14$, $M_3^{2,2} = 10$, $M_4^{2,2} = 5$ が得られる．

第 2 回目のイテレーションでは，状態量 $x_k$ と制御量 $u_k$ の量子化単位を，それぞれ 0.5 とする．すなわち，制御操作 0 については，$u_k \in \{-0.5, +0.5\}$ で代表させ，一方，制御操作 1 については，$u_k \in \{+0.5, +1.5\}$ で代表させる．経路候補-1 について，$I^{1,2} = 21$ として並列複合アルゴリズムを適用した結果は図 3 のとおりで，経路;$-0.5 \to -0.5 \to -0.5 \to +0.5 \to +1.5$ が得られる．ただし，終端条件のミスマッチによるペナルティー項のコスト 2.5($\mu P = 10 \times 0.5^2$) を加え，$I^{1,3} = f*^{1,2}_{0,N} = 19$（最小コスト）となる．一方，経路候補-2 について，$\tilde{I}^{2,2} = 19$（(29) 式を使って $I^{2,2} = 22$ を強化）としたときの

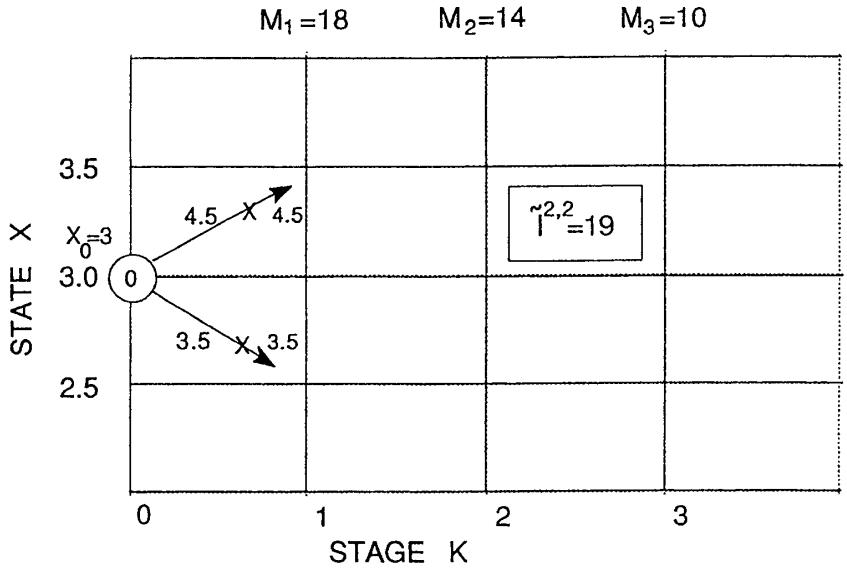


図 4　経路候補 $-2$ に対する iteration $i = 2$ での Parallel HDP のスケッチ

結果は図 4 の通りで，ただちに限定操作が行われ，新たな経路は生成されない．

3. 同様にして，第 3 回目のイテレーションを示すことは容易．

ただし，この例では $\Delta\epsilon = 0, \mu = 10$ として示した．図中，1 段当たりのコスト値 $L_k$ を矢線中央に，最小コスト関数の候補値 $T_k$ を矢線頭部に，また，最小コスト関数値 $f_k$ を丸で囲った数字で示す．ただし，ペナルティーコストを加えた $f_N$ を長方形で囲った数字で併記した．一方，優先順位計算（Dominance Order Calculation, DOC と略記）によって確定した最適制御を太い黒矢印で，(分枝）限定操作（Branch and Bound, B&B と略記）によって削除された経路を×印で示す．

## 5　動的計画法複合アルゴリズムのスペースプレーン上昇経路最適化問題への応用

地球の周回軌道，すなわち，人工衛星などが投入される軌道に，地上から人間や人工衛星などの機材を運ぶには，従来，打ち上げロケットを用いてきた．これに対して，次世代においては，打ち上げロケットに替わり，1 段式で水平離着陸する宇宙飛行機（スペースプレーン）となることが予想され，大気中を飛行する間は空気を利用する空気吸い込み型（エアブリージング）エンジンが用いられる [7]．本稿では，水平離陸後，地球低高度軌道へ飛行する，エアーブリージングエンジンを搭載したスペースプレーン型極超音速機の上昇経路最適化問題に対して繰り返し複合アルゴリズムを適用した場合を考察する．

スペースプレーンの経路最適化問題に対して，繰り返し動的計画法複合アルゴリズムを適用することで，本アルゴリズムの適応性を検証することができる．スペースプレーンもまた，基

本的には航空機と同じ運動方程式によって表現することができる．しかし，スペースプレーンは，機体速度がマッハ 25 付近まで及び，また機体性能を表す空力微係数 $C_{L\alpha}, C_{D0}$ やエンジン推力が高度の非線形性を持つ点で，一般の航空機よりも複雑な飛行性能をもつ．したがって，このような複雑な性質を持つスペースプレーンに対して，繰り返し複合アルゴリズムがシステマティックに最適経路を生成することができるならば，このことはアルゴリズムの性能評価と適応性をチェックできることになる．

## 5．1　スペースプレーンの上昇経路最適化問題

### (1)　原問題

スペースプレーンの運動方程式は，通常，地球の自転等による影響を考慮するが，それ以外は基本的に，通常の航空機の運動方程式と同じである．両者で異なるところは，機体の揚力や抗力に影響を及ぼす $C_L(\alpha, M), C_D(\alpha, M)$ などの空力特性とエンジンの推力特性 $T(h, v, \phi)$ に関する性能上の相違がある．それらの特性に基づくスペースプレーンの運動は複雑である [6]．

いま，水平離陸後，地球低高度軌道へ飛行 (投入) する空気吸い込み型エンジンを搭載したスペースプレーンの上昇経路最適化問題を，燃料消費最小化の評価指標の下で考える．スペースプレーンを質点と仮定し，図 5 のように，制御変数に

$\alpha(t)$: 時刻 $t$ における迎角

を，機体の状態を規定する状態変数に

$v(t)$: 時刻 $t$ における速度

$\gamma(t)$: 時刻 $t$ における経路角

$r(t)$: 時刻 $t$ における地球中心からの半径

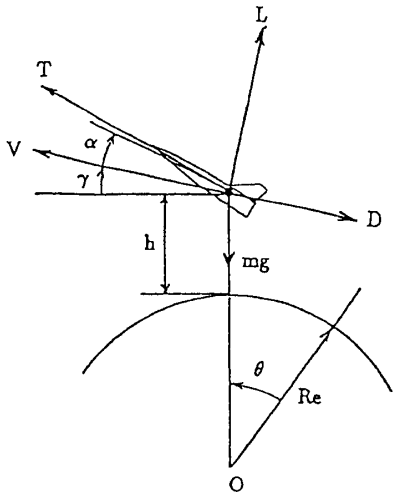


図 5　スペースプレーンの状態量の定義

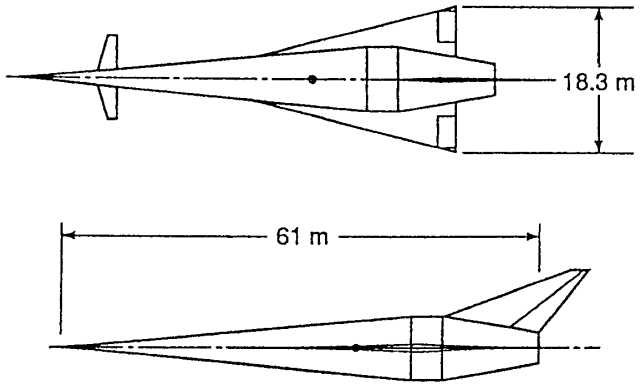


図 6　水平離陸機体モデル

$\theta(t)$: 時刻 $t$ における基準方向からの角度

$m(t)$: 時刻 $t$ における燃料を含む機体の質量

$h$: 機体の高度

を用いて機体上昇時の垂直面内での運動を考える．ただし，$h$ は機体の地上からの高度であり，$R_e$ を地球中心から地表までの半径とすると，$(h = r - R_e)$ で表す．スペースプレーンの最小燃料消費上昇経路問題は，

原問題 P:

$$\text{Minimize} \qquad J = \int_{t_0}^{t_f} |\dot{m}| dt \tag{30}$$

Subject to

$$\dot{v} = \frac{T(h,v,\phi)\cos\alpha - D(h,v,\alpha)}{m} - \frac{\mu \sin\gamma}{r^2} \tag{31}$$

$$\dot{\gamma} = \frac{T(h,v,\phi)\sin\alpha + L(h,v,\alpha)}{mv} + \left(\frac{v}{r} - \frac{\mu}{vr^2}\right)\cos\gamma \tag{32}$$

$$\dot{r} = v\sin\gamma \tag{33}$$

$$\dot{\theta} = \frac{v\cos\gamma}{r} \tag{34}$$

$$\dot{m} = -\frac{T(h,v,\phi)}{g_0 I_{sp}} \tag{35}$$

$$v \in V, \quad \gamma \in \Gamma, \quad r \in R, \quad \theta \in \Theta, \quad m \in W, \quad \alpha \in U$$

と書くことができる [6]．ここで，各記号は，$T(h,v,\phi)$：機体の推力 (表関数で所与)，$D(h,v,\alpha)$：抗力，$L(h,v,\alpha)$：揚力，$I_{sp}(h,v)$：比推力 (表関数)，$g_0$ :地表重力加速度 $(g = \mu/r^2)$，$\mu$：重力常数，$t$：時間，$\phi$：燃料等価比である．ただし，抗力と揚力は，関係式 $D(h,v,\alpha) = C_D(\alpha,M)\rho v^2 S/2$ および $L(h,v,\alpha) = C_L(\alpha,M)\rho v^2 S/2$ を用いて計算できるものと仮定する．ここで，$C_D(\alpha,M) = C_{D0}(M) + C_{L\alpha}(M)\alpha^2$：抗力係数 (表関数)，$C_L(\alpha,M) = C_{L0}(M) + C_{L\alpha}(M)\alpha$：揚力係数 (表関数)，$C_{L\alpha}(M)$：揚力傾斜，$\rho = \rho(h)$：空気密度 (表関数)，$M = v/a$：マッハ数，$S$：機体の基準面積，$a = a(h)$: 音速 (表関数) である．このとき，スペースプレーンの最小燃料消費上昇経路問題は，燃料消費 $J$ を最小化する迎角 $\alpha(t)$ の制御列を求める問題と解釈できる．

**(2)　機体モデル**

ここで用いるスペースプレーンの機体モデルは Shaughnessy ら [6] による機体を一部簡単化したものであり，形状を図 7 に示す．全備重量 136,080 kg，基準翼面積 335 $\mathrm{m}^2$，全長 61 m，エアーブリージング推進系の推力は，

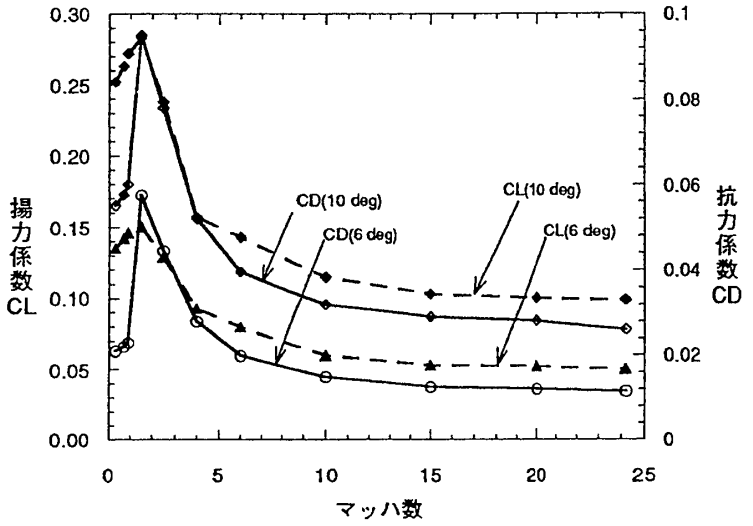


図 7　マッハ数に対する推力係数 $C_T$，揚力係数 $C_L$，抗力係数 $C_D$ および比推力 $I_{SP}$

$$T(h, v, \phi) = C_T q \tag{36}$$

によって与えられるものとした．$q$ は動圧，$C_T$ は推力係数である．$C_T$ と $I_{sp}$ はマッハ数，動圧，燃料等価比 $\phi$ の関数であり，揚力係数 $C_L$ と抗力係数 $C_D$ は，マッハ数と迎角の関数である．この機体モデルのマッハ数に対する推力係数 $C_T$，揚力係数 $C_L$，抗力係数 $C_D$ および比推力 $I_{SP}$ の関係を図 7 に示す．ただし標準大気モデルは JIS W-0201 に準拠しており，表関数で与えた [8]．

**(3)　制約条件**

軌道上の主要な制約として，動圧 $q$ と加熱比 $Q$ の上限，すなわち，

$$q = \frac{1}{2}\rho v^2 \leq q_{max} \tag{37}$$

$$Q = K\sqrt{\rho} v^3 \leq Q_{max} \tag{38}$$

を考える．ここで，$K$ は定数である．

## 5．2　下界値計算とエネルギー近似

いま，スペースプレーンの単位質量当りの総エネルギー $E$ が，その運動エネルギーと位置エネルギーの和，すなわち $E = \frac{1}{2}v^2 + gh = \frac{1}{2}v^2 + g(r - R_e)$ で表されると仮定して，$E$ を時間微分した結果に (31) と (33) 式を代入すると，$E$ の時間変化率は，

$$\dot{E} = \frac{v(T(h, v, \phi)\cos\alpha - D(h, v, \alpha))}{m} - \frac{2\mu h v \sin\gamma}{r^3} \equiv f_1 \tag{39}$$

と書くことができる．また，(35) と (39) 式より，

$$-dm = \frac{T(h,v,\phi)}{g_0 I_{sp} f_1} dE \tag{40}$$

となる．したがって，エネルギー近似を用いて目標高度と速度に燃料最小で到達するためには，(40) に対して各エネルギーレベルで，$g_0 I_{sp} f_1 / T(h,v,\phi)$ を最大化すればよい．(40) 式は未定の経路角 $\gamma$ を含んでいるが，$\gamma \geq 0$ ならば，(39) 式の第 2 項目を取り除くことにより，(40) 式はより小さな燃料消費を導く．また一般に，スペースプレーン上昇時には，軌道に沿って動圧の制約条件が課せられるため，この制限動圧曲線に沿った軌道の高度は単調増加となる．以上より，$\gamma \geq 0$ と仮定する．

(39) 式の第 1 項に対して，$\alpha = 0$ を代入すると，$T(h,v,\phi)\cos\alpha - D(h,v,\alpha)$ の値は，$T(h,v,\phi)\cos\alpha - D(h,v,\alpha)$ の下界値となる．また，厳密な $m$ の下界値を得るために，燃料を含めた機体質量 $m(t)$ の下限値の推定値，または，燃料を除く質量 $m_b$（一定）を利用する．これらの修正の下での緩和問題，

緩和問題 E：修正エネルギー近似

$$\text{Minimize} \quad J^M = \int_{t_0}^{t_f} |\dot{m}| dt \cong \int_{E_0}^{E_f} \frac{T(E,v,\phi)}{I_{sp} g_0 f_1} dE \tag{41}$$

の最適値は，原問題の評価関数の最適値を必ず下回る．すなわち，与えられた初期点から終端点への最小燃料上昇経路は，初期点のエネルギー高度 $E_0$ から終端点のエネルギー高度 $E_f$ までの各等エネルギー線上で $I_{sp} g_0 f_1 / T(E,v,\phi)$ を最大化する $E$ 毎の $v$ によって与えられる．修正エネルギー近似では，隣接する各エネルギー高度間の最小燃料消費で，そのエネルギー高度間の燃料消費を代表させる．初期点から終端点までの燃料消費 $J$ の下界値は，航空機の通過区間に対応する最小燃料消費の総和で表す．ただし，これらの計算は，動圧 $q$ と加熱比 $Q$ の制約条件を満たす $(h,v)$ 平面上で行う．

### 5．3　数値例

水平離陸時の初期条件，終端条件，および，軌道上での動圧制限を，それぞれ，

$$h(t_0) = 0\,\text{km}, \quad v(t_0) = 0.5\,\text{Mach}, \quad \gamma(t_0) = 0$$
$$h(t_f) = 80\,\text{km}, \quad v(t_f) = 24\,\text{Mach}, \quad \gamma(t_f) = \text{無指定}.$$

$$q \leq q_{max} = 9{,}766\,\text{kg/m}^2 (2{,}000\text{lb/ft}^2) \tag{42}$$

とする．動圧制約をこのように設定すると，スペースプレーンの実行可能な経路は，高度と速度の 2 次元平面 $(h,v)$ 上に引かれたこの動圧制約曲線よりも上側に存在しなければならないこ

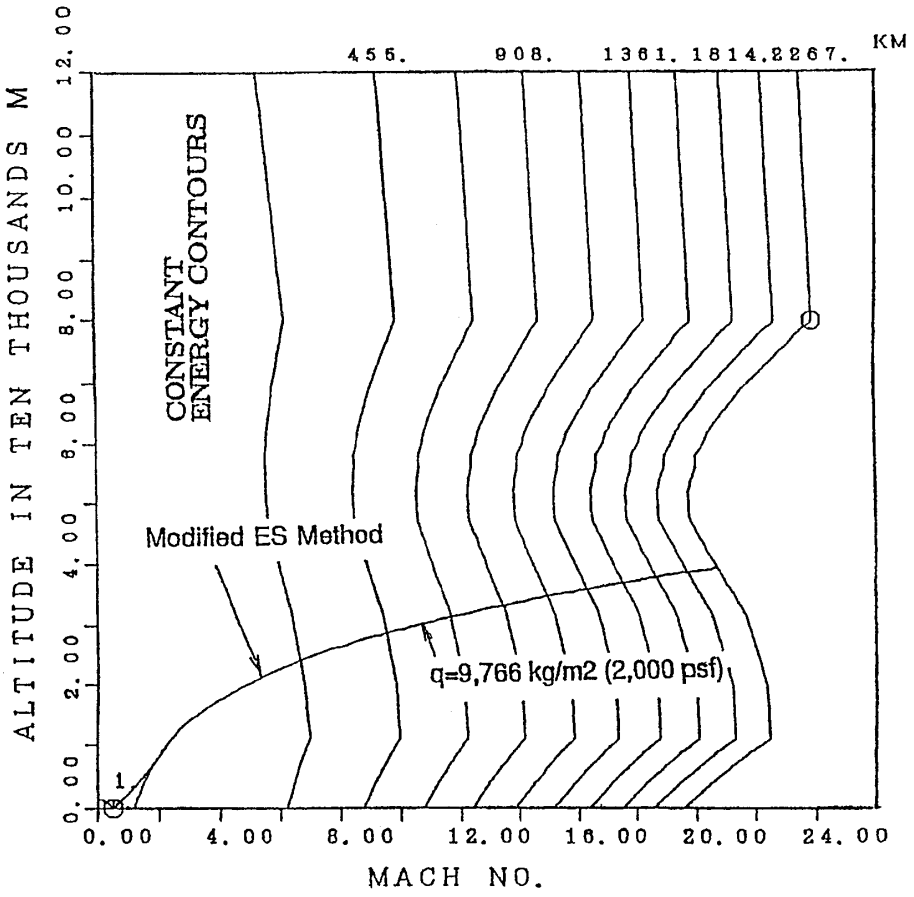


図8　下界値を与える修正エネルギー近似法による最小燃料消費経路

とになる．また，この数値例では，終端条件としてのマッハ数が 24 Mach，終端高度が 80 km と設定しているため，高度の非線形性を運動方程式に含む問題となる．

基本形複合アルゴリズムによって必要となる下界値を修正エネルギー近似法によって，求めるために，(42) 式の動圧制限下で緩和問題 E を解いた．この解は，2 次元平面 $(h, v)$ 上での関数の評価と比較によって求めることができるが，20 MIPS の計算機で 0.8 sec 必要であった．このとき得られた修正エネルギー近似解は図 8 に示すように動圧制限線にぴったりと沿った形となった．次に，基本形複合アルゴリズムを適用するため，状態空間を，$h, v, \gamma, m$ に関して，それぞれ 64, 32, 32, 8 個のブロックに，また，独立変数を $E$ に関して 20 レベルに量子化した．$\theta$ は他の状態変数の変化に影響を与えず，個別に計算することが可能であるため，量子化を行わなかった．さらに，制御変数 $\alpha$ を 13 レベルに量子化した．この量子化数を用いた場合，従来の動的計画法では，約 10,000,000 点の格子点でのコスト関数の評価と約 130,000,000 回のコスト関数の計算が必要となる．

最適値の上界値は，修正エネルギー近似解を参考にして決める．このとき，より粗い離散化の下で基本形複合アルゴリズムを数回試行的に適用すれば，上界値の範囲を絞ることが可能で

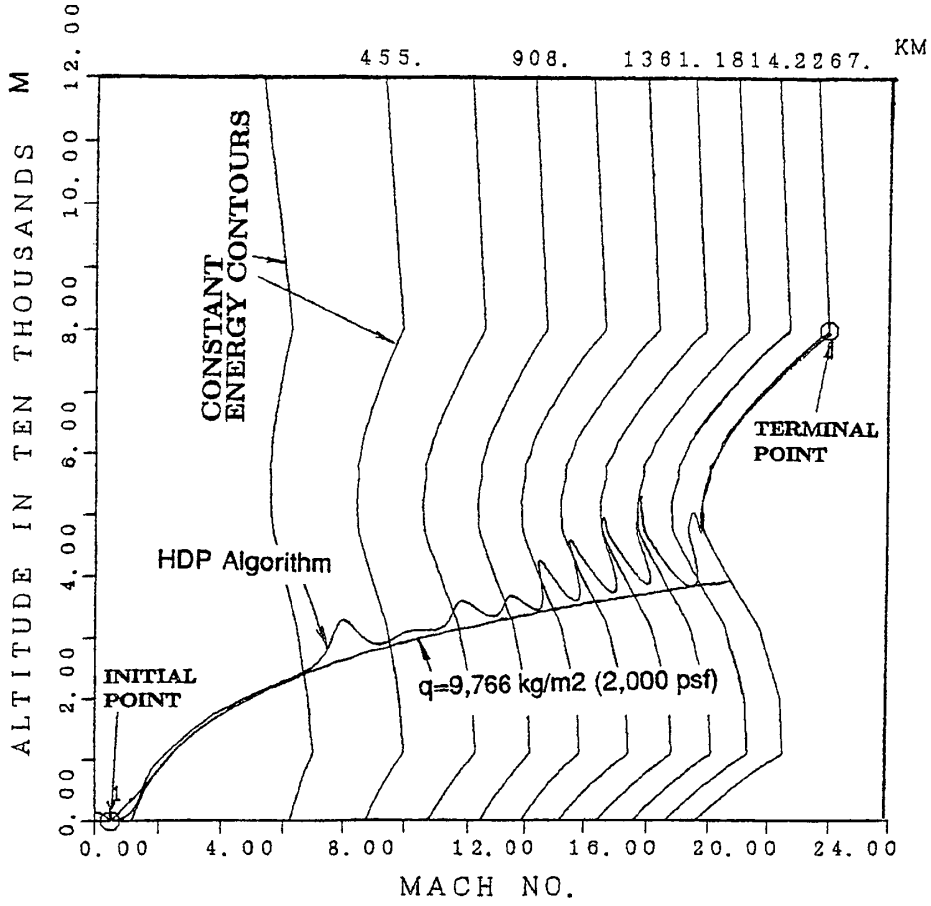


図 9　繰り返し複合アルゴリズムによる最小燃料上昇経路 ($i = 1$)

ある．得られた上下界値の下で，また得られた基本形複合アルゴリズムによる解を初期値とし，$i = 2$ 以降，繰り返し複合アルゴリズムによって解を逐次近似し，解の精度を改良した．繰り返し複合アルゴリズムの初回の実行 ($i = 1$) によって得られた最小燃料消費経路を図 9 に示す．この図より，最適経路は動圧制約曲線の上側に存在し，マッハ 0.5 からマッハ 6 付近までは最小燃料消費上昇経路は動圧制約曲線に一致し，それ以降はかなり複雑に波を打っていることがわかる．一方，8 回 ($i = 8$) の繰り返しで得られた最小燃料消費経路を図 10 に示す．図 10 では，マッハ 0.5 からマッハ 18 付近まで最小燃料消費上昇経路は動圧制約曲線に一致していることがわかる．それ以降は，高度 80 km の目標終端点に到達するために動圧制約曲線から徐々に離れ，最終段では終端点の位置するエネルギー線上を急上昇している．これらより，初回 ($i = 1$) の計算で得られた最適経路はかなり複雑に曲がりくねった劣悪な解となっているが，繰り返し複合アルゴリズムを用いることによって，スムーズな最適経路へ変化していくことがわかる．これは，繰り返し複合アルゴリズムが各繰り返し過程で，前回の経路を参照軌道とするのではなく，前回の経路は上下界値の計算にのみ利用しているためである．さらに，繰り返し複合アルゴリズムのコスト計算が，前回の最適経路の近傍に限定されないことを意味している．この問題に

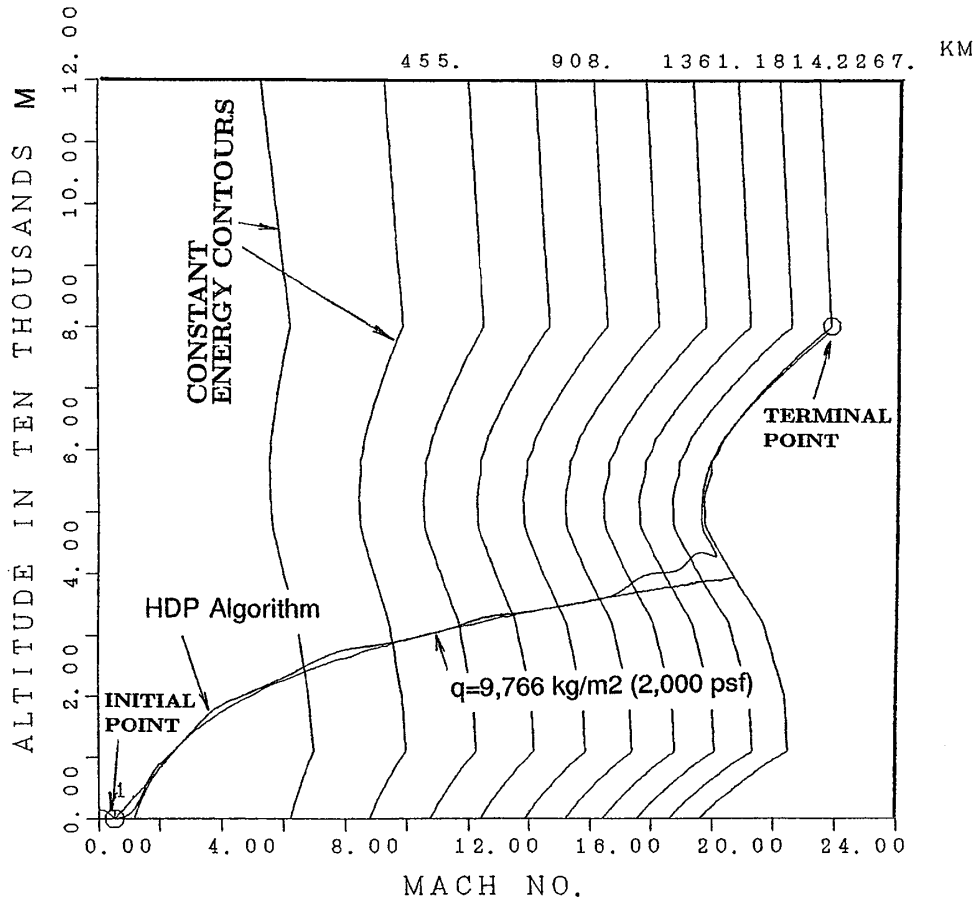


図 10　繰り返し複合アルゴリズムによる最小燃料上昇経路 ($i = 8$)

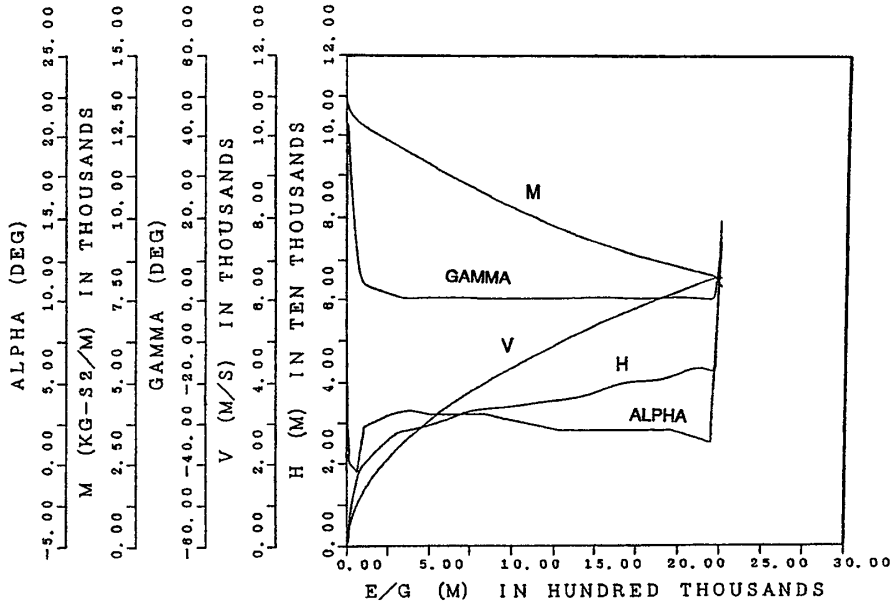


図 11　最小燃料上昇経路 ($i = 8$) における状態変数と制御変数の履歴

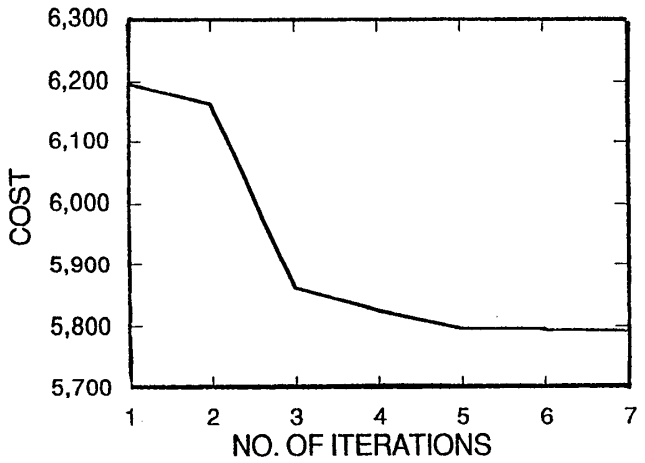


図 12　イテレーションに対するコストの収束

対する最小燃料上昇経路 ($i = 8$) における状態変数と制御変数の履歴を図 11 に示す．また，各イテレーションに対するコストの収束の様子を図 12 に示す．この図より，コスト値はイテレーション数が $i = 3$ 回までに急速に減少した後，徐々に減少し，$i = 5$ 回付近でほぼ一定値となり収束していることが読みとれる．

コスト関数が評価された実行空間を図 13 の下端より 4 つの線図中 ($i = 1, i = 2, i = 7$ および $i = 8$) に示す．ただし，同図中，上端より 2 つの線図 ($i = 1, i = 8$) は最小燃料上昇経路であり参考のために示す．これらの図より，繰り返し複合アルゴリズムによる実行可能経路は正確に動圧制約条件 (動圧制約曲線にて表示) を満たしていることがわかる．

繰り返し複合アルゴリズムの初回 ($i = 1$) の計算時間は，修正エネルギー近似法での下界値計算の 0.8 sec を含め，59 sec であった．複合アルゴリズムの従来の動的計画法に対する計算領域の比は 1,000 分の 1 以下，また，コスト関数の計算回数の比は 2,000 分の 1 以下であった．

以上より，繰り返し複合アルゴリズムを用いることによって，スペースプレーンの最適経路問題における最適解の近似解と対応する最適コストを確実に，統一的に求めることが可能であることがわかった．したがって，繰り返し複合アルゴリズムは航空機の最適経路問題の解法において適応性を持つことが確認できた．

## 6　まとめ

今回提案した並列繰り返し型動的計画法複合アルゴリズムでは，動的計画法と分枝限定法を併用した基本複合アルゴリズムに対し，計算上の工夫として，コストの下界値を精度よく推定する繰り返し論理と，局所解を避ける並列処理を組み入れている．それらによって，数値計算上の負荷（計算数とサイズ）を大幅に削減し，局所解を回避できることが判明した．また，本法は，スペースプレーンの上昇経路最適化問題の解を，確実に，かつ統一的に求めることがで

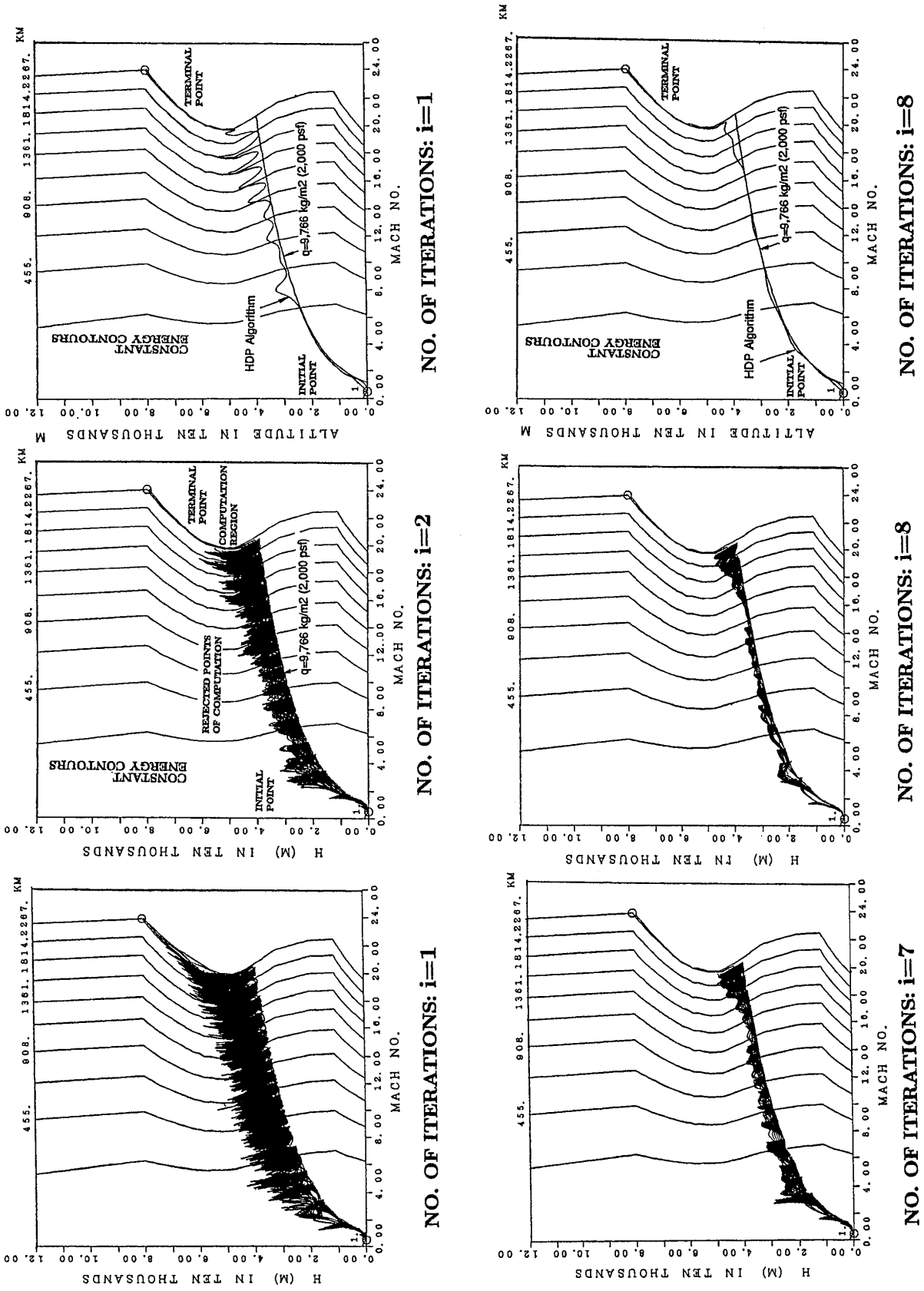


図 13　コスト関数が評価された実行空間

きることが判明した.

## 参考文献

[1] R.E. Bellman: Dynamic Programming, Princeton University Press, Princeton, N.J. (1957)

[2] O.G. Alekseev and I.F. Volodos: Combined Use of Dynamic Programming and Branch-and-Bound Methods in Discrete-Programming Problems, Automation and Remote Control, 37, 557/565 (1976)

[3] T.L. Morin and R.E. Marsten: Branch-and-Bound Strategies for Dynamic Programming. Opns. Res., 24, 611/627 (1976)

[4] T. Hanaoka and T. Tanabe: A New Dynamic Programming Algorithm and its Application to Optimal Reentry Problems, Proc. of 13th Int'l Symp. Space Tech. Sci., 1031/1036 (1982)

[5] T. Hanaoka: A Lower Bound Computation Method Using Energy-State Approximation and Its Application to Supersonic Aircraft Shortest Path Problems, Journal of the Operations Research of Japan, 41-2, 289/310 (1998)

[6] J.D. Shaughnessy, S.Z. Pinckney, J.D. McMinn and M.-L. Kelley: Hypersonic Vehicle Simulation Model: Winged-Cone Configulation, NASA TM-102610 (1990)

[7] 航空宇宙工学便覧: pp.817 (1992)

[8] 日本工業標準調査会: JIS W-0201, 日本規格協会 (1979)