並列化補外法による重力少体問題の高速計算

伊藤孝士・福島登志夫（国立天文台三鷹）
tito@cc.nao.ac.jp

全地球史解読計画『とけい班』の精神的基礎を成す概念のひとつとして、地球の軌道運動が全地球史的時間スケールで安定であったという仮定がある。私達はこの仮定の妥当性を検証するために、超長期にわたる惑星運動の数値積分を行っている。本稿では、こうした数値計算を高速に実行するための一方法として、補外法を並列化して高速化する方法を提案する。天体数Nが少ない場合の重力相互作用の数値計算はベクトル化や並列化には極めて不向きであると考えられて来たが、補外法の並列化可能性に着目することにより、平易な算法のみで計算速度の大幅な向上を実現することができた (Ito and Fukushima, 1997)。

1. 補外法とその並列化

1.1 補外法とは

常微分方程式の初期値問題の数値解法では、刻み幅hを0に近づけるに従って計算機上の仮想的な系が現実の物理系に近づいて行く。このことを利用し、いくつかの有限のhに対して得られた解を多項式近似を用いて組み合わせて$h \to 0$の極限を補外するのが補外法と呼ばれる数値積分法である (Gragg, 1965; Bulirsch and Stoer, 1966) この方法は従来の数値積分法に比べて極めて精度が良いが、その代わりに長い計算時間が必要である (Murison, 1989)。けれども良く考えると、補外法で使用される各hに対する計算はすべて独立であり、干渉や通信を行うことなく別個に計算できる (Hairer *et al.*, 1993; Fukushima, 1996)。つまり、補外法は極めて並列化しやすい算法だということである。

まずは補外法の一般論を概説する。一次の常微分方程式の初期値問題は一般に以下のように書き下すことができる。

$$\frac{dy}{dt} = f(y,t), \quad y(t_0) = y_0. \tag{1}$$

補外法はこの問題を以下の二段階を経由して解く方法である。

1. 段数pと基本刻み幅Hを決め、刻み幅h_i $(i = 1, 2, \ldots, p)$ を$h_i = H/2n_i$として定める。n_iの決め方は問題に依存する。何らかの適当な方法を用い、微分方程式 (1) を刻み幅h_iで数値積分し、対応する解Y_iを得る。

2. Y_iとh_i $(i = 1, 2, \ldots, p)$ から、ある種の補外法（多項式補外や有理関数補外）を用いて$h_i \to 0$の場合の真の解Y_∞を補外する。

1. に於ける数値積分法としては、${h_i}^2$の収束性を持つ修正中点法がしばしば用いられる。h_iでの解$Y_i \equiv y(t_0 + H) \quad (i = 1, 2, \ldots, p)$ を求める方法は以下のように記述できる。

$$f_0 := f(y_0, t_0); \quad y_1 := y_0 + h_i f_0, \tag{2}$$

$$f_j := f(y_j, t_j); \quad y_{j+1} := y_{j-1} + 2h_i f_j, \quad (j = 1, \ldots, 2n_i - 1) \tag{3}$$

$$f_{2n_i} := f(y_{2n_i}, t_{2n_i}); \quad Y_i := \frac{1}{2}[y_{2n_i} + (y_{2n_i-1} + h_i f_{2n_i})], \tag{4}$$

ここで $t_j \equiv t_0 + jh_i$である。n_iの選択に関してはいわゆる harmonic sequence を採用する。

$$n_i = 1, 2, 3, 4, \dots, \tag{5}$$

次には解 Y_i らの補外であるが、ここでは Aitken-Neville らによる多項式補外を用いる (Aitken, 1932; Neville, 1934)。有理関数補外でも得られる結果に大差はないが、問題の種類に依存することは当然である。実際の補外は表 1の順に行われ、$T_{j,k+1}$ と $T_{j,k}$の関係は以下で表される。

$$T_{j,k+1} = T_{j,k} + \frac{T_{j,k} - T_{j-1,k}}{(h_j/h_{j-k})^2 - 1} \tag{6}$$

表 1. 補外法で使用される補外表。$T_{i,1}$は式 (4) の Y_iに等しく、式 (6) の関係を用いて $T_{p,p}$の方向に補外が進む。

$$\begin{array}{llllll}
Y_1 \equiv T_{1,1} & & & & & \\
 & \searrow & & & & \\
Y_2 \equiv T_{2,1} & \rightarrow & T_{2,2} & & & \\
 & \searrow & & \searrow & & \\
Y_3 \equiv T_{3,1} & \rightarrow & T_{3,2} & \rightarrow & T_{3,3} & \\
 & \searrow & & \searrow & & \searrow \\
\vdots & & \vdots & & \vdots & \quad \vdots \\
 & \searrow & & \searrow & & \searrow \quad\quad \searrow \\
Y_p \equiv T_{p,1} & \rightarrow & T_{p,2} & \rightarrow & \cdots & \rightarrow T_{p,p-1} \rightarrow T_{p,p}
\end{array}$$

1.2 並列化

補外の段数を p とした場合、数値積分の刻み幅は偶数系列 $(1/2p)$ で p 段取るのが普通である (Deuflhard, 1983)。この場合、各段の計算量は p に比例するが、単純に p 段目を p 番目のプロセッサに割り当てたのでは負荷バランスが悪くなり、並列効率が低下する。そのため私達は i 段目と $p - i$ 段目を組み合わせて同一のプロセッサ上で計算させ、各プロセッサの負担を均一化する工夫をした。例えば $p = 8$ の場合には、Y_1 と Y_8をプロセッサ 1 が計算し、Y_2 と Y_7をプロセッサ 2 が計算し、$\cdots$ という具合である。

$$\begin{array}{lcll}
Y_1, Y_8 & \cdots & \text{プロセッサ 1} & (\text{負荷 } 1 + 8 = 9) \\
Y_2, Y_7 & \cdots & \text{プロセッサ 2} & (\text{負荷 } 2 + 7 = 9) \\
Y_3, Y_6 & \cdots & \text{プロセッサ 3} & (\text{負荷 } 3 + 6 = 9) \\
Y_4, Y_5 & \cdots & \text{プロセッサ 4} & (\text{負荷 } 4 + 5 = 9)
\end{array} \tag{7}$$

これによって負荷バランスは極めて向上する。同様な議論は一般の p 段の場合にも適用できる。表 2には $p = 10$ までの負荷分散の状況を示した。

表 2. 補外の段数 p を変化させた場合に n_p 個のプロセッサに与えられる負荷の最適化分散。

	n_p (number of processors)									
i	1	2	3	4	5	6	7	8	9	10
$[p=1]$										
1	Y_1									
$[p=2]$										
1	Y_1, Y_2	Y_1								
2	—	Y_2								
$[p=3]$										
1	$Y_1, \cdots, Y_3$	Y_1, Y_2	Y_1							
2	—	Y_3	Y_2							
3	—	—	Y_3							
$[p=4]$										
1	$Y_1, \cdots, Y_4$	Y_1, Y_4	Y_1, Y_2	Y_1						
2	—	Y_2, Y_3	Y_3	Y_2						
3	—	—	Y_4	Y_3						
4	—	—	—	Y_4						
$[p=5]$										
1	$Y_1, \cdots, Y_5$	Y_1, Y_3, Y_4	Y_1, Y_4	Y_1, Y_4	Y_1					
2	—	Y_2, Y_5	Y_2, Y_3	Y_2	Y_2					
3	—	—	Y_5	Y_3	Y_3					
4	—	—	—	Y_5	Y_4					
5	—	—	—	—	Y_5					
$[p=6]$										
1	$Y_1, \cdots, Y_6$	Y_1, Y_2, Y_3, Y_4	Y_1, Y_6	Y_1, Y_2, Y_3	Y_1, Y_2	Y_1				
2	—	Y_5, Y_6	Y_2, Y_5	Y_4	Y_3	Y_2				
3	—	—	Y_3, Y_4	Y_5	Y_4	Y_3				
4	—	—	—	Y_6	Y_5	Y_4				
5	—	—	—	—	Y_6	Y_5				
6	—	—	—	—	—	Y_6				
$[p=7]$										
1	$Y_1, \cdots, Y_7$	Y_1, Y_6, Y_7	Y_1, Y_3, Y_5	Y_1, Y_6	Y_1, Y_2	Y_1, Y_2	Y_1			
2	—	Y_2, Y_3, Y_4, Y_5	Y_2, Y_7	Y_2, Y_5	Y_3, Y_4	Y_3	Y_2			
3	—	—	Y_4, Y_6	Y_3, Y_4	Y_5	Y_4	Y_3			
4	—	—	—	Y_7	Y_6	Y_5	Y_4			
5	—	—	—	—	Y_7	Y_6	Y_5			
6	—	—	—	—	—	Y_7	Y_6			
7	—	—	—	—	—	—	Y_7			
$[p=8]$										
1	$Y_1, \cdots, Y_8$	Y_1, Y_2, Y_7, Y_8	Y_1, Y_3, Y_8	Y_1, Y_8	Y_1, Y_6	Y_1, Y_4	Y_1, Y_2	Y_1		
2	—	Y_3, Y_4, Y_5, Y_6	Y_2, Y_4, Y_6	Y_2, Y_7	Y_2, Y_5	Y_2, Y_3	Y_3	Y_2		
3	—	—	Y_5, Y_7	Y_3, Y_6	Y_3, Y_4	Y_5	Y_4	Y_3		
4	—	—	—	Y_4, Y_5	Y_7	Y_6	Y_5	Y_4		
5	—	—	—	—	Y_8	Y_7	Y_6	Y_5		
6	—	—	—	—	—	Y_8	Y_7	Y_6		
7	—	—	—	—	—	—	Y_8	Y_7		
8	—	—	—	—	—	—	—	Y_8		
$[p=9]$										
1	$Y_1, \cdots, Y_9$	$Y_1, \cdots, Y_4, Y_6, Y_7$	$Y_1, \cdots, Y_5$	Y_1, Y_2, Y_9	Y_1, Y_8	Y_1, Y_8	Y_1, Y_8	Y_1, Y_8	Y_1	
2	—	Y_5, Y_8, Y_9	Y_6, Y_9	Y_3, Y_8	Y_2, Y_7	Y_2, Y_7	Y_2, Y_7	Y_2	Y_2	
3	—	—	Y_7, Y_8	Y_4, Y_7	Y_3, Y_6	Y_3, Y_6	Y_3	Y_3	Y_3	
4	—	—	—	Y_5, Y_6	Y_4, Y_5	Y_4	Y_4	Y_4	Y_4	
5	—	—	—	—	Y_9	Y_5	Y_5	Y_5	Y_5	
6	—	—	—	—	—	Y_9	Y_6	Y_6	Y_6	
7	—	—	—	—	—	—	Y_9	Y_7	Y_7	
8	—	—	—	—	—	—	—	Y_9	Y_8	
9	—	—	—	—	—	—	—	—	Y_9	
$[p=10]$										
1	$Y_1, \cdots, Y_{10}$	$Y_1, \cdots, Y_4, Y_7, Y_{10}$	$Y_1, \cdots, Y_4, Y_9$	Y_1, Y_2, Y_3, Y_7	Y_1, Y_{10}	Y_1, Y_9	Y_1, Y_9	Y_1, Y_9	Y_1, Y_9	Y_1
2	—	Y_5, Y_6, Y_8, Y_9	Y_5, Y_6, Y_7	Y_4, Y_{10}	Y_2, Y_9	Y_2, Y_8	Y_2, Y_8	Y_2, Y_8	Y_2	Y_2
3	—	—	Y_8, Y_{10}	Y_5, Y_9	Y_3, Y_8	Y_3, Y_7	Y_3, Y_7	Y_3	Y_3	Y_3
4	—	—	—	Y_6, Y_8	Y_4, Y_7	Y_4, Y_6	Y_4	Y_4	Y_4	Y_4
5	—	—	—	—	Y_5, Y_6	Y_5	Y_5	Y_5	Y_5	Y_5
6	—	—	—	—	—	Y_{10}	Y_6	Y_6	Y_6	Y_6
7	—	—	—	—	—	—	Y_{10}	Y_7	Y_7	Y_7
8	—	—	—	—	—	—	—	Y_{10}	Y_8	Y_8
9	—	—	—	—	—	—	—	—	Y_{10}	Y_9
10	—	—	—	—	—	—	—	—	—	Y_{10}

2.　数値計算例

2.1　基本刻み幅 H を固定した場合

実際の並列計算には国立天文台天文学データ解析計算センターのスーパーコンピュータシステム VPP300/16R（富士通）を使用した (伊藤, 1996)。言語処理系はデータパラレル型の Fortran90/VPP を用い、手続き分割による並列化を行った。この方法により、天体数 N が 5 から 10 という少数体系の進化を追う計算に於いて、4 プロセッサを使用した並列計算で 3 倍以上の高速化という高い並列化効率が実現された。基本刻み幅 $H(\equiv n \times h)$ と段数 p を固定した場合の計算結果を図 1として掲載している。

並列計算に於ける効率向上のためのもうひとつのポイントは各プロセッサ間でのデータ転送である (van der Houwen, 1993)。容易に予測されるように、一度に転送されるデータの量が多いほど並列化の効率は高くなる。図 1に示したのと同様な計算を天体数 N を変えて行い、その場合の並列化効率の変化を示したのが図 2である。予想通り、N が大きい場合すなわち一度に転送されるべきデータが大きい方が効率の良い並列化が行われていることがわかる。

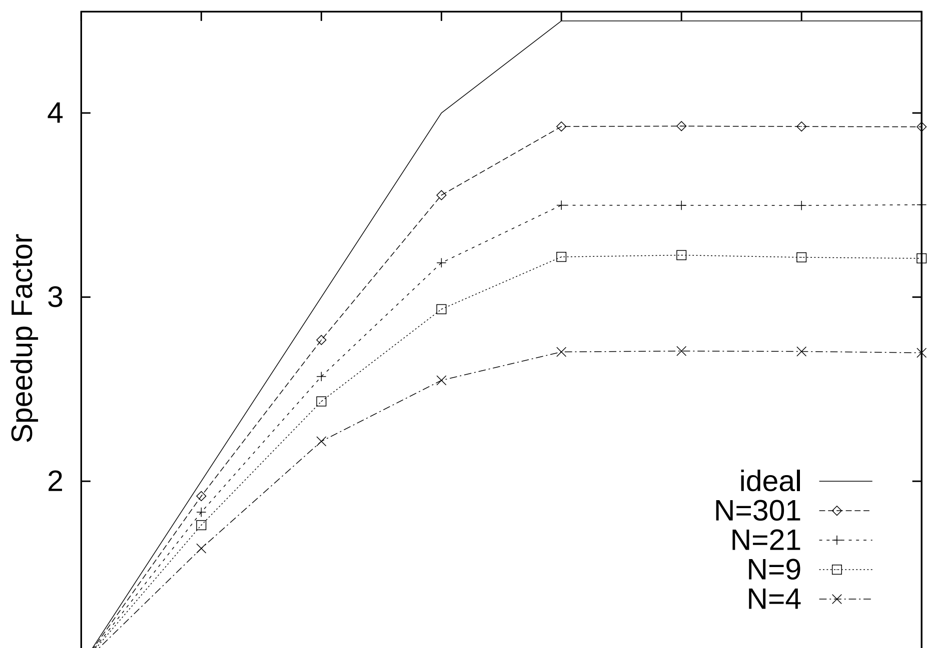

図 1. 並列化した補外法による高速化の一例。基本刻み幅と段数と固定したもの。横軸が使用したプロセッサ数、縦軸は非並列計算に対する高速化の比率である。ここでは $p = 8$ とした。N は天体の数を表す。理想的な並列化がなされた場合の高速化率も実線 (ideal) で併記されている。

2.2　基本刻み幅 H が変化する場合

長い楕円軌道上を運動するような天体の運動を扱う場合には、基本刻み幅 $H(\equiv n \times h)$ を可変にした数値積分を行う必要がある。このような場合にも前述した方法は高い並列化効率を実現する

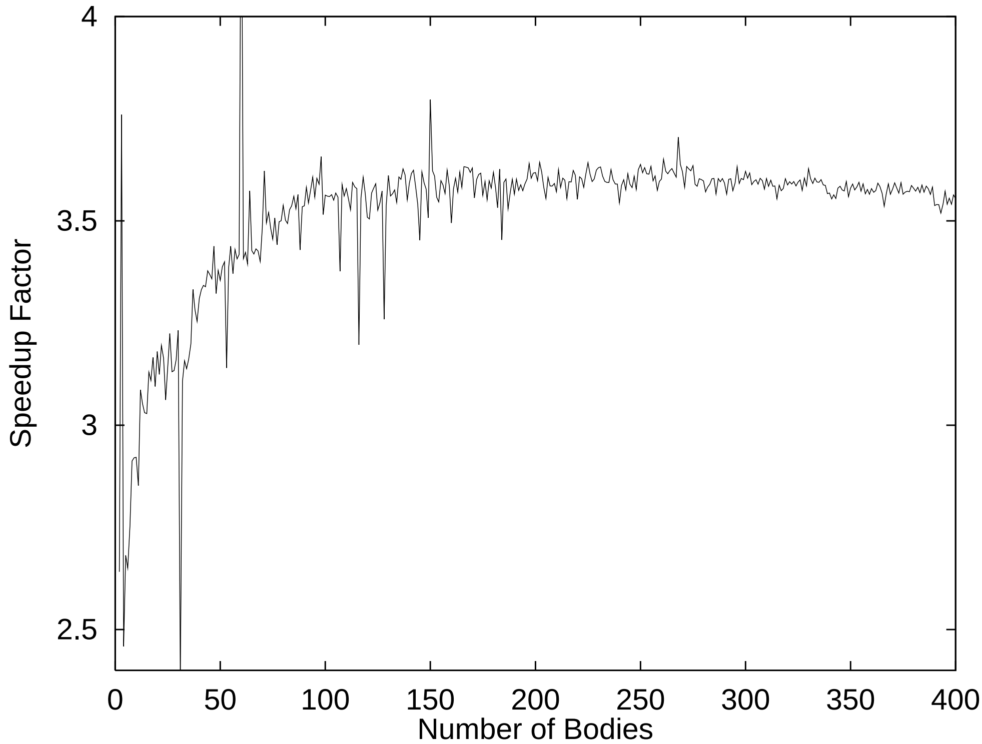

図 2. 並列化効率のサイズ N 依存性。横軸が天体数 N で、縦軸が四並列計算での並列化による速度向上率を示した。

ことがわかった。基本刻み幅 H を変化させる方法は多くあるが (Gear, 1971; Montenbruck, 1992; Press *et al.*, 1988)、私達は以下のような方式で新しい刻み幅 H' を決定した。

$$H' = H\epsilon_p^{\frac{-1}{2p-1}}, \tag{8}$$

ここで

$$\begin{aligned} \epsilon_p &= \|T_{p,p-1} - T_{p,p}\| \\ &\equiv \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{T^i_{p,p-1} - T^i_{p,p}}{s}\right)^2} \end{aligned} \tag{9}$$

である。$T_{p,p}$ と $T_{p,p-1}$ は補外表 (表 1) 中の最終行の値である。s は許容誤差であり、ここでは $s = 1.0 \times 10^{-14}$ とした。

これを用いて計算効率を測定した例が図 3 である。図 3 と同様な数値積分を行った結果だが、基本刻み幅 H を固定した場合とほとんど同様に効率的な並列化がなされていることが見てとれる。

2.3 段数 p も変化する場合

惑星との近接遭遇を起こすような天体の場合には、刻み幅のみならず段数 p 自体をも可変にして、振動的に硬い運動に対応する必要がある。可変段数方式の補外法は既に数多く提唱されているが (Deuflhard, 1983)、これを並列化する際にはひとつの問題が発生する。使用するプロセッサ数があらかじめ確定している並列化の場合には、使用する段数がプロセッサ数より少なくなった

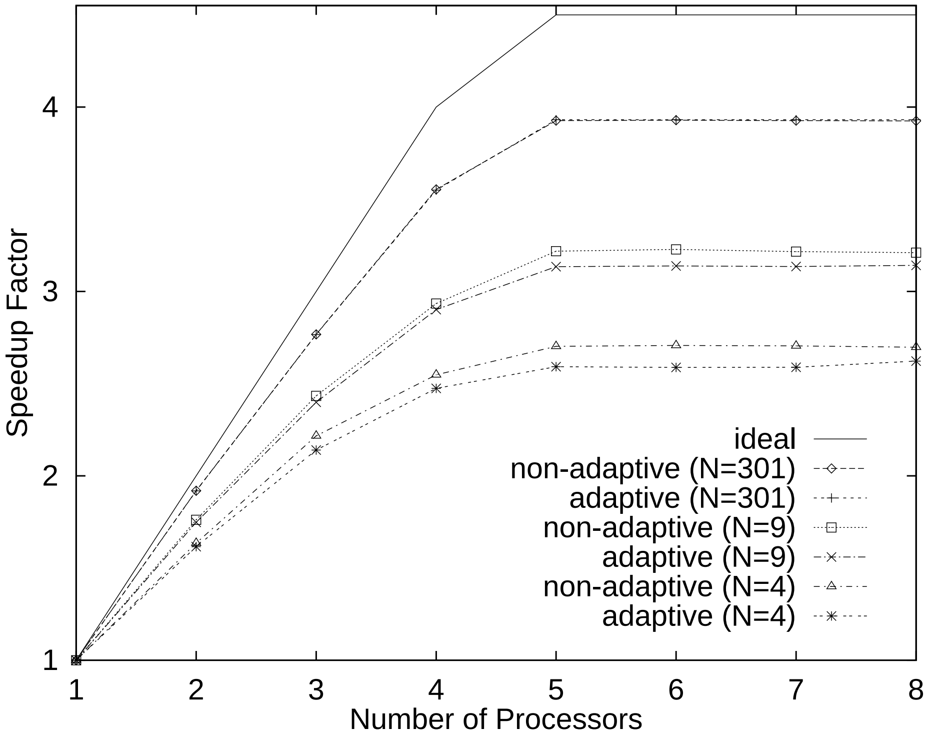

図 3. 基本刻み幅 H を可変にした場合の並列化効率。計算の詳細は図 1 と同一。

段階でプロセッサの稼働率が落ち、実質的な並列化効率が落ちてしまうのである。このため私達は従来の可変段数方式を改良し、使用する段数を固定する代わりに基本刻み幅の分割数を変更して可変段数をエミュレートするという算法を開発し、実装した。

まず、解くべき微分方程式の右辺の関数評価回数を A_p で定義し、一ステップあたりの計算量を W_p で表すことにする。

$$A_1 = n_1 + 1, \quad A_p = A_{p-1} + n_p, \tag{10}$$

$$W_p = \frac{A_p}{H_p}. \tag{11}$$

ここで H_p は 式 (8) の H' と同一のものである。

私達の方法では、ある時刻での解を得た時点で $\epsilon_p < 1$ であれば補外表の中の $T_{p,p}$ を解として受け入れ、次のステップに進む。次のステップでの段数と基本刻み幅は以下のように定める。

$$p_{new} = \begin{cases} p-1 & (W_{p-1} < 0.9 W_p \text{ の場合}) \\ p+1 & (W_p < 0.9 W_{p-1} \text{ の場合}) \\ p, & (\text{その他の場合}) \end{cases} \tag{12}$$

$$H_{new} = \begin{cases} H_{p_{new}} & (p_{new} \leq p \text{ の場合}) \\ H_{p_{new}} \left(\dfrac{A_{p-1}}{A_p} \right). & (p_{new} = p+1 \text{ の場合}) \end{cases} \tag{13}$$

一方、$\epsilon_p \geq 1$ であるかまたは $p > p_{max}$ であれば補外は収束しなかったものと看做し、新たなる段数と基本刻み幅

$$p_{new} = p_{basic}, \quad H_{new} = \beta H_{p_{old}}. \tag{14}$$

をもって計算をやり直す。$\beta(<1)$ は刻み幅の減数係数である。

並列計算用の工夫はここからである。私達の方法では基本刻み幅 H を p 個に分割はするものの、補外に使用する実際の段数は p_{basic}個で固定とする。系の時間発展を追う中で高い精度での計算が要求される段階が来たら、p_{basic}個の小さい刻み幅系列での積分結果を用いた補外を行う。例えば $p_{basic} = 8, p = 10$ の場合には、刻み幅 $H/1$ と $H/2$ については計算をせず、$H/3$ から $H/10$ についてのみ計算を行い、これらを補外して最終的な解を得る。H/p が大きな領域での解は実質的に補外後の解にあまり影響を与えないので、この方法による精度の低下はほとんどない。また、p_{basic}を固定することで使用するプロセッサ数をあらかじめ決めることができ、並列化効率の劣化もない。

この方法を実装し、極めて離心率の大きな彗星の運動を数値積分した場合の並列化効率を図示したのが図 4, 5である。若干のばらつきはあるもの、やはり効果的に並列化された計算が実行されていることがわかる。全エネルギーや全角運動量の保存状況から、計算の精度も従来の非並列な方法とほぼ等しい水準に保たれていることがわかった。

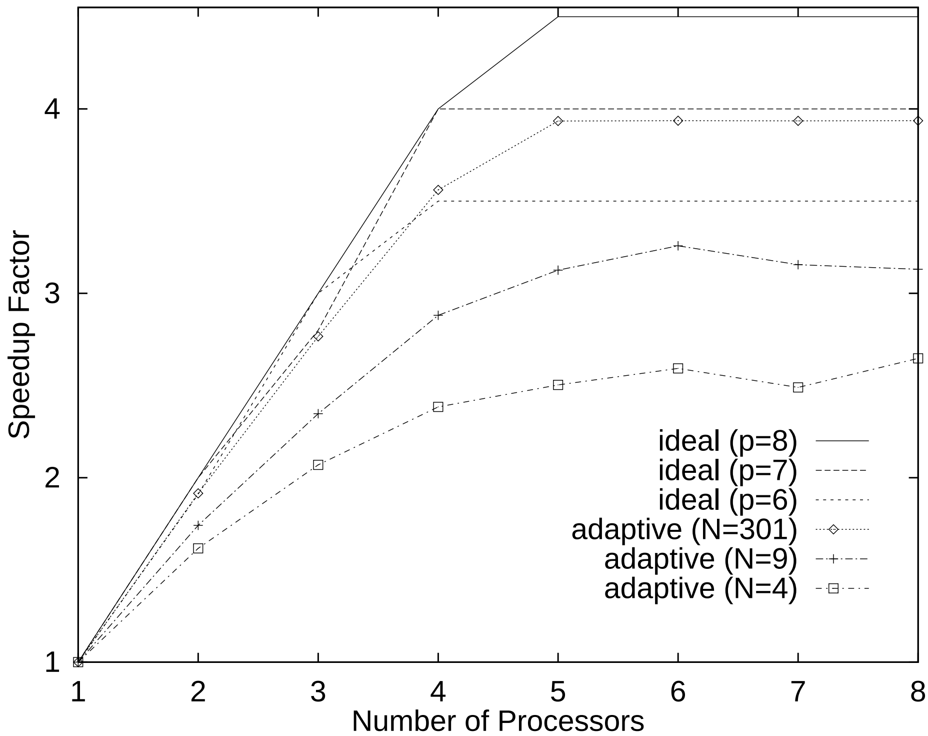

図 4. 基本刻み幅 H と段数 p を可変にした場合の計算の並列化効率。軸は図 1と同一。

3.　まとめ

ここに記した方法は天体数 Nが多い場合にはもちろん有効にベクトル化され得るものである。もちろん、天体数 Nが小さい場合の高速化も上述した方法により保証されている。並列計算機が次第に普及し (岩崎ほか, 1996; Ito, 1997)、PC 上でも簡単に並列計算コードの実装が可能になりつつある現在 (野澤, 1998)、重力少体問題の解法を並列化することの意義はますます大きくなって行くことであろう。私達の研究はその端緒を形成するものである。

参考文献

Aitken, A.C. (1932) On interpolation by iteration of proportional parts, without the use of differences, *Proc. Edinburgh Math. Soc. Second ser.*, **3**, 56–76.

Bulirsch, R. and Stoer, J. (1966) Numerical treatment of ordinary differntial equations by extrapolation methods, *Num. Math.*, **8**, 1–13.

Deuflhard, P. (1983) Order and stepsize control in extrapolation methods, *Numer. Math.*, **41**, 399–422.

Fukushima, T. (1996) Reduction of round-off errors in the extrapolation methods and its application to the integration of orbital motion, *Astron. J.*, **112**, 1298–1301.

Gear, C.W. (1971) *Numerical initial value problems in ordinary differencial equations*, Prentice-Hall, Englewood Cliffs, New Jersey.

Gragg, W.B. (1965) On extrapolation algorithms for ordinary initial value problems, *SIAM J. Numer. Anal.*, **2**, 384–403.

Hairer, E., Nørsett, S.P., and Wanner, G. (1993) *Solving Ordinary Differential Equation I – Nonstiff Problems* (second revised edition), Springer-Verlag, New York.

Ito, T. (1997) VPP300 series in National Astronomical Observatory, *FUJITSU Sci. Tech. J.*, **33**, 74–87.

Ito, T. and Fukushima, T. (1997) Parallelized extrapolation method and its application to the orbital dynamics, *Astron. J.*, **114**, 1260–1267.

Montenbruck, O. (1992) Numerical integration methods for orbital motion, *Celes. Mech. Dyn. Astron.*, **53**, 59–69.

Murison, M.A. (1989) On an efficient and accurate method to integrate restricted three-body orbits, *Astron. J.*, **97**, 1496–1509.

Neville, E.H. (1934) Iterative intepolation, *Ind. Math. Soc. J.*, **20**, 87–120.

Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T. (1988) *Numerical Recipes in C*, Cambridge University Press, Cambridge.

van der Houwen, P.J. (1993) Parallel step-by-step methods, *Appl. Numer. Math.*, **11**, 69–81.

伊藤孝士 (1996) 国立天文台 VPP300 システムの運用, *FUJITSU*, **47**, 494–507.

岩崎洋一, 宇川彰, 梅村雅之 (1996) 計算物理学と CP–PACS 計画, 情報処理, **37**, 11–17.

野澤恵 (1998) 自作によるパソコン並列計算機の実現, 日本天文学会 1998 年秋季年会, 山形大学.

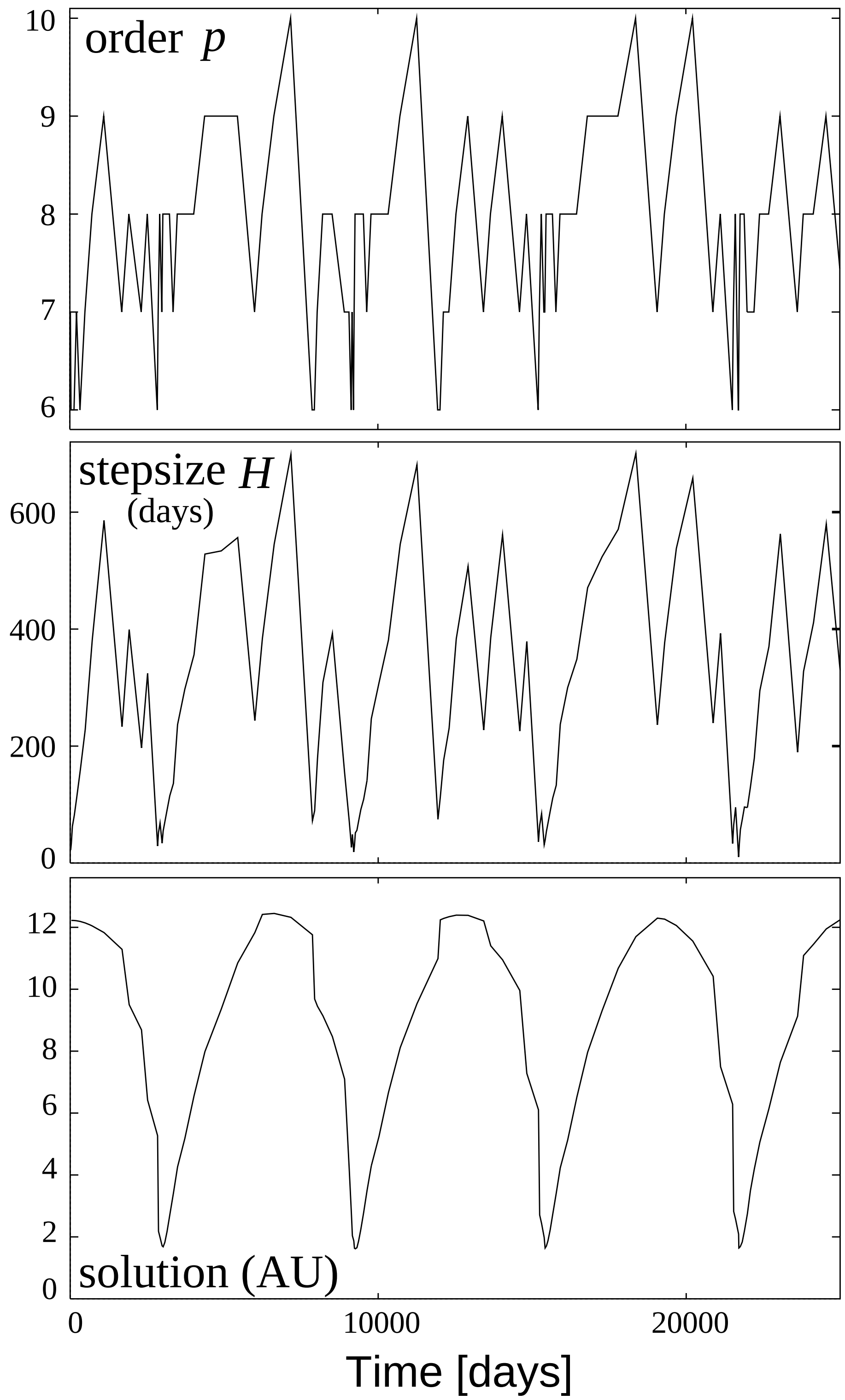

図 5. 図 4に示した計算での段数 p(上), 基本刻み幅 H(中), 解 (太陽と彗星との距離 (AU), 下)。