計算機アーキテクチャ 111－3
（１９９５．３．１０）

並列システムのスケーラビリティによる評価

関口智嗣　佐藤三久
電子技術総合研究所
Email: {*sekiguchi,msato*}*@etl.go.jp*

並列システムのスケーラビリティはシステム設計の立場からすれば，プロセッサ数を変化させた時のシステム性能の振る舞いであるといえる．また，ユーザの視点でいえば，計算サイズ を変化させた時のシステム性能の振る舞いであるともいえる．本稿では並列システムのスケーラビリティに関して述べた．従来の性能評価指標に関して束縛条件，評価尺度，性能推定モデル，基準点という視点から定式化を，試みる．こうした性能評価モデルと指標は線形モデルという単純なものに基づいているため，その得られて数値を議論することはあまり意味がないことを主張した．そこで，実測値を基になんらかの性能推定曲面 $\hat{T}$ を求め，それに対して，考察を行なう方法を示した．いわゆるスケーラビリティは平均速度向上率の変化率と定義し，CM-5 による実例を示した．その結果は直観的なスケーラビリティとも合致している．

Performance Modeling by Scalability for Parallel Computing Systems

Satoshi SEKIGUCHI　Mitsuhisa SATO
Electrotechnical Laboratory
Email: {*sekiguchi,msato*}*@etl.go.jp*

Scalability in parallel computing systems has two aspects for performance evaluation, while its definition is intuitive. Those are understanding their behavior of system performance according to the variation of the number of processors for system architect, or of the number of computation size for an application program user. In this paper, we propose a new technique to understand the scalability with performed data. So far, the performance metrics, speed-up ratio, efficiency, *etc.* are basically based on the simple linear model. We claim it non-sense that the numbers computed by the simple linear model are discussed or compared. We define a "Scalability Functions" derived from the performance estimated surface, which is computed with the measured data. The function shows us a behavior of changing ratio of averaged speedup anywhere you wish to know. It corresponds fairly well between our intuition and the scalability derived.

1 はじめに

ハイパフォーマンスコンピューティングを目指して，各種の並列アーキテクチャ，計算モデルに基づいた並列計算機システム (以下，並列システム) が開発，また製品化されている．こうした並列システムは，その構成ならびに用いられるアプリケーションプログラムがますます複雑になってきている．しかしながら，それに伴って，計算機システムの性能を把握し，かつ，その性能の構成要因を理解するのがますます困難となってきており，並列システムの開発者やユーザの中には並列システムの性能評価に関してさまざまな疑問，誤解，等が生じているように見受けられる．

並列システムの性能評価においては，絶対性能としての実行時間以外にも，効率 (efficiency)，性能向上度 (speedup)，Mflop/s[1] といった指標がこれまでよく用いられてきた．また，最近ではスケーラビリティ (scalability) という指標もよく用いられてきている．こういった指標は，なんらかの意味で並列システムの特徴を表現している数値には間違いないが，実際にシステムの何をどのように表現しているかという洞察が置き去りにされて数値だけが独り歩きしているのが現状である．

本稿では，並列システムのスケーラビリティに関する性能評価指標に関して議論する．本稿では従来の性能評価指標について再考し，新たにスケーラビリティを定量的に捉える試みについて述べる．2章において，性能曲面を定義し，効率とスケーラビリティについて述べ，3章においてスケーラビリティとその新たな定義と具体的な数値例について述べ，4章ではまとめと考察を行なう．

2 Performance Surface (性能曲面) とスケーラビリティ

2.1 スケーラビリティについて

並列システムのスケーラビリティはシステム設計の立場からすれば，プロセッサ数を変化させた時のシステム性能の振る舞いであるといえる．また，ユーザの視点でいえば，計算サイズ (computation size[10]) を変化させた時のシステム性能の振る舞いであるともいえる．したがって，スケーラビリティはいくつかのプロセッサ数や計算サイズに関連するいくつかの性能評価指標によって表現されると考えられる．この『スケーラビリティ』という用語は『スケーラブルな並列システム』や『… な並列システムはスケーラブルである』のように用いられている．直観的にいえば，

> 並列システムの重要なパラメータでプロセッサ数や計算サイズを増加することによりプロセッサ数や計算サイズに見合う性能が得られる度合い

を意味する言葉である．しかし，『スケーラブルな』とか『スケーラビリティの良い』というような定性的な表現はされるものの，定量的な定義は充分な議論がなされていない．

ここでいう並列システムはハードウエア，アーキテクチャそれ自体ではなく，並列アルゴリズム，並列プログラミング，実行時のオペレーティングシステムなどを含むものである．並列システム性能評価が困難であることの本質的な問題は，要素プロセッサ自体の性能に加えて，プロセッサ数とそれらを結合する機構の形態や性能，通信機構ソフトウエアの構成等，システムの構成要素が多岐にわたり，またそれらの関係が非線形である点にある．すなわち，演算器，メモリ，キャッシュ，プロセッサ台数，ネットワークなどのハードウェアアーキテクチャだけではなく，オペレーティングシステムやコンパイラのソフトウエアの能力や環境も考慮しなくてはならない．さらには I/O チャネル，ディスクの容量，速度といった周辺装置の能力も重要である．このように多くの要素が複雑に関連してくると，その性能評価においては何のために (why)，何を (what)，どうやって (how) 評価するのかについて明確化する必要がある．

2.2 Performance Surface

並列システムの性能評価という問題を実験データ解析の問題として定式化する．参考文献 [6] の記法に従えば，

- 測定対象を並列システム $\mathcal{F}$ とする
- r 回の測定値 (誤差を含む，既知) の組を

$$\boldsymbol{T} = \{T_1, T_2, \cdots, T_r\}$$

- 測定条件 (誤差を含む，既知) を

$$\boldsymbol{q}_i = \{q_i^{(1)}, q_i^{(2)}, \cdots, q_i^{(l)}\},\ (i = 1 \sim r)$$

- 真の測定値 (誤差を含まない，未知) を

$$\boldsymbol{T}^* = \{T_1^*, T_2^*, \cdots, T_r^*\}$$

- 真の測定条件 (誤差を含まない，未知) を

$$\boldsymbol{q}_i^* = \{q_i^{(1)*}, q_i^{(2)*}, \cdots, q_i^{(l)*}\},\ (i = 1 \sim r)$$

とする．実際の並列システムの測定における測定条件は，演算器，メモリ，キャッシュ，プロセッサ台数，ネットワークなどのハードウェアアーキテクチャや，

[1] FLOPS という表記を行なわない [3]

実行させたプログラムのサイズ，アルゴリズム，実行環境，周辺装置の能力等を表現しているものとする．具体的な表現は一般的には不可能である．測定値としては，実行時間 (Elapse) 以外にも CPU 時間，メモリ使用量，キャッシュヒット率，ネットワーク使用率などの値がある．

スケーラビリティを議論するに当たっては，プロセッサ数と計算サイズが最も重要である．このため，問題を簡単にするため，以下の仮定を導入する．

- 測定条件の独立変数をプロセッサ数 p と計算サイズ s の 2 次元に限って考える．
- 一般に，実行時間が最も重要な評価対象であり，測定も比較的容易であるため，測定値 $\boldsymbol{T}$ は経過時間とする
- 測定条件 $q_i = \{p_i, s_i\}$ は誤差を含まず，真の点が得られているとする．これは，測定条件を観測者が正確に設定できることによる．
- 測定値 $\boldsymbol{T}(q_i)$ は誤差を含まず，真の点 $\boldsymbol{T}^*$ が得られているとする．すなわち，何度計時しても全く同じ結果が得られるとする．通常のシステムではシステムからの外乱 (測定不能)，タイミング，環境等によって異なる．

定義 1 $\mathcal{D} \equiv [0,\infty) \times [1,\infty)$ において，真のモデルを $f^*(\boldsymbol{x}^*)$，推定モデルを $\hat{f}(\hat{\boldsymbol{x}})$ とする．但し，$\boldsymbol{x}^*$ は真のモデルの m^* 次元パラメータ，$\hat{\boldsymbol{x}}$ は推定モデルの $\hat{m}$ 次元パラメータとする． ∎

定義 2 $T^* = f^*(\mathcal{D})$ を性能曲面 *(Performance Surface)*，または特に区別する必要がある場合には，真の性能曲面とする，$\hat{T} = \hat{f}(\mathcal{D})$ を推定性能曲面とする．このとき，実測点については $T_i^* = f^*(q_i; \boldsymbol{x}^*)$ であり，推定点については $\hat{T}_i = \hat{f}(q_i; \hat{\boldsymbol{x}})$ となる． ∎

定義 3 $T(p,s)$ で計算サイズ s の計算をプロセッサ数 p で実行した場合の実行時間 T を表現する．但し，$(p,s) \in \mathcal{D}, T \geq 0$ とする．また，T は s,p に関して適当な微分可能性を持つとする． ∎

本来離散点においてしかデータ測定が可能ではなく，またシステムの固有の数値 (4 台毎にバスを共有するとかいったシステムの場合の 4 の倍数など) によって，性能は階段状に変化する．しかし，ここでは適当な微分可能性を仮定することにより，システム固有の特異な振る舞いをある程度排除することを期待している．

2.3 Performance Surface と性能指標

推定性能曲面 $\hat{T}$ は推定モデル $\hat{f}$ の定義によって様々な性能評価モデルと関連する指標を導くことができる．性能評価指標 P を議論するに当たっては，評価尺度:U，束縛条件:B，推定モデル:M，基準点:O を明らかにすることにより，P(U, B, M, O) の組として規定しなければならない．以下では，従来の性能評価指標を上記の規定から整理を試みる．

2.3.1 束縛条件

Performance Surface の定義からも明らかなように，性能評価においては様々な要因が関与している．ここでは簡単のために要因を (p,s) としているが，それでも考察は 3 次元で行なわなければならない．しかし，実際の性能評価の場面では，$p-s$ 平面において適当な束縛条件により，唯一の独立変数 t により，曲線 $C(t) = (p(t), s(t))$ に沿って行なうと考える．典型的かつ，重要な意味を持つ 3 例を示す．

計算サイズ固定 (SF) 計算サイズ s は固定して，プロセッサ台数 p を変化させる場合．プロセッサ台数に対するどこまで性能向上のが理解できるため，システム設計の立場からの興味がある．すなわち，$\mathrm{d}s = 0$, $\dfrac{\mathrm{d}s}{\mathrm{d}p} = 0$ という条件と表記できる．

プロセッサ台数固定 (PF)) プロセッサ数 p は固定して，計算サイズ s を変化させる場合．与えられたプロセッサ台数のシステムで自分の問題はどこまで性能向上が得られるかが理解できるため，ユーザの参考になる条件である．同様に，$\dfrac{\mathrm{d}p}{\mathrm{d}s} = 0$, $\mathrm{d}p = 0$ と表記できる．

計算サイズ適応 (SV) プロセッサ数 p に応じて計算サイズ s を変化させる．並列システムになれば，プロセッサ数に応じて (例えば比例させて) 計算サイズを大きくするというユーザの期待に沿った条件である．

2.3.2 評価尺度の定義

いくつかの尺度について，先に定義した T を用いて定義する．

速度向上率

定義 4 速度を計算サイズ s と実行時間 T の比として定義すると，*2* つの測定点 q_1, q_2 における速度の比 (図 *1*)

$$\text{速度向上率}: Speedup(q_1, q_2) \equiv \frac{s_1}{T(p_1,s_1)} \Big/ \frac{s_2}{T(p_2,s_2)} \tag{1}$$

を (一般) 速度向上率と定義する． ∎

この場合，計算サイズが浮動小数点演算数で与えられたとすれば，速度は Mflop/s に対応する．速度向上率は性能曲面 T から任意の 2 点間における値を求める

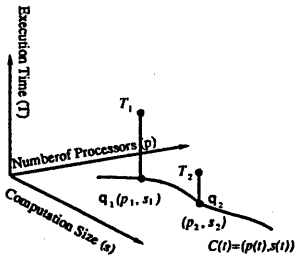

図 1: Performance Surface

ことができる．しかし，その値はプログラム，計算サイズ，測定点にすべて依存するため，値そのものを比較することは実際上の意味を持たない．

2点のうち1点を基準点Oとして $(1, a)$ すなわち，逐次時間 $T(1, s)$ を用いると次の対逐次速度向上率が得られる．

定義 5 式 *(1)* において，$s = s_1 = s_2, p_1 = p,\ p_2 = 1$ としたときに，

$$\text{対逐次速度向上率}: Speedup(p) \equiv \frac{T(1,s)}{T(p,s)} \quad (2)$$

は逐次における実行時間 $T(1, s)$ からの実行時間改善率，すなわち対逐次速度向上率となる． ∎

これは，一般速度向上率と較べて，基準点が逐次時間 $T(1, s)$ にあるため，測定点において一意に対逐次速度向上比を計算できる．また，適当な推定モデル $\hat{f}$ に基づいて，$T(p, s)$ を推定点 $\hat{T}(p, s)$ で置き換えることにより，並列システムの台数効果，計算サイズとの関係について言及することができる．このことは，つまり，並列システムがこれまでの逐次計算の高速化を目標としていたためである．そこで，このため，単に速度向上といった場合には式 (2) を意味することが一般的である．

定義 6 プロセッサ平均速度を速度のプロセッサ台数 p での平均 $(s/T)/p$ として定義する．*2* つの測定点 q_1, q_2 におけるプロセッサ平均速度の比 (図 *1*)

$$\text{平均速度向上率}: R(q_1, q_2) \equiv \frac{s_1/p_1}{T(p_1,s_1)} \Big/ \frac{s_2/p_2}{T(p_2,s_2)} \quad (3)$$

を (一般) 平均速度向上率と定義する． ∎

これは，速度向上率をプロセッサ台数で正規化したものである．式 (3) からわかるように，プロセッサ台数と計算サイズの双方を考慮した形になっている．束縛条件 SV で性能評価を行なうためには必須の指標である．

効率 速度向上率では並列システムとして期待される速度向上の達成度が不明である．システム設計の立場からは期待される速度向上に近付く度合いがその良否を決定する．そこで，式 (1) によって求められた一般速度向上比に対して，T の代わりに推定モデルにより $\hat{T}$ で求められる速度向上率との比 η を考える．すなわち，

$$\begin{aligned} \eta &= \frac{\dfrac{s_1}{T(p_1,s_1)} \Big/ \dfrac{s_2}{T(p_2,s_2)}}{\dfrac{s_1}{\hat{T}(p_1,s_1)} \Big/ \dfrac{s_2}{\hat{T}(p_2,s_2)}} \\ &= \frac{\hat{T}(p_1,s_1)}{T(p_1,s_1)} \Big/ \frac{\hat{T}(p_2,s_2)}{T(p_2,s_2)} \end{aligned}$$

任意の点 q_2 で実測したとして，そこでは $\hat{T} = T$ だとすると，結局，

$$\eta = \frac{\hat{T}(p,s)}{T(p,s)} \quad (4)$$

という指標を得る．これは，従来の効率の定義に他ならない．この効率を導出した際には，基準点はどこでもよかったことに注意してほしい．

定義 7 実測値 T とある推定モデルに基づく推定値 $\hat{T}$ の比として効率を定義する．

$$\text{効率}: Efficiency \equiv \frac{\hat{T}(p,s)}{T(p,s)} \quad (5)$$

∎

2.3.3 性能推定モデル

前節のように，評価指標を定義しておいて，各推定モデルを考察する．以下，$\hat{}$ がついているものはモデルからの推定値，ついていないもの，例えば $T(1, s)$ は測定した結果，または測定する必要のある数値である．

線形モデル これまで，最も多く用いられてきたモデルである．すなわち，p 台における推定実行時間 $\hat{T}(p, s)$ は1台のときの実行時間 $T(1, s)$ の $1/p$ とするモデルである．

$$\hat{T}(p,s) = \frac{1}{p} T(1,s) \quad (6)$$

式 (2) に代入すると，$Speedup(p) = p$ が得られる．このモデルを用いると，プロセッサ数 p を増加するとそれに比例した対逐次速度向上比が期待されている．

また，式 (5) に代入すると，

$$Efficiency = \frac{1}{p} \cdot \frac{T(1,s)}{T(p,s)} \quad (7)$$

となる．これは，p 台における実行時間 $T(p, s)$ は逐字における実行時間 $T(1, s)$ の $1/p$ であるという期待が読

みとれる．この式の問題点は，どのような計算サイズに対しても$T(1,s)$の測定が必須な点である．最近の大規模システムでは$T(p,s)$は求まるものの，$T(1,s)$が求まらないことが多い．また，$T(1,s)$を基準にする必然性はない．

Amdahlのモデル　並列システムの限界についてしばしば言及されるのが**Amdahlの法則**[1]である．これは，$\hat{T}(p,s)$を並列実行部分$\hat{T}_{para}(p,s)$と逐次実行部分$\hat{T}_{seri}(p,s)$に分離して考えられたモデルである．すなわち，

$$\begin{aligned}\hat{T}(p,s) &= \hat{T}_{para}(p,s)+\hat{T}_{seri}(p,s)\\ \hat{T}_{para}(p,s) &= \frac{1}{p}T_{para}(1,s)\\ \hat{T}_{seri}(p,s) &= T_{seri}(1,s)\end{aligned}$$

である．$T(1,s)=T_{para}(1,s)+T_{seri}(1,s)$に留意して，対逐次速度向上率(式(2))に代入すると，

$$\begin{aligned}Speedup(p) &= \frac{T(1,s)}{\hat{T}(p,s)}\\ &= \frac{T_{para}(1,s)+T_{seri}(1,s)}{\frac{1}{p}T_{para}(1,s)+T_{seri}(1,s)}\\ &\leq \frac{T_{para}(1,s)}{T_{seri}(1,s)}+1\end{aligned}$$

線形モデルが性能向上に関して非常に楽観的な側面しか提示していないのに対し，このAmdahlの法則はわずか1%の逐次部分が存在したとしてもその対逐次性能向上は100倍を越えることはないことを主張している．

このモデルは$\hat{T}_{para}(p,s)=\frac{1}{p}T_{para}(1,s)$としていることからもわかるように，線形モデルの改良である．このため，同様に$T(1,s)$が基準となっており，さらにその逐次部分$T_{seri}(1,s)$の解析が重要となっている．

Karpらは逆にこの逐次部分(serial fraction)の解析をおこなうことで，並列システムの性能指標を行なっている[5]．これによると，

$$\alpha \equiv \frac{T_{seri}(1,s)}{T(1,s)}$$

とすると

$$T(p,s)=\{\alpha+\frac{1}{p}(1-\alpha)\}T(1,s)$$

したがって，αについて解くと

$$\alpha=\frac{T(p,s)/T(1,s)-1/p}{1-1/p}$$

となる．しかし，このαは定義により$0\leq\alpha\leq 1$となるはずが，[8]で実測したように実際のシステムには適用できないことがある．すなわち，キャッシュ効果などの要因で線形モデル以上の性能向上が現れる．すなわち，線形モデルに基づいている本解析の限界である．

Gustafsonのモデル　Gustafsonは Amdahlの法則の欠点として，問題のサイズによって逐次部分T_{seri}と並列部分T_{parap}の実行時間の比が変化することが考慮されていない点を指摘した．そこで，Amdahlとは逆に，測定した$T(p,s)$を基準にして$T(1,s)$を推定した[2]．$\hat{T}(1,s)$を並列実行部分$\hat{T}_{para}(1,s)$と逐次実行部分$\hat{T}_{seri}(1,s)$に分離して，

$$\begin{aligned}\hat{T}(1,s) &= \hat{T}_{para}(1,s)+\hat{T}_{seri}(1,s)\\ \hat{T}_{para}(1,s) &= p\cdot T_{para}(p,s)\\ \hat{T}_{seri}(1,s) &= T_{seri}(p,s)\end{aligned}$$

である．$T(p,s)=T_{para}(p,s)+T_{seri}(p,s)$に留意して，対逐次速度向上率(式(2))に代入すると，

$$\begin{aligned}Speedup(p) &= \frac{\hat{T}(1,s)}{T(p,s)}\\ &= \frac{p\cdot T_{para}(p,s)+T_{seri}(p,s)}{T_{para}(p,s)+T_{seri}(p,s)}\\ &= p\cdot(1-\alpha(p))+\alpha(p)\end{aligned}$$

となる．ただし，

$$\alpha(p)\equiv\frac{T_{seri}(p,s)}{T(p,s)}$$

とする．このモデルも基本的には線形モデルを用いて，$T(p,s)$から$T(1,s)$の推定を行なっている．

サイズ相対効率のモデル　佐藤らは$Efficiency$が式(5)からもわかるように，計算サイズsに依存している点に着目して，その影響を排除するようにサイズ相対効率を提案した[8]．ここでは，計算サイズの影響を正規化することで排除して性能を評価しようと試みたものであった．この指標の背景にはモデルとして，$\hat{T}(p,s)=\tilde{T}(s)\cdot s(p)$と分解できると仮定した．ここで，$\tilde{T}(s)$は計算サイズあたりの理想的な計算時間を与える未知の関数である．具体的にはsで表される flop数を単体プロセッサの演算能力 Mflop/s で除したものを想定すれば良い．また，$s(p)$はpでスケーリングした計算サイズsを与える関数である．これを用いて

$$\begin{aligned}\text{サイズ相対効率}\ \zeta &\equiv \frac{\hat{T}(p,s)}{T(p,s)}\cdot\frac{1}{\tilde{T}(s)}\\ &= \frac{s(p)}{T(p,s)}\end{aligned}$$

と定義する．但し，[8]では$\frac{\mathrm{d}}{\mathrm{d}s}\tilde{T}(s)=0$という条件が入っていた．

2.4 各モデルの関連

以上の議論からからわかることは，それぞれに推定モデルを用いているものの，多くのモデルが線形な推定モデルに帰着していることである．特に対逐次速度向上率 $Efficiency(p)$ はこれまでよく用いられているものの，以下のような本質的な欠陥が明らかになった．

- 線形モデルに基づくため，本来 1 (100%) にはなりえない．
- Amdahl の法則が示すように，モデルを簡単に変形することで効率の意味が異なってくる．
- したがって，なんらかの測定によって得られた対逐次速度向上率が 1 に近かったといっても，それはプログラムの性質か，アーキテクチャが優れていたのかを結論付けることはできない．
- ましてや，異なるアーキテクチャ相互の比較や異なるプログラムとの比較を行なうことは数字の上で可能であっても，実際的な意味は全くない．

速度向上率に関しても，どこを基準点として，どういった推定モデルに基づいた結果であるかを併せて明記しないと評価としての意味がない．また，基準点の測定が並列システム $\mathcal{F}$，プログラム毎に異なるため基準点の異なる速度向上率を相互に比較することも推奨できない．例えば，線形モデルに基づいて $T(1,s)$ を基準点とする場合，この値を大きめに測定しておけば得られる $Speedup(p)$ や $Efficiency$ は常に高めに得られる．

なお，$Speedup(p)$ と $Efficiency$ の関係に関しては，$T(1,s)$ を基準点とする線形モデルに基づいた場合に限り，効率と対逐次速度向上率の間には

$$\begin{aligned} Efficiency &= \frac{\hat{T}(p,s)}{T(p,s)} = \frac{1}{p}\frac{T(1,s)}{T(p,s)} \\ &= \frac{1}{p}\cdot Speedup(p) \end{aligned}$$

という簡単な関係が生じる．また，線形モデル以外にも，この関係を摘要することがあるが，それは間違いである．

3 Scalability とその新たな定義

3.1 Scalability 関数

われわれは，並列システムの性能評価指標としてスケーラビリティに関する議論を行なってきた．スケーラビリティを議論するには性能 P(U, B, M, O) を規定する必要があると述べた．さらに，束縛条件 SV を扱うには一般速度向上率ではなく，一般平均速度向上率を使う必要があることを述べた．さて，推定性能曲面 $\hat{T}$ が得られたとする．スケーラビリティを性能 P(U, B, M, O) の B に沿った変化率 とみなす．そこで，次のようにスケーラビリティ関数を定義する．

定義 8 *Scalability Function:* 性能 $P(R,$B, M, O$)$ を考える．このとき，Dt で全微分を表す演算子とすると，

$$\begin{aligned} \mathcal{S}(t) &\equiv \mathrm{Dt}R(q,\mathrm{O}) \\ &= -\frac{s\,\mathrm{Dt}(p)\,T(p,s)}{p^2} + \frac{\mathrm{Dt}(s)\,T(p,s)}{p} \\ &\quad + \frac{s\left(\mathrm{Dt}(s)\,T^{(0,1)}(p,s) + \mathrm{Dt}(p)\,T^{(1,0)}(p,s)\right)}{p} \end{aligned}$$

とにより，S をスケーラビリティ関数と定義する． ∎

この関数は t で定まる点における平均速度向上率の変動を示す．すなわち，

- $\mathcal{S}(t) > 0$ であれば，この点において平均速度向上率はまだ増加している．すなわち，B に沿って性能は向上を続けることが期待されている．
- $\mathcal{S}(t) = 0$ であれば，この点において平均速度向上率の変化はない．すなわち，最もスケーラブルな状態である．
- $\mathcal{S}(t) < 0$ であれば，この点において平均速度向上率は低下している．性能の劣化が見られる状況である．

ことを定量的に示している．一般に，計算サイズ s はそのまま求めることができない．そこで，計算サイズ s を決める問題サイズ n を以下の議論では用いる．特に，B がサイズ固定，プロセッサ数固定の場合には，S は以下のようになる．

サイズ固定 (SF) スケーラビリティ関数

$$\mathcal{S}(p) = -\frac{s(n)}{p^2\,T(s(n),p)} \cdot \frac{s(n)\,T^{(0,1)}(s(n),p)}{p\,T(s(n),p)^2} \tag{8}$$

プロセッサ数固定 (PF) スケーラビリティ関数

$$\mathcal{S}(t) = \frac{1}{p\,T(s(n),p)} - \frac{s(n)\,T^{(1,0)}(s(n),p)}{p\,T(s(n),p)^2} \tag{9}$$

3.2 具体例

具体的なスケーラビリティを考察するために，分割配置した行列乗算の並列プログラムについて実測を行なった．行列 A, B について Column-wise の分割を施し，行列積 $A \times B^t$ を求める問題である．実測対象としたシステムは **Thinking Machine Corp. CM-5**，— 32 プロセッサ (プロセッサは Sparc 32MHz，キャッシュは 64KB．ネットワークは fat tree．バンド幅は 5MB/s から 20MB/s，通信ライブラリは

CMMD) — である．プログラムは標準の最適化オプション (-O) でコンパイルした．

行列のサイズ $n \times n$ とプロセッサ数 p による実行時間 T の関係を調べて基礎データとするため，n は 10 から 510 まで 10 刻みの 50 点，p は 2 台から 32 台まで 2 台刻みで変化させて 16 通りの計 800 点において測定を行なった．その結果を図 2にグラフで示す．

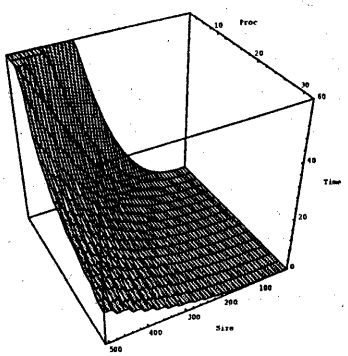

図 2: CM-5 の Matrix Multiply 実測値

この実測値に対して，推定 Performance Surface を定める．今回は推定モデルとして，佐藤のモデルの一般化を用いた．すなわち，$\hat{T}(p,s) = \hat{T}(s) \cdot P(p)$ とし，$\hat{T}$ は n の 3 次多項式 (ただし，$\hat{T}(0) = 0$ としたので実質は 2 次式) を仮定し，$P(p) = \frac{\alpha}{p} + \beta$ と置いた．各測定値 $T_i(p_i, s(n_i))$ に対して，まず $p_i T_i$ で正規化し，$\hat{T}$ を最小 2 乗法による fitting を行なった．次に，$T_i/\hat{T}_i(s(n_i))$ として正規化し，$P(p)$ の fitting を行なった．この fitting には *Mathematica 2.2* を使用した．その結果，得られた推定 Performance Surface は以下のようである．

$$\hat{T} = (0.0300746\,n - 0.00011629\,n^2 + 3.33514 \times 10^{-6}\,n^3) \cdot (0.00868232 + \frac{0.767314}{p})$$

また，グラフに示したのが図 3である．これを元に，式 (8) ならびに式 (9) に代入して，求めたスケーラビリティ曲面を図 4に示す．

SF の場合には，プロセッサ台数が少なく，計算サイズの大きな領域ではスケーラビリティが良くない．ここでは，いくらプロセッサ数を増やしても，期待通りの性能が得られないという傾向を示している．しかし，もともとそういった領域では推定性能曲面を求める際の誤差も大きいことには留意する必要がある．また，プロセッサに適度な演算与えられてくるとスケーラビリティが良くなる (0 に近付く) ことが読みとれる．

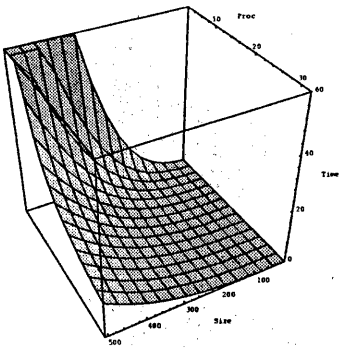

図 3: CM-5 の Matrix Multiply 推定性能曲面

SF の場合には，プロセッサ数が小さい領域でのスケーラビリティが正の領域で高くなっている．これは，以前より指摘されていたスーパーリニア速度向上の可能性がある領域である．すなわち，プロセッサ台数が増加することにより，計算に必要なワーキングセットが小さくなるため，キャッシュの効果が得られるためである．しかし，キャッシュの効果であれば計算サイズに因ってその効果が現れる領域が変わるはずである．しかし，この図では計算サイズに因らないように見える．その理由は，推定モルが s と p の関係を分解したために生じたものと考えられる．こうした点にも推定モデル選択の重要性がわかる．

このように，スケーラビリティに関する大まかな傾向を読みとることが可能である．

3.3 Performance Surface の推定

これまでの議論から，並列システムの性能評価において重要なことは速度向上率や効率ではないことを示した．すなわち，これまでは単純な線形モデルや Amdahl モデル等いったモデルを基に速度向上率，効率が議論されてきた．しかし，ここにいたって，性能評価の目的がいかにして正確な推定モデル $\hat{f}$ を構成するかに帰着した．すなわち，効率の式 (5) において，いたるところ 1 となるようなモデル $\hat{f}$ を構築する問題となる．これは，従来スケーリング則と呼ばれていた．PAX[4] では PAX のアーキテクチャに基づいて様々なスケーリング則を構成し，正確に性能評価を行なっ

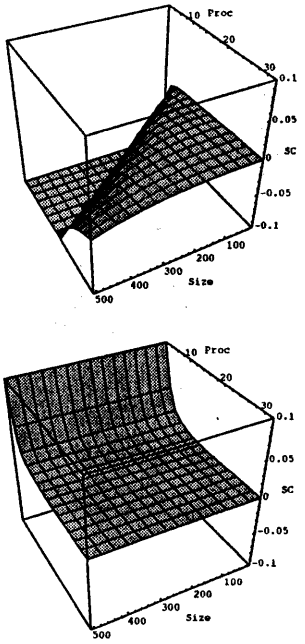

図 4: CM-5 のスケーラビリティ関数 (上:PF, 下:SF)

ている非常に優れた例である．最近でも坂田らは文献 [7] において，通信と演算に関しての詳細なスケーリングを行なっている．しかしながら，仮に，アーキテクチャを固定したとしても一般には，プロセッサ単体性能，キャッシュの効果，プロセッサ間ネットワーク，同期機構など，正確なスケーリングを行なうのは非常に困難になってきている．

$\hat{f}$ を構成する方法には 2 通りある．

- PAX においてなされてきたように，先験的なアーキテクチャの知識を用いて，推定モデルのパラメータ $\vec{a}$ に関する関係式として $\hat{f}$ を構築する．例えば，演算時間 $T_{\text{演算}}$ と通信時間 $T_{\text{通信}}$ が全くオーバラップできないようなシステムであれば

$$T_{\text{全体}} = T_{\text{演算}} + T_{\text{通信}}$$

という関数形になるし，オーバラップすれば

$$T_{\text{全体}} = \max(T_{\text{演算}}, T_{\text{通信}})$$

という関数形になるであろうといった知識を用いる．

- 一方，こうした先験的な知識を用いない推定方法もある．いわゆる Bayesian 推定に基づいて推定性能曲面を導出するという方法が考えられる．具体的な手法については別途報告する．

これにより，測定と推定を繰り返すことにより，より正確な推定モデルを構成する．また，常に，推定モデルとの差がどこに起因するのかを考察すること [9] がシステム設計やプログラム設計に大きな知見を与える．

4 おわりに

並列システムのスケーラビリティに関して述べた．従来の性能評価モデルと指標は線形モデルという単純なものに基づいているため，その得られて数値を議論することはあまり意味がないことを主張した．そこで，実測値を基になんらかの性能推定曲面 $\hat{f}$ を求め，それに対して，考察を行なう方法を示した．いわゆるスケーラビリティは平均速度向上率の変化率と定義し，CM-5 による実例を示した．その結果は直観的なスケーラビリティとも合致している．

今後は，他のシステムに適用し，いくつかの性能推定曲面からスケーラビリティを求めて比較を行なう．また，性能推定の方法について，精緻なモデルの構成法を検討する．

謝辞 本研究は科技庁原子力特別研究『複雑現象の解明における超高速計算機利用技術の研究』に基づくものである．本研究遂行に当たり，議論いただいた，電子技術総合研究所数理情報研究室 赤穂昭太郎氏　通信知能研究室 松井俊浩氏ならびに日頃より御討論いただく The Hokke-Club メンバー各位に感謝致します．CM-5 の結果は佐藤の共同研究の一環として RWC つくば研究センターにおいて計測させていただいたものです．

参考文献

[1] Amdahl, G., "Validity of the single-processor approach to achieving large scale computing capabilities", *AFIPS Conference Proceedings*, Vol. 30, 1967,.

[2] Gustafson, J., "Reevaluating Amdahl's Low", *CACM*, Vol. 31, No. 5, 1988,.

[3] Hockney, R., and Berry, M., "PARKBENCH Report: public international benchmarks for parallel computers", *Scientific Programming*, Vol. 3, No. 2, 1994, pp. 101–146.

[4] 星野力, *PAX* コンピュータ, オーム社, 1985.

[5] Karp, A., and Flatt, H., "Measuring Parallel Processor Performance", *CACM*, Vol. 33, No. 5, 1990,.

[6] 中川徹, 小柳義夫, 最小二乗法による実験データ解析, 東京大学出版会, 1982.

[7] 坂田聡子, 日向寺祥子, 長島雲兵, 佐藤三久, 関口智嗣, 細矢治夫, "ワークステーションクラスタによるホモロジー解析", 情報処理学会研究報告 *HPC-52-18*, 1994,.

[8] 佐藤三久, 関口智嗣, "並列計算機システムのスケーラビリティについて", 情報処理学会研究報告 *HPC-49-2*, 1993,.

[9] 関口智嗣, 佐藤三久, "*HPC* における性能評価 — 測定から科学へ —", 研究報告 93-HPC-46-5, 情報処理学会, April 1993.

[10] Singh, V., Kumar, V., Agaha, G., and Tomlinson, C., "Scalability of Parallel Sorting on Mesh Computer", *Proc. of International Parallel Processing Symposium*, pp. 92–101, 1991.