Vol.2010-HPC-124 No.12
2010/2/23

大規模計算機システムの資源選択を支援するエキスパートシステム

國府理央[†1] 佐藤仁[†1] 松岡聡[†1,†2]

大規模計算環境上でアプリケーションを実行する際，ユーザは複雑な資源設定のためのパラメタを選択する必要がある．しかし，ユーザは必ずしも専門家ではないため，自身で適切な資源選択を行うことは困難である．この問題を解決するために，アプリケーションに応じた適切なパラメタ設定を提案するエキスパートシステムを提案する．実際の大規模計算機システムである TSUBAME の実行ログを多変量解析し，エキスパートユーザの利用パターンを抽出することにより，資源選択予測を行うモデルを構築した．またモデルの検証により，実際にアプリケーションに適切な資源選択予測ができることを確認した．

A Resource Selection Support Expert System for Large-Scale Computing Environments

Rio Kokubu,[†1] Hitoshi Sato[†1]
and Satoshi Matsuoka[†1,†2]

Batch queue systems on large-scale shared-resource supercomputers present the users with numerous, often cryptic, site-specific options in order for them to specify the resources, and appropriately instantiate them for effective application executions. Failure to specify the appropriate parameters, often the case for novice users of the system, not only could degrade the quality of service for the user, but also could compromise the effectiveness and the stability of the system. We instead aim to develop an " advisor " expert system which will present the users with appropriate batch queue parameters and other usage information depending on his needs by answering a simple set of questions. We surveyed job execution logs of the Tokyo Tech TSUBAME supercompter and statistical analysises of them allowed us to construct a correlative usage model which revealed several distinctive user usage patterns, which we could lead to create such an expert system based on the model to reflect the user needs accurately.

1. はじめに

近年，大規模計算環境の利用がますます盛んになっており，その裾野は非コンピュータ専門のユーザも対象として，より広がりを見せている．しかし，TSUBAME[1)]，T2K[2)] を代表とする大規模計算環境は，その利用方法が複雑である．特に，こうした環境では主にバッチスケジューリングシステムが採用されており，アプリケーション実行時に利用する資源の選択を行う必要がある．この選択項目には，使用する CPU コア数，メモリ容量，並列実行数といった多種多様な項目が設けられている．これらを適切に設定することにより，アプリケーションの種別や規模・目的などに応じた資源選択が行えるため，高性能計算の専門家にとっては使い勝手がよいものの，大多数の一般ユーザにとってはどのように資源選択を行えばよいのかわからないという問題がある．この場合，予期せぬ実行エラーや無駄な待ち時間・課金の増大などが起こり，有効に資源を活用することができない．

我々はこの問題を解決するために，一般ユーザのアプリケーションタイプを入力として受け取り，エキスパートユーザの利用パターンをもとに適切な資源選択オプションを提示するエキスパートシステムを提案する．エキスパートユーザの資源選択方法には，アプリケーションの種別・実行タイプごとにパターンが存在すると仮定し，一般ユーザのアプリケーションがどのパターンに類似しているかをもとに，近いパターン内で頻繁に設定されているオプション設定を提示する資源予測モデルを構築する．この予測モデルの有効性を確認するために，実際の大規模計算機システムである TSUBAME の実行ログをオプション設定・実行結果に注目して多変量解析を行い，結果，いくつかの実行パターンを抽出し，これを用いて，パターンへの類似度から設定すべきオプション設定を導き出す資源選択予測モデルの構築を行った．このモデルを用いて資源選択予測を行った結果，実際に適切なオプション設定が出力できることを確認した．

2. 資源選択支援エキスパートシステム

図 1 は資源選択エキスパートシステムの概要である．本システムでは，アプリケーショ

†1 東京工業大学
Tokyo Institute of Technology
†2 国立情報学研究所
National Institute of Informatics

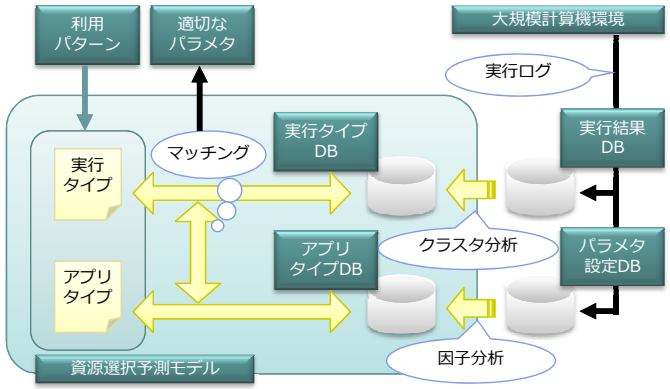

図 1 資源選択支援エキスパートシステム
Fig. 1 An Resource Selection Support Expert System

ンの利用パターンを 1. アプリケーションタイプ，2. 実行タイプの二つにより判断する．前者はアプリケーションの実行前に行う資源選択オプション設定のパラメタに着目して抽出されるパターンであり，後者はアプリケーションの実行後の CPU 利用時間，メモリ使用量などに着目して抽出されるパターンである．この両者に着目することにより，アプリケーションの種別・規模の両方の観点から，より正しく分類することが可能となる．利用パターンの分類は次のように行う．まず，エキスパートユーザによる実行ログを 1. パラメタ設定，2. 実行結果に分類してデータベース化する．次に，パラメタ設定データベースに対して因子分析を行い，アプリケーションタイプを抽出してデータベース化する．実行結果データベースに対してはクラスタ分析を行い，実行タイプを抽出してデータベース化する．利用者は，利用パターンとして実行するアプリケーションのアプリケーションタイプと実行タイプの両方を入力し，システム側がこれを受けとって資源選択予測モデルによる利用パターンデータベースとのマッチングを行い，適切なオプション設定を出力する．

3. 関 連 研 究

大規模計算機環境における資源選択の最適化手法については，多くの研究がなされてきた．その手法には，大きく分けて環境全体の最適化を行うもの，アプリケーション単体の性質に基づいた最適化を行うものがある．環境全体の最適化を行うものには，ロードバランシングを行う手法[3]，ジョブと資源のマッチメイキングにより遊休資源の最小化を行う Condor[4]，予算と deadline の指定によりコストの最適化を行う Nimrod[5]，利用されるマシン数の最適化を行う手法[6] などがある．これらの手法ではシステム側で定めた何らかのパラメタのみを最適化しており，個々のユーザのアプリケーションの特性や規模，目的を考慮していない．アプリケーション単体の性質に基づいた最適化を行うものは，さらにアプリケーション解析による性能予測に基づくもの，過去のジョブ実行結果による性能予測に基づくものに分類される．前者[7),8)] ではアプリケーションを事前に実行することにより性能モデルを生成して，実行時間を予測，それを元に資源選択を行う．後者では，過去のジョブ実行結果との類似性から，アプリケーションの性能を予測する．時系列での解析を行う手法[9),10)] や，過去のジョブを分類して類似性を判定する手法[11),12)] がある．実行ログの解析を行うものでは，アプリケーション実行時にユーザが指定した資源選択パラメタと，過去の実行ログ解析結果とを照合し，実行結果を予測してユーザの設定パラメタを補正する手法[13)] が提案されている．本研究では，過去の実行ログ解析からパターンを抽出し，ユーザがそれを用いて利用希望を入力し，適切な資源選択パラメタの設定を出力することを目的としている．

4. TSUBAME ログ調査

エキスパートユーザの利用パターンを抽出するため，実際の大規模計算機システムである TSUBAME の実行ログの解析を行った．TSUBAME では，**表 1** の資源選択オプションが指定可能である．また，実行キュー選択によっても利用できる資源が異なり，割り当てられるノードの CPU 周波数，最大利用可能 CPU 数，メモリ容量，GPU の有無が選択可能である．

4.1 TSUBAME の利用状況

表 2 は 2008 年 11 月から 2009 年 9 月における TSUBAME のアプリケーション別利用状況である．アプリケーション種別が不明である未分類アプリケーションが全体の 9 割を占めていること，全体としてのジョブ異常終了率は 8.8%であるもののアプリケーションごとの異常終了率にばらつきがあることなどから，実行ログの分析については，まずアプリケーションごとに分けた上で，さらにキューごと，課金グループごとなどに細かく分類して精細に行う必要がある．

4.2 分 析 区 分

精細な結果を得るために，アプリケーションごとに実行ログを**図 2** のような区分に分けて分析を行う．正常終了したジョブのみをエキスパートユーザによるものとみなし，分析対象

表 1 TSUBAME における資源選択オプション
Table 1 Resource Selection Parameters Available in TSUBAME Grid Cluster

オプション	引数	概要
-mem	<memory size>	各プロセスのメモリサイズを指定．単位:GB
-rank0mem	<memory size>	Rank0 の使用するメモリサイズを指定．単位:GB
-mpi	<cpus>	MPI での実行およびジョブに必要な CPU 数を指定．
-smp	<cpus>	ジョブに必要な CPU 数を指定．同一ノード内でのみ実行．
-proc	<cpus>	平行実行に必要な CPU 数を指定．同一ノード内でのみ実行．
-rt	<run time>	SLA または BES のキュー実行時間上限．単位:min
-pl	<premium level>	使用するプレミアムレベルを指定．
-q	<queue name>	ジョブを実行するキューを指定．
-linda	<cpus>	Gaussian Linda で実行する CPU 数を指定．
-ddi	<cpus>	GAMESS を実行する CPU 数を指定．
-stripe	<count>	Gaussian における Lustre FS の組み方を指定．

表 2 TSUBAME 利用状況 ('08.11-'09.9)
Table 2 Utilization of TSUBAME('08.11-'09.9)

アプリケーション	総ジョブ数	異常終了ジョブ数	ジョブ異常終了率 (%)
未分類	2546442	203420	7.99
Gaussian(CLI)	192737	5900	2.94
Gaussian(GUI)	299	14	4.68
ABAQUS	46647	29407	63.04
MATLAB	34006	3323	9.77
AMBER	13332	3222	22.09
LS_DYNA	6405	4118	26.16
GROMACS	3801	904	23.78
GAMESS	3565	204	9.57
FLUENT	1262	208	16.48
Mathematica(GUI)	638	147	23.04
Mathematica(CLI)	170	5	2.94
NASTRAN	197	10	5.08
Molpro	194	15	7.73
POV-Ray	163	4	2.45
MOPAC	82	2	2.44
NWChem	37	0	0.00
EnSight	15	2	13.33
MAPLE	10	0	0.00
UTChem	8	0	0.00
全体	2850010	250905	8.80

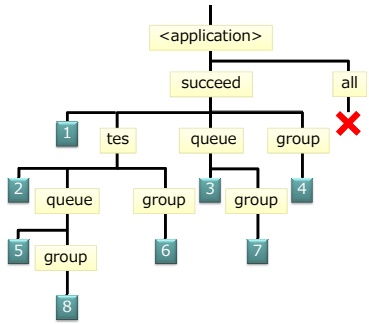

図 2 分析区分
Fig. 2 Analysis Segmentation

とする．各枝において，tes は GPU を利用したジョブのみ，queue は各キューに投入されたジョブのみ，group は各課金グループによる実行ジョブのみ，という条件を表している．キュー種別は利用されたキューによって，ベストエフォートサービスである BES，性能保証サービスである SLA，事前予約サービスである HPC のいずれかに分類し，課金グループは TSUBAME で利用申請されている約 330 グループをそのまま活用した．この条件分けにより，各アプリケーションごとに 8 個の区分を設けて分析を行った．

5. 資源選択予測モデル

資源選択予測モデルでは，利用パターンを入力として受け取り，エキスパートユーザのどの利用パターンに類似しているかを判断して適切なオプション設定を出力する．このために，実行ログのパラメタ設定と実行結果の両方を分析する必要がある．パラメタ設定の分析の目的は，数が多く複雑なパラメタからアプリケーションに固有の設定パターンを抽出し，より単純なアプリケーションタイプを入力とするモデルを構築することである．そこで，複数の観測変数に共通して影響を与えている因子を抽出する分析手法である因子分析を用いる．実行結果の分析では，CPU 使用時間やメモリ使用量などの実測値の分布から実行パターンを抽出し，因子分析で抽出したパターン内で，どのようなアプリケーション規模・実行方法を行っているかの位置づけをすることが目的である．そこで，複数の観測変数を持つデータ群を，近い観測変数の組み合わせを持つデータのグループに分類する分析手法であるクラスタ分析を用いる．

5.1 アプリケーションタイプ分析

アプリケーションタイプの分析では，pmem(プロセスあたりメモリ量)，rank0mem(rank0 のメモリ量)，mpi(MPI 並列数)，smp(smp 並列数)，proc(プロセスあたりメモリ指定を行うときの smp 並列数)，rt(実行時間上限)，pl(プレミアレベル)，TES(GPU 有無)，CPUFreq(CPU 周波数)，CPUMax(最大 CPU 数)，MEMCap(メモリ容量)，linda(Linda 実行数)，ddi(ddi 実行数)，stripe(Lustre FS ストライプ数) の 14 項目のパラメタ設定について因子分析を行う．ここで，TES とは GPU の有無による 0 または 1 の値，CPUFreq はノードの CPU 周波数，CPUMax は最大利用可能 CPU 数，MEMCap はノードのメモリ容量である．また，pmem は表 1 の mem にあたり，その他の項目は表のとおりである．

5.1.1 因 子 分 析

因子分析は，m 個の各個体における n 個の観測変数 $\{x_{i1}, x_{i2}, \ldots, x_{in}\}$ に共通して影響をおよぼす因子 $\{f_{i1}, f_{i2}, \ldots, f_{ik}\}$ を抽出するための分析手法である $(i = 1, 2, \ldots, m)$．各変数が，共通因子によって表される部分，独立因子で表される部分からなる，という考え方に基づいている，因子数を k 個と置いたとき，

$$x_{i1} = a_{11} f_{i1} + a_{12} f_{i2} + \cdots + a_{1k} f_{ik} + \epsilon_{i1}$$
$$x_{i2} = a_{21} f_{i1} + a_{22} f_{i2} + \cdots + a_{2k} f_{ik} + \epsilon_{i2}$$
$$\ldots$$
$$x_{i2} = a_{21} f_{i1} + a_{22} f_{i2} + \cdots + a_{2k} f_{ik} + \epsilon_{i2}$$

を満たすような因子負荷行列 A および m 個の個体ごとの因子得点をまとめた因子得点行列 F，独自因子 ϵ をまとめた独自因子行列 E を求めるのが目的である．因子負荷行列の各要素は，変数への因子の影響の大小を表しており，正の値ならば因子得点の増加に対して比例して変数の値が増加し，負の値ならば減少する．ここで，各列に個体ごとのデータをまとめた $n * m$ 行列を X とおくと，全データを共通因子と独自因子とで表現する因子モデルは

$$X = AF + E$$

で表される．実行ログのパラメタ設定を観測変数とすれば，アプリケーションタイプが共通因子にあたり，因子分析を行うことはアプリケーションタイプの抽出の目的を満たすといえる．

5.1.2 分 析 結 果

全区分の因子分析により，64 個の因子を抽出した (**図 3**)．図 3 は，各因子の各パラメタに対する因子負荷量を表しており，色が赤に近いほど因子得点の増加に対してパラメタの値が大きく増え，緑に近いほど大きく減る．黄色に近いほど因子のパラメタに与える影響が小さくなる．全ての区分において，因子分析前に各パラメタの平均が 0，分散が 1 になるように標準化を行った．初期解求解は最尤法，因子軸回転はバリマックス法，因子得点求解は回帰法にて行った．また，因子数は，2 から 15 個の全ての場合で分析し，最も適合度の高かったものを採用した．適合度とは，各因子モデルにおける,「その因子数での因子空間が元データ空間に適合する」という帰無仮説のもとにカイ二乗検定を行ったときの p 値を示しており，0.05 を超えれば適合しているとみなす (有意水準 5%)．区分内の全サンプルで設定されていないパラメタは分析対象から除外した．あるサンプルについて，パラメタ設定が全く同じ組み合わせの他サンプルが存在した場合はそれらを除外した．サンプル数が 30 以下，または，パラメタ数が 2 個以下のものについてはすでにアプリケーションタイプが絞られているとみなし，因子を抽出する必要がないとして，因子分析を行わなった．因子分析が行われた区分は合計 70 個となり，その分析結果は**表 3** のようになった．グループ分類のある区分では，分析が行われたグループのうち最もサンプル数が多いもののみ示している．

5.1.3 モデル選択

表 3 より，未分類アプリケーション (以下，notype) の区分 1，区分 4 の数グループ，区分 7 の HPC において有意水準を満たさないが，大半の区分では高い適合度を示していることがわかる．また，各区分で抽出された因子と全体の因子 (図 3) の照合により，notype 以外のアプリケーションでは，細かい区分で抽出された因子が区分全体で抽出された因子に包括されること，notype ではグループ分類の区分で多くの因子が抽出され，区分全体で抽出された因子ではそれらを包括できないことがわかった．そこで，notype 以外のアプリケーションでは全体 (区分 1) の因子分析結果を利用するモデルで十分であり，notype では，利用タイプを多くカバーするという点ではグループ分類まで行う (区分 4，7，8) のが望ましいと考えられる．しかし，notype に関しては，実際に課金グループによる分類を行うのは，区分数が多すぎることから，利用パターン入力時にユーザが選択するのが困難と考えられ，現実的でない．実際，TES 分類およびキュー分類までの区分で十分な適合度を示しており，また，因子同士は必ずしも直交ではないため，notype については区分 3，5 を採用すれば，十分他の因子をカバーできると考えられる．よってアプリケーションタイプとして，Gaussian，ABAQUS，AMBER，FLUENT，GAMESS の区分 1，notype の区分 3(BES，SLA，HPC)，区分 5(SLA，HPC) の全 10 区分における因子分析モデルを利用する．なお，notype の区分 5 の BES についてはサンプル数が 4 であったため除外した．

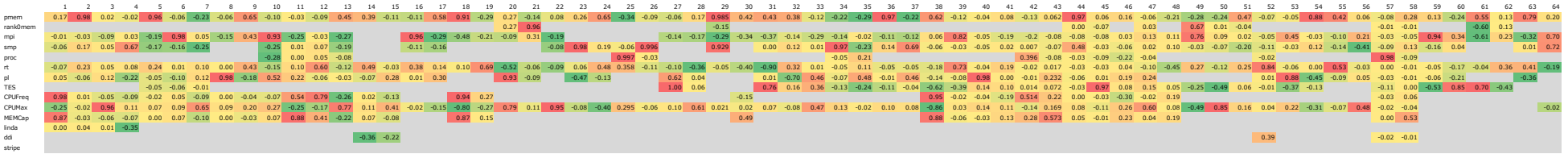

図 3 因子一覧
Fig. 3 Factors Extracted

表 3 因子分析結果
Table 3 Result for Fact Analysis

アプリケーション	区分	サンプル数	パラメタ数	因子数	適合度	備考
Gaussian	1	529	9	4	0.94	
Gaussian	3(BES)	56	5	2	0.32	
Gaussian	3(SLA)	427	8	3	0.21	
ABAQUS	1	235	8	4	0.98	
ABAQUS	3(SLA)	177	7	3	0.95	
ABAQUS	4(1)	38	5	2	0.52	
AMBER	1	112	7	3	0.99	
AMBER	3(SLA)	83	7	3	0.82	
FLUENT	1	72	9	4	0.19	
FLUENT	3(SLA)	41	5	2	0.27	
GAMESS	1	93	7	3	0.99	
notype	1	3601	12	5	0.003	
notype	2	237	6	2	0.03	
notype	3(BES)	569	9	4	0.86	
notype	3(SLA)	2463	12	6	0.74	
notype	3(HPC)	347	7	3	0.07	
notype	4	286	8	4	0.02	全 21 グループ
notype	5(SLA)	165	6	2	0.31	
notype	5(HPC)	72	5	2	0.69	
notype	6	117	6	2	0.13	全 3 グループ
notype	7(BES)	107	8	2	0.96	全 3 グループ
notype	7(SLA)	212	8	4	0.93	全 18 グループ
notype	7(HPC)	74	5	2	0.03	全 4 グループ
notype	8(SLA)	86	6	2	0.66	全 3 グループ

5.2 実行タイプ分析

実行タイプ分析では，wait(キューでの待ち時間)，wallclock(実行時間)，CPU(CPU 利用時間)，mem(メモリ使用量)，mpi(MPI 並列数)，smp(smp 並列数)，proc(プロセスあたりメモリ指定を行うときの smp 並列数) の 7 項目の実測値についてクラスタ分析を行う．

5.2.1 クラスタ分析

クラスタ分析は，m 個の個体における観測変数である n 次元ベクトルに関して，ベクトル間の距離に基づいていくつかのクラスタに分類するための分析手法である．距離の近いものから順にクラスタを成していく階層型分析と，初めにクラスタ数 k を決定しておき，その数だけのクラスタに分類する非階層型分析がある．今回はデータ数が膨大であるため，後者の代表例である k-means 法を用い，距離はユークリッド法で算出する．

5.2.2 分析結果とモデル選択

実行タイプ分析では，アプリケーションタイプで定まったパターンの中でさらに実測値に基づく分類分けを行うのが目的であるため，アプリケーションタイプで選択したものと同一の 10 個の区分を利用する．各区分においてはクラスタ数が 2 から 15 個の全ての場合について順に分析し，クラスタ内の個体間距離の和である郡内平方和の減少の幅が少なくなるときのクラスタ数を採用した．**図 4** は notype の区分 3(BES) のクラスタ数に対する郡内平方和である．この場合はクラスタ数 8 個の時に郡内平方和の減少が緩やかになっているため，8 クラスタモデルを採用する．

5.3 モデルの利用方法

因子分析とクラスタ分析により求めたアプリケーションタイプと実行タイプは次のように用いる．

(1) アプリケーション名とキューの選択
どの因子分析区分を利用するかを選択する

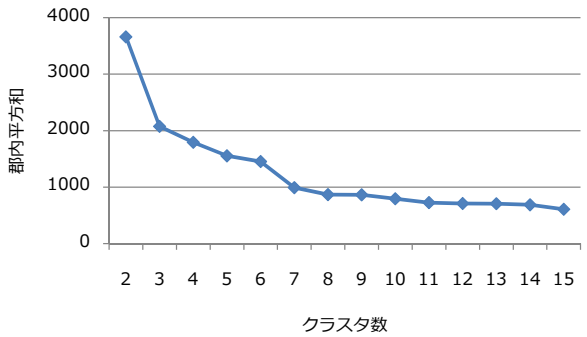

図 4 notype3BES におけるクラスタ数に対する郡内平方和
Fig. 4 Within-Groups Sum of Squares for Each Cluster Number in Notype3BES

(2) アプリケーションタイプの入力
因子分析で抽出された各因子 (アプリケーションタイプ) に対し，どれくらいあてはまるかを入力

(3) 実行タイプの入力
クラスタ分析で抽出した実行タイプのどれにあてはまるかを入力

6. モデル評価と考察

モデル評価のため，前章で選択した 10 個の区分のうち，未分類アプリケーション (notype) のキューによる分類を行った区分 3 で，BES キューで実行されたもの (以下，notype3BES) の分析結果について検証する．notype3BES のアプリケーションタイプを表す因子負荷行列は**図 5**，実行タイプを表すクラスタは**表 4** のようになった．クラスタ分析結果では，各クラスタに所属するサンプル数と資源利用実測値の平均値を示している．

6.1 モデル評価方法

モデルが正しく資源選択予測を行えることを確認するために，実行ログの元データを，モデルを用いて予測したデータと比較する．評価基準として，2 データ間のカイ二乗距離を用いる．ここで，n 次元ベクトル $x = \{x_1, x_2, \ldots, x_n\}$ と $y = \{y_1, y_2, \ldots, y_n\}$ のカイ二乗距離を次のように定義する．

$$d_{chi} = \sqrt{\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{p_i}}$$

ただし，

$$p_i = \frac{|x_i| + |y_i|}{N}$$

$$\left(N = \sum_{i=1}^{n} (|x_i| + |y_i|) \right)$$

	Factor1	Factor2	Factor3	Factor4
pmem	-0.1862	-0.0658	0.30854	0.44938
mpi	0.99112	-0.0532	0.07855	0.06064
smp	-0.1871	-0.1052	0.1628	0.41489
proc	-0.0364	-0.0178	-0.0168	-0.1686
rt	-0.0058	0.83902	-0.0213	0.08069
pl	0.01223	0.01189	-0.5874	0.06536
TES	-0.0212	-0.0101	0.02299	-0.1062
CPUMax	0.03494	0.03634	0.15944	0.07889
ddi	-0.0097	0.38632	0.04154	-0.0332

図 5 notype3BES における因子負荷行列
Fig. 5 Factor Loading Matrix for notype3BES

表 4 notype3BES のクラスタ分析結果
Table 4 Result of notype3BES Cluster Analysis

	size	wait (sec)	wallclock (sec)	cpu (sec)	mem (GB)	mpi	smp	proc
1	7356	1852	16510	52739	12395	0	4	0
2	9428	7854	438504	1062315	732133	46	0	0
3	65694	169	8229	18936	18175	62	0	0
4	1757321	2485	2452	2950	1885	0	0	0
5	28452	616	4804	20879	12434	178	0	0
6	41025	263	2367	5793	2469	27	0	0
7	5326	14888	10314	80991	91618	0	11	0
8	30673	11133	141308	248058	791440	13	0	0

6.2 因子分析モデルの評価

因子分析で抽出したアプリケーションタイプを用いてパラメタ設定の予測を行い，元のデータと比較した．結果は**表 5** のようになった．**図 6** は notype の区分 3(BES) のサンプル

表 5 因子分析モデル評価結果
Table 5 Evaluation of Fact Analysis Model

区分	最大距離	最小距離	平均距離	有意水準値
Gaussian1	26.57	0.85	3.15	12.59
ABAQUS1	17.99	1.05	2.89	12.59
AMBER1	10.78	1.36	3.04	7.82
FLUENT1	11.43	1.55	4.14	12.59
GAMESS1	10.85	1.21	3.13	7.82
notype3BES	16.27	0.63	2.95	12.59
notype3SLA	53.37	0.85	4.26	16.92
notype3HPC	13.05	1.39	2.91	7.82
notype5SLA	13.34	1.12	3.40	9.49
notype5HPC	7.67	0.05	2.37	3.84

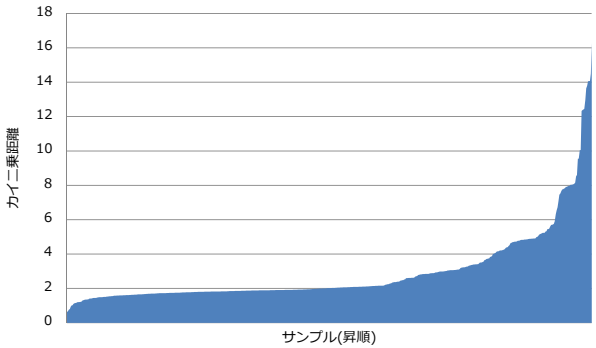

図 6 notype3BES のサンプルごとのカイ二乗距離
Fig. 6 Chi Square Valule for Each Samples in notype3BES.

ごとのカイ二乗距離の分布，**図 7** は同区分における worst・best ケースの元データと予測データの比較である．この区分におけるカイ二乗分布は自由度 6 であり，有意水準 (5%) 値は 12.59 である．図 6 より，worst ケースではこの値を超えるものの，大半のサンプルが有意水準値を下回っており，適合度の高いモデルだといえる．また，図 7 では，best ケースで元データを正確にデータを予測しており，worst ケースにおいてもほぼ正しく予測できていることがわかる．

6.3 クラスタ分析モデルの評価

クラスタ分析モデルの評価を行うために，前節における notype の区分 3(BES) の worst・best ケースのデータに関して，因子分析前に行った標準化を元に戻して検討する (**表 6**)．**表 7** は，worst・best ケースでのアプリケーション実行完了後における資源利用の実測値である．これらのパラメタ設定と実測値の比較により，worst ケースでは rt パラメタの過大設定 (実測実行時間 303(sec) に対し，31680(min)) があること，best ケースではメモリ上限の過大設定 (実測使用量 0.2(GB) に対し，3*512(GB)) があることがわかる．ここで，パラメタ設定をクラスタ分析結果 (表 4) と照らし合わせてみると，worst ケースはクラスタ 2，best ケースはクラスタ 5 に所属することがわかる．クラスタ分析モデルを用いることによって，クラスタ 2 の wallclock の平均値は 438504(sec) であることから，-rt パラメタが不必要に高いこと，またクラスタ 5 でも mem の平均値は 12434(GB) であり，-mem パラメタが不必要に高いことを，事前に知り補正することができると考えられる．さらに，best ケースでは，実際の待ち時間が長い (クラスタ内平均 616(sec) に対して 8604(sec)) 理由が高い MPI 数 (クラスタ内平均 178 に対して 512) によるものでないかなどの推測ができ，必要に応じて設定を変えることができる．以上のように，クラスタ分析モデルは，実測値を見ないとわかりにくいパラメタを設定する上で有用であるといえる．

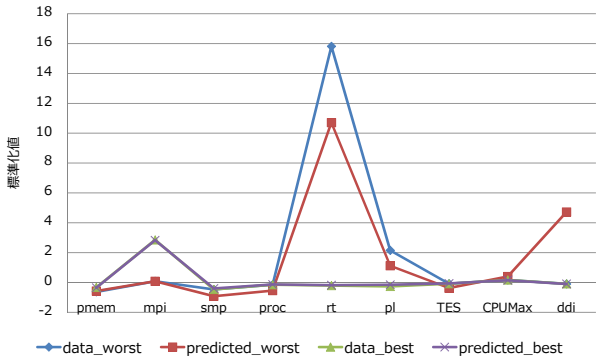

図 7 notype3BES の worst・best ケースの元データと予測データの標準化値
Fig. 7 Real and Predicted Data for the Worst and the Best Cases in notype3BES

6.4 考　　察

以上の評価結果から，本モデルの導入により，資源選択方法の改善とアプリケーションの異常終了率低減が期待できる．資源選択方法について，従来は資源選択オプションを絶対値で指定しなければならず，とくに実行時間上限やメモリ上限などは実行前からはわかりにく

表 6 notype3BES における worst・best ケースのパラメタ設定
Table 6 Parameter Settings in the Worst and the Best Case of notype3BES

	pmem	mpi	smp	proc	rt	pl	TES	CPUMax	ddi
worst_data	1	64	0	0	31680	2	0	1888	0
worst_predicted	1.4	63.3	-1.8	-0.5	21575.4	1.6	-0.0	1955.7	3.6
best_data	3	512	0	0	30	1	0	1888	0
best_predicted	2.9	509.7	0.3	0.0	70.2	1.1	0.00	1883.2	-0.0

表 7 notype3BES における worst・best ケースの資源利用実測値
Table 7 Measurement Value of Resource Usage in the Worst and the Best Case of notype3BES

	wait	wallclock	cpu	mem	mpi	smp	proc
worst_data	0	303	4522	4505	64	0	0
best_data	8604	17	40	0.2	512	0	0

いという問題があった．本モデルでは，アプリケーションタイプが提示され，どのタイプに近いかを漠然と入力するのみでよく，絶対値を設定する必要がない．また，他のユーザが指定している傾向が自然と取り入れられるため，非専門ユーザが気づきにくい，指定することが推奨されるオプションを自動的に設定することができる．また，正常終了したエキスパートユーザによる実行結果を元に構築されたモデルであるため，無理のある設定は自然と除外され，アプリケーションの異常終了を防止することにも役立つ．

7. まとめと今後の課題

大規模計算機システムにおいて，アプリケーションに応じた資源利用オプション設定を提示するエキスパートシステムを構築するために，TSUBAME の実行ログを多変量解析し，アプリケーションタイプと実行タイプの 2 つの観点から利用パターンを抽出した．また，これを利用し，より単純で直観的な項目選択によって資源選択パラメタ設定を提示するモデルを構築した．モデルの検証により，実際に適切な資源選択が行えることを確かめた．今後の課題として，エキスパートシステムの実運用，および実際のユーザ利用による評価，利用パターン入力の自動化などがあげられる．今回はモデルが正しく元のデータ通りに資源選択を予測できることのみ検証を行ったが，今後は実際にアプリケーションを用意し，システムを利用して得たオプション設定によって実行した場合の検証も行っていきたい．また，現在は利用パターンの入力時に，ユーザ自身の判断によりアプリケーションタイプと実行タイプを入力する必要がある．この方法では，ユーザの利用希望が反映しやすいという長点がある反面，ユーザが正しく入力を行えない可能性がある．これを防ぐために，事前に無料キューである程度実行し，その結果からアプリケーションの特性を判断し，自動で利用パターンをシステムに渡すような機構についても今後考えていきたい．

参 考 文 献

1) TSUBAME Grid Cluster. http://www.gsic.titech.ac.jp/c̆cwww/.
2) T2K Open Supercomputer. http://www.open-supercomputer.org/.
3) B.A. Shirazi, A.R. Husson, and K.M. Kavi. Scheduling and load balancing in parallel and distributed systems. *IEEE Computer Society Press*, 1995.
4) J.Basney and M.Livny. Deploying a high throughput computing cluster. *High Performance Cluster Computing*, Vol.1, , 1999.
5) R.Buyya, D.Abramson, and J.Giddy. Nimrod/g: An architecture of a resource management and scheduling system in a global computational grid. In *4th International Conference on High Performance Computing*, Vol.1, pp. 283–289, Asia-Pacific Region, 2000.
6) R.Huang, H.Casanova, and A.Chien. Automatic resource specification generation for resource selection. In *conference on Supercomputing*, Vol.1, Reno, Nevada, 2007. ACM/IEEE.
7) J.Schopf and F.Berman. Performance prediction in production environments. In *IPPS/SPDP*, Vol.1, 1998.
8) V.Taylor, X.Wu, J.Geisler, X.Li, Z.Lan, M.Hereld, I.Judson, and R.Stevens. Prophesy: Automating the modeling process. In *Third International Workshop on Active Middleware Services*, 2001.
9) R.Wolski, N.Spring, and J.Hayes. The network weather service: A distributed resource performance forecasting service for metacomputing. *Future Generation Computer Systems*, Vol.15, pp. 757–768, 1999.
10) P.Dinda. Online prediction of the running time of tasks. In *10th IEEE Symposium on High Performance Distributed Computing*, 2001.
11) A.Downey. Predicting queue times on space-sharing parallel computers. In *International Parallel Processing Symposium*, 1997.
12) AHistorical ApplicationProfiler for Useby ParallelSchedulers. Online prediction of the running time of tasks. *Lecture notes on Computer Science*, pp. 58–75, 1997.
13) W.Smith, V.Taylor, and I.Foster. Predicting application run-times using historical information. In *IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing*, 1998.