目次

Cover : Designed by TANI Takushi / T&T Design Lab. (August,2012)

全国共同利用版

広報

「新スーパーコンピュータ運用開始」

中島　浩◎山口倉平、池田健二、斎藤紀恵、疋田淳一◎平石　拓

ISSN 1347-3581

京都大学学術情報メディアセンター
Academic Center for Computing and Media Studies, Kyoto University

Vol. 11, No.1 号の発刊にあたって

京都大学学術情報メディアセンター
岩下　武史

今号では、2012年5月にリプレースされました新スーパーコンピュータシステムに関する特集を組ませていただきました。利用者の皆様には申し訳ありませんが、センター教職員が新システムの安定稼働に注力できるよう、例年よりもNo. 1号の発行を遅らせて頂きました。これは、新システムの性能評価に関する記事を掲載する為の措置でもあります。

本号では、まず新スパコンシステムの仕様策定委員会委員長でもあった中島浩センター長より、システムに関する紹介とその狙いについて寄稿を頂きました。利用面における旧システムからの継続性を保ちつつ、限られた電力容量の中で2012年時点に相応しい性能を実現する為のシステム設計について述べて頂きました。次に、山口氏、池田氏、斎藤氏、疋田氏より新システムの利用ガイドを寄稿頂きました。掲載されている情報の多くはセンターWEBページにも掲載されていますが、本広報をリファレンスとしてお手元で活用頂ければと考えています。さらに、平石氏からは新システムの性能評価に関する記事を頂きました。本性能評価は新システムの導入において利用したいくつかのベンチマークプログラムを用い、新システムの性能を示すものとなります。本記事により、旧システムと比べて新システムが性能面でどの程度改善されたかを理解することができます。

中島センター長の記事にもありましたように、今回のリプレースではシステムの納入ベンダが変更されたため、ユーザの皆様にはプログラム実行に際して、コマンド実行、ジョブスクリプト作成、数値計算ライブラリ利用等の全てのフェーズにおいて従来からの変更をお願いすることになりました。また、プログラムの変更を行わないと実行ができない、あるいは性能がでないというような場合もあったかと思います。スパコン調達の手続き上、納入ベンダの変更は常に可能性のあるものであり、センターとしてはなるべくユーザの皆様のご負担を少なくすべく努力をする所存です。今回は旧システムの主要コンポーネントであったHX600クラスタ（マシン名: thin）と同じx86系のプロセッサにより全てのサブシステムが構成されているため、その点は移行のハードルを下げていると考えていますが、移行に関してお気づきの点があればセンターコンサルト（consult@kudpc.kyoto-u.ac.jp）宛に御意見を頂ければ幸いです。

また、本年度から、本センターも資源提供機関として参加しているHPCI（革新的ハイパフォーマンス・コンピューティング・インフラ）の運用が始まります。HPCIは「京」を中核とし、国内の大学・研究所のスパコンを連携し、これを効率的に運用する仕組みとなります。現時点において、HPCIに含まれるスパコンのシングルサインオンやGfarmによる分散ファイルシステムの利用が可能となる予定です。例えば、京で計算した結果に基づいて、京大センターのスパコンリソースによりデータ解析を行うといったことがより簡単にできるようになります。本運用はこれからですので、運用開始直後はいろいろと使い勝手が悪いところもあるかもしれませんが、段階的にそれも改善されるかと思いますので、国内最大の計算リソース集合体であるHPCIの活用についても、ご検討いただければと考えます。

本センターの新システム及びHPCIを皆様のご研究に活用いただけるようセンター教職員も尽力をしていきますので、今後ともご利用、ご支援の程、よろしくお願いします。

新スーパーコンピュータシステム
―そのコンセプトと構成―

中島 浩
京都大学・学術情報メディアセンター

1　はじめに

2012 年 5 月に稼動を開始した学術情報メディアセンターの新しいスーパーコンピュータシステムは、2011 年度まで稼動した T2K オープンスパコン [1, 2] HX600 クラスタのコンセプトを受け継ぎつつ、最先端の技術を駆使した国内有数の大規模・高性能システムとなっている。すなわち、ピーク性能 300 TFlops の Cray 社製 XE-6 **(Camphor)**、243 TFlops の Appro 社製 GreenBlade 8000 **(Laurel)**、および 10.6 TFlops の Appro 社 製 2548X クラスタ **(Cinnamon)** に、これらと InfiniBand QDR で結合された 5PB の大容量ストレージを加えた新システムは、最新のプロセッサ、メモリ、ネットワーク、ディスクなどのハードウェア要素技術を用いることで、旧システムに比べて演算性能で約 8 倍、メモリ容量で約 6 倍、ディスク容量で約 6 倍という、大幅な性能・規模の向上を達成したものとなっている。さらに 2014 年度には、ピーク性能 400 TFlops の Cray 社製 Cascade と 3 PB のストレージが加わり、ピーク性能総計で約 1 PFlops のシステムとなる予定である。

以下本稿では、まずシステムの設計方針について概説し、続いてハードウェアとソフトウェアの両面についてシステムの構成を紹介する。

2　設計方針

旧システムの中核である T2K オープンスパコン HX600 クラスタは、コモディティ技術を基本とするハードウェアアーキテクチャのオープン性、オープンソースを基本とするソフトウェアスタックのオープン性、および研究室の計算サーバからの連続性を担保して幅広い応用を受け入れるユーザニーズへのオープン性の、3 本の柱に基づいて設計された。新システムでも、たとえば計算サブシステムのプロセッサはいずれも x86 アーキテクチャに基づき、OS は全て Linux ベースであるなど、この 3 本柱は確実に受け継がれている。

また旧システムの仕様策定は、「選定から創造へ」、「最先端技術を京大センターへ」、「優れた価格・電力・面積／性能比」、「京大センター固有の応用への対応」という、4 つの設計方針に基づいて行ったが [3]、今回の仕様策定でもこれらの方針は堅持されている。特に電力／性能比は仕様の中で最重要項目とし、旧システムの 8 倍以上の性能向上を、ピーク消費電力の増加を 25 %に抑えつつ達成することに成功した。この結果、総合研究 5 号館地下マシンルームの電源設備の厳しい制約をクリアし、利用負担金で支えられる運用経費の大半を占める電気料金の増加を最小限に抑え、さらに関西地区の厳しい電力供給事情の下でも運用を維持できるシステムとすることができた。

また、本センターで稼動している多種多様なプログラムの新システムへのスムースな移行のために、システムの連続性には細心の注意を払いつつ、最先端技術に基づく新たなコンセプトを数多く取り入れて、連続と飛躍がバランスした仕様設計を行った。まず HX600 クラスタからの連続性は、ノード構成やノード間結合網が類似し、かつソフトウェアスタックも基本的に互換である Laurel により担保し、多様な商用ソフトウェアも含めた容易な移行を確保した。しかし、単に HX600 クラスタのノードや結合網を最新のハードウェアで置換して規模を拡大するだけでは、たとえば結合網の規模が過大となるといった問題が生じ、また電力性能比の観点でも飛躍的な向上が望めないことも明らかであった。

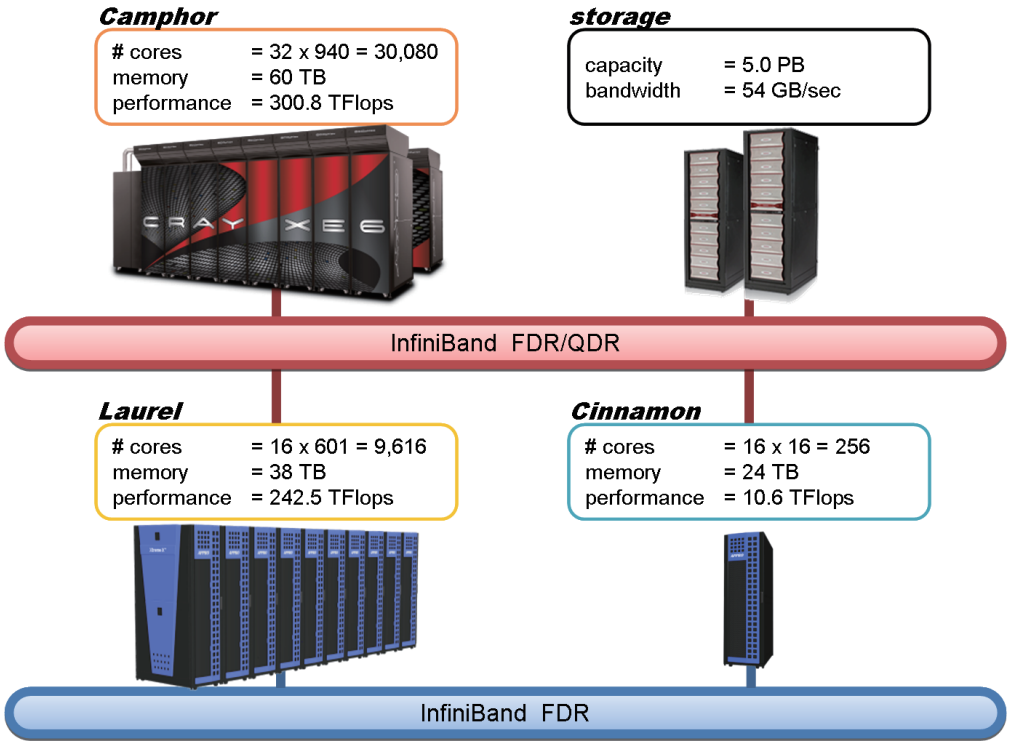


図 1: システム構成

そこで、ノードのコア数やメモリ容量は HX600 と同等以上としつつ、より大規模な構成が可能な結合網や、大規模並列処理に特化したソフトウェアスタックを有する Camphor を加えることで、いわゆるクラスタでは達成困難な規模・性能を実現することとした。すなわち Camphor は、MPI/OpenMP で記述された in-house あるいはオープンソースの大規模並列プログラムのためのプラットフォームであり、HX600 で開発されたソフトウェアや応用をより高いレベルへ発展させることを目的としている。また Laurel においても、601 ノード中の 64 ノードに最近注目を集めている GPGPU を備えることで、先進的なユーザが GPU プログラミングに挑戦できる環境を整えた。

一方、前々世代の HPC2500 から前世代の SPARC Enterprise M9000 と続く、いわゆる fat node の系譜は、これまでの特殊なハードウェア技術からの脱却とコモディティ技術の大胆な導入によって継続することとした。すなわち Cinnamon は、Laurel と同じプロセッサ技術を用いつつ、最先端の 32 GB メモリを多数搭載することで、M9000 を超えるノードあたり 1.5 TB という大容量のメモリ空間を提供するクラスタであり、分散メモリでは実現困難な大容量共有メモリプログラミングの連続性を担保している。それと同時に、コモディティ技術の利用によって、M9000 と同等以上の演算性能とメモリ容量を、約 1/10 の電力と設置面積で達成できている。

3 システム構成

新システムは図 1 に示すように、Campphor, Laurel, Cinnamon の 3 つの演算サブシステムと、ディスク容量 5 PB のストレージシステム、およびそれらを結合する InfiniBand FDR/QDR 結合網などから構成されている。以下、各演算サブシステムとストレージシステムについて概説する。

3.1 Camphor

Camphor は図 2 に示すように、2 基の 16 コアプロセッサ (AMD Opteron “Interlagos”) を中心に構成されたノードを、Gemini ルータを介して 3 次元トーラス状に 940 ノード結合したものである。各プロセッサは 2 つの 8 コアモジュールに分割されており、各コアに 16 KB の 1 次データキャッシュが、2 コアから

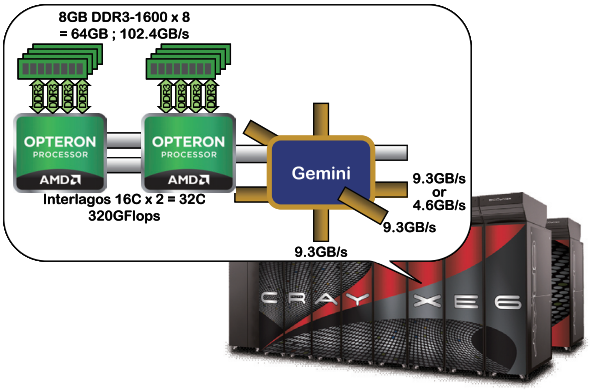


図 2: Camphor の構成

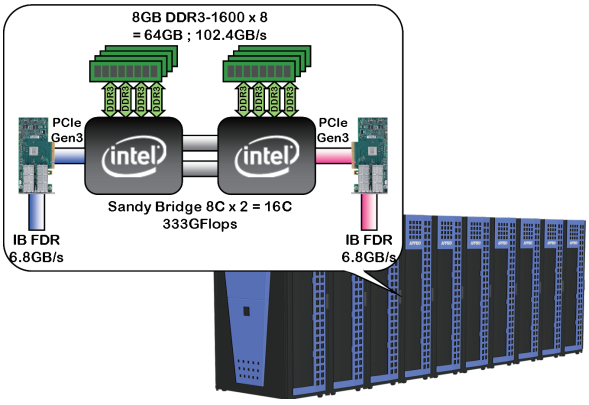


図 3: Laurel の構成

なる “Bulldozer Module” (BM) に 64 KB の 1 次命令キャッシュと 2 MB の 2 次キャッシュが、また 8 コアモジュールに 8 MB の 3 次キャッシュが、それぞれ備えられている。各々の BM は、2 つの倍精度浮動小数点積和演算を同時に実行可能な SIMD 演算器を 2 個有しており、最大演算実行数 8 FLOP に動作周波数 2.5 GHz を乗じたピーク演算性能は 20 GFlops となる。したがって、BM を 8 個有するプロセッサあたりでは 160 GFLops、ノードあたりでは 320 GFlops という、HX600 の 2 倍以上の高い演算性能が達成される。

各 8 コアモジュールには、DDR3-1600 テクノロジの高速メモリ 16 GB が直結され、モジュールとの間のピーク転送性能は 25.6 GB/s に達する。したがってノードのメモリ容量は 64 GB, 総転送性能は 102.4 GB/s となり、それぞれ HX600 の 2 倍、2.5 倍と演算性能向上にバランスした容量・性能となっている。またノード上の 4 つの 8 コアモジュールと Gemini ルータは HyperTransport-3 のリンクで接続され、ノード内の高速な共有メモリアクセスと高速・低遅延のノード間通信が実現されている。

Gemini ルータは 2 ノードに共有され、3 次元トーラスを構成するための 6 本のリンクで隣接するノードペアと結合される。このノードペアを 2 個搭載したブレードが物理的な実装単位であり、x および z 方向の全てのリンクと、ブレード上で隣接する y 方向のリンクの転送速度が 1 方向あたり 9.3 GB/s、ブレード間を繋ぐ y 方向リンクの転送速度が 4.6 GB/s となっている。1 つのキャビネットには 24 枚のブレード (96 ノード) が搭載され、10 キャビネット/940 ノードで構成される Camphor 全体のピーク演算性能は 300 TFlops、メモリ容量は 58.75 TB、バイセクションバンド幅は 1.7 TB/s となっている。また Linpack 性能は 239.41 TFlops であり、2012 年 6 月の Top500 ランキングでは 73 位に位置している。

3.2 Laurel

Laurel は図 3 に示すように、2 基の 8 コアプロセッサ (Intel Xeon “SandyBridge”) を中心に構成されたノードを、InfiniBand FDR 結合網を介して 601 ノード結合したものである。各コアには 32 KB の 1 次命令キャッシュ、32 KB の 1 次データキャッシュ、および 256 KB の 2 次キャッシュが備えられているほか、全コアが共有する 20 MB の L3 キャッシュも備えられている。また各コアは、4 つの倍精度浮動小数点積和演算を同時に実行可能な SIMD 演算器を有しており、最大演算実行数 8 FLOP に動作周波数 2.6 GHz を乗じたピーク演算性能は 20.8 GFlops となる。したがって、プロセッサあたりでは 166 GFLops、ノードあたりでは 333 GFlops という、HX600 の 2 倍以上の高い演算性能が Laurel ノードにおいても達成されている。

各プロセッサには、DDR3-1600 テクノロジの高速メモリ 32 GB が直結され、プロセッサとの間のピーク転送性能は 25.6 GB/s に達する。したがってノードのメモリ容量は Camphor ノード同様に 64 GB, 総転送性能は 102.4 GB/s であり、やはり演算性能の向上とメモリ容量・性能の向上とのバランスが保たれている。またノード上の 2 つのプロセッサは QuickPath のリンクで接続され、ノード内の高速な共有メモリアクセスが実現される。

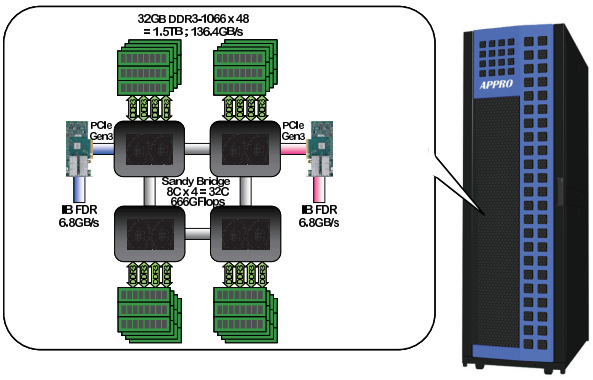


図 4: Cinnamon の構成

各プロセッサは最先端の I/O バスである PCI Express Gen3 のインタフェースを備えており、これを経由した 2 本の InfiniBand FDR リンクでノード間は接続されている。したがってノードあたりの転送性能は 1 方向あたり 6.8 GB/s × 2 = 13.6 GB/s であり、HX600 の 1.7 倍の性能となっている。この 2 本のリンクは個別に、36 ポートスイッチと 324 ポートスイッチから構成される 2 階層の fat-tree 結合網に接続されている。すなわち、耐故障性に優れた 2 系統の独立した結合網が構成されており、同時に 3.1 TB/s の高いバイセクションバンド幅も実現できている。

Laurel の物理的構成単位は 1 ノードからなるブレードであり、601 ノードの内 537 ノードのブレードは高さ 8U のサブラックに 16 枚搭載され、1 キャビネットには 3 サブラック・48 ノードが搭載されている。また残りの 64 ノードには、ピーク性能 664 GFlops の GPGPU である nVIDIA 社の Tesla M2090 がそれぞれ 1 個ずつ備えられ、それぞれ 8 ブレードを搭載するサブラック 4 個に実装されている。Laurel 全体は 13 ラックに実装され、ピーク演算性能は CPU 分が 200 TFlops、GPGPU 分が 42.5 TFlops、総メモリ容量は 37.56 TB となっている。また Linpack 性能は 135.40 TFlops であり、2012 年 6 月の Top500 ランキングでは 126 位に位置している。

3.3 Cinnamon

Cinnamon は、Camphor や Laurel に比べると小規模ではあるが、そのノード構成は図 4 に示すようにユニークなものとなっている。まず各ノードは 4 基の SandyBridge プロセッサを備えているが、この最新技術に基づく構成を実現したのは本センターが世界で初めてである。また各プロセッサには、これも最先端の 32 GB DDR3 DIM を 12 モジュール用いた 384 GB のメモリが接続され、ノード全体で 1.5 TB という大容量共有メモリを実現している。

各ノードには、Laurel ノードと同様に InfiniBand FDR のリンクが 2 本備えられ、やはり独立した 2 系統のスイッチで Cinnamon の全 16 ノードが接続されている。Cinnamon 全体のピーク性能は 10.6 TFlops、総メモリ容量は 24 TB であり、旧システムの M9000 クラスタと比べてそれぞれ 1.2 倍、3.4 倍の向上となっているが、ノードあたり 2U のコンパクトな実装により、1 キャビネットに全てが収まる省スペース・省電力構成となっている。

3.4 ストレージシステム

ストレージシステムの構成単位は、280 個の 3 TB 大容量 SATA ドライブからなる DDN 社の SFA 10000 Lustre ディスクアレイに、4 台の OSS を結合したものである。このユニット 6 台の並列構成とすることで、物理容量 5 PB、8+2 の RAID-6 構成での実効容量 3.5 TB の大容量ファイルスペースを実現している。各ユニットは 4 本の InfiniBand QDR のリンクを有し、Camphor/Laurel/Cinnamon とは総計 24 リンク、実効転送性能 54 GB/s の高速結合網で接続されている。またメタデータは、5+5 の RAID-10 構成の高信頼ディスクアレイに格納され、MDS も 2 サーバによる冗長構成とすることで、ファイルシステムの信頼性を確保している。

3.5 ソフトウェアスタック

新システムのソフトウェアスタックは図 5 に示す構成であり[1]、基本的には旧システムと同じプログラム開発環境が提供され、Laurel と Cinnamon のユーザには商用アプリケーションも同等のものが用意されている。なお、旧システム固有の科学技術計算ライブラリ SSL-II の主要な機能を他のライブラリを用いて実現した KUML を開発するなど、できる限り互換性を保つ工夫を随所で行っている。

また旧システムで確立した、契約資源量（ノードまたはコア数）に基づくジョブスケジューリングを

[1] 一部のソフトウェアは、ライセンスの関係で京都大学所属のユーザにのみ提供される。

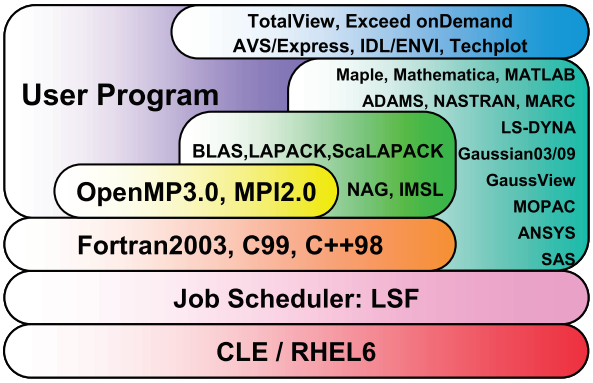


図 5: ソフトウェアスタック

継承するとともに、複数の研究グループが共有するジョブキューにおけるフェアシェアスケジューリングなど、より高度なスケジューリング機能も取り入れている。

4　おわりに

本稿では、新スーパーコンピュータシステムの構成を紹介し、旧システムに比べて飛躍的に性能や規模が向上したことを述べた。特に利用負担金に直結する電力／性能比の向上を、仕様策定での最重要課題として取り組んだ結果、別稿で述べるように負担金あたりの演算性能、メモリ容量、ファイル容量は 4 ～ 5 倍に向上し、日常的に大規模並列計算が可能な環境を構築できたと自負している。

新システムの利用や運用については、供給ベンダーが代わったこともあって、ご不便をお掛けしている点も少なからずあるが、システムの成熟に向け運用スタッフ一丸となって鋭意努力を続けている。この努力の源泉としてユーザ各位からの忌憚のないご意見を今後も引き続きお願いするとともに、スタッフの奮闘を暖かく見守っていただくことも併せてお願いしたい。

参考文献

[1] Hiroshi Nakashima. T2K Open Supercomputer: Inter-University and Inter-Disciplinary Collaboration on the New Generation Supercomputer. In *Intl. Conf. Informatics Education and Research for Knowledge-Circulating Society*, pp. 137–142, January 2008. Invited Talk.

[2] 石川裕, 中島研吾, 中島浩, 朴泰祐. T2K オープンスパコンが創る新しい計算機環境. 計算工学, Vol. 14, No. 1, January 2009.

[3] 中島浩. 新スーパーコンピュータシステムの構成について. 学術情報メディアセンター全国共同利用版広報, Vol. 7, No. 1, 2008.

スーパーコンピュータ利用ガイド

新システムを利用するために

山口 倉平　　池田 健二　　斎藤 紀恵　　疋田 淳一

京都大学 情報部

1　はじめに

本センターでは、2012 年 5 月より新しいスーパーコンピュータ（以下、スパコン）のサービスを開始しております。本稿では、システム構成について概説し、利用方法について説明します。

2　システム構成

スパコンは、システム A、システム B、システム C、システム G（システム B の GPU 搭載ノード）の 4 種の計算機群および大規模ストレージで構成しています（図 1）。システム構成の詳細は、本誌掲載の『新スーパーコンピュータシステム―そのコンセプトと構成―』をご覧ください。

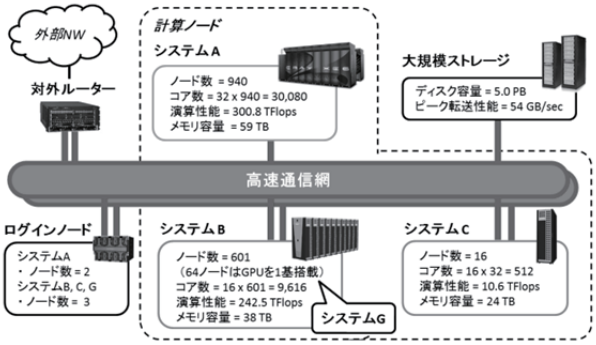


図 1　ノード構成

2.1 ノード設計

● ログインノード

プログラムのコンパイルや会話型ジョブ、バッチジョブの投入に利用します。システム B とシステム C は共通のログインノードとして 3 ノードを割り当てており、DNS ラウンドロビンによる負荷分散を実施しています。システム A は、2 ノードを専用に割り当ており、システム A のサービスを申し込んだ方のみログインを許可しています。

● 計算ノード

会話型ジョブおよびバッチジョブを実行するノードです。サービスの申請状況に応じて付与されるバッチシステムのキューを介し、プログラムを実行することができます。

● 専用クラスタ

専用クラスタは、システム B の一部ノードを切出し、占有して利用できるようにしたサービスで、専用クラスタ内の 1 ノードをログインノードとし、外部ネットワークから直接ログインできるようにグローバルアドレスを割り付けています。

3　スパコンへのログイン

ログインノードのホスト名は、表 1 のようになります。

表 1　ホスト名

システム	ホスト名（FQDN）
A	camphor.kudpc.kyoto-u.ac.jp （システム A の利用者のみログイン可能）
B, G	laurel.kudpc.kyoto-u.ac.jp （全利用者がログイン可能）
C	

以前のシステムでは、スパコンへのログインを SSH（Secure SHell）のパスワード認証により行っていましたが、今回のシステムでは、よりセキュアなサービスを実現するために SSH のパスワード認証を禁止し、鍵認証によるログインに限定しています。スパコンにログインするには、以下の流れで作業を行ってください。

1. SSH クライアントのインストール
2. 鍵ペアの作成（公開鍵、秘密鍵）
3. 利用者ポータルから公開鍵の登録
4. SSH クライアントで秘密鍵を使用してログイン

詳細な手順は次の URL で案内しています。
http://web.kudpc.kyoto-u.ac.jp/manual/ja/login

3.1 X 環境でのログイン

PC X サーバソフトウェアとして Exceed onDemand バージョン 8 を提供しています。Exceed onDemand は、データ圧縮技術により通信トラフィックを抑えているため、遠隔地でも快適に X 環境を利用することができます。インストールや利用方法は、次の URL で案内しています。
http://web.kudpc.kyoto-u.ac.jp/manual/ja/login/eod

4 ファイルシステム

4.1 ホームディレクトリ（$HOME）

ホームディレクトリのパスは、図 2 に示すように、/home の後に利用者番号の先頭のアルファベット一文字、次に利用者番号となっています。

利用可能な領域として、利用者あたり 60GB の容量を割り当てています。そのうち半分をバックアップ領域としており、実際に利用可能な領域は 30GB となります。システム A、B、C で共通の領域となっており、全てのログインノード、計算ノードから同じパスでアクセスすることが可能です。

図 2 ホームディレクトリ

4.2 大容量ディスク領域（/LARGE）

4.2.1 容量と構成

パーソナル、グループ、専用クラスタの各コースをご利用の方は、大容量ディスク(/LARGE)を利用できます。割り当て容量は各コースの資源により、表 2 に示す値の通り決まっています(バックアップ領域を含む)。

表 2 コース毎の/LARGE 容量

コース	/LARGE 容量
パーソナル	1TB
グループ タイプ A1、B1、G1	1 ノードあたり 2TB
グループ タイプ A2、B2	1 ノードあたり 1.2TB
グループ タイプ A3、B3	1 ノードあたり 2TB
グループ タイプ C1	1 ソケットあたり 2TB
グループ タイプ C2	1 ソケットあたり 1.2TB
専用クラスタ	1 ノードあたり 2TB

大容量ディスクは/LARGE0 と/LARGE1 の 2 つの領域で構成されており、初期設定では/LARGE1 はバックアップ領域となっています。

また、図 3、図 4 に示すように、/LARGE*の直下にユーザ名、グループ名のディレクトリを配置しています。

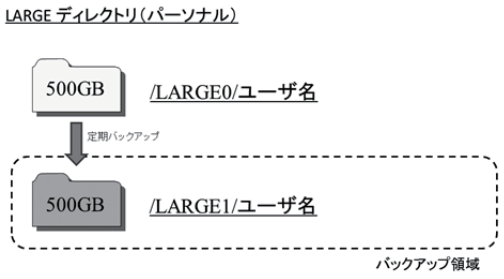


図 3 パーソナルの大容量ディスク

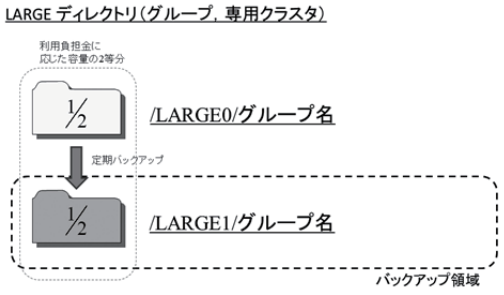


図 4 グループ・専用クラスタの大容量ディスク

4.2.2 バックアップ設定

バックアップ領域は、group_backup コマンドによりバックアップを解除することで通常の領域として利用できます。バックアップ設定の変更は、パーソナルコースの場合は利用者本人、グループ・専用クラスタコースの場合はグループ管理者（サービスの申請者）が行うことができます。

● 現在の設定を確認

```
$ group_backup -g gr10000 -l
 Num  Filesystem     Status  Filesystem       Status
  1)  /LARGE0/gr10000..Safe /LARGE1/gr10000 ... Backup
```

● バックアップを解除

```
$ group_backup -g gr10000 -u 1
/LARGE0/gr10000: Safe   => UnSafe
/LARGE1/gr10000: Backup => UnSafe
```

4.3 利用状況の確認（quota）

ディスクの利用状況はquota コマンドで確認することができます。確認できる項目を表 3 に示します。また、-g オプションを使用することで、大容量ディスクの利用状況を確認することができます。

● ホームディレクトリの利用状況

```
$ quota
Disk quotas for user b59999 (uid 59999):
fsys blocks quota limit grace files quota limit grace
 /home 100 30000000 31000000 - 50 600000 700000  -
```

表 3　利用状況の確認項目一覧

項目	内容
blocks	使用中のファイル容量(KB)
quota	ファイル容量／数の制限値(ソフトリミット)
limit	ファイル容量／数の絶対的制限値(ハードリミット)
grace	制限値越え(ソフトリミット超過)の許容期間
files	使用中のファイル数

5　利用環境のカスタマイズ

5.1 環境設定ファイル

標準のログインシェルはtcsh に設定しており、ログイン時に環境変数の設定などを行いたい場合は、.cshrc ファイルに必要な設定を記述することで反映されます。bash を利用する場合には、.bash_profile や.bashrc に設定を記述することで反映されます。

なお、システム A、B、C で、ホームディレクトリが共有されているため、環境設定ファイルは同じものが読み込まれます。複数のシステムを利用している場合、例えば次のように hostname コマンドの結果に応じた条件文を記述することで、各システムに応じた処理を行うことができます。

● .cshrc の例

```
switch(`hostname`)
    case xe*:
        #システム A の場合の処理
        breaksw
    case ap*:
        #システム B/C の場合の処理
        breaksw
endsw
```

● .bashrc の例

```
case `hostname`  in
    xe*)
        #システム A の場合の処理
    ;;
    ap*)
        #システム B/C の場合の処理
    ;;
esac
```

5.2 ログインシェルの変更

ログインシェルは chsh コマンドで変更することができます。対応しているシェルは sh、bash、csh、tcsh、zsh、ksh の 6 種類です。

● ログインシェルを bash に切り替える

```
$  chsh  bash
```

5.3 システムからの通知メール転送

バッチ処理システムからのエラー通知など、システムからの通知メールはシステムのローカルユーザ宛てに送信されます。このメールは mutt コマンドでも確認できますが、いち早く閲覧できるようホームディレクトリに.forward ファイルを作成し、普段利用しているメールアドレスに転送することを推奨

しています。

転送するには、次のコマンドのメールアドレス部分を自身のものに置き換えて実行してください。

- user@example.com にメールを転送

```
$ echo user@example.com > ~/.forward
```

5.4 メッセージの日本語化

SSH クライアントの文字コードを UTF-8 に設定した上で、次の通り環境変数 LANG を設定することで、コマンド出力、コンパイラリンク、オンラインマニュアル(man)などのメッセージを日本語表示に変更することができます。

- 環境変数 LANG を設定する(tcsh の場合)

```
$ setenv LANG ja_JP.UTF-8
```

- 環境変数 LANG を設定する(bash の場合)

```
$ export LANG=ja_JP.UTF-8
```

6 ソフトウェア環境設定ツール

コンパイラ、ライブラリ、アプリケーションを利用する際に必要になる環境設定を行うツールとして、module コマンドを提供しています。あらかじめ用意されているモジュールファイル(実行コマンドやライブラリのパスが定義されたファイル)と呼ばれる定義ファイルを module コマンドにより指定することで、異なるバージョンのソフトウェアを使用する際の、環境設定を簡単に切り替えて利用することができます。module コマンド一覧を表 4 に示します。詳細な使い方は module help と実行していただくことでヘルプが表示されます。

表 4 module コマンド

コマンド	説明
module list	ロード済みモジュールファイルの一覧表示
module avail	使用可能モジュールファイルの一覧表示
module show	モジュールファイルの設定内容表示
module load	モジュールファイルのロード
module unload	モジュールファイルのアンロード
module switch	モジュールファイルの切り替え

メインとなるコンパイラやライブラリはログイン時に自動で設定が行われます。システム A は Cray コンパイラ環境、システム B、C では、Intel コンパイラ環境が設定されます。

- ロードされている module ファイルの確認

```
$ module list
Currently Loaded Modulefiles:
 1) lsf/SystemB   2) intel/12.1   3) intel/impi
```

- コンパイラの切り替え

```
$ module switch intel/12.1 intel/12.0.5
```

ISV アプリケーションを利用する場合は、module avail コマンドにより利用可能なアプリケーションとバージョンを確認し、module load コマンドにより環境設定を行った後で、起動コマンドを実行してください。

- matlab の起動手順

```
$ module load matlab/R2012a
$ xrun matlab
※ xrun コマンドについては次章をご覧ください
```

7 プログラムの実行方法

7.1 ログインノードの利用

ログインノードは、多数の利用者が同時にログインし作業を行うため、エディタを使ったプログラミングやコンパイル、小規模な逐次処理のプログラム実行に限定して利用してください。過度な負荷がかかることを防止するために、表 5 に示すプロセスリミットを設定しています。

表 5 ログインノードのプロセスリミット

項目	初期値	最大値
CPU 時間(CPU Time)	1 時間	20 時間
メモリサイズ(Virtual memory)	2GB	4GB

tcsh を利用している場合は、limit コマンド、unlimit コマンドを使ってプロセスリミットの設定確認・変更を行うことができます。

- 設定確認(tcsh)

```
$ limit
cputime       1:00:00
vmemoryuse    2097152 kbytes
```

- 最大値まで拡張(tcsh)

```
$ unlimit
```

bash を利用している場合は、ulimit コマンドを使って設定確認・変更を行うことができます。

● 設定確認(bash)

```
$ ulimit -a
cpu time          (seconds, -t)  3600
virtual memory    (kbytes, -v)  2097152
```

● 最大値まで拡張(bash)

```
$ ulimit -t 72000
$ ulimit -v 4194304
```

7.2 会話型ジョブの利用

会話型ジョブは通常のコマンド実行のように、対話的にジョブを実行する形式です。プログラム自体は計算ノード上で実行されますが、ログインしているターミナル上にプログラムの出力内容がすぐに返されるため、プログラムのデバッグや対話的に利用するアプリケーションを実行する際にご利用ください。

7.2.1 会話型ジョブ用の計算ノード

会話型ジョブ専用の計算ノードとして、システムAは12ノード、システムBは12ノード、システムCは1ノード割り当てています。全利用者で共用するため、一部のユーザが占有しないように利用可能な資源に制限をかけています。制限値を表 6、表 7に示します。

表 6 会話型ジョブ用計算ノードの制限値（標準）

	システムA	システムB	システムC
コア数	1	1	1
メモリ	1920MB	3840MB	45GB
CPU 時間	制限なし	1時間	
経過時間	1時間	8時間	

表 7 会話型ジョブ用計算ノードの制限値（最大）

	システムA	システムB	システムC
コア数	32	16	8
メモリ	60GB	60GB	350GB
CPU 時間	制限なし	20時間	
経過時間	20時間	24時間	

7.2.2 tssrun コマンド

会話型ジョブを実行するには tssrun コマンドを利用します。MPI プログラムを実行する場合は、システムAとシステムB、Cで利用方法が異なりますのでご注意ください。

● tssrun コマンドの利用方法

```
tssrun [-q queue] [-A p=x:t=x:c=x:m=x] ¥
 [-C cpulimit] [-W elapsetime] ./a.out
※ オプションの詳細は表 9、表 10 を参照
```

● 【共通】逐次実行プログラム

```
$ tssrun ./a.out
```

● 【共通】OpenMP プログラム（4 スレッド）

```
$ tssrun -A p=1:t=4:c=4 ./a.out
```

● 【システムA】MPI プログラム（8 プロセス）

```
$ tssrun -A p=8 ./a.out
```

● 【システムB、C】MPI プログラム（16 プロセス）

```
$ tssrun -A p=16 mpiexec.hydra ./a.out
※ mpiexec.hydra コマンドを介してプログラムを起動
```

7.2.3 xrun コマンド

Exceed onDemand によりログインした状態で、xrun コマンドを利用することで、GUI のアプリケーションを計算ノードで実行させることができます。コマンドの利用方法は tssrun と同様です。

7.2.4 パーソナル、グループコース用のバッチキューの利用

tssrun および xrun コマンドは、会話型ジョブ用の計算ノードでプログラムを実行しますが、-q オプションにより、グループコースやパーソナルコースで利用可能なバッチキューを指定することで、前述の計算ノードの制限を受けずに、本格的なジョブ実行を対話的に行うことができます。

7.3 バッチジョブの利用

バッチ処理を実現するための、ジョブスケジューリングソフトウェアとして、LSF を導入しています。

LSF にバッチジョブを投入するには、実行したいプログラムを記述したシェルスクリプト（ジョブス

クリプト）を作成し、専用のコマンドによってスクリプトの実行依頼を行います。

スパコンの計算資源は有限であるため、大規模な計算や本格的な実行は、バッチジョブで行っていただくことが基本となります。

7.3.1 バッチキュー名と投入権限

利用可能なバッチキューは申し込んだサービスコースにより決まります。キューの種類を表 8 に示します。

表 8　キューの一覧

キュー名	利用条件
eb	すべての利用者が利用可能
p{a|b|c}	パーソナルコース利用者
グループ名 + {a|b|c|g}	グループコース利用者
t{a|b|c}	会話型ジョブ用
priority	障害時のリスタート専用(投入不可)

7.3.2 ジョブスクリプト

ジョブスクリプトは、原則的には bash スクリプトとして記述してください。スクリプトはジョブ投入オプションを記述した LSF オプション領域と実行するプログラムを記述したユーザプログラム領域から構成されます。

```
#!/bin/bash
#============ LSF Options ============
#QSUB -q gr10000b                  #キュー指定
#QSUB -ug gr10000                  #実効グループ指定
#QSUB -W 2:00                      #経過時間上限指定
#QSUB -A p=1:t=1:c=1:m=1920M   #計算資源量の指定
#============ Shell Script ============
./a.out
```

LSF ジョブスクリプトのサンプルを次の Web ページで公開しています。サンプルをベースに LSF オプションの修正や必要なスクリプトを追記してください。

- システム A
  http://web.kudpc.kyoto-u.ac.jp/manual/ja/run/batchjob/systema
- システム B、C
  http://web.kudpc.kyoto-u.ac.jp/manual/ja/run/batchjob/systembc

7.3.3 LSF オプション

ジョブスクリプトのオプション領域にて、#QSUB 句の後に LSF オプション指定することで、ジョブの属性を設定することができます。LSF の主要なオプションを表 9 に示します。

表 9　LSF オプション

オプション	説明
-q *QUEUENAME*	キュー指定
-ug *GROUPNAME*	実効グループ指定
-W *HOUR:MINUTE*	経過時間上限値指定
-A p=*X*:t=*X*:c=*X*:m=*X*	計算資源量の指定
-c *HOUR:MINUTE*	CPU 時間上限値指定
-rn	障害発生時のジョブ再実行禁止

また、-A オプションで指定できる計算資源量を表 10 に示します。

表 10　LSF オプション（-A の詳細）

オプション	説明
p=procs	プロセス数　e.g.　p=8
t=threads	プロセスあたりのスレッド数　e.g. t=16
c=cores	プロセスあたりのコア数（基本的に t と同じ値を指定）　e.g. c=16
m=memory	プロセスあたりのメモリ容量(単位:M,G,T)　e.g.　m=1920M

7.3.4 ジョブスクリプトで利用可能な環境変数

ジョブスクリプト上で利用可能な環境変数のうち、代表的なものを表 11 に示します.

表 11　ジョブスクリプトで利用可能な環境変数

環境変数名	説明
LSB_JOBID	当該ジョブのジョブ ID
LSB_QUEUE	ジョブを投入したキュー
LS_SUBCWD	ジョブを投入したカレントディレクトリ
LSB_PROCS	ジョブ実行時のプロセス数（-A オプションの p の値）
LSB_THREADS	ジョブ実行時のスレッド数（-A オプションの t の値）
LSB_CPUS	ジョブ実行時のコア数（-A オプションの c の値）
LSB_MEMORY	ジョブ実行時のメモリ容量（-A オプションの m の値）
LSB_PPN	ジョブ実行時のノードあたりのプロセス数（-A の値から自動で算出、システム A のみ）

7.3.5 プログラムの実行方法

シェルスクリプト領域に実行したいプログラムを記述しますが、システム A とシステム B、C でプログラム実行方法が異なりますのでご注意ください。

- システム A

逐次実行、OpenMP、MPI ともに、次のように aprun コマンドを介してプログラムを起動する必要があります。aprun を利用しないと計算ノード上で正しくプログラムが起動しません。

```
aprun -n $LSB_PROCS -d $LSB_CPUS ¥
 -N $LSB_PPN ./a.out
```

表 12　aprun コマンドのオプション

オプション	説明
-n procs	プロセス数
-d cores	コア数
-N ppn	ノードあたりのプロセス数

- システム B、 C

逐次実行、OpenMP の場合は直接実行するプログラム名を記述してください。MPI プログラムの場合は mpiexec.hydra コマンドを介してプログラムを起動します。起動するプロセス数や計算ノードの情報は、mpiexec.hydra が LSF から直接データを取得するので、明示的な指定は不要です。

```
## 逐次、OpenMP プログラムの場合
./a.out
## MPI プログラムの場合
mpiexec.hydra ./a.out
```

7.3.6 ジョブ投入できるキューの確認（qstat）

ジョブを投入できるキューを確認するためには、qstat コマンドを使用します。表示される項目のうち、MAX 列はキューが利用できる CPU コア数の上限を表します。また NJOBS、PEND、RUN は、それぞれ投入ジョブ数、実行待ちジョブ数、実行中ジョブ数を表します。

```
$  qstat
QUEUE_NAME PRIO STATUS MAX JL/U JL/P JL/H NJOBS
PEND RUN SUSP
eb 30 Open:Active 16 32  - - 0 0 0 0
gr10000b 30  Open:Active 4096 4096 - - 0 0 0 0
```

7.3.7 ジョブ実行(qsub)

キューにジョブを投入するためには、qsub コマンドを使用します。qsub コマンドにジョブスクリプトファイルを指定しジョブを依頼すると、システムからジョブ ID が発行されます。

```
$ qsub sample.sh
Job <5610> is submitted to queue <gr10000b>.
```

7.3.8 ジョブの状態を確認する(qjobs)

投入したジョブの状態を確認するためには、qjobs コマンドを使用します。投入したジョブのうち、実行中(RUN)、実行待ち(PEND)のジョブが表示されます。

```
$  qjobs
JOBID USER STAT QUEUE FROM_HOST EXEC_HOST JOB_NAME
SUBMIT_TIME
5610 b59999 RUN gr10000b gb-0001 gb-0200 ./a.out
May 1 00:01
```

7.3.9 ジョブの詳細情報を確認する(qs)

投入したジョブの詳細情報を確認するためには、qs コマンドを使用します。

```
$ qs
QUEUE USER JOBID STATUS PROC THRD CORE MEM
ELAPSE(limit)
gr10001b b59999 123 RUN 4 8 8 1920M 00:06(01:00)
```

7.3.10 ジョブスクリプトの内容を確認する(qcat)

実行中のジョブについて、ジョブスクリプトの内容を確認するためには、qcat コマンドと-s オプションを使用し、確認したいジョブの ID を指定します。

```
$ qcat -s 5610
#QSUB -q gr10000b
#QSUB -ug gr10000
#QSUB -W 5:0
#QSUB -A p=1:t=4:c=4:m=2G

./a.out
```

7.3.11 ジョブの途中経過を確認する(qcat)

実行中のジョブについて、ジョブの途中経過を確認するためには、qcat コマンドを使用します。ジョブの標準出力の表示は-o オプションを、標準エラー出力の表示は-e オプションを使用し、確認したいジョブの ID を指定します。

```
$ qcat -o 5610
Tue May 1 00:00:01 JST 2012
Subroutine A step1 finished
Subroutine A step2 finished
Subroutine A step3 finished
```

```
$ qcat -e 5610
Tue May 1 00:00:01 JST 2012
STDERR 1
```

7.3.12 グループのジョブを確認する(qgroup)

グループコースのキュー利用状況は、qgroup コマンドで確認できます。表示される項目の内容を表 13 に示します。

```
$ qgroup
QUEUE SYS|RUN PEND OTHER|ALLOC(MIN/STD/MAX)
-----------------------------------------
gr10000b B | 1 0 0| 80( 32/ 64/128)
gr10001b B | 0 0 0| 0( 64/256/512)
QUEUE USER|RUN(ALLOC)PEND(REQUEST)
OTHER(REQUEST)
-----------------------------------------
-
gr10000b b59999|1( 80) 0( 0) 0( 0)
```

表 13 qgroup コマンドで表示される項目一覧

項目	説明
RUN、PEND、OTHER	ジョブの件数
ALLOC	割当コア数
REQUEST	要求コア数
MIN	キューに対する最低保障コア数
STD	キューに対する標準コア数
MAX	キューに対する最大コア数

7.3.13 投入したジョブをキャンセルする(qkill)

投入したジョブをキャンセルするためには、qkill コマンドを使用します。キャンセルしたいジョブの ID を引数に指定することで、実行中、実行待ちに関わらずキャンセルすることができます。

```
$ qkill 5610
Job <5610> is being terminated
```

7.3.14 実行結果の確認

実行が終了すると、LSF の結果ファイルが qsub コマンドを実行したディレクトリに作成されます。デフォルトで作成される出力ファイルは、表 14 に示す形式で作成されます。

表 14　ジョブ実行結果ファイル

システム	内容	ファイル名
A	標準出力・ジョブ情報	A*mmddhh*.o*XXXXXX*
	標準エラー出力	A*mmddhh*.e*XXXXXX*
B	標準出力・ジョブ情報	B*mmddhh*.o*XXXXXX*
	標準エラー出力	B*mmddhh*.e*XXXXXX*
C	標準出力・ジョブ情報	C*mmddhh*.o*XXXXXX*
	標準エラー出力	C*mmddhh*.e*XXXXXX*
G	標準出力・ジョブ情報	G*mmddhh*.o*XXXXXX*
	標準エラー出力	G*mmddhh*.e*XXXXXX*

mmddhh : ジョブが開始された月・日・時

XXXXXX : ジョブ ID

8　プログラム開発環境

システム A と B、C では、利用できるコンパイラが異なります。システム A のメインコンパイラは Cray コンパイラ、サブコンパイラは Intel コンパイラです。システム B、C でのメインコンパイラは Intel コンパイラです。ログインした際にメインコンパイラ環境がデフォルトでご利用いただけるよう設定されています。

8.1 システム A の場合

8.1.1 コンパイルコマンドとライブラリ

Cray コンパイラでサポートしているプログラミング言語とコンパイルコマンドを表 15 に、サポートしているライブラリを表 16 に示します。

システム A で Intel コンパイラを利用する場合、module コマンドを実行し、コンパイラ環境を切り替えた上で、コンパイルコマンドを実行してください。Cray コンパイラ、Intel コンパイラどちらも同じコンパイルコマンド(ftn、cc、CC)で実行することができます。

表 15　プログラミング言語とコンパイルコマンド

言語	規格・バージョン	コマンド
Fortran	ISO/IEC 1539-1:2004 (Fortran 2003)	ftn
C	ISO/IEC 9899:1999(C99)	cc
C++	ISO/IEC 14882:2003	CC

表 16　MPI、数値計算ライブラリ

ライブラリ	サポート
MPI	MPICH2 (MPI2.0)
数値計算ライブラリ	LibSCi (BLAS,LAPACK,SCALAPACK) ACML, IMSL

8.1.2 Fortran

Fortran のコンパイル・リンクは ftn コマンドを利用します。

● Fortran でのコンパイル・リンク

```
$ ftn sample.f90
```

Cray コンパイラはデフォルトで高い最適化を行うため、最初はオプションなしでコンパイル・リンクすることを推奨します。

● 数値計算ライブラリのリンク例

```
#ACML を使用する場合
$ ftn -h noomp sample.f90 -lacml
#IMSL を使用する場合
$ module load imsl/fnl701-cce
$ ftn $F90FLAGS sample.f90 $LINK_FNL
```

LibSci ライブラリは、オプション指定なしに自動でリンク処理を行うため、コンパイル時のライブラリ指定は不要です。

8.1.3 C/C++

C のコンパイル・リンクは、cc コマンド、C++のコンパイル・リンクは、CC コマンドを利用します。

● C でのコンパイル・リンク

```
$ cc sample.c
```

● C++でのコンパイル・リンク

```
$ CC sample.cpp
```

● 数値計算ライブラリのリンク例

```
#ACML を使用する場合
$ cc -h noomp sample.c -lacml
```

8.1.4 コンパイルオプション

Cray コンパイラがサポートする主なオプションを表 17 に、メッセージ出力とデバッグのオプションを表 18 に、Fortran 言語固有のオプションを表 19 に示します。

表 17　主なオプション

オプション	説明
-o *FILENAME*	オブジェクトファイルの名前を指定します
-O{0|1|2|3}	最適化のレベルを指定します （デフォルトは 2）
-h omp	OpenMP 指示子を有効にしてコンパイルします（デフォルトで有効）
-h noomp	OpenMP 指示子を無効にしてコンパイルします
-h autothread	自動並列化を行います
-h byteswapio	ビッグエンディアン形式のファイルをサポートします
-h pic –dynamic	2GByte を超えるメモリをサポートします

表 18　メッセージ出力とデバッグのオプション

オプション	説明
-ra	コンパイル時のレポートを作成します
-m2	すべての警告メッセージを表示します
-h msgs	実施した最適化についての情報を表示します
-h negmsgs	実施しなかった最適化についての情報を表示します
-Rb	配列の領域外参照を検出します

表 19　Fortran 言語固有オプション

オプション	説明
-f fixed	プログラムが固定形式で記述されていることを指示します
-f free	プログラムが自由形式で記述されていることを指示します
-em	モジュールを有効にします

8.1.5 自動並列化

ftn、cc、CC のコンパイルコマンド実行時に、-h autothread オプションを指定することで、コンパイラが自動的にプログラムを並列化します。

- 自動並列化のコンパイル・リンク

```
$ ftn -h autothread sample.f90
```

8.1.6 OpenMP

OpenMP は、共有メモリ型並列計算機における並列プログラミングの標準化インターフェイスです。Fortran では!$OMP、C/C++では#pragma omp で始まる指示子をユーザがソースプログラム中に挿入し、-h omp オプションを指定することで、プログラムを並列化します。

- OpenMP のコンパイル・リンク

```
$ ftn -h omp sample.f90
```

8.1.7 MPI

MPI(Message Passing Interface)は、並列処理アプリケーション用メッセージパッシングライブラリです。システム A では MPI プログラムも ftn、cc、CC の各コマンドでコンパイルを行います。

- MPI のコンパイル・リンク

```
$ ftn sample_mpi.f90
```

8.1.8 Intel コンパイラへの切り替え

Cray コンパイラから Intel コンパイラに切り替えるには、次のコマンドを実行してください。

```
$ module switch PrgEnv-cray PrgEnv-intel
```

8.2 システム B、C の場合

8.2.1 コンパイルコマンドとライブラリ

Intel コンパイラでサポートしているプログラミング言語とコンパイルコマンドを表 20 に、サポートしているライブラリを表 21 に示します。

表 20　プログラミング言語とコンパイルコマンド

言語	規格・バージョン	コマンド
Fortran	ISO/IEC 1539-1:2004 (Fortran 2003)	ifort
C	ISO/IEC 9899:1999(C99)	icc
C++	ISO/IEC 14882:1998	icpc

表 21　MPI、数値計算ライブラリ

ライブラリ	サポート
MPI	Intel MPI(MPI-2.2)
数値計算ライブラリ	MKL、IMSL、NAG

8.2.2 Fortran

Fortran のコンパイル・リンクは ifort コマンドを利用します。

- Fortran でのコンパイル・リンク

```
$ ifort sample.f90
```

- 数値計算ライブラリのリンク例

```
#MKL を使用する場合
$ module load intel/mkl
$ ifort sample.f90 -mkl

#IMSL を使用する場合
$ module load imsl
$ ifort $F90FLAGS sample.f90 $LINK_FNL

#NAG を使用する場合
$ module load nag/smp/22_intel
$ ifort sample.f90 $FLINK
```

8.2.3 C/C++

C のコンパイル・リンクは icc コマンド、C++のコンパイル・リンクは icpc コマンドを利用します。

- C でのコンパイル・リンク

```
$ icc sample.c
```

- C++でのコンパイル・リンク

```
$ icpc sample.cpp
```

- 数値計算ライブラリのリンク例

```
#MKL を使用する場合
$ module load intel/mkl
$ icc sample.c -mkl

#NAG を使用する場合
$ module load nag/smp/22
$ icc sample.c $FLINK
```

8.2.4 コンパイルオプション

システム B、C のメインコンパイラである Intel コンパイラがサポートする主なオプションを表 22 に、メッセージ出力とデバッグのオプションを表 23 に、Fortran 言語固有のオプションを表 24 に示します。

表 22 主なオプション

オプション	説明
-o *FILENAME*	オブジェクトファイルの名前を指定します
-fast	プログラムの速度が最大になるように最適化します
-O{0|1|2|3}	最適化のレベルを指定します (デフォルトは 2)
-ipo	複数ファイル間で、手続き間の処理を最適化します
-openmp	OpenMP 指示子を有効にしてコンパイルします
-parallel	自動並列化を行います
-mcmodel=medium -shared-intel	2Gbyte を超えるメモリをサポートします
-convert big_endian	ビッグエンディアン形式のファイルをサポートします

表 23 メッセージ出力とデバッグのオプション

オプション	説明
-opt-report	実施した最適化についての情報を表示します
-par-report	実施した自動並列化についての情報を表示します
-vec-report	実施したベクトル化についての情報を表示します

表 24 Fortran 言語固有オプション

オプション	説明
-nofree	プログラムが固定形式で記述されていることを指示します
-free	プログラムが自由形式で記述されていることを指示します
-warn all	すべての警告メッセージを表示します
-CB	配列の領域外参照を検出します

8.2.5 自動並列化機能

ifort、icc、icpc のコンパイルコマンド実行時に、-parallel オプションを指定することで、コンパイラが自動的にプログラムを並列化します。

- 自動並列化のコンパイル・リンク

```
$ ifort -parallel sample.f90
```

8.2.6 OpenMP

コンパイル時に-openmp オプションを指定することで、プログラムを並列化します。

● OpenMP のコンパイル・リンク

```
$ ifort -openmp sample.f90
```

8.2.7 MPI

システム B、C では MPI プログラムのコンパイルに、mpiifort (Fortran)、mpiicc (C)、mpiicpc (C++) を使用します。

● MPI のコンパイル・リンク

```
$ mpiifort sample_mpi.f90
```

9 ISV アプリケーション

スパコンを用いた大規模な科学技術計算、および計算化学、構造解析、統計解析、可視化などのために様々なアプリケーションソフトウェアのサービスを提供しています(表 25～表 31)。利用できるアプリケーションはシステム毎に異なります。利用者の所属により利用が制限される場合がありますので、詳細は次の URL でご確認ください。
http://web.kudpc.kyoto-u.ac.jp/manual/application

表 25 可視化・図形処理アプリケーション

名称	システム A	システム B	システム C
AVS/Express	—	○	○
ENVI/IDL	—	○	○
Tecplot	—	○	○

表 26 数式処理アプリケーション

名称	システム A	システム B	システム C
Maple	—	○	○
Mathematica	—	○	○

表 27 技術処理アプリケーション

名称	システム A	システム B	システム C
MATLAB	—	○	○

表 28 計算化学アプリケーション

名称	システム A	システム B	システム C
Gaussian09	—	○	○
Gaussian03	—	○	○
GaussView	—	○	○
MOPAC	—	○	○

表 29 構造解析・機構解析アプリケーション

名称	システム A	システム B	システム C
MSC Nastran	—	○	○
Adams	—	○	○
Marc	—	○	○
Marc Mentat	—	○	○
Patran	—	○	○
LS-DYNA	—	○	○
ANSYS	—	○	○

表 30 統計解析アプリケーション

名称	システム A	システム B	システム C
SAS	—	○	—

表 31 デバッグ用アプリケーション

名称	システム A	システム B	システム C
TotalView	○	○	○

10 おわりに

本稿では、システム A、B、C の各スパコンシステムの利用に必要となる情報を解説しました。より詳細な情報、最新の情報は以下 Web ページをご確認ください。

● スーパーコンピュータの使い方
http://web.kudpc.kyoto-u.ac.jp/manual/

新スーパーコンピュータにおけるベンチマークテスト報告

平石 拓*

*学術情報メディアセンター

1 はじめに

京都大学学術情報メディアセンターでは，2012 年 5 月より新しいスーパーコンピュータのサービスを開始した．

本稿では，本スーパーコンピュータシステム（以下，本システム）におけるいくつかのマイクロベンチマークおよび実アプリケーションに基づくベンチマークプログラムによる性能評価の報告を行うとともに，2008 年 5 月–2012 年 2 月の間サービスを行っていたスーパーコンピュータシステム（以下，旧システム）の Thin システム（T2K オープンスーパーコンピュータ）における同ベンチマークの評価結果も示し，新旧システム間および新システムのサブシステム間の性能の違いを明らかにする．

2 システム構成

まず，両システムのカタログスペックを表 1（計算ノード）および表 2（ストレージ）に示す．新システムは，高い電力性能比を目指したサブシステム A，旧システムの Thin クラスタの高い互換性を維持しつつ性能を向上させたサブシステム B，旧システムの Fat クラスタの後継として 1 ノードあたりに大容量のメモリを搭載したサブシステム C，およびストレージサブシステムから構成される．

システム自体の詳細は，新システムについては [3]，旧システム（ここで掲載していない Fat システムを含む）については [2] もそれぞれ参照いただきたい．

3 ベンチマークプログラム

本稿で示す評価に用いたベンチマークプログラムは以下の通りである．

(a) HPLinpack: Innovative Computing Laboratory, University of Tennessee (ICL-UT) が配布する HPCC ベンチマーク [1] に含まれる，密行列の連立一次方程式を解くベンチマークプログラムであり，Top500 の評価にも用いられる．本評価では，複数ノード（各サブシステムの一部）を使用したときの演算性能を測定した．

(b) 行列積演算プログラム（DGEMM）: 上記 HPCC ベンチマークに含まれる．単一ノードによる性能測定を行った．

(c) 高速フーリエ変換プログラム（FFTE）: 上記 HPCC ベンチマークに含まれる．単一ノードおよび複数ノードによる性能測定を行った．

(d) 並列化 ICCG 法による 2 次元ポアソン方程式の差分解析プログラム（piccg）: Fortran90 および OpenMP で実装されている．単一ノードによる性能測定を行った．

(e) プラズマシミュレーションプログラム（kempo）: C 言語，Fortran90 により記述されている．Flat MPI 版と MPI と OpenMP によるハイブリッド並列版の実装がある．単一ノードの性能測定をハイブリッド版で行い，複数ノードの性能測定を双方のプログラムの測定結果のうち良い方を採用（MPI プロセスあたりの OpenMP スレッド数も最大性能になるように任意に設定）することで行った．

表 2: スーパーコンピュータの性能諸元（ストレージ）

	新システム (Data Direct SFA10000)	旧システム (Fujitsu ETERNUS2000 model 200)
容量	5PB	883TB
転送速度	54GB/s	16GB/s

表 1: スーパーコンピュータの性能諸元（計算ノード）

		新システム			旧システム
		サブシステム A (Cray XE6)	サブシステム B (Appro GreenBlade 8000)	サブシステム C (Appro 2548X)	Thin システム (Fujitsu HX600)
ノード数		940	601	16	416
性能	理論演算性能	300.8TFlops	242.5TFlops (含 GPU) 193.0TFlops (除 GPU)	10.6TFlops	61.2TFlops
	Linpack 性能	239.4TFlops	135.4TFlops	—	50.5TFlops
CPU	種類	AMD Opteron 6200 Series (Interlagos) 2.5GHz	Intel Xeon E5-2670 (Sandy Bridge) 2.6GHz	Intel Xeon E5-2670 (Sandy Bridge) 2.6GHz	AMD Opteron 8350 (Barcelona) 2.3GHz
	ソケット数 /node	2	2	4	4
	コア数/node (/socket)	32 (16)	16 (8)	32 (8)	16 (4)
	キャッシュ	L1D: 64KB/core, L2: 512KB/core, L3: 12MB/socket	L1D: 32KB/core, L2: 256KB/core, L3: 20MB/socket	L1D: 32KB/core, L2: 256KB/core, L3: 20MB/socket	L1D: 64KB/core, L2: 512KB/core, L3: 2MB/socket
GPU	種類	—	NVIDIA Tesla M2090 (64 ノードのみに搭載)	—	—
メモリ	規格	DDR3-1600	DDR3-1600	DDR3-1066	DDR2-667
	総バンド幅	102.4GB/s	102.4GB/s	136.4GB/s	42.7GB/s
	容量	64GB	64 GB	1.5TB	32GB
ネットワーク	ノード間接続	Cray Gemini (9.3GB/s or 4.6GB/s per link), 3D-torus	Infiniband FDR×2 (6.8GB/s×2), Fat tree	Infiniband FDR×2 (6.8GB/s×2), Fat tree	Infiniband DDR×4 (2GB/s×4), Fat tree
	バイセクションバンド幅	1.7TB/s	3.1TB/s	217GB/s	3.33TB/s

(f) 格子ボルツマン法流体シミュレーションプログラム（CFD）:Fortran90 で実装されている．OpenMP 版と Flat MPI 版がある．OpenMP 版による単一ノードの性能測定と，Flat MPI 版による複数ノードの性能測定を行った．

(g) 高速多重極展開法によるマイクロマグネティックシミュレーション（FMM）: C 言語で実装され，MPI による並列化が行われている．複数ノードによる性能測定を行った．

(h) 計算流体力学プログラム（LBM）: Fortran90 で実装されており，OpenMP と MPI によるハイブリッド並列化が施されている．複数ノードによる性能測定を行った．

(i) MPI ベンチマーク（mpibench）: Ping Pong テスト，MPI_sendrecv，MPI_allreduce，MPI_reduce，MPI_allgather，MPI_bcast の実行により MPI の基本性能を測定するベンチマークプログラムである．複数ノードによる性能測定を行った．

(j) SPEC ベンチマーク： The Standard Performance Evaluation Corporation（SPEC）が作成・配布する，CPU 性能を測定するベンチマークである．SPEC CPU2006（CINT2006 および CFP2006）と SPEC OMP2001 による単一ノード性能の評価を行った．

(k) メモリアクセス性能評価ベンチマーク（membench）:（合計性能が最大になるような）任意の数のスレッドを生成し，それぞれのスレッドが巨大なサイズの配列に対するロード・ストアを行うことでメモリの性能評価を行うベンチマークである．単一ノードによる性能評価を行った．

(l) ファイルアクセス性能評価ベンチマーク（thput-fsys，thput-mpi）: 80,000MiB のサイズのファイルのシーケンシャルリード／ライトを行うことにより計算ノードのノードディスク（サブシステム B および C）あるいはストレージシステム（サブシステム A および Thin システム）へのアクセス性能を測定する．thput-fsys では，1 プロセス 1 スレッドによるアクセス性能を測定する．thput-mpi では，（合計性能が最大になるように）任意の数の MPI プロセスを起動し，それぞれのプロセスがシーケンシャルアクセスを行う．

4 測定結果

前節で列挙した各ベンチマークの性能評価結果を示す．なお，コンパイラはサブシステム A においては Cray Compiler Version 8.0，サブシステム B・C においては Intel Compiler Version 12，Thin システムにおいては富士通コンパイラを用いている．

(a) HPLinpack　測定結果を表 3 に示す．表中の「全体推定」とは，このベンチマークをサブシステムシステム全体で走らせたときの性能の参考値 R であり，

$$R = P(N/n)^{\alpha}$$

により求めている．ただし，P は測定性能値，N はサブシステムの総計算ノード数（サブシステム C においては総コア数），n は測定に用いた計算ノード数（サブシステム C においてはコア数），α はノード数を増加させたときの性能向上の度合いを示す 0 以上 1 以下の係数であり，各ベンチマークの性質に応じて適当な値を設定している．完全に線形に性能向上すると期待されるアプリケーションでは $\alpha = 1$，全く性能向上しないと考えられるアプリケーションでは $\alpha = 0$ となる．この評価指標は以降のベンチマークでも用いる．

(b) DGEMM　測定結果を表 4 に示す．なお，サブシステム C の評価ではノード 32 コア中 16 コアのみを使用している．

(c) FFTE　測定結果を表 5 に示す．サブシステム C の単一ノード評価ではノード 32 コア中 16 コアのみを使用している．

(d) piccg　測定結果を表 6 に示す．サブシステム C の単一ノード評価ではノード 32 コア中 16 コアのみを使用している．

(e) kempo　測定結果を表 7 に示す．サブシステム C の単一ノード評価ではノード 32 コア中 16 コアのみを使用している．また，複数ノード評価においては，サブシステム A と Thin システムでは MPI プロセスあたり 4 OpenMP スレッドのハイブリッド並列版，サブシステム B および C では Flat MPI 版の結果を採用した．

(f) CFD　測定結果を表 8 に示す．サブシステム C の単一ノード評価ではノード 32 コア中 16 コアのみを使用している．また，サブシステム A の単一ノード評価には FFTW ライブラリの置き換えやメモリアクセス改善等の改善が施されたプログラムを使用している．なお，単一ノードと複数ノードの評価では問題サイズが異なる．評価値は実行時間なので，小さいほうが好ましい．そのた

表 3: HPLinpack の性能測定結果

システム	A	B	C	Thin
使用ノード数	128	48	2	4
測定値（TFLOPS）	32.2	13.4	1.05	0.45
全体推定（$\alpha = 0.95$）	214	148	7.57	37.1

表 4: DGEMM の性能測定結果

システム	A	B	C	Thin
測定値（GFLOPS）	277	295	279	120

表 5: FFTE の性能測定結果

システム	A	B	C	Thin
測定値（GFLOPS）	19.3	20.9	20.4	2.8

(a) 単一ノード性能

システム	A	B	C	Thin
使用ノード数	128	48	2	4
測定値（GFLOPS）	938	437	39.7	10.1
全体推定（$\alpha = 0.80$）	4623	3301	210	415

(b) 複数ノード性能

表 6: piccg の性能測定結果

システム	A	B	C	Thin
測定値（Kgrid/s）	254	302	313	92.9

表 7: kempo の性能測定結果

システム	A	B	C	Thin
測定値（Mpart/s）	34.6	59.5	52.1	15.7

(a) 単一ノード性能

システム	A	B	C	Thin
使用ノード数	48	32	2	16
測定値（Mpart/s）	3071	2561	418	291
全体推定（$\alpha = 0.90$）	44666	35872	2716	5462

(b) 複数ノード性能

表 8: CFD の性能測定結果

システム	A	B	C	Thin
測定値（実行時間 s）	(15.0)	170.8	191.5	339

(a) 単一ノード性能

システム	A	B	C	Thin
使用ノード数	32	32	16	16
測定値（実行時間 s）	57.9	41.7	48.1	273.8
全体推定（$\alpha = 0.90$）	2.76	2.98	48.1	14.6

(b) 複数ノード性能

表 9: FMM の性能測定結果

システム	A	B	C	Thin
使用ノード数	47	32	2	16
測定値（実行時間 s）	10.7	16.8	69.4	55.4
全体推定（$\alpha = 0.90$）	0.721	1.20	10.7	2.95

表 10: LBM の性能測定結果

システム	A	B	C	Thin
測定値（実行時間 s）	27.0	36.6	39.2	60.0

(a) 単一ノード性能

システム	A	B	C	Thin
使用ノード数	64	32	2	16
測定値（実行時間 s）	9.52	12.1	79.8	57.4
全体推定（$\alpha = 0.90$）	0.848	0.864	12.3	3.06

(b) 複数ノード性能

表 11: mpibench の性能測定結果

システム	A	B	C	Thin
MPI プロセス数（ノードあたり）	32 (1)	32 (1)	64 (32)	4 (1)
Ping Pong 4MB（MB/s）	5930	6310	12414	4390
Ping Pong 8B（MB/s）	5.0	7.82	7.56	1.91
sendrecv 4MB（MB/s）	10175	12028	23046	5460
sendrecv 8B（MB/s）	7.9	13.8	13.6	3.13
allreduce 4MB（MB/s）	1005	1544	482	250
reduce 4MB（MB/s）	892	3717	1239	390
allgather 4MB（MB/s）	169	166	27.65	460
bcast 4MB（MB/s）	1084	4718	1526	2510

表 12: SPEC ベンチマークの性能測定結果

システム	A	B	C	Thin
SPEC CINT2006 Rate	—	583	1020	122
SPEC CFP2006 Rate	—	433	813	105
SPEC OMP2001 Mpeak	—	71093	127534	26080

表 13: membench の性能測定結果

システム	A	B	C	Thin
性能値（GB/s）	40	54	91	20

表 14: thput-fsys，thput-mpi の性能測定結果

システム	A	B	C	Thin
thput-fsys read（GB/s）	1.99	1.44	6.50	0.883
thput-fsys write（GB/s）	0.539	1.11	0.967	0.313
thput-mpi read（GB/s）	9.17	10.2	3.58	1.01
thput-mpi write（GB/s）	8.63	9.48	2.81	1.21

め，「全体推定」の値 T_R は $1/T_R = 1/T(N/n)^{\alpha}$（$T$ は実測定における実行時間）により求めている（以下の FMM と LBM も同様）．

(g) FMM　測定結果を表 9 に示す．

(h) LBM　測定結果を表 10 に示す．サブシステム C の単一ノード評価ではノード 32 コア中 16 コアのみを使用している．単一ノードと複数ノードの評価では問題サイズが異なる．

(i) mpibench　測定結果を表 11 に示す．Ping Pong テストおよび MPI_sendrecv においては，バンド幅だけでなく遅延も評価するため，小さいバイト数での通信性能も測定した．新システムのほうが Thin クラスタより一部の集団通信の性能値が小さくなっているが，MPI プロセス数が多い測定環境のほうが性能値上は不利であることに注意されたい．reduce や bcast においてサブシステム A よりサブシステム B のほうが性能値が小さくなっているのは，ネットワークトポロジーの違いが影響していると考えられる．

(j) SPEC ベンチマーク　測定結果を表 12 に示す．サブシステム A においては SPEC ベンチマークは未測定である．

(k) membench　測定結果を表 13 に示す．なお，このベンチマークではサブシステム A のコンパイラとして Intel Compiler Version 12 を用いている．

(l) thput-fsys，thput-mpi　測定結果を表 14 に示す．

5　おわりに

本稿では，メディアセンターの新しいスパコンと旧スパコンにおけるいくつかのベンチマークの測定結果を示し，システム間の実性能を比較した．HPL や SPEC 等ではカタログスペックから期待される性能が得られており，実アプリケーションベースのベンチマークでも，たとえばサブシステム A と Thin システムのそれぞれ全体の比較で，潜在的に 2–6 倍の性能向上が期待できることがわかる．

サブシステム A と B の比較では，単一ノード間では大きな性能差は見られない．ただし，ノードあたりの利用負担金はシステム A のほうが安く設定されているため，ノード間並列を利用することにより，同一コストで高い性能を得られることが期待できる．ただし，そのような開発にあたっては両者のネットワークトポロジーの違いにも注意する必要がある．

本稿がシステムの利用検討およびアプリケーション開発の一助になれば幸いである．

謝辞

ご多忙の中，旧システム等に関する資料調査・提供をしていただいた京都大学情報部情報基盤課研究支援グループの皆様に感謝いたします．

参考文献

[1] Innovative Computing Laboratory, University of Tennessee: HPC Challenge. http://icl.cs.utk.edu/hpcc/.

[2] 斎藤紀恵, 疋田淳一, 平野彰雄: スーパーコンピュータ利用ガイド—Thin/Fat クラスタを利用するために—, 京都大学学術情報メディアセンター 全国共同利用版広報, Vol. 7, No. 2 (2008).

[3] 中島浩: 新スーパーコンピュータシステム—そのコンセプトと構成—, 京都大学学術情報メディアセンター全国共同利用版広報, Vol. 11, No. 1 (2012).

Thin SMP クラスタ運転状況 （2011/10 ～ 2012/3）

1）保守作業に伴うサービス休止およびシステムダウン障害発生状況

保守作業に伴うサービス休止

保守開始日時	サービス再開日時	保守時間[h]
2011/10/10 6:30	2011/10/11 16:20	33.8
2011/12/19 9:00	2011/12/19 18:00	9.0

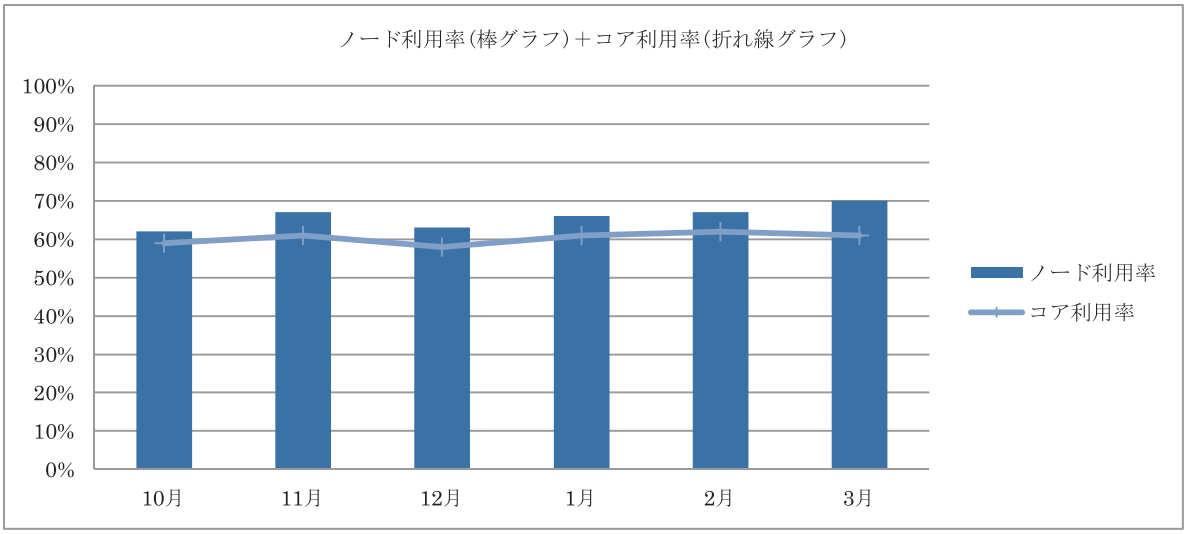
※　システム移行に伴い 2012/3/15 17:00 にサービスを終了

システムダウン障害発生状況

障害発生日時	サービス再開日時	ダウン時間[h]
該当なし		

2）サービス状況

	サービス時間[h]	バッチ						TSS			
		処理件数	経過時間[h]	占有時間[h]	CPU時間[h]	平均稼動ノード数	ノード利用率	セッション数	セッション時間[h]	CPU時間[h]	平均稼動ノード数
10 月	710	262,550	232,668	2,613,408	2,539,522	381.5	62%	14,088	5,657	158,352	25.7
11 月	720	143,227	208,665	2,628,248	2,569,905	390.0	67%	15,493	6,749	105,929	26.0
12 月	744	220,089	221,443	2,470,989	2,402,044	387.1	63%	13,701	5,084	66,608	25.9
1 月	744	112,458	348,639	2,829,293	2,567,286	390.0	66%	16,694	7,008	121,069	26.0
2 月	672	119,088	219,696	2,560,561	2,413,093	390.0	67%	13,570	15,311	111,119	26.0
3 月	353	32,709	144,877	1,544,749	1,404,710	368.1	70%	5,761	9,694	46,074	25.7
計	3,943	890,121	1,375,988	14,647,247	13,896,561	384.5	66%	79,307	49,503	609,152	25.9



※ 占有時間 = 合計(経過時間×占有コア数)

※ 平均稼動ノード数 = 電源 ON 状態のノード数の月平均 (10 分間隔のサンプリングデータより算出)

※ ノード利用率 = 稼動ノードに対するジョブが実行されているノードの割合

※ TSS = ログインノード＋専用クラスタについてのデータ

Fat SMP クラスタ運転状況　(2011/10 ～ 2012/3)

1）保守作業に伴うサービス休止およびシステムダウン障害発生状況

保守作業に伴うサービス休止

保守開始日時	サービス再開日時	保守時間[h]
2011/10/10 6:30	2011/10/11 16:20	33.8
2011/12/19 9:00	2011/12/19 18:00	9.0

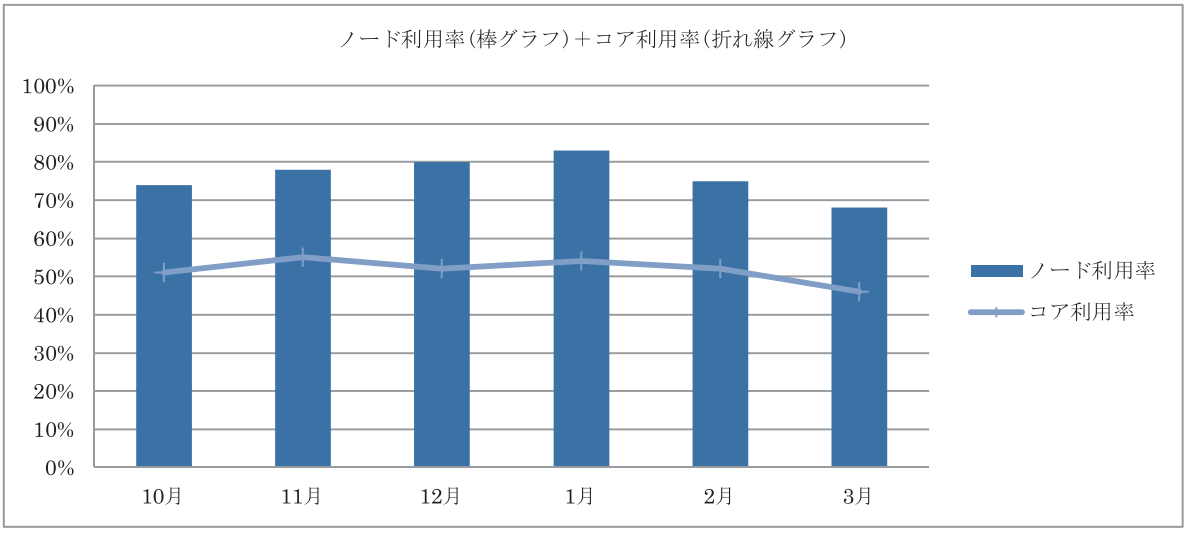
※　システム移行に伴い 2012/3/15 17:00 にサービスを終了

システムダウン障害発生状況

障害発生日時	サービス再開日時	ダウン時間[h]
2011/12/15 12:50	2011/12/15 18:10	5.7

2）サービス状況

	サービス時間[h]	バッチ						TSS			
		処理件数	経過時間[h]	占有時間[h]	CPU時間[h]	平均稼動ノード数	ノード利用率	セッション数	セッション時間[h]	CPU時間[h]	平均稼動ノード数
10 月	710	921	15,943	278,580	231,686	6.8	72%	2,196	937	1,765	1.0
11 月	720	1,700	16,711	314,482	279,662	7.0	75%	2,236	481	5,520	1.0
12 月	738	1,155	16,244	309,253	292,080	6.9	78%	2,214	232	10,823	1.0
1 月	744	1,137	16,644	297,035	296,020	7.0	81%	2,408	282	1,944	1.0
2 月	672	854	8,456	295,904	261,473	7.0	71%	2,089	581	4,161	1.0
3 月	353	280	4,313	127,997	119,768	6.6	67%	983	271	349	1.0
計	3,938	6,047	78,310	1,623,250	1,480,688	6.9	74%	12,126	2,784	24,562	1.0



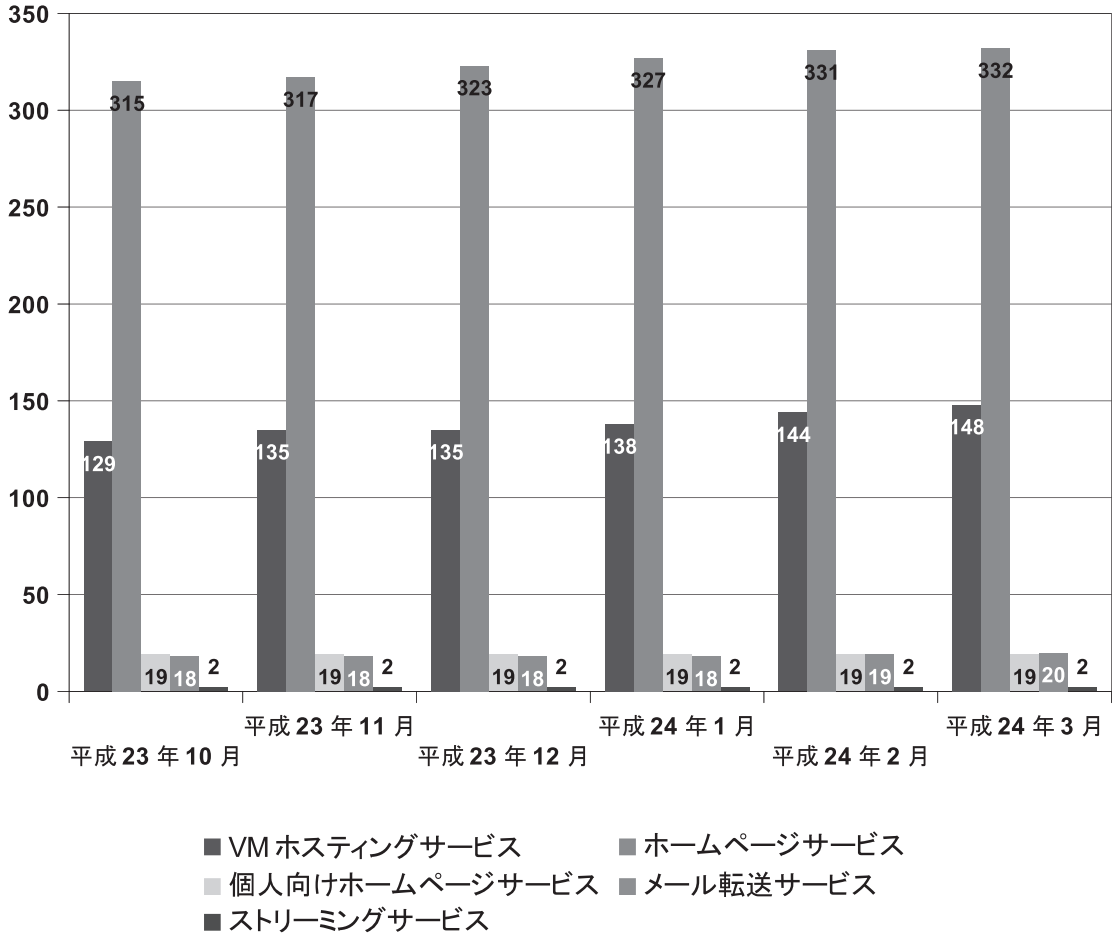
※ 占有時間 = 合計(経過時間×占有コア数)

※ 平均稼動ノード数 = 電源 ON 状態のノード数の月平均 (10 分間隔のサンプリングデータより算出)

※ ノード利用率 = 稼動ノードに対するジョブが実行されているノードの割合

汎用コンピュータシステムのサービス状況

1　ホスティング・ホームページサービス利用状況



（平成 23 年 10 月から平成 24 年 3 月）

大型計算機システム利用承認件数について

平成 24 年 3 月末現在、大型計算機システムの利用件数は、2,355 件となっています。

大型計算機システム利用負担金

別表1　スーパーコンピュータシステム

コース	タイプ	セット	利用負担額	提供サービス					
				システム	パッチ	システム資源	経過時間（時間）	ディスク（GB）	利用者番号
エントリ	−	基本	12,600 円/年	B	共有	最大1ノード相当((16コア、64GBメモリ)×1)	1	60	−
パーソナル	タイプA	基本	100,000 円/年	A	共有	最大4ノード相当((32コア、64GBメモリ)×4)	168	1,000	−
	タイプB	基本	100,000 円/年	B	共有	最大4ノード相当((16コア、64GBメモリ)×4)	168	1,000	−
	タイプC	基本	100,000 円/年	C	共有	最大2ソケット相当((8コア、384GBメモリ)×2	168	1,000	−
グループ	タイプA1	最小	200,000 円/年	A	優先	4ノード((32コア、64GBメモリ)×4)	336	8,000	8
		追加単位	200,000 円/年			4ノード((32コア、64GBメモリ)×4)	−	8,000	8
	タイプA2	最小	240,000 円/年		準優先	8ノード((32コア、64GBメモリ)×8)	336	9,600	16
		追加単位	120,000 円/年			4ノード((32コア、64GBメモリ)×4)	−	4,800	8
	タイプA3	最小	600,000 円/年		占有	8ノード((32コア、64GBメモリ)×8)	336	16,000	16
		追加単位	300,000 円/年			4ノード((32コア、64GBメモリ)×4)	−	8,000	8
	タイプB1	最小	250,000 円/年	B	優先	4ノード((16コア、64GBメモリ)×4)	336	8,000	8
		追加単位	250,000 円/年			4ノード((16コア、64GBメモリ)×4)	−	8,000	8
	タイプB2	最小	300,000 円/年		準優先	8ノード((16コア、64GBメモリ)×8)	336	9,600	16
		追加単位	150,000 円/年			4ノード((16コア、64GBメモリ)×4)	−	4,800	8
	タイプB3	最小	750,000 円/年		占有	8ノード((16コア、64GBメモリ)×8)	336	16,000	16
		追加単位	375,000 円/年			4ノード((16コア、64GBメモリ)×4)	−	8,000	8
	タイプC1	最小	400,000 円/年	C	優先	4ソケット((8コア、384GBメモリ)×4)	336	8,000	16
		追加単位	200,000 円/年			2ソケット((8コア、384GBメモリ)×2)	−	4,000	8
	タイプC2	最小	240,000 円/年		準優先	4ソケット((8コア、384GBメモリ)×4)	336	4,800	16
		追加単位	120,000 円/年			2ソケット((8コア、384GBメモリ)×2)	−	2,400	8
	タイプG1	最小	250,000 円/年	B (GPU)	優先	2ノード((16コア、64GBメモリ + 1GPU)×2)	336	4,000	8
		追加単位	250,000 円/年			2ノード((16コア、64GBメモリ + 1GPU)×2)	−	4,000	8
大規模ジョブ	タイプA	最小	20,000 円/週(7日)	A	占有	8ノード((32コア、64GBメモリ)×8)	−	−	−
		追加単位	5,000 円/週(7日)			2ノード((32コア、64GBメモリ)×2)	−	−	−
	タイプB	最小	24,000 円/週(7日)	B	占有	8ノード((16コア、64GBメモリ)×8)	−	−	−
		追加単位	6,000 円/週(7日)			2ノード((16コア、64GBメモリ)×2)	−	−	−
	タイプC	最小	20,000 円/週(7日)	C	占有	4ソケット((8コア、384GBメモリ)×4)	−	−	−
		追加単位	10,000 円/週(7日)			2ソケット((8コア、384GBメモリ)×2)	−	−	−
専用クラスタ	—	最小	750,000 円/年	B	−	8ノード((16コア、64GBメモリ)×8)	−	16,000	16
		追加単位	375,000 円/年			4ノード((16コア、64GBメモリ)×4)	−	8,000	8
ライセンスサービス			20,000 円/年	可視化ソフト(AVS,ENVI/IDL)およびプリポストウェアの1ライセンスにつき					

備考

1. 利用負担額は、年度単位で算定している。また、総額表示である。
2. 大型計算機システムの全ての利用者は、上記表のサービスの他、次のサービスを受けることができる。
   1） 大判プリンタサービス
   2） その他、大型計算機システムが提供するサービス、機器の利用
3. 上記表の大規模ジョブコース、ライセンスサービスの申請には、大型計算機システムの利用者であることが必要である。
4. 「共有」: 当該カテゴリのユーザ間で一定の計算資源を共有するベストエフォートのスケジューリングを行う。
   「準優先」: 定常稼働状況において記載値（以上）の計算資源が確保されるように優先スケジューリングを行う。
   また、稼働状況によらず記載値の1/4の計算資源が確保されることを保証する。
   「優先」: 定常稼働状況において記載値（以上）の計算資源が確保されるように優先スケジューリングを行う。
   また、稼働状況によらず記載値の1/2の計算資源が確保されることを保証する。
   「占有」: 稼働状況によらず記載値（以上）の計算資源が確保されることを保証する。
5. ディスク容量はバックアップ領域（最大で総容量の1/2）を含む。
6. グループコース及び専用クラスタコースのシステム資源は、下記の負担額を支払うことにより増量することができる。
   なお増量は各月1日に実施し、増量した資源は当該年度末までの期間にわたって利用されるものとする。

コース	タイプ	追加負担金額（増量単位あたり）	システム資源増量単位	ディスク増量（GB）
グループ	タイプA1	20,000 円/月	4ノード((32コア、64GBメモリ)×4)	8,000
	タイプA2	12,000 円/月	4ノード((32コア、64GBメモリ)×4)	4,800
	タイプA3	30,000 円/月	4ノード((32コア、64GBメモリ)×4)	8,000
	タイプB1	25,000 円/月	4ノード((16コア、64GBメモリ)×4)	8,000
	タイプB2	15,000 円/月	4ノード((16コア、64GBメモリ)×4)	4,800
	タイプB3	37,500 円/月	4ノード((16コア、64GBメモリ)×4)	8,000
	タイプC1	20,000 円/月	2ソケット((8コア、384GBメモリ)×2)	4,000
	タイプC2	12,000 円/月	2ソケット((8コア、384GBメモリ)×2)	2,400
	タイプG1	25,000 円/月	2ノード((16コア、64GBメモリ + 1GPU)×2)	4,000
専用クラスタ	—	37,500 円/月	4ノード((16コア、64GBメモリ)×4)	8,000

7. グループコース及び専用クラスタコースを通年でなく利用する場合には、下記の負担額を支払うものとする。
   ただし、利用期間は当該年度内に限るものとする。

利用期間			3ヶ月	6ヶ月	9ヶ月
グループコース	タイプA1	最小	80,000 円	120,000 円	180,000 円
		追加単位	80,000 円	120,000 円	180,000 円
	タイプA2	最小	96,000 円	144,000 円	216,000 円
		追加単位	48,000 円	72,000 円	108,000 円
	タイプA3	最小	240,000 円	360,000 円	540,000 円
		追加単位	120,000 円	180,000 円	270,000 円
	タイプB1	最小	100,000 円	150,000 円	225,000 円
		追加単位	100,000 円	150,000 円	225,000 円
	タイプB2	最小	120,000 円	180,000 円	270,000 円
		追加単位	60,000 円	90,000 円	135,000 円
	タイプB3	最小	300,000 円	450,000 円	675,000 円
		追加単位	150,000 円	225,000 円	337,500 円
	タイプC1	最小	160,000 円	240,000 円	360,000 円
		追加単位	80,000 円	120,000 円	180,000 円
	タイプC2	最小	96,000 円	144,000 円	216,000 円
		追加単位	48,000 円	72,000 円	108,000 円
	タイプG1	最小	100,000 円	150,000 円	225,000 円
		追加単位	100,000 円	150,000 円	225,000 円
専用クラスタ コース	—	最小	300,000 円	450,000 円	675,000 円
		追加単位	150,000 円	225,000 円	337,500 円

8. グループコース及び専用クラスタコースの利用者番号は利用者あたり年額5,000円を負担することで追加できる。

9. 機関・部局定額制度
他機関又は学内における部局(『国立大学法人京都大学の組織に関する規程』第3章第2節から第11節で定める組織をいう。)の組織が、その組織単位でグループコースサービス(年間)の利用を申請する場合、料金表(年間)に掲載額の1.5倍を利用負担金とする。なお、利用負担金額が150万円未満の場合は100人、150万円を超える場合は、150万円毎に100人までの利用者を認める。

別表2(汎用コンピュータシステム)

区分	利用負担額	単位
VMホスティングサービス	126,000円／年	1仮想マシンにつき
ホームページサービス	31,500円／年	1ドメイン名につき
個人向けホームページサービス	12,600円／年	1アカウントにつき
メール転送サービス	12,600円／年	1ドメイン名につき

備考

1. 利用負担額は、総額表示である。

2. 上記表の汎用コンピュータシステムのサービスを利用するためには、大型計算機システムの利用者であることが必要である。

3. ホームページサービス及びVMホスティングサービスにおいて、下記の負担額を支払うことによりオプションサービスを利用することができる。

オプションサービス種別	利用負担額	単位
データベース(Oracle)	63,000円／年	1アカウントにつき
ストリーミング(Helix Server)	31,500円／年	1アカウントにつき

4. VMホスティングサービスのシステム資源は、下記の負担額を支払うことにより増量することができる。

種別	利用負担額	単位
ディスク	10,500円／年	100GBにつき
システム資源	100,800円／年	1台につき

システム資源1台とは、CPU:2コア,メモリ:2GB である。

5. VMwareを用いたVMホスティングサービスは、下記の負担額を支払うことにより利用・増量することができる。ただし、システム資源が非常に限られているためサービスを提供できる場合が限定される。

種別	利用負担額	単位
標準機能サポート	25,200円／年	1仮想マシンにつき
ディスク	10,500円／年	100GBにつき
システム資源	201,600円／年	1台につき

システム資源1台とは、CPU:1コア,メモリ:2GB である。

6. 利用負担額は、当該年度（4月から翌年3月まで）の利用に対して年額として算定するが、年度途中から利用を開始する場合には月数に応じて減額する。

別表3 スーパーコンピュータシステム(民間機関利用)

システム	システム資源	経過時間(時間)	ディスク(GB)	利用者番号	利用負担額
A	8ノード(32コア、64GBメモリ)×8)	336	9,600	16	960,000 円/年
	12ノード(32コア、64GBメモリ)×12)	336	14,400	24	1,440,000 円/年
	16ノード(32コア、64GBメモリ)×16)	336	19,200	32	1,920,000 円/年
B	8ノード(16コア、64GBメモリ)×8)	336	9,600	16	1,200,000 円/年
	12ノード(16コア、64GBメモリ)×12)	336	14,400	24	1,800,000 円/年
	16ノード(16コア、64GBメモリ)×16)	336	19,200	32	2,400,000 円/年

備考

1. 利用負担額は、年度単位で算定している。また、総額表示である。
2. ディスク容量はバックアップ領域(最大で総容量の1/2)を含む。
3. 通年でなく利用する場合には、下記の負担額を支払うものとする。
ただし、利用期間は当該年度内に限るものとする。

システム	システム資源	利用期間		
		3ヶ月	6ヶ月	9ヶ月
A	8ノード	240,000 円	480,000 円	720,000 円
	12ノード	360,000 円	720,000 円	1,080,000 円
	16ノード	480,000 円	960,000 円	1,440,000 円
B	8ノード	300,000 円	600,000 円	900,000 円
	12ノード	450,000 円	900,000 円	1,350,000 円
	16ノード	600,000 円	1,200,000 円	1,800,000 円

全国共同利用版広報・Vol.10(2011)総目次

[スーパーコンピュータ共同研究制度（大規模計算支援枠）研究報告]

[2010 年度京都大学学術情報メディアセンターコンテンツ作成共同研究　研究報告]

[サービスの記録・報告]

[資料]

[編集後記]

― サービス利用のための資料一覧 ―

1. スーパーコンピュータシステム・ホスト一覧

- システム A：camphor.kudpc.kyoto-u.ac.jp
- システム B・C：laurel. kudpc.kyoto-u.ac.jp
  - システム B（SAS 利用時）：sas.kudpc.kyoto-u.ac.jp

※ ホストへの接続は SSH(Secure SHell) 鍵認証のみ、パスワード認証は不可

2. 問い合わせ先 & リンク集

- 情報環境機構のホームページ
  http://www.iimc.kyoto-u.ac.jp/

- 学術情報メディアセンターのホームページ
  http://www.media.kyoto-u.ac.jp/

- スーパーコンピュータシステムに関する問い合わせ先
  - 利用申請などに関する問合わせ先

  【共同利用支援グループ・共同利用第一掛（北館窓口）】
  E-mail：zenkoku-kyo@media.kyoto-u.ac.jp / Tel：075-753-7424 / Fax：075-753-7449
  URL: http://www.iimc.kyoto-u.ac.jp/ja/services/comp/
  - システムの利用など技術的な問い合わせ

  【研究支援グループ】
  E-mail：consult@kudpc.kyoto-u.ac.jp / Tel：075-753-7426
  URL: http://www.iimc.kyoto-u.ac.jp/ja/services/comp/contact.html

- ホームページ・ホスティングサービス関する問合わせ先
  【情報環境支援グループ】
  E-mail：whs-qa@media.kyoto-u.ac.jp / Tel：075-753-7494
  URL: http://www.iimc.kyoto-u.ac.jp/ja/services/whs/

- コンテンツ作成支援サービスに関する問合わせ先
  【コンテンツ作成室】
  E-mail：cpt@media.kyoto-u.ac.jp / Tel：075-753-9012
  URL: http://www.iimc.kyoto-u.ac.jp/ja/services/content/

編 集 後 記

最近、趣味の一つに旅が加わりました。普段行かないような場所へ行き、そこへ行かないと見ることが出来ない風景を見る。そしてその風景を記憶として刻む瞬間が、自分にとって幸せな時間です。それ以外にも、移動中に列車の窓から見える景色、旅先ならでわの食事、時には天候不順や交通手段のトラブル等のアクシデントですら立派な旅の醍醐味になります。これからもいろんな場所へ行ってみたいのですが、旅行には先立つものが必要。そのため頻繁に出かけることができないのが難点です・・・。

旅好き

2年ほど前に一眼レフカメラを手に入れて以来、写真を撮るのが趣味になりました。ターゲットはズバリ飛行機。飛行機の観察や撮影をたしなむ人は「スポッター」と呼ばれますが、駆け出しのスポッターとして腕を磨くべく、機会を見つけて撮影に出かけています。重たいカメラと望遠レンズ、高速で動く被写体、行ってみないと分からない空模様、視界を遮る空港の柵や壁など、いろいろ難しいことはありますが、それだけに納得のゆく写真を撮れたときの満足感はさしずめファーストクラス…乗ったことはないので想像ですが。

エコノミースポッター

京都大学学術情報メディアセンター全国共同利用版広報　Vol. 11, No. 1

2012年　8月　27日発行

編集者　京都大学学術情報メディアセンター
　　広報教育委員会・全国共同利用版広報編集部会
発行者　〒606-8501　京都市左京区吉田本町
　　京都大学学術情報メディアセンター
　　Academic Center for Computing and Media Studies
　　Kyoto University
　　Tel. 075-753-7400
　　http://www.media.kyoto-u.ac.jp/
印刷所　〒616-8102　京都市右京区太秦森ヶ東町21-10
　　株式会社エヌジーピー
　　http://www.nextgp.jp