国際共同実験 ALICE のための LCG Tier 2 センター及び Grid 解析環境の構築

広島大学理学研究科物理科学専攻
クォーク物理学研究室
M060609 成田拓人

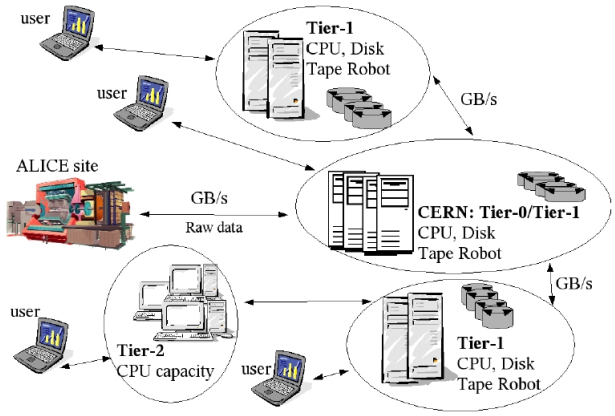
平成 20 年 2 月 8 日

概 要

現在建設中の LHC 加速器においては、史上最大衝突エネルギーとなる核子対当たり衝突エネルギー 5.5TeV の鉛+鉛原子核衝突が実現される。その中で ALICE 実験のような高エネルギー重イオン衝突では、発生する粒子数が多く粒子密度も高いため、収集されるデータサイズは非常に大きい。そのため、ALICE 実験で要求されるストレージは 1 年間で約 25PB、CPU は約 35MSI2k と、多くの資源が必要であると見積もられている。

一方、近年開発が進んでいる技術として Computing GRID というものがある。ALICE 実験を含む LHC 加速器における実験では、その莫大なデータを処理するために Grid が用いられている。Grid とは、コンピュータ、ネットワーク、ストレージなどの資源を仮想コンピューティング・システム化することで、Grid を利用するユーザーやアプリケーションが、その情報資源を利用できるようにするものである。

広島大学は LCG(LHC Computing Grid) に参加し、その中で ALICE 実験をサポートする。LCG では、主に大学や研究所をひとつの単位としてサイトと呼んでおり、それらの複数のサイトが右図のように階層構造を成すようなモデルが採用されている。本論文では、Tier 2 という階層のサイトを広島大学において構築し、ALICE 実験の国内唯一のサイトとする。実際にはサイトを構築するにあたって、バッチジョブシステムをもちジョブを処理するマシン、実際にジョブが実行されるマシン、ストレージを提供するマシン、サイトのモニタリングを行うマシン、ユーザーが Grid へアクセスするためのマシンをそれぞれ、YAIM を用いて Grid ミドルウェアを導入することによりセットアップした。その他に Grid としてのサービスを実行するために必要なマシンは、CERN 等が提供するマシンを用いた。次に、広島サイトの各マシンで必要とされる各サービスが起動していることの確認や、複数のマシンが関わるサービスに対してはネットワークを介してそれらがやり取りをできるよう設定を行った。そしてサイト全体として、ユーザーの認証を行い、ジョブを受け入れジョブを実行しデータを保存



すること、サイトのモニタリング、Grid の情報へのアクセスなどのサービスを実行することのできることを確認し、サイトとしての環境を整えた。その結果、Tier 2 センターとしての要求を満たし、約 50kSI2k の CPU を ALICE 実験のために提供することができた。そして、このサイトの計算力や、内部および外部とのバンド幅などのパフォーマンスについても議論を行った。また、広島大学では ALICE 実験 PHOS 検出器の開発も行っており、その解析用として ALICE のソフトウェアを使うことのできる研究室内解析環境も構築した。

目 次

表 目 次

図 目 次

第1章　序論

1.1　背景

1.1.1　QGP

QCD[1] が持つ漸近的自由性の性質から、ハドロン相は非常に高温・高密度の状態になると閉じ込めから解放され、クォークとグルーオンが自由に飛び回る新しい物質相に相転移することが予想されている。この新しい物質相をQGP 相という。人工的に QGP[2] 相への相転移を実現することが出来れば、QCD の非摂動領域の検証になり、真空構造の研究や閉じ込め機構の解明に役立つと考えられる。また、ビッグバン直後、数μ秒までのわずか 10^{-5} 秒の間、高温高密度の宇宙は QGP 相と呼ばれる現在とは全く異なった物質状態にあった。

つまり、非摂動的領域の強い相互作用によるクォーク多体系現象を解明することは、QCD による強い相互作用の理解を進めるだけでなく、宇宙誕生の謎に迫り、物質宇宙の起源と時空発展の解明に向けての重要なステップである。

粒子加速器を用いた高エネルギー原子核衝突の 実験的研究 により、このクォーク物質状態を制御可能かつ系統的に実験室中に再創成し、そこで発現する多様な素粒子物理現象を探求することができる。

1.1.2　ALICE

BNL[3] 研究所 RHIC[4] 加速器による高エネルギー原子核衝突実験 PHENIX[5] での、核子対あたりの衝突エネルギー 200GeV の原子核衝突系において新たな現象が次々に発見された。それにより、QCD が予言する QGP 相に相当すると考えられるクォーク物質相の存在ををを示す証拠をつかんだ。ALICE[6] 実験では、それらをより精密にするために、より理想的な条件である衝突エネ

[1]量子色力学 (Quantum Chromo Dynamics)
[2]クォーク・グルーオン・プラズマ (Quark Gluon Plasma)
[3]ブルックヘブン国立研究所 (Brookhaven National Laboratory)
[4]Relativistic Heavy Ion Collider
[5]Pioneering High Energy Nuclear Interaction eXperiment
[6]A Large Ion Collider Experiment

ルギーを増した条件で定量的に測定し、その変化量を把握し解釈することが重要である。

RHIC 加速器による成果を踏まえ、LHC 加速器施設において、ALICE 実験では史上最大衝突エネルギーとなる核子対当たり衝突エネルギー 5.5TeV の鉛+鉛原子核衝突を実現し、クォーク物質の物性解明を目指している。LHC 加速器実験では、RHIC 加速器より 28 倍高い衝突エネルギーを供給することにより、より理想的な高温・高密度のクォーク物質を長時間形成することができるため、クォーク物質相で引き起こされる現象を明確に調べることが可能である。ALICE 実験は LHC 加速器に導入される 4 実験の中で、クォーク物質探求に最適化した唯一の実験装置である。

1.1.3 Grid の概要

近年、広域ネットワーク上に分散した計算資源およびストレージ資源を統合し、仮想計算機環境を構築することを目的とした Grid と呼ばれる技術が注目されている。現在、様々な分野で Grid が使われはじめており、高エネルギー物理の分野でも注目を浴びている。高エネルギー物理では、エネルギーが上がるにつれてそのデータ量も膨大に増加するため、それらを処理し解析を実行するためには Grid の基盤が必要である。また、実験自体の規模も大きくなっているため様々な国から様々な機関が参加しており、その中でデータを扱っていかなければならないということに対しても最適である。実際に LHC 加速器に導入される 4 実験でも使われ、もちろん ALICE 実験でも開発が進んでいる。

広島大学も Grid に参加し、大学のコンピュータ資源の一部を ALICE 実験のデータ解析用に供給を開始した。日本における ALICE 実験のための Grid は、広島大学が唯一である。

1.2 目的

ALICE 実験では史上最大衝突エネルギーとなる核子対当たり衝突エネルギー 5.5TeV の鉛+鉛原子核衝突を実現し、クォーク物質の物性解明を目指している。その衝突エネルギーは RHIC 加速器 PHENIX 実験 (核子対当たりの衝突エネルギー 200GeV) に比べ約 28 倍であり、粒子生成多重度も 5〜10 倍高い。そのため ALICE 実験では多量のデータを扱わなければならず、そのためには各研究機関の持つ計算資源を Grid として再構築し、共有することにより、解析に必要な計算能力を確保することが不可欠である。それゆえ、広島大学においても、計算機資源の一部を Grid として構築し、ALICE 実験におけるデータ解析の一端を担うことは急務である。

本論文では、広島大学における Grid サイトの構築について述べる。また、サイトを構築すると同時にユーザーとして Grid を使うことも目的としている。そのパフォーマンスとしてのバンド幅についても測定を行い、議論する。また、広島大学では ALICE 実験 PHOS 検出器の開発も行っており、その解析用として構築した研究室内解析環境施設についても述べる。

第 1 章では背景および目的、弟 2 章では ALICE 実験の概要、弟 3 章では Grid の概要を述べる。そして弟 4 章では Grid サイトと研究室内解析環境の構築について、またそのテストやパフォーマンスを調べるための方法を述べ、弟 5 章でその結果の記述および考察を行い、最後に弟 6 章で結論および今後の展望を述べる。

第2章　ALICE実験

2.1　CERN

CERN[1] はスイスのジュネーブ近郊にある世界最大規模の素粒子物理学研究所である。加速器を用いた素粒子物理学および原子核物理学の研究のほか、研究に必要な有用な技術の開発を行っている。フェルミ国立加速器研究所と共同で、研究用独自オペレーティングシステムである Scientific Linux の開発もしている。これは Red Hat Enterprise Linux の商標に関する部分を削除し、ソースコードから再コンパイルした無償配布のディストリビューションである。また、HTML や World Wide Web の発祥の地でもある。

2.2　LHC

LHC[2] は高エネルギー物理実験を目的として CERN に建設された世界最大の衝突型円型加速器である。図 2.1 のようにスイス・ジュネーブ郊外にフランスとの国境をまたいで地下約 100m に設置され、周長は 27km である。ALICE、ATLAS、CMS、LHCb の４つの実験が計画されている。

ALICE 実験は 4 実験の中で唯一、鉛+鉛原子核衝突による現象を主題としている。ATLAS 実験及び CMS 実験はどちらも素粒子物理実験に主題をおき、素粒子理論の標準模型でその存在が予言されているヒッグス粒子発見をその第一目標に据えており、同時に SUSY 粒子探索など標準理論を超える新たな発見を目指している。LHCb 実験はボトムクォーク生成に関与する物理現象に焦点を当て、CP 非保存の素粒子物理現象の解明を目的としている。

[1]欧州原子核研究機構
[2]大型ハドロン衝突型加速器 (Large Hadron Collider)

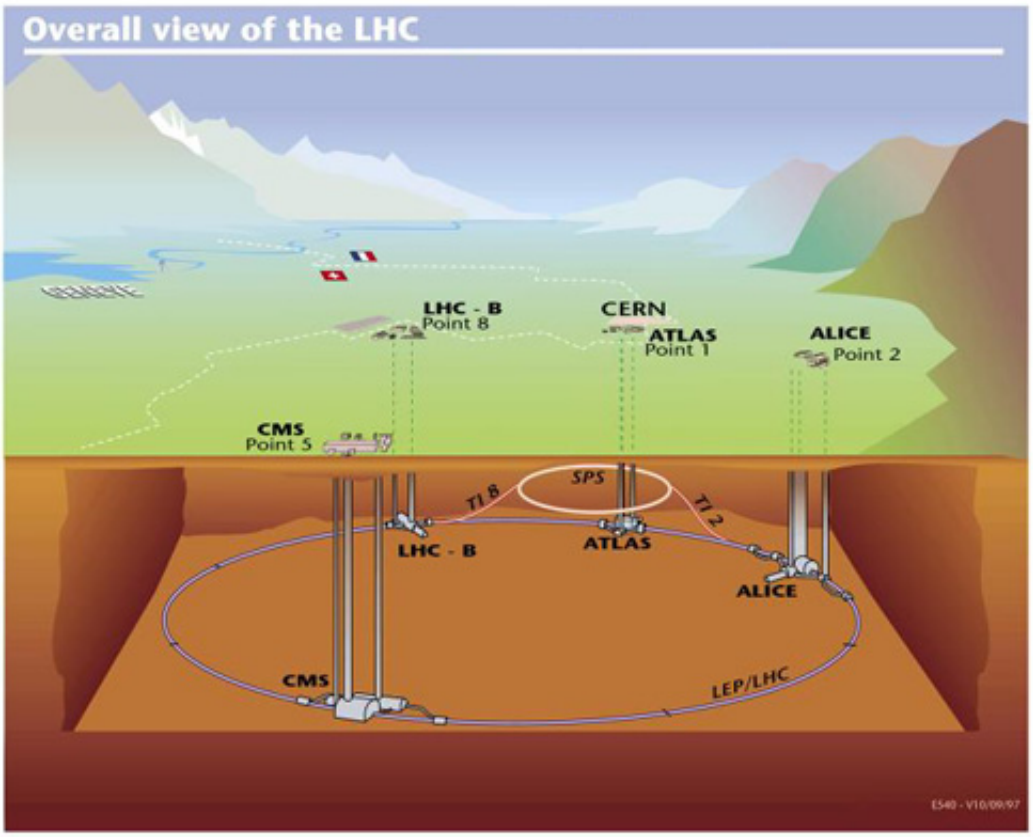


図 2.1: LHC 加速器の概観。LHC 加速器はスイス・ジュネーブ郊外にフランスとの国境をまたいで地下約 100m に設置され、ALICE、ATLAS、CMS、LHCb の 4 つの実験が計画されている。

LHC の立上日程

- 2008 年 4 月加速器、実験閉鎖
- 2008 年 5 月ビーム調整 (p+p, $\sqrt{s_{NN}}$=14TeV)
- 2008 年 7 月衝突開始
- 2008 年末 p+p, $\sqrt{s_{NN}}$=14TeV, $10^{32}cm^{-2}s^{-1}$

また、当初数年間に期待される実験条件について表 2.1 に示す。

表 2.1: LHC 加速器において当初数年間に期待される実験条件

核種	エネルギー ($\sqrt{s_{NN}}$)	ルミノシティ	期間
p+p	14TeV	$10^{31} cm^{-2} s^{-1}$(ALICE)	
Pb+Pb	5.5TeV	$10^{27} cm^{-2} s^{-1}$	2-3 年
p+Pb	8.8TeV	$10^{29} cm^{-2} s^{-1}$	1 年
Ar+Ar	6.3TeV	$10^{29} cm^{-2} s^{-1}$	1 年

2.3 ALICE 実験

ALICE 実験は、LHC での原子核原子核衝突で QGP 相の物理を研究するために設計された重イオン衝突実験である。ALICE コラボレーションには現在、29 カ国から 86 の機関、1000 人以上が参加している。

2.3.1 ALICE 実験の検出器

図 2.2 のように ALICE 実験は巨大なソレノイド電磁石の中心にビーム衝突点をおき、衝突点を囲むように中心飛跡検出器、電子同定検出器、飛行時間差検出器、電磁カロリーメータ、チェレンコフ検出器を配置している。更に、衝突点のビーム軸方向下流に複数の事象識別検出器とミュウ粒子検出器を置いている。ALICE 実験は、これらの検出器により得られる複数の情報を総合してクォーク物質の探索を行う複合型スペクトロメータである。

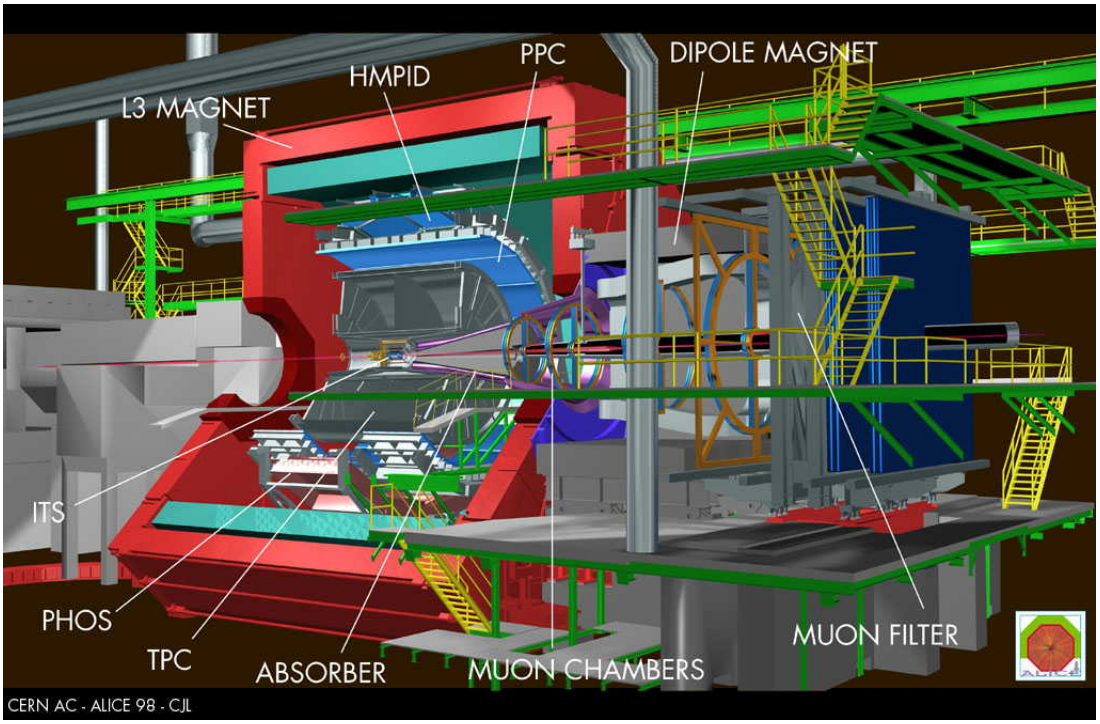


図 2.2: ALICE 検出器の全貌。巨大なソレノイド電磁石の中心にビーム衝突点をおき、衝突点を囲むように中心飛跡検出器、電子同定検出器、飛行時間差検出器、電磁カロリーメータ、チェレンコフ検出器を配置している。更に、衝突点のビーム軸方向下流に複数の事象識別検出器とミュウ粒子検出器を置いている。

衝突点を囲む検出器

ITS　ITS(Inner Tracking System) は飛跡検出器であり、その主な目的はセカンダリーバーテックスと低い運動量の荷電粒子の飛跡の検出である。半径 3.9cm から 45cm まで、6 層の円筒形のピクセル型及びストリップ型シリコン半導体検出器により構成されている。

TPC　TPC(Time Projection Chamber) は ALICE での主な飛跡検出器で、その仕事は飛跡を見つけること、荷電粒子の運動量測定、エネルギー損失率 (dE/dx) 測定による粒子識別である。半径 88cm から 2.50m まで軸長 5m、読み出しチャンネル数 570K で構成されている。

TRD　TRD(Transition Radiation Detector) は飛跡検出を行い、TPC の飛跡情報と併せることにより最も軽い荷電粒子である電子や陽電子を同定することができる。Xenon/CO_2 で満たされたワイヤチェンバーの 6 つの層からなっており、半径 2.95m から 3.69m までを囲んでいる。

MRPC　MRPC(Multigap Resistive Plate Chambers) は粒子の飛行時間差 (TOF) 測定を行い、運動量と飛行時間差の関係から各粒子の質量を算出し、膨大な荷電粒子の 99%以上を占めるハドロンを識別する。半径 3.95m から 4.31m までの間隔にあり、擬ラピディティ領域 $-0.9 < \eta < 0.9$ を覆う円筒形検出器の最外殻のため、約 160K チャンネルで検出器面積は約 140m^2 と大きくなる。

HMPID　HMPID(High Momentum PID) は高速で飛行する粒子の発する光を捉えることにより、高運動量ハドロンの粒子識別を行っている。TOF 測定法では高運動量ハドロンの粒子識別は困難なため、中心ラピディティ領域に限り TOF 検出器の外側の方位角幅 60 度を覆う空間に置かれている。

PHOS　PHOS(PHoton Spectrometer) は PWO(タングステン酸鉛単結晶) と APD(アバランシェ・フォトダイオード) を使って作られた、高分解能の電磁カロリーメータであり、上記の検出器で検出されることのない光子の測定を行う。PWO は、密度が 8.28g/cm^3、モリエール半径が 22mm、放射長が 8.9mm と高い密度で短いモリエール半径や放射長をもつため、検出器の細分化が可能である。APD は小型で磁場中でも使え、量子効率が高いなどの特徴がある。また PWO の欠点として光量が少ないこと、APD の欠点として増倍率が小さく雑音が大きいということが挙げられるが、単結晶から前置増幅器までを-25 ℃に冷却することで、光量を増やし、雑音を小さくしている。

PHOS は HMPID とほぼ同じ半径上で方位角をずらした中心ラピディティ領域に置かれ、衝突点からできるだけ均一距離に配置するため 5 基の同型モジュール構造に分割し、半径 4.4m の円弧状に配置されている。1 基のモジュールは 20 度の方位角を張り、5 基のモジュールにより方位角幅 100 度、軸方向には擬ラピディティ領域 $-0.12 < \eta < 0.12$ を覆っている。$22 \times 22 \times 180$mm^2 の結晶を用いて、56×64 個の PWO+APD を 1 基のモジュールとし、5 基のモジュールで計 17920 チャンネルとなる。

EMCAL　EMCAL(Electromagnetic Calorimeter) は、PHOS と HMPID により残された方位角領域に置かれ、方位角幅 110 度を覆うサンプリング型鉛シンチレータ電磁カロリーメータである。エネルギー分解能は求めないが、PHOS 検出器に比べて広い擬ラピディティ領域 $-0.7 < \eta < 0.7$ を覆っている。

事象識別検出器

FMD　FMD(Forward Multiplicity Detector) はシリコン短冊状検出器であり、$|\eta| > 1.7$ の荷電粒子多重度を測定する。

PMD　PMD(Photon Multiplicity Detector) は前置電磁シャワー形成部を備えた比例増幅ガス検出器であり、前方ラピディティ領域 $2.3 < \eta < 3.5$ で光子多重度を測定する。

ZDC　ZDC(Zero Degree Calorimeters) は衝突で発生した中性子数を測定し、衝突中心度の決定に使われる。2 つのハドロンカロリーメータで、それぞれ衝突点から両側に 116m の位置にある。

T0　T0 検出器は衝突時刻を 50ps 以下の精度で決定する。

V0　V0 検出器は衝突中心度の決定に使われる。

ミュー粒子検出器

ミュー粒子検出器はソレノイド電磁石外側のビーム軸下流に置かれる。ビーム軸に近いため生成ミュー粒子の運動量は極めて高くなり、ハドロン粒子を遮蔽する吸収材を簡単に突き抜けてくるが、他方、その運動量を精度良く測定するには強力な磁場が必要となる。そこで大型二極電磁石を配置し、その前後に飛跡検出器と粒子識別用吸収層と識別検出器層の積層構造を置いている。ミュー粒子検出は電子陽電子対による測定と同等の物理を追求する相補的な測定法であり、J/ψ や Υ などのベクトル中間子生成に関わる研究を進める。

2.3.2　ALICE 実験のデータ

2 つの原子核が検出器の中心で衝突し、検出器を横切って数千の二次粒子を生成し、そのときにデータが記録される。それぞれの二次粒子は検出器要素の中にいくつものヒットをつくる。すべてのイベントに対するすべてのチャンネルで完全に読み出しをするならば、50GB/s をこえるほどのデータレートとなるが、オンラインの処理などにより最終的なイベントサイズは中心衝突で 12.5MB になる。そのイベント記録レートは約 100Hz となり、永続的なストレージへのデータレートは約 1.25GB/s となる。

RAW データの読み出しと記録は ALICE の DAQ である DATE によってなされる。データは実験サイトに配置されたディスクキャッシュに記録され、そこには 24 時間分のデータを保つことができる。そして、CERN ネットワー

クを通って、Meyrin サイトのコンピューターセンター (ALICE 実験サイトから約 3km) にある CERN の永続的なストレージシステムの CASTOR に転送される。

2.3.3 ALICE 実験のデータ形式

物理結果を得る段階で、RAW、ESD、AOD と呼ばれる 3 つのデータ形式が存在する。

- RAW

  DAQ もしくはシミュレーションによって記録されるデータ。

- ESD(Event Summary Data)

  RAW データをリコンストラクションすることによって得られるデータ。

- AOD(Analysis Object Data)

  解析に必要な計算資源を減らすために、ESD から情報の一部を抽出したデータ。

RAW データは様々な検出器要素へのヒットに対応する形で記録される。物理解析を行うために、検出器を横切る単一の粒子によって生成されるヒットをつながぐことにより飛跡とし、それはその後物理のパラメータ (位置、運動量など) を抽出するのに使われる。ヒットを含む RAW データファイルは、飛跡パラメータを含む ESD に変換される。RAW から ESD フォーマットで、データ量は 5 分の 1 に減少することが予想される。

ESD は Tier 1 センターに運ばれる。RAW データは CERN から Tier 1 センターにネットワーク許容量が許す速度で複製される。したがって元のデータの第二の複製を作っている (CERN の RAW データファイルは永続的なストレージに保存される)。ひとつの Run に対応する RAW データはひとつの Tier 1 センターにだけ複製される。その結果の ESD ファイルはそれらが作りだされた Tier 1 センターに保存される。元の RAW データの再解析によって ESD ファイルはいつでも作り出すことができるので、ESD データのバックアップ施設は一般には要求されない。

最後の物理信号を得るために、さらなる選択が飛跡のサンプルになされなければならない。まずはゆるいカットを使うことによって、ESD から AOD へと変換される。ESD と比べて典型的にデータ量は 10 分の 1 に相当する。そして最終的に解析を行うことによって最後の物理信号を得る。

AOD 生成と AOD データに基づいた最終的な解析は、Tier 1 と Tier 2 センターの両方で行われる。同じ RAW データ/ESD に基づいて作られたいくつかの AOD が、異なった選択基準によって存在することになる。AOD ファイルは典型的に Run の期間全体のデータを含むので、異なった Tier 1 セン

ターに存在する ESD ファイルに基づいている。エンドユーザーの解析はほとんど AOD データに基づいて行われる。

2.3.4 ALICE 実験のコンピューティングモデル

ALICE 実験を含む 4 つの LHC での実験では全て、MONARC[3] というモデルに基づいたコンピューティングモデルを使っており、そのモデルでは資源は階層構造をもっている。まず CERN にある Tier 0 を中心とし、それに直接繋がっているコンピューティングセンターである Tier 1、次により小さな地域のコンピューティングセンターとして Tier 2 があり、Tier 1 を通して CERN と繋がっている。その概念図を図 2.3 に示す。

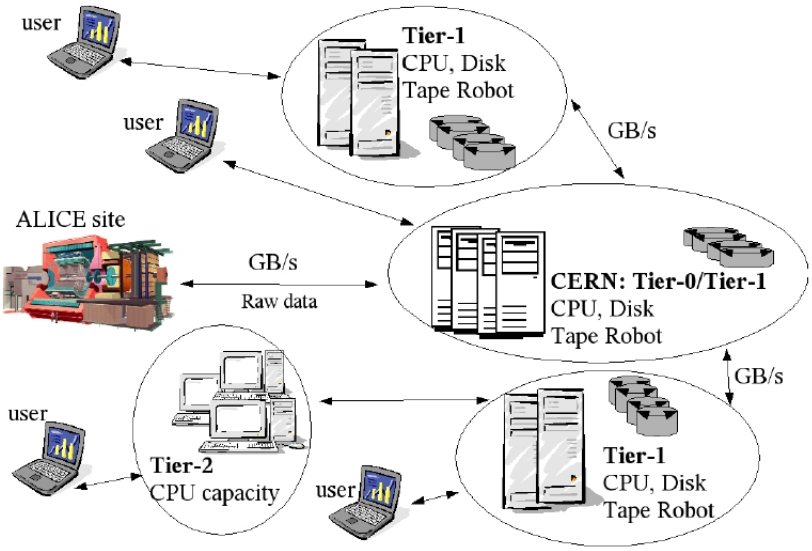


図 2.3: ALICE のコンピューティングモデル。Tier 0、Tier 1、Tier 2 の階層構造となっている。

- Tier 0 の約割り

  first pass リコンストラクションを行う。RAW データ、キャリブレーションデータ、first pass ESD の保存をする。

- Tier 1 の約割り

  subsequent リコンストラクションとスケジュールされた解析を行う。RAW データや ESD や AOD の保存をする。

[3]Models Of Networked Analysis at Regional Centers

- Tier 2 の約割り

  シミュレーションとエンドユーザーの解析を行う。ESD や AOD の保存をする。

ALICE コラボレーションは 8 つの Tier 1 センターをセットアップするよう計画している。CERN、Lyon、Catania、Karlsruhe、Rutherford Appleton Laboratory in Didcot(UK)、Amsterdam、collaborating US center、distributed center in the Nordic countries の 8 つである。広島が Tier 2 として繋ぐ Tier 1 センターはフランスの Lyon を予定している。

第3章　Grid

3.1　Gridとは

Gridとは、コンピュータ、ネットワーク、ストレージなどの資源を仮想コンピューティング・システム化することで、Gridを利用するユーザーやアプリケーションが、その莫大な情報資源を利用できるようにするものである。Gridのユーザーは、インターネットユーザーがWebコンテンツを見るように、大規模な仮想コンピューターをひとつのシステムとして利用することができる。ネットワーク上の情報資源 (サービス) を安全に安定して簡単に利用することが理想である。つまり、セキュリティを確保して必要なときに必要なだけネットワークを意識せずに利用したいのである。そのためにユーザー認証、資源管理、データ管理、情報サービスなどに様々な技術が用いられている。Gridの分類としては、コンピューティングGridやデータGridなどがある。

またGridでは、あるひとつの目標に対して、インターネット上の複数の資源がコラボレーションで作用することを可能にしている。

Gridは現在様々な分野で使われており、物理では高エネルギー物理だけでなく、原子核物理や宇宙線物理、他には核融合、計算化学、新薬開発などにも利用されている。

3.1.1　Gridミドルウェアと組織

Gridの発想は、1990年代後半のWorld Wide Webの成功が起源となっている。Gridを構成するためには、セキュリティを考えつつ資源の効率的な利用やそれらの利用を平等に行えるような仕組みが必要である。また、分散しているサーバーの情報へアクセスする際に、認証を要求せずに行うことができるようにすることも重要である。そして、ユーザーがGridの構造に対する詳細な知識を持たずとも、計算資源をうまく使って処理されるようにすべきである。それらのことを実現するためには、オペレーティングシステムとアプリケーションソフトウェアの中間層としてのミドルウェアが必要である。

WLCG[1] 計画は主にLHC実験のために必要な、世界中に分散配置された計算資源を利用するための計画である。CERNの計算機資源だけでは、LHC

[1] Worldwide LHC Computing Grid

加速器を用いる 4 実験のデータ解析を行うには不十分であるためにこのような計画が必要とされている。WLCG では、2004 年に終わった EDG[2] 計画のような他の Grid 組織とのコラボレーションの中で、LHC での解析を実行するために必要なミドルウェアの開発を手がけている。EDG は EU が資金を出した計画の EGEE[3] へと変わり、それは gLite と呼ばれるミドルウェアを開発している。LCG では以前は LCG と呼ばれるミドルウェアを開発していたが、今は gLite へ移っていっている。

ヨーロッパでの活動に平行して、大きなスケールの Grid 組織がアメリカにもある。Globus 計画は長年 Grid ミドルウェアを開発しており、多くの他のミドルウェアの基礎を形作っている Globus Toolkit の開発元である。

3.1.2 GOC

LCG では GOC[4] が Grid 全体の運用を調整しており、それは機器構成の情報や連絡先の詳細のような運用情報の中心地点としての役割としても働いている。GOC では Grid 全体の運用状態の監視、Grid の最適な運用のためのメカニズムの開発や管理、エラーの検出、問題の管理、特定、追跡などを行っている。また、Grid に参加する機関に対してのサポートも行っている。

GOC はある地域ごとに存在し、ROC[5] と呼ばれる。わたしたちの場合は、APROC[6] に主にサポートしてもらいサイトを構築した。

3.1.3 仮想組織

仮想組織 (Virtual Organization, VO) とは、同一の目標を達成するために選択された資源とユーザーの動的な集合である。資源の割り当ては VO 単位で管理されるため、ユーザーは必ずいずれかの VO に属する必要がある。広島大学の場合は、ALICE VO に属することになる。

3.1.4 ALICE のサイト

世界中に散在する Grid の単位をサイトと呼び、基本的に大学、研究所単位である。それぞれの Grid には複数のサイトが属していることになり、個々のサイトには複数の計算機があり、なんらかのローカルスケジューラで管理されている。図 3.1 および図 3.2 で ALICE のサイトを示す。

[2] European Data Grid
[3] Enabling Grids for E-sciencE
[4] Grid Operation Center
[5] Regional Operation Center
[6] AsiaPacific Operation Center

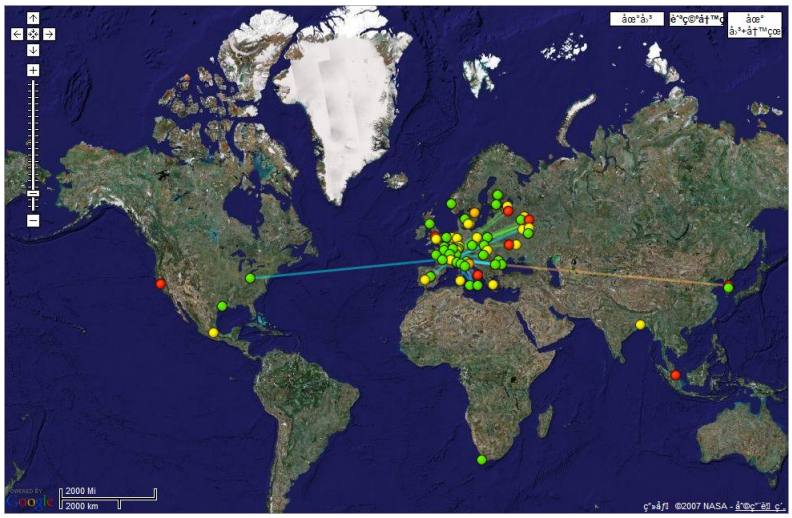


図 3.1: ALICE のサイト

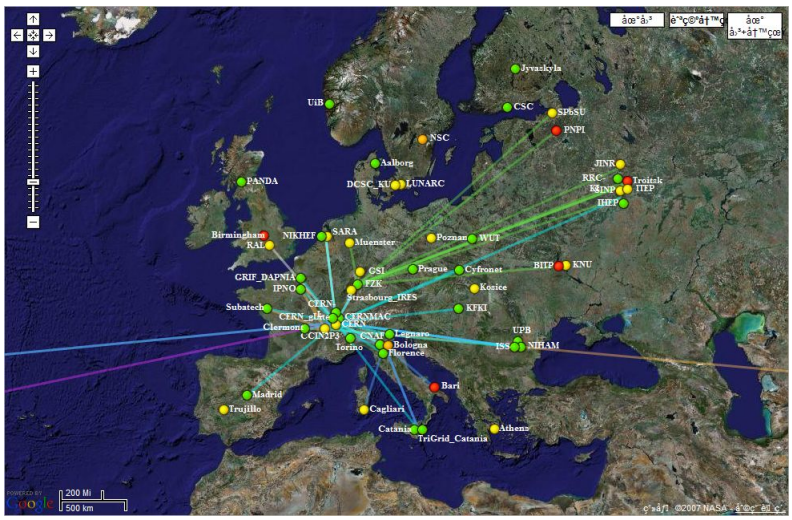


図 3.2: ヨーロッパにおける ALICE のサイト

3.1.5　各ノードの説明

Grid はそれぞれのマシンがある役割を果たすようミドルウェアがインストールされなければならない。そのマシンのことをノードと呼んでおり、以下でそれぞれの役割について述べる。

VO-BOX

VOBOX は 2006 年夏から (LCG2.7 から)WLCG によって提供されているサービスであり、LHC の 4 つの VO で利用できる。そして gLite3.1 へ向けてアップグレードされている。VOBOX の特徴として以下のことがあげられる。

- 実験グループ (VO) がアクセスし、サービスなどを走らせることができる。
  VO-BOX のホスト証明書は、CERN のプロキシサーバー (myproxy.cern.ch) に許可されていなければならない。プロキシを VO-BOX に登録することで sgm アカウントを使い VO-BOX へログインできる。VO-BOX は普通は myproxy サーバーとコンタクトして、sgm のプロキシをリフレッシュする。

- 実験グループのソフトウェアをもつエリアがある。
  環境変数 VO_ALICE_SW_DIR で指定されていなければならず、そのエリアは WN と VO-BOX の両方からマウントされていなければならない。VO-BOX へのアクセスは、sgm アカウントにのみ保証されていなければならなく、ソフトウェアエリアへの書き込み権限は sgm アカウントにのみ保証される。

ALICE では Tier 0、Tier 1、Tier 2 の全てのサイトが資源を ALICE に提供するために VOBOX を置くことを要求している。

Computing Element:CE

このサーバーはコンピューティングリソースのゲートウェイとして働き、バッチシステムを管理する。具体的には RB から受け取ったジョブをバッチシステムにより WN で計算させる。

CE のコンポーネント　CE は以下のもので構成されている。

- gatekeeper
  - RB からのジョブを受け取る
  - CE へのアクセスを認める：リモートユーザーを認証して権限を与える
  - jobmanager にジョブを渡す
- Job manager
  - ローカルバッチシステムへのインターフェイスを提供する
  - ジョブをサブミットするかキャンセルするかのみを行う

  - grid monitor がジョブの状態を問い合わせる

- Batch system(torque/maui)
  - 利用できる WN 上でのジョブの実行を扱う
  - バッチシステムは、torque resource manager と maui job scheduler から成り立っている

Torque　Torque[7] は、バッチジョブや分布した計算資源を越えてのコントロールを提供するリソースマネージメントである。Torque システムは以下のものから成り立っている。

- pbs_server 基本的なバッチサービスを提供する
- job scheduler どのジョブを実行しなければならないかを決定するために使われるサイトのポリシーを含む
- pbs_mom ジョブを実行の状態に置く

Maui　現在 LCG でサポートされているいくつかのバッチシステムに対するジョブスケジューラーとして使われる。資源をコントロール、予約、制限する。

Worker Node:WN

ここでは実際にジョブが実行され、それは CE のバッチシステムによって管理される。Grid クライアントコマンドとライブラリーがここにインストールされる。また、ジョブの実行に必要な input sandbox を RB から、output sandbox を RB へ転送する。

Storage Element:SE

ストレージ資源を Grid に提供する。典型的には DPM[8] ソフトウェアを使う。これにより、単一のストレージプールを形作るためにいくつかのディスクサーバーを管理することができる。つまり、規模が大きなサイトでは、複数のディスクノードをヘッドノードが統括するという形になる。

User Interface:UI

ユーザーがログインして Grid にアクセスするために用いられるマシン。

[7] Tera-scale Open-source Resource and QUEue management
[8] Disk Pool Manager

MON

各サイトをモニターするためのマシン。

Barkely Directory Information Index:BDII

トップ BDII には、Grid には何があるのかという情報があり、サイト BDII には、そのサイトの名前や情報検索に必要な名前が書かれている。一般的にはトップ BDII は中心のマシンを使い、サイトごとにサイト BDII を置く。

Resource Broker:RB

Grid 資源情報を参照することにより、ジョブを割り当てるのに適切な資源を探してくれる。このマシンも通常は各サイトでもつ必要はなく、中心のものを使う。

3.1.6 認証のメカニズム

PKI

Grid は世界中の様々な人が使うため、誰が Grid にアクセスしているのかを知る仕組みが必要があり、それに用いられる仕組みが公開鍵基盤 (PKI) である。

公開鍵と秘密鍵

公開鍵暗号は非対称暗号とも言われ、公開鍵と秘密鍵という鍵対を使う。公開鍵で暗号化された文書を秘密鍵で平文化できる。公開鍵を使って暗号化された情報は、秘密鍵を手に入れない限り見ることができない。また逆に、秘密鍵で暗号化された情報を公開鍵で平文化することもできる。

ユーザー認証

秘密鍵で暗号化された情報 (証明書) を公開鍵で平文化することもでき、ユーザー認証にはこの方法が利用される。ユーザーは証明書をネットワークを通して送り、認証する側はユーザーにチャレンジストリングを送る。そして、ユーザーがチャレンジストリングを秘密鍵で暗号化し、認証する側が公開鍵で復号化することによって、ユーザーが秘密鍵を所有しているということがわかるので証明書に書かれている subject として認証されることになる。

証明書に添付された署名はだれでも公開鍵で平文化することができるが、秘密鍵がないと証明書を偽造することはできない。ユーザーが本人であるこ

とを証明する公平な第三者機関があり、認証局[9] と呼ばれる。認証局は登録局で審査に合格したユーザーについて証明書を発行し、その証明書には認証局の署名がついている。この証明書つきの鍵対を使うことによってユーザーを認証している。また、秘密鍵を使う場合は、それが持ち主によってなされていることを確認するためにパスフレーズを要求される。

私たちの場合は日本では KEK GRID CA が LCG もサポートしているため、そこから証明書を発行してもらうことになる。

登録局

認証局は証明書を発行することを役割としているが、証明書を発行すべきかの判断は登録局[10] で行う。通常は面接や電話での確認を伴い、証明書発行を要求する人が本人であることを登録局は確認する。

プロキシ認証

Grid での作業は非常に沢山の数のサーバーを通して行うことになるため、その都度パスフレーズを入力する必要があり実際的ではない。そのため PKI インフラを拡張し、プロキシ (代理) 認証という仕組みを用いる。プロクシ認証ではユーザーの鍵対の代わりに一時的な証明書を作成し、それを用いて認証する。これにより、シングルサインオンを実現している。

3.1.7 情報サービス

Grid 上にどのようなサイトがあって、どのような資源が割り当てられており、どのくらい使われているかなどの情報を効率的に集めるために、情報検索の仕組が用意されている。アクセスプロトコルとして LDAP[11] をベースとし、情報のフォーマットを定義するために GLUE[12] スキーマを使っている。また、図 3.3 のようにトップ BDII、サイト BDII、GRIS が階層構造を成すことにより情報を収集している。その概念図を図 3.4 に示す。GRIS で生成された情報が、サイト BDII で集められ、さらにトップ BDII により集められる。以前は BDII の代わりに GIIS[13] が使われていた。

[9] CA(Certification Authority)
[10] RA(Registration Authority)
[11] Lightweight Directory Access Protocol
[12] Grid Laboratory for a Uniform Environment
[13] Grid Index Information Service

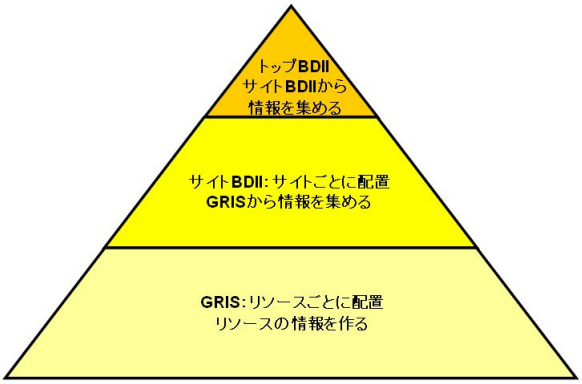


図 3.3: 情報サービスの構成。トップ BDII、サイト BDII、GRIS が階層構造を成している。

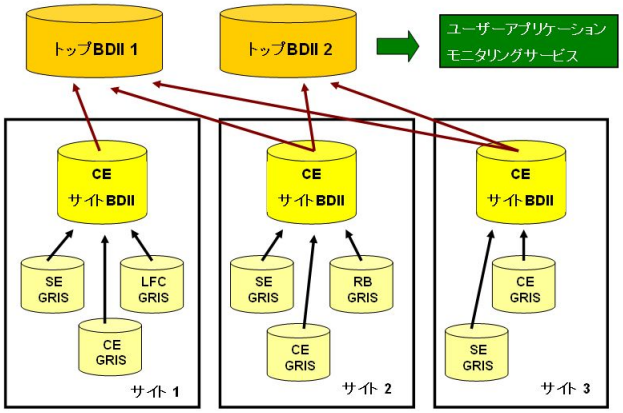


図 3.4: 情報サービスにおけるサービスのつながり。GRIS で生成された情報が、サイト BDII で集められ、さらにトップ BDII により集められる。

LDAP

LDAP とはディレクトリサービスへアクセスするためのプロトコルであり、その情報の構造はツリー構造である。ディレクトリサービスとは、ネットワークを利用するユーザー名やマシン名などの様々な情報を管理するためのサービスのことで、ユーザー名などのキーとなる値から様々な情報を検索することができる。LDIF[14] というファイルに情報が記述されており、そのエントリは DN[15] により一意に識別されている。

情報サービスのデータは GLUE スキーマという情報規格化組織が決めたスキーマ (データベースのテーブル) に従っており、それは Grid のリソースを

[14]LDAP Data Interchange Format
[15]Distinguished Name

描写するための一般的なデータモデルを提供する。

トップ BDII

Grid には何があるのかという情報を問い合わせるエントリーポイントであり、Grid の情報を調べる一番最初の手がかりがある。その情報はサイト BDII から集める。

サイト BDII

サイト BDII には接続されているサイトの名前と情報検索に必要な名前などが書かれている。ここにはサイトが持っている CE、SE、RB などの情報への目次がある。その情報は GRIS から集める。

GRIS

個別の CE や SE などの Grid 資源は自分のもっている資源の詳細な情報を提供する。それが GRIS[16] というサービスで、これも LDAP をもとにしており直接問い合わせることができる。

3.1.8 データ管理

SE は複数のディスクプールを管理せねばならず、その要求を満たすために DPM[17] などが用いられる。また、SRM[18] により異なったストレージシステムに保存されたデータでも扱うことができ、GridFTP によりセキュアなファイル転送を実現している。DPM での GridFTP サーバーは、通常の GridFTP サーバーを少し修正したものとなっている。そしてデータは異なった場所に保存されるため、LFC[19] と呼ばれるファイルカタログにより、一様な外観を提供している。

データ管理における名前

ストレージにあるデータファイルは世界中から参照できるようにするため、ファイルカタログに登録されなければならない。そして効率の面から考えてそのファイルはレプリカが分散配置されることがある。レプリカであっても同じファイルを参照するため、同じファイルのレプリカは GUID[20] という 40

[16] Grid Resource Information Service
[17] Disk Pool Manager
[18] Storage Resource Manager
[19] LCG File Catalog
[20] Globally Unique Identifier

バイトの文字列により識別される。この GUID は 16 進数の表現であり実用的でないため、その GUID に対応する論理ファイル名[21] をつける。

一方、GUID には複数のレプリカが対応しており、それぞれ格納されている SE の名前やそこでの格納場所などの情報で表される。それは物理ファイル名に該当するが、SURL[22] と呼ばれる。つまり、ひとつの GUID には任意数の SURL が割り当てられることになる。SURL は場所を表しているが、そのファイルにアクセスするためにはそこの SE がサポートする転送プロトコルを知る必要があるため、SURL から作られる TURL[23] という表現を用いることによってファイル転送に必要な情報を得ることができる。

それらの名前の関係を図 3.5 に示す。

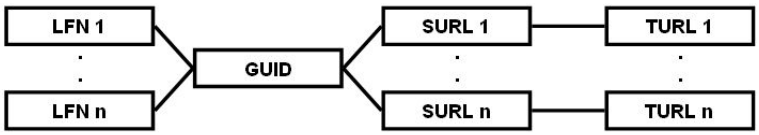


図 3.5: データファイルの名前の関係。一般的には、ひとつの GUID に対して複数の LFN および SURL、TURL が割り当てられることとなる。

3.1.9 Grid ジョブの概要

ジョブ記述ファイル

Grid は全体で一つの仮想的な計算機と考えることができる。しかしそれぞれのサイトは固有の資源、ハードウエア、OS、ソフトウエア等で構成されているため、共通の言語である JDL[24] で書かれたジョブリクエストをそれぞれのサイトの固有の制御情報に変換してジョブを実行させることになる。

JDL に使われる言語は、ClassAd[25] 言語という Condor プロジェクト (計算機クラスター制御のプロジェクト) で開発されたものを元にしている。ユーザはジョブをこの ClassAd で記述することになる。

ジョブの流れ

ジョブの流れを図 3.6 に示す。

[21] LFN(Logical File Name)
[22] Storage URL
[23] Transport URL
[24] Job Description Language
[25] Classified Advertizement

1. 信頼された CA からデジタル証明書を受け取り、VO に登録し、UI のアカウントを得れば、ユーザーは Grid を使う準備ができる。

2. ユーザーはジョブを UI から RB へサブミットする。このとき、UI から WN へコピーするためのファイルを、はじめに RB へコピーする。このファイルのセットは Input Sandbox と呼ばれる。ここでジョブの状態は SUBMITTED となる。

3. WMS はジョブを実行するために最も便利な CE を探す。計算やストレージの資源の状態を見るために BDII が、インプットファイルの場所を見つけるためにファイルカタログが用いられる。ここでジョブの状態は WAITING となる。

4. RB はジョブを CE へサブミットするための準備をする。ここでジョブの状態は READY となる。

5. CE はジョブのリクエストを受け取り、そのジョブを実行するために LRMS へ送る。ここでジョブの状態は SCHEDULED となる。

6. LRMS は WN でのジョブの実行を扱う。Input Sandbox のファイルは RB からジョブが実行される WN へコピーされる。ここでジョブの状態は RUNNING となる。

7. ジョブのアウトプットファイルは Grid へアップロードし、他の Grid ユーザーが利用できるようにすることができる。ファイルを Grid へアップロードするということは、SE へコピーしファイルカタログへ登録するということである。

8. もしジョブがエラー無しで終わったならば、アウトプットファイルのサイズが小さいならば Output Sandbox として RB ノードへ転送される。ここでジョブの状態は DONE となる。

9. ユーザーはそのジョブのアウトプットを UI へ回収する。ここでジョブの状態は CLEARED となる。

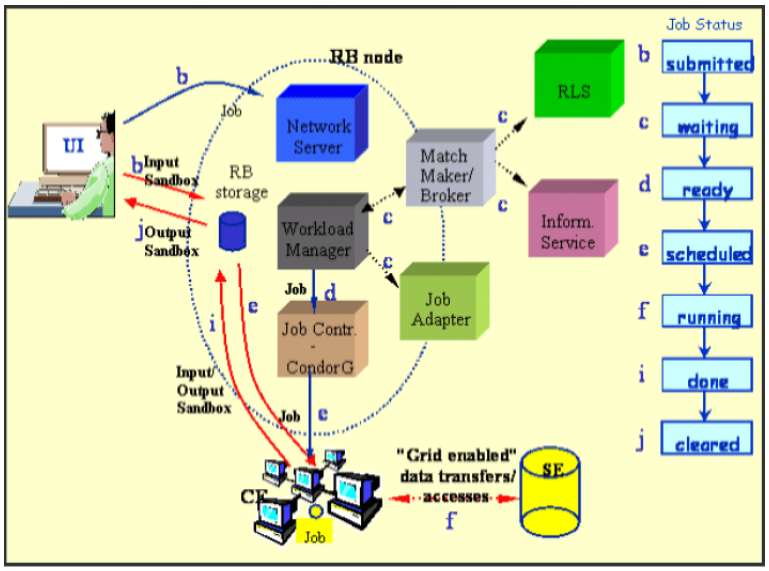


図 3.6: WLCG Grid でのジョブの流れ。

ジョブの状態の問い合わせは、UI から LB へ行うことができ、また UI から BDII へ資源の状態を問い合わせることもできる。

もしジョブを送られたサイトがそれを受け入れたり走らせることができなかったならば、そのジョブは自動的にユーザーの要求を満足する他の CE へ再サブミットされる可能性がある。再サブミットの最大回数に達したら、そのジョブは中止される。ユーザーはジョブの経過についての情報を、LB サービスへ問い合わせることにより得ることができる。

3.2 ALICE における Grid 環境及び解析環境

3.2.1 AliEn

AliEn[26] システムは ALICE コラボレーションによって、シミュレーションやリコンストラクションや物理データの解析に対しての環境として、世界中に分布したコンピュータシステムへのアクセスを提供するために 2001 年に開発が始まった。

AliEn は、LDAP を使って構成を維持し、SOAP やウェブサービスを使って XML メッセージを交換する、最近のインターネット標準の上に成り立っている。AliEn をインストールすることで、そのシステムは AliEn を ALICE

[26] ALICE Environment

解析のトップレベルのユーザーインターフェイスとして残したまま、簡単に他のミドルウェアシステムと結びつけて適用することができる。

3.2.2 ALICE 解析のソフトウェア

ROOT

高エネルギー物理学で一般的に用いられている解析ソフト。全ての ALICE オフラインソフトウェアは ROOT に基づいている。ROOT フレームワークは AliRoot で利用されるいくつかの重要な要素を提供する。

AliRoot

AliRoot はシミュレーションやリコンストラクションや物理データの解析のための ALICE オフラインフレームワークである。それは ROOT を土台として使っている。

GEANT

GEANT は媒質を通過する粒子の輸送をシュミレートする。最初は高エネルギー物理実験のために設計されたが、今日では医療、生物科学、放射線防護、宇宙航行学のような分野にも利用されている。

GEANT により、ある実験セットアップでの検出器応答のシミュレーションのための粒子のトラッキングや、セットアップや粒子の軌跡のグラフィック描写をすることができる。

FLUKA

FLUKA は様々な粒子に対する輸送を扱うことができる素粒子事象シミュレーションプログラムである。その開発は、CERN において 1960 年代から継続的に続けられている。

第4章　構築、検証

4.1　ALICE でのイベント数、データ量

Pb-Pb5.5TeV で目標としているルミノシティ$5 \times 10^{26} cm^{-2} s^{-1}$ が達成されたとき、衝突レートは4×10^{3}Hz となる。そのときのイベントサイズは平均して 12.5MB であるため (重イオン衝突の場合、衝突中心度によりデータ量は大きく異なる)、データ転送レートの要請からイベント記録レートは 100Hz となる。1 ケ月 Run が行われるとし、実効的な時間は約10^{6}s と考えると、10^{8}イベント取れることになる。

RAW データは複製され 2ヶ所で保存されるため、RAW データだけでもそのデータ量は、

$$12.5MB/event \times 10^{8} event \times 2 = 2.5PB$$

となり、それだけでも 2.5PB という莫大なデータ量となる。

また、リコンストラクションのみにかかる時間を測定すると、1 コア (Xeon 5160 3.0GHz) あたり、1 イベントあたりで約 9 分かかることがわかる。そのため、例えば重イオンの Run が 1 ケ月行われ、それを 1 ケ月でリコンストラクションするならば、

$$9\text{ 分}/event \times 10^{8} event \div (60\text{ 分} \times 24\text{ 時間} \times 30\text{ 日}) = 20833$$

となり、約 20000 コア必要であることがわかる。そのため、次の Run に間に合うように処理し解析するためには、大きな計算力が必要となることが予想される。

ALICE 実験グループにより見積もられた、試運転の期間ではなく標準データ収集年における必要な資源を、表 4.1 に載せた。またこの表には、そのうち Tier 2 センター全体として要求される資源について、また Tier 2 センターに要求されるバンド幅についても載せた。

表 4.1: 標準データ収集年において ALICE 実験で要求される資源。Tier 2 のみについてと、全ての Tier について示している。また Tier 2 に要求されるバンド幅も示している。

	Total	Tier 2
CPU(MSI2k)	35.0	14.4
Transient storage(PB)	14.1	5.1
Permanent storage(PB/year)	10.6	-
Bandwidth in(Mb/s)	-	10
Bandwidth out(Mb/s)	-	600

このような大量のデータを扱う実験において Grid は必要不可欠なものであり、それを実現するためにも各々のグループが資源を出し合い参加していかなくてはならない。以下の節ではそのための、広島 WLCG Tier 2 サイトの構築について述べる。

4.2 研究室内部のマシンおよびネットワーク構成

4.2.1 マシンのホスト名と役割

各マシンのホスト名、役割、IP アドレスを以下に示す。

- grid01 Grid での VOBOX。
  IP : 202.13.220.41, 172.16.10.241
- grid02 Grid での CE。
  IP : 202.13.220.42, 172.16.10.242
- grid03 Grid での MON, SE。
  IP : 202.13.220.43, 172.16.10.243
- grid04 Grid での UI。
  IP : 202.13.220.44, 172.16.10.244
- nfs01 IP : 172.16.10.221
- nfs02 IP : 172.16.10.222
- nfs03 IP : 172.16.10.223
- qs01 ALICE offline code サーバーおよび qx017〜qx064 に対する Condor サーバー。
  IP : 172.16.10.238

- qx001〜qx016 Grid での WN。
  IP：172.16.10.1〜172.16.10.16
- qx017〜qx031, qx034〜qx064 ローカルクラスターでの実際に計算を行うマシン。
  IP：172.16.10.17〜172.16.10.31、

また、これらのマシンのネットワーク構成を図 4.1 に載せた。

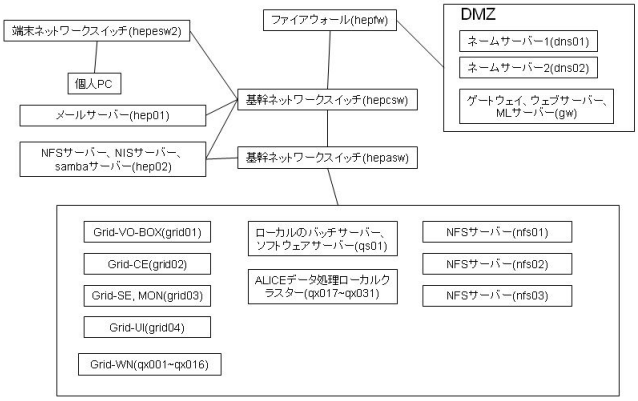


図 4.1: クォーク物理学研究室のマシンの役割とネットワーク構成。

4.3 Grid サイトの構築

第 3 章で述べた Grid の仕組みとしての、認証、情報サービス、データ管理やジョブの扱いなどの環境を作るために主に Grid ミドルウェアを用いてサイトを構築する。そしてその環境を確かめることも行う。

4.3.1 各マシンの役割

Grid サイトを構築するにあたってまず考えなければならないことは、そのサイトを構成するマシンにどのように約割を割り振るかである。ALICE の Tier 2 サイトとして働くために必要なノードは、VOBOX、CE、WN、SE、MON、UI の 6 つである。（ちなみに、BDII、RB、PROXY などは CERN や APROC のマシンを使っている。）

結論としては、各マシンの役割は表 4.2 のようにした。以下でその理由を述べる。

現時点で SLC4 に対応したミドルウェアが WN, UI の 2 つであるため、それ意外のサービスは SLC3 のマシンへ導入せねばならない。もちろん今後こ

れらのミドルウェアはSLC4にむけて開発されていくので、それに応じて変更していく必要はある。しかし今回はその規制のため、grid01〜grid03をSL3もしくはSLC3、grid04とqx001〜qx016をSLC4とした。

VOBOXとCEに対し、それぞれ一つのノードを割り当てたのは、これらはともに負荷が大きく、他のサーバーと共存させることが困難なためである。またgrid02ではCEと同時にサイトBDIIの役割も担っている。

grid03はSE(DPM head node、DPM disk node)、MONとした。本来DPMとMONを同じマシンへインストールすることは推奨されてはいないが、使用するポートを調整することで可能であるため同じマシンへインストールすることにした。今後、ディスクスペースは拡大する予定である。今回は大きなディスクをもっているマシンのOSがまだ現在のミドルウェアではサポートされていないために使っていない。また、NFSを用いてディスクをサーブすると、DPMが時々ハングしたりファイルが破損するという問題が知られているためNFS使うことはできない。

そしてSLC4をインストールしたgrid04をUIとし、残りのマシンのqx001〜qx016をWNとした。

表 4.2: 各マシンの役割とそのOS

grid01	VO-BOX	SL3
grid02	CE	SL3
grid03	MON, SE(DPM head node, DPM disk node)	SLC3
grid04	UI	SLC4
qx001〜qx016	WN	SLC4

4.3.2 GOCDBへの登録

まず初めに、GOCDB[1]への登録が必要である。LCGのサイトとして認められるためには、GOCDBへ登録されている必要がある。日本の場合はGOCはAPROCなので、APROCへ決められたフォーマットに従って情報を提出する。それが承認されたならばAPROCによりGOCDBへ登録され、それ以降はサイト管理者がGOCDBのウェブページから情報を書き換えることができる。

実際に広島のサイトの情報は、他サイト同様GOCDBのウェブページから閲覧することができる。
https://goc.gridops.org/site/list?id=205001
その様子を図4.2、図4.3に示す。

[1]Grid Operation Center DataBase

図 4.2: GOCDB のウェブページにおける広島サイトの情報。

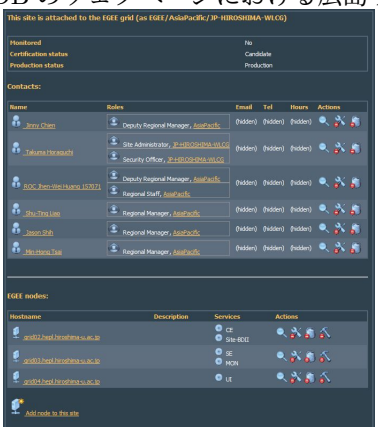


図 4.3: GOCDB のウェブページにおける広島サイトのコンタクトおよびノードの情報。

4.3.3 ホスト証明書の取得

Grid の認証の仕組みを実現するために、WN、UI、BDII を除く全てのノードには、CA から発行されるホスト証明書が必要である。ユーザー証明書と同様に、ホスト証明書も KEK から取得した。

必要なファイルは、秘密鍵を含む hostkey.pem と、公開鍵および証明書を含む hostcert.pem であり、セキュリティのため、hostkey.pem は root 権限では読み込みのみできるように、hostcert.pem は root 権限では読み込みおよび書き込み、その他の全ての人から読み込みのみできるようにしておかなければならない。それらのファイルは具体的には、各マシンの/etc/grid-security/以下に置いておく必要がある。

```
[root@grid01 root]# ls -l /etc/grid-security/*.pem
-rw-r--r--    1 root     root         1432 Oct  9 19:43 hostcert.pem
```

```
-r--------    1 root     root          887 Oct  9 19:43 hostkey.pem
```

4.3.4 各マシンの設定

OS のインストール

現段階で推奨されているミドルウェアは、WN、UI の 2 つは gLite3.1 であり、それ以外のノードは gLite3.0 である。WN、UI 以外のノードもいずれは gLite3.1 に移行していくはずである。そしてこれらに推奨されている OS は、gLite3.0 では SLC3、gLite3.1 では SLC4 である。そこで、grid01〜grid03 へは SLC3.0.8、grid04、qx001〜qx016 へは SLC4.5 をインストールした。

NTP の設定

Grid では様々なマシンが繋がっているため、そのログを残す場合に時間が同期していることは重要なことである。そのため、全てのマシンにおいて NTP[2] を使って時間の同期をする。現在は JST[3] に合わせている。広島大学の NTP サーバー及びクォーク研の NTP サーバー (pxdb) に同期している。

/etc/ntp.conf を編集し、サービスを再起動することで、ある程度の時間待てば同期する。

```
[root@grid01 root]# /etc/init.d/ntpd restart
ntpd: Removing firewall opening for ns.hiroshima-u.ac.jp po[  OK  ]
ntpd: Removing firewall opening for pxdb.hepl.hiroshima-u.a[  OK  ]t 123
Shutting down ntpd:                                        [  OK  ]
ntpd: Opening firewall for input from ns.hiroshima-u.ac.jp [  OK  ]
ntpd: Opening firewall for input from pxdb.hepl.hiroshima-u[  OK  ]ort 123
Starting ntpd:                                             [  OK  ]
[root@grid01 root]# ntpq -p
remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
+pxdb.hepl.hiros ntp-b2.nict.go.  2 u  259  512  377    0.327  -24.897   5.916
*ns.hiroshima-u. ntp.hiroshima-u  2 u  197  512  377    0.421  -10.936   1.792
```

このコマンドにより、ns.hiroshima-u.ac.jp と同期していることがわかる。また delay や offset の値も確認することができる。

NFS の設定

VO-BOX には、実験グループのソフトウェアエリアを配置する必要がある。これは、ALICE により要求されていることで、そのサイトの WN は、VO-BOX 上のソフトウェアを使いジョブを実行することになる。そのため

[2] Network Time Protocol
[3] Japan Standard Time

VO-BOX と WN でそのエリアを共有していなければならない。その要求を満たすために NFS[4] を利用している。

- サーバー側 (VOBOX:grid01) の設定

  1. portmap と nfs のサービスを起動する

     ```
     # service portmap start
     # service nfs start
     ```

  2. /etc/exports を編集して全ての WN に対してエクスポートする

     ```
     # exportfs -ra
     ```

  3. エクスポートができているか確認する。

     ```
     # exportfs
     ```

- クライアント側 (WN:qx001〜016) の設定

  1. /opt/exp_soft のディレクトリを作る

     ```
     # mkdir /opt/exp_soft
     ```

  2. portmap のサービスを起動する

     ```
     # service portmap start
     ```

  3. /etc/fstab を編集してマウントする

     ```
     # mount -a
     ```

  4. マウントができているか確認する

     ```
     # df -h
     ```

Java のインストール

Java は各マシンにインストールされなければならないが、ライセンスの関係でミドルウェアのインストール時に自動的にインストールを行うようにできず、ミドルウェアをインストールする前に事前にインストールを行う必要がある。また、CERN のリポジトリなどに置くことができないため、SUN Java のウェブサイトから RPM をもってくる必要がある。gLite3.0 の場合は Java SDK が必要である (http://java.sun.com/j2se/1.4.2/download.html)。gLite3.1 の場合は Java SDK に加えて Java JDK 1.5 以上が必要である。(http://java.sun.com/javase/downloads/index_jdk5.jsp)

[4]Network File System

YAIM のインストール

続いて、Grid 環境を実現するために最も重要となるミドルウェアをインストールするためのツールをインストールする。ミドルウェアのインストールには YAIM というツールが存在する。YAIM は/opt/glite/yaim にインストールされ、”site-info.def”、”users.conf”、”groups.conf”、”wn-list.conf”の 4 つのファイルを編集し、それを使ってミドルウェアがインストールされる。

”site-info.def”は、4 つのファイルの中で最も重要なファイルで、ミドルウェアのインストールにおけるノードの構成に関する設定など (例えば VO-BOX のホスト名など) を記載しておく。”users.conf”というファイルはプールアカウントの設定ファイルで、CE や WN などのノードに作成されるローカルユーザーが定義が定義されている。

ファイルの一例 (1 部分)

```
151:dpmmgr:151:dpmmgr:x:dpm:
10417:alice001:1395:alice:alice::
10418:alice002:1395:alice:alice::
10420:alice003:1395:alice:alice::
10454:alice004:1395:alice:alice::
```

”groups.conf”というファイルには VOMS のグループをどのようにローカルのグループへマッピングするかについて書く。

ファイルの一例 (1 部分)

```
"/VO=alice/GROUP=/alice/ROLE=lcgadmin":::sgm:
"/VO=alice/GROUP=/alice/ROLE=production":::prd:
"/VO=alice/GROUP=/alice"::::
```

”wn-list.conf”というファイルはバッチシステムの設定のための WN のリストで、そのサイトの全ての WN の名前を書く。

ファイルの一例 (1 部分)

```
qx001.hepl.hiroshima-u.ac.jp
qx002.hepl.hiroshima-u.ac.jp
qx003.hepl.hiroshima-u.ac.jp
```

しかし、gLite もそうだが、YAIM も同時に開発が進行中の段階であり、そのために多少インストール等の手順には気を付けなければならない。

YAIM 自体は RPM が用意されており、YAIM3.0.1 や YAIM3.1.0 をインストールした場合は、ミドルウェアのインストール時に最新版 (現時点では YAIM3.1.1) へ自動的にアップデートされる。

apt もしくは yum の設定

apt もしくは yum を使うことで、ミドルウェアのインストール前に OS をアップデートしておかなければならない。apt の場合は以下のようにコマンドを打つことでアップデートができる。

```
# apt-get update
# apt-get dist-upgrade
```

yum の場合は以下のようにコマンドを打つことでアップデートができる。

```
#yum update
```

また、gLite3.1 の場合はまだ YAIM を用いたミドルウェアのインストールがサポートされていないため、yum を用いてインストールを行った。

ミドルウェアのインストールおよびコンフィギュアー

gLite3.0 の場合は、YAIM を用いてインストールおよびコンフィギュアーが可能である。gLite3.1 では現時点では、コンフィギュアーは可能だがインストールには対応していないため、インストールは yum を用いて行わなければならない。

yaim3.1.1 では/opt/glite/yaim/bin/以下にスクリプトが用意してあり、それを使い、ミドルウェアのインストールやコンフィギュアーができる。

```
# ./yaim <action> <parameters>
```

action:
-i — –install : インストールする
-c — –configure : すでにインストールされたサービスをコンフィギュアーする
parameters:
-s — –siteinfo : site-info.def ファイルの場所
-m — –metapackage : インストールするメタパッケージの名前
-n — –nodetype : コンフィギュアーするノードタイプ

例:MON のインストール
./yaim -i -s /opt/glite/yaim/etc/site-info.def -m glite-MON
例:MON のコンフィギュアー
./yaim -c -s /opt/glite/yaim/etc/site-info.def -n MON

4.3.5 ファイアウォールの設定

Grid でのノードにはそれぞれの役割があり、その役割を果たすために様々なサービスが実行されている。それらのサービスがマシン間でやりとりを行うことができるよう、ファイアウォール装置および各マシンのファイアウォールを適切に設定する必要がある。必要なポートだけを開けることにより、不正浸入のリスクを抑えることができる。

4.3.6 テスト

ここまで、サイトを作りあげることについて説明してきたが、ここからはその作りあげたサイトが正常に動作するかどうかを確認する作業について述べる。

全ノードに対するテスト

NTP 時間が同期していれば、デバッグ等でログを調べるときにきちんと時間を追っていけるため時刻同期は重要である。また、そもそも時刻がどうきしていなければエラーになってしまうコマンドもある。以下のコマンドにより、時間が同期しているかどうかを確認する。

```
[root@grid01 root]# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
+pxdb.hepl.hiros ntp-b2.nict.go.  2 u  569 1024  377    0.515   23.865   2.867
*ns.hiroshima-u. ntp.hiroshima-u  2 u  515 1024  377    1.237   23.513   0.991
```

プールアカウント ローカルのアカウントは、ジョブなどを流すときに割り当てられるアカウントとして必要なため、その確認を行わなければならない。

CE、SE、WN、VO-BOX にはプールアカウントが作成されるため、/home/ 以下のホームディレクトリが”users.conf”に記述してあるものと同じであること、これらのアカウントが正しいオーナーとグループの設定がされているかを確認する。また/etc/passwd および/etc/group にも”users.conf”に記述してある uid や gid が反映されているかについても確認する。

アカウントマッピング ジョブなどを流すときにユーザーの DN をローカルユーザーにマップするために使われるファイルである”grid-mapfile”に、その情報が記述されているかを確認する。

```
[root@grid01 root]# cat /etc/grid-security/grid-mapfile | grep Narita
"/C=JP/O=KEK/OU=CRC/CN=Takuto Narita" .alice
```

このファイル中のエントリで DN の後ろについている”.alice”というのは、ALICE VO のプールアカウントに動的にマップされているということを示している。

”grid-mapfile”を更新するには‘edg-mkgridmap‘というコマンドが用意されている。

証明書と鍵のチェック　証明書の情報の”Exponent”と”Modulus”の項目が、秘密鍵の”publicExponent”と”Modulus”の項目と合致しているかを確かめる。

```
[root@grid02 root]# openssl x509 -noout -text -in /etc/grid-security/hostcert.pem -modu
[root@grid02 root]# openssl rsa -noout -text -in /etc/grid-security/hostkey.pem -modulu
```

そしてホスト証明書がインストールされた信頼された CA に対して有効かどうか、またホストの秘密鍵の整合性があるかどうかをそれぞれ以下のコマンドにより確かめる。

```
[root@grid02 root]# openssl verify -CApath /etc/grid-security/certificates/ /etc/grid-s
/etc/grid-security/hostcert.pem: OK
[root@grid02 root]# openssl rsa -in /etc/grid-security/hostkey.pem -noout -check
read RSA key
RSA key ok
```

WN のテスト

サービス　必要なサービスが正しいポートで実行されているかを確認する。WN ではリモートログインのための sshd および、ジョブを実行するための pbs_mom がなければならない。

表 4.3: WN で必要なサービスとそのポート

22	sshd	ssh デーモン
15002, 15003	pbs_mom	Local Resource Management System

ssh による CE へのログイン　WN として機能するためには、プールアカウントで ssh を使ってパスワードなしで CE へログインできる必要がある。これはバッチシステムのために必要である。具体的には CE の”/etc/ssh/shosts.equiv”に WN のエントリを加えることによりそれを可能にしている。

```
[root@qx001 ~]# su - alice001
[qx001] /home/alice001 > ssh grid02
Last login: Tue Jan 15 20:06:28 2008 from qx002.hepl.hiroshima-u.ac.jp
[alice001@grid02 alice001]$
```

CE への gridftp のテスト　WN から CE へ GridFTP を用いてファイルを転送できなければならず、これは Workload Management System に必要である。

```
[qx001] /home/alice001 > grid-proxy-init
Your identity: /C=JP/O=KEK/OU=CRC/CN=Takuto Narita
Enter GRID pass phrase for this identity:
Creating proxy ..................................... Done
Your proxy is valid until: Fri Feb  8 03:26:05 2008
[qx001] /home/alice001 > globus-url-copy -vb file:///etc/group gsiftp://grid02/tmp/test.$$
Source: file:///etc/
Dest:   gsiftp://grid02/tmp/
  group  ->  test.10230
           3374 bytes         0.01 MB/sec avg         0.01 MB/sec inst
```

CE のテスト

サービス　必要なサービスが正しいポートで実行されているかを確認する。CE では、ジョブを管理するために gatekeeper、pbs_server、maui スケジューラ、wl-logd、リソースの情報を管理するために GRIS、サイト BDII、slapd、GridFTP によるデータ転送のために ftpd が必要である。

表 4.4: CE で必要なサービスとそのポート

2119	edg-gatekeeper	globus gatekeeper サービス
9002	edg-wl-logd	workload manager logging デーモン
40559, 40560	maui	maui スケジューラ
15004	maui	maui スケジューラ
15001	pbs_server	Local Resource Management System
2135	slapd	GRIS
2170	bdii-fwd	サイト BDII
2171, 2172, 2173	slapd	
2811	ftpd	ftp デーモン

WN の状態を見る　PBS により現在のバッチシステムの WN の状態をチェックする。

```
[root@grid02 root]# pbsnodes -a
qx001.hepl.hiroshima-u.ac.jp
     state = free
     np = 2
     properties = lcgpro
     ntype = cluster
     status = opsys=linux,uname=Linux qx001.hepl.hiroshima-u.ac.jp 2.6.9-55.0.9.EL.cernsmp #1 SMP T
```

例えば qx001 に対しては上のような情報をとってくることができ、state が free であることや status にはそのマシンの情報が表示されていることがわかる。

qsub を使ったジョブサブミッション　ここではまずローカルに Torque に直接ジョブを流すことにより、その機能を確認する。

プールアカウントに変わる。

```
[root@grid02 root]# su - alice001
```

以下のスクリプトをテストジョブ用に使う。

```
[alice001@grid02 alice001]$ cat testbatch.sh
#!/bin/sh
echo This is a test
echo Today is ‘date‘
echo This is ‘hostname‘
echo The current working directory is ‘pwd‘
ls -alF /home
uptime
```

alice queue にジョブを流す。

```
[alice001@grid02 alice001]$ qsub -q alice testbatch.sh
160.grid02.hepl.hiroshima-u.ac.jp
```

そうすると<job id>.<ce name>が表示される。

queue の状態を確認するには以下のコマンドを用いる。

```
[alice001@grid02 alice001]$ qstat -q

server: grid02.hepl.hiroshima-u.ac.jp

Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
alice              --   48:00:00 72:00:00   --    0   0 --   E R
ops                --   48:00:00 72:00:00   --    0   0 --   E R
dteam              --   48:00:00 72:00:00   --    0   0 --   E R
                                               ----- -----
                                                   0     0
```

出力ファイルは testbatch.sh.o<job id>、エラーファイルは testbatch.sh.e<job id> という名前でできる。

これにより CE のみでネットワークを介さない場合にはジョブを流すことができることを確認できた。後に UI からネットワークを介しての動作確認を行う。

MON のテスト

サービス　必要なサービスが正しいポートで実行されているかを確認する。MON では R-GMA のために java、mysqld、資源の情報を扱うために slapd、fmon-server、リモートログインのために sshd が必要である。

表 4.5: MON で必要なサービスとそのポート

8005, 8009, 8080, 8088, 8445	java	R-GMA web サービス
3306	mysqld	R-GMA backend database
2135	slapd	GRIS
2136	slapd	GridICE-MDS - fabric monitoring mds
12409, 12411	edg-fmon-server	GridICE-MDS fabric monitoring mds
22	sshd	ssh デーモン

MON ではデフォルト違うポートをひとつ使っている。grid03 には MON だけでなく DPM もインストールしているため、本来は java で使われるポート 8443 は srmv1 がそのポートを使用している。そのため tomcat の設定によりポート 8443 を 8445 へ変更して使うことにしている。

具体的には”/etc/tomcat5/server.xml”という設定ファイルを書き換えることによりポートの変更が可能となる。

ブラウザ　R-GMA サーバーは、ウェブブラウザからも確認することができる。
https://grid03.hepl.hiroshima-u.ac.jp:8445/R-GMA/
その様子を図 4.4 に示す。

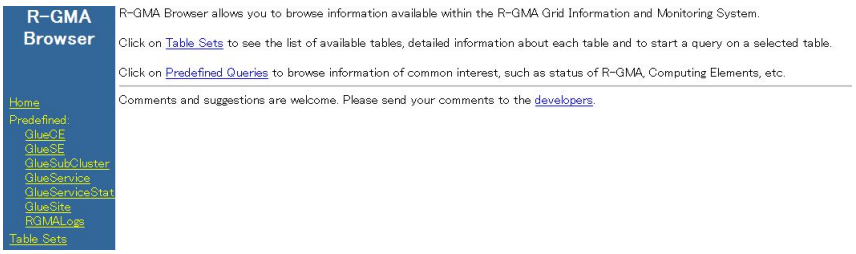


図 4.4: R-GMA サーバーのトップページ。

テストスクリプト　R-GMA サーバーをチェックするためのテストスクリプトが用意されているので、それを用いてテストを行う。

```
[root@grid03 rgma-server]# rgma-server-check
*** Running R-GMA server tests on grid03.hepl.hiroshima-u.ac.jp ***
Checking Tomcat is running on the local machine...
Successfully connected to Tomcat.
Java VM version: 1.4.2_14 (OK)
Connecting to https://lcgic01.gridpp.rl.ac.uk:8443/R-GMA/SchemaServlet...
Successfully connected to Schema.
Using PongServlet (1) on https://lcgic01.gridpp.rl.ac.uk:8443/R-GMA/PongServlet.
Using certificate /var/lib/tomcat5/conf/hostcert.pem.
Using key /var/lib/tomcat5/conf/hostkey.pem.
Checking other servlets...
Connecting to https://grid03.hepl.hiroshima-u.ac.jp:8445/R- GMA/PrimaryProducerServlet:
Checking clock synchronization: OK
Connecting to https://grid03.hepl.hiroshima-u.ac.jp:8445/R- GMA/SecondaryProducerServle
Checking clock synchronization: OK
Connecting to https://grid03.hepl.hiroshima-u.ac.jp:8445/R-GMA/OnDemandProducerServlet:
Checking clock synchronization: OK
Connecting to https://grid03.hepl.hiroshima-u.ac.jp:8445/R- GMA/ConsumerServlet:OK
Connecting to streaming port 8088 on grid03.hepl.hiroshima-u.ac.jp:OK
Checking clock synchronization: OK
*** R-GMA server test successful *** [#jfe11da3]
```

DPM のテスト

サービス　必要なサービスが正しいポートで実行されているかを確認する。DPM ではデータ管理のために DPNS デーモン、DPM デーモン、SRM デーモン、mysqld、資源の情報管理のために slapd、データ転送のために rfiod、ftpd、リモートログインのために sshd が必要である。

表 4.6: DPM で必要なサービスとそのポート

5001	rfiod	
5010	dpnsdaemon	Disk Pool Name Server daemon
5015	dpm	Disk Pool Manager
2811	ftpd	GridFTP server
8443	srmv1	Storage Resource Manager Version 1
8444	srmv2	Storage Resource Manager Version 2
3306	mysqld	MySQL Database Server
22	sshd	ssh デーモン
2135	slapd	

DPNS　DPNS のサービスが使えるかどうかのテストを DPNS サーバー上で行う。

DPNS でディレクトリを作り、それをリストする。

```
[root@grid03 root]# dpns-mkdir /dpm/hepl.hiroshima-u.ac.jp/home/alice/testdir
[root@grid03 root]# dpns-ls -l /dpm/hepl.hiroshima-u.ac.jp/home/alice/
drwxrwxr-x 186 root     root                       0 Feb 01 04:43 generated
drwxrwxr-x   0 root     root                       0 Sep 20 03:23 testdir
```

UI およびそれを使ったテスト

ここからは UI を用いてサイトの機能をテストする。そのため、UI に uid が 504 の”narita”というローカルアカウントを作成する。

サービス 必要なサービスが正しいポートで実行されているかを確認する。UI ではリモートログインのための sshd のみが必要である。

表 4.7: UI で必要なサービスとそのポート

22	sshd	SSH daemon

プロキシをつくる 自分のユーザー証明書および秘密鍵を /.globus/以下に置く。ホスト証明書と同様にセキュリティのため、userkey.pem は root 権限では読み込みのみできるように、usercert.pem は root 権限では読み込みおよび書き込み、その他の全ての人から読み込みのみできるようにしておかなければならない。

```
[grid04] /home/narita > ls -l .globus/ | grep pem
-rw-r--r--  1 narita narita 1551 Dec 12 21:23 usercert.pem
-r--------  1 narita narita 1207 Dec 12 21:23 userkey.pem
```

続いて、プロキシを作りその内容を確認する。

```
[grid04] /home/narita > voms-proxy-init -voms alice
Cannot find file or dir: /home/narita/.glite/vomses
Enter GRID pass phrase:
Your identity: /C=JP/O=KEK/OU=CRC/CN=Takuto Narita
Creating temporary proxy ........................................... Done
Contacting  voms.cern.ch:15000 [/DC=ch/DC=cern/OU=computers/CN=voms.cern.ch] "alice" Done
Creating proxy ..................................................... Done
Your proxy is valid until Fri Feb  8 05:17:17 2008
[grid04] /home/narita > voms-proxy-info -all
subject   : /C=JP/O=KEK/OU=CRC/CN=Takuto Narita/CN=proxy
issuer    : /C=JP/O=KEK/OU=CRC/CN=Takuto Narita
identity  : /C=JP/O=KEK/OU=CRC/CN=Takuto Narita
type      : proxy
strength  : 512 bits
path      : /tmp/x509up_u504
timeleft  : 11:59:43
=== VO alice extension information ===
VO        : alice
```

```
subject   : /C=JP/O=KEK/OU=CRC/CN=Takuto Narita
issuer    : /DC=ch/DC=cern/OU=computers/CN=voms.cern.ch
attribute : /alice/Role=NULL/Capability=NULL
attribute : /alice/lcg1/Role=NULL/Capability=NULL
```

この情報からわかるように、通常は 12 時間有効なプロキシがローカルマシンの/tmp/以下に x509up_u<uid>というファイル名で作成されている。以降のテストはこのプロキシを用いて行う。

CE　まずはネットワークを介して直接 CE へジョブを流す。これはフォアグラウンドに実行され、結果が stdout で表示される。

```
[grid04] /home/narita > globus-job-run grid02 /bin/hostname
grid02.hepl.hiroshima-u.ac.jp
```

このように特にジョブマネージャーを指定しなかった場合は fork が使われる。

ちなみに、CE 上で利用できるジョブマネージャーの名前は/opt/globus/etc/grid-services/以下を見ればよい。

```
[root@grid02 root]# ls /opt/globus/etc/grid-services/
jobmanager  jobmanager-fork  jobmanager-lcgpbs
```

続いて、バッチシステムにジョブを流す。これはバックグラウンドで処理される。

```
[grid04] /home/narita > globus-job-submit grid02 -queue alice /bin/hostname
https://grid02.hepl.hiroshima-u.ac.jp:20001/15203/1202374786/
```

このコマンドにより GLOBUS_ID が返ってくる。この ID を用いてジョブの状態をチェックする。

```
[grid04] /home/narita > globus-job-status https://grid02.hepl.hiroshima-u.ac.jp:20001/1
DONE
```

完了しているので、出力ファイルを回収する。

```
[grid04] /home/narita > globus-job-get-output https://grid02.hepl.hiroshima-u.ac.jp:200
grid02.hepl.hiroshima-u.ac.jp
```

4.4　Grid を使う

ここでは、実際の Grid の使用方法について記す。

4.4.1　情報の検索

Grid の情報を検索するために、‘lcg-infosites‘と ‘lcg-info‘を使う。‘lcg-infosites‘はシンプルで簡単に使うことができる一方、‘lcg-info‘を使うことによってより複雑な問い合わせをすることができる。

lcg-infosites

まず CE について問い合わせてみる。

```
[grid04] /home/narita > lcg-infosites --vo alice ce
valor del bdii: lcg00126.grid.sinica.edu.tw:2170
#CPU Free Total Jobs Running Waiting ComputingElement
----------------------------------------------------------
  54    1    0           0    0 obsauvergridce01.univ-bpclermont.fr:2119/jobmanager-lcgpbs-alice
  40    1    1           0    1 iut15auvergridce01.univ-bpclermont.fr:2119/jobmanager-lcgpbs-alice
  20    1    1           0    1 iut43auvergridce01.univ-bpclermont.fr:2119/jobmanager-lcgpbs-alice
```

同様に SE についても問い合わせることができる。

```
[grid04] /home/narita > lcg-infosites --vo alice se
Avail Space(Kb) Used Space(Kb)  Type SEs
----------------------------------------------------------
4378660000      301320000       n.a  se1.egee.man.poznan.pl
98161828        518573252       n.a  clrauvergridse01.in2p3.fr
888143872       84934656        n.a  se001.ipp.acad.bg
```

また、CE とそれに一番近い SE を表示させることもできる。

```
[grid04] /home/narita > lcg-infosites --vo alice closeSE
valor del bdii: lcg00126.grid.sinica.edu.tw:2170
Name of the CE: obsauvergridce01.univ-bpclermont.fr:2119/jobmanager-lcgpbs-alice
clrauvergridse01.in2p3.fr

Name of the CE: iut15auvergridce01.univ-bpclermont.fr:2119/jobmanager-lcgpbs-alice
clrauvergridse01.in2p3.fr

Name of the CE: iut43auvergridce01.univ-bpclermont.fr:2119/jobmanager-lcgpbs-alice
clrauvergridse01.in2p3.fr
```

問い合わせに使う BDII は$LCG_GFAL_INFOSYS によって指定されているので、その環境変数を変えるか、–is オプションを使って指定することによって BDII を変えることができる。

lcg-info

サポートされている属性のリストを表示させるには、以下のようなコマンドを打てばよい。

```
[grid04] /home/narita > lcg-info --list-attrs
Newline in left-justified string for printf at /opt/lcg/bin/lcg-info line 308, <DATA> line 225.
Attribute name      Glue object class     Glue attribute name

WorstRespTime       GlueCE                GlueCEStateWorstResponseTime
CEAppDir            GlueCE                GlueCEInfoApplicationDir
TotalCPUs           GlueCE                GlueCEInfoTotalCPUs
```

また、例えば ALICE の CE で Memory が 1025 で、Memory と CINT2000 の値を表示させたいときは、以下のようにする。

```
[grid04] /home/narita > lcg-info --list-ce --vo alice --query 'Memory=1025' -attrs 'Mem
- CE: ce002.ipp.acad.bg:2119/jobmanager-lcgpbs-alice
- Memory              1025
- CINT2000            1000

- CE: grid109.kfki.hu:2119/jobmanager-lcgpbs-alice
- Memory              1025
- CINT2000            700

- CE: hep-ce.cx1.hpc.ic.ac.uk:2119/jobmanager-pbs-heplt2
- Memory              1025
- CINT2000            1687
```

4.4.2 ジョブを流す

JDL ファイル

まずジョブを流すための JDL ファイルを作る。

```
[grid04] /home/narita > cat testJob.jdl
Executable= "testJob.sh";
StdOutput  = "testJob.out";
StdError  = "testJob.err";
InputSandBox  = {"./testJob.sh"};
OutputSandBox = {"testJob.out","testJob.err"};
VirtualOrganisation = "alice";
```

ここでは、実行するコマンド、アウトプットファイル、エラーファイル、InputSandBox、OutputSandBox、VO を指定する。

実行するコマンドとして指定している”testJob.sh”も作成する。

```
[grid04] /home/narita > cat testJob.sh
#!/bin/bash
date
hostname
echo "****************************************"
echo "env | sort"
echo "****************************************"
env | sort
echo "****************************************"
echo "mount"
echo "****************************************"
mount
echo "****************************************"
echo "rpm -q -a | sort"
echo "****************************************"
/bin/rpm -q -a  | sort

sleep 20
date
```

マッチメイキング

ちなみにこの JDL ファイルを使ってジョブが走ることのできるサイトを調べることができる。

```
[grid04] /home/narita > edg-job-list-match testJob.jdl

Selected Virtual Organisation name (from JDL): alice
Connecting to host rb129.cern.ch, port 7772

**********************************************************************
                        COMPUTING ELEMENT IDs LIST
 The following CE(s) matching your job requirements have been found:

                    *CEId*
 a01-004-128.gridka.de:2119/jobmanager-pbspro-aliceL
 a01-004-128.gridka.de:2119/jobmanager-pbspro-aliceS
 a01-004-128.gridka.de:2119/jobmanager-pbspro-aliceXL
 a01-004-128.gridka.de:2119/jobmanager-pbspro-aliceXS
 alice003.nipne.ro:2119/jobmanager-lcgpbs-alice
```

コンフィギュレーションファイル

自分のコンフィギュレーションファイルを使うこともできる。特に指定しなかった場合は UI におけるデフォルトの設定 (/opt/edg/etc/edg_wl_ui_cmd_var.conf) が使われることになる。

```
[grid04] /home/narita > cat my-defaults.conf
[
rank = - other.GlueCEStateEstimatedResponseTime;
requirements = other.GlueCEStateStatus == "Production";
RetryCount = 1;
ErrorStorage = "$HOME/jobError/";
OutputStorage = "$HOME/jobOutput/";
ListenerPort = 44000;
ListenerStorage = "/tmp";
LoggingTimeout = 30;
LoggingSyncTimeout = 30;
NSLoggerLevel = 0;
DefaultLogInfoLevel = 0;
DefaultStatusLevel = 0;
DefaultVo = "unspecified";
]
```

ジョブのサブミット

いよいよジョブをサブミットする。

```
[grid04] /home/narita > edg-job-submit -c my-defaults.conf testJob.jdl

Selected Virtual Organisation name (from JDL): alice
```

```
Connecting to host rb129.cern.ch, port 7772
Logging to host rb129.cern.ch, port 9002


****************************************************************************************
                              JOB SUBMIT OUTCOME
 The job has been successfully submitted to the Network Server.
 Use edg-job-status command to check job current status. Your job identifier (edg_jobId

 - https://rb129.cern.ch:9000/phPGGZKK14fcoC38KtgNAQ


****************************************************************************************
```

ジョブは無事にサブミットすることができ、実際にジョブを受け取った RB は rb129.cern.ch である。そして JOBID が表示されたので、この JOBID を使ってジョブの状態を見たり結果を回収したりする。

ジョブの状態

ジョブの状態を確認する。

```
[grid04] /home/narita > edg-job-status https://rb129.cern.ch:9000/phPGGZKK14fcoC38KtgNA


*************************************************************
BOOKKEEPING INFORMATION:

Status info for the Job : https://rb129.cern.ch:9000/phPGGZKK14fcoC38KtgNAQ
Current Status:     Done (Success)
Exit code:          0
Status Reason:      Job terminated successfully
Destination:        ce122.cern.ch:2119/jobmanager-lcglsf-grid_alice
reached on:         Thu Feb  7 12:09:26 2008
*************************************************************
```

完了しているようである。

ジョブの出力の回収

ジョブが完了したら、ジョブの出力を回収しなければならない。

```
[grid04] /home/narita > edg-job-get-output -c my-defaults.conf https://rb129.cern.ch:90

Retrieving files from host: rb129.cern.ch ( for https://rb129.cern.ch:9000/phPGGZKK14fc

********************************************************************************
JOB GET OUTPUT OUTCOME

 Output sandbox files for the job:
 - https://rb129.cern.ch:9000/phPGGZKK14fcoC38KtgNAQ
```

```
have been successfully retrieved and stored in the directory:
/home/narita/jobOutput//narita_phPGGZKK14fcoC38KtgNAQ

**********************************************************************************
```

また、何かジョブに問題がありそうなときには、‘edg-job-get-logging-info -v 2 <JOBID>‘によって詳細な情報を見ることもできる。

4.5 研究室内解析環境の構築

クォーク物理学研究室では ALICE 実験 PHOS 検出器の研究開発も行っており、その解析を行うためにも Grid へマシンを提供するだけでなく、研究室内のみで使用できる解析環境を構築することはとても有益である。以下ではその設定について述べる。

4.5.1 各マシンの設定

解析を行うために必要な環境を実現するために、バッチシステム及び NFS により共有ができるソフトウェアエリアを導入した。また、利便性のため NIS を用いてアカウントを統一した。さらにそのバッチシステムによる CPU 使用率等をモニターできるシステムも導入した。

OS のインストール

qs01、qx017〜qx031 へ SLC4.5 をインストールした。これは、ALICE Offline Code を使用していくための制約も含まれる。

NTP

NTP を使って時間の同期をする。JST(Japan Standard Time) に合わせている。広島大学の NTP サーバー及びクォーク研の NTP サーバー (pxdb) に同期している。/etc/ntp.conf を編集し、サービスを再起動することで、ある程度の時間待てば同期する。

NIS

NIS[5] を使うことにより、各ユーザーが同じアカウントを用いて全てのマシンを使うことができるようにした。hep02 をサーバーとし、qs01、qx017〜qx031、qx034〜qx064 をクライアントとした。

[5]Network Information Server

NFS

バッチジョブシステムにおいてソフトウェアを共有できるように、qs01 をサーバー、qx017〜qx031 をクライアントとして、qs01 の/opt 以下を qx017〜qx031 がマウントするようにした。

また、各個人の作業ディレクトリである hep02 の/home を qx017〜qx031 がマウントするようにもした。

サーバー側　portmap と nfs のサービスを起動する。

```
# service portmap start
# service nfs start
```

/etc/exports を編集して qx017〜qx031 に対してエクスポートする。

```
# exportfs -ra
```

エクスポートができているか確認する。

```
# exportfs
```

クライアント側　portmap のサービスを起動する。

```
# service portmap start
```

/etc/fstab を編集してマウントする。

```
# mount -a
```

マウントができているか確認する。

```
# df -h
```

Condor

バッチジョブシステムには Condor を用いた。Condor を使うことにより、複数の資源を効率良く、かつ公平に使うことができる。

Condor のインストールには、”condor_config”というファイルに様々な設定を書き込むことによって行われる。そのため、一度そのファイルを作成しておけば、新たにインストールする際にも簡単に導入ができる。また、マシンによる差分を”condor_config.local”というファイルに書き込むことができる。以下にインストールの手順を記載する。

Condor のウェブページから RPM をダウンロードし、インストールする。ウィザードに従って、NFS で共有している/opt 以下にインストールする。

```
# rpm -ivh ....
```

利便性のため、Condor のディレクトリへ condor という名前でシンボリックリンクをはる。

```
# ln -s condor-6.8.5 condor
```

condor を実行するユーザーとして condor ユーザーを追加する。これはサーバーとクライアントの全てのマシンで行う。

```
# useradd -u 91 -d /home/condor condor
```

ローカルディレクトリを作成する。これは、サーバーとクライアントの全てのマシンで行う。

```
# mkdir -p /usr/local/condor/local
```

condor_config.local を/usr/local/condor/local 以下に置く。サーバー側、クライアント側それぞれにファイルを置く。環境変数を設定する。これはサーバーとクライアントの全てのマシンで行う。

```
# export CONDOR_CONFIG=/opt/condor/etc/condor_config
# export PATH=${PATH}:/opt/condor/bin
```

/opt/condor/etc/condor_config を編集する。このファイルの内容を変えることで様々な設定ができる。

コンフィギュアーをする。ここで、インストールディレクトリや condor アカウントができることを指定する。

サーバー側では、サブミット、実行、管理ができるようにする。

```
# /opt/condor/condor\_configure --install-dir=/opt/condor --owner=condor --type=submit,execute,mar
```

クライアント側では、サブミット、実行ができるようにする。

```
\# /opt/condor/condor\_configure --install-dir=/opt/condor --owner=condor --type=submit,execute\\
```

init script(condor) を/etc/init.d/以下に置く。これにより condor の起動、停止が簡単になる。Condor のをスタートする。これもサーバーとクライアントの全てのマシンで行う。

```
# /etc/init.d/condor start
```

ALICE Offline Code のインストール

バッチジョブシステムのサーバーとなるマシンである qs01 へ、ALICE Offline Code もインストールした。そしてそのインストールディレクトリを NFS マウントすることにより、どのマシンからでもコードが使えるようにした。

モニタリングシステムのインストール

Condor ジョブなどによる qx の CPU 使用率などをモニターするために、qs01 をサーバーとして Munin を導入した。
http://qs01.hepl.hiroshima-u.ac.jp/munin/condor.html
図 4.5 は Munin による qx022 のモニターの図で、左図が一日の CPU 使用率、右図が一週間の CPU 使用率を表している。赤色が主に Condor による利用分で、青色が空き時間、桃色が I/O 待ち時間である。

qx022

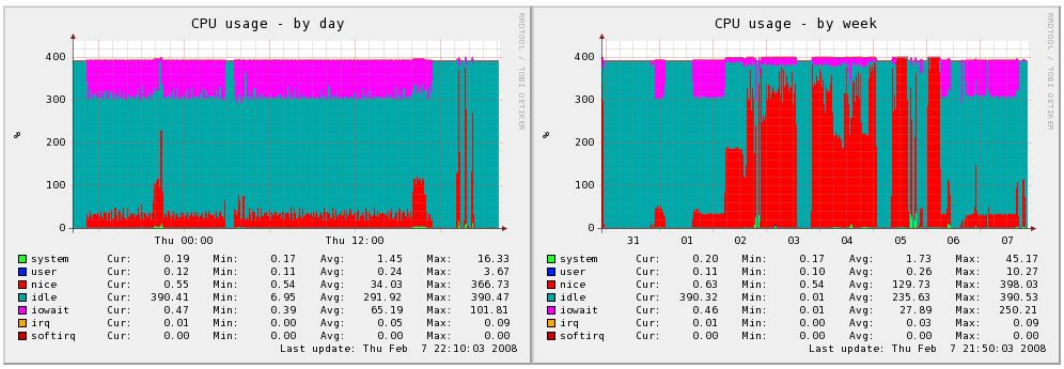


図 4.5: Munin によるモニター。左図が一日の CPU 使用率、右図が一週間の CPU 使用率を表している。赤色が主に Condor による利用分で、青色が空き時間、桃色が I/O 待ち時間である。

4.6 マシンおよびネットワーク機器の性能

4.6.1 マシンの性能

主な仕様

PowerEdge1950 grid01〜grid04、qx001〜qx032、dns01〜dns02。

- CPU デュアルコアインテル $Xeon5160 \times 2$(クロック周波数 3.0GHz)
- メモリ 4GB Fully Buffered DDR2 SDRAM メモリ (最大 32GB)
- ハードディスク 300GB SAS ハードディスク (10,000 回転)
- ネットワークオンボード デュアル ギガビット・イーサネット・コントローラ

PowerEdge2950　gw、hep01〜hep02、nfs01〜nfs03。
CPU のみマシンにより違いがあり、gw、hep01〜hep02 は Xeon5150、nfs01〜nfs03 は Xeon5110 である。

- CPU
  - gw、hep01〜hep02 デュアルコアインテル $Xeon5150 \times 2$(クロック周波数 2.66GHz)
  - nfs01〜nfs03 デュアルコアインテル $Xeon5110 \times 2$(クロック周波数 1.6GHz)
- メモリ 4GB Fully Buffered DDR2 SDRAM メモリ (最大 32GB)
- ハードディスク 300GB SAS ハードディスク (10,000 回転)×5(RAID5+HS)
- ネットワークオンボード デュアル ギガビット・イーサネット・コントローラ

4.6.2　ネットワーク機器の性能

主な仕様

Catalyst 6503E　このスイッチは基幹ネットワークスイッチのひとつで、グローバルアドレスが必要なマシンおよび、プライベートアドレスをもつマシンが繋がっているスイッチ (Catalyst 4948, Catalyst 3560G-24TS) の上流に位置し、ネットワークを供給している。また、nfs01〜03 に対して NAT[6] を、それ以外のプライベートアドレスをもつマシンに対して IP マスカレードを行っている。
　主な仕様は以下の通りである。

- 32Gbps のスイッチング機能を有す
- SSH によるセキュアな接続を実装
- NAT 機能を有す
- 二重化電源を備える

Catalyst 4948　このスイッチには、プライベートアドレスをもったマシンが下に繋がっている基幹ネットワークスイッチのひとつである。具体的には grid01〜04, qs01, qx001〜032, nfs01〜03 などのマシンがぶら下がっている。
　主な仕様は以下の通りである。

- 1000Base-T インターフェースポートを 48 ポート有す

[6]Network Address Translation

- バックプレーンは 96Gbps のスイッチング機能を有す
- ジャンボフレームに対応
- 72Mpps のパケット転送速度を有す

Catalyst 3560G-24TS　このスイッチは各居室の個人端末にネットワークを供給している端末ネットワークスイッチである。
主な仕様は以下の通りである。

- 32Gbps のスイッチング機能を有す
- 全二重の 100Mbps/1Gbps の自動切替ポートを 24 ポート有す

NetScreen ISG 2000　これはファイアウォール装置であり、ファイアウォール装置は、外部からの侵入を防ぐためになくてはならないものである。主な仕様は以下の通りである。

- ファイアウォールパフォーマンスは 2Gbps の処理能力を有す
- 3DES 時のファイアウォールパフォーマンスは 1Gbps の処理能力を有す
- 同時 512,000 セッションの処理能力を有す
- 同時 VPN トンネル数は 10,000 の処理能力を有す
- 168 ビット 3DES の暗号化方式を実装
- 二重化電源を備える

4.7　研究室外部のネットワーク構成

4.7.1　SINET3

SINET3[7] とは、日本全国の大学、研究機関等の学術情報基盤として、NII[8] が構築、運用している情報ネットワークである。SINET3 は、平成 19 年 4 月から従来の学術情報ネットワーク基盤であった SINET とスーパー SINET との基盤を統合して運用が開始された。SINET3 では、75 箇所の接続拠点があり、データセンター内に IP ルータを設置した中継ノード (12 箇所)、加入機関回線等を収容する一般ノード (62 箇所)、商用接続およびアジア向け接続としての相互接続拠点 (1 箇所) で構成されている。

[7]学術情報ネットワーク (Science Information Network)
[8]国立情報学研究所

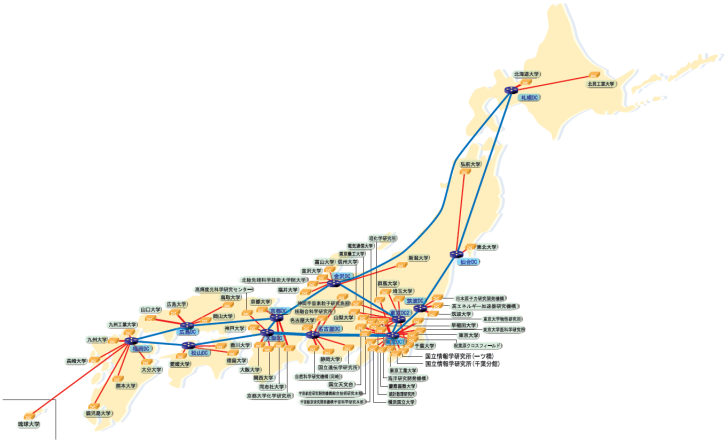


図 4.6: SINET3 の構成

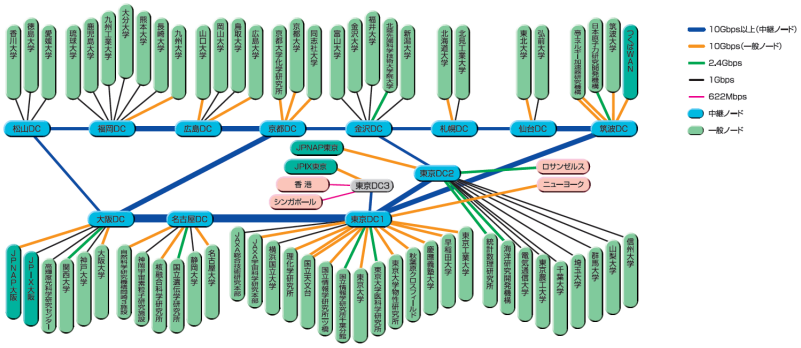


図 4.7: SINET3 の構成の詳細

4.8 バンド幅

4.8.1 Iperf

Iperf はネットワーク性能測定ツールであり、TCP プロトコル転送時における帯域幅を測定することができる。

ウェブページよりダウンロードし、使用できる。

http://dast.nlanr.net/projects/Iperf/

インストール方法

公式ページよりダウンロード

```
# wget http://dast.nlanr.net/Projects/Iperf2.0/iperf-2.0.2.tar.gz
```

解凍および展開

```
# tar xvfz iperf-2.0.2.tar.gz
```

ディレクトリの移動

```
# cd iperf-2.0.2
```

コンフィギュアーおよびコンパイル

```
# ./configure; make
```

4.8.2 ウィンドウサイズ

TCP[9] は OSI 参照モデルのトランスポート層に属しているプロトコルである。TCP ではデータを送信すると、受信側はそれに対して確認応答を返す。その繰り返しによりデータを送っている。その際に受信側からの確認応答を待たずに TCP セグメントを続けて送れるようにするためにウィンドウ制御という方法が用意されており、その大きさのことをウィンドウサイズと呼ぶ。

ウィンドウサイズの拡大

デフォルトのウィンドウサイズでは、ギガビット・イーサネットの通信に対しては小さすぎるため、ウィンドウサイズを拡大する。具体的には、/proc/sys/net/core/以下の rmem_max、wmem_max の値を 80M バイト、rmem_default、wmem_default の値を 8M バイトとし、また/proc/sys/net/ipv4/以下の tcp_rmem、tcp_wmem に対しても最大を 80M バイト、デフォルトを 8M バイトとした。

本来ならば、バンド幅や RTT[10] などに応じた最適値を求めることにより良い性能を発揮することができるが、今回は最適化は行っていない。

4.8.3 研究室内部のバンド幅測定

広島での Grid に使うマシンの間でネットワークが物理的につながり、それらがある程度のバンド幅をもっていることを確認するために Iperf を用いて TCP でのバンド幅測定を行った。また研究室内での解析のためのクラスターについても同様にバンド幅を測定した。具体的には、grid01-grid02、grid02-grid03、grid01-grid03、grid01-qx001〜qx016、grid02-qx001〜qx016、grid03-qx001〜qx016、qs01-qx017〜qx030 の間のバンド幅を、各々60 秒間測定した。

[9] Transmission Control Protocol
[10] 往復遅延時間 (Round Trip Time)

4.8.4 研究室外部とのバンド幅測定

研究室外部ではどのくらいのバンド幅なのかを調べるため今回は SINET3 の一般ノードとして接続している、高エネルギー加速器機構 (KEK) とのバンド幅測定を行った。

また、今後接続予定の Tier 1 センターであるリヨンとの速度測定も行う予定である。

4.9 計算力

4.9.1 ローカルクラスターを用いた計算力測定

上記のベンチマーク値は、例えばシミュレーションデータを扱った場合はどの程度の力に対応するのか、調べた。

イベントを生成し、物理結果を得るまでのシミュレーションは以下の 3 つのステップに分けることができる。

- Simulation 特定の物理事象に対して粒子を発生させる。そのコードにはいくつか代表的なものが存在し、PYTHIA、HIJING などがある。PYTHIA では電子、陽電子、陽子衝突を、HIJING では重イオン衝突をシュミレートできる。そして Event Generation により発生した粒子が検出器を通る際に、検出器内でどのような反応をするか、そこで損失したエネルギーがどのように検出器を伝わり読み出し電子回路によりデジタル情報に変換されるか、をシミュレートする。

- Reconstruction 読み出し電子回路からのデジタル情報から粒子情報を引き出す。この部分は実データの解析においても行われるステップである。

- Analysis 粒子情報を用いて、物理結果となるヒストグラムなどを求める。この部分も実データの解析の場合と同じである。

今回は Simulation と Reconstruction を行った場合について調べた。

衝突係数 0-1fm、全検出器で、マシンを Condor で 14 台 (56core) 使い計算を行うと、1core あたりに換算すると Simulation にかかる時間は約 115 分、Reconstruction にかかる時間は約 9 分となった。つまり 1 台あたりに換算すると、Simulation には約 29 分、Reconstruction には約 2 分程度かかることになる。

この値を使い、研究室内解析環境において例えば一ヶ月シミュレーションを走らすと、

$$30\text{日} \times 24\text{時間} \times 60\text{分} \div 31\text{分}/event \times 15\text{台} = 20903event$$

となり、約 2.1×10^4 イベントを作ることができる。これは、ALICE で一年間に収集できる鉛+鉛衝突のデータの約 0.02%である。

第5章　結果と考察

5.1　サイトの構築

前章で述べた構築およびその動作確認により、広島大学において VO-BOX、CE、SE、MON、WN、UI のセットアップをすることでサイトの構築が完了した。それにより、Grid メンバーの認証、CE のバッチシステムによる WN でのジョブの実行、SE によるディスクの管理、UI を用いた情報検索およびジョブのサブミットなどができるようになった。そして APROC からサイト証明を取得中であるため、それが認められれば正式に LCG Tier 2 センターとなることができる。

5.2　バンド幅

5.2.1　研究室内部のバンド幅測定

図 5.1 は grid01 から grid02 へデータを 60 秒間送ったときの結果である。縦軸がバンド幅 (Mbps)、横軸が時間 (sec) であり、赤線が送信側 (grid01)、黒線が受信側 (grid02) のバンド幅を表している。受信側は平均約 940Mbps のバンド幅で受信ができている。

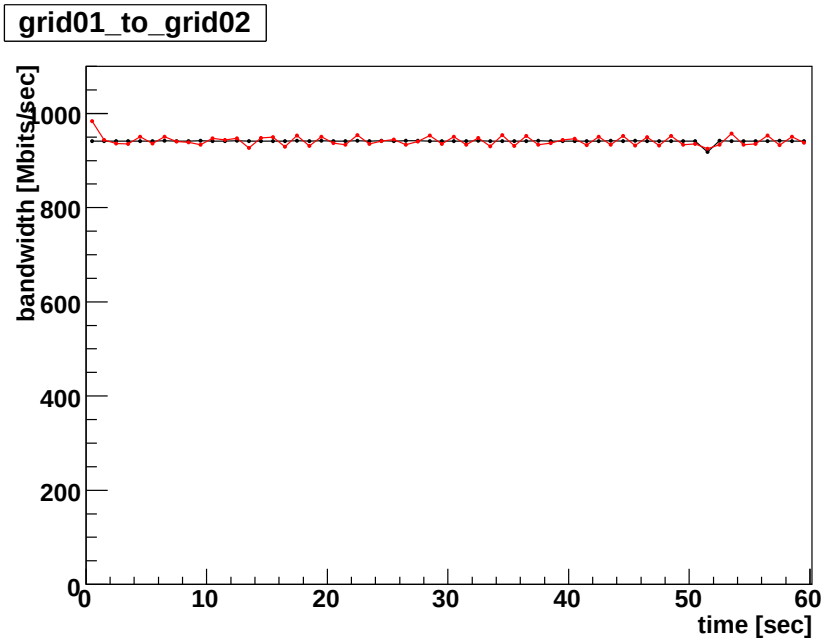


図 5.1: grid01 から grid02 への転送レート

回線の太さが 1Gbps であり、これらのマシンの間をつなぐネットワークスイッチも性能上は 1Gbps のバンド幅を出すことを保障されているため、理想的には 1Gbps のバンド幅となるはずである。この測定ではなんらかの原因によりバンド幅は約 940Mbps となっている。今回の測定では各々のマシン間でネットワークが確立されており約 940Mbps という十分な転送レートをもっているということがわかった。

また、例えば図 5.2 では送信側には多少の揺らぎが見られるが、受信側では安定して 940Mbps のバンド幅をもっているため特に問題無い。

図 5.3 などでは転送レートが落ちている点がある。そのため長時間測定を行ったところ、平均して 500 秒通信を行えば 1 秒程度はこのように転送レートが落ちるときがあるようだが、原因は不明である。しかしその程度の頻度で起こることであれば通信速度にはほとんど影響を与えないため、この点においては問題はない。

結果としては、研究室内部は現在のバンド幅で十分であると言うことができる。全てのグラフは付録 A に載せた。

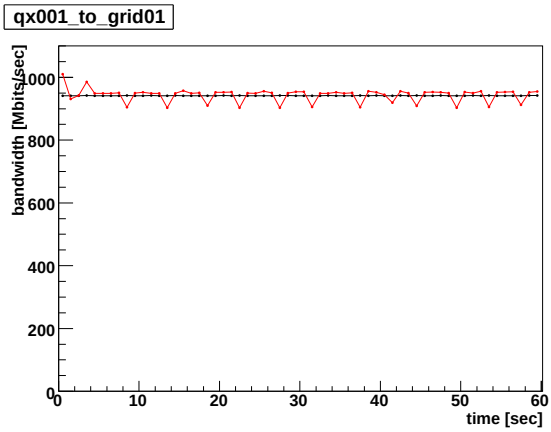


図 5.2: qx001 から grid01 への転送レート

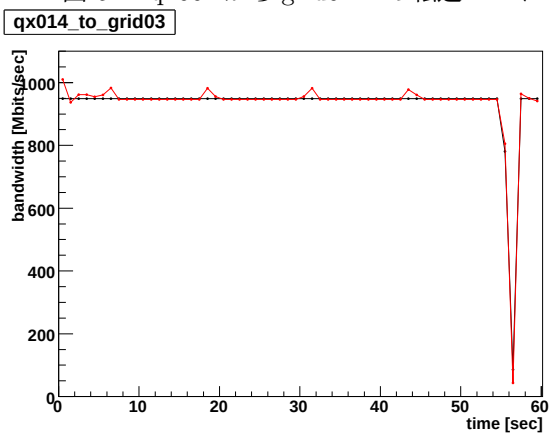


図 5.3: qx014 から grid03 への転送レート

5.2.2 研究室外部とのバンド幅測定

KEK-広島大学間のバンド幅測定では、安定していれば 900Mbps 程度の速度が出ている場合もあるが、不安定なときはパケットロスを繰り返すことがわかった。しかし結果としては、パケットロスをしながらも KEK から広島大学へのバンド幅は約 240Mbps、その逆方向の広島大学から KEK へは約 750Mbps のバンド幅が得られた。Tier 2 センターとして求められているバンド幅は、入ってくる方向が 10Mbps、出ていく方向が 600Mbps であるため、その要求は満たしていることが確認できた。

ただし、ネットワークがより不安定な状況であれば速度は落ちる可能性がある。KEK から広島大学へおいての特にパケットロスが多い場合が図 5.4 である。

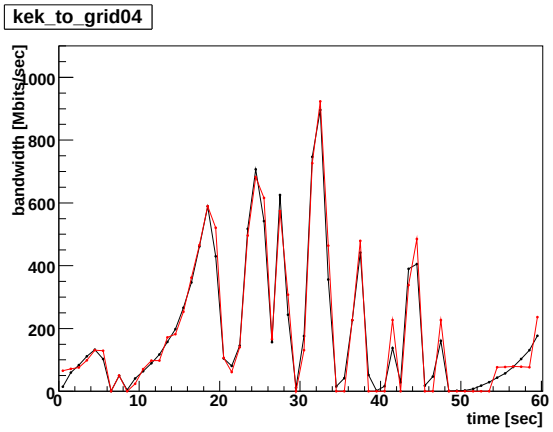


図 5.4: KEK のマシンから grid03 への転送レート

この図を見るとわかるように、パケットロスを繰り返している。これは途中のルーターにおいてデータを保持仕切れなくなりロスしているために引き起こされている。それをなるべく抑えるために QoS を使うという方法がある。QoS とは指定した速度でパケットを送出する制御のことである。QoS を使って 300Mbps、400Mbps、600Mbps 以上速度が出ないようにしたときの結果がそれぞれ、図 5.5、図 5.6、図 5.7 である。縦軸が転送速度 (Byte/sec)、横軸が時間 (sec) である。

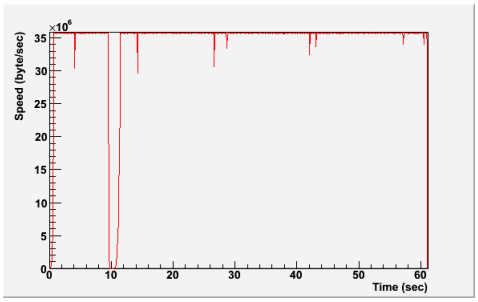


図 5.5: QoS で 300Mbps 以上出ないように速度制御したときの KEK-広島間の転送レート

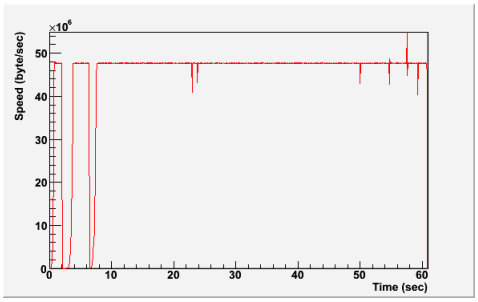


図 5.6: QoS で 400Mbps 以上出ないように速度制御したときの KEK-広島間の転送レート

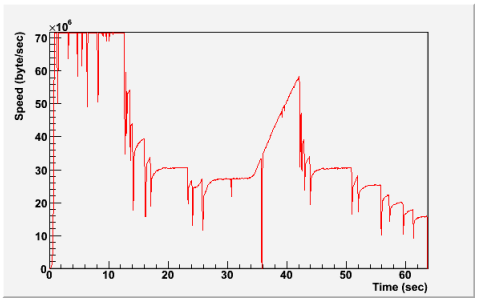


図 5.7: QoS で 600Mbps 以上出ないように速度制御したときの KEK-広島間の転送レート

通常、イーサネットではパケットを数 k〜数 M バイト分まとめて出し、その後しばらく休むということを繰り返して送信を行っている。例えば 1Gbps だと 1500 バイトのパケットを送るのに 12 マイクロ秒程度しかかからず、このマイクロ秒のオーダーでは 1Gbps はいつも出ていることになる。そうなると、共用回線で途中に混雑区間がありルータなどのバッファで保持しきれなくなることにより、パケットロスを引き起こしやすくなる。そうならないためにはパケット間にギャップを入れるのが有効であり、それにより速度を制限することができる。

今回の測定結果では、600Mbps 以下に速度を制限した図 5.7 を見るとわかるように、TCP が減速フェーズにあってもさらにパケットロスすることがあり、一度そうなると回復が非常に遅いようである。また、転送速度を下げただけではパケットロスを完全に消すことはできないようだが、転送速度が高いとパケットロスする確率が高くなるということがいえる。今回の場合では、400Mbps 程度まで減速した方が良いようである。

Hep-net において広島大学とほぼ同等の環境である神戸大学では約 900Mbps のバンド幅であるため、その原因を特定することができれば広島大学におい

ても 900Mbps の速度を達成することは十分考えられる。

また、CERN とのバンド幅はまだ測定していないため、それは今後測定していく必要がある。そういったときにより速度を上げようとするならば、パケットロスの原因をつきとめること、今回用いた QoS を導入すること、また別のアプローチとして GridFTP などのプロトコルを用いて測定を行うことなどが方法としては考えられる。

5.3 計算力

計算力を数値で表す際に、ALICE 実験では SI2k という単位を用いている。これは広く一般に用いられている、SPEC[1] というベンチマークの標準化促進を目指す非営利団体が規定したベンチマークテストによる、プロセッサの整数演算処理性能を示す値の 2000 年版である。広島のサイトの WN および研究室内解析環境のデータ処理マシンである DELL Power Edge 1950 に対してもこのベンチマークの結果があり、それを用いると 1 台あたり約 3.1kSI2k となる。この値を単純に WN の台数分掛けることにより、広島サイトの計算力を求めると、$3.1kSI2k \times 16$ 台 = 約 $50kSI2k$ となる。また、同様に研究室内解析環境では、$3.1kSI2k \times 15$ 台 = 約 $47kSI2k$ となる。

4.1 で述べたように、Tier 2 全体として要求されている値は 14.4MSI2k であり、広島サイトはその約 0.35%の計算力をもつことになる。要求されている値がどれだけ大きいものであるかがわかると同時に、Grid のように資源を出し合わなければ達成できないということがわかる。

[1] The Standard Performance Evaluation Corporation

第6章　結論と今後の展望

本論文ではLCG Tier 2センターとしてのGridサイトを構築するためのマシンおよびネットワークのセットアップを行い、APROCからのサイト証明を取得中である。その計算力は約50KSI2kである。今後サイトとして動きはじめれば、資源の増強やノードの追加を行ったり、サイトの運用に関するノウハウを蓄積していくことで、よりALICE実験に貢献することができるだろう。

また、本論文ではユーザーとしてGridを扱い、情報検索およびジョブの投入を行うことができた。サイトの認証が完了し、これらのことも広島サイトのマシンを使って行うことができれば、Gridサイトとして働いていることの確認ともなる。

バンド幅については研究室内およびKEKとの間で測定を行い、研究室内では十分、KEKとの間でも十分であるという結果を得た。そしてより速度を上げることが可能であり、そのための方法はいくつかあるという結論となった。また、今後の課題としてはCERNとのバンド幅も測定するということが挙げられる。

また、研究室内のみで使用する解析環境も整えることができた。

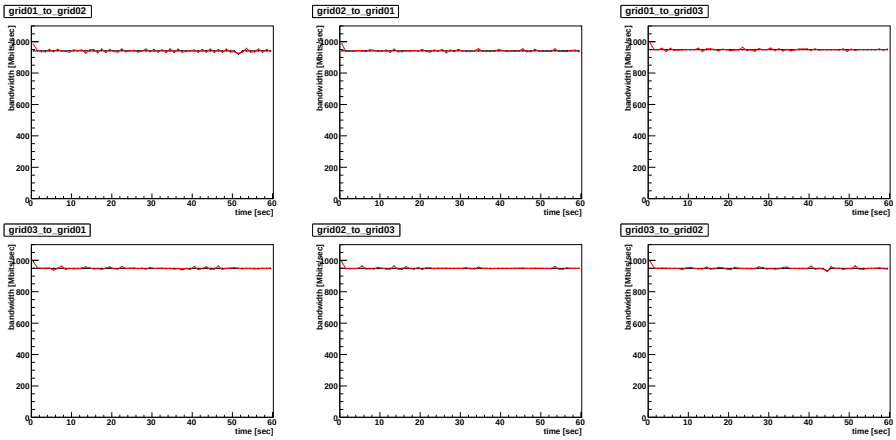
謝辞

本論文を書き上げるために多くの方々から御支援をいただきました。論文を書く上での重要点や構成におけるアドバイスなどをいたいた、指導教官である杉立先生に深く感謝いたします。この論文のテーマを決めるところから、論文の構成、方針など多岐にわたり指導的立場で助けてくださった洞口先生に深く感謝します。研究室のミーティングや、その他の場所で多くの助言をいただいた、志垣先生、本間先生に深く感謝します。質問に対して快く答えてくださり、多くのアドバイスもくださった鳥井さん、槌本さんに深く感謝します。研究室外のバンド幅測定のお願いに対して、快く協力していただき、さらに多くのアドバイスをくださった高エネルギー加速器研究機構の鈴木様 (山形様) に深く感謝します。また、研究室の同期、先輩、後輩のお陰で楽しい研究生活を過ごすことができ、感謝しております。特に同期入学である久保さんには、怪我をしたときや雪の日などにも助けていただいたり、来島くんとともに、同じ修士論文を書く仲間として楽しく過ごすことができ、感謝しております。また、これまでの生活を支えてくれた家族には深く感謝しております。

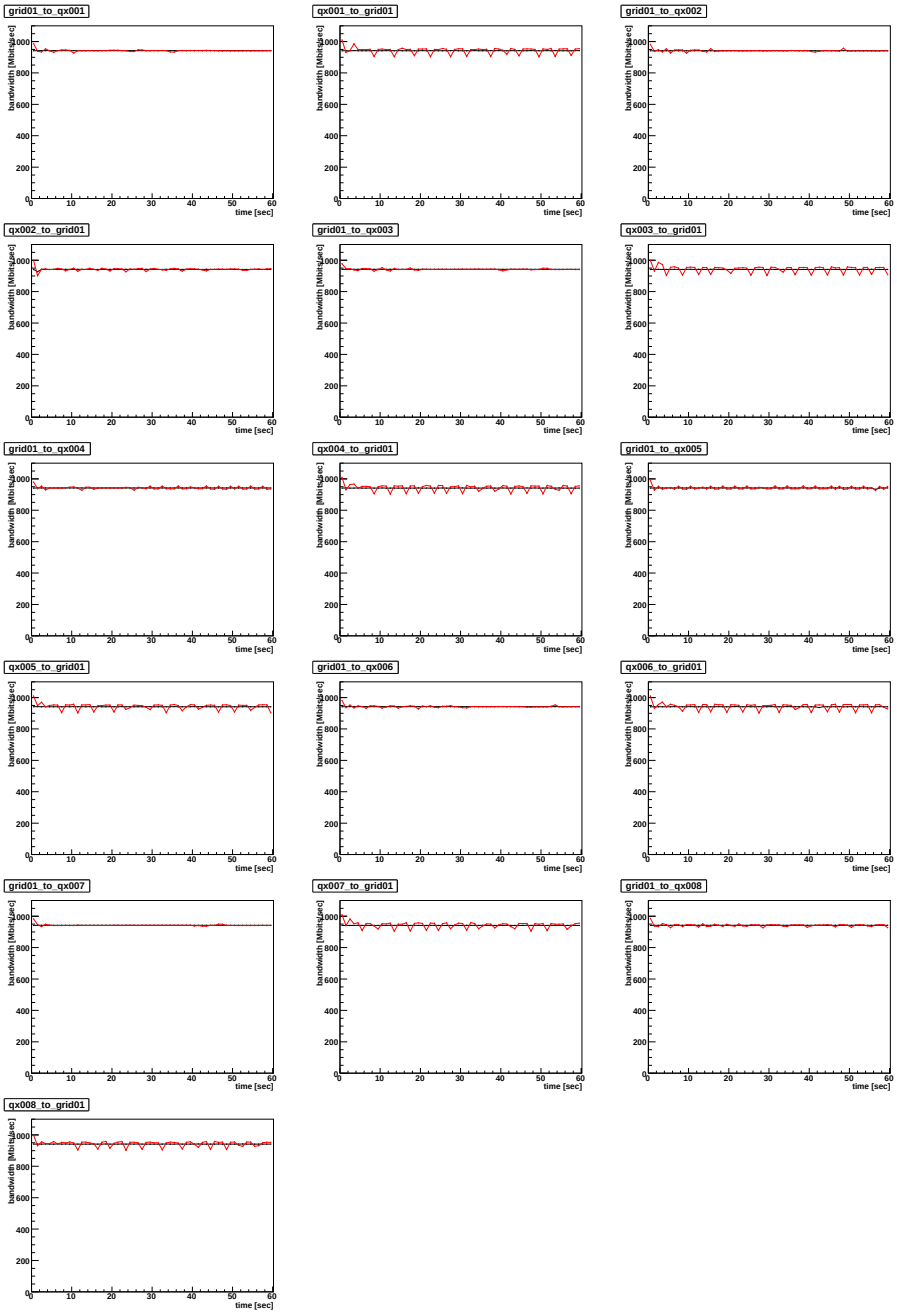
本論文は多くの方々のお陰で完成したものと思っています。皆様本当にありがとうございました。心より感謝いたします。

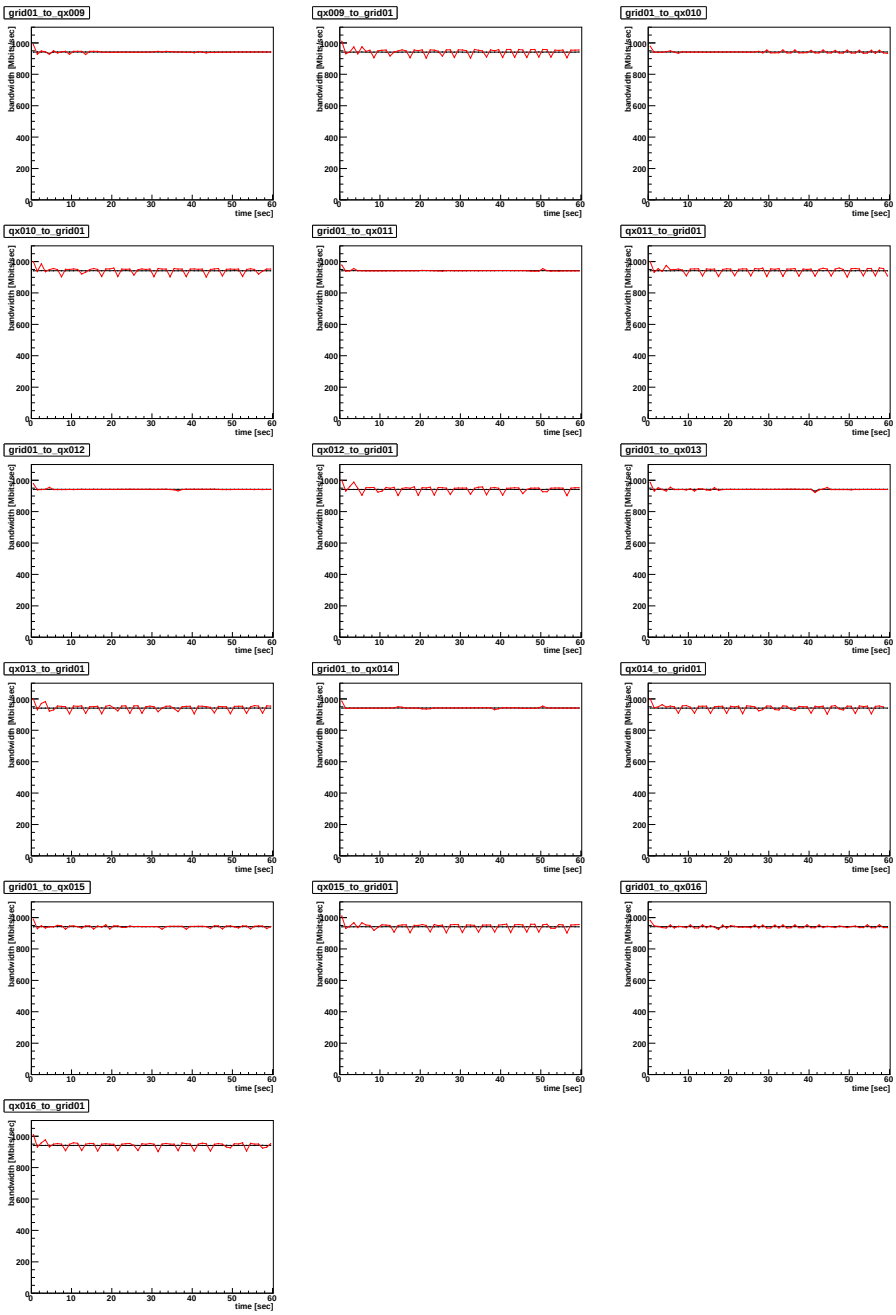
付 録A　Iperfによる研究室内部の速度測定

A.1　grid01, grid02, grid03間

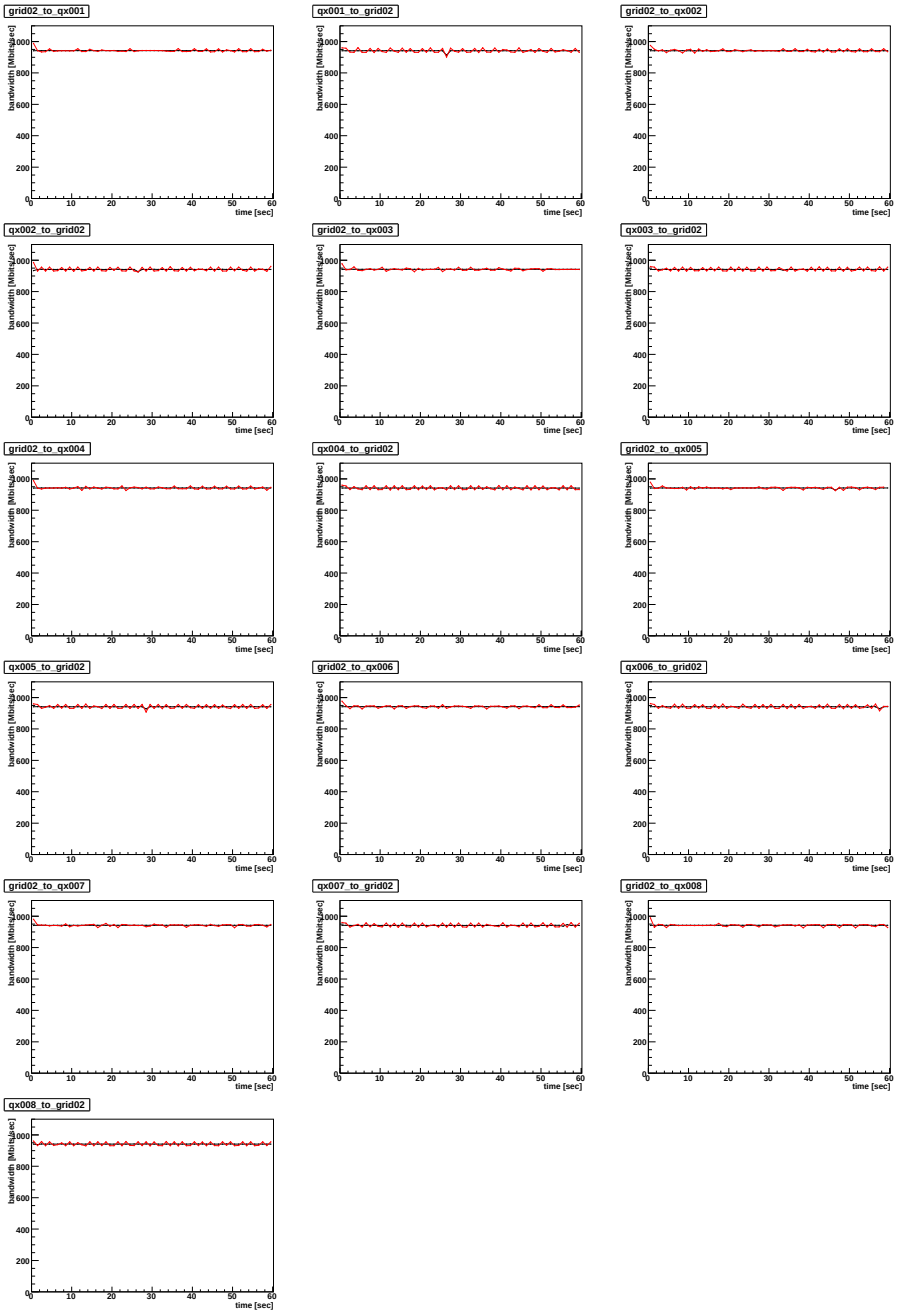


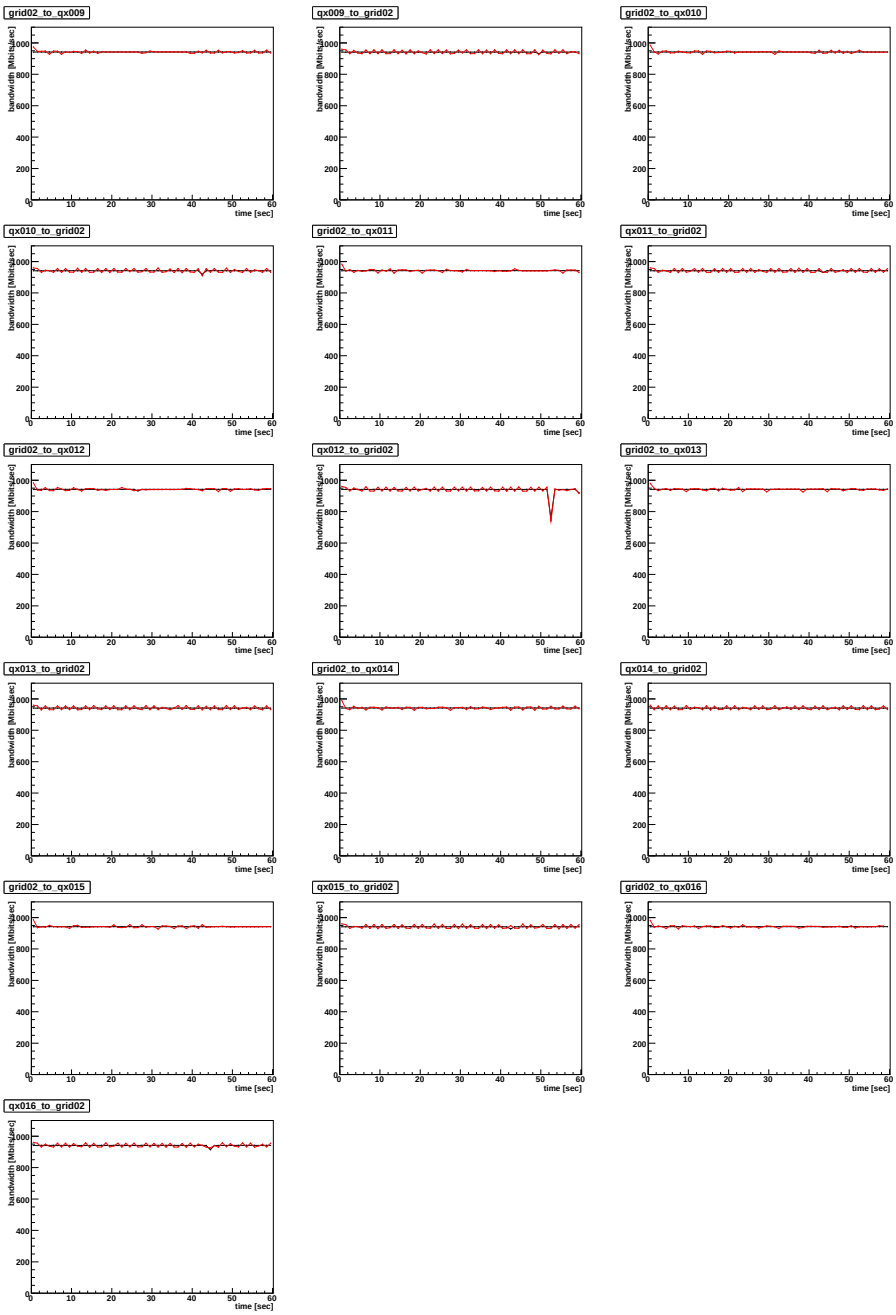
A.2 grid01, qx001〜qx016 間



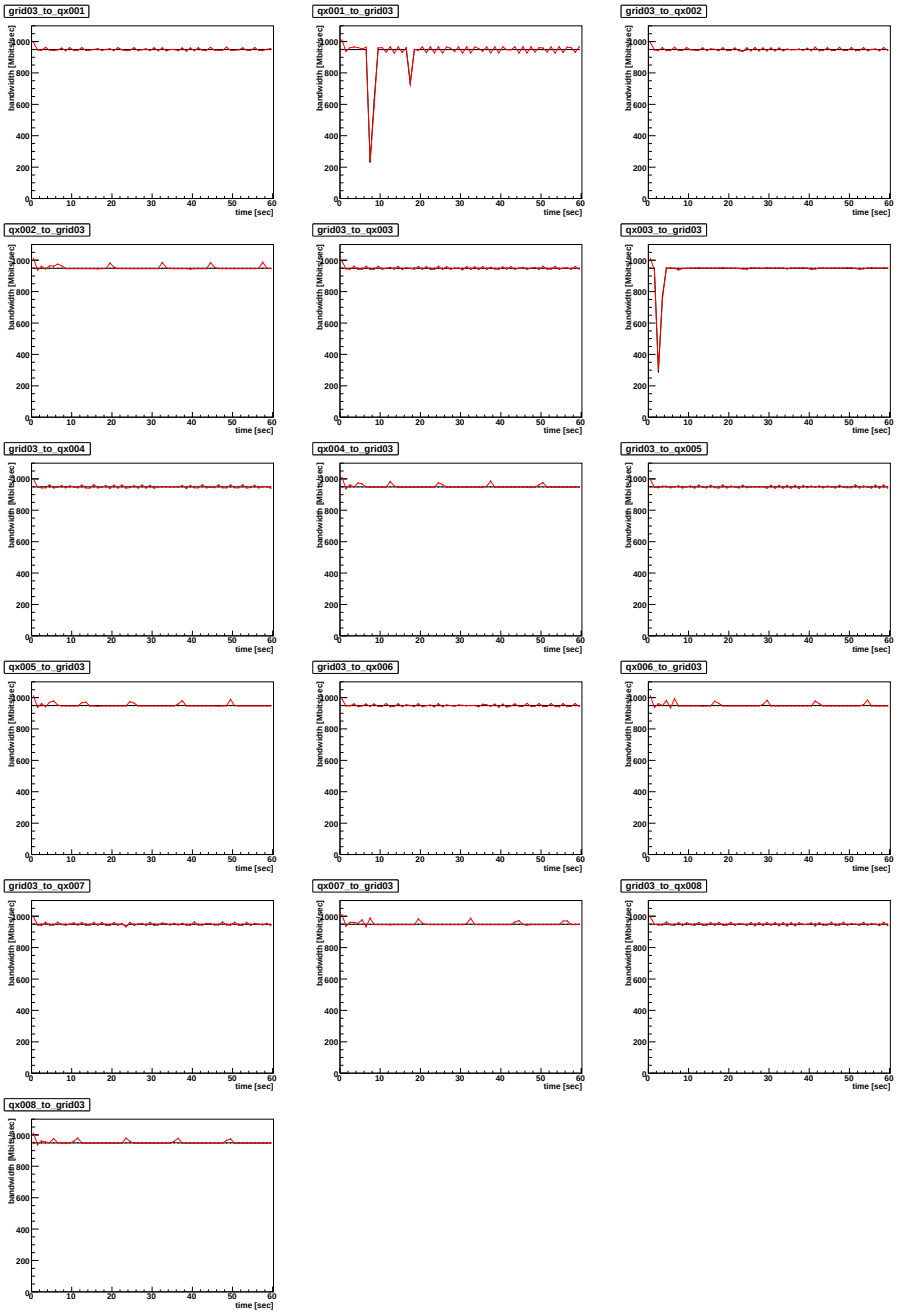

grid01_to_qx009
qx009_to_grid01
grid01_to_qx010
qx010_to_grid01
grid01_to_qx011
qx011_to_grid01
grid01_to_qx012
qx012_to_grid01
grid01_to_qx013
qx013_to_grid01
grid01_to_qx014
qx014_to_grid01
grid01_to_qx015
qx015_to_grid01
grid01_to_qx016
qx016_to_grid01
bandwidth [Mbit/sec]
time [sec]

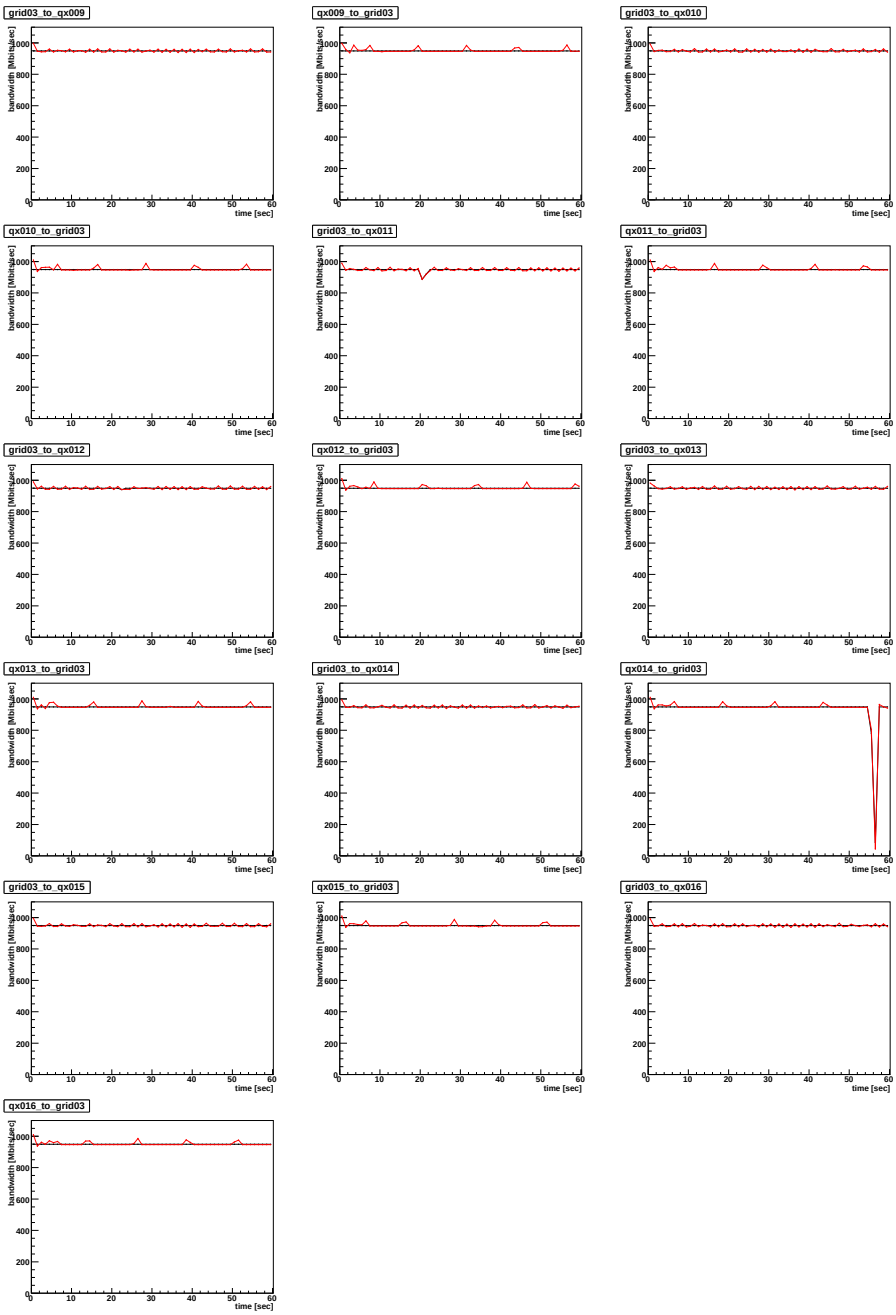
A.3 grid02, qx001〜qx016 間



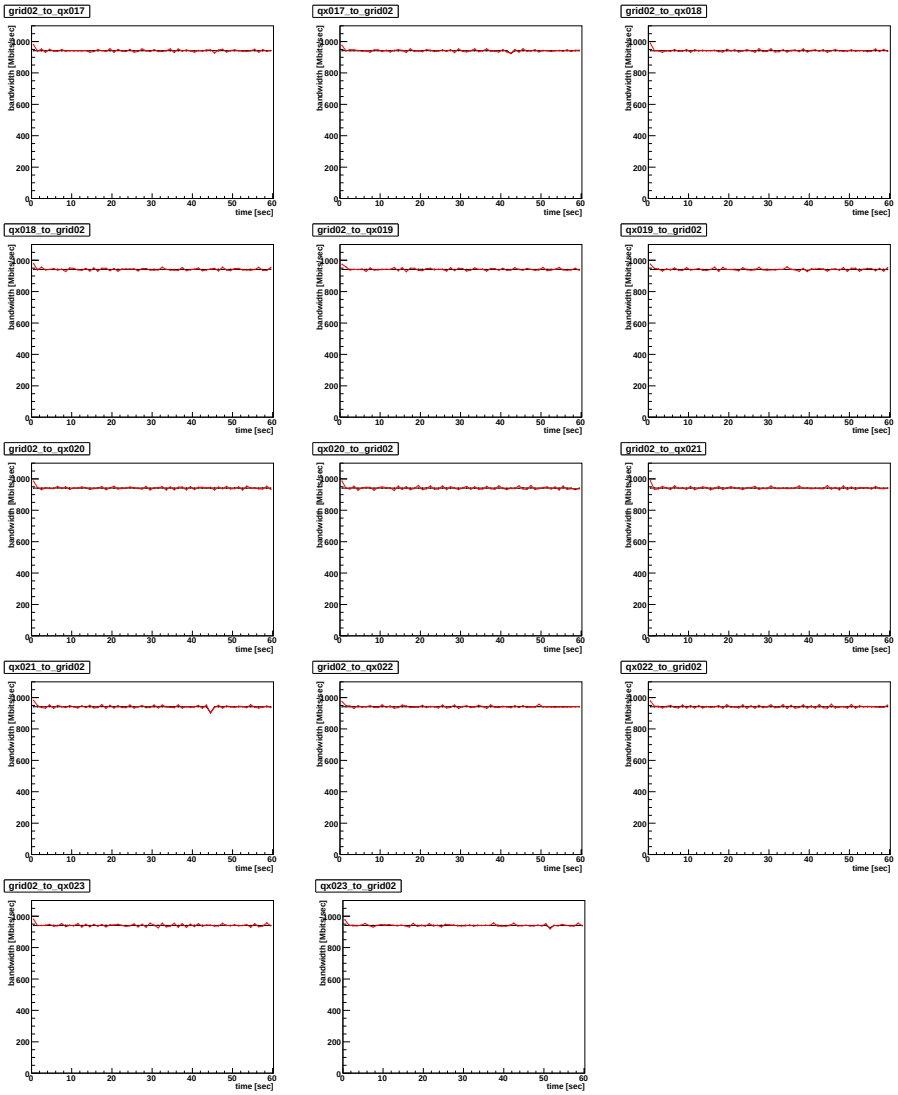

grid02_to_qx009
qx009_to_grid02
grid02_to_qx010
qx010_to_grid02
grid02_to_qx011
qx011_to_grid02
grid02_to_qx012
qx012_to_grid02
grid02_to_qx013
qx013_to_grid02
grid02_to_qx014
qx014_to_grid02
grid02_to_qx015
qx015_to_grid02
grid02_to_qx016
qx016_to_grid02
bandwidth [Mbit/sec]
time [sec]

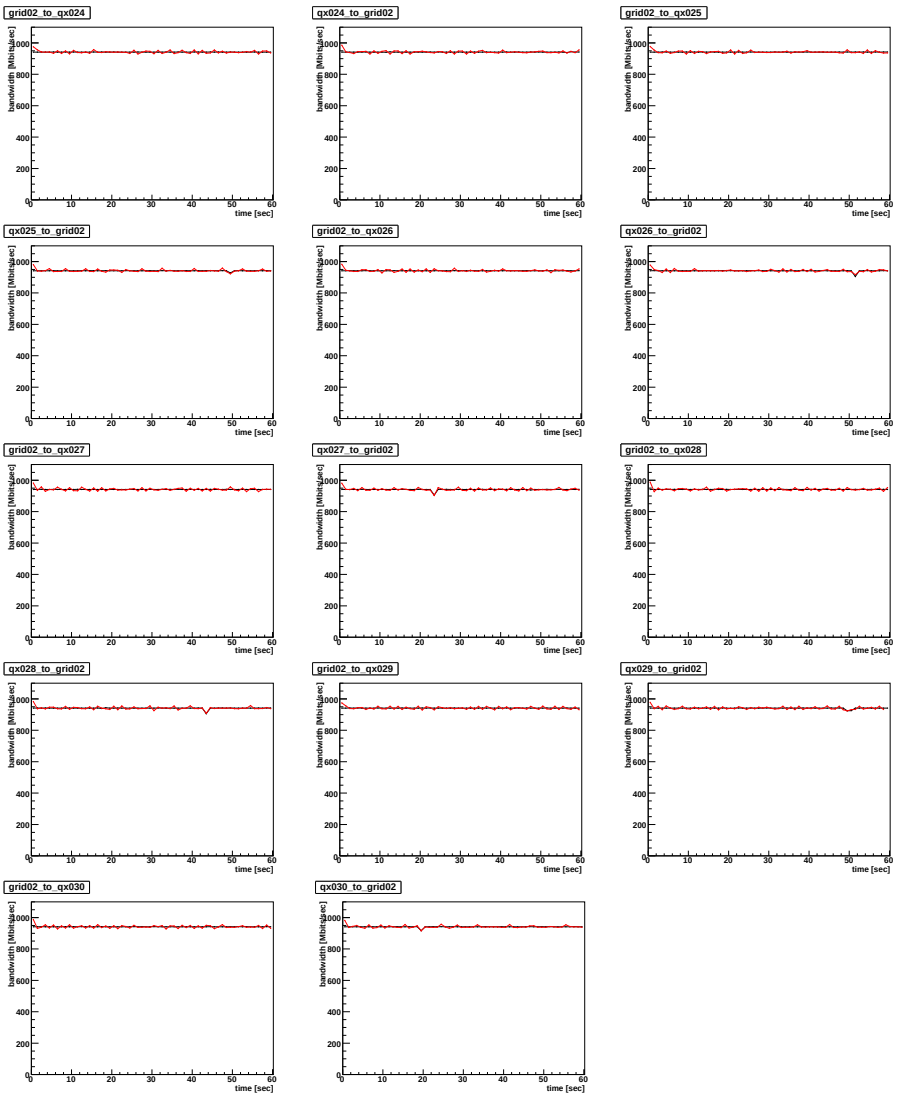
A.4 grid03, qx001〜qx016 間




grid03_to_qx009
qx009_to_grid03
grid03_to_qx010
qx010_to_grid03
grid03_to_qx011
qx011_to_grid03
grid03_to_qx012
qx012_to_grid03
grid03_to_qx013
qx013_to_grid03
grid03_to_qx014
qx014_to_grid03
grid03_to_qx015
qx015_to_grid03
grid03_to_qx016
qx016_to_grid03
bandwidth [Mbit/sec]
time [sec]

A.5　grid02, qx017〜qx030 間




grid02_to_qx024
qx024_to_grid02
grid02_to_qx025
qx025_to_grid02
grid02_to_qx026
qx026_to_grid02
grid02_to_qx027
qx027_to_grid02
grid02_to_qx028
qx028_to_grid02
grid02_to_qx029
qx029_to_grid02
grid02_to_qx030
qx030_to_grid02
bandwidth [Mbit/sec]
time [sec]

関連図書

[1] CERN-LHCC-2005-018、ALICE Technical Design Report of the Computing、 ALICE TDR 012(2005 年 6 月 15 日)

[2] P.Hristov 著、 AliRoot Primer

[3] Andreas Peters 著、 ALICE Analysis User Guide、 V.0.0μ(2006 年 6 月 15 日)

[4] Stephen Burke et al. 著、 gLite 3 User Guide、 CERN-LCG-GDEIS-722398(2007 年 1 月 17 日)

[5] 杉立徹 著、平成 15 年度〜平成 17 年度科学研究費補助金基盤研究 (B)(2) 研究成果報告書、 課題番号 15340079(2006 年 3 月)

[6] KEK GRID CA Web Repository、 https://gridca.kek.jp

[7] The Globus Alliance、 http://www.globus.org

[8] TWiki.LCG、 https://twiki.cern.ch/twiki/bin/view/LCG/WebHome

[9] Asia Pacific Regional Operations Center、 http://lists.grid.sinica.edu.tw/apwiki/APROC

[10] ALICE Off-line Project、 http://aliceinfo.cern.ch/Offline

[11] official FLUKA site http://www.fluka.org