Vol.2015-IOT-31 No.1
Vol.2015-SPT-15 No.1
2015/9/25

Overlay Cloud で構成する論文再現環境

横山 重俊[†1] 政谷 好伸[†1] 小笠原 理[†2]
大田 達郎[†3] 吉岡 信和[†1] 合田 憲人[†1]

バイオインフォマティクス分野などデータセントリックサイエンス分野では，論文発表された実験の再現性確保に対する要請が強く，例えばバイオインフォマティクス分野では DNA 塩基配列の公共データベース構築等によるデータ共有，およびデータ解析ソフトウェアのオープンソース化による処理プログラムの共有が進んでいる．しかし，再現性確保にはプログラム実行に関する次の課題が存在する．1.データ処理ソフトウェアが複雑化・多様化している．2. 次世代シーケンサー普及により大量データが分散発生する．3. データ解析量が増大している．本研究では，これらの課題を解決するため，Overlay Cloud アーキテクチャを活用し，インタークラウド上でのバイオインフォマティクスワークフロー再現環境を構築することを目的とする．構築する論文再現環境では，1. データ解析ソフトウェアのコンテナ化によるクラウドを跨る実行可能性確保（可搬性），2. コンテナ分散配置によるデータとデータ解析プログラム間遅延削減（分散処理[データ]），3. 分散処理基盤利用によるデータ解析ソフトウェアの処理性能向上（分散処理[コンピュート]），を実現することにより，これらの課題を解決することを目指している．本論文では，特に第一の課題である可搬性確保に取り組んだ結果について報告する．

A Scientific Paper Reproducible Environment with Overlay Cloud Architecture

The Inter-Cloud is a promising approach for the distributed application demands in some of HPC applications, like Next-Generation-Sequencing Data analytics. However, building the Inter-Cloud environments requires IT expert knowledge. This paper presents an architecture called Overlay Cloud and Virtual Cloud Provider (VCP), which is a middle-ware to automatically build a set of virtual resources on the Inter-Cloud and ease the knowledge requirements. That aims to help to realize ubiquitous scientific paper reproducible environments.

Shigetoshi Yokoyama[†1], Yoshinobu Masatani[†1] , Osamu Ogasawara[†2],
Tazro Ohta[†3], Nobukazu Yoshioka[†1], Kento Aida[†1]

1. はじめに

科学論文の再現性は長年の課題である．また，科学研究の四パラダイムのうち，計算環境に支えられる第三パラダイム：Computational と第四パラダイム：Date-intensive computing の重要性が様々な科学分野で高まっている．これら計算環境を使った実験に基づく論文は本来的には再現性確保が比較的容易なはずである．しかし，計算環境自体の再現性確保策の未成熟さから，再現性確保が一般化するまでには至っていない．これらの背景により，諸外国でも昨今論文再現検証環境への関心が高まっている．本論文では科学論文で扱う実験，特に計算科学による実験自体の再現環境のことを論文再現環境と呼び，研究対象とする．具体的には，コンテナ技術を用いた計算環境の再現性確保環境を整備し，それを計算科学に基づく分野の科学論文の再現性に適用する．

一方，国立情報学研究所（以下，NII）では，アカデミックインタークラウド構想[1]に基づきアカデミックコミュニティ向けクラウド基盤の構築を検討している．アカデミックコミュニティクラウド内のリソースだけではなく，プライベートクラウド側のリソース，そしてパブリッククラウド側リソースに跨ったクラウド連携の方法としてアカデミックインタークラウド構想では，Overlay Cloud アーキテクチャ[2]で構成する広域に分散した Virtual Cloud を提供する Virtual Cloud Provider の実現を前提としている．本稿では，論文再現性環境の例として Overlay Cloud アーキテクチャーにもとづいて構成したバイオインフォマティクス分野の論文再現環境について紹介する．以下，第２章で論文再現環境に関連する背景状況について述べ，第３章で筆者らの提案するアプローチについて述べる．第４章で論文再現環境に関する要求を整理，第５章では今回構築した論文再現環境の設計と実装例について説明する．また，第６章では，その評価について報告し，第７章でまとめる．

2. 背景

2.1 論文再現環境への要望の高まり

再現性は科学的方法の根幹をなす主要な原則であり，「近代科学」を支えてきた重要な概念の一つである．再現性のある実験を実施し，それを社会で共有し，積上げることで新しい科学的事実を獲得して行く手段として科学論文が存在する．従って，科学論文に記述されている内容は再現可

†1 国立情報学研究所 National Institute of Informatics
†2 国立遺伝学研究所 National Institute of Genetics
†3 ライフサイエンス統合データベースセンター
Database Center for Life Science

能であることが前提であり，それが学術コミュニティ活動の礎となっている．しかしながら，医学生物学系の論文の90%近くが再現性を持たなかったというレポート[3]があり，現実にはその礎が崩壊しかねない状況が出現している．この状況に対応して論文再現に関するレポート[4]に見られるように一部の分野で，論文再現検証環境の構築に関するトライアルが始まっている．

一方，科学研究手法は現在まで三つのパラダイムを経ていると言われている（第一パラダイム： empirical，第二パラダイム： theoretical, 第三パラダイム： computational）さらに，これらに加えて，膨大な一次データ，二次データ，の存在と，遍在する計算能力の存在を前提にした，第四パラダイム： date-intensive computing が台頭しようとしている．また学術のあらゆる分野で，今後第三パラダイム，第四パラダイム（Computational Science）による科学的発見が増加すると考えられる．Computational Science に基づく論文は計算機環境という本来的には再現性の高い環境を前提としているため再現性確保が比較的容易なはずである．しかしながら，論文再現率を示す別のレポート[5]によれば2011 年時点でも再現のためのソースコードが公開されている比率は 21%,データは 9%となっている．さらに障壁となっている理由に関するサーベイ[6]では主な問題は論文環境を整理しドキュメント化する時間（77%），論文読者の質問への対応（52%）となっている．一方同じサーベイで，再現性の有用性については共感があることも報告されている．

上記障壁の根本要因の一つは，本来可能であるはずの計算環境自体の再現性確保が，現実には研究者に簡単に手の届く状態に置かれていないことにある．

2.2 論文再現環境構築に向けた取り組み

実際，バイオインフォマティクス分野の最近の動きを見ても，現状のクラウド技術や仮想化技術を前提とした，いわゆる仮想アプライアンスで論文の再現環境を実現する Genomics Virtual Laboratory [7], Pitagora-galaxy [8] のような仮想マシン技術やクラウド技術を活用した計算科学実験環境の可搬性確保への取り組みが進んでおり，ゲノム解析ワークフローの共有を進められている．しかしながら，可搬性に関して，次の問題が残されている．

(1) クラウド基盤毎の固有の仮想アプライアンスフォーマットやアプリケーションインタフェースに依存するため，一つのクラウド基盤の上で動作した論文再現環境を他のクラウドへ移植する際に手間がかかり，それが論文再現検証環境の流通を阻害する．

(2) 複数マシンで構成される環境の復元について，クラウド基盤固有の手段を持っているクラウドプロバイダは存在するけれど，それぞれクラウド基盤に固有の手段であり，クラウド基盤を跨る標準的な手段が提供されていない．これが論文再現検証環境の流通を阻害している．

また，他の課題である分散処理（[データ]，[コンピュート]) についてはまだ本格的な取り組みはなされていないのが現状である．

2.3 インタークラウドを実現する Overlay Cloud

インタークラウドを実現するアーキテクチャの一つである Overlay Cloud アーキテクチャーは, インタークラウドの構成要素となる既存のクラウドプロバイダ(Real Cloud Provider と呼ぶ)からマシンリソース（仮想マシンや物理マシン）やネットワークリソースをそれらクラウドプロバイダ毎に決められた利用方法に従って提供してもらい，それらを SINET[9,10]のような広域網で接続する．

それぞれのマシンリソースの上に docker のようなコンテナ実行環境と Overlay Network 環境を予め組み込んでおくことで，それらの上にクラウドをまたがる分散処理基盤 Mesos クラスタ[11]などを容易に構築することができる．アプリケーションを直接ここで提供されるコンテナに導入することは可能であるけれど，筆者らはこのレベルのコンテナをベースコンテナと呼び，Mesos などの分散処理基盤を構成するノードを格納するアーキテクチャを提案している．

実際のアプリケーションが入るコンテナは，Mesos に代表される近年の実行基盤はコンテナ実行機能を持っていることが想定できるので，アプリケーション提供者は，アプリケーションレベルのコンテナ（アプリケーションコンテナと呼んでいる）にアプリケーションを格納し，ベースコンテナで実現された実行基盤でアプリケーションを実行することとなる．つまり実装上，アプリケーションコンテナはベースコンテナの中に入れ子状に入っている「コンテナ内コンテナ」の状態となる．

さらには，Academic Inter-Cloud(AIC)[12]のように SINET に直結しているベアメタルクラウドがインタークラウドの構成要素である場合，それらを SINET L2VPN でオンデマンド接続し，それぞれのベアメタルクラウド内の物理マシン接続することでセキュアに接続された物理マシンクラスタが出来上がり，その上に構築したコンテナで作ったクラスタもセキュアにしかも，性能上も[2]で示したように，ほぼ物理マシン，物理ネットワークの性能を享受できる．

3. 提案アプローチ

本研究では, コンテナ技術を用いた Overlay Cloud の上に計算環境自体の再現性確保環境を整備し，それを科学論文の再現性に適用する研究を行っている．Overlay Cloud アーキテクチャを用いることで論文の筆者が使った実験環境を論文の読者が選択したどのクラウド基盤上でも容易に再現できることとなり現状の再現のためのソースコードとデータの公開が促進され，科学論文の価値を高めることができる．現在普及して来ているクラウドコンピューティング基盤を活用し，その上にコンテナ技術を用いた Overlay Cloud を構成することで，グローバルに活用できる計算環境の再

現検証手法を整備する．図 1 で示すように Overlay Cloud は既存のクラウド環境（プライベート/コミュニティ/パブリック）の上にクラウド環境に依存しないコンテナ環境を Overlay し，アプリケーション利用者とクラウド基盤間のインタフェースを分離する技術である．このため，利用者は自由にコンテナ実行基盤を選択できる．論文読者はこの環境を用いることで，論文にリンクされているボタンを押下するだけで，簡単に，自分の選択したクラウド環境上で論文再現検証環境が得られるようになる．

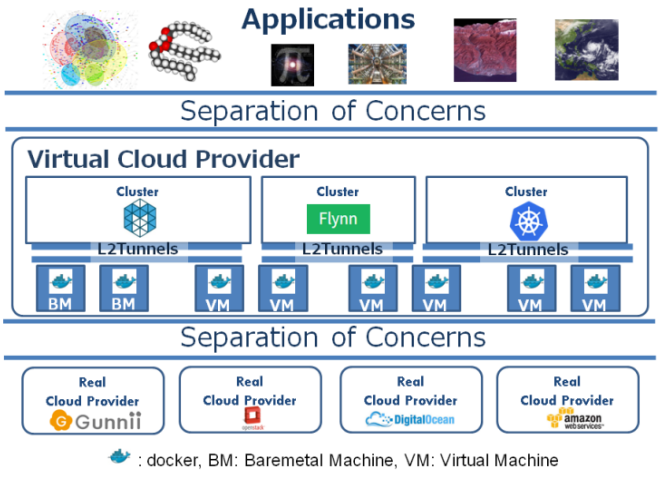


図 1. Overlay Cloud アーキテクチャ

4. 論文再現環境への要求整理

第 4 章以降は分野をバイオインフォマティクスに絞ってその論文再現環境について議論する．しかしながら，ベースコンテナ上のアプリケーション実行基盤をそれぞれのデータセントリックサイエンス分野向けにカスタマイズすることで，それぞれの分野への横展開もで可能な部分は大きいと筆者らは考えている．

バイオインフォマティクス分野において，研究者自身が複数のデータ解析ツールを組み合わせてワークフロー的に解析作業を行うことが多くある．特に，次世代シーケンサー(NGS)を活用したゲノム解析に用いるソフトウェアの数は飛躍的に増え，そのための専門的な知識や大規模なハードウェアが必要となることが多い．また，データが巨大化していること，さらにはプライバシー保護のためもありデータ自身を移動できないケースが増えているため，各地に分散したデータを合わせて分析しなければならない場合が増加すると予想される．このような背景を踏まえ，NGS データ解析のためのサービスを論文再現環境のターゲットとして，その実現に必要な機能要件を以下の 4.1 から 4.4 節のように整理した．

4.1 可搬性

(1) ツールやライブラリのバージョンなどによる実行環境の統一化

同一バージョンのツールやライブラリを利用者間で共有できなければならない．

(2) ワークフロー記述の統一化

実行環境毎のジョブスケジューラなどの違いを吸収し，実行環境に依存しない形式でワークフローを記述する必要がある．

(3) 既存研究の追試実験

ある研究者が行った計算と同じ結果を得るために，その研究者が実行したワークフロー（使用したプログラムとその組み合わせ手順），および実行時パラメータを再現できる必要がある．

(4) 既存研究データの再利用による派生実験

追試実験に加えて，二次解析実験やオプショナルな実験を同じデータを使って行う必要がある．

(5) テスト実行機能

テスト実行として，EXPERIMENT（次世代シークエンサからの出力データ単位）の一部を切り出すなどにより，少量のデータセットとテスト用のノード環境を利用してワークフローを実行する．

4.2 分散処理[データ]

国立遺伝学研究所の DNA Data Bank of Japan - DDBJ - に代表される公開のゲノム情報データベースのデータサイズは年々増加しており，ゲノム情報分析ツールやワークフローの実行場所への複写が実験再現に要する時間の増大を招いている．さらに，次世代シーケンサーの低価格化と性能向上に伴い各研究機関側で発生するデータ量も増加している．これら分散しているデータを一か所に常に転送し，そちらでゲノム情報分析ツールやワークフローの実行するという計算実験を続けることが難しくなって来ている．従って，データを転送するのではなくゲノム情報分析実行ソフトウェアを分散したデータのある場所にそれぞれ転送し，データの近くで実行する分散処理の仕組みがデータサイドの状況変化から求められている．

4.3 分散処理[コンピュート]

NGS データ解析のツールやワークフローについて，クラウド内で分散処理を適用する際には目的に応じて以下のようなバリエーションが考えられる．

(a) 1 個の EXPERIMENT を分割し，同じ処理を並列に実行する．単一のツール内における処理速度を向上させることを目的とする．

(b) 1 個の EXPERIMENT に対して異なる処理を並列に実行する．異なる実行オプション・パラメータによる試行を効率的に行うことを目的とする．

(c) N 個の EXPERIMENT に対して同じ処理を並列に実行する．大量の検体サンプルを扱う処理を効率的に行うことを目的とする．

データサイズや使えるリソース量によっては両者の組み合わせもありうる．

4.4 非機能要件

(1) セキュリティ

非公開のヒトゲノムデータの安全な管理が必要である．

解析の入力・出力データ，および中間データを配置するノードの信頼性を確保する必要がある．

(2) 性能

コンテナ化しない場合と比較して数%以内のオーバーヘッドに収まる必要がある．

5. 論文再現環境の設計・実装例

5.1 システムイメージ

バイオインフォマティクスワークフロー実行エンジンGalaxy[13]による解析ワークフロー実行環境として，Galaxyをフロントエンドに，分散処理実行環境としてMesos/Aurora[14]をバックエンドにした，図2に示すようなDockerを活用した基盤システムを構築する．

Galaxyのジョブ実行環境にはMesos/Auroraクラスタを使用し，コンテナ化されたGalaxy Toolがクラスタ内のノードで実行される．各ノードはネットワーク共有ストレージをマウントし，入出力データを共有する．

また，検証のために使用したNGSデータ解析はFAMTOM5プロジェクト[15]の基礎解析ワークフローである．他のワークフローの適用についてはGalaxyコミュニティなどでの活動と連携し，対象を増やして行きたい．

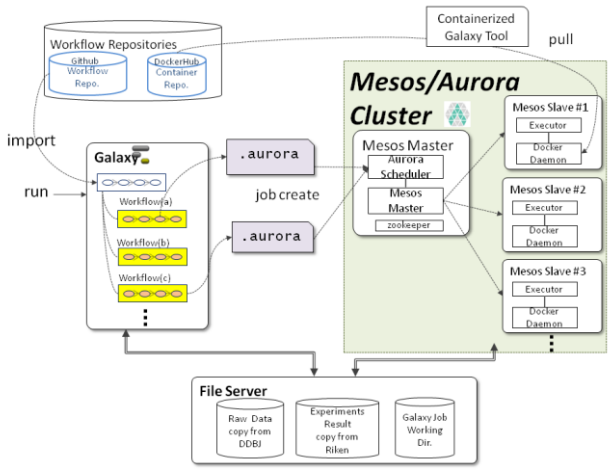


図2　バイオインフォマティクス論文再現環境

5.2 アクター

本基盤システムの構築から利用までを想定したユースケースシナリオにおいて，システムに関与する人とその役割を以下に示す．

(1) システム基盤管理者

システム基盤を提供・管理・運用する．

(2) Galaxy サービス管理者

Galaxy およびそのツールをコンテナ化し，複数拠点上で統一的に実行できるようにサービスを構築する．このGalaxy サービスを VCP 上で実現することを想定しており，VCP 自体の構築を行うアクター（VCP 管理者）が別途存在する．

(3) パッケージメンテナー

Pitagora-Galaxy のように，各種ツールに加えてワークフローやその他の便利な機能（Galaxy と連携して動作するアプリケーションなど）も含めた「Galaxy パッケージ」を作成する．複数のツールを組み合わせた際の依存関係の解決，それらのバージョン管理も行う．

(4) Galaxy サービス利用者

(4-1) 解析ワークフロー作成者

Galaxy ワークフロー，Galaxy ジョブ（1つのワークフローを実行するために必要な情報をひとまとめにしたもの）を作成するユーザ．

(4-2) 解析ワークフロー利用者

Galaxy ワークフローを利用するユーザ．
利用したワークフローを改変することで作成者となることもある．

(4-3) データ解析ツール作成者

Galaxy ワークフローから利用できるようにコンテナ化まで行うこともある．ツール作成者が解析ワークフローを必ずしも実行するとは限らない．

5.3 Mesos/Aurora 環境構築

Apache Aurora では，ローカルな All-in-One 環境で簡単に実行環境（開発向け）を構築するための Vagrant スクリプトがチュートリアル用として提供されている．この内容を参考に，複数ノードで構成される実運用環境を想定した Mesos クラスタの構築手順を確立した．

クラスタの各ノードは，VCP で構成されたホスト上にベースコンテナ形式でデプロイすることを前提とし，ここで確立した手順により作成したコンテナイメージは現在 Docker Hub から取得可能 である． [16]

Docker コンテナ上で実行環境を構築し，各プロセスを起動するために必要な点について以下に説明する．

(1) パッケージ追加

provision-dev-cluster.sh において，apt-get コマンドで取得可能な OS 標準パッケージに加え，以下のパッケージを別途インストールする．

- thrift-compiler_0.9.1_amd64.deb
- mesos_0.21.1-1.0.ubuntu1204_amd64.deb
- mesos.native-0.21.1-py2.7-linux-x86_64.egg

(2) Docker-in-Docker サポート

Mesos Slave ノードでは Aurora Executor が Docker Containerizer を利用するため，コンテナベースの Slave ノードにおいて Docker daemon を起動しておく必要がある．このためのサポートツールとして，Docker-in-Docker に含まれる wrapdocker スクリプトを利用する． なお，Slave ノード用のベースコンテナ起動時 (docker run) には，

"--privileged" オプションを指定しなければならない.

(3) Aurora のビルド

admin_client, client, executor, observer, scheduler の各コマンドとサービスをビルドする.

(4) ZooKeeper 起動

Mesos Master ノードにおいて ZooKeeper を起動する.

(5) Aurora クラスタ設定

/etc/aurora/cluster.json を作成する.

(6) Mesos Master 起動

自ホストの IP アドレスを指定する.

(7) Mesos Slave 起動

自ホストおよび ZooKeeper ホストの IP アドレス、CPU 数、メモリ・ディスクのリソース量（単位:MB）をそれぞれ指定する.

(8) Aurora Scheduler 起動

Mesos Master が稼働するホストにおいて，自ホストの IP アドレスを指定する.

(9) Thermos Observer 起動

Mesos Slave が稼働するホストにおいて起動する.

5.4 Galaxy 設定

各ツールに対応する Galaxy Tool 定義ファイルを作成し，Galaxy へ組み込む.

(1) Galaxy Server

データセットの保存先としてネットワーク共有ファイルシステムからマウントしたディレクトリを Galaxy Server 設定ファイルに記述する.

(2) Galaxy Job

Galaxy のジョブ実行先として Mesos/Aurora 環境に接続するための Job Runner を設定する.

5.5 Galaxy - Aurora 連携

5.5.1. 接続方式案

Galaxy ワークフローの内容を Aurora ジョブに変換する際の方式として，一括方式とステップ単位方式の 2 種類が考えられる.

(a) 一括変換方式

ワークフロー全体を 1 つの Aurora ジョブとして生成する. この方式の利点としては，ワークフロー全体を Aurora の単一のジョブとしてほぼそのまま再現できることである.

一方で，現行 Aurora の仕様によりツール実行先コンテナイメージが Job の単位でしか指定することができないため，ツール単位でコンテナを使い分けることが不可能となる.

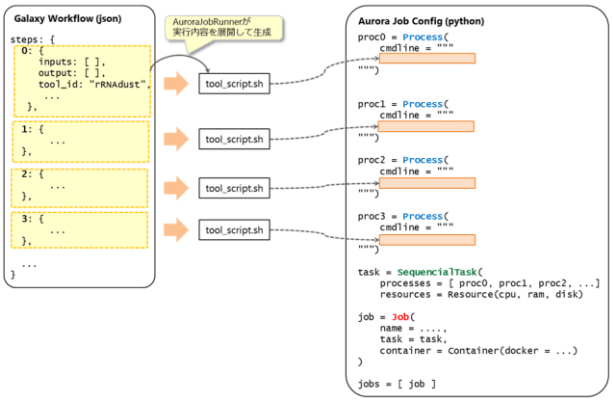


図３　一括変換方式

(b) ステップ単位変換方式

ワークフローの各ステップ単位で Aurora ジョブを生成する.

この方式の利点としては，ツール単位で実行先コンテナを指定することが可能であり，Galaxy Web UI と Aurora のジョブ実行を完全に連携させることができる.

その反面，ワークフローのステップコントロールを Galaxy が担うことになり，複数タスクから構成されるようなジョブを定義可能な Aurora を利用する意義が薄れてしまう.

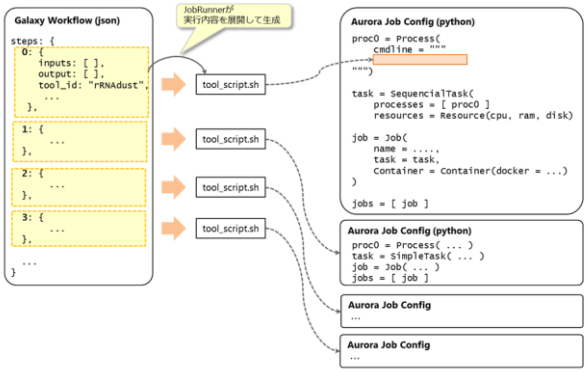


図４　ステップ単位変換方式

5.5.2. 実現方式

「(a) 一括変換方式」による動作検証を行った.

FANTOM5 基礎解析ワークフローで使用される 3 種類の解析ツールすべてを含むコンテナイメージ を作成し，それを "Job" 単位に設定する.

Galaxy JobRunner については，Aurora ジョブの "Process" 単位に設定されるコマンドラインを生成する機能に限定したものをプラグインとして実装した.

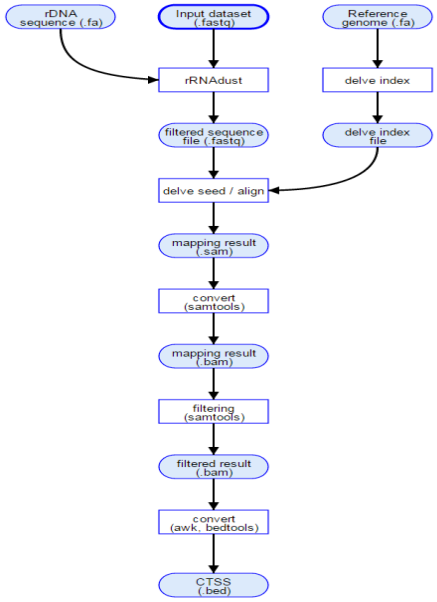
6. 評価

前述の設計方針のもとに，FANTOM5 基礎解析パイプラ

インの再現実験環境について，異なる複数クラウド基盤での構築・動作確認と複数クラウドを跨るインタークラウド環境で構築・動作確認し，それぞれワークフロー実行時間を測定した．

6.1 FANTOM5 基礎解析パイプライン概要

FANTOM5 プロジェクトにおいて利用され，Web サイト上で公開されているワークフロー（図５）を Galaxy 上に再現したものを本評価で利用した．



(http://wiki.pitagora-galaxy.org/wiki/index.php/Workflow_CAGE_01)

図５　FANTOM5 基礎解析パイプライン

(1) 使用する解析ツール

rRNAdust… 2 ミスマッチまでの rRNA 配列を除去

delve

index…マッピングのためのインデックス配列を作成

seed/align…　マッピングとマッピングクオリティの算出

samtools… SAM 形式から BAM 形式へのフォーマット変換，およびマッピングのクオリティで結果のフィルタリング

bedtools … BAM 形式から BED 形式の CTSS ファイルに変換

(2) 入力データ

rRNA 配列 (U13369.1) リファレンスデータ

ゲノム配列 (UCSC hg19) リファレンスデータ

シーケンスファイル（DRA 内 FASTQ 形式ファイル）

照合対象とする結果の CTSS ファイル

(3) ツール実行パラメータ

rRNAdust…リファレンスに対するミスマッチ数 [2]

delve seed… シード長 [12]， ステップサイズ [8]

delve align… 出力されるアラインメント数 [1]

samtools…マッピングクオリティ[q>20, %-identity>85%]

6.2 可搬性確認

FANTOM5 基礎解析パイプラインの再現実験環境を 2 つのクラウド基盤にそれぞれ構築し，実行時間のベンチマークを実施した．このことで再現実験環境の可搬性を示す．

ここでは 2 つのクラウド基盤として，AIC (NII Academic Inter-cloud)と，AWS (Amazon Web Service)を対象とする．

(1)　AIC (NII Academic Inter-cloud)

表１に AIC 上評価環境のノード構成と計算ノードのハードウェア仕様を記す．また，AIC での実行時間測定結果を図６に示す．横軸が実行スレッド数であり，縦軸はワークフロー全体を実行するのに要した時間を分を単位として記したものである．色の異なる棒グラフは入力リード数を変化させた場合の実行時間を示している．

表１　AIC 上評価環境のノード構成とハードウェア仕様

ノード構成

Mesos Master, Galaxy (shared directory: NFS)
Mesos Slave #1
Mesos Slave #2
Mesos Slave #3
Mesos Slave #4
AICN Node　(NAT Gateway)
NFS Server @AIC　(/mnt1/nig, /mnt1/riken)
Docker Image Registry

計算ノード(Mesos Slave) ハードウェア仕様

・Intel Xeon E5-2670　2.60GHz (#1, #2, #3) ・Intel Xeon E5-2650　2.00GHz (#4)
・32cpu (2socket x 8core x 2thread)
・RAM 96GB

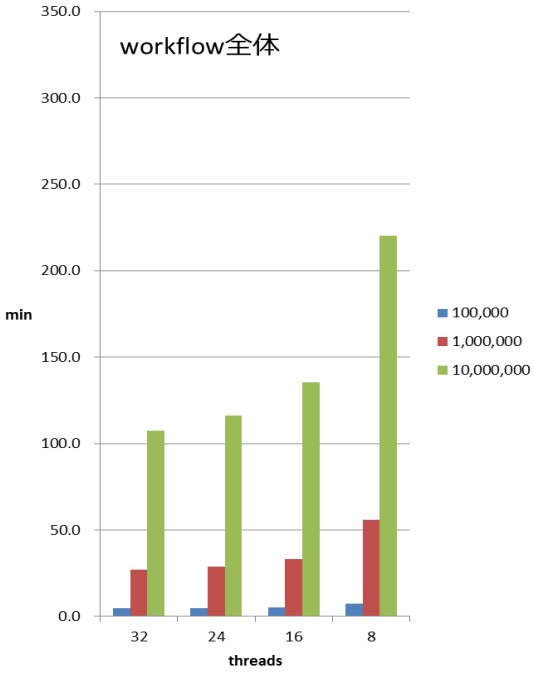


図６　AIC での実行時間測定結果

(2) AWS (Amazon Web Service)

表２に AWS 上評価環境のノード構成と計算ノードのハードウェア仕様を記す．また，AWS での実行時間測定結果を図７に示す．横軸が実行スレッド数であり，縦軸はワークフロー全体を実行するのに要した時間を分を単位として記したものである．色の異なる棒グラフは入力リード数を変化させた場合の実行時間を示している．

表２　AWS 上評価環境のノード構成と
ハードウェア仕様

ノード構成

Mesos Master, Galaxy (shared directory: NFS)
Mesos Slave #1 (AWS-EC2: c3.4xlarge)
AICN Node　(NAT Gateway)
Docker Image Registry

計算ノード(Mesos Slave) ハードウェア仕様

・Intel Xeon E5-2666 v3　2.90GHz
・16cpu (1socket x 8core x 2thread)
・RAM 30GB

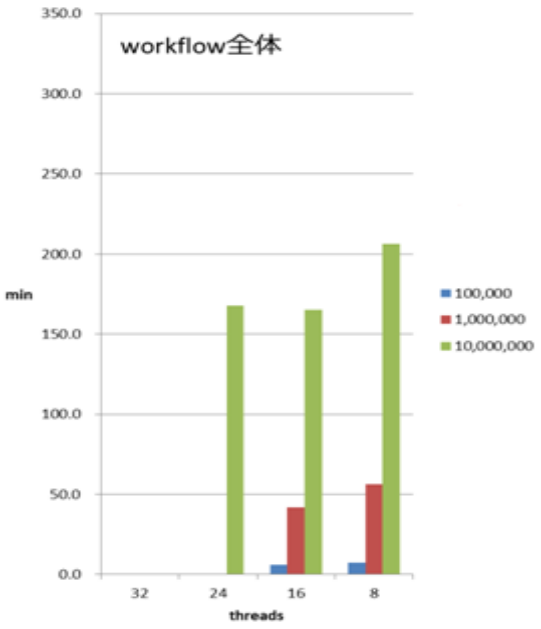


図７　AWS での実行時間測定結果

6.3 インタークラウド実現性

さらに, 3 拠点（NII クラウド, 北大クラウド, Amazon Web Services）の Real Cloud 群から構成されるインタークラウド上で論文再現環境を開発・構築した. このことで FANTOM5 基礎解析ワークフローの再現実験環境がインタークラウド環境で構築できることを確認し, 実行時間も図８ インタークラウドでの実行時間測定結果に示す通り，取得したので以下に報告する．

AIC 側にデータ入出力ディレクトリ(NFS)と, ジョブスケジューラ(Galaxy + Mesos Master)を配置し，北大クラウドの 2 ノード, および AWS EC2 インスタンス 1 ノードを計算ノード(Mesos Slave) に割り当てる．

AIC－北大間はベアメタルノードどうしが SINET L2VPN で直結され，その上に L2TPv3 トンネルを構成し，ベースコンテナ間を仮想 L2 ネットワークで接続する．

AIC－AWS 間は公衆網を経由するために IPsec による経路暗号化を行った上で, 同様に L2TPv3 トンネルを構成する．

6.3.1 インタークラウドネットワーク構成

AIC 側にデータ入出力ディレクトリ(NFS)と, ジョブスケジューラ(Galaxy + Mesos Master)を配置し，北大クラウドの 2 ノード, および AWS EC2 インスタンス 1 ノードを計算ノード(Mesos Slave) に割り当てる．

AIC－北大間はベアメタルノードどうしが SINET L2VPN で直結され，その上に L2TPv3 トンネルを構成し，ベースコンテナ間を仮想 L2 ネットワークで接続する．

AIC－AWS 間は公衆網を経由するために IPsec による経路暗号化を行った上で，同様に L2TPv3 トンネルを構成する．

6.3.2 MTU 設定

AIC-AWS 間のような公衆網を経由した L2TPv3 over IPsec を構成する場合，各接続先ホストのネットワーク I/F とトンネルネットワーク用の I/F で MTU を適切な値に設定する必要がある．今回，AIC－AWS 間接続における MTU 値をイーサネットのデフォルト値である 1500 から順次フラグメンテーションの起きない値を調査することで，フラグメンテーションによる性能を低下させないためには以下の値に設定を変更する必要があることを確認した．

IPsec のみ: MTU = 1422

L2TPv3 over IPsec: MTU = 1368

利用するクラウド環境に応じて MTU サイズを調整することにより，複数拠点のクラウド利用による解析環境は実用上問題無いことがわかった．

6.3.3 インタークラウドノード構成

表３にインタークラウド上評価環境のノード構成と計算ノードのハードウェア仕様を記す．また，インタークラウドでの実行時間測定結果を図８に示す．横軸が実行スレッド数であり，縦軸はワークフロー全体を実行するのに要した時間を分を単位として記したものである．色の異なる棒グラフは入力リード数を変化させた場合の実行時間を示している．

表３　インタークラウド評価環境のノード構成と
ハードウェア仕様

ノード構成

Mesos Master, Galaxy (shared directory: NFS)
Mesos Slave #1 @hokudai
Mesos Slave #2 @hokudai
Mesos Slave #3 @AWS-EC2: c3.4xlarge
AICN Node　(NAT Gateway)

NFS Server @AIC (/mnt1/nig, /mnt1/riken)
Docker Image Registry

計算ノード(Mesos Slave) ハードウェア仕様

Mesos Slave #1, #2 (hokudai)

・Intel Xeon E7-8870 2.40GHz
・40cpu (4socket x 10core)
・RAM 128GB

Mesos Slave #3 (AWS)

・Intel Xeon E5-2666 v3　2.90GHz
・16cpu (1socket x 8core x 2thread)
・RAM 30GB

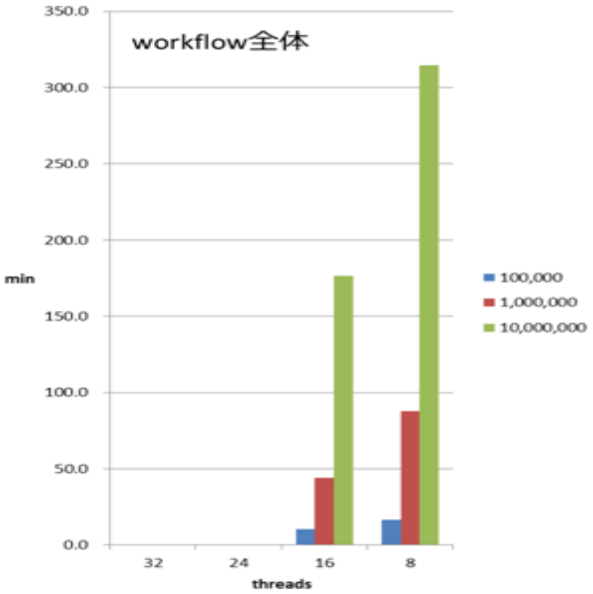


図８　インタークラウドでの実行時間測定結果

6.4 考察

この論文再現環境の実装と性能評価を通じて，以下のことが確認できた．6.2 節に述べた可搬性確認結果により Overlay Cloud で構成する論文再現環の可搬性が示せたと考える．なお，構築に要する作業時間は，Mesos クラスタを構成する各ノードに対応する Docker コンテナイメージを Registry に登録しておくことで，クラウド基盤上で IaaS ホストを起動後数分程度になっている．

Mesos により割り当てられたワークフローが動作するノードによって性能は変わるが，実行性能はクラウドに共通して同様の傾向となった．

ただし，NFS を AIC 側に固定しているため，計算ノードとの間のネットワーク帯域やレイテンシの大きさによっては性能面での影響を受ける可能性がある．実用に向けてストレージ部分についてはさらに考察が必要である．

7. まとめ

バイオインフォマティクス分野における HPC アプリケーションをターゲットとする Overlay Cloud により可搬性確保に対する有効性が確認できた．今後，以下の残課題に取り組む必要がある．

(1) 分散処理

今回のプロトタイプでは大量 EXPERIMENT を並列に処理することを想定したが，データサイズが大きな単一 EXPERIMENT を分割して処理するワークフローへの必要性は高いと考えられる．解析ツール毎に処理特性が異なるため，ワークフロー全体として最適な分割数や分割・統合のタイミングを判断する方式について検討が必要である．

また，データの分散に起因する広域分散による実行性能向上についても評価が必要である．

(2) クラウドストレージ機能

バイオインフォマティクス分野におけるデータ解析では大規模データの取り扱いが課題となる．今回のプロトタイプでは Galaxy と Mesos/Aurora クラスタの共有ストレージとして NFS を前提に構成したが，ジョブ実行場所でのデータの準備などを効率的に行うためにローカルストレージやクラウド上のオブジェクトストレージといったストレージ群をまとめるための機能の実現が必要である．

(3) 他分野への展開

他のデータセントリックサイエンス分野での論文再現性についての適用性を評価する必要がある．

謝辞　VCP の評価環境構築，VCP 評価にご協力頂いたアスケード社の那須野淳氏に謹んで感謝の意を表する．

参考文献

1) 合田 憲人，横山 重俊，吉岡 信和: アカデミックインタークラウドの構想，電子情報通信学会 サービスコンピューティング研究会，信学技報 113(376) 1-6，2014 年 1 月
2) 横山重俊，政谷好伸，吉岡信和，合田憲人: Overlay Cloud で構成するインタークラウド環境，電子情報通信学会 サービスコンピューティング研究会，信学技報 115(72) 1-6, 2015 年 6 月
3) Reproducibility in Computational Science: Why, What, and How?, Columbia University (July 2014)
4) Best Practices for Computational Science: Software Infrastructure and Environments for Reproducible and Extensible Research
5) Reproducibility in Computational Science: Opportunities and Challenges Victoria Stodden, Columbia University (March 2014)
6) Survey of the Machine Learning Community, NIPS (Stodden 2010)
7) Genomics Virtual Lab : https://genome.edu.au/(2015.9.1).
8) Pitagora-Galaxy : http://www.pitagora-galaxy.org/(2015.9.1).
9) SINET: http://www.sinet.ad.jp/(2015.9.1).
10) Shigeo URUSHIDANI, Shunji ABE, Kenjiro YAMANAKA, Kento AIDA, Shigetoshi YOKOYAMA, Hiroshi YAMADA, Motonori NAKAMURA, Kensuke FUKUDA, Michihiro KOIBUCHI, and Shigeki YAMADA, New Directions for a Japanese Academic Backbone Network,IEICE TRANS. INF. & SYST. E-98-D(3) Mar 2015.
11) Mesos: http://mesos.apache.org/(2015.9.1)
12) Shigetoshi Yokoyama, Nobukazu Yoshioka, “On-demand Cloud Architecture for Academic Community Cloud”, 4th International Conference on cloud computing and Service Science, 2014.
13) Galaxy: https://galaxyproject.org/(2015.9.1)
14) Aurora: http://aurora.apache.org/(2015.9.1)
15) FANTOM5: http://fantom.gsc.riken.jp/5/(2015.9.1)
16) https://registry.hub.docker.com/u/nasuno/mesos-aurora/(2015.9.1)