3E-4

グリッド技術を用いたGIS処理の制御と効率化 *

草薙 信照 †
大阪経済大学　経営情報学部

石原 康秀 ‡
富士通株式会社
計算科学ソリューションセンター

1　グリッド技術と汎用アプリケーション

本研究の目的は，グリッド環境のために最適化された専用の応用プログラムを開発するのではなく，既存の汎用アプリケーションをグリッド環境下で効果的に運用することにある．そこで，グリッド技術の適用例として，汎用的なGISアプリケーションによる空間分析を実データで実行し，グリッドコンピューティングの有効性を制御方式と処理効率の両面から検証した．本検証には，グリッド制御用のミドルウェアとしてSystemwalker CyberGRIP（富士通社製，以下CyberGRIP）を，GISアプリケーションとしてArcGIS（Ver.9.1，ESRI社製）を用いた．

ArcGISは代表的な汎用GISアプリケーションであり，行政や企業の実業務でも広く利用されている．通常，ArcGISの処理は1台のPC上でインタラクティブに行われるが，多くの空間情報を扱ったり複雑な解析を行う場合には，バッチ型で実行されることもある．そのような処理には数十分から数時間を要することもあり，これをうまく分割して並行処理し，スループットを向上させることができれば，実用的な意義は大きい．

このような実業務での利用の場合，総合的な処理効率の向上だけでなく，自動的かつ安定的な実行と，障害発生時の適切な対処等の制御が重要な課題となる．

2　実験の概要

実験環境を図1に示した．複数のGIS処理の実行制御には，CyberGRIPのオーガニックジョブコントローラ（以下，OJC）を用いた．OJCとは大量のジョブの依存関係や待ち合わせを含めて定義する専用スクリプト環境のことで，スクリプトを解釈してジョブ投入や実行状況の監視を行う機能を有する．これにより，ジョブ管理が容易に行えるだけでなく，先行ジョブの結果に基づく後続ジョブの動的生成や部分的なジョブの再実行など，インテリジェントなジョブ実行制御が可能になっている．

実験に用いたArcGISによる処理の概要は以下のとおりである．

① 対象地域として大阪府南部の13市町（堺市～岬町）を選び，数値地図2500（空間データ基盤，大阪-4）から当該市町の道路・鉄道・主要建物に関するフィーチャ（Shapeファイル）を作成．

② 各市町に立地する鉄道駅を中心として，半径1kmの円形バッファエリアを作成．

③ ArcGISのClip操作により，①のフィーチャを②のバッファで切り取り，新たなフィーチャとして出力．

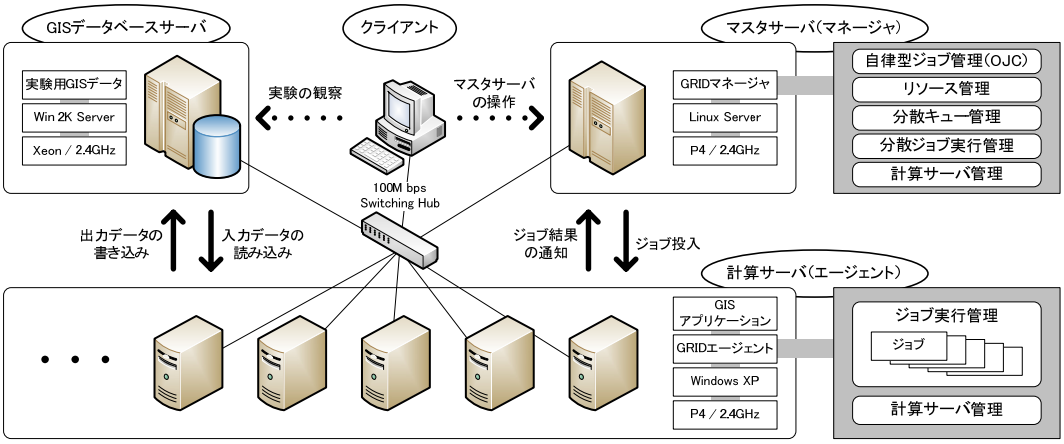

図1　実験環境

---

* Control Method and Efficient Operation of GIS Processing by Using Grid Technology.
† Nobuteru KUSANAGI; Faculty of Information Management, Osaka University of Economics
‡ Yasuhide ISHIHARA; Computational Science and Engineering Center, FUJITSU Limited

本研究は大阪経済大学・共同研究費（2005～2006年度，「グリッドコンピューティング技術を用いた大規模地域統計情報の処理に関する研究」）の交付を受けて行っており，本稿はその成果の一部として公表するものである．

1つの市町村に関する③の操作を1ジョブとして合計 13 個のジョブを用意した．各ジョブを 1 台の PC で直接実行したときの処理時間（計測 3 回の平均値）は図２のとおりで，１ジョブの最大処理時間は「堺市」の 1 分 21 秒，13 ジョブの合計処理時間は 9 分 06 秒である．

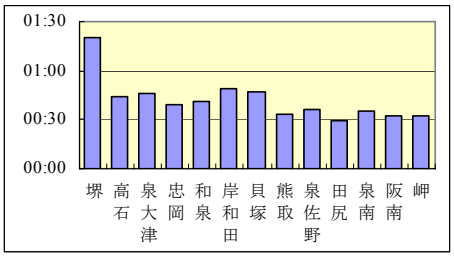

図２　各ジョブの処理時間

３　グリッド技術を用いた制御方式

CyberGRIP が提供する OJC スクリプトを用いれば，アプリケーションを直接制御することは可能であるが，アプリケーション固有の設定や呼び出し方式をラッピングするスクリプトを，OJC スクリプトのもとで採用することによって安定性が増す．ArcGIS が提供する機能の多くは Python スクリプトのもとで実行可能であることから，前述の 13 個のジョブを Python スクリプトで記述することとした．

まず，各ジョブが Windows 環境において動作することを確認し，それらのジョブを OJC スクリプト内に埋め込んで，１つずつ CyberGRIP 環境下でも問題なく動作することを確認した．

次に，13 個のジョブを CyberGRIP 環境下で連続投入したところ，処理が完了しないまま異常終了するジョブのあることが判明した．このエラーには再現性がなく，いまなお原因は不明のままである．そこで，OJC スクリプトにおいて，未完のジョブがあれば 10 回まで再投入（リトライ）するように制御アルゴリズムを変更したところ，すべてのジョブを完了できることが確認された．

４　実験結果と評価

エージェントの台数を 1 台から 13 台まで変化させながら，それぞれの処理時間とリトライ回数を集計した結果（いずれも計測 3 回の平均値）を図３及び図４に示した．

CyberGRIP 環境下で 13 個のジョブを 1 台のエージェントに連続投入したところ，処理時間は 8 分 54 秒となり，1 台の PC で直接実行したときの合計処理時間（9 分 06 秒）よりも短くなった．このとき，合計リトライ回数は 15.7 回であった．エージェントの台数を増やすにつれて，合計処理時間は短く，合計リトライ回数は少なくなり，13 台のケースでは合計処理時間が 1 分 23 秒，合計リトライ回数が 1.0 回となった．なお，すべてのケースにおいて未完のジョブは無い．

このように，Python スクリプトで記述された複数のジョブは，グリッド環境で問題なく分散処理が行われ，エージェント台数と合計処理時間の間には，理論どおりほぼ反比例に近い関係が見られることが確認された．また，アプリケーション実行時に未知のエラーが発生しても，グリッド側で適切に制御を行うことで，結果的にすべてのジョブを完了できることも確認された．

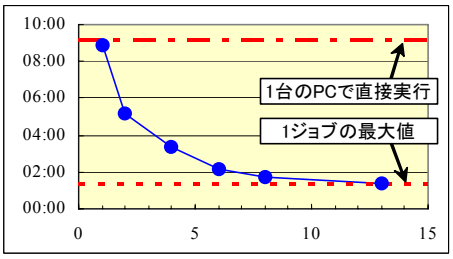

図３　エージェント台数と合計処理時間

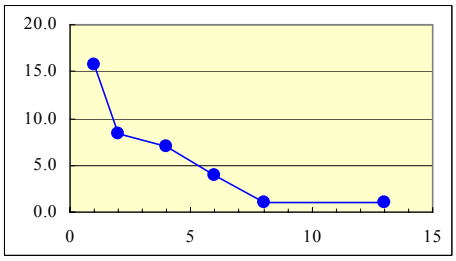

図４　エージェント台数とリトライ回数

５　考察と今後の課題

直接実行時には発生しなかったエラーが，グリッド環境下で発生した原因は不明であるが，アプリケーション側の要因である可能性が高い．エージェント PC が処理途中でダウンした場合，代替 PC でジョブを実行するグリッド環境は多いが，アプリケーション側の未知のエラーに対してもリトライを適切に実施し，確実にすべてのジョブを完了できる制御方式を検証することができた．

今回の実験のように，各ジョブの大きさ（処理時間）に差があることが予見される場合，大きいジョブから順に投入することは有効である．あるいは，エージェントの性能に差がある場合には，高性能なエージェントに大きなジョブを割り当てることも有効であろう．

GIS 処理におけるグリッド技術の実用性は必要十分な水準にあるといえる．さらに安定的で柔軟なジョブ制御機能と，動的なジョブ解析・自動実行機構が組み込まれれば，多くの実業務分野においてグリッド技術の実用性は飛躍的に高まるものと期待される．