様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　２ 年　 ６ 月 １７ 日現在

機関番号：２６４０２
研究種目：基盤研究(C)（一般）
研究期間：2017～2019
課題番号：１７Ｋ００１０７

研究課題名（和文）資源アクセス情報を利用したプログラム実行環境の移送方式

研究課題名（英文）Migration method of program execution environment using resource access information

研究代表者
横山　和俊（Yokoyama, Kazutoshi）

高知工科大学・情報学群・教授

研究者番号：５０５５９１３５

交付決定額（研究期間全体）：（直接経費）　　3,400,000円

研究成果の概要（和文）：クラウドコンピューティングの環境では，処理を適切に分散し，APが走行する計算機を変更し再配置することが頻繁に行われる．つまり，広域分散システム環境で，APの実行環境を移送する機能が重要となっている．本研究では，広域分散システムを対象としたAPの移送技術を確立した．確立した移送技術では，APの必要最小限のソフトウェア資源を，APの資源アクセス情報を利用し特定する．また，移送に伴うAPの停止時間を短縮するため，移送内容を分割して転送する．さらに，計算機間に跨る関連の深いAPグループを移送するため，socket通信を監視し，計算機間に跨るAPグループの特定を実現している．

研究成果の学術的意義や社会的意義
従来の計算機のソフトウェア資源をすべて移送していた問題点について，必要最小限の移送内容を特定し移送することが独創的である．特に，移ファイルの共有関係やアクセス種別（ReadOnlyやReadWriteなど)に着目し，一括して移送する必要があるプログラムファイル群とデータファイル群を追跡する点が独創的である．また，本研究では，移送内容を転送することに伴うAPの停止時間を短縮するため，ファイルサイズやファイルの更新頻度を用いて転送する順序を制御する点が独創的である．

研究成果の概要（英文）：In the cloud computing environment, the computer on which the AP runs is often changed and relocated. In other words, the function to migrate the execution environment of AP is important in the wide-area distributed system environment. In this research, we established the AP the migration method for wide-area distributed systems. With the proposed migration method, the minimum required software resources of the AP are specified by using the resource access information. In addition, in order to reduce the down time of the AP, the software resources to be migrated are divided and transferred. Furthermore, in order to migrate AP groups that are closely related to each other between computers, socket communication is monitored, and the AP groups that span computers are specified.

研究分野：分散処理

キーワード：クラウドコンピューティング　プログラム実行環境移送　プリコピー移送

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

クラウドコンピューティングの進展に伴い，多様な応用プログラム(以降，AP と略す)が計算機を共用しながらサービスを提供するシステム形態が普及している．近年では，保有する計算機が増加しているため，地理的に分散した広域分散システムとして実現されることが多い．これらのシステム形態では，処理を適切に分散し，計算機を効率良く利用することが強く求められるため，AP が走行する計算機を変更し再配置することが頻繁に行われる．つまり，広域分散システム環境で，AP の実行環境を移送する機能が重要となっている．

２．研究の目的

AP の実行環境を移送させる代表的な技術として，仮想マシンを移送する方法がある．しかし，広域分散システム環境での仮想マシンの移送は，仮想マシンのすべてのソフトウェア資源を移送先に転送する．つまり，移送したい AP のソフトウェア資源だけでなく，他の AP のソフトウェア資源も移送されてしまう．このため，余分なソフトウェア資源が移送されてしまい，移送に時間がかかる問題がある（問題 1）．この問題を解消するため，AP の実行に必要なソフトウェア資源を人手により記述することで，最小限のソフトウェア資源を他の計算機への移送しているものがある[1]．しかし，通常 AP は多数のソフトウェア資源を利用するため，人手によりすべてのソフトウェア資源を指定することは多くの工数がかかる（問題 2）．

本研究では，広域分散システムを対象とした AP の移送技術を確立する．問題 1 を解決するため，AP の必要最小限のソフトウェア資源（以降，移送内容と呼ぶ）を特定する技術を実現する．その実現にあたり，問題 2 を克服するため，AP の資源アクセス情報を利用し，移送内容を自動的に特定する方法を確立する．また，移送に伴う AP の停止時間を短縮するため，移送内容を分割して転送する技術を確立する．さらに，計算機間に跨る関連の深い AP 群（以降，AP グループと呼ぶ）を移送する技術へ拡張する．

３．研究の方法

本研究は，(ステップ 1)移送内容を自動的に特定する技術，実行停止時間を短縮する分割転送技術，(ステップ 2) AP グループを対象とした移送技術，の 2 段階で研究を行う．

(1) ステップ 1（計算機内に閉じた資源の移送技術）

主に，移送内容を自動的に特定する技術，実行停止時間を短縮する分割転送技術について技術を確立する．

(1-a) 移送内容を自動的に特定する技術

AP 単位で必要最小限な移送内容を自動的に特定する技術を研究する．具体的には，AP が発行するシステムコールを監視し，プログラムファイル，ライブラリファイル，データファイル，プロセス間通信など）を自動的に特定する技術を確立する．その際に，ファイルなどの共有関係に着目し，一緒に移送する必要がある資源を特定する．

(1-b) 実行停止時間を短縮する分割転送技術

AP が停止する時間を最小限にするため，移送内容を分割して転送する技術を研究する．この手法では，移送内容を一度に転送するのではなく，幾つかのグループに分けて転送し，最後のグループを転送する時だけ AP の実行を停止する．しかし，一度転送したファイルに対して更新が発生すると該当ファイルを再転送するため，ネットワークに負荷がかかる問題がある．この問題に対し，本研究では，ファイルの属性（ReadOnly，ReadWrite など）とファイルの更新頻度を用いて転送する順序を制御することで，再転送の発生を抑止する．例えば，更新頻度の高いファイルは再転送の可能性が高いためできるだけ後回しで転送し，ReadOnly ファイルや更新頻度の低いファイルは早い段階で転送する制御を行うことで，再送を抑制する．

(2) ステップ 2（計算機に跨る資源の移送技術）

複数の計算機を対象に，関連深い AP 群である AP グループを対象とした移送技術を研究する．本課題では，AP グループが使用しているソフトウェア資源を計算機に跨って特定し，移送先において，計算機に跨る同様な分散実行環境を再構築することを目指す．このために，(ステップ 1)の技術を拡張し，ネットワーク通信の状況を監視し，計算機を跨ってソフトウェア資源を追跡する機能を実現する．また，移送先での AP グループの動作環境を再構築するため，移送元システムと移送先システムのネットワーク通信の環境の差異を隠蔽する機能を実現する．

４．研究成果

(1) 計算機内に閉じた資源の移送技術

(1-a) 移送内容を自動的に特定する技術

転送対象となる AP のファイルへのアクセスを監視し，その AP が利用するファイル群を特定し転送対象とする．具体的には，AP が発行する open システムコールを十分な時間監視し，ファイル名とアクセス種別(Read-Only,Read-Write)を記録する．記録した情報から転送対象となるファイルを決定する際には，対象となるファイルを転送すると影響を受けるファイルとの共有関係を追跡し，影響を受けるファイルも含めたファイル群を特定する．ファイルの共有関係を考慮した追跡の様子を図 1 用いて説明する．図 1 に示すように，AP として AP1 と AP2 が実行されている．また，ファイル資源として，ファイル A，ファイル B，ファイル C，ファイル D がある．

ここで AP1 が利用するファイルは，ファイル A とファイル B であり，AP2 が利用するファイルは，ファイル A とファイル C である．すなわち，ファイル A が 2 つの AP によって共有されている．ここで AP1 を転送する場合を考える．AP1 がファイル A に対し，何らかの書き込み処理を行う場合,AP2 がその情報に依存して動作している場合がある．そのため AP1 とファイル A，ファイル B のみを移送した場合，AP2 が AP1 からの情報を受け取ることができず，意図しない動作をする可能性がある．この可能性を考慮し，AP1 を移送する場合，追跡プロセスが，AP1，ファイル A，ファイル B に加え，ファイル A を共有する AP2 と，AP2 が利用しているファイル C も特定し移送対象とする．

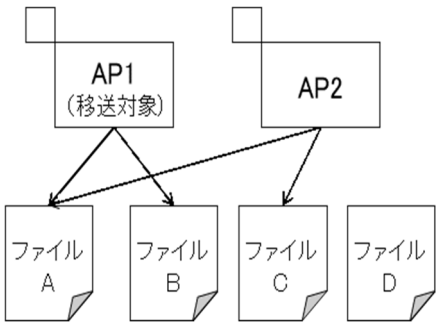

図 1　移送対象の追跡

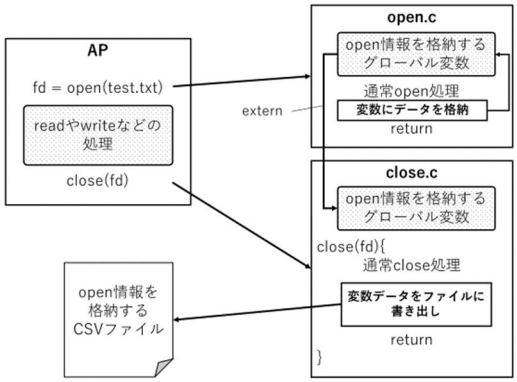

図 2　open システコールの監視

図 2 に open システコールの監視機構の実現方法を示す．また，提案する監視機構のオーバーヘッドの評価結果を図 3 に示す．図 3 より，提案する監視機構のオーバーヘッドは，十分に小さいことが分かる．

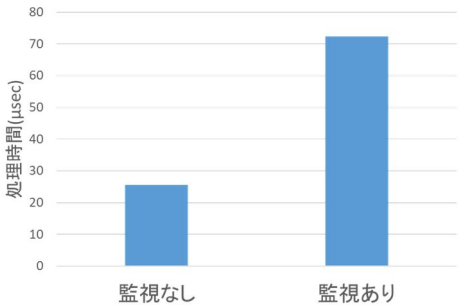

図 3　open システムコールの監視オーバーヘッド

(1-b) 実行停止時間を短縮する分割転送技術

(1-a)の技術で決定した移送対象のファイル群を転送する.このとき,移送対象をいくつかのグループに分割して転送する.本ステップでは，AP を動作させながら転送を行う「プリコピーフェーズ」と，AP を停止させて転送を行う「最終コピーフェーズ」の 2 段階のフェーズで全ファイルの転送を行う．これら 2 つのフェーズについて以下で説明する．

＜プリコピーフェーズ＞

プリコピーフェーズにおける具体的な転送例を図 4 に示す.図 4 では,ファイル群を，ある転送順序に基づいて，グループ A からグループ J までの 10 個のグループに分割している．プリコピーフェーズでは，このグループ単位で逐次転送する．プリコピーフェーズは AP を動作させながらファイルの転送を行うため，転送されたファイルに対して,再度更新処理が発生する場合がある．その場合は，更新された該当ファイルを転送対象に戻し，再送する．

＜最終コピーフェーズ＞

最終コピーフェーズにおける転送例を図 5 に示す．図 5 は，プリコピーフェーズの終了直後である．図 5 の網掛けで表わしているファイルは，プリコピーフェーズで一度転送されたものの，その後更新処理が発生し，転送対象に戻っているファイルである．最終コピーフェーズでは，これらの転送対象のファイルを全て，AP を停止して転送する．つまり，AP のサービス停止時間を短縮するためには，再送されるファイルを少なくし，最終コピー量を削減することが必要である．

実際の AP としてカーネル make を用いて，転送順序による転送結果の違いと提案手法の有効性をシミュレーションにより評価する．転送速度は 5Mbps を想定し，Write システムコール 50 回をプリコピー1 周期として，プリコピー500 周期で最終コピーフェーズに移行するものとする．これを下記の 3 種類の転送順序で評価する．

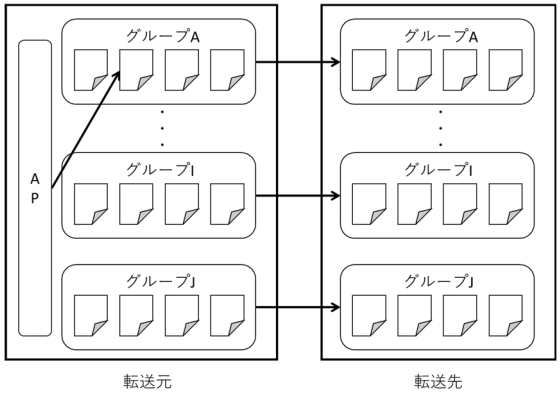

図 4　プリコピーフェーズ

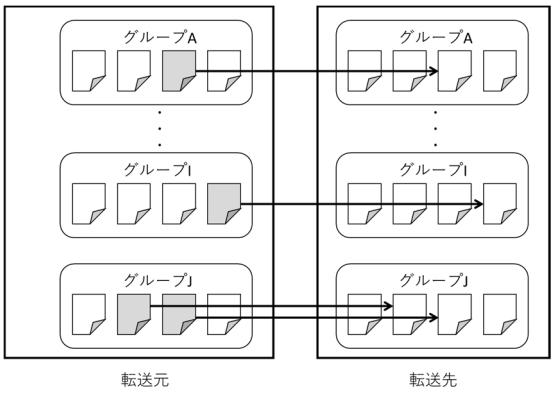

図 5　最終コピーフェーズ

・ファイルサイズの降順(サイズ降順と略す)
・ファイルサイズの昇順(サイズ昇順と略す)
・アクセスされた回数の昇順(アクセス頻度と略す)

評価結果を図 6 に示す．結果としては，サイズ降順の最終コピー量が 482MB，サイズ昇順が 211MB，アクセス頻度が 118MB となった．また，全てのファイルの総サイズに対する最終コピー量の削減率は，アクセス頻度が約 64%，サイズ昇順が約 56%，サイズ降順が約 0%となった．提案手法は，全容量の 482MB に対し最も削減できた場合の最終コピー量が 118MB となっており，約 64%の削減に成功しており，提案手法の有効性が確認できた．

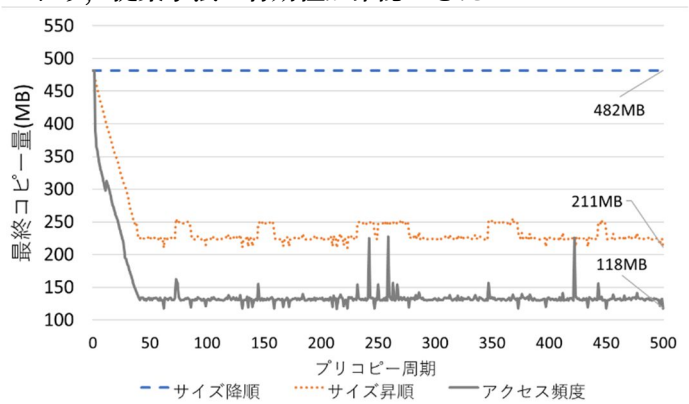

図 6　分割転送技術の評価結果

(2) 計算機に跨る資源の移送技術

計算機を跨る資源の追跡を可能につるため socket 通信を監視する．具体的には，accept/connect の各システムコールを監視して，どの AP がどのファイルを利用しているのかを全て特定し，実行環境を追跡する． 例を図 7 に示す．socket 通信は socket を始めとする，いくつかのシステムコールの組み合わせにより行われる．この時，送信側では connect，受信側では accept を利用する．accept/connect システムコールの定義を以下に示す．

```
int accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
int connect(int sockfd, const struct sockaddr *addr, socklen_t addrlen);
```

これらのシステムコールを監視し，どの AP 間で通信が発生しているかを追跡する．この追跡は，以下の 2 段階で行われる．

(a) 通信を行っているプロセスの特定

上記の 2 つのシステムコールには，sockaddr 構造体(図中では addr)を持ち，その中にはそれぞれ接続先の IP アドレスとポート番号が格納されている．システムコールライブラリ内で sockaddr 構造体から取得した情報を，特定の場所に保存しておく．具体的には，クライアント側は自分の PID と IP アドレス，サーバの IP アドレスとポート番号を記録する．サーバ側は，自分の PID と IP アドレス，接続してきたクライアントの IP アドレスとポート番号を記録する．この IP アドレスを突き合わせることで，通信を行っているプロセスを特定する．

(b) プロセスが実行している AP の特定

アクセスを行った AP を特定するために，プロセス生成時(execve システムコール発行時)に，PID と実行される AP のパスの組み合わせを特定の場所に記録しておく．その記録と PID を対応付け，AP のパスを取得する．

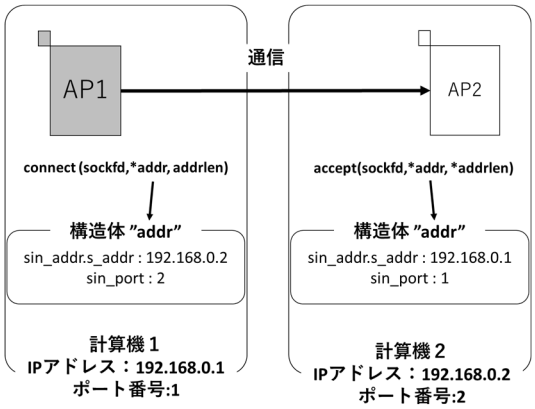

図 7　socket 通信の例

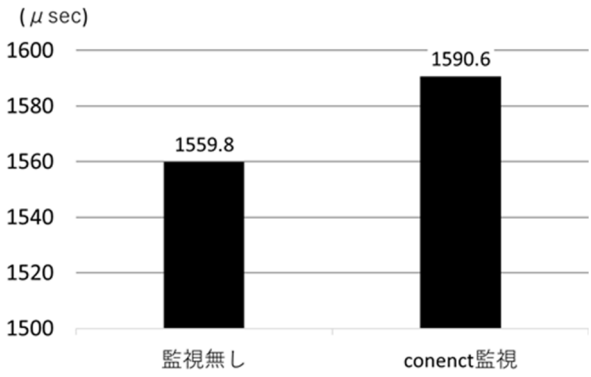

図 8　connect 監視オーバーヘッド

connect 監視のオーバーヘッドを図 8 に示す．監視なしの場合，プログラム実行時間は約 1560 マイクロ秒となった．監視時は，約 1591 マイクロ秒となり，監視なしの場合と比較して約 2% オーバーヘッドがあった．また，accept システムコールでの監視オーバーヘッドの増加は約 35 マイクロ秒であった．

５．主な発表論文等

〔雑誌論文〕　計0件

〔学会発表〕　計8件（うち招待講演　0件／うち国際学会　0件）

１．発表者名
黒木勇作，大西史洋，横山和俊，谷口秀夫

２．発表標題
サービスの停止時間を短縮するプログラム実行環境のプリコピー移送手法

３．学会等名
情報処理学会第175回マルチメディア通信と分散処理研究発表会

４．発表年
2018年

１．発表者名
澤田優真，黒木勇作，横山和俊，谷口秀夫

２．発表標題
ソフトウェア実行環境のプリコピー移送方式におけるサービス停止時間の評価

３．学会等名
平成30年度電気関係学会四国支部連合大会

４．発表年
2018年

１．発表者名
黒木勇作，西拓人，横山和俊，谷口秀夫

２．発表標題
複数計算機上に跨るプログラム実行環境の特定手法の提案

３．学会等名
情報処理学会第81回全国大会

４．発表年
2019年

１．発表者名
黒木勇作，畑翔太，横山和俊

２．発表標題
ファイルの更新期待値を用いたソフトウェア実行環境のマイグレーション方式

３．学会等名
情報処理学会 第25回マルチメディア通信と分散処理ワークショップ

４．発表年
2017年

１．発表者名
黒木勇作，大西史洋，横山和俊，谷口秀夫

２．発表標題
ファイルサイズとアクセス頻度を用いたプログラム実行環境の移行手法

３．学会等名
情報処理学会 第80回全国大会

４．発表年
2018年

１．発表者名
大西史洋，黒木勇作，横山和俊，谷口秀夫

２．発表標題
プログラム実行環境移送のための資源追跡機能のユーザレベルでの実現

３．学会等名
情報処理学会 第80回全国大会

４．発表年
2018年

１．発表者名
有薗里奈，澤田優真，黒木勇作，横山和俊，谷口秀夫

２．発表標題
サービスの停止時間を短縮するプログラム実行環境のプリコピー移送手法の評価

３．学会等名
第27回マルチメディア通信と分散処理ワークショップ

４．発表年
2019年

１．発表者名
黒木勇作，横山和俊，谷口秀夫

２．発表標題
複数計算機上に跨るプログラム実行環境の特定手法の評価

３．学会等名
情報処理学会第82回全国大会

４．発表年
2020年

〔図書〕　計0件

〔産業財産権〕

〔その他〕

―

６．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考
研究分担者	谷口　秀夫 (Taniguchi Hideo) (70253507)	岡山大学・自然科学研究科・教授 (15301)	