5R-07

エクスターナルグリッドに対する依存関係を利用した不正解析のリスクを軽減する手法

山口 晃右 * 稲元 勉 † 樋上 喜信 ‡ 小林 真也 §

愛媛大学工学部情報工学科 * 愛媛大学大学院理工学研究科 † ‡ §

1 研究背景

グリッドコンピューティングは, ネットワーク上の多数の計算機を利用して, 高い処理能力や記憶容量を安価に得るための技術である. グリッドコンピューティングの一つに, インターネット上の計算機を利用するエクスターナルグリッド (以降, 単にグリッドと呼ぶ) がある. インターネット上のコンピュータを利用するので, 実質的に無制限に計算資源を獲得することができる. また, 高性能な計算機を用いることなく, 安価に高い処理性能を得ることができる. しかし, プログラムが「正しく処理されるという保証」や「情報漏洩対策」等の安全性の保証が十分でないことが課題である. 本稿では, グリッドの安全性を向上を目指し, 共謀した悪人集団による, 分割されたプログラム間の依存関係を利用した不正解析のリスクを軽減する手法を提案し, その性能の評価を行う.

2 セキュアプロセッシング

グリッド上で実際に計算をする計算機を, 処理ノードと呼ぶ. また, 不正行為を働く処理ノードを, 悪人と呼ぶ. セキュアプロセッシングは, グリッド上の悪人が行う不正行為への対策のための技術の集合である. 不正行為には, 不正な解析と改竄がある. 不正な解析は, 悪人が, 渡されたデータを不正に利用し, 元のプログラムを解析することである. 改竄は, 悪人が本来の正しい処理結果と異なった結果を返すことである. 以下に, セキュアプロセッシングの技術の一部を示す.

プログラム分割 グリッドに依頼されたプログラムを分割し, プログラム断片を生成し, 各処理ノードに処理させる技術である. 不正な解析に対しても効果がある.

処理の多重化 改竄対策の一つで, 同一のプログラム断片を複数の処理ノードに渡すことで, 処理結果の信頼性を高める技術である. 同じ種類のプログラム断片を処理する処理ノードの台数を多重度と呼ぶ.

3 先行処理

処理の多重化では, 同一のプログラム断片の処理時間に, 処理ノード間で差が現れる. 同一プログラム断片を処理する処理ノードの中で, 最も速く処理するノードの処理結果を用い, 先行して処理を進めることで, 全体の高速化を行える. これを, 先行処理という. ただし, 先行処理をするノードの処理の結果が, 正しくない場合がある. このような場合は, 先行していた処理は無効となる. そこで, 同一のプログラム断片を処理する処理ノードの出す結果毎に, 網羅的に先行処理を行うことで, 再処理する時間を短縮し, より高速化ができる. これを網羅法という. 網羅法は, 安定してグリッドを高速化できる反面, 利用するノード数が膨大になるという欠点がある.[1]

4 悪人集団の共謀による不正解析の問題

プログラム分割で得られた断片間には依存関係が存在する. 依存関係で繋がったプログラム断片を連続したプログラム断片と呼び, 連続する長さを連続長と呼ぶ. 悪人は, この依存関係を利用して不正な解析を行うと考えられる. また, 悪人がグループを形成し, 各悪人が取得したプログラム断片を持ち寄ることで, 悪人グループが取得できる連続長が大きくなり, 不正な解析のリスクがより高まると考えられる.

5 不正な解析に対する対策

不正解析に対し, プログラム断片の一部を, 悪人ではないことが保証された信頼できる処理ノードに処理を依頼することで, 悪人が取得できる連続長を短くすることができる. 信頼できる処理ノードでプログラム断片を処理することを, 保護処理と呼ぶ. また, 保護処理されるプログラム断片を被保護断片と呼ぶ. 被保護断片を, 連続したプログラム断片を等分する位置に挿入する. 例えば, 連続したプログラム断片の長さを N, 被保護断片数を L とする. このとき, 連続したプログラム断片は $L+1$ 個の区間に分割され, 各区間の長さは,$\lceil (N-L)/(L+1) \rceil$ と $\lfloor (N-L)/(L+1) \rfloor$ の高々二種類しか存在しない.

6 評価方法

提案手法の性能評価を, 以下の評価項目に基づいて行う. 各評価項目を実際に評価する際には, 提案手法の有無による比較と, 被保護断片数の異なる場合同士を比較する.

1. 悪人集団による不正な解析に対して, どの程度の効果があるか
2. グリッドの処理ノードの総利用数に, どの程度の影響を与えるか
3. グリッドの処理時間に, どの程度影響を与えるか

評価項目 1 においては, 悪人集団が取得する連続長に注目し, 解析的手法や, シミュレーションによって, 評価を行う. 解析的手法を取る際に [2] で導出された式を利用する.

評価項目 2 については, 被保護断片数の違いにより, エクスターナルグリッド全体で, プログラムを実行する際に, 実際に利用される処理ノード数にどのような違いが出るかを, シミュレーションにより評価する. エクスターナルグリッドは, 処理ノードを無制限に利用できるとみなしているが, 不正な解析の危険性を鑑みると, 処理ノードの利用数は少ない方が良い.

同様に, 評価項目 3 においても, 被保護断片数の違いが, 処理時間にどの程度影響するかを評価する.

シミュレーション条件は以下の通りである.

- グリッドを構成する処理ノードは, 無数に存在する.
- 処理ノードに含まれる悪人を識別することはできず, 悪人は, 存在確率に応じて, グリッド中に存在する.
- 各処理ノードの処理性能は, 各処理ノードが単位時間当たりに処理できるプログラムのサイズであるとし, 形状尺度 $k=5$, 尺度分母 $\Theta=2/5$, 期待値 2 となるガンマ分布に従う.
- 入力された同一条件での試行を 1000 回行う.

7 評価結果と考察

7.1 悪人が取得する連続長

網羅法を用いないグリッドで, 連続したプログラム断片のうち, 悪人集団が取得できる最大連続長が k 以上となる確率を, 解析的に求めた. 図 1 の, 横軸は k, 縦軸は, 悪人集団が取得できる連続長が k 以上となる確率である. また, 多重度は 20,

A method to reduce the risk of malocious analysis with dependence relations in external grid computing

* Khosuke Yamaguchi, Department of Computer Science in Engineering, Ehime University

† Tsutomu Inamoto, ‡ Yoshinobu Higami, § Shinya Kobayasi, Graduate School of Science and Engineering, Ehime University

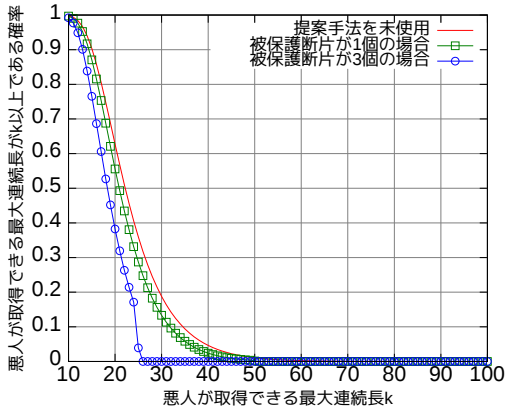

図 1: 悪人が取得する最大連続長が k 以上である確率

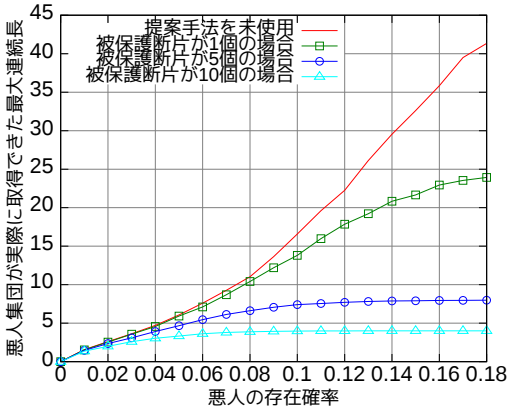

図 2: 悪人が取得できる最大連続長の平均

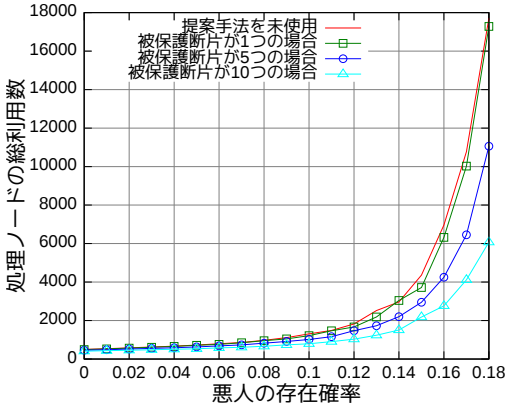

図 3: 処理ノードの総利用数の平均

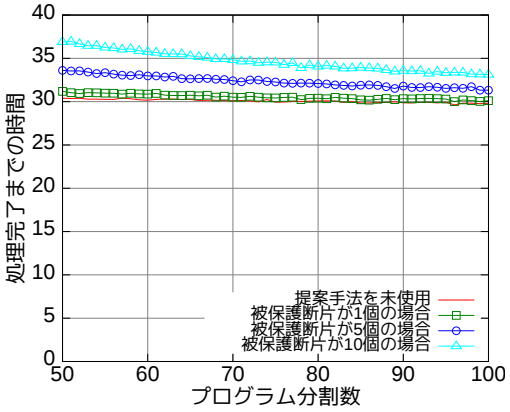

図 4: 最終的な処理完了時間の平均

悪人の存在確率は 0.1 である. 図より, 被保護断片を設ける提案手法は, 被保護断片を設けない場合と比べ, 悪人集団が取得できる最大連続長が k 以上となる確率が低いことがわかる. また, 被保護断片の数が多いほど, 提案手法の効果が高くなっていることがわかる.

次に, 網羅法を用いた場合に, 悪人集団が実際に取得した最大連続長を, シミュレーションにより求めた. 図 2 の, 横軸は, 悪人の存在確率, 縦軸は, 悪人集団が実際に取得できた最大連続長である. また, 分割数は 50 で, 多重度は 10 である. 図 2 より, 提案手法を使用した方が, また, 被保護断片数が多いほど, 悪人集団が取得できた最大連続長が減少していることがわかる. 存在確率が 0.18 の場合, 被保護断片数 10 とすると, 提案手法を用いないものと比べ, 最大連続長を 1/10 以下にできる.

7.2 処理ノードの総利用数

網羅法を使用した際に, グリッドに依頼された処理が終了するまでに利用した処理ノードの総数を, シミュレーションにより求めた. 図 3 に, 悪人の存在確率と, 処理ノードの総利用数の関係を示す. 分割数は 50, 多重度は 10 である. 図 3 から, 提案手法を用いている場合の方が, 処理ノードの総利用数が抑えられ, 被保護断片の数が多い方が, より利用数を抑えられていることがわかる. 存在確率が 0.18 の場合, 被保護断片数 10 においては, 提案手法を用いないものに比べ, 総利用数を約 1/3 に抑えている.

7.3 処理時間

シミュレーションによって, 依頼された処理が完了するまでの処理時間を求めた. 図 4 の横軸は, プログラム分割数, 縦軸は, 処理完了までの時間である. また, 多重度は 10 で, 悪人の存在確率は 0.1 である. 提案手法を未使用の場合の方が, より処理時間が短く, また, 被保護断片の数が大きいほど, より処理時間が長くなっている. 被保護断片を信頼できる処理ノードで処理するため, 高速な処理ノードでの実行を期待できる投機的効果を持つ多重化の特性を活かせないことが原因と考えられる. しかし, 分割数 100 の場合, 被保護断片数 10 とすると, 提案手法を使用しない場合と比べ, 処理時間は 10%程度しか伸びていない. また, 分割数が多いほど, 差が小さい.

8 結論

本稿では, グリッドにおける, 悪人集団による不正な解析のリスクを軽減するために, 信頼できる処理ノードに処理の一部を実行する方法を提案した. また, 提案手法の性能を定量的に評価した. その結果, プログラム分割数が 100 で, 被保護断片数が 10 の場合, 提案手法を用いることで, 処理時間が約 10%伸びてしまう. しかし, 悪人の存在確率が増加する程, 利用した処理ノードの総数を抑制できる効果がある. 更に, 悪人の存在確率が 0.1 で, 被保護断片数が 10 の場合, 提案手法を用いることで悪人が取得する最大連続長が 1/4 に抑えられ, 悪人が取得する最大連続長を減少させた. 提案手法は, 処理時間を伸ばすが, それ以上に, 処理ノードの利用数の抑制や不正な解析のリスクの軽減といった恩恵を与える.

謝辞

本研究は JSPS 科研費 26234567 の助成を受けたものです.

参考文献

[1] 広瀬 吉隆, 稲元 勉, 樋上 喜信, 小林 真也: “セキュアプロセッシングにおける先行処理による処理時間改善に対する定量的評価”, 第 14 回情報科学技術フォーラム (FIT2015) 講演論文集, Vol. 4, pp. 241-242, 2015

[2] 中矢 匠, 稲元 勉, 樋上 喜信, 小林 真也: “プログラム断片の連続性に基づくセキュアプロセッシングの秘匿性に関する調査”, DICOMO(Multimedia, Distributed, Cooperative, and Mobile System) 2015, pp.287-294, 2015