5H-03

FPGA NIC とソフトウェアによる協調型重複排除アーキテクチャ

鈴木 滉司† 松谷 宏紀‡
慶應義塾大学大学院理工学研究科

1. はじめに

昨今の加速的なビジネスデータの増大により、ストレージにかかるコストも増加の傾向にある。こういった背景の中で注目を集める技術の 1 つが重複排除技術である。データの総量（サイズ）を圧縮することができる重複排除は、大容量のデータを保管するためのストレージの容量単価を削減し、コストの増加を抑える効果を期待されている。ただし、重複排除処理の方式によってはストレージやデータ送信者に大きな負荷やコストが要求される。

FSC（Fixed-Size Chunking）はデータの重複を検出する際に用いられる手法の 1 つである。データを固定長のチャンクに分割し、チャンク単位での比較・重複検出を実行する。単純ゆえに高速に動作するものの、チャンクサイズは事前に設定されたサイズで固定となる。そのためチャンクサイズが取り扱うデータセットに適してない場合、チャンクの末尾・先頭の極一部が異なるために重複と見なされず、圧縮率の低下を引き起こすこともある。

本論文では FPGA(Field Programmable Gate Array)ベースの NIC(Network Interface Card)上にて高速に FSC 処理を行いつつ、ソフトウェアを併用することで最も適切であると思われるチャンクサイズを逐次的に変更・選択し続ける重複検出システムを提案する。重複排除処理の一部である重複検出の機能を FPGA NIC にオフロードすることで、後続のストレージにかかる負荷を一部軽減することが期待できる。

2. 関連技術

重複排除とは、バックアップ等の際に対象データを解析し、重複データを自動的に検出して排除する技術である。データの重複をなくすことで、バックアップする際のデータ転送量や格納容量を大幅に削減することが可能となる。重複排除の粒度としては、ファイルレベルに対するブロックレベルのようにデータをより小さなチャンク単位に分割した上で重複排除を行うことでより効果的な圧縮が可能となる。

チャンク作成の手法は固定長の FSC と可変長の CDC(Content-Defined Chunking)に大別され、前者は計算量が少なく低コストである。後者は重複をより確実に発見するために Rabin fingerprint[1]等の高度なアルゴリズムを利用する。

重複排除の機能をストレージへ実装する場合、ストレージへのデータ転送フローのどの場面で重複排除を実行するか、3 つのタイミングが存在する。すなわち、ストレージへデータを送るクライアント側で実行するプリプロセス、ストレージに書き込むまでの転送中に実行するインライン、一度重複も含めて全てのデータを書き込んでから実行するポストプロセスである。

インラインでの重複排除システムには、FPGA ベースのハードウェアを用いるものもある。特に FSC は処理が単純であるため、FPGA 等のハードウェアを用いた高速化との相性が良い。ただし、チャンクサイズは事前に設定されたものであるため、取り扱うデータにとって最適なサイズであるとは限らない。CDC のようにデータを意味を持つ形で過不足なく分割することができれば、より効率的な重複排除が実現できる。

3. 提案

本研究では、FPGA NIC 上での高速な重複検出と、ソフトウェアがもつ処理や演算の柔軟さを組み合わせることで、より柔軟なインライン方式の重複検出システムを実現することを提案する。システムの概要を図 1 に示す。

具体的には、重複排除のためのハッシュテーブルを 10GbE FPGA NIC 上にオフロードし、専用回路として実現する。重複検出の際には FSC を使用するが、チャンクの長さを連携して動作するソフトウェアが制御することで従来の FSC よりも効率的な重複排除を目指す。一度処理したチャンクはこのハッシュテーブルに記憶されるため、同じチャンクが発見されたとしてもそれを検出し、重複か否かを識別できる。

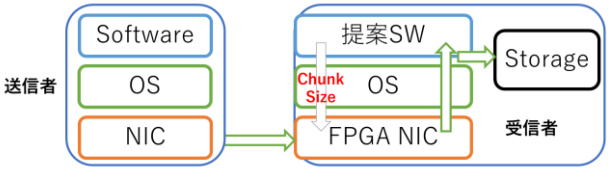

図 1 提案システムにおけるデータの流れ

A Deduplication Architecture with FPGA NIC and Software Co-design
† Koji Suzuki, Keio University
‡ Hiroki Matsutani, Keio University

重複検出を NIC で実行することにより、ストレージ側は重複に対しては該当する既存データへのアクセス情報へと置換する必要があるためストレージ探索・比較が依然必要となるが、新規のデータに対しては重複検出のためのストレージ探索・比較をする必要がなくなる。よってストレージ側で実行される処理の負荷を軽減することが期待できる。

4. 実装

本論文では 10GbE インターフェイスを有する FPGA ボードとして NetFPGA-SUME [2]を採用した。NetFPGA-SUME ではリモートマシンからのパケットは 4 つの 10GbE インターフェイスを、ホストマシンからのパケットは PCI-Express Gen3 x8 インターフェイスを介して FPGA に渡される。

本論文ではハッシュテーブルは簡易的に Block RAM として実装している。テーブルのアドレス幅は 15bit、データ幅は本実装におけるチャンクサイズの最大値と同じ 512bit とした。

重複排除コアモジュールではパケットのペイロードに対して FSC を実行し、各チャンクのハッシュ値を利用して重複検出を行う。その結果を 256bit のビットマップとして表現し、当該パケットの末尾に添付する。ビットマップは 16bit のフィールド 16 個で構成されており、第 1 フィールドにはそのパケットのチャンキングに用いたチャンクサイズを、残りのフィールドには重複であると判定されたチャンクの開始位置を記述する。そのため、本実装では最大 15 個の重複を後続へと伝達できる。

ホストマシンの CPU で動作する提案ソフトウェアは、受け取ったパケットの中身から、パケット内のチャンクの数及びその長さを可能な限り正確に把握する。パケットを一定数受け取るまでこれを繰り返したのち、得られた情報を集計し、新たなチャンクサイズを決定する。その後、NIC に命令を送りチャンクサイズの変更を反映させる。

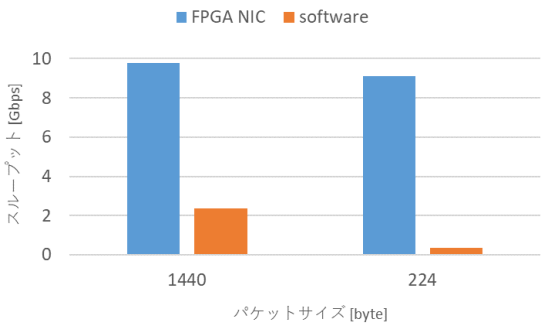

図 2　パケット処理のスループット比較

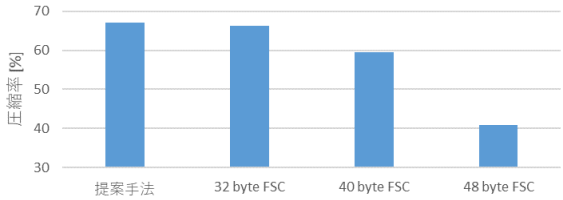

図 3　提案手法と FSC の圧縮率

5. 評価・考察

パケットジェネレータを用いた負荷実験の結果を図 2 に示す。図中では単純比較のためにソフトウェアの処理スループットをパケット受信機能のみのスループットとしている。実際には受信パケットへの解析・チャンキング処理（パケットサイズに依存）、比較処理が追加されるため更に性能が落ちる可能性が高い。

ソフトウェアの受信スループットはパケットサイズに比例しているうえ、受信の時点で既に性能が最大 3Gbps に満たないことが分かる。これに対し、FPGA NIC は重複検出を実行したうえで 10GbE の限界に近い性能を発揮している。

続いて重複排除率圧縮率）を図 3 に示す。テストデータには Linux カーネルソースのリストを使用し、チャンクサイズの範囲は NetFPGA-SUME の仕様に合わせた 32～64byte としたうえでシミュレーションを行った。FSC はチャンクサイズが小さいほど高い圧縮率が得られる。一方提案手法ではチャンクサイズを変更する度に既存データが重複検出に使えなくなるため圧縮率が一時的に大きく低下する。しかし今回の実験では最終的な圧縮率は提案手法の圧縮率が 32byte FSC の圧縮率を僅かに上回る結果となった。なお、常に事前に最適なチャンクサイズを得られると仮定した場合の圧縮率は 7 割を超える。チャンクサイズ変更を実行する閾値や間隔には改善の余地があると考えられる。

6. おわりに

本研究では FPGA NIC 上での高速な FSC 処理と、ソフトウェアによるチャンクサイズ選択を組み合わせた重複検出システムを提案した。チャンクサイズの変更は重複検出の上では大きなリスクとなるが、サイズ変更を適切に実行することで、実装上の最小サイズである 32byte での FSC を僅かに上回る圧縮率を実現した。

参考文献

[1] Rabin, M. O.: Fingerprinting by random polynomials. Center for Research in Computing Techn., Aiken Computation Laboratory, Univ., 1981.
[2] “The NetFPGA Project”. http://netfpga.org/.