環境適応アプリケーション配置適正化の初期検討

Initial study of environment-adaptive application placement optimization

山登庸次
Yoji Yamato

日本電信電話（株）　ネットワークサービスシステム研究所
Network Service Systems Laboratories, NTT Corporation

１．まえがき

近年，IoT[1]-[4]，AI 等新たな領域で，GPU や FPGA 等のヘテロハードウェアの利用が増えているが，それらの最大限活用には技術的壁が高い．私は，一度記述したコードを，配置先環境ハードウェアに合わせて，変換，リソース設定，配置等を自動で行い，高性能に動作させる，環境適応ソフトウェアを提案している[5]．GPU や FPGA へのコードの自動変換は等に取組んできた．本稿では，環境適応ソフトウェアの新たな要素として，変換したアプリケーションの，NW 上の適正配置に取り組む．

リソースの NW 上配置に関しては，仮想 NW 収容サーバを NW 上に適正配置する研究がある．しかし，狙いは，単一リソースの仮想 NW の最適配置によるキャリア設備コストや全体的応答時間削減であり，変換され個々のユーザ毎に異なるアプリケーションの処理時間や，個々のユーザのコストや応答時間要求等は考慮されていない．

本稿では，GPU 等に配置できるように自動変換したアプリケーションを，ユーザの要求を満たして，コスト，応答時間等を低減する適正配置手法を初期検討する．

２．配置考慮事項の整理

従来，アプリケーションはクラウド[6][7]に配置され，IoT デバイス等の収集データはクラウドに転送され，分析されていた．しかし，現在エッジコンピューティングとして，リアルタイム性が必要な処理はエッジで処理し，品質を上げる試みが現れている．本稿も，ネットワークエッジやユーザエッジにも配置できる前提とする．ただし，エッジは，クラウドに比べサーバが分散しているため，計算リソースのコストはクラウドに比べ割高となる．

計算ノードは CPU，GPU，FPGA の 3 種に分けられる．GPU や FPGA を備えるノードには CPU も搭載されているが，仮想化技術により，GPU，FPGA インスタンスとして，CPU リソースも含む形で分割して提供される．

アプリケーションを配置する際に，ユーザは 2 種類のリクエストを発出できる．一種類目は，コスト要求であり，アプリケーションを動作させるために許容できるコストを指定する形である．二種類目は，応答時間要求であり，動作させる際の許容応答時間を指定する形である．

既存研究では，設備設計の一環として，仮想 NW を収容するサーバの配置場所を，トラフィック増加量等の長期的傾向を見て，計画的に時間をかけ設計している．

それに対し本稿では 2 つ特徴がある．一つ目は，配置されるアプリケーションは静的に定まっているのではなく，GPU や FPGA 向けに自動変換され，進化計算等を通じて利用形態に適したオフロードパターンが実測を通じて抽出されるため，コードや性能は動的に変わる（例えば，同じ FFT でも，ユーザ A と B でデータサイズが異なる場合に，GPU オフロードループ文が異なったり，A には 10 倍性能だが B には 5 倍性能だったりする）．二つ目は，キャリア設備コストだけを低減すれば良いのでなく，コストや応答時間に対するユーザ要求を満たす必要があり，アプリケーションの配置ポリシーも動的に変わる．

3．アプリケーション適正配置の初期検討

前記の特徴も踏まえ，アプリケーション配置は，ユーザからの依頼があったら，コードの変換を行い，変換アプリケーションの性能とコストに応じてその時点で適切なサーバに順次配置していく．変換しても，コストパフォーマンスが向上しない場合は変換前アプリケーションを配置する．既に上限まで計算リソースや帯域が使われてしまっている場合はそのサーバには配置はできない．

適切なアプリケーション配置場所を計算するため，線形計画手法を用いる．ユーザ要求が，コスト要求か応答時間要求かで，目的関数と制約条件が変わる．コスト要求の場合，応答時間の最小化が目的関数となり，コストが幾ら以内は制約条件の一つとなり，サーバの計算リソース，帯域上限を超えていないかの制約条件も加わる．応答時間要求の場合は，目的関数が逆となる．

線形計画の定式化を行い，NW トポロジーやアプリケーションタイプ，ユーザ要求，既配置状況等の条件に対して，解を導出することで，適切な配置を計算する．

4．まとめ

環境適応ソフトウェアの適正配置手法を初期検討した．定式化を行い，種々条件に対し GLPK 等のソルバで解く．

参考文献

[1] Y. Yamato, et al., "Security Camera Movie and ERP Data Matching System to Prevent Theft," IEEE CCNC 2017, pp.1021-1022, Jan. 2017.
[2] Y. Yamato, "Experiments of posture estimation on vehicles using wearable acceleration sensors," IEEE BigDataSecurity 2017, pp.14-17, May 2017.
[3] Y. Yamato, et al., "Analyzing Machine Noise for Real Time Maintenance," ICGIP 2016, Oct. 2016.
[4] Y. Yamato, "Proposal of Vital Data Analysis Platform using Wearable Sensor," ICIAE 2017, Mar. 2017.
[5] Y. Yamato, "Study of parallel processing area extraction and data transfer number reduction for automatic GPU offloading of IoT applications," Journal of Intelligent Information Systems, Springer, 2019.
[6] Y. Yamato, et al., "Fast and Reliable Restoration Method of Virtual Resources on OpenStack," IEEE Transactions on Cloud Computing, Sep. 2015.
[7] Y. Yamato, "Use case study of HDD-SSD hybrid storage, distributed storage and HDD storage on OpenStack," IDEAS'15, pp.228-229, July 2015.