スカラ置換に基づく分岐発散の低減

福原淳司　滝本宗宏
東京理科大学　理工学研究科　情報科学専攻

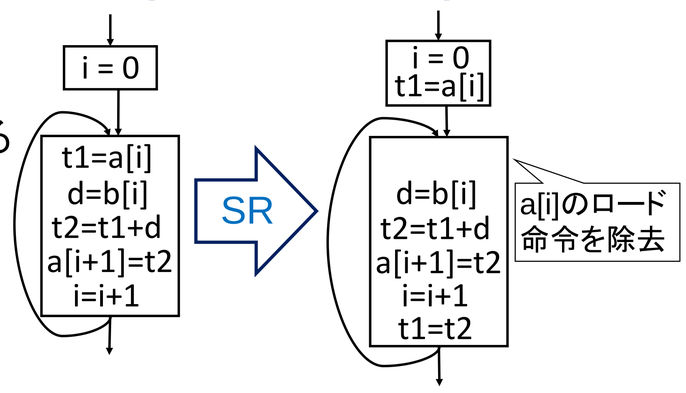
背景

- GPUプログラミングの普及
- GPUの多くはSIMD型の実行形式

GPUの実行効率を低下させる分岐発散が生じる可能性がある

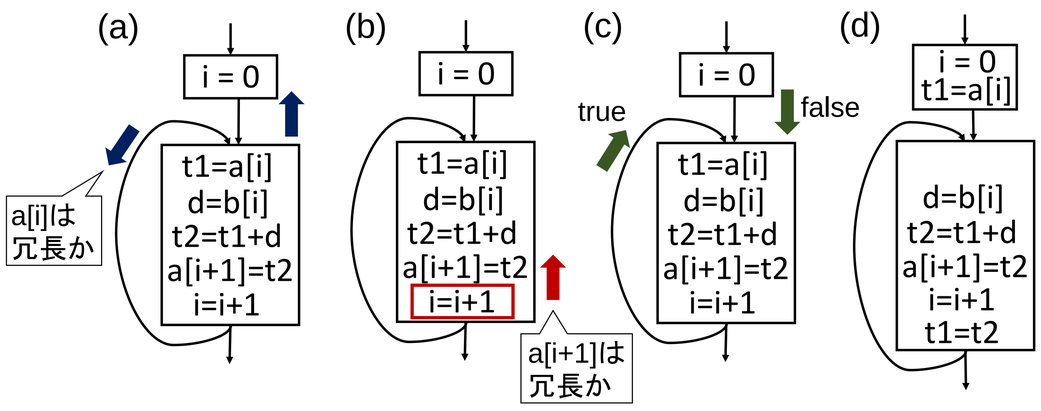
スカラ置換(Scalar Replacement)

ループの繰返しを超えて
冗長となる配列参照を
レジスタ参照に置き換える
手法



質問伝播(Question Propagation)

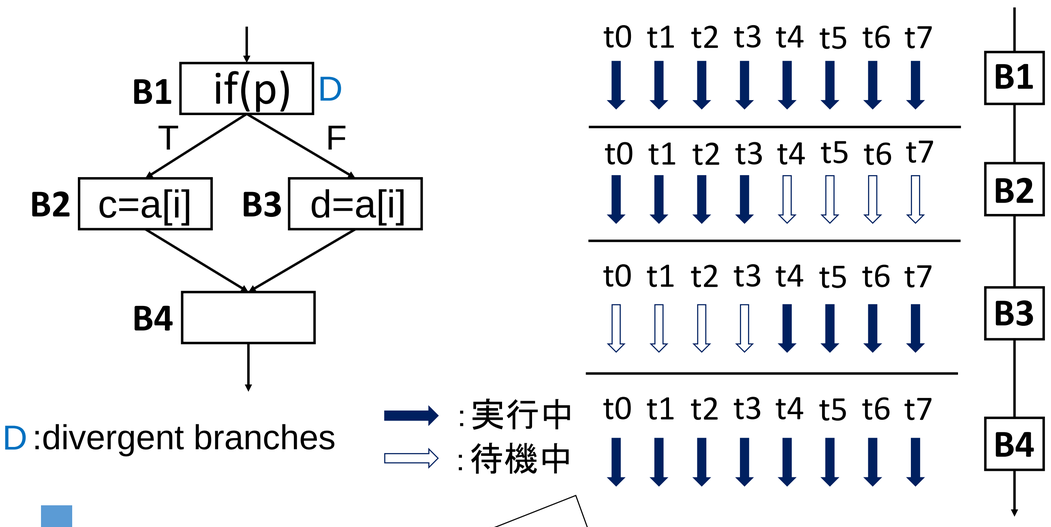
CFG上にクエリを伝播させ、式の冗長性を検査する手法。
クエリの解trueとfalseは、クエリが伝播した実行経路上で
同じ式が出現したかどうかを表す。



分岐発散(Branch Divergence)

SIMD処理で同一ウォープ内のスレッドが異なる分岐先に分岐するときに生じる実行効率の低下のこと。
分岐発散が生じると、各分岐先の命令を逐次的に実行する。
一方の経路に分岐したスレッドが命令を実行している間、もう一方の分岐経路に従うスレッドは休止状態となっている。

→ 並列性が失われ、GPUの実行効率が低下



分岐条件pがtrueであるスレッドt0～t3がB2の命令を並列実行し、その間分岐条件pがfalseであるスレッドt4～t7は休止している。次に、スレッドt4～t7がB3の命令を並列実行し、その間スレッドt0～t3は休止している。B2を実行したあとにB3を実行するという逐次実行になっている。

分岐発散を生じるプログラムに対しては、特定の経路にコードを挿入する従来のコード最適化は、分岐発散を増大させ実行効率のさらなる低下を招く場合があり、適用が難しい。

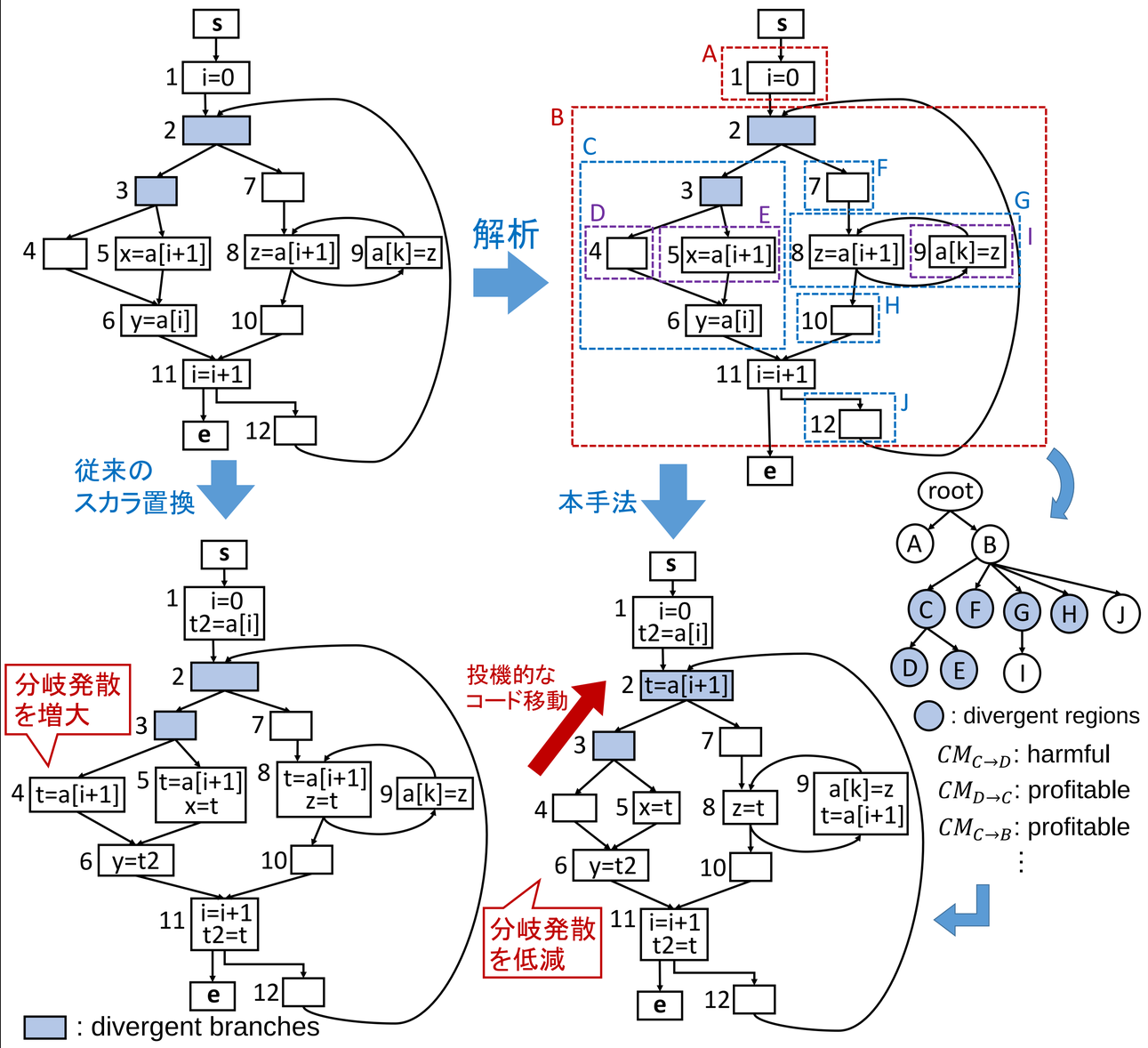
目的

分岐発散を低減し、GPUの実行効率を改善することで、プログラムの実行速度の改善を図る。

提案手法

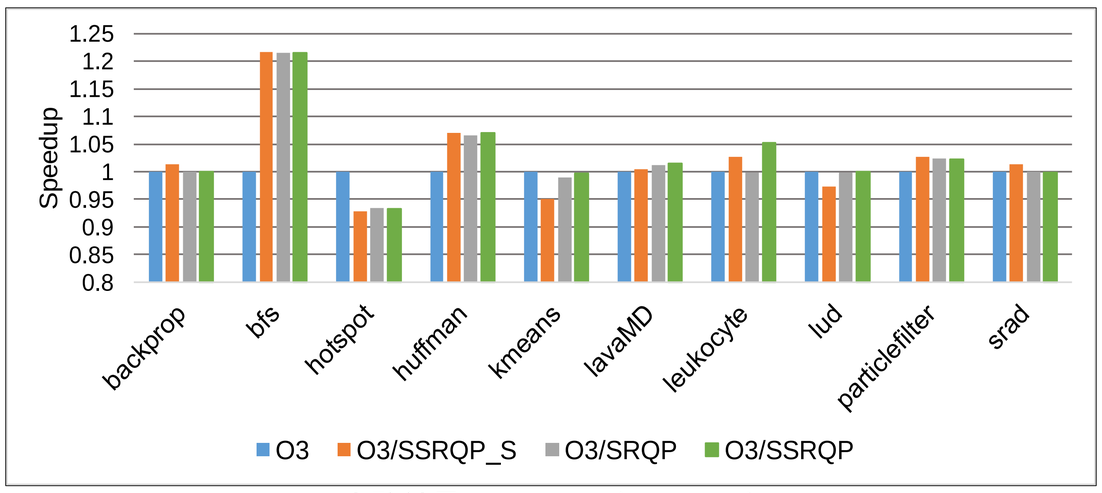
- 従来のスカラ置換と投機的なコード移動を組み合わせる
- 分岐発散している分岐では両方の分岐先を実行する性質から、片方の分岐先にしか存在しない式でも、実行効率を減じずに分岐前に巻き上げることができる
- CFGを分岐発散の有無に応じた領域に分けることで、コード移動に基づいた分岐発散の増減を定義する

→ スカラ置換によって多くの冗長な式を除去するだけでなく、
分岐発散も低減し、実行効率を改善する

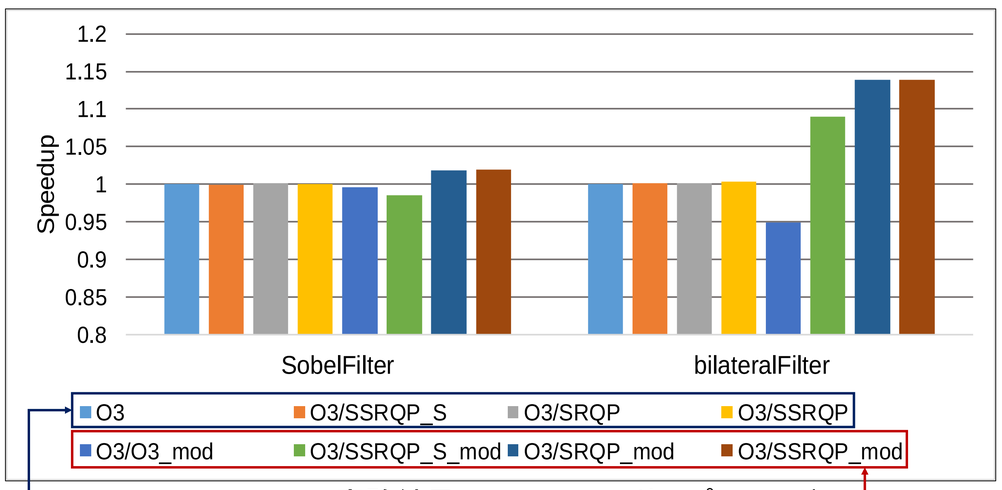


実験

本手法をベンチマークプログラムに適用し、適用前後で実行速度を比較した。
最大で約1.2倍の実行速度の改善が得られ、本手法の有効性を確認した。



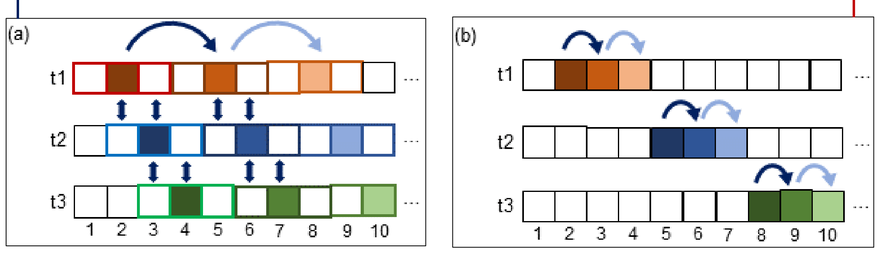
実験結果1: Rodiniaベンチマーク



実験結果2: Nvidia SDK サンプルコード

O3: nvcc –O3
SSRQP_S:分岐発散を考慮せずに投機的移動
SRQP:従来のスカラ置換
SSRQP:本手法
_mod:下図(b)のようにメモリアクセスを変更した各手法

実験環境
- OS: Ubuntu 16.04 LTS
- CPU: Intel Core i7-4770K
- GPU: GTX TITAN Black
- CUDA Toolkit 5.0



メモリアクセスパターンを
(a)コアレスアクセスから
(b)連続アクセスに変更