GPGPUにおけるシストリックアルゴリズムの効率的な実現について

*法政大学　理工学研究科　応用情報工学専攻
†法政大学　理工学部　応用情報工学科
‡大阪府立大学　理学系研究科　情報数理科学専攻

茂木啓輔*　和田幸一†　藤本典幸‡

目次

- はじめに
- 研究の目的
- GPUのアーキテクチャ
- 理論モデルについて
- シストリックアルゴリズムとは
- GPU上でのシストリックアルゴリズムの実装方法
- 今後の展望

はじめに

- GPUとは,3Dグラフィックスの表示に必要な計算処理を行う半導体チップ.
- GPGPUとは,本来画像処理を専門とする演算装置であるGPUを画像処理以外の汎用的な目的に応用する技術である.

研究の目的

シストリックアルゴリズムをGPUを用いて実装する手法を考案

理論的な評価と実際の結果とを比較

より効率的なシストリックアルゴリズムの実装方法の提案

GPUのアーキテクチャ

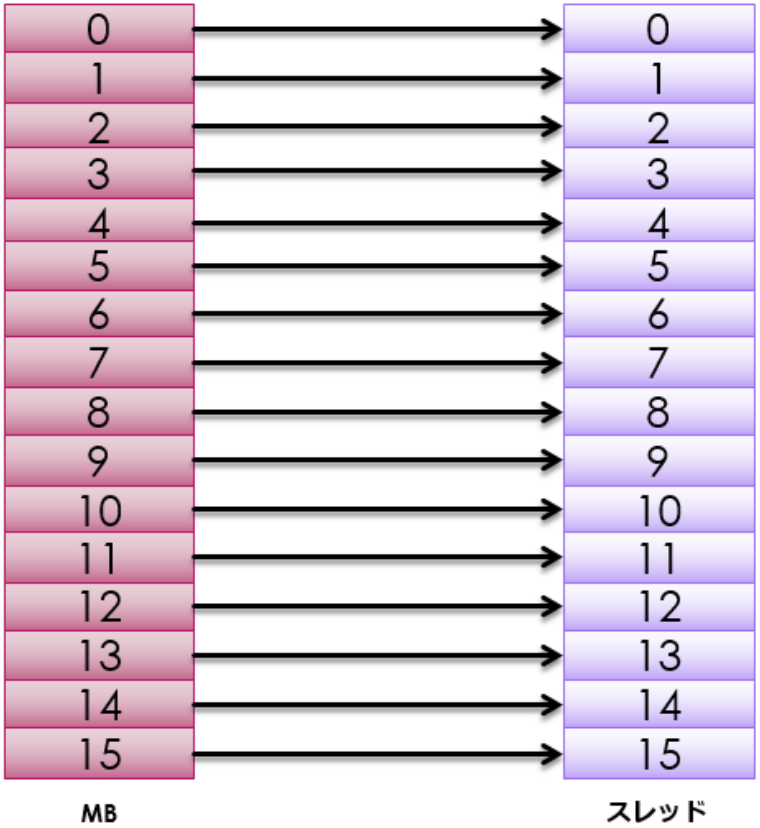
- GPUは,グリッド,ブロック(MP:マルチプロセッサ),スレッドの三階層の構成になっている.各命令はスレッド単位で実行される.
- これらはカーネル関数を実行する際に指定する.

グローバルメモリとシェアードメモリ

- GPUのメモリには2種類のメモリがある.

	グローバルメモリ	シェアードメモリ
メモリの場所	オフチップ	オンチップ
容量	1.5Gbyte	16-64Kbyte
アクセス時間	400-600clockcycle	4-6clockcycle

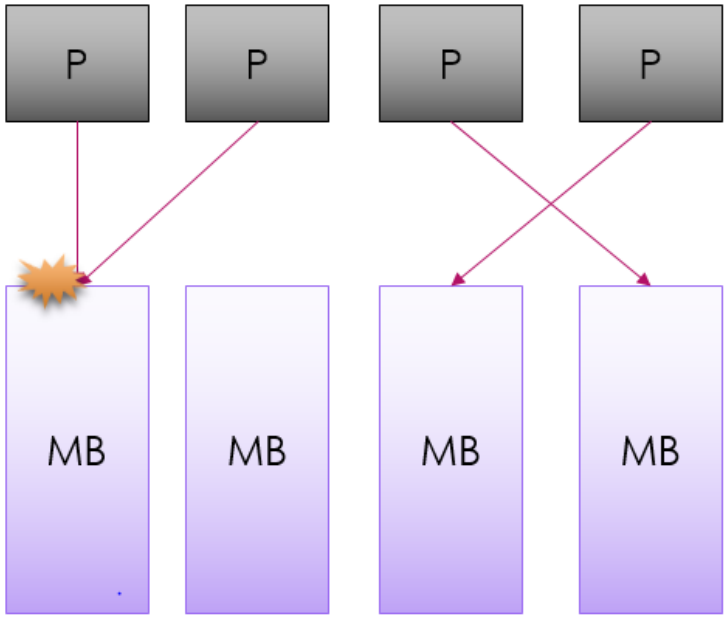
コアレッシング



- グローバルメモリの特性で,連続するデータにプロセッサがアクセスするとき,32バイト,64バイト,128バイトのまとまったデータ量を転送する事である.またハーフワープ(16スレッド)で転送される.

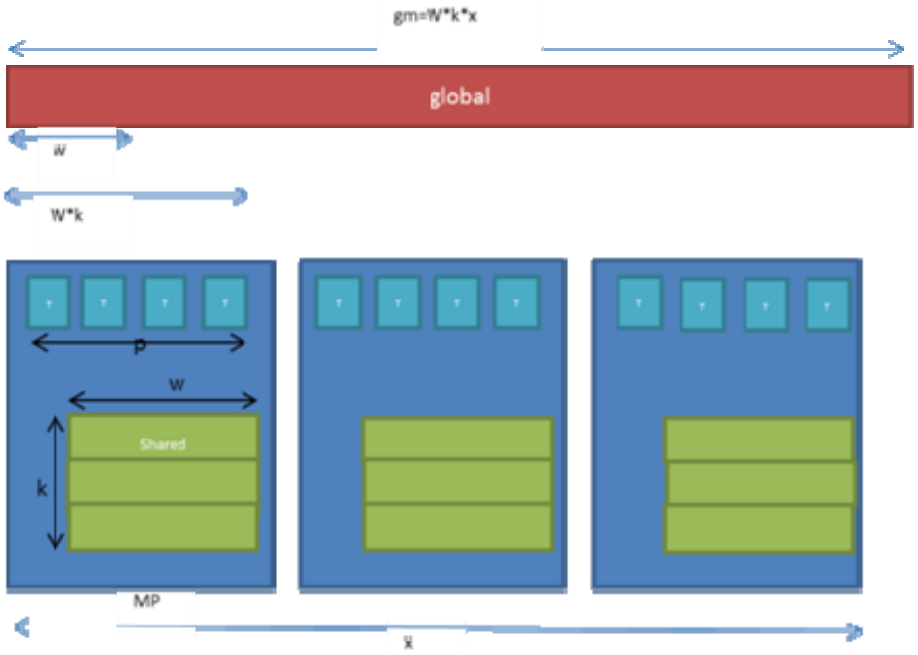
- 1ワード4バイトの時,64バイト単位
  1ワード8バイトの時,128バイト単位
  1ワード16バイトの時,128バイト単位で2回

バンクコンフリクト



- シェアードメモリの特性で,スレッドがシェアードメモリの同じメモリバンクへアクセスを行った際に,シーケンシャルに処理されてしまう.
- これを防ぐプログラミングが必要.

評価に使用する理論モデル



	各パラメータの説明
k	1バンク内のメモリ数
x	MP数
Lgm	グローバルメモリのレイテンシ
Lsm	シェアードメモリのレイテンシ
w	シェアードメモリのバンク数
wg=2*p	グローバルメモリのバンク数
gm=N=w*k*x	グローバルメモリ要素数
sm=w*k	シェアードメモリの要素数
N	入力要素数

評価に使用する理論モデルのポイント

- 本研究で使用する理論モデルは,グローバルメモリのコアレッシングとシェアードメモリのバンクコンフリクトの特徴を踏まえて二段階の構成にしている.

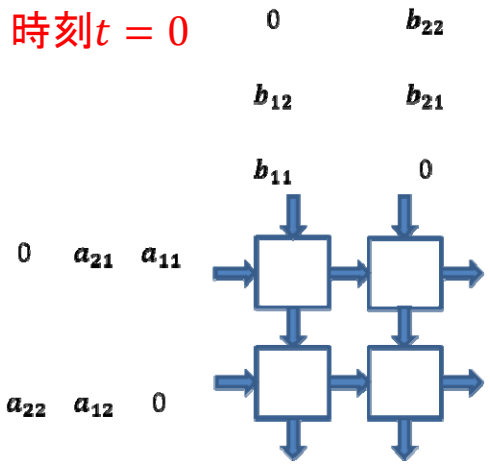
- グローバルメモリ,シェアードメモリへのアクセス時間を区別するために,Lgm,Lsmの2種類の独自のパラメータを設定した.

シストリックアルゴリズムとは

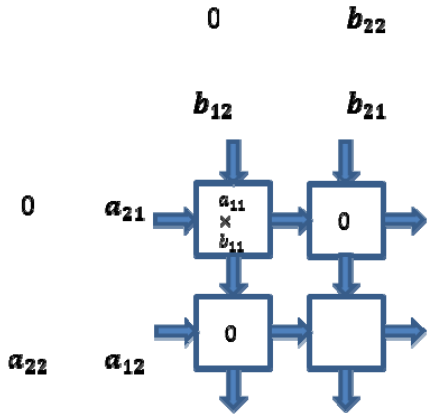
- シストリックアルゴリズムとは一般的に,” 多数の同一構造の基本演算器(セル)が1次元または2次元アレイ状に規則的に配置されたシステム”のことを示す.
- このアルゴリズムでは,データの流れが規則正しく波となって一列に進んでいく.

ネットワークの動作例(2*2の行列積)

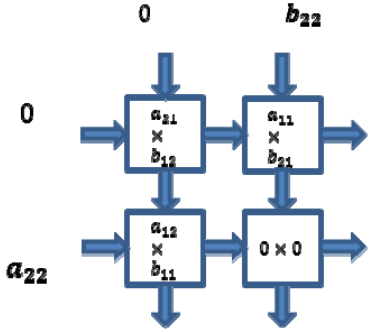
時刻$t=0$



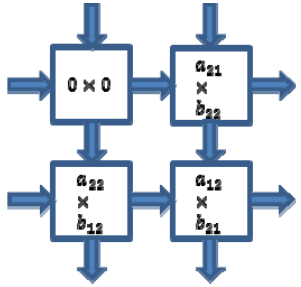
時刻$t=1$



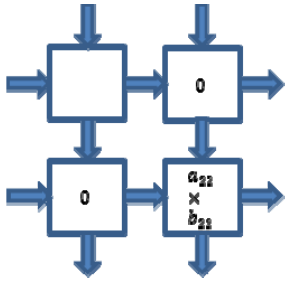
時刻$t=2$



時刻$t=3$



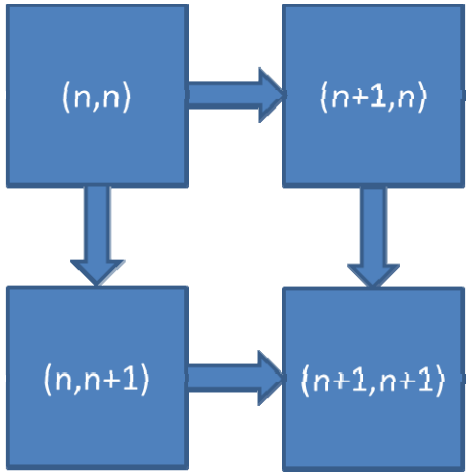
時刻$t=4$



シストリックアルゴリズムの特徴

特徴①

- 近接したモジュール(セル)間の相互結合を有効に使うという点で,非常に良い並列アルゴリズムである.

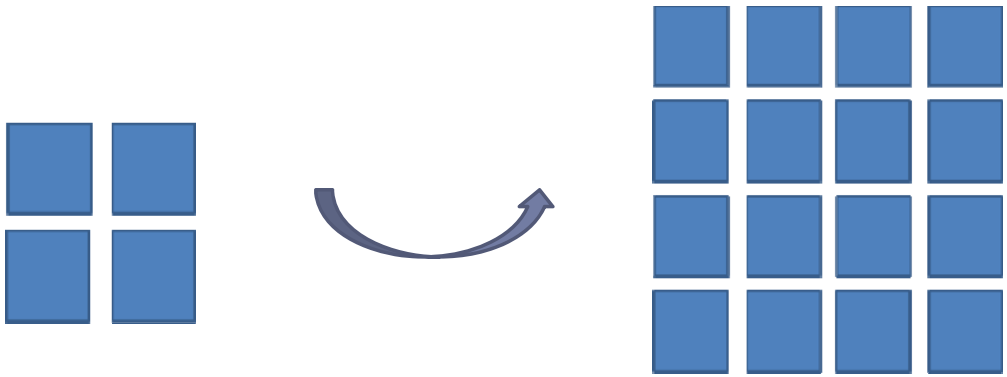


規則的な入出力により,効率的なメモリアクセスが可能になるのではないか?

シストリックアルゴリズムの特徴

特徴②

- 問題のサイズに応じて,規則的に拡張することが出来る.

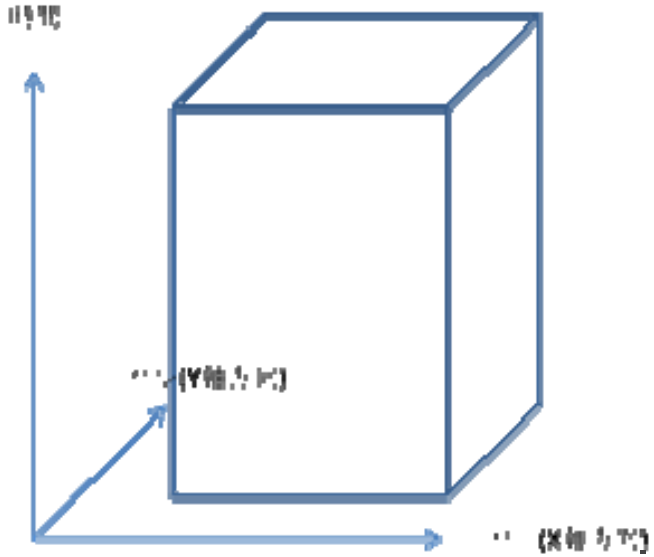


統一的にネットワークを記述することが出来る.

シストリックアルゴリズムの特徴

特徴③

- 時刻とセルを指定することで,処理を行う部分を決定することが出来る.

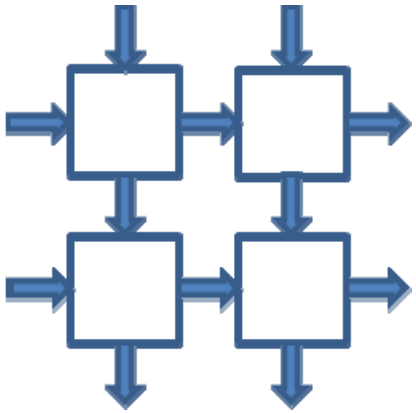


GPU上で実装する場合,
①セル単位で演算
②時刻単位で演算
2つの方法が考えられる.

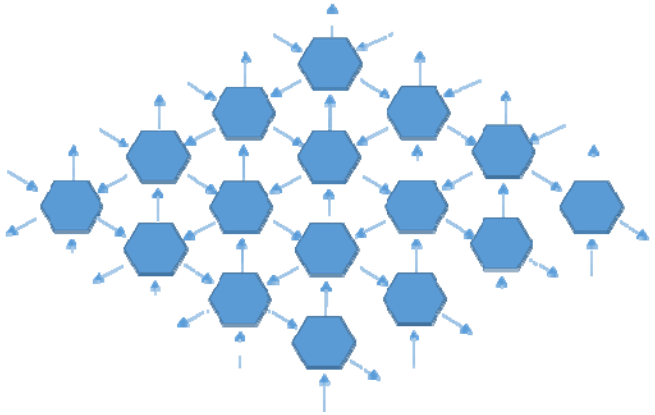
ネットワークの種類

- 本研究ではGPU上で以下の二種類のネットワークの実装を検討している.

①二次元のネットワーク

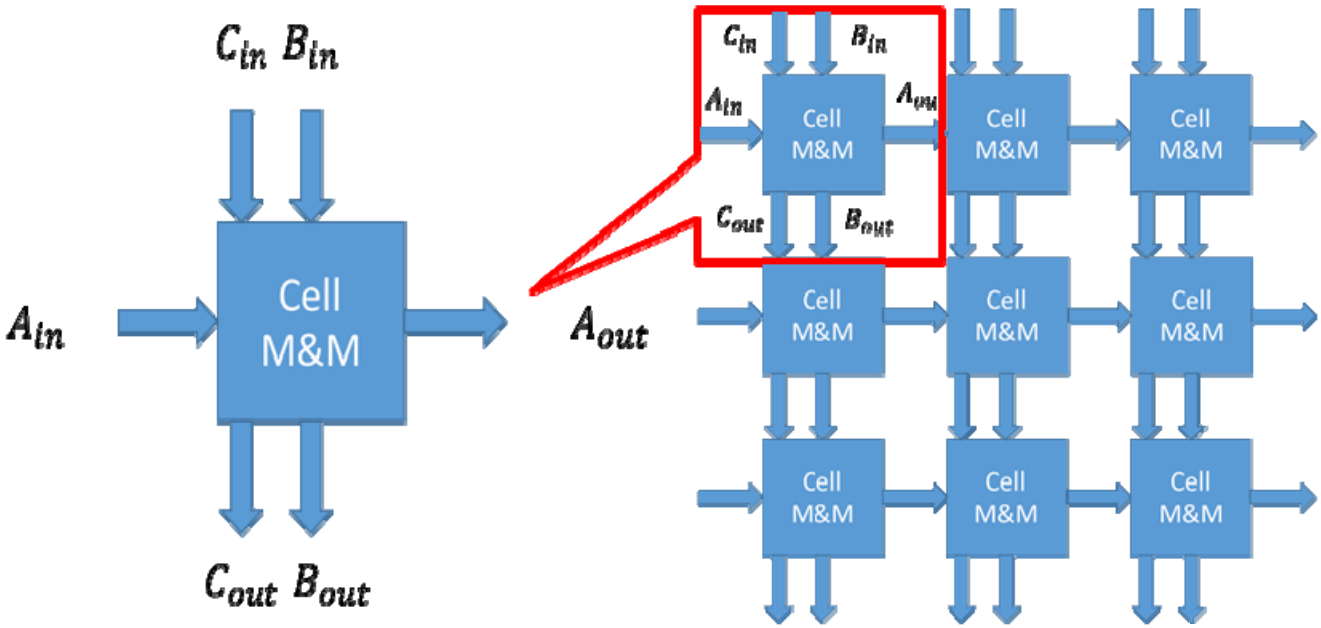


②六角接続のネットワーク



ネットワークの記述方法

- 以下のように記述することで,GPU上でシストリックアルゴリズムを実現する.



ここでは引き続き,二次元のネットワークで行列積を求めるものを例として示す

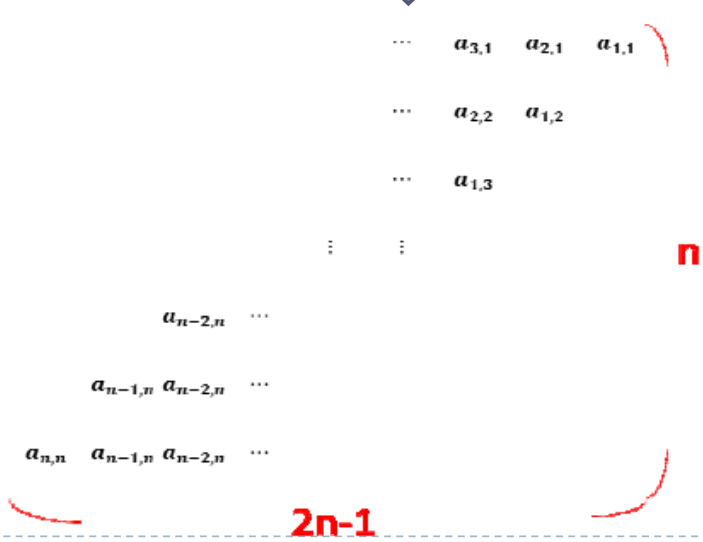
ネットワークの記述方法

- 時間と問題の記述

時刻	**t** $t \geq 0$
問題(行列)のサイズ	n n>0
横からの入力行列	a
上からの入力行列	b
結果として出力される行列	c
行列の行番号	i $1 \leq i \leq n$
行列の列番号	j $1 \leq j \leq n$
横からの入力のタイミング	$a_{i,j}$ $M_{\&}M(1,j)$ for t = 1,2n-1 $a_{1,t+j-1}$
上からの入力のタイミング	$b_{i,j}$ $M_{\&}M(i,1)$ for t = 1,2n-1 $b_{t+1-1,1}$

例:横方向からの入力行列

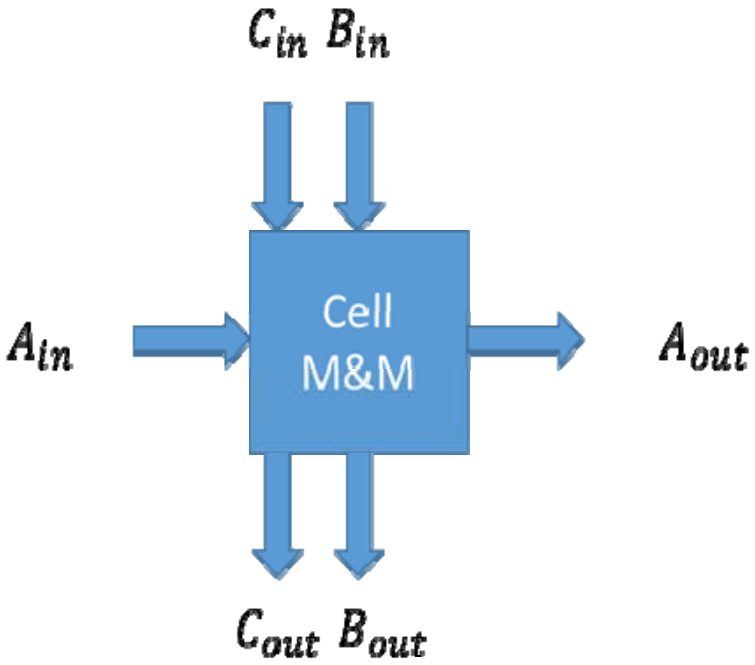
$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{1n} & \cdots & a_{nn} \end{pmatrix}$$



ネットワークの記述方法

- セルの記述

セルの名前	$M_{\&}M(int\ x, y)$
上からの入力	A_{in}
左からの入力	B_{in}
下への出力	A_{out}
右への出力	B_{out}
セルでの演算結果の入力	C_{in}
セルでの演算結果の出力	C_{out}
セルのx座標	x $0 \leq x \leq n-1$
セルのy座標	y $0 \leq y \leq n-1$
セル内での演算	$A_{out}(t+1) = A_{in}(t)$
セル内での演算	$B_{out}(t+1) = B_{in}(t)$
セル内での演算	$C_{out}(t+1) = A_{in}(t) * B_{in}(t) + C_{in}(t)$

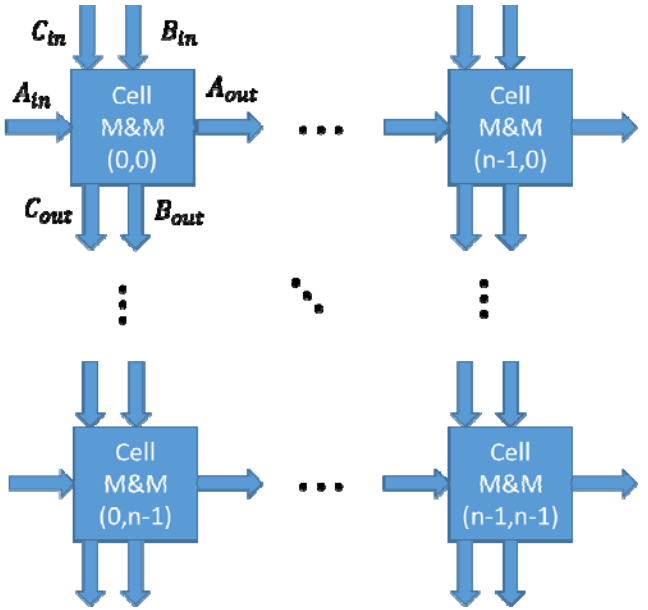


ネットワークの記述方法

- ネットワークの接続関係の記述

for x = 0,n-1 do
　　for y = 0,n-2 do
　　$M_{\&}M(x,y+1)B_{in}(t) \Leftrightarrow M_{\&}M(x,y)B_{out}(t)$

for y = 0,n-1 do
　　for x = 0,n-2 do
　　$M_{\&}M(x+1,y)A_{in}(t) \Leftrightarrow M_{\&}M(x,y)A_{out}(t)$



GPU上での実装方法

- シストリックアルゴリズムをGPU上で実装する際には,以下の3つの方法が考えられる.

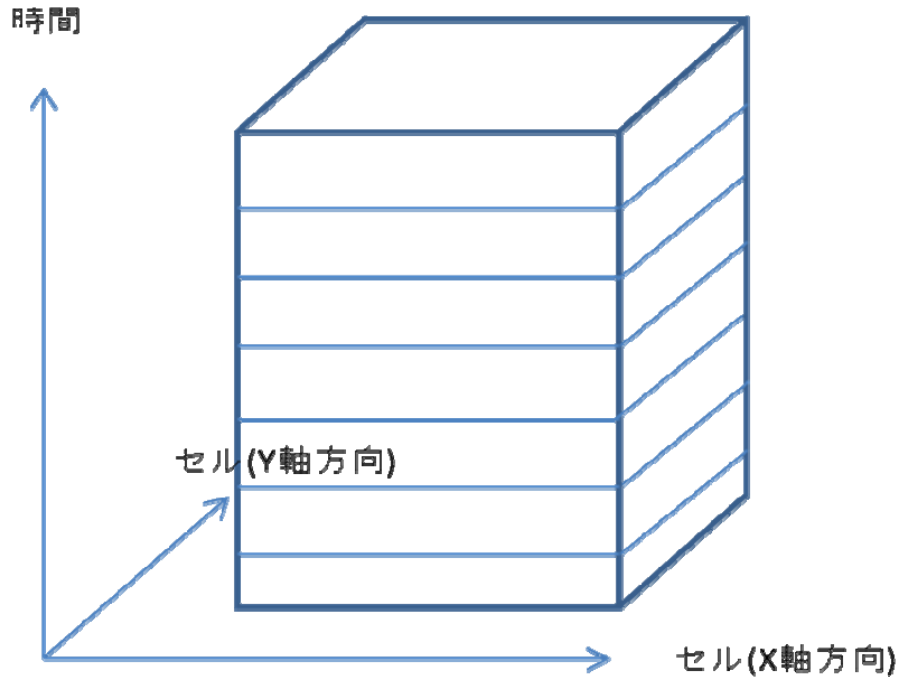
方法①:先ほどの記述を元に,1単位時刻分の演算を行い,入力の大きさに応じて繰り返し演算を行なう

方法②:①の拡張版で複数単位時刻をまとめて演算する

方法③:1単位時刻分の演算をDPを用いて分割して演算を行い,入力の大きさに応じて繰り返し演算を行なう

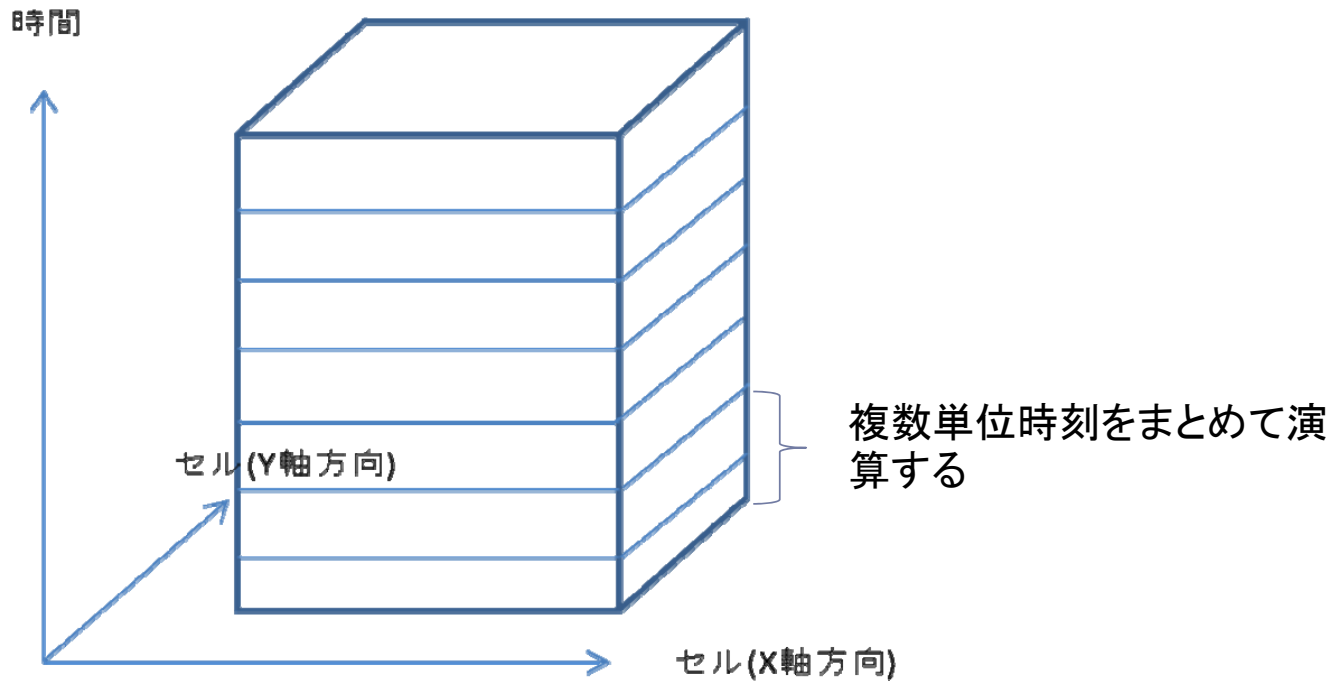
GPU上での実装方法

方法①：先ほどの記述を元に，1単位時刻分の演算を行い，入力の大きさに応じて繰り返し演算を行なう



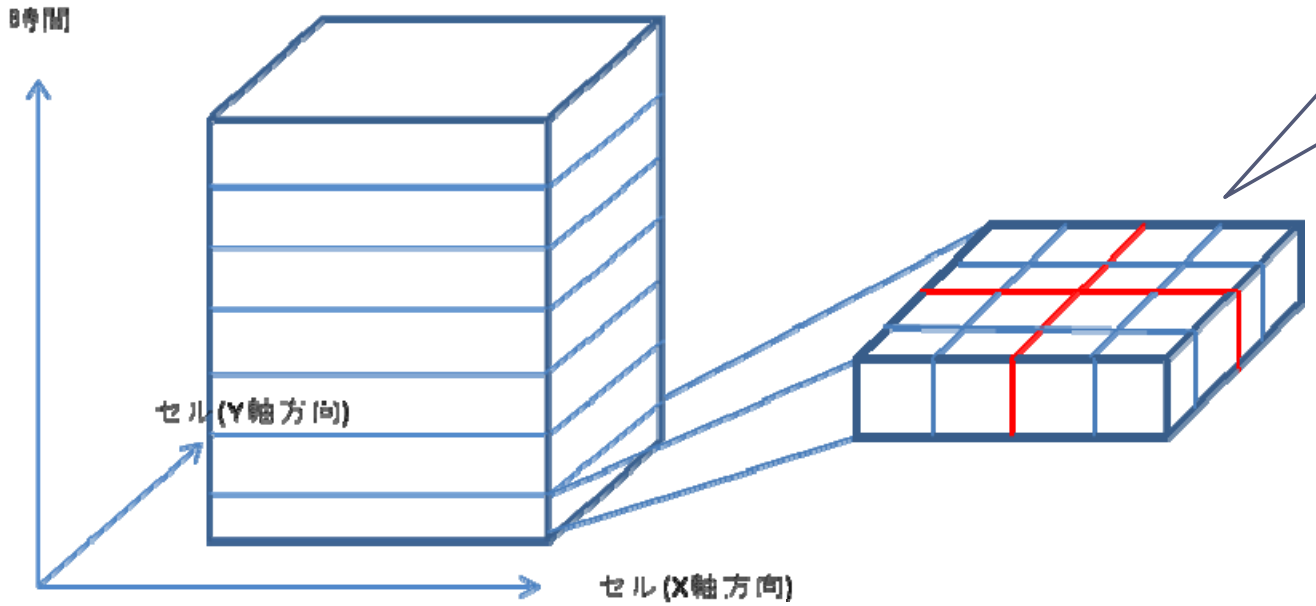
GPU上での実装方法

方法②:①の拡張版で複数単位時刻をまとめて演算する



GPU上での実装方法

方法③:DP(dynamic parallelism)を用いて行う



分割した際の境界となる部分の要素を重複して演算してしまう

様々な分割の方法を検討中

今後の展望

任意のシストリックアルゴリズムをGPU上で実現できるような手法の提案を行う

＊問題のサイズとネットワークの種類を入力として与えるだけで,自動的にネットワークを生成し,実行を行えるようなシステムの実装
＊シストリックアルゴリズムを理論モデル上でシミュレートし,実際の実装結果との比較を行う

GPU上での効率的なシストリックアルゴリズムの実装方法の提案を行う

最後に

- ご清聴ありがとうございました.