GPUおよびCPUを併用する高効率のFFTライブラリ

東京工業大学　理学部情報科学科
03-0537-2　尾形泰彦
指導教員　松岡聡

2007/2/16

背景

- GPUを用いた汎用計算
  - GPUはCPUに対して性能比が良い
    - 500GFLOPSが8万円
    - CPUは25GFlopsが2万円

    } 理論性能比**20倍**
    価格性能比**5倍**
  - GPU単体でいかに効率よく計算するかが中心

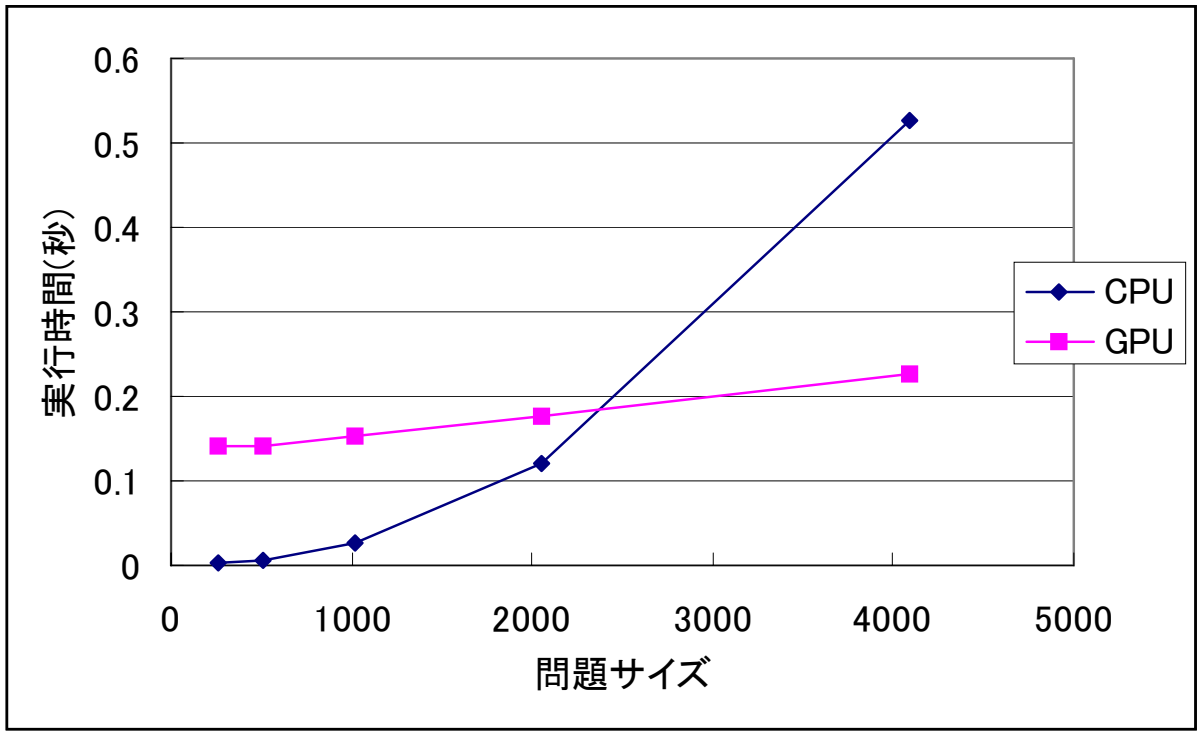
→CPUとGPUを併用することで
　より効率よく計算

→GPUとCPUの性能特性を知る必要がある

背景：CPU/GPU性能特性の違い

- GPUは計算時間以外のオーバヘッドが存在

  →小さなサイズの場合、CPUで計算する方が速い



注：1024列
計算した実行時間

- GPUではメモリサイズが限られている

  →大きなサイズの場合、GPUメモリにデータが入りきらない

関連研究

- GPUFFTW[Nagaら,'06]
  - GPUを用いたFFTライブラリ
  - CPUを用いることは無考慮
- cg-gemm[大島ら '06]
  - CPUとGPUを併用したGEMMライブラリ
  - cg-gemmでは、CPUの処理とGPUの操作に関する処理をCPU上で並列する必要はない

→FFTの計算はGEMMより複雑

- GEMM→GPU内で計算が完結
- FFT→CPU側からの制御が必要

- そのままアルゴリズムをFFTには適用できない

目的と提案

- 目的
  - CPUとGPUの併用の有効性を検証
  - CPUとGPU並列FFTの最適分割率決定
- 提案
  - 性能向上のために、CPU、GPUに対して計算量を配分

    →性能モデルを行って最適分割率を予測

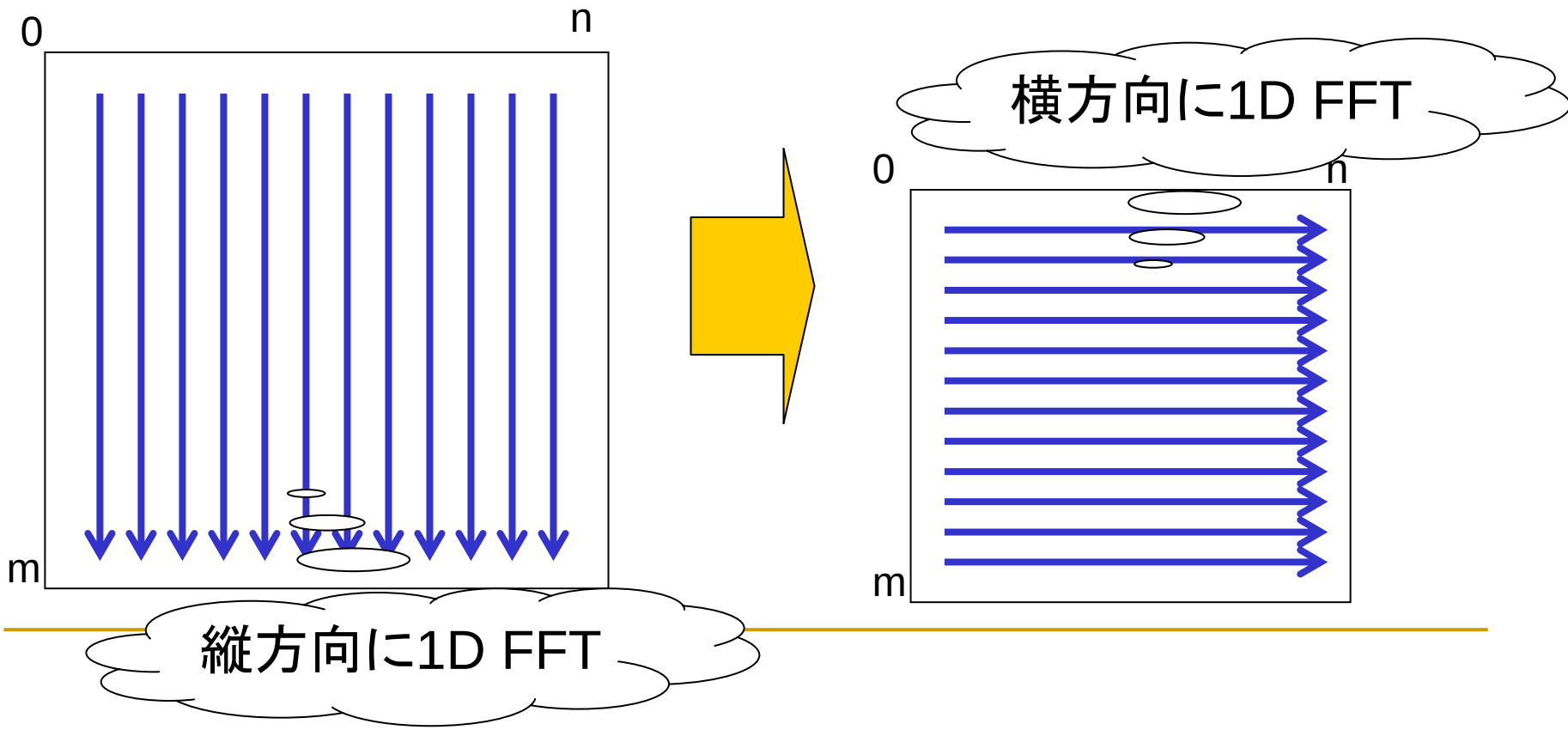
    →予測した最適性能で問題を実行

成果

- 成果
  - GPUとCPUを併用する2D FFTライブラリのアルゴリズムの提案とプロトタイプ実装
  - ライブラリの実行時間に対する性能モデルの構築
    - 性能モデルと実測値を比較し有効性を確認
    - 性能モデルから他の問題サイズの最適値を推測
  - 既存のFFTと比較して2倍の性能

2D FFTアルゴリズム

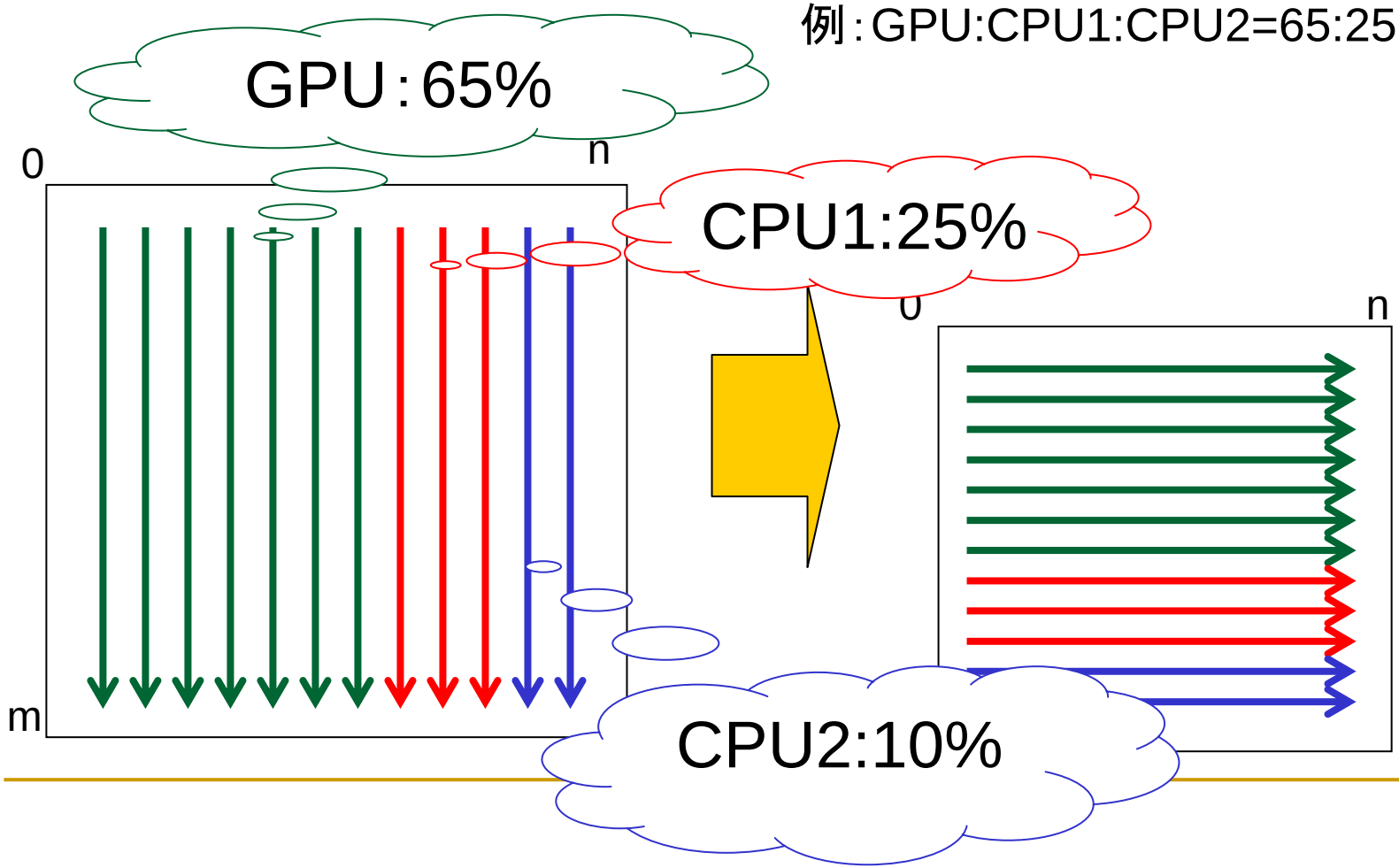
- 高速離散フーリエ変換
  - スペクトル解析、流体シミュレーションetc
- 2D FFT = 1D FFTを各軸に実行



2D FFTの分割法

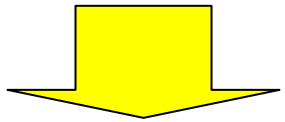
- 2D FFTを分解することで得られる、1D FFTをCPU/GPUに分割率に応じて割り当て

例：GPU:CPU1:CPU2=65:25:10



最適分割率の決定法

FFTの計算に対する性能モデルを構築



実行前：

プレ実行から性能モデルのパラメータの決定

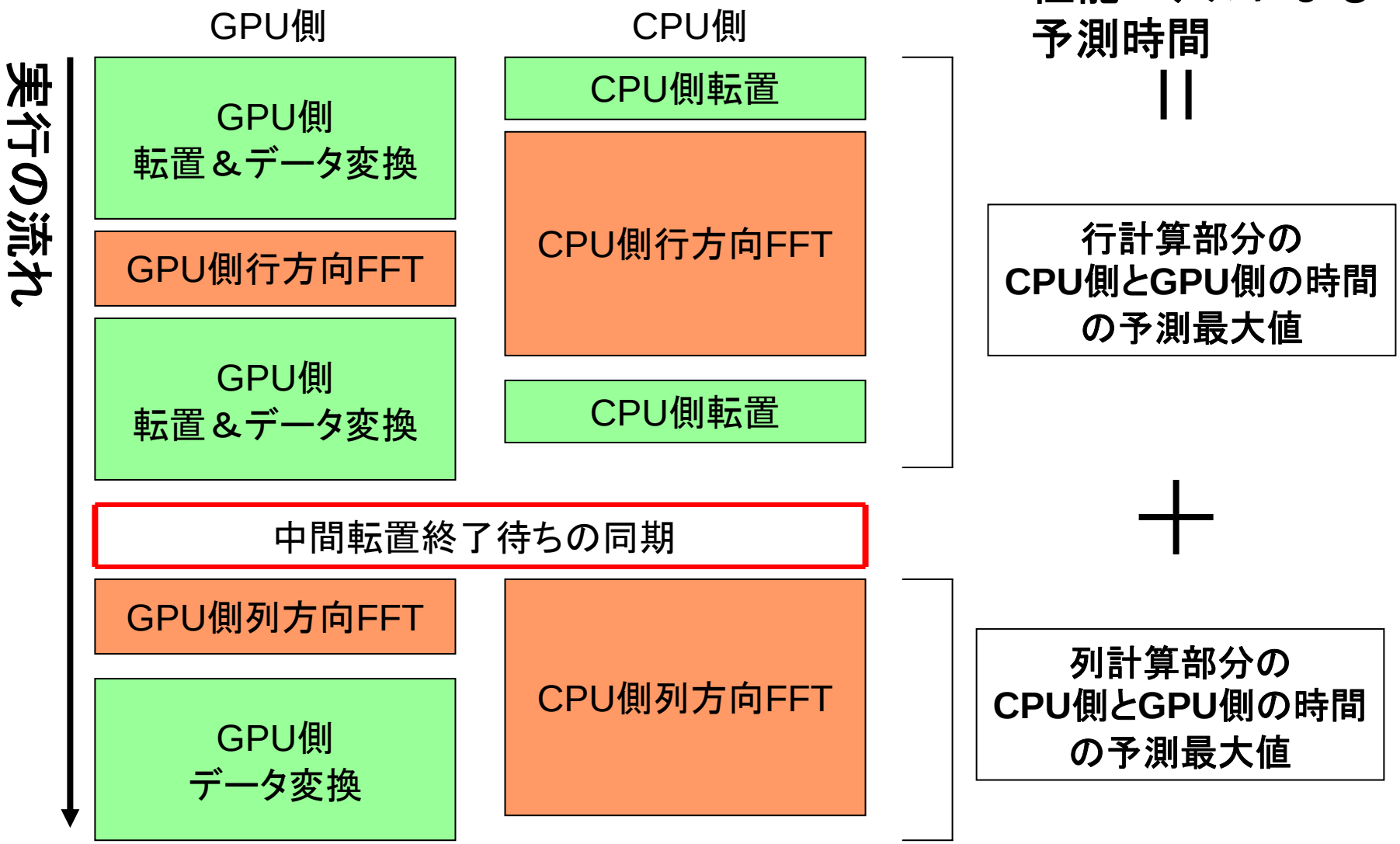
実行開始時：

求めたパラメータと問題サイズを性能モデルに代入し、最適割り当て率を推定

ex) 5%刻みでGPU割り当て率に応じた実行時間を予測

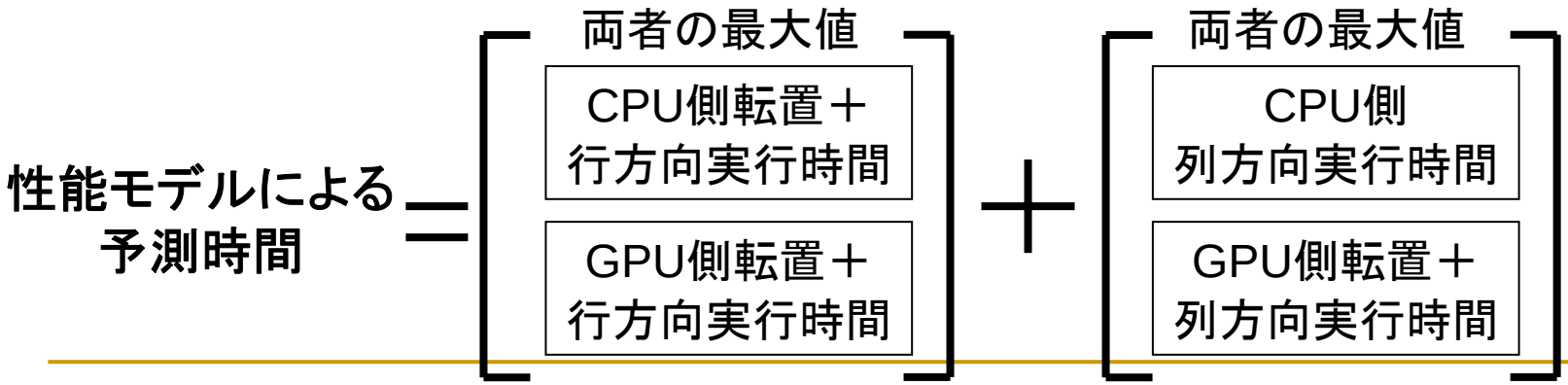
→その中で最短時間となる割り当て率を最適とする

実行時間の性能モデル（1）



実行時間の性能モデル（2）

メモリコピー・ 転置時間	行列サイズ^2 × 全体に対する転置・コピーを行う割合 × 定数
GPU側 FFT計算時間	⌈行列サイズ/1回の最大計算FFT数⌉ ×（GPUの最低時間＋ 定数 × 行列サイズ × 1回のFFT数）
CPU側 FFT計算時間	行列サイズ^2 × log(行列サイズ) × 計算本数 × 定数



実装

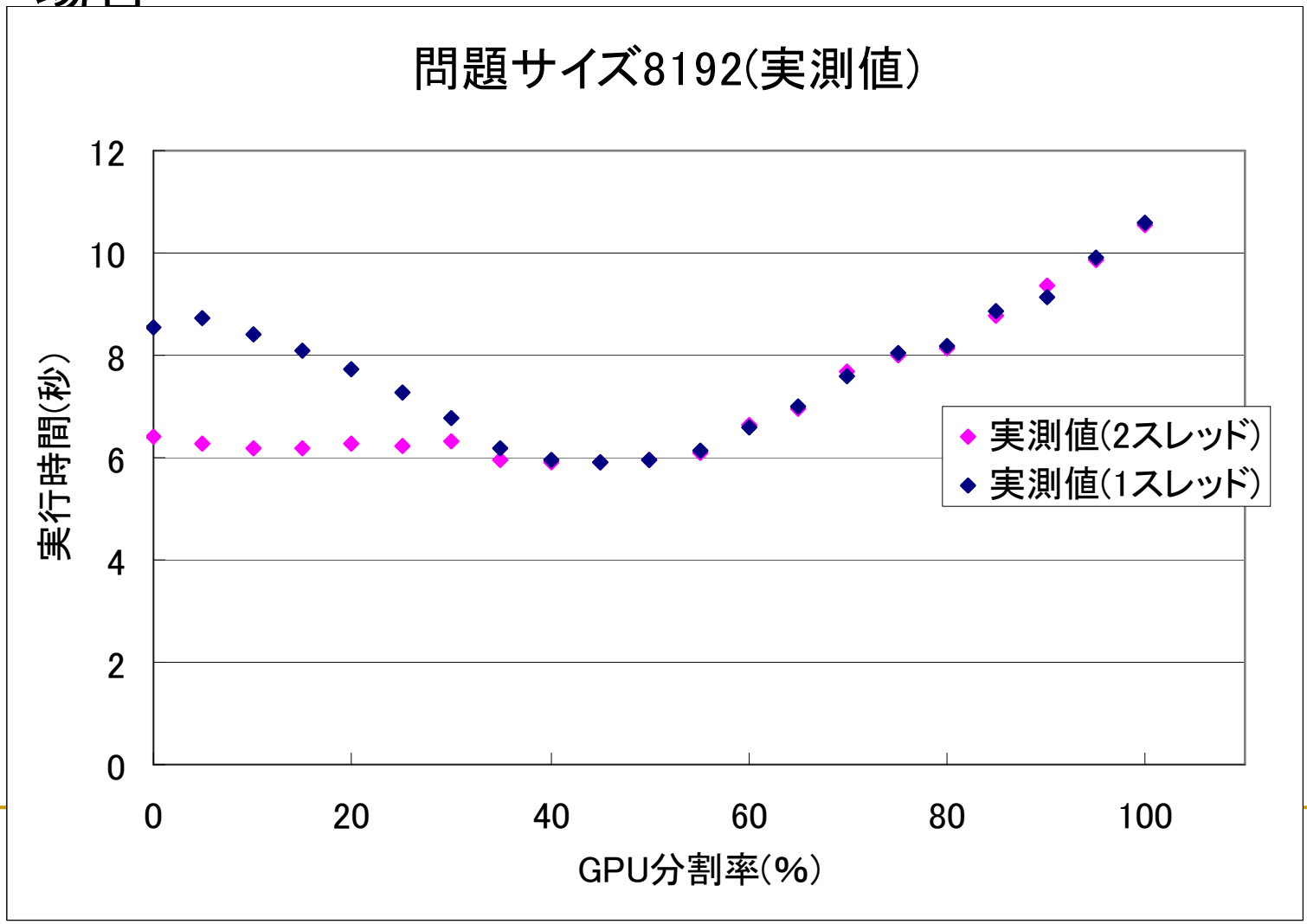
- 汎用のFFTライブラリを使用
  - CPUライブラリとしてFFTW[Frigoら]、GPUライブラリとしてGPUFFTW[Nagaら]を使用
- 行列の転置をブロック化により高速化
- 行列の転置の順序を調整することでメモリアクセス量の低減

評価

- 評価項目
  - ライブラリの実行性能の評価
  - 性能モデルの検証
- 評価環境
  - CPU:Core 2 duo E4300@2.13Ghz
  - memory:4GB
  - GPU:Geforce8800GTX
  - Graphic memory:768MB
  - Bus type:PCI Express 16X
  - Linux 2.6.18,GNU GCC4.1.2
  - NVIDIA Linux Display Driver version 7946

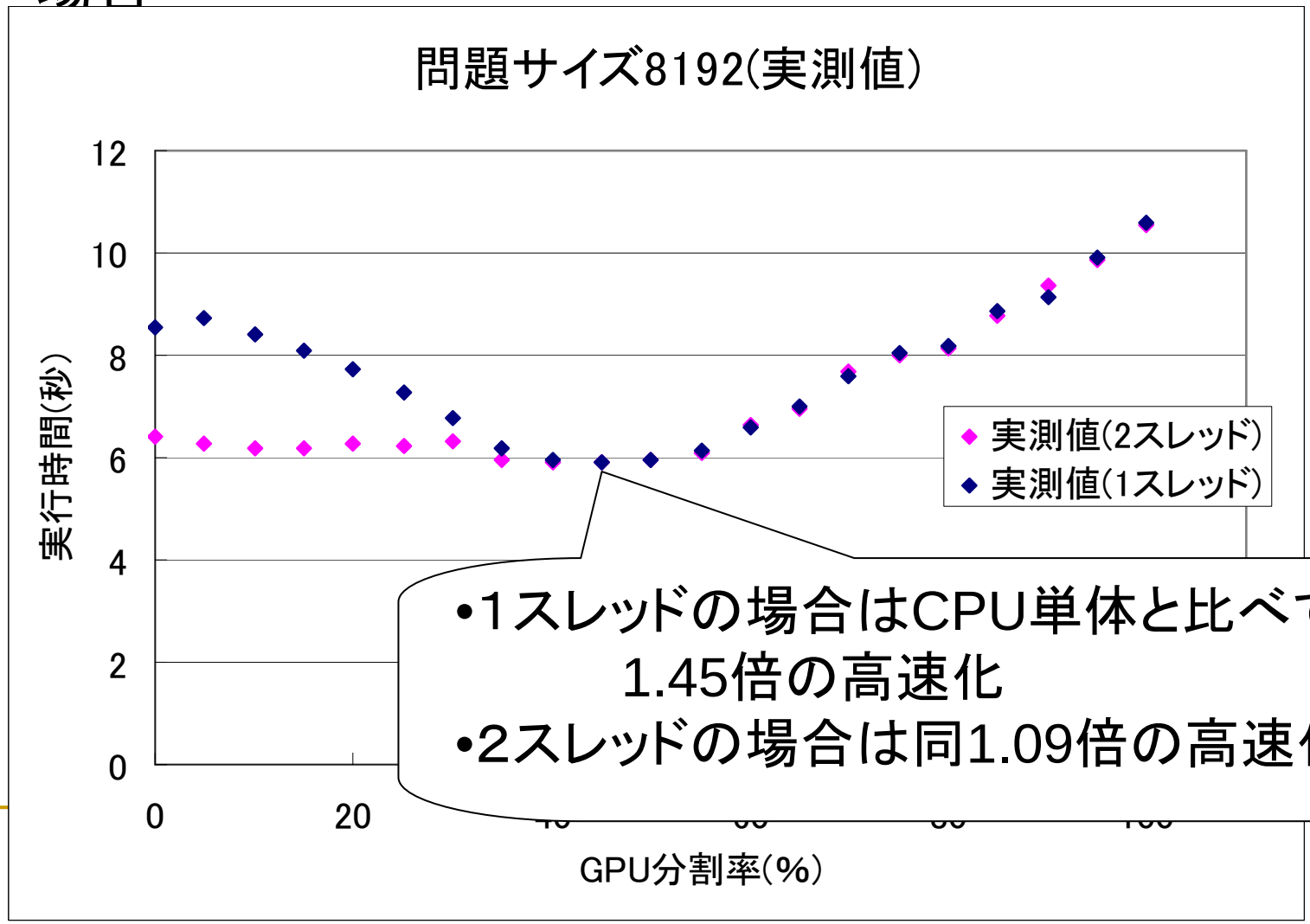
評価：本ライブラリの実行性能

- 問題数8192^2、CPUが1スレッドと2スレッド(等分割)の場合



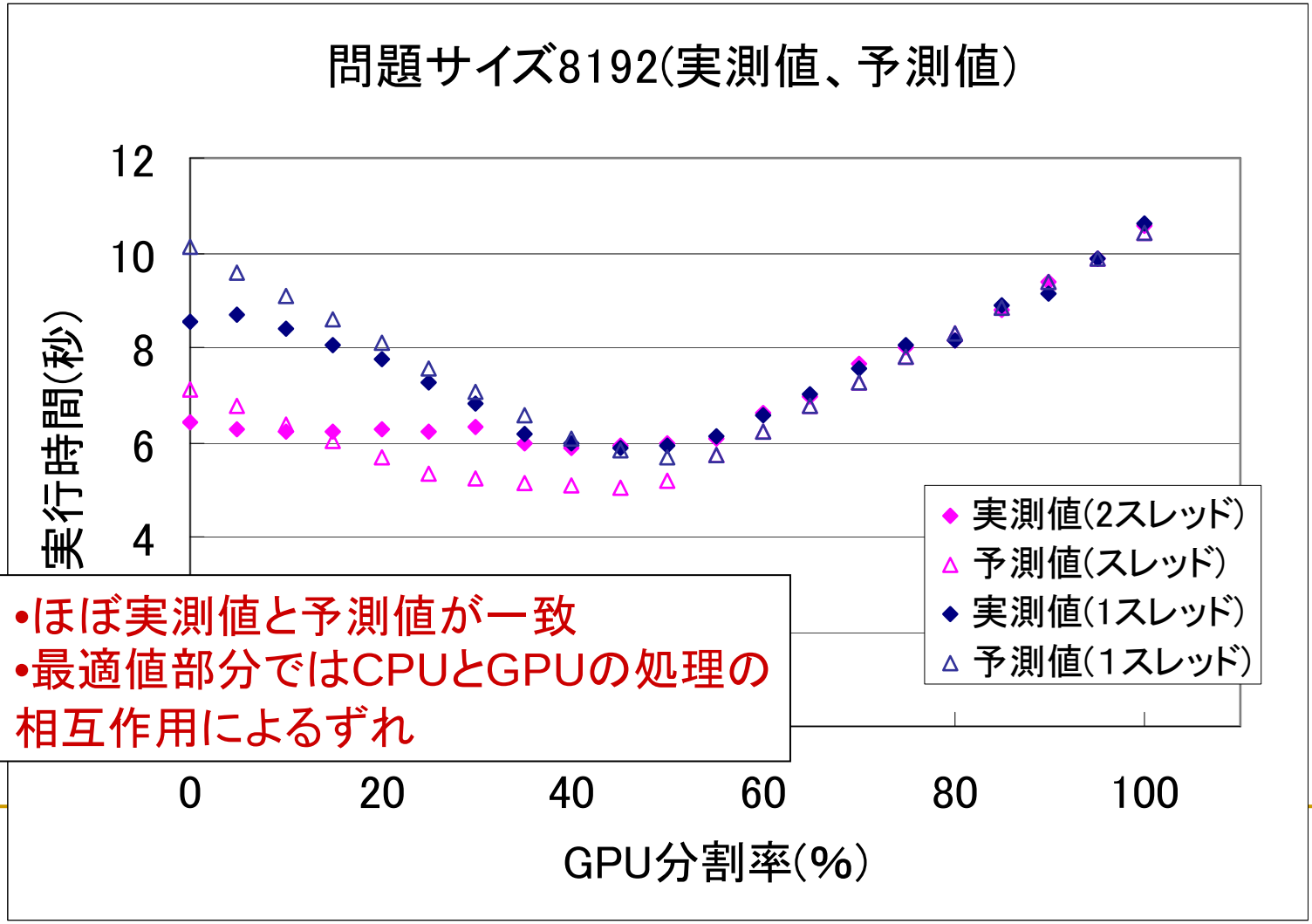
評価：本ライブラリの実行性能

- 問題数8192^2、CPUが1スレッドと2スレッド(等分割)の場合



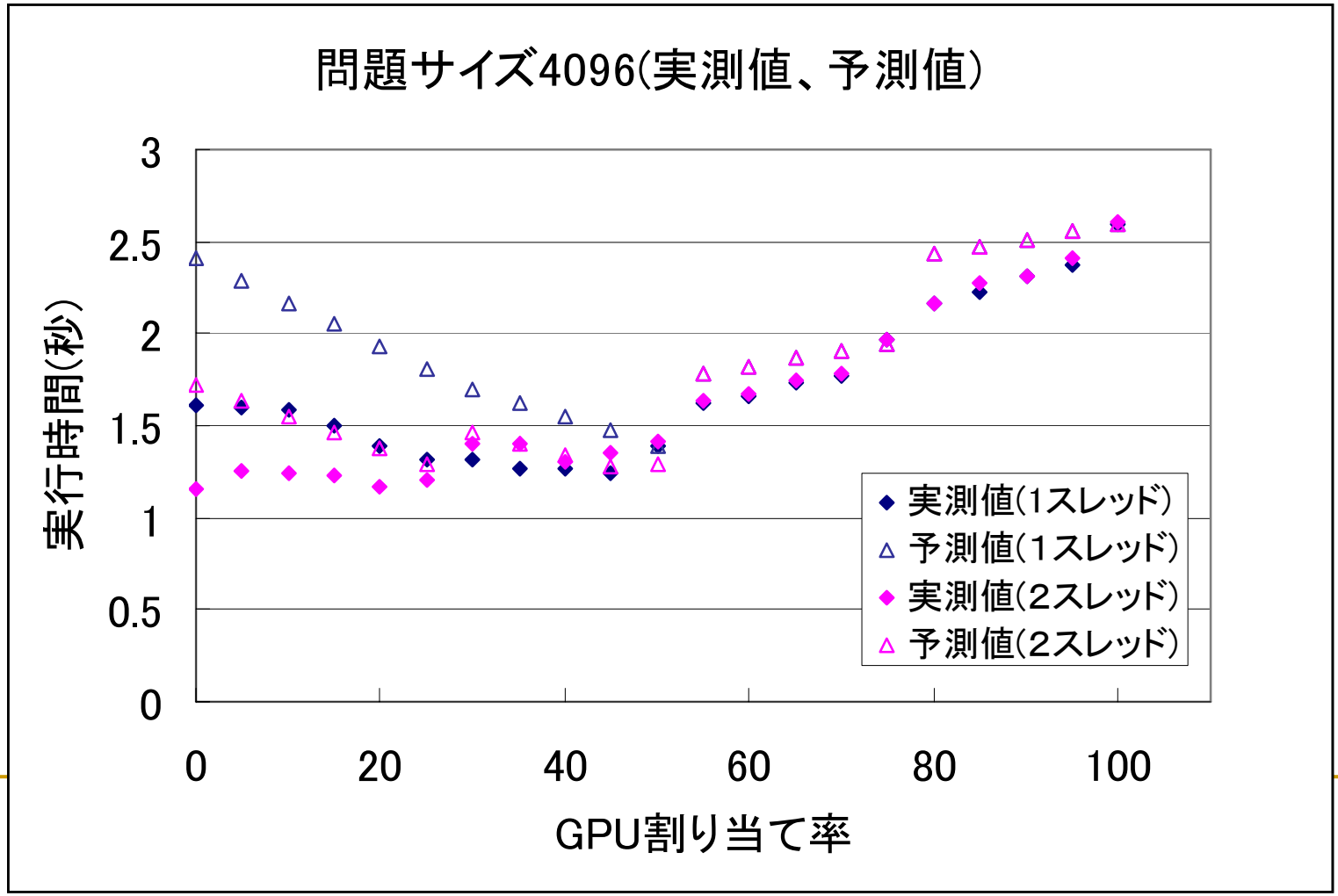
評価：性能モデルの検証（1）

- 問題数8192^2、CPUは1スレッドと2スレッド(等分割)の場合
- 性能モデルによる予測値は問題数8192^2のデータから作成



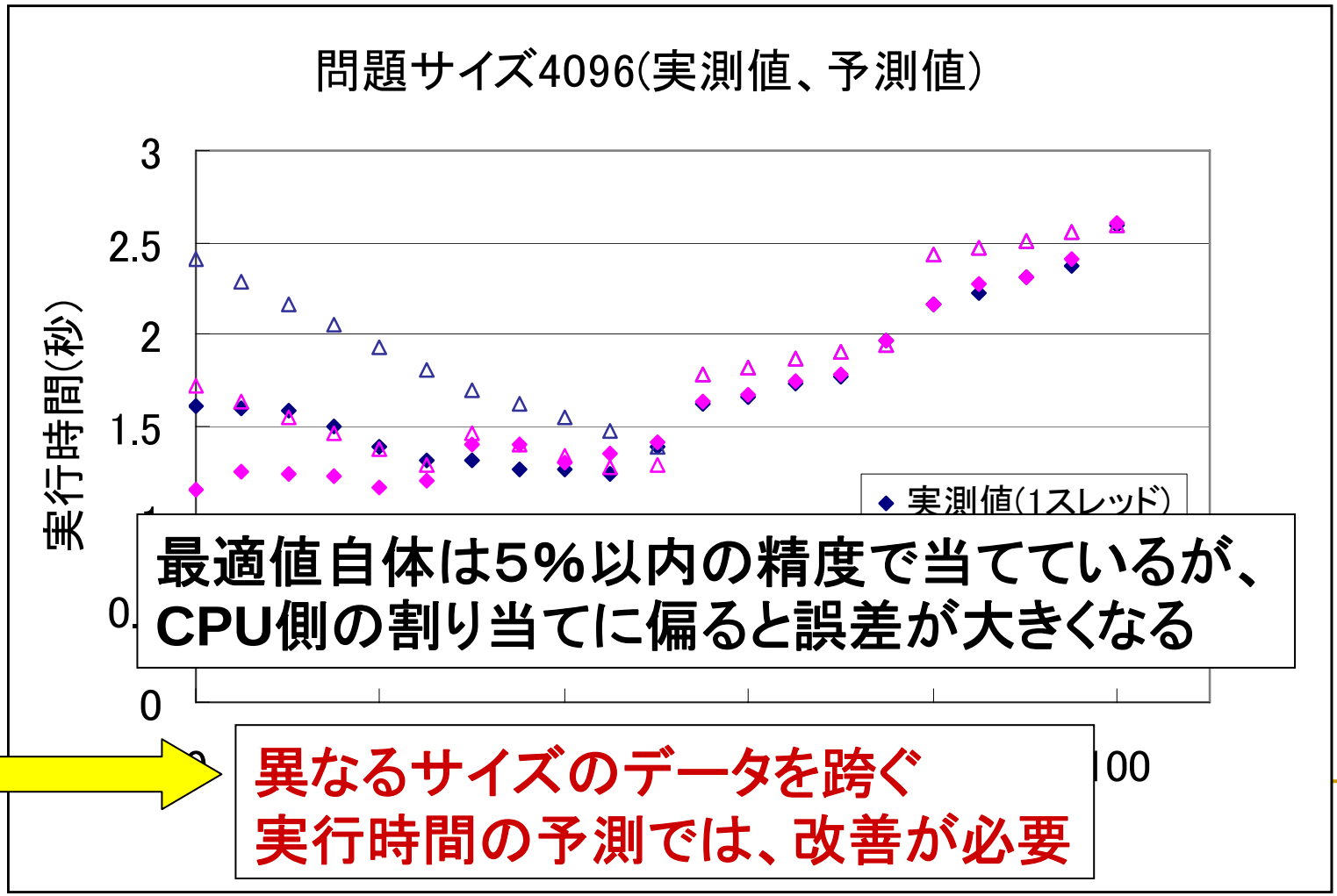
評価：性能モデルの検証(2)

- 先ほどの要素数8192^2の場合のデータから、要素数4096^2の傾向を予測



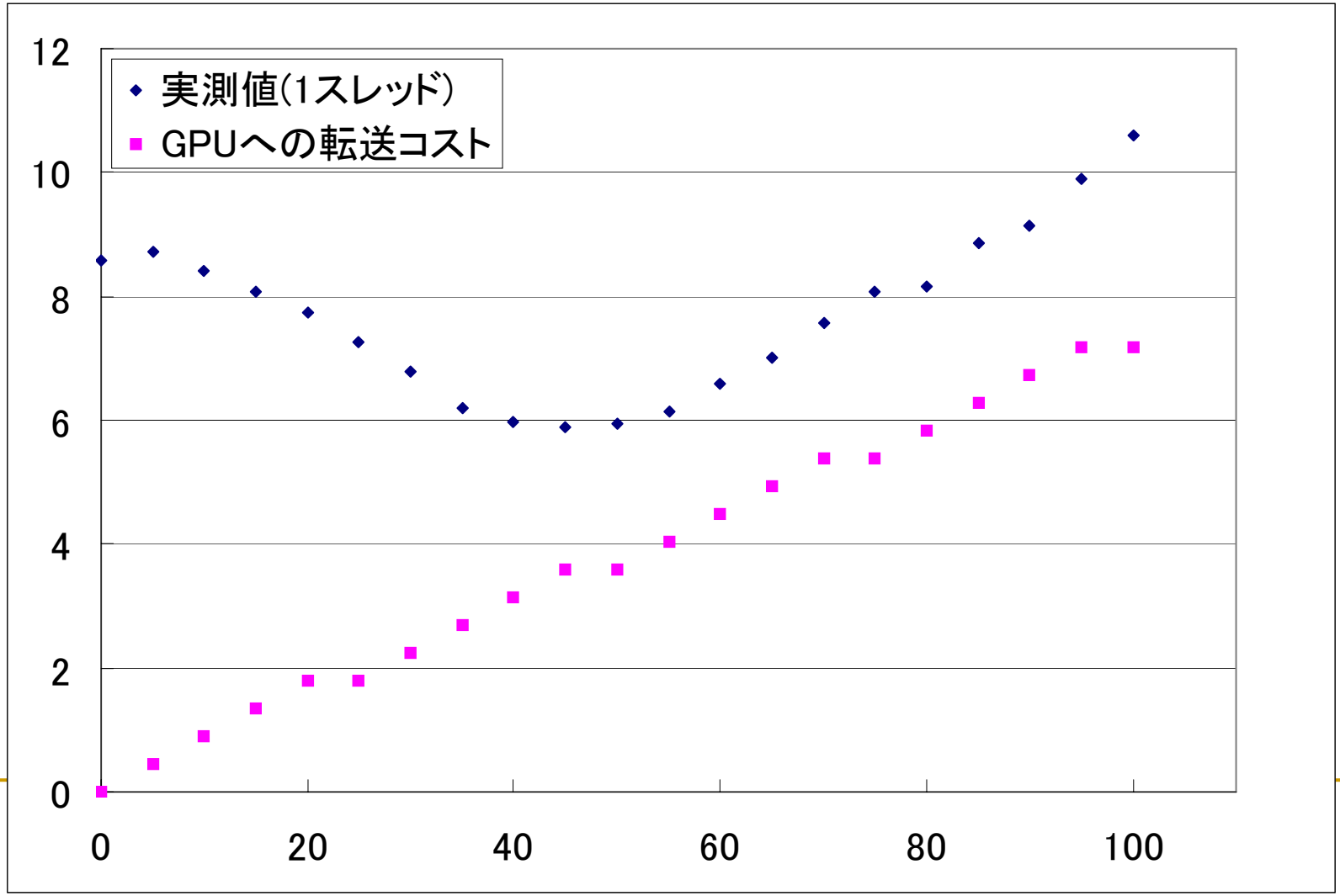
評価：性能モデルの検証(2)

- 先ほどの要素数8192^2の場合のデータから、要素数4096^2の傾向を予測



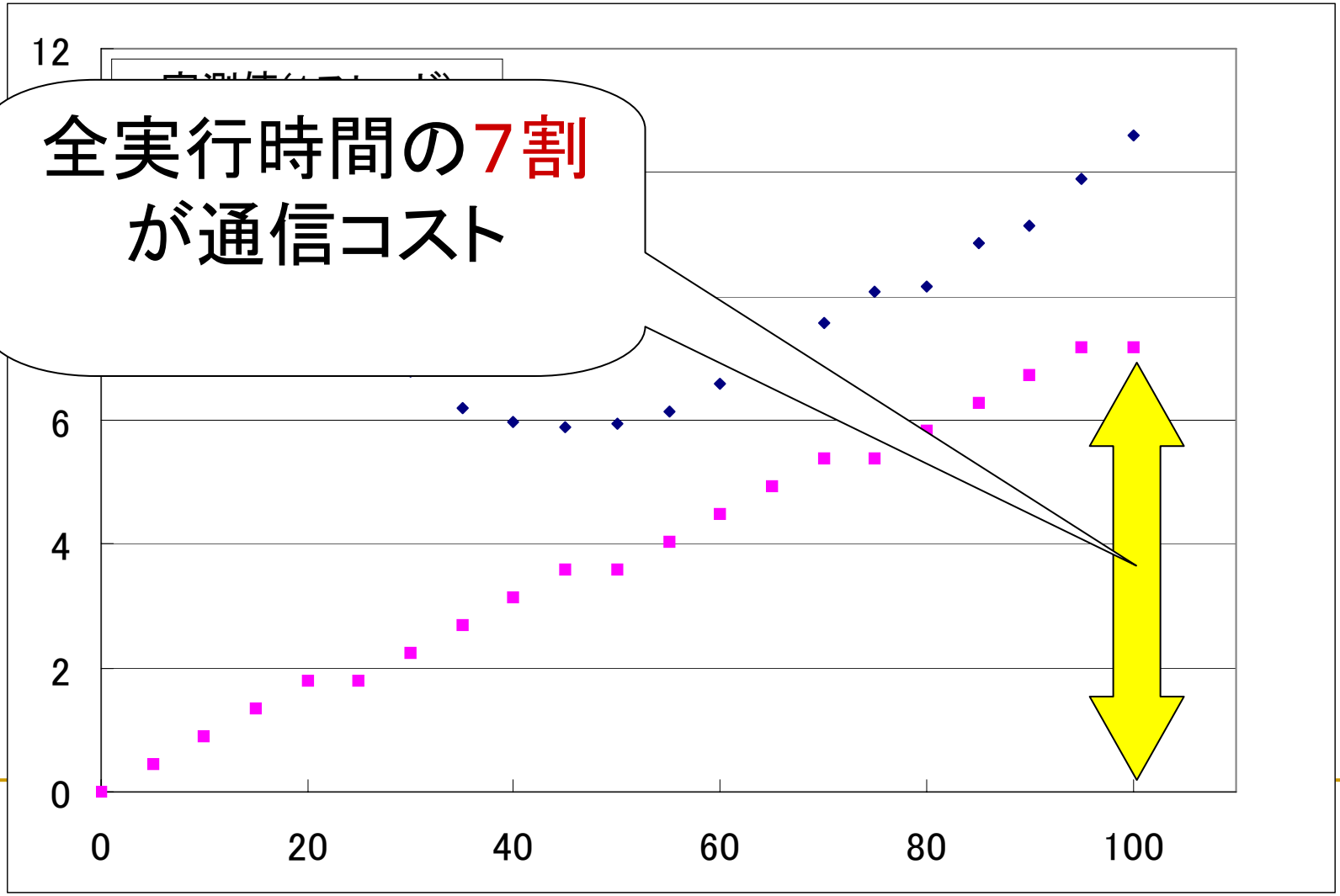
GPUとの通信コスト

- GPUとの通信に掛かるコストが非常に大きい
- 性能モデルからGPUへの転送コストの部分のみ抜き出し



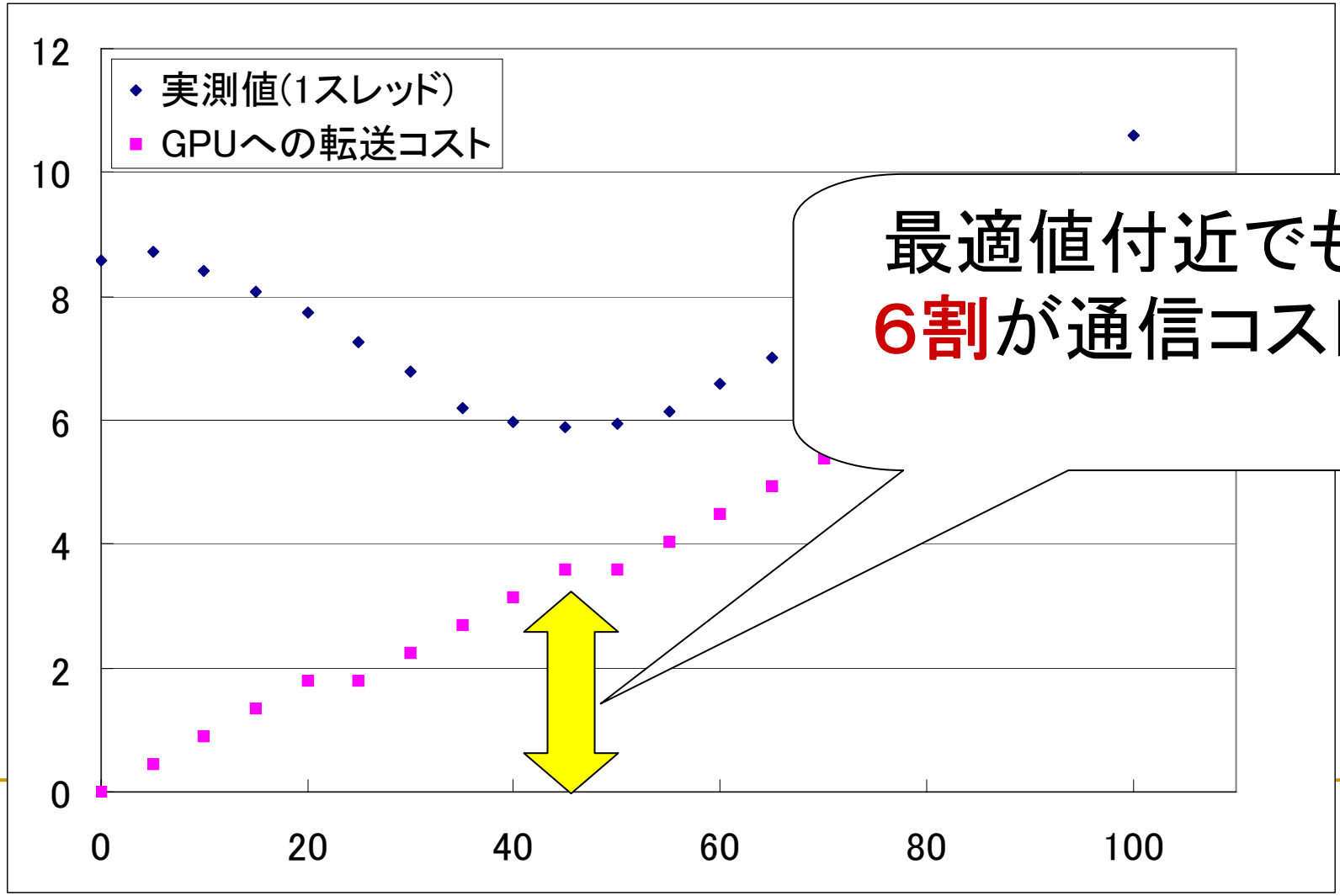
GPUとの通信コスト

- GPUとの通信に掛かるコストが非常に大きい
- 性能モデルからGPUへの転送コストの部分のみ抜き出し



GPUとの通信コスト

- GPUとの通信に掛かるコストが非常に大きい
- 性能モデルからGPUへの転送コストの部分のみ抜き出し



GPUの通信コストの大きさ

- 現状のGPUとの通信コストは大きすぎる

→GPUをライブラリとして扱うことには限界

- ライブラリベースのGPGPUではなく言語統合型のGPGPU（Peak stream, CUDA, Accelerator）
  - これらはGPU内でデータの再活用が可能

まとめ

- CPU/GPUの並列での計算の有効性
- CPUとGPUを並列する2D FFTライブラリ
- 実行時間の性能モデル
  - 予測の際にデータを取得した問題サイズと同サイズの実行時間を予測
  - 予測の際にデータを取得した問題サイズと異なるサイズの実行時間を予測
- GPUの転送コストは非常に大きい

今後の課題

- 性能モデルの正確性の向上
  - CPU・GPUによるお互いの負荷は未検証
  - 問題サイズを跨ぐ予測の精度の向上

- 言語統合型のGPGPUを採用し、通信コストの削減