性能解析ツールScalascaについて

2014年3月7日
中村朋健
理化学研究所 計算科学研究機構

K computer

性能解析ツール(プロファイラ)

性能解析ツールとは？

- プログラムの詳細な実行性能を取得するソフトウェア
- 取得した実行性能をわかりやすく表示するソフトウェア

いつ性能解析ツールを使うのか？

- プログラムを解析して効率のよいプログラムを書きたいとき
  - 計算機の資源を活かしたプログラム
  - 計算機の特性を活かしたプログラム

なぜ性能解析ツールを使うのか？

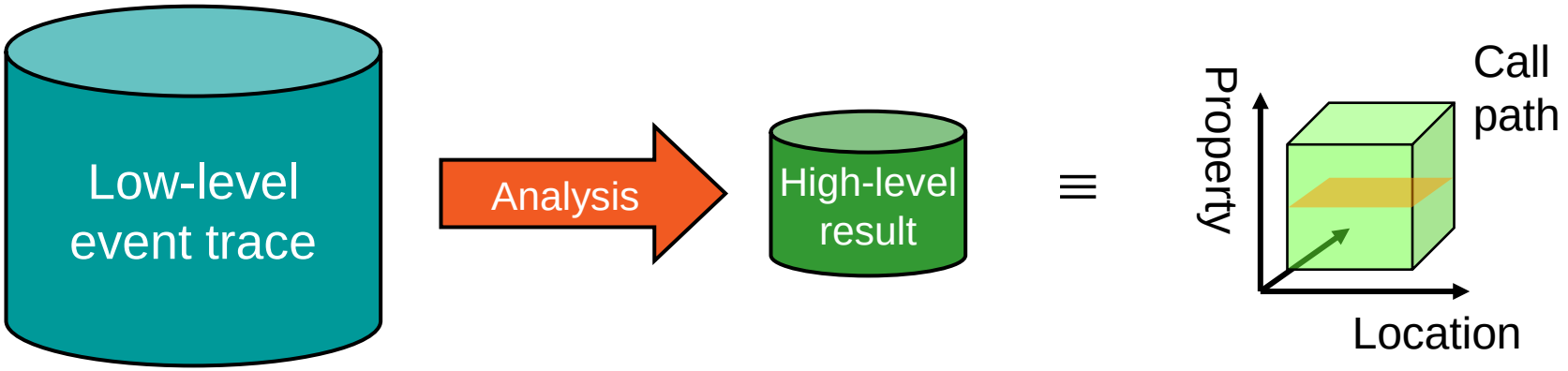
- 楽をしてプログラムを解析したいため
  - 解析ツールがプログラムの詳細な実行性能を提供
    - タイム関数の挿入などのコード改変による性能解析は面倒
    - プログラムの実行時間だけではわからないことが多い
  - 複数の計算機上の実行性能を容易に把握可能

性能解析ツールのサマリ

- 性能解析
  - 収集したメトリックの大まかな分析
  - ほとんどのツールは関数/コールパスやプロセス/スレッド単位での解析が可能
  - gprof, mpiP, ompP, Scalasca, TAU, Vampirがよく使われる
- タイムライン解析
  - イベントを時系列に可視化
  - 実行トレースデータが必要
  - Vampir, Paraver, JumpShot, Intel TAC, Sun Studioがよく使われる
- パターン解析
  - 非効率的な部分を検出
  - タイムライン解析を使って手動で検出可能
  - 自動ではKOJAK, Scalasca, Periscopeがよく使われる

Scalasca

- アイデア
  - 非効率的なプログラムの振る舞いのパターンを自動検索
  - プログラムの振る舞いの分類や重要性を定量化



  - 性能解析結果の詳細(metric)
  - Call-treeパス(program location)
  - システム・ロケーション(process/thread)

- すべてのイベントトレースを網羅していることを保証
- 可視化によるマニュアルトレース解析より早い自動解析

Scalascaプロジェクト

- 今の開発チーム
  - Juelich Supercomputing Centre
    (この資料はここの方に頂いたものを参考に作成)
  - German Research School for Simulation Sciences

- Scalascaプロジェクトの目的
  - 主要なプログラミングパラダイムに対してスケーラブルな性能解析ツールを開発すること
  - 具体的なターゲットアプリケーションは
    - IBM BlueGene やCray XTなどで動く100万プロセスまたはスレッド以上で動くアプリケーション

Scalascaの特徴

- オープンソース+ New BSD ライセンス
- 移植先
  - IBM BlueGene, IBM SP & blade clusters, Cray XT, SGI Altix, Fujitsu FX10 & K computer, NEC SX, Intel Xeon Phi, Solaris & Linux clusters, ...
- 並列プログラミングパラダイム＆言語
  - MPI, OpenMP, これらのハイブリッド
  - Fortran, C, C++
- インストルメント, 実行, 分析の統合ツールセット

- 最新版
  - v1.4.3（2013年3月）
    - 「京」用にポーティングされたもの
  - 現在Score-Pで開発中のv2.xxをポーティング中

Scalascaのワークフロー

Measurement library

HWC

Instr. target application

Optimized measurement configuration

Summary report

Report manipulation

Local event traces

Parallel wait-state search

Wait-state report

Scalasca trace analysis

Instrumented executable

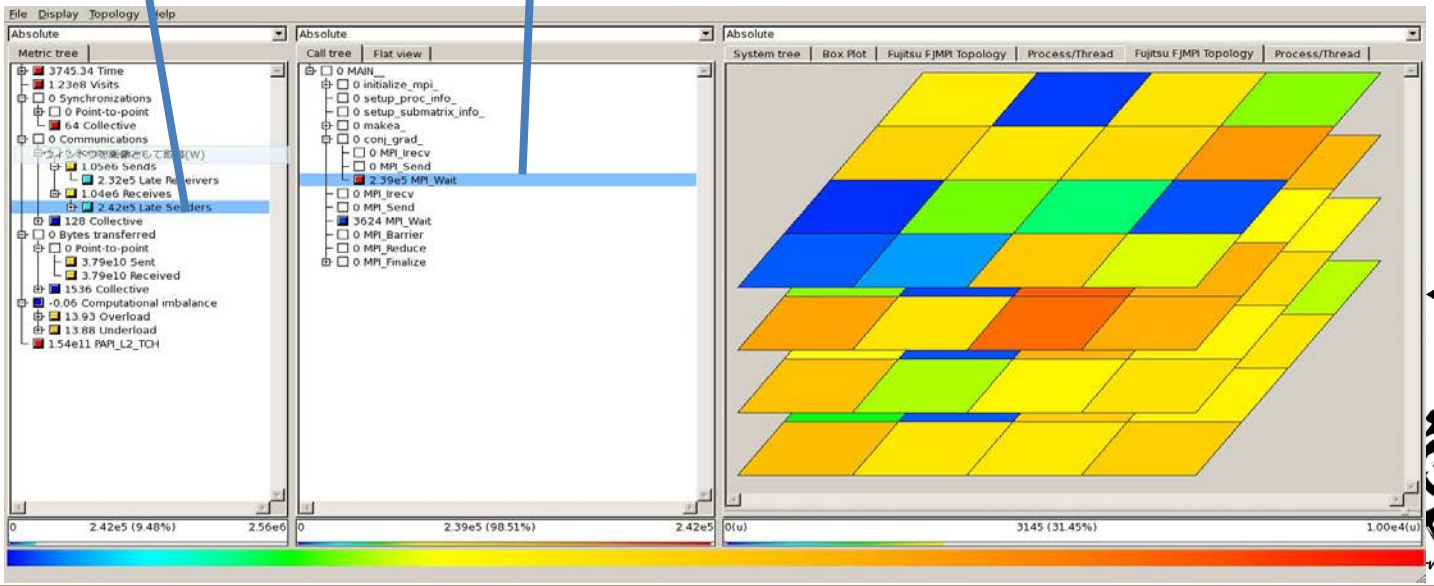
Instrumenter compiler/ linker

Source modules

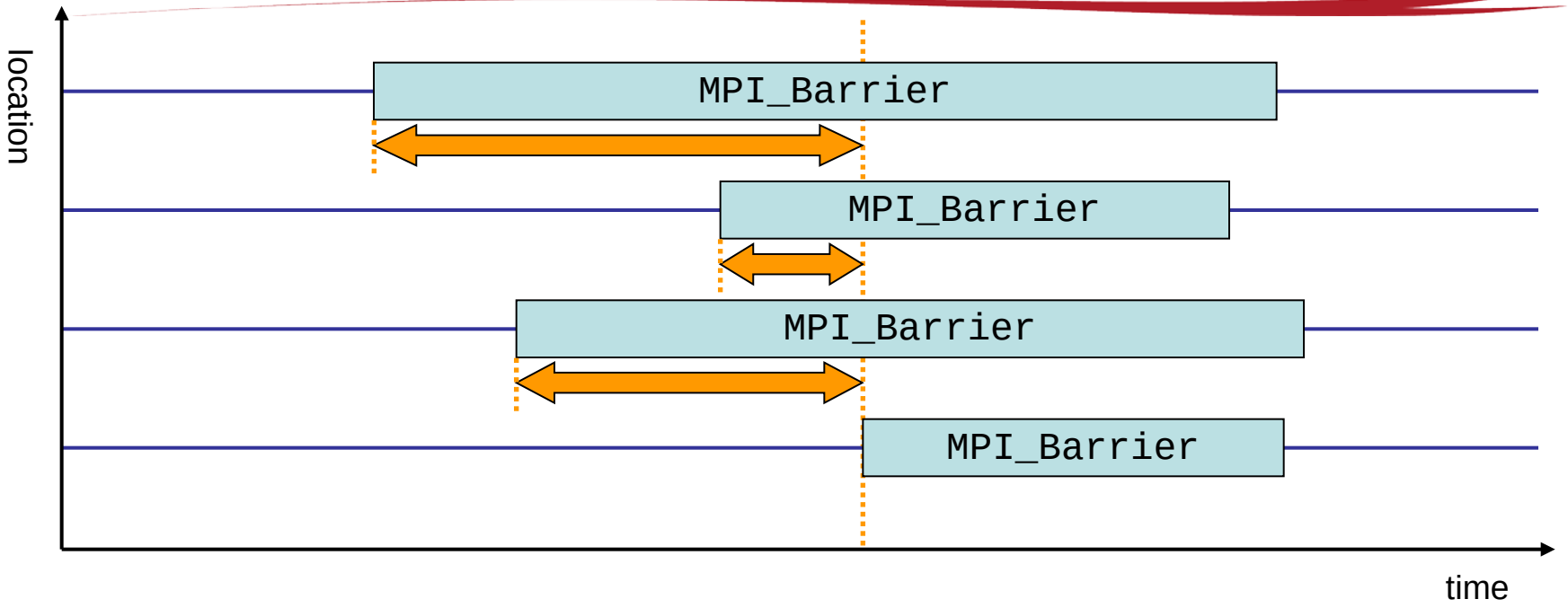
Which problem?

Where in the problem?

Which process?

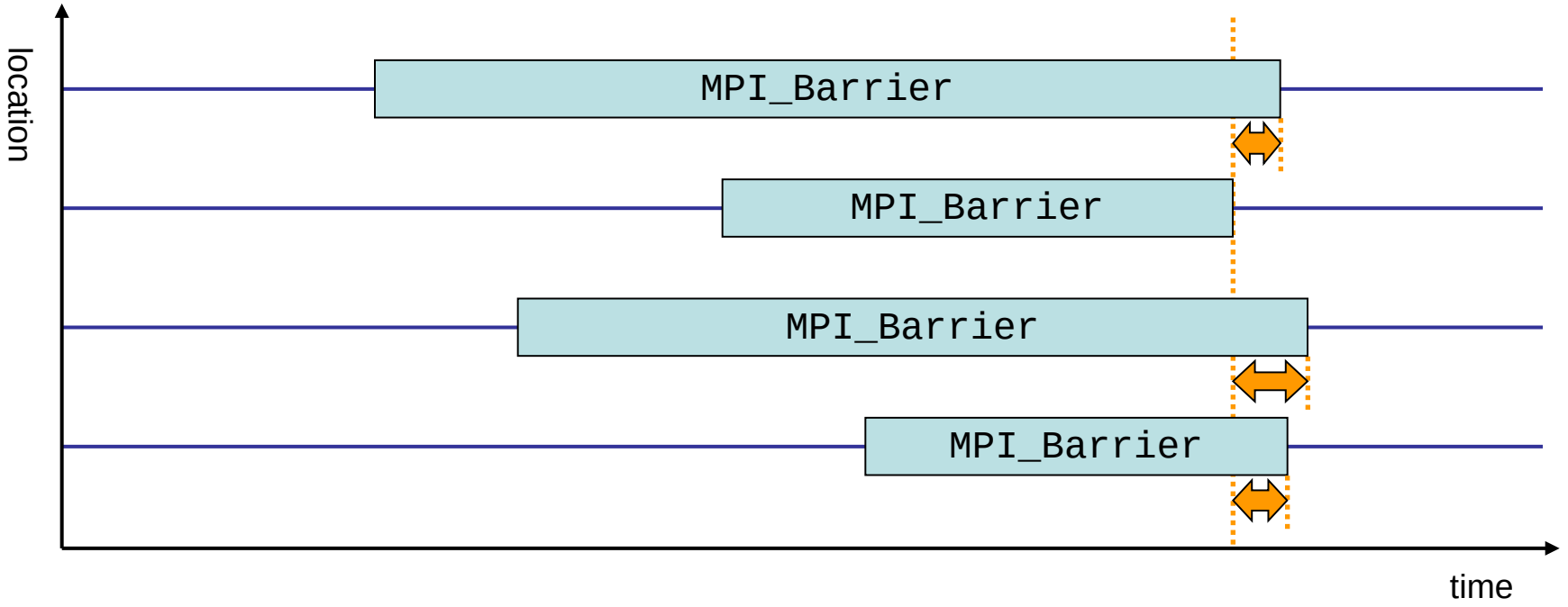


検出可能な待ち時間 (MPI->Sync->Collective)
Wait at barrier



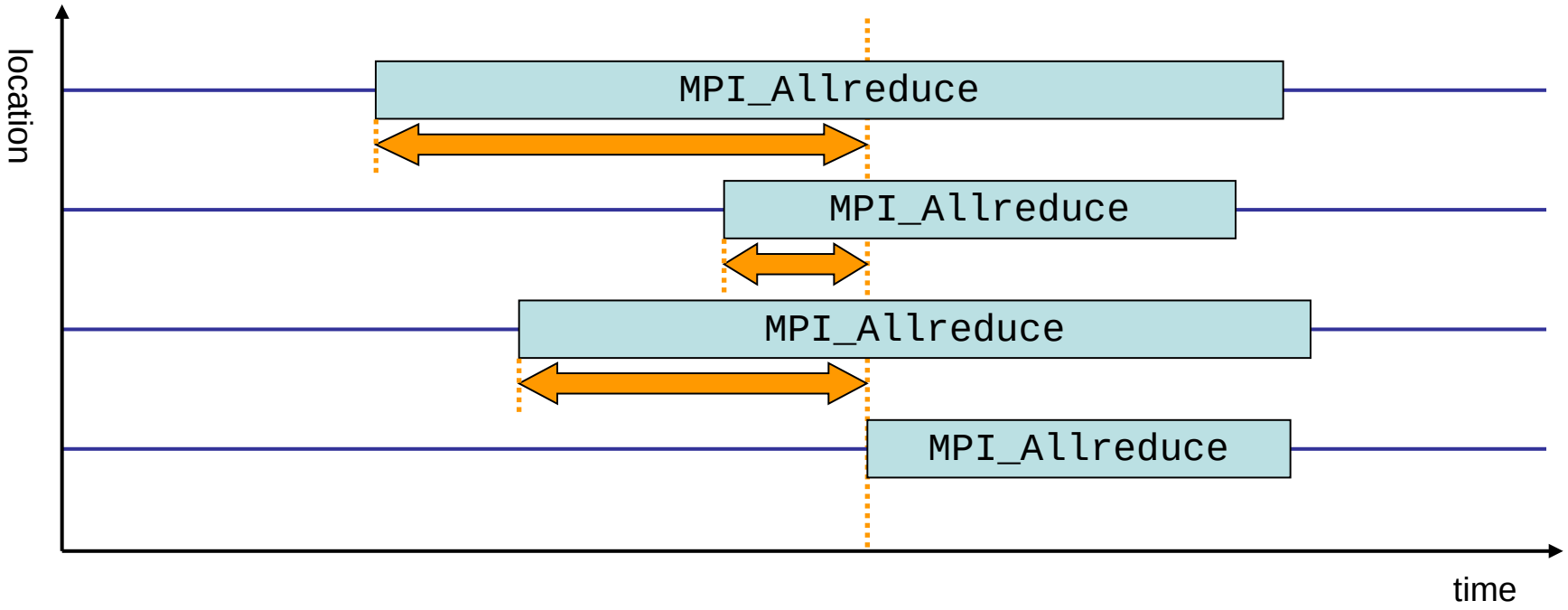
- 最後のプロセスがバリアに到達するまで，他のプロセスが待つ時間
- 表示ツールCUBEのMPI_Barrier部分に計上

検出可能な待ち時間(MPI->Sync->Collective) Barrier Completion



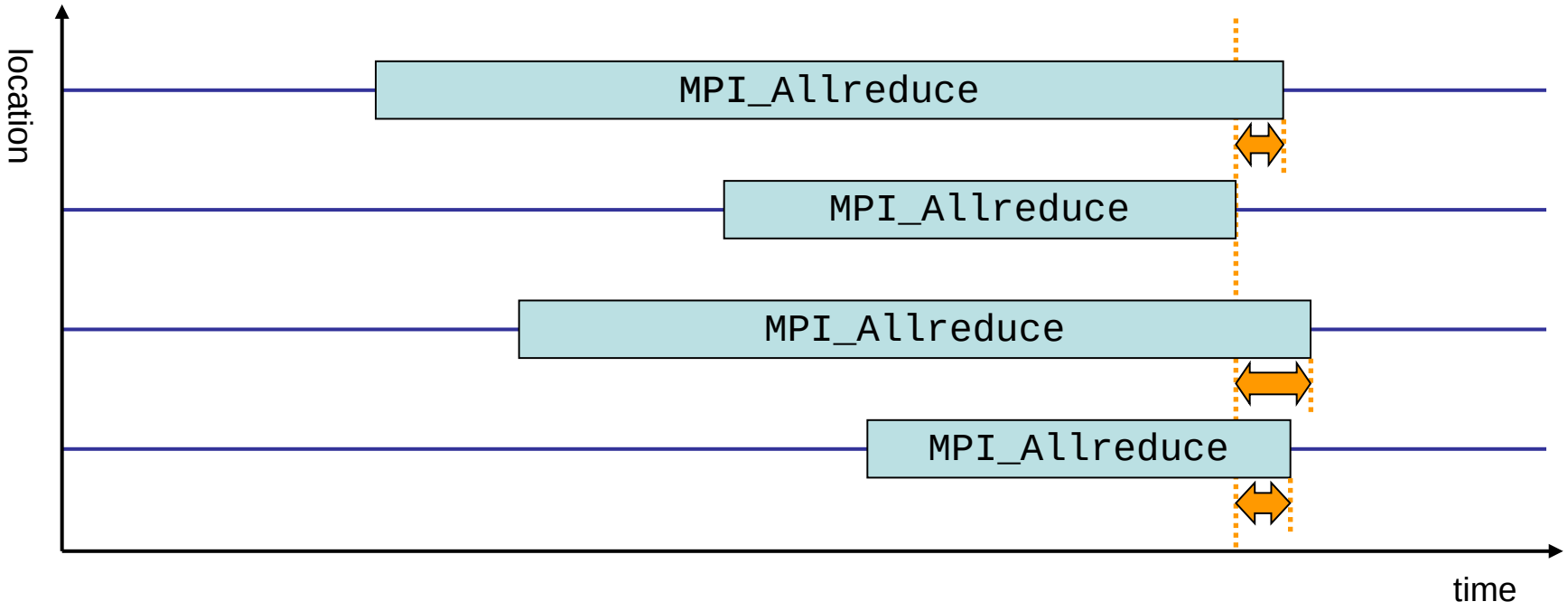
- 最初にあるプロセスがバリアが終わってから，他のプロセスがバリアが終わるの待つ時間
- 表示ツールCUBEのMPI_Barrier部分に計上

検出可能な待ち時間(MPI->comm->collective)
Wait at N x N



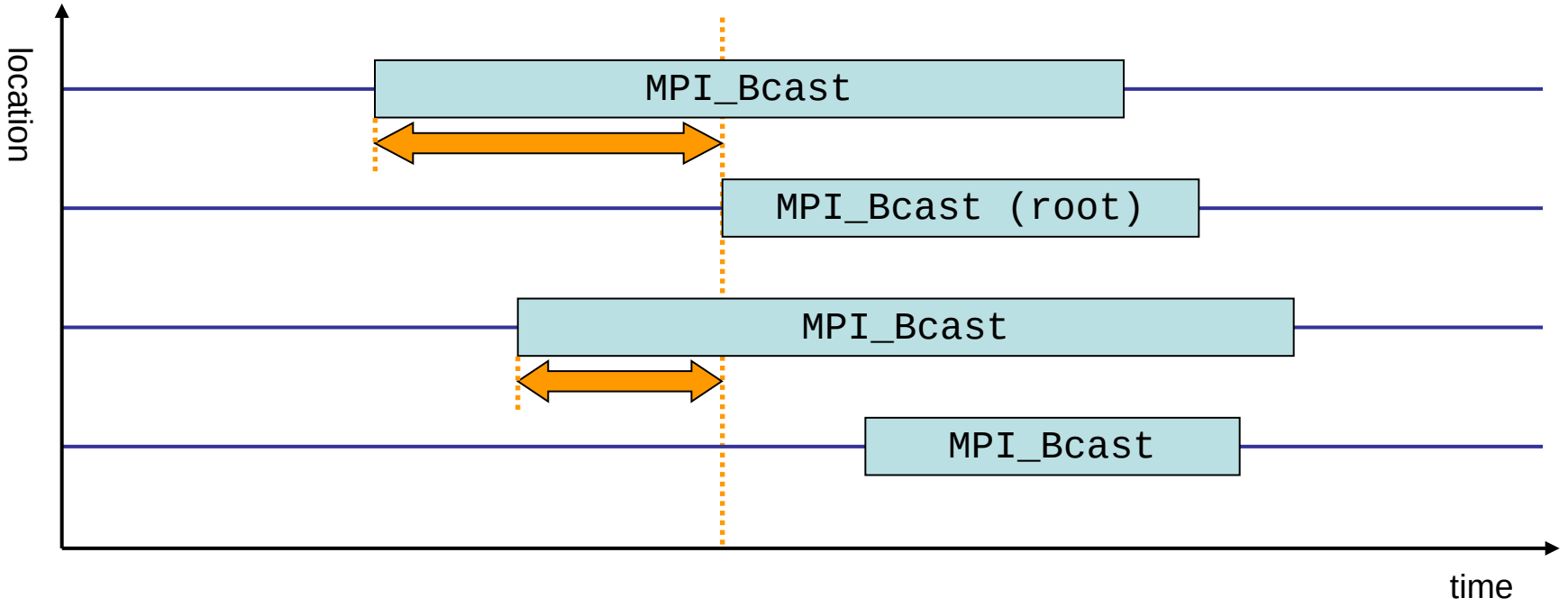
- 最後のプロセスがcollectiveオペレーションの同期に到達するまで，他のプロセスが待つ時間
- 表示ツールCUBEのMPI_Allgather, MPI_Allgatherv, MPI_Allreduce, MPI_Alltoall, MPI_Reduce_scatter, MPI_Reduce_scatter_block部分に計上

検出可能な待ち時間(MPI->comm->collective)
N x N Completion



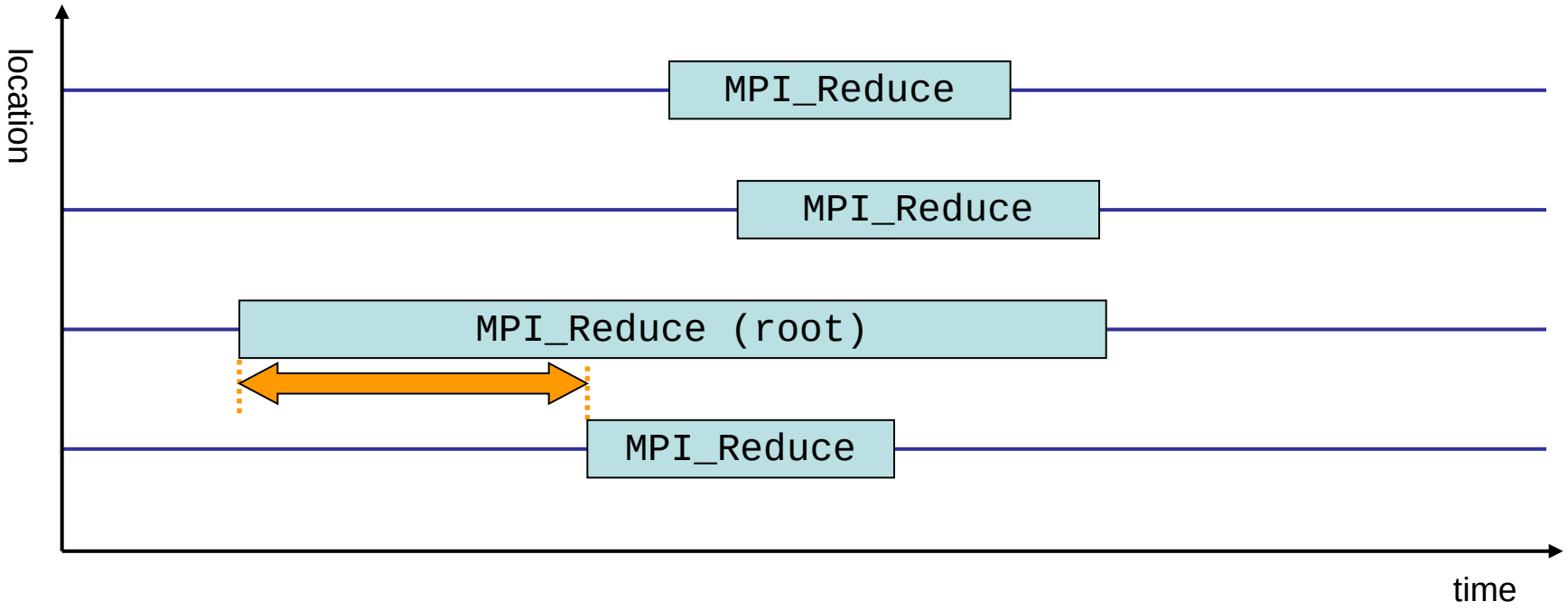
- 最初にあるプロセスがcollectiveオペレーションの同期が終わってから，他のプロセスが終わるのを待つ時間
- 表示ツールCUBEのMPI_Allgather, MPI_Allgatherv, MPI_Allreduce, MPI_Alltoall, MPI_Reduce_scatter, MPI_Reduce_scatter_block部分に計上

検出可能な待ち時間(MPI->comm->collective)
Late Broadcast



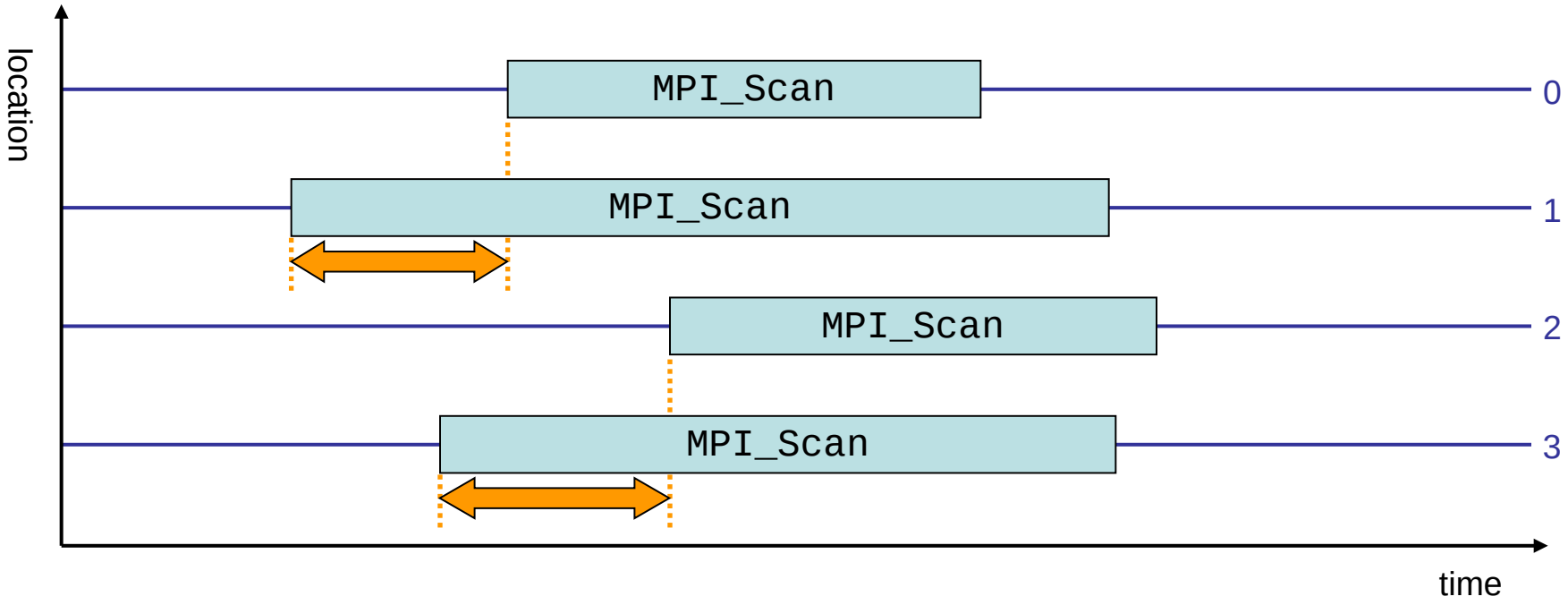
- 1対Nのcollective通信でソース(root)となるプロセスより，他のプロセスが早く到達して待つ時間
- 表示ツールCUBEのMPI_Bcast, MPI_Scatter, MPI_Scatterv部分に計上

検出可能な待ち時間(MPI->comm->collective)
Early Reduce



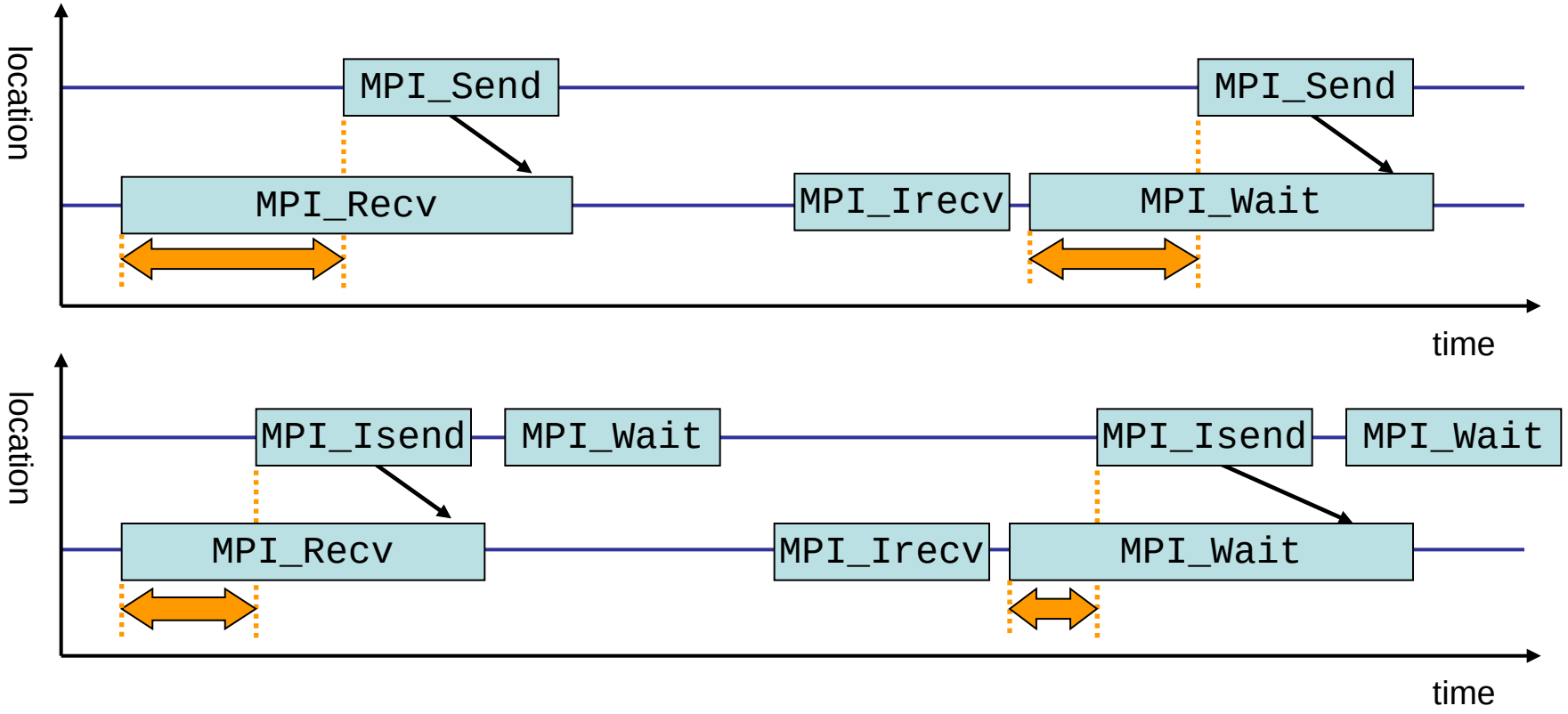
- N対1のcollective通信でソース(root)となるプロセスが，他のプロセスから転送が始まるまで待つ時間
- 表示ツールCUBEのMPI_Reduce, MPI_Gather, MPI_Gatherv部分に計上

検出可能な待ち時間(MPI->comm->collective)
Early Scan



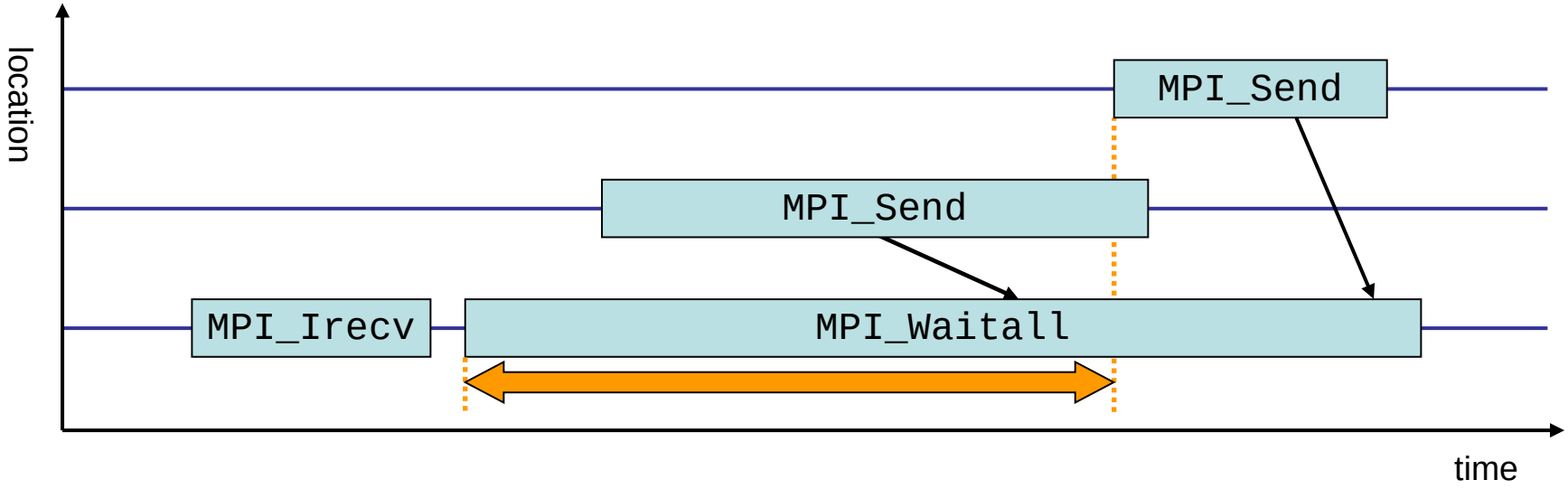
- 受けとるプロセスがソースプロセスのオペレーション開始を待つ時間
- 表示ツールCUBEのMPI_Scan, MPI_Exscan部分に計上

検出可能な待ち時間(MPI->comm->P2P)
Late Sender (1)



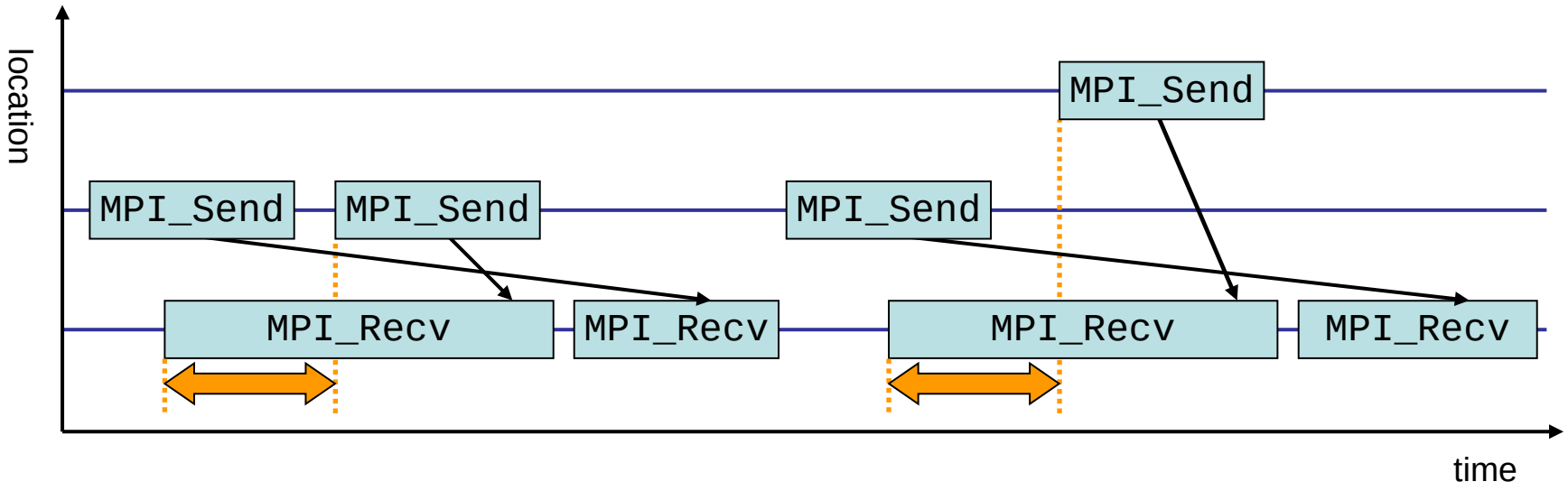
- 受け取るプロセスのReceiveやWaitがソースのSendオペレーションの開始を待つ時間

検出可能な待ち時間(MPI->comm->P2P) Late Sender (2)



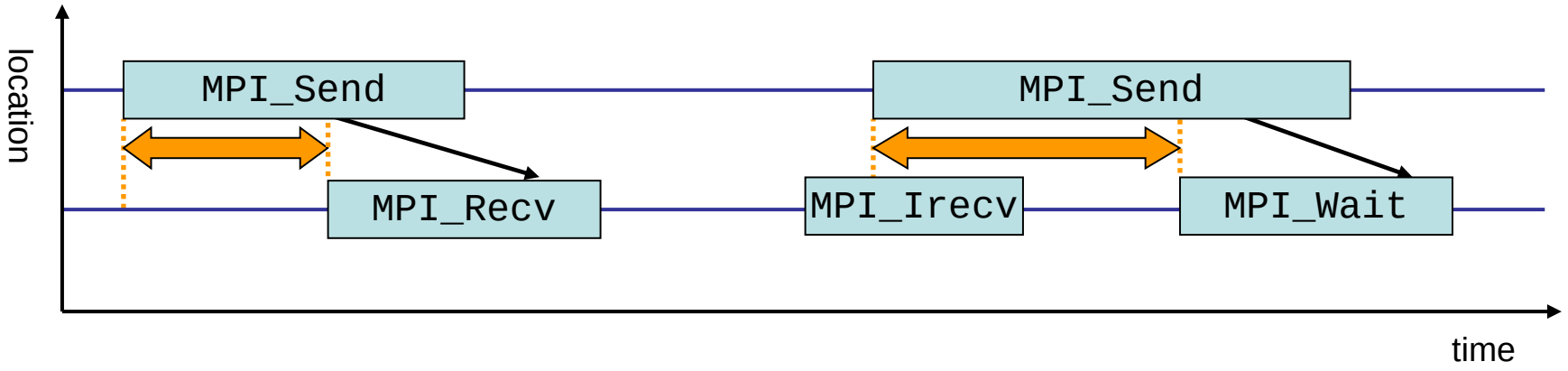
- WaitオペレーションがSendの開始が最も遅いプロセスのSendオペレーション開始を待つ時間

検出可能な待ち時間(MPI->comm->P2P)
Late Sender, Messages in Wrong Order



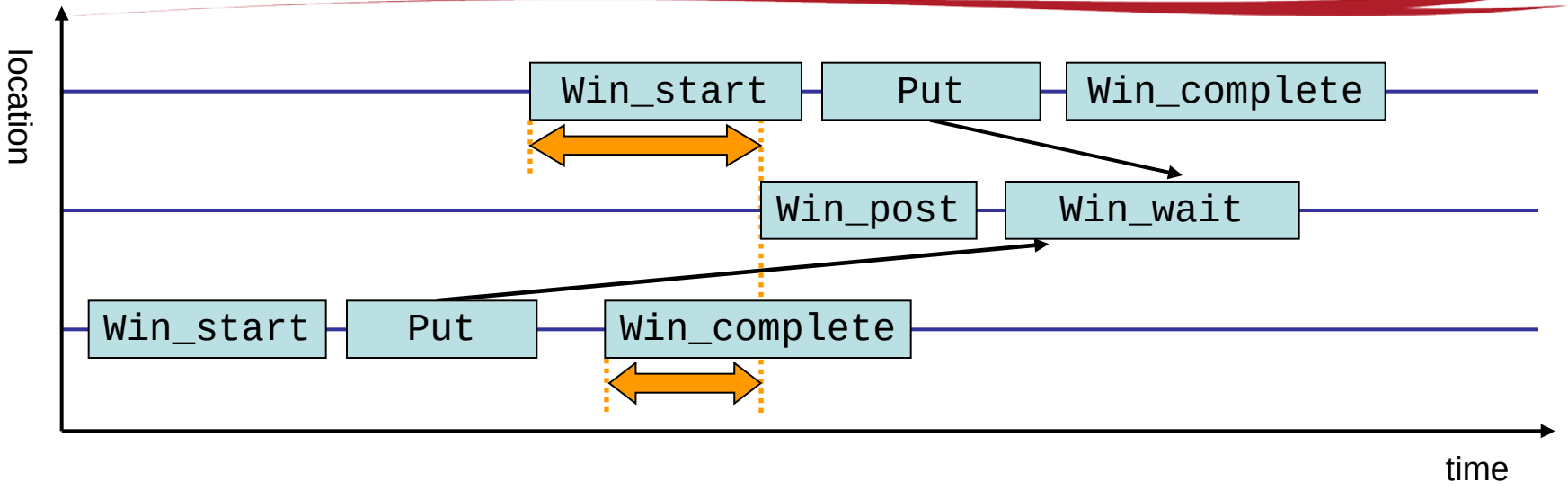
- Sendすべき順番が間違っているときの，Receiveオペレーションが間違ったSendオペレーションの開始を待つ時間

見つけられる待ち (MPI->comm->P2P) Late Receiver

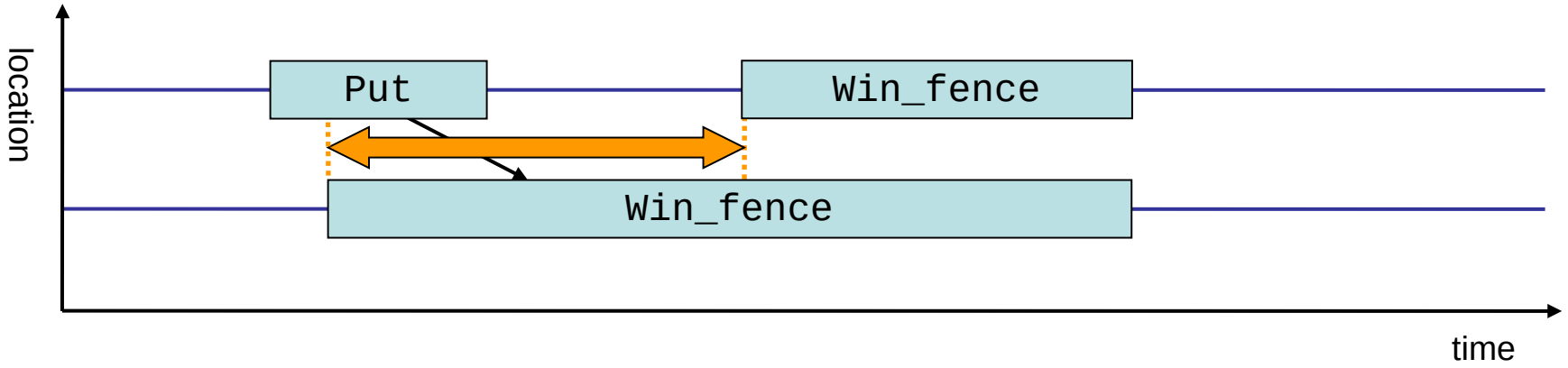


- SendオペレーションがReceiveやWaitオペレーションの開始を待つ時間

見つけられる待ち (MPI->Sync->RMA)
Late Post

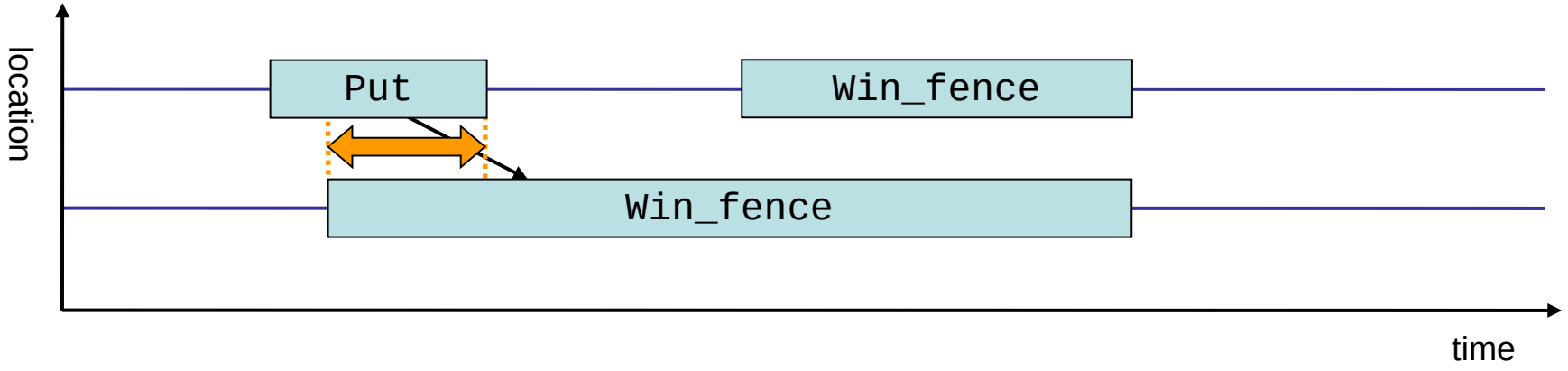


見つけられる待ち (MPI->Sync->RMA)
Wait in Fence



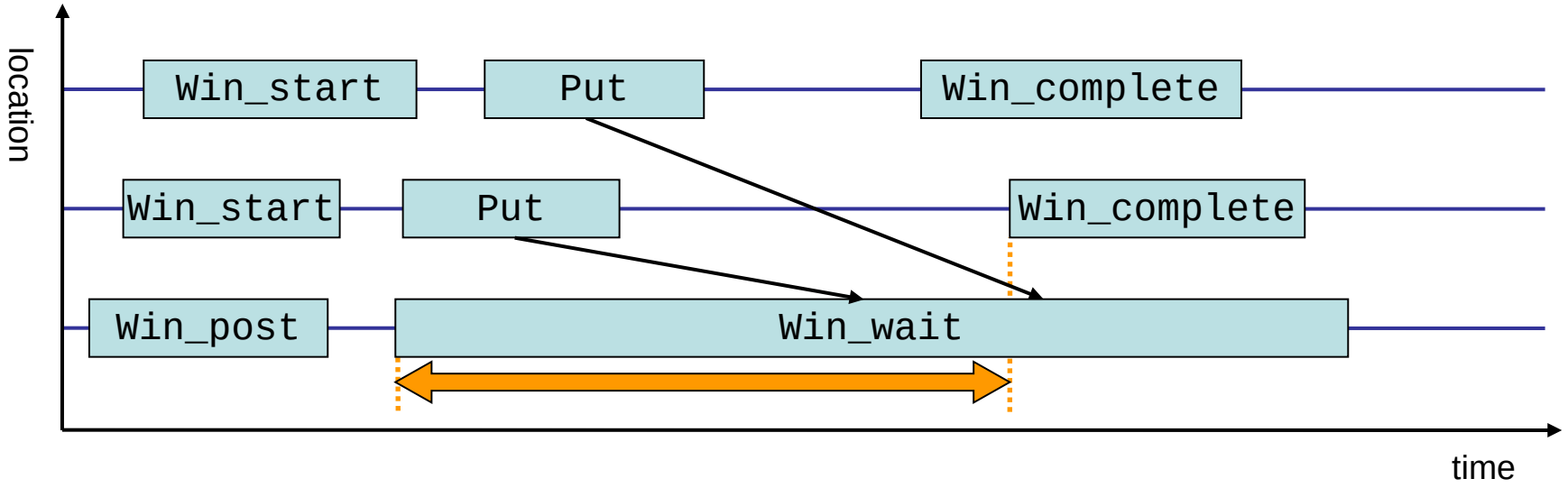
- MPI_Win_fence同期で最初に到達したものが最後に到達したプロセスを待つ時間

見つけられる待ち (MPI->Sync->RMA) Early Fence

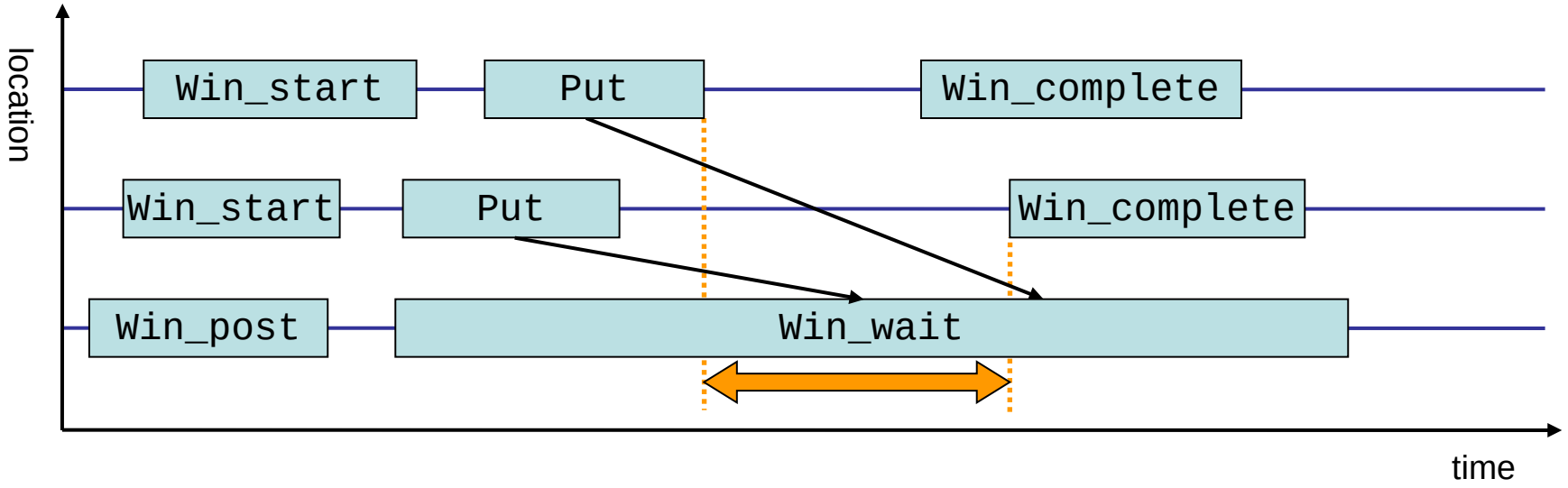


- Win_Fenceがそれ以前に終わっていないRMAオペレーションの終わりを待つ時間

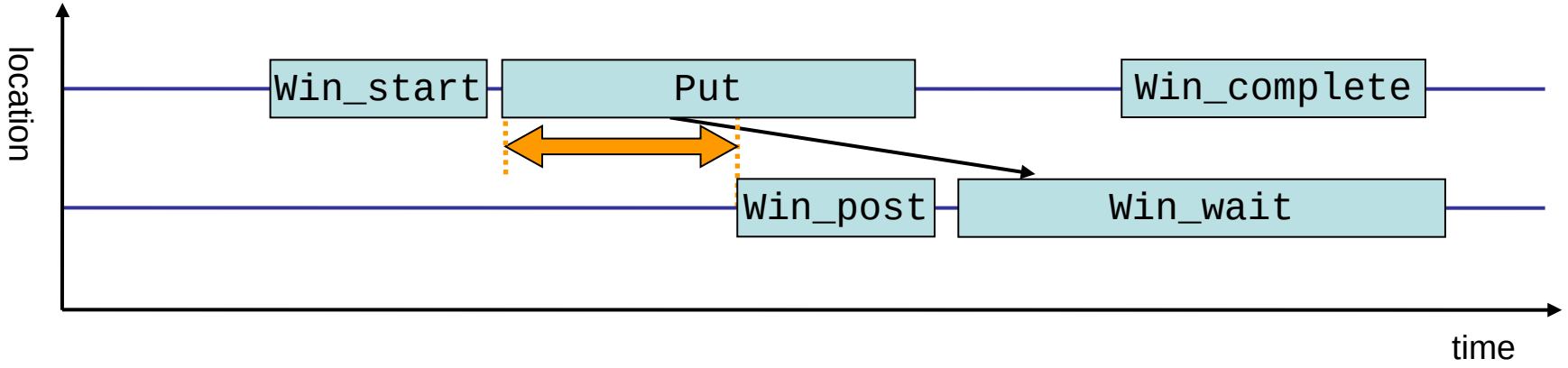
検出可能な待ち時間(MPI->Sync->RMA)
Early Wait



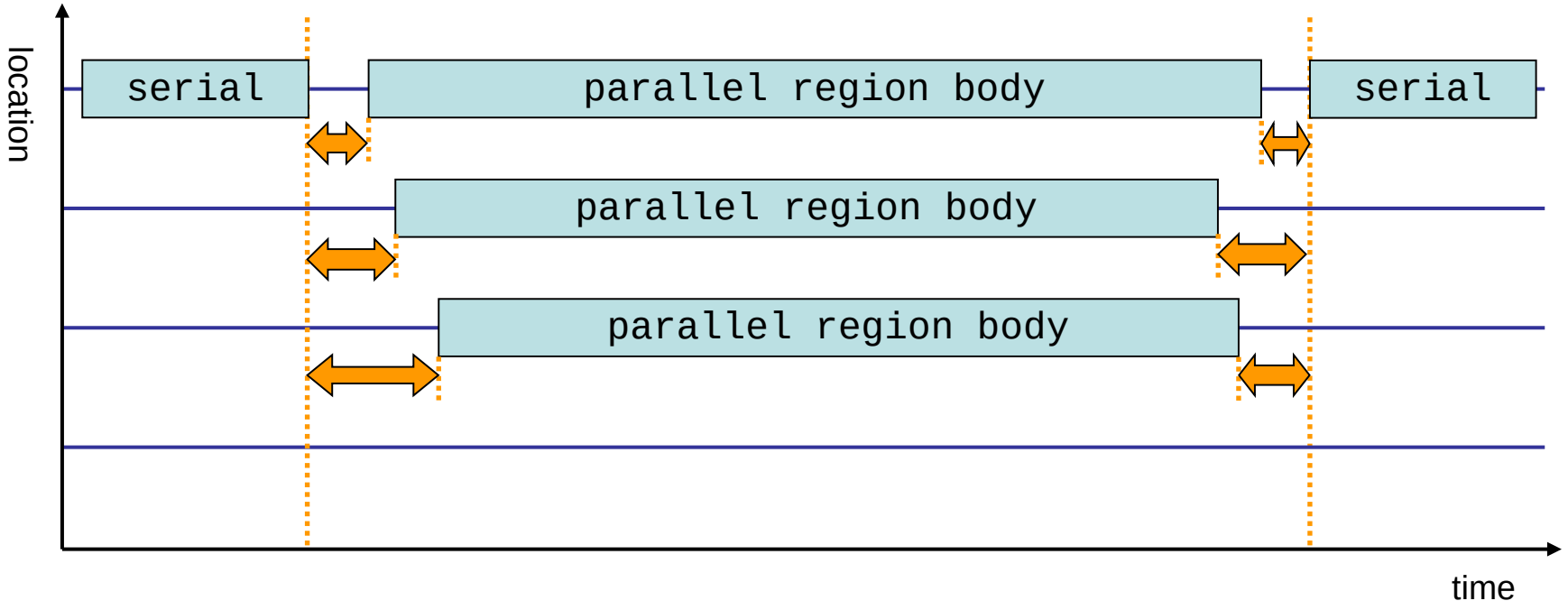
検出可能な待ち時間(MPI->Sync->RMA)
Late Complete



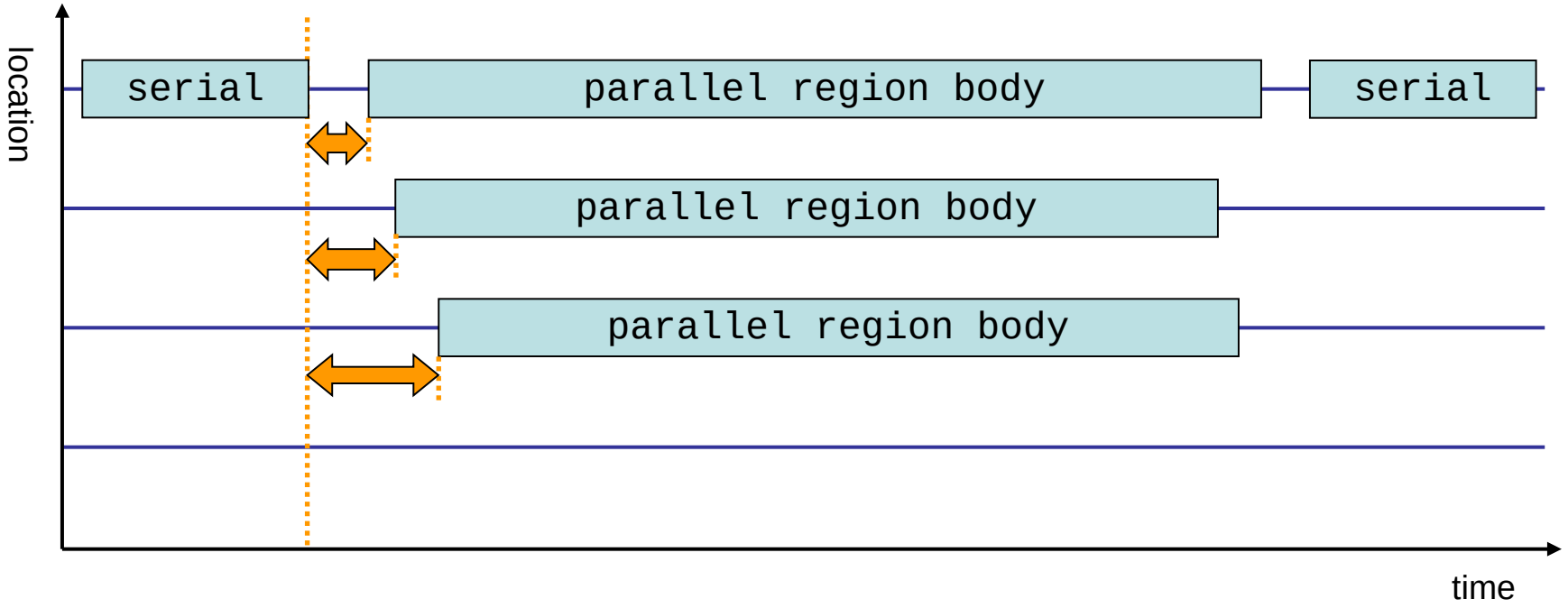
検出可能な待ち時間(MPI->Comm->RMA)
Early Transfer



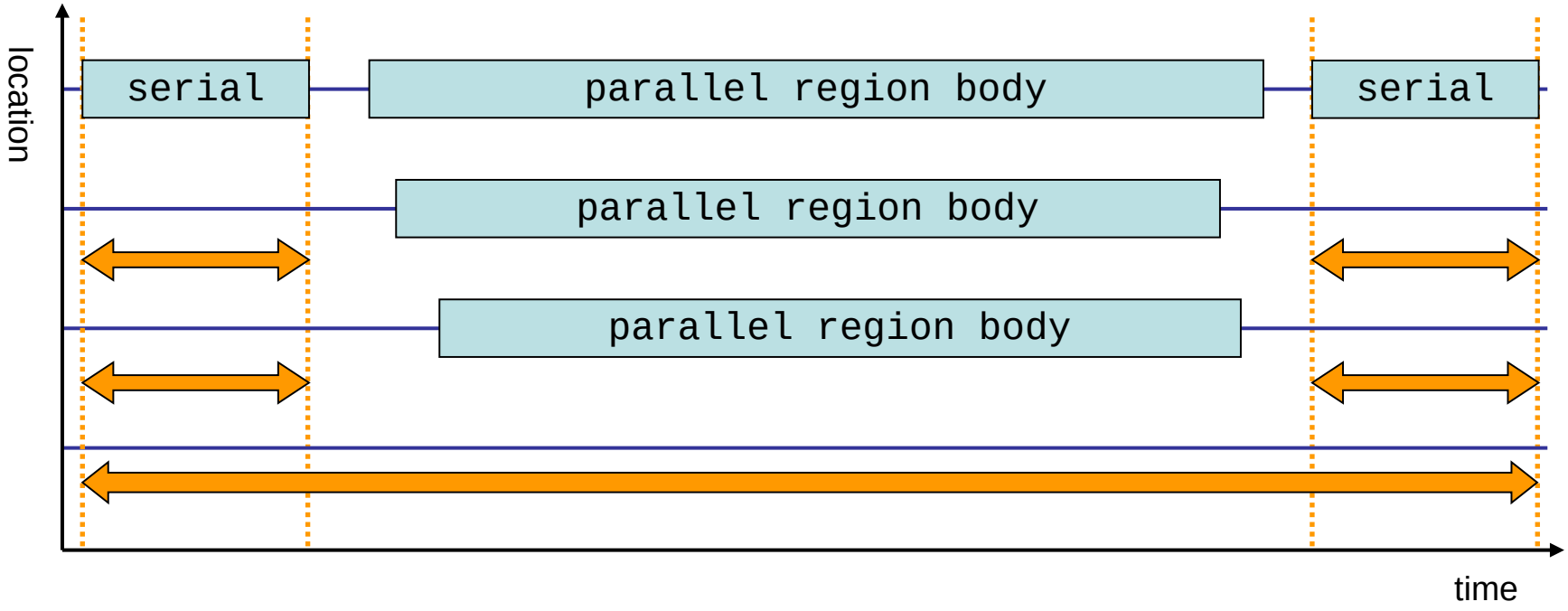
検出可能な待ち時間(Time->Exec->OMP)
OpenMP Management Time



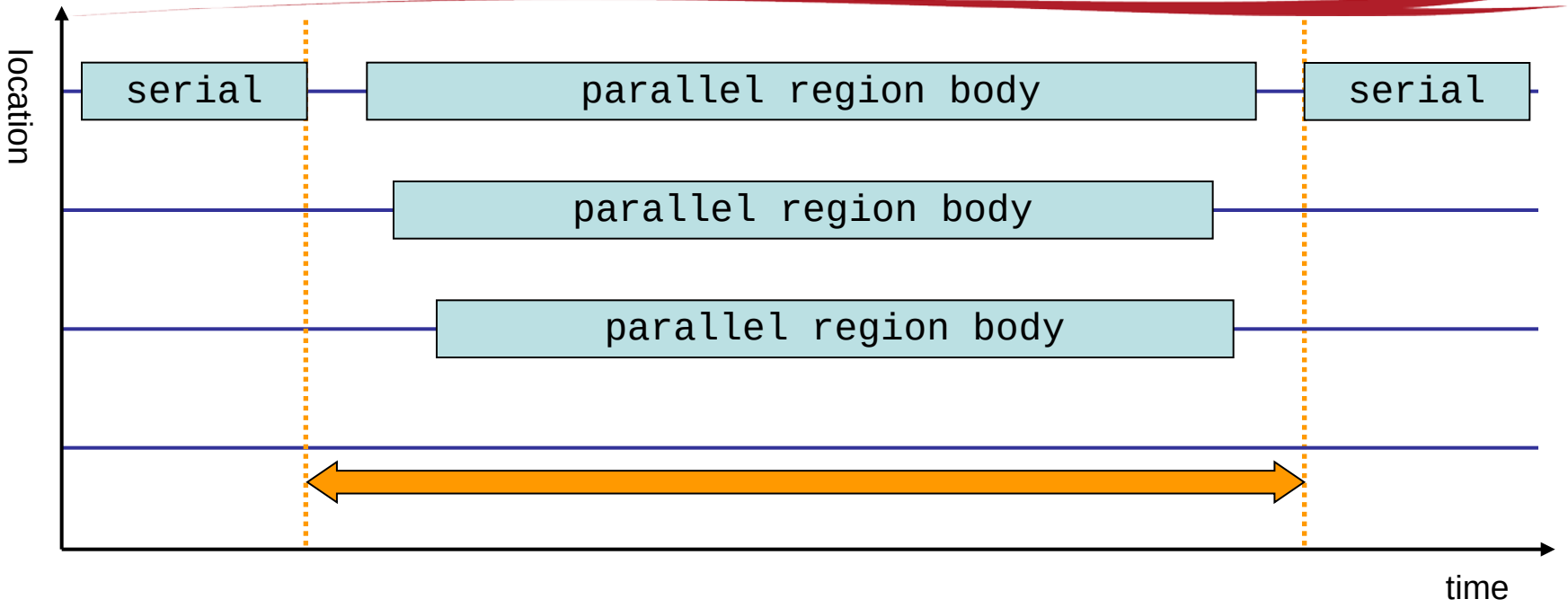
検出可能な待ち時間(Time->Exec->OMP)
OpenMP Fork Time



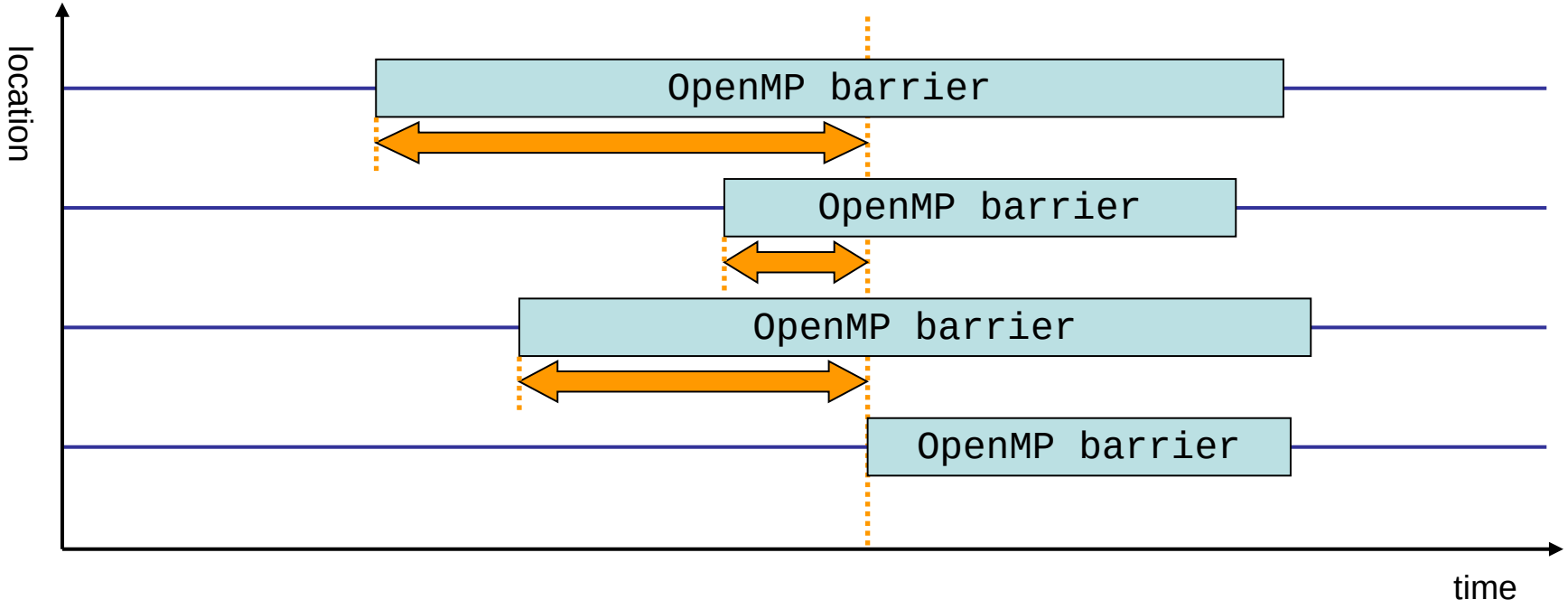
検出可能な待ち時間(Time->Exec->OMP)
OpenMP Idle Threads



検出可能な待ち時間(Time->Idle Threads)
OpenMP Limited Parallelism



検出可能な待ち時間(OMP->Sync->Barrier)
Wait at Barrier



検出可能な待ち時間(OMP->Sync->Lock)
Lock Completition