

構造化オーバーレイネットワークの 動的適応性向上

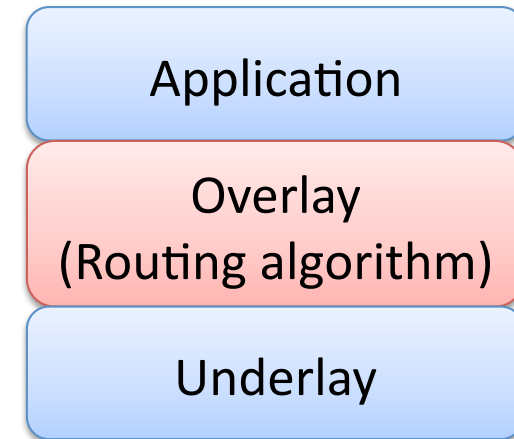
野口 悟

奈良先端科学技術大学院大学
情報科学研究科 M2

第2回 広域センサネットワークとオーバーレイネットワークに関するワークショップ
2008/11/1

構造化オーバーレイの多様性

- 構造化オーバーレイ
→ アプリケーションの基盤
- 多数の処理方式
 - トポロジ
 - 探索方式
 - データ配置方式
- アプリケーションの要求は多様
 - latency, bandwidth, durability, availability...
- 利用環境も多様
 - ノード数、データ量、アクセス頻度、churn....



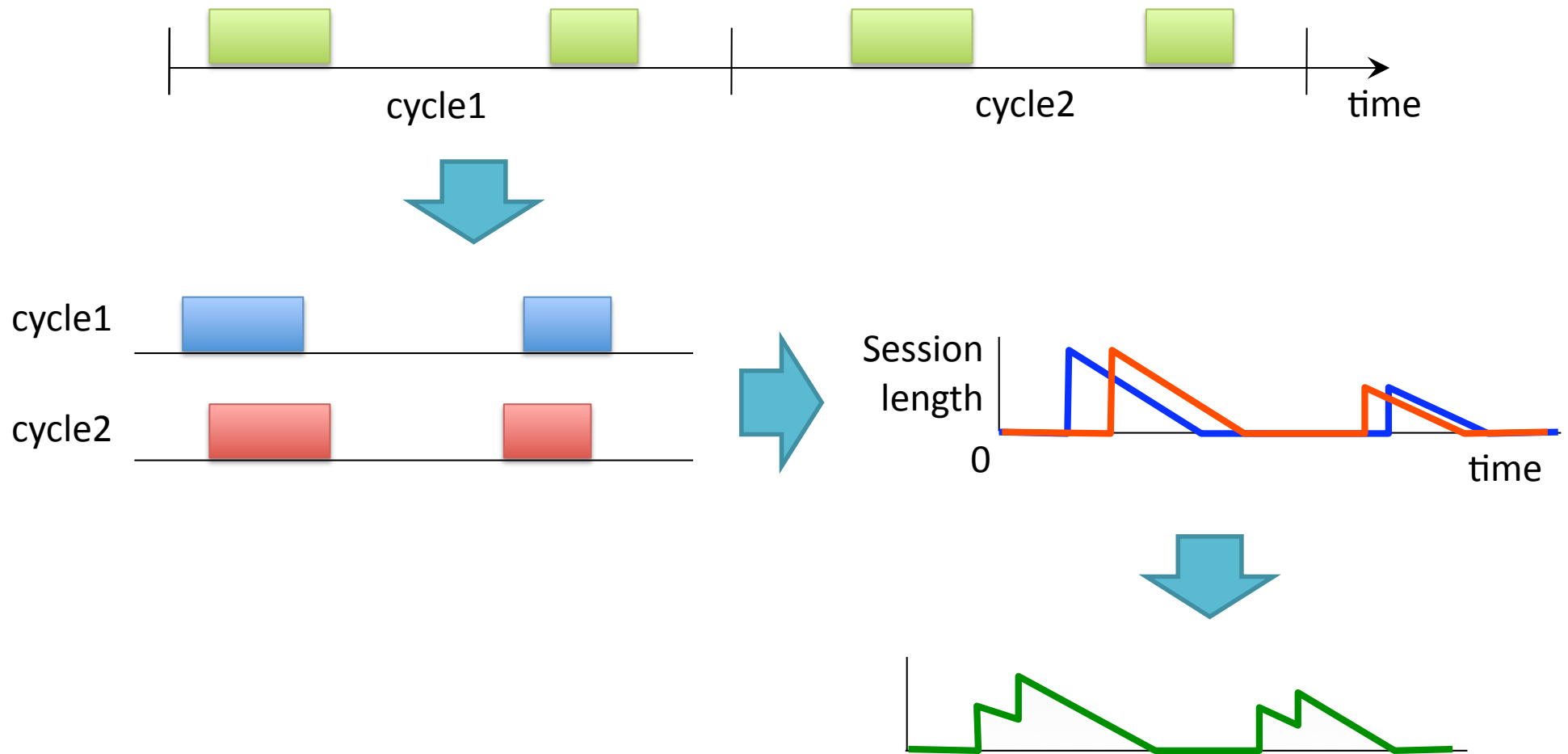
適応性の向上

- アプリケーションと処理方式のマッチング
- 多様な利用状況への適応
 - 静的・画一的設定はP2Pに不向き
 - 変動する環境下で動的な設定調整を行う必要がある
 - 特にchurn
- churn への動的適応
 - データ配置：消失防止、遅延低減、維持コスト低減
 - ノード探索：遅延低減、探索コスト低減

アプローチ：ノード選択の改善

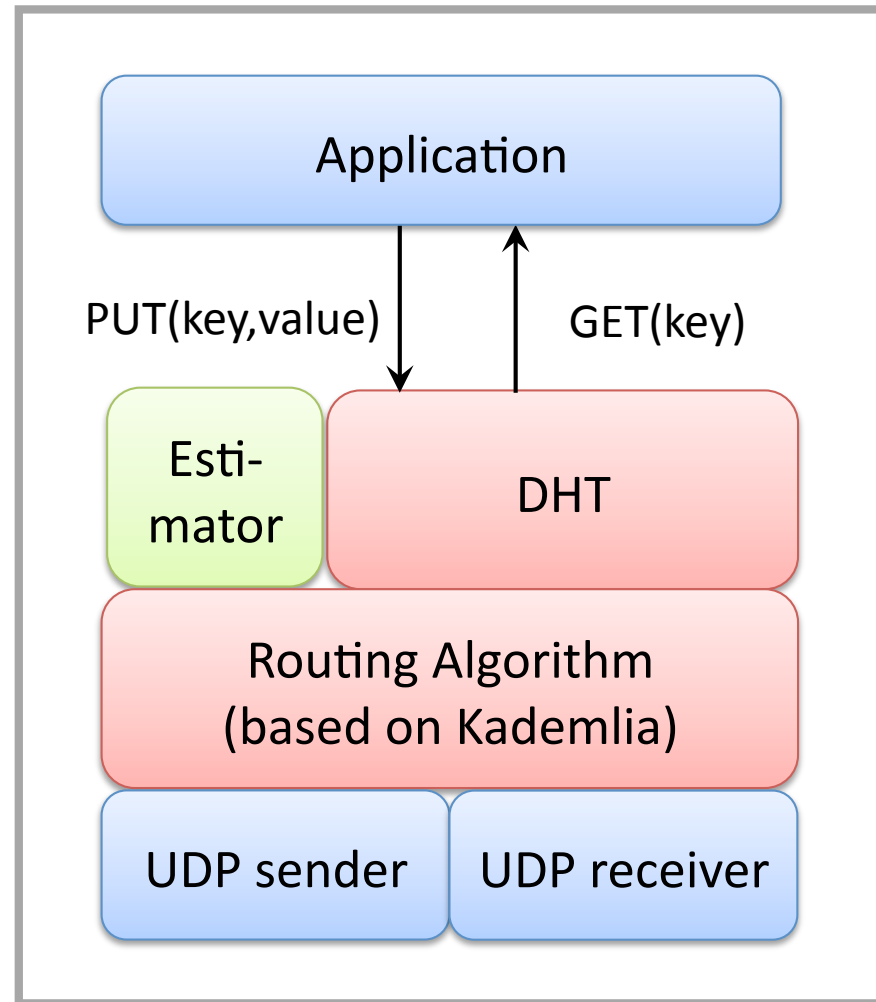
- データ配置：任意のノードを**選択**、データ配置
- ノード探索：次hopの**選択**
- 要求に応じたノード選択を行う
- ノード選択の基準
 - 基本：ID
 - Network：Bandwidth, RTT, 下位層のトポロジ情報
 - Machine：Storage, CPU
 - **User：Session Time** (過去 or **未来**)
 - Random

Session - length estimation



Architecture

- Java SE6
- KBR-based abstraction
- Non-blocking I/O
- DHT
 - JOIN (firstNodeSocket)
 - PUT (key, value)
 - GET (key)
- Routing Algorithm
 - Based on Kademlia
 - Route (destSocket, Message)
 - Route (destID, Message)
 - Concurrent query
 - Recursive lookup (with ACK)



Problem

- Implementation
 - Lookup : Iterative or Recursive
 - IO : Blocking or Non-blocking
 - Storage : On-memory or File or DB
- Evaluation
 - how to emulate
 - scheduling