データベースシステム
バッファマネージャ・外部ソート

これまでのコストの考え方

- メモリに読み込むページは、すぐ直前に読み込んだページであろうと関係なく、もう一度読み込む
  - 非現実的なコスト結果が出る場合があったのはそのため

実際はOSの場合と同じく、一度読み込んだページは再度読み込む可能性がある場合、メモリ上に保管しておく仕組みをDBMSは用意している

バッファマネージャ

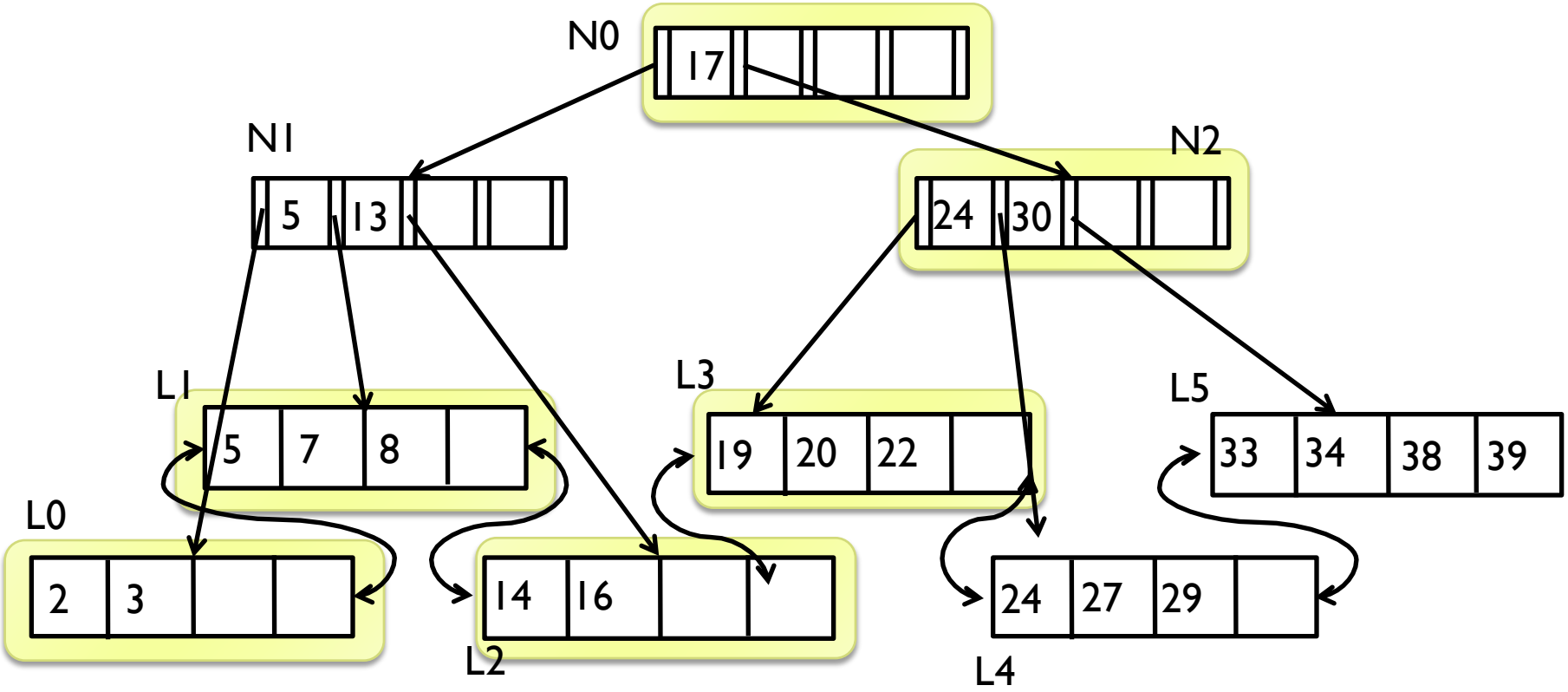
バッファマネージャ

共有メモリにページを確保する領域（バッファプール）を用意し、よくアクセスされるページはそこからアクセスできるようにすることで、アクセスを高速化する

バッファマネージャを使うことによって、特に索引のノードはバッファプールに常に置いておかれることになり、問合せが高速化される

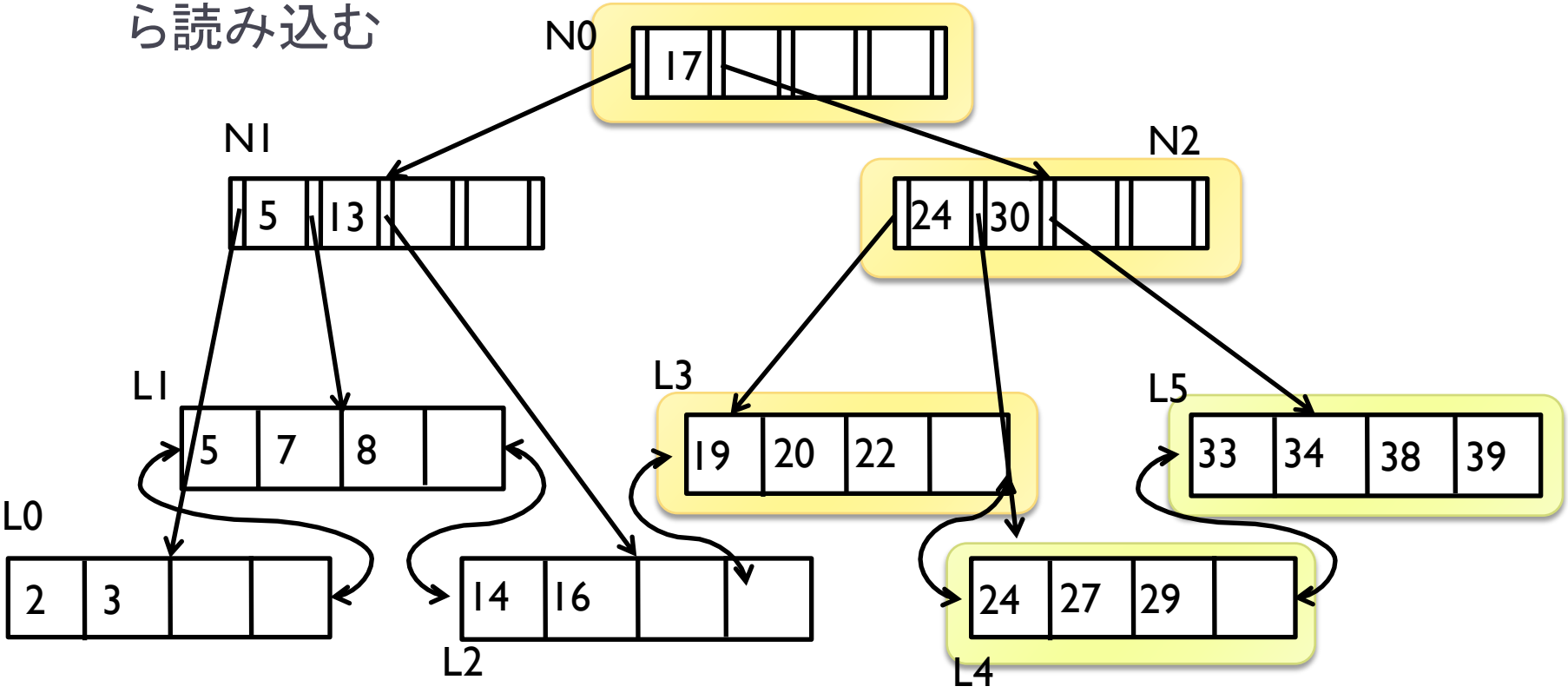
バッファマネージャを使った例(1/2)

- ユーザAが以下の一次索引で age < 19を検索し
- ページN0,N2,L0~L3はメモリ上のバッファプールに保管される



バッファマネージャを使った例(2/2)

- その直後にユーザBが以下の一次索引で age >= 22を検索したとする
  - N0,N2,L3はバッファプールから読み込み、L4,L5はディスクから読み込む

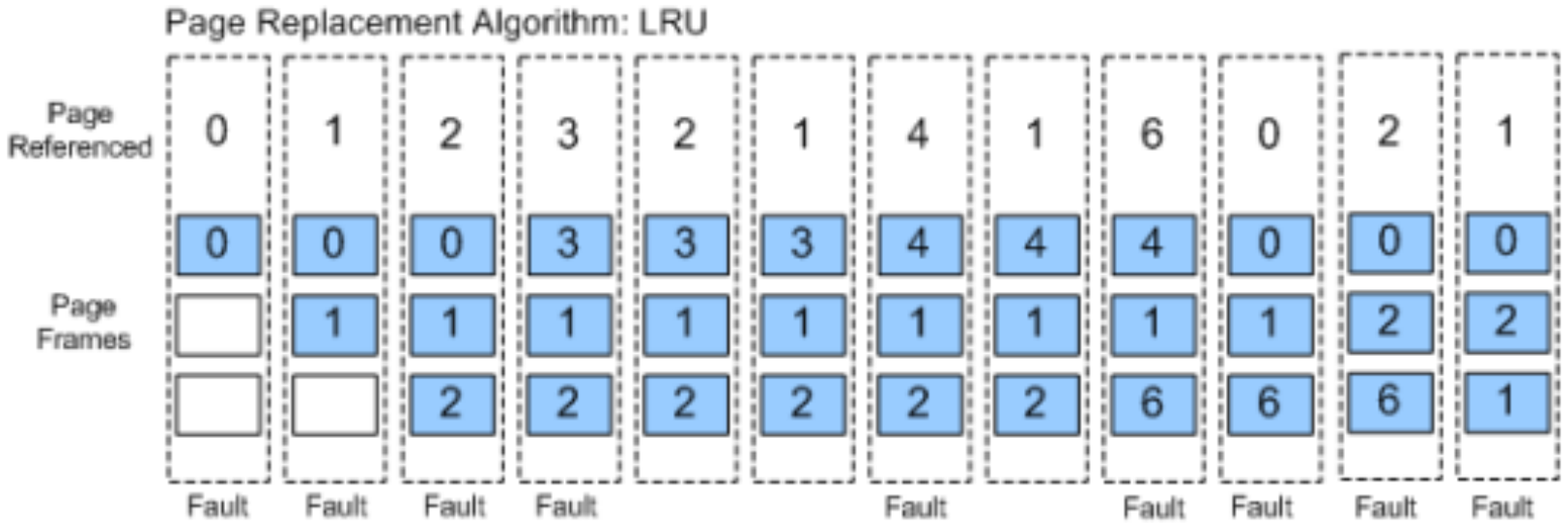


演習: 問合せ処理時間の計算

- 前の例でユーザAとユーザBの行った問合せの処理時間を見積もろう
  - メモリへのアクセス時間を2ミリ秒とする
  - ディスクへのアクセス時間を40ミリ秒とする

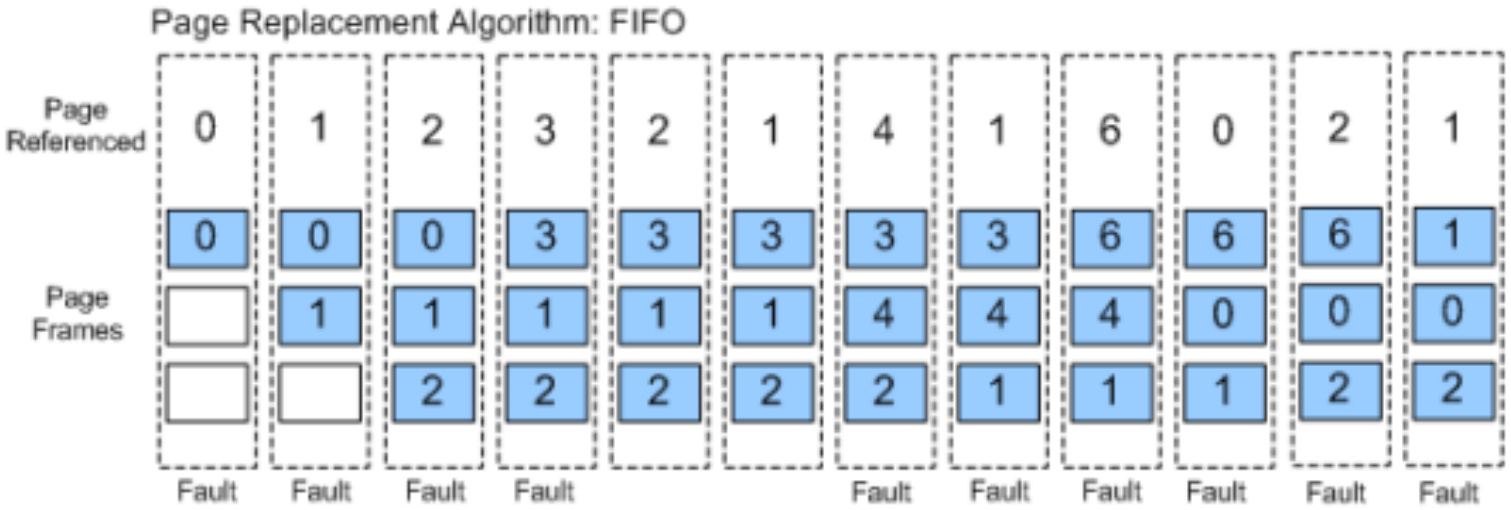
バッファ置き換え戦略

- 基本的にはOSのページ置き換え戦略と同じ
- LRU (Least Recent Used)
  - バッファの中で一番長く使われていないページを置き換える



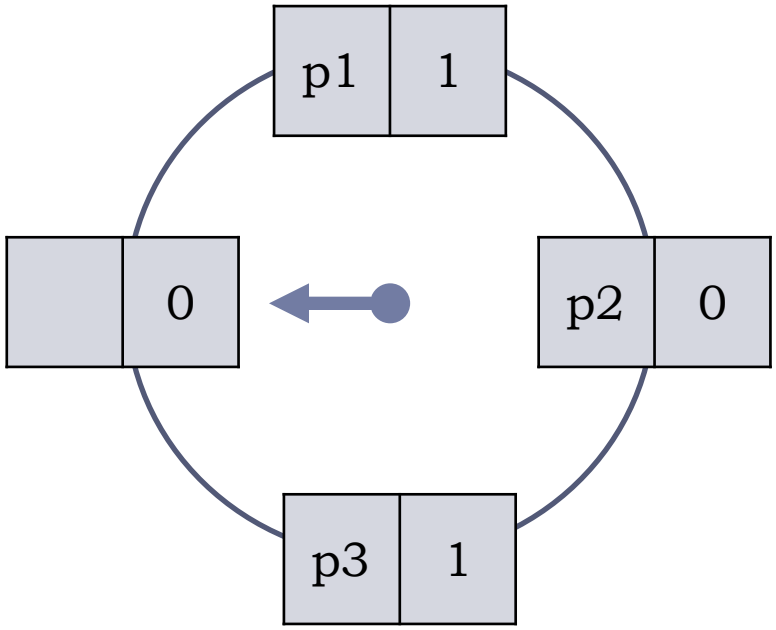
バッファ置き換え戦略

- FIFO (First In First Out)
  - バッファの中で一番最初に置かれたページを置き換える
    - B-treeのルートなど，頻繁にアクセスするページはFIFOだと何度も読み込まれ不便



バッファ置き換え戦略

- クロックアルゴリズム
  - バッファを円環状に並べ
    時計まわりにポインタを
    まわす
  - flagの値
    - バッファを置いた時と
      アクセスがあった時
      →1にする
    - 時計の針が過ぎたとき
      →0にする
  - ポインタが回ってきたとき
    flagの値が0だったら
    バッファを置き換える



スラッシング

- DBMSがOSの仮想メモリを利用し，バッファプールを大きくとりすぎているときにおこる現象
- 仮想メモリ
  - 実メモリ以上のメモリ空間を確保するための仕組み
  - スワッピング：あまり使わないページをディスクに退避させる
- バッファプールが大きすぎると…
  - バッファプールの中でスワッピングが発生し、ページ置き換えが頻発してしまう

おまけ：インメモリデータベース

- インメモリデータベース
  - すべてのデータをメモリにおいてしまうデータベースシステムのこと
  - 大抵のDBMSはインメモリモードを搭載
    - MySQL, sqlite, Apache Derby
    - Microsoft SQL Server 2014
- データの永続性を保証する方法
  - レプリケーション（複製）
    - ネットワーク上に複数のサーバを立て、データを複製する
  - ログをディスクに保存