# TCMallocの利用による Cプログラムの高速化

スーパーコンピューティング研究部門

林　雅江

## mallocによる動的なメモリ確保

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
        int  *A;

        A = (int*)malloc(sizeof(int)*3);  /* int型の領域確保 */
        if( A == NULL )   /* 領域確保に失敗したか */
        {
                printf( "%dバイトの領域確保に失敗", sizeof(int)*3 );
                return 1;
        }
        A[0] = 1;
        A[1] = 2;
        A[2] = 3;

        free( A );  /* 領域解放 */

        return 0;
}
```

- どんなときに使うのか
  - 配列サイズがプログラム実行中に決まる
  - 決定したサイズで配列のメモリを確保したい（無駄がない）
- malloc関数
  - 引数に書かれた分のバイト領域をメモリ上に確保し，その領域のアドレスを返す
  - 戻り値のアドレスをポインタ変数に代入
  - ポインタの型に合わせてキャスト
  - 使い終わった領域はfree関数で開放
- 実際のアプリではAL, AU, B, X, INL, IAL…など動的に確保.

## mallocの実装いろいろ

- メモリ管理の実装はOSとハードのアーキテクチャに大きく依存
  - glibc malloc（ptmalloc2）
  - Doug Lea malloc dlmalloc
  - OpenBSD malloc
  - Hoard
  - TCMalloc
  - 自作mallocなど

## TCMallocとは

- Thread Caching Malloc
- Performance toolsで提供されるツールの一つ
  - Googleが公開するパフォーマンス測定用のプロファイラ
  - CやC++のプログラムの高速化やバグの検出
  - 全部で4つのツールを提供
    - TCMalloc
    - Heap Checker:メモリリーク検出のライブラリ
    - Heap Profiler:各関数のメモリ使用量を計測・表示ツール
    - CPU Profiler:各関数の実行時間を計測・表示ツール
- 32KB以下の小さなオブジェクトのmalloc/freeにglibc 2.3 malloc (ptmalloc2)約300nsecに対しTCMallocでは約50nsec(2.8GHz　P4上，google-perf-toolsサイト*より）

*http://goog-perftools.sourceforge.net/doc/tcmalloc.html

# 標準以外のmallocを使う

- mallocのライブラリを用意する
  コンパイルまたはダウンロードによりmalloc.soをインストール
- アプリケーションコンパイル時にリンク

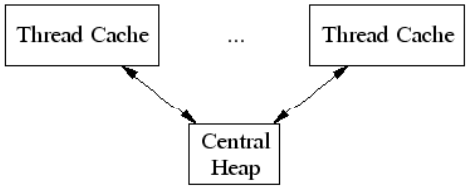
```
$ gcc –L/path/to/lib –ltcmalloc   ソースファイル.c
```

- コンパイルしない場合
  - LD_PRELOAD環境変数で実行時にダイナミックリンク
  - 指定するmallocライブラリに優先権
  - bash

```
$ LD_PRELOAD=/path/to/lib/libtcmalloc.so  a.out
```

# TCMallocの実装(1/4)

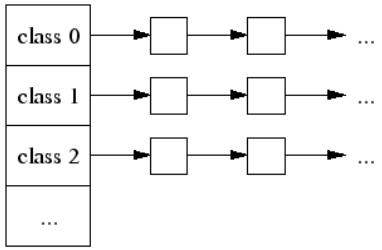
- マルチスレッドプログラムにおけるロックの競合を削減することで高速化
- 管理するメモリは各スレッドごとに用意されるスレッドキャッシュと全スレッドで共有する中央ヒープからなる.
  - 32KB以下のオブジェクトはスレッドキャッシュ上に確保（ロック必要なし）
  - 32KBより大きなオブジェクトは中央ヒープに確保．ページレベル（4K aligned region）でアロケータを使用




図はhttp://goog-perftools.sourceforge.net/doc/tcmalloc.htmlより引用

# TCMallocの実装(2/4)

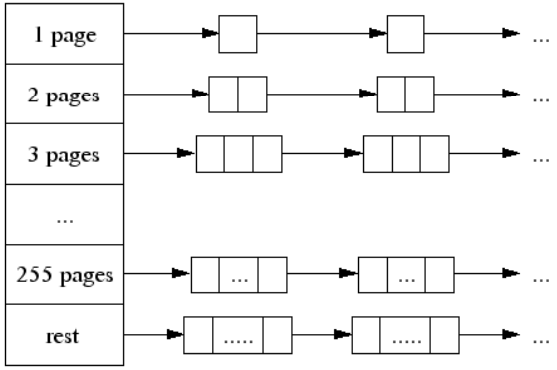
- Small Object Allocation




http://goog-perftools.sourceforge.net/doc/tcmalloc.htmlより引用

# TCMallocの実装(3/4)

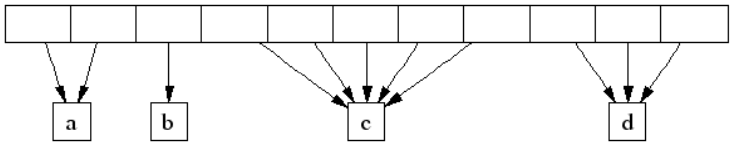
- Large Object Allocation




http://goog-perftools.sourceforge.net/doc/tcmalloc.htmlより引用

# TCMallocの実装(4/4)

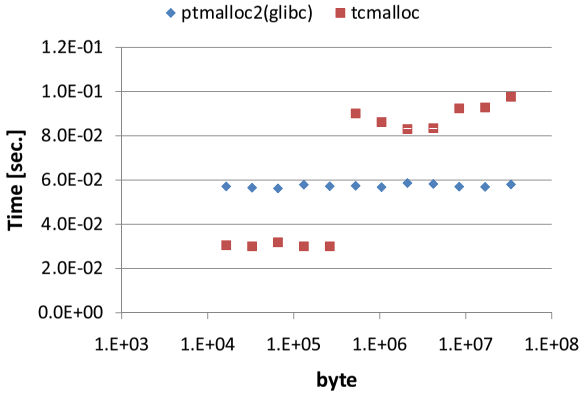
- Spans




http://goog-perftools.sourceforge.net/doc/tcmalloc.htmlより引用

# 実験1

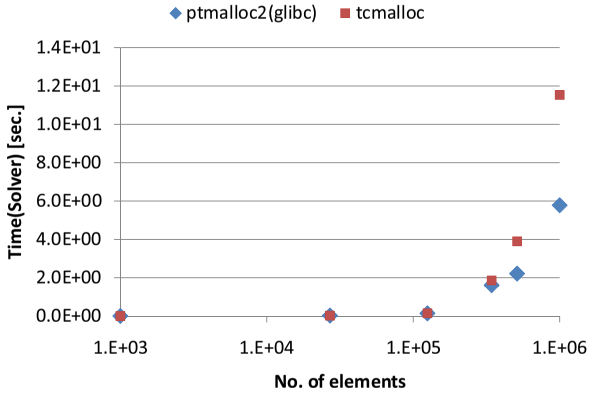
- malloc/freeを100万回繰り返すプログラム




256KBまではtcmallocが早いが512KBからptmallocが逆転

# 実験2

- 講習会プログラム
- reorder case3a




50^3elementsまでほとんど差がないが，70^3elementsからptmalloc2の方が効果的

# まとめ

- Cプログラムのマルチコア環境での高速化の手段として TCMallocの利用を紹介した
- 小さいオブジェクトのmalloc/freeを繰り返すアプリケーションでは効果あり
  - mysql, firefox, thunderbird, gnome関連，google desktopなど大物アプリでは効果大（google-perf-toolsサイトより）。
- 本講習会アプリケーションのような数値アプリケーションではmalloc/freeの回数もたかだが数十回かつ配列サイズも比較的大きくtcmallocを使う高速化は期待できない