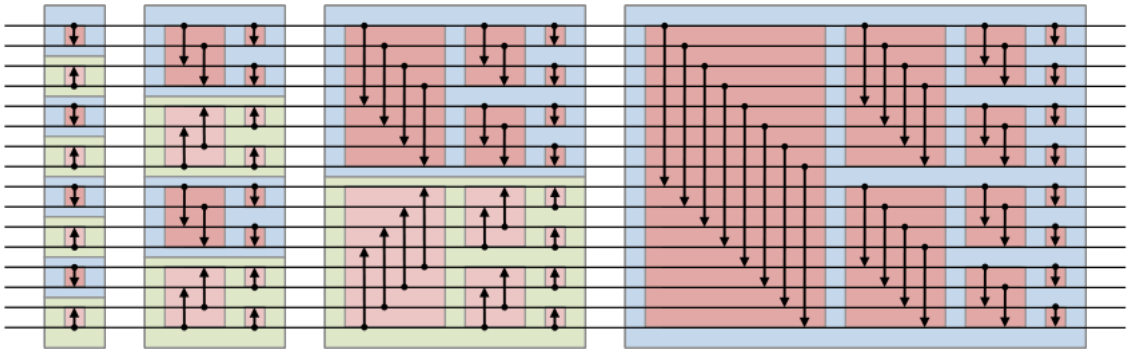
連続系アルゴリズム演習 1

05211006

王 朔

1. 達成した最大の N
   2^23 個（0.17 秒前後）

2. アルゴリズムの説明
   バイトニックソートと C++の標準ライブラリの std::sort()を組み合わせたアルゴリズムを実装した。バイトニックソートは 2 数の比較を繰り返し単調増加 & 減少列を交互に作りながらソートしていく手法で、2 数の比較&スワップは並列に行うことができるので、各プロセスに適切に配分して高速化を図ろうとしたのが狙いである。



（Wikipedia から引用）

3. 工夫した点
   要素数やプロセス数は 2 の冪乗であるので、あらゆる演算を bitwise AND やシフト演算で代用することで高速化をしている。
   また、上の図で各ブロック内の最も右にある矢印が隣接の 2 要素の大小関係を表した矢印になっている。これを見ると昇順になっているサイズが 2, 4, …と増えていくのがわかる。この時、初めからバイトニックソートのアルゴリズムを適用するとき、各プロセス内の要素を昇順(降順)に並べる事になるが、これは std::sort()を使って各プロセス内の要素を昇順(降順)に並べた方がショートカットになり実行時間を短縮できたので、std::sort()と組み合わせている。

4. 考察
   最終的にプロセス数を 32 として実行した時に最も早いという結果を得た。私の実装

上プロセス数を増やすと、上の図でいう青と緑のブロックが増えてより多くの通信時間がかかり、バイトニックソートを適用するべきステップが増えてしまう。後半のステップになればなるほど比較の回数も増えていくので、より遅くなる。一方で、プロセス数を減らすと std::sort()にかかる時間はおよそ 2 倍になる。これらが丁度バランスするのがプロセス数 32 の場合であることが予想される。

5. コンパイルオプションについて
   mpic++ -O3 main.cpp