アルゴリズム及び演習 第 12 回 補足

小野 孝男

2007 年 7 月 9 日

問題 2 について

一般的な環では (b) が成り立たないことがあります. つまり, (b) の証明では「$(x+1)(x-1) = 0$ かつ $x-1 \neq 0$ ならば $x+1 = 0$ であることが重要ですが, これは一般の環で成り立つとは限りません. 特に, $x-1$ が零因子であるような環では成り立ちません. 例えば $Z/16Z$ (16 を法とする剰余環) における 1 の原始 4 乗根は 3, 5, 11, 13 の 4 個ありますが, これらを $4/2 = 2$ 乗しても $-1 \equiv 15$ にはなりません (9 になります. もちろん 9 は 1 の原始 2 乗根の 1 つです).

逆に考えれば, 零因子を持たない環 (つまり整域) であれば (b) は必ず成り立ちます.

問題 3 について

いっそのこと, 「再帰を使わずに FFT を実現する」という方針も考えられます. 実際, このアルゴリズムで再帰を使っているのは「アルゴリズムを簡単にするため」でしかありません. より具体的には, 「連続する n 個のデータに対して Fourier 変換を求める」というアルゴリズムを考えているために, 再帰を使っている (これで `y`, `z` を破壊しないようにしている) のです.

考え方を変えて, 「s 個ずつ離れていれば, 連続しなくてもよい」としてしまえば `butterfly` → `shuffle` → `fft` という流れで処理できます. 最後の `fft` では $2s$ 個ずつ離れた, 半分の個数のデータに対して (1 回ずつ) 処理を行うことになります. このようにすれば, 最初の `butterfly` で使った作業領域は `shuffle` が終われば不要となるので, $O(n)$ の領域計算量であることは簡単にわかります.

この場合の変更点は

- 全ての関数で s が引数に追加される.
- `fft` で `shuffle` と `fft` の順序が変わる (この結果, 末尾再帰になるので実は繰り返しに変更できる).

となります.

問題 4 について

`fft` を呼び出したあとで n で割る以外に, `butterfly`, `shuffle`, `fft` のいずれかの最後で 2 で割るという方針もあります. 関数 `fft` の再帰の深さは $\log n$ なので, 再帰的に実行するたびに 2 で割れば最終的に $2^{\log n} = n$ で割ったことになる, という理屈です.