**3.2 LMS cần DCT cho Compression Ratio ( dịch là tỷ lệ nén ) (CR) có sức cạnh tranh**

Trong **Eisa**, LMS đã là đủ để nhận được chất lượng nén âm thanh tốt. Mặc dù áp dụng LMS cho hình ảnh là ý tưởng ban đầu và cho kết quả có chất lượng tốt, thế nhưng CRs không đủ cạnh tranh khi so sánh với phần lớn các công trình được công bố gần nhất. **Figure 2** biểu diễn một bức ảnh được xử lý với việc sử dụng LMS mà không có DCT. Sự chính xác và chất lượng của kết quả khác biệt 1 cách rõ ràng khi so sánh với kết quả được sử dụng cả LMS và DCT (**Fig. 3**). Ngoài ra, để củng cố lập luận này, chúng tôi đã áp dụng LMS cùng DCT và không cùng DCT với tất cả các hình của chúng tôi với CRs khác nhau. Kết quả được tóm tắt trong **Table 1**. Chúng tôi nhận thấy rằng không có trường hợp đơn lẻ nào sử dụng chỉ LSM mà cho kết quả tốt hơn, trong khi điều ngược lại thì đúng.

Để thử xem có cách nào để cải thiện LMS một cách đáng kể không, chúng tôi đã thực hiện các thử nghiệm LMS khác nhau mà không có DCT, như là LSM 2 chiều(phù hợp bề mặt). Và kết quả rất sát sao. Chúng chỉ tốt hơn 1 chút so với LMS 2 chiều, nhưng phức tạp hơn nhiều. Ví dụ, Lena được xử lý bằng LSM với cả CR = 2.67 bpp, PSNR 1 chiều bậc 1 là xấp xỉ 31, trong khi PSNR 2 chiều bậc 1 là xấp xỉ 32, sự khác biệt có thể thấy được về chất lượng là không có. Ngoài ra, chúng tôi cân nhắc việc nghiên cứu LMS với cùng CR nhưng khác nhau về bậc đa thức và kết quả cũng cho thấy rằng không có sự khác biệt về chất lượng hoặc thậm chí là chất lượng thấp hơn với những bậc cao hơn trong khi nhiều hệ số đa thức bị bóp méo bởi việc quantized. Như 1 hệ quả của điều này, chúng tôi xem xét việc thực hiện LMS 1 chiều với bậc đa thức cố định để ít phức tạp hơn và dễ tính toán hơn.

**3.3 Tạo ra luồng nhị phân (Production of Binary Streams)**

Ma trận cuối cùng được lưu lại hoặc chuyển đi phải bao gồm 1 giá trị phạm vi nguyên giống với các pixels để cho phép thực hiện 1 chuyển sang các binary streams ( dịch là : luồng nhị phân ). CR được tính toán bằng giá trị bpp mới. 1 cách tối ưu là giá trị nguyên tối thiểu của việc quantized để nhận được các giá trị quantized nhất quán của đa thức bậc 1. Chúng tôi chuyển đổi các số nguyên có tần số cao nhất thành 0 và phân bổ các số nguyên khác quanh số 0. Chúng tôi nhận ra rằng phần lớn các số nguyên của {a0} và {a1} trong các phân phối là khác nhau, sao cho a0 cần ít bpp hơn a1 được biểu diễn. Để làm rõ hơn, hãy để zimage là bpp cho hình được test. Trong ma trận hệ số đa thức, cột của hệ số a0 có thể được biểu diễn bởi (zimage – 1) bpp. Các cột khác biểu diễn cho số hạng bậc nhất a1 có thể được biểu diễn bằng (zimage + 1) bpp. Như 1 ví dụ trong các thử nghiệm của bài báo này, zimage = 8 bpp, a0 có thể được biểu diễn bằng 7 bpp và a1 có thể được biểu diễn bằng 9 bpp. Figure 4 và 5 biểu diễn các phân phối của a0 và a1 cho hình Lena.

**3.4 Tính toán độ phức tạp (Complexity Computation)**

Cả thuật toán mã hoá và giải mã, đều bao gồm 2 phần, được thảo luận trong phần này. Bên trong các phần mã hoá và giải mã (coding and encoding part), chúng ta có DCT 2 chiều, thuật toán nghịch đảo DCT, và và chuỗi triển khai LSM 1 chiều. Để đơn giản, và vì tất cả các hình được test ở trong bài báo là hình vuông, chúng tôi giả sử rằng N = M, N1 = M1, N2 = M2. Bằng cách nhìn vào **Eqs. (1, 2)**, và như được tìm thấy trong tài liệu, sự khai triển của các phương trình sử dụng 4 vòng lặp “for”, vì vậy độ phức tạp là O(N4) cho ma trận N x N. Nó cũng được biết đến trong tài liệu thuật toán dựa trên DCT, như là JPEG, DCT 1 chiều có độ phức tạp O(N2) có thể được sử dụng bằng cách chạy 2N lần để có DCT 2 chiều, vì vậy, độ phức tạp có thể được giảm xuống còn O(N3). Thậm chí O(N2) DCT 1 chiều có thể được thay thể bởi 1 nhân tố tương tự để có 1 Fast Fourier Transform ( dịch là chuyển đổi Fourier nhanh ) có độ phức tạp là O(NlogN), vì vậy độ phức tạp tổng hợp của DCT 2 chiều có thể giảm xuống còn O(N2logN). Thuật toán nghịch đảo có độ cùng độ phức tạp. Đối với LSM, nó được áp dụng lần với mỗi hàng trong ma trận N1xN1. Sự phức tạp của của 1 triển khai LMS là tích x (N1) để có được sự phức tạp của toàn bộ chuỗi triển khai LMS để bao trùm toàn bộ ma trận N1 x N1. Bằng cách nhìn vào **Eqs. (3, 4)** và dựa vào bất kỳ tài liệu đại số tuyến tính số nâng cao nào, ta biết rằng đối với 1 triển khai LSM, chúng ta có ưu thế về độ phức tạp trong giải quyết các vấn đề hệ thống, như được thấy trong **Eq. (5)** và nó là O((n+1)2Ns). Vì vậy độ phức tạp của toàn bộ chuỗi triển khai LSM là O(N12(n+1)2). Tổng hợp độ phức tạp phần coding của thuật toán được nêu ra như sau:

* DCT 2 chiều cho ma trận N x N, O(N2logN)
* Nghịch đảo DCT 2 chiều cho ma trận, O(N12logN1)
* LSM cho ma trận N­1 x N­1­, O(N12(n+1)2).

Toàn bộ độ phức tạp phần giải mã (encoding) ( khôi phục ) của thuật toán được nêu ra như sau:

* Thực thi bằng cách khôi phục các giá trị phù hợp từ các hệ số LSM, O(N12)
* DCT 2 chiều cho ma trận N­1 x N­1­, O(N12logN1)
* Nghịch đảo DCT 2 chiều cho ma trận N x N, O(N2logN)
* Áp dụng bộ lọc ( không bắt buộc )

Vì độ phức tạp trong toán tử O chính xác hơn, khi N tiến tới vô hạn, chúng ta thấy rằng với ảnh 512x512, như là ảnh chúng ta đang test, độ phức tạp của phần giải nén nhiều hơn phần nén trong LSM. Đó là bởi vì 512 là không đủ lớn để có thời gian xử lý gần/giống với khai triển LSM ( trong phần nén ) và chỉ thay thế trở lại các giá trị ( trong phần giải nén ). Thêm nữa, có sự khác biệt lớn giữa O(N12(n+1)2) và O(N12) trong phần mã nén (compression coding) và mã giải nén (decompression coding) tương đương. Với tất cả các nguyên nhân này, việc giải nén của thuật toán này nhanh hơn rõ rệt khi so sánh với việc nén. Từ những thử nghiệm của chúng ta, như được biểu diễn trong **Fig. 6**, thời gian xử lý cho 1 bộ xử lý phổ biến (laptop) và bằng Matlab đề xuất các ứng dụng thời gian thực cho cách tiếp cận được đề xuất trong bài bnáo này

Định nghĩa quantize ở trang này: [Khoa học dữ liệu (phamdinhkhanh.github.io)](https://phamdinhkhanh.github.io/2020/11/23/Quantization.html)

**Eisa** là mục 17 trong tài liệu