

# Compressão de Sinais Multidimensionais utilizando Recorrência de Padrões Multiescalas com Segmentação Flexível

Waldir S. S. Júnior, Eddie B. L. Filho, Eduardo A. B. da Silva, Murilo B. de Carvalho e Gelson Vieira Mendonça

**Resumo**—O algoritmo MMP (*Multidimensional Multiscale Parser*) é um método de compressão de sinais universal, baseado em recorrência de padrões multiescalas, no qual cada bloco do sinal de entrada é aproximado por um ou mais elementos do seu dicionário. Este último é construído durante o processo de codificação, adicionando-se versões dilatadas e contraídas de concatenações de padrões previamente codificados. O MMP é estruturado sob a forma de uma árvore de segmentação binária, construída de modo a minimizar um dado critério taxa-distorção. Sabe-se que, para sinais polinomiais unidimensionais, métodos estruturados sob a forma de uma árvore binária possuem um desempenho taxa-distorção dado por  $D(R) \sim d_0 \sqrt{R} 2^{-d_1 \sqrt{R}}$ . Recentemente, verificou-se que ao se permitir a união de nós-folhas (*prune-join*), o desempenho taxa-distorção passa a ser dado por  $D(R) \sim c_0 2^{-c_1 R}$ . Esse fato motivou o desenvolvimento de uma modificação no MMP original, proporcionando-se a união de blocos de imagem através de uma extensão do *prune-join* unidimensional. As simulações realizadas mostram que o algoritmo resultante é eficaz, apresentando um desempenho taxa-distorção superior ao do MMP original e imagens reconstruídas com maior qualidade subjetiva.

**Palavras-Chave**—Casamento de Padrões Recorrentes, Decomposição Multiescalas, Compressão de Sinais Multidimensionais, *Side-Match*, *Prune-join*.

**Abstract**—The MMP (*Multidimensional Multiscale Parser*) algorithm is a universal signal compression method, based on multiscale recurrent patterns, in which a block of the input signal is matched to one or more elements in its dictionary. The latter is built during the encoding process, with dilated and contracted versions of concatenations of patterns already encoded. The MMP is structured in the form of a binary segmentation tree, built in order to minimize a given rate-distortion criterion. It is known that, for one-dimensional polynomial signals, methods structured in the form of a binary tree present a rate-distortion performance given by  $D(R) \sim d_0 \sqrt{R} 2^{-d_1 \sqrt{R}}$ . Recently, it was shown that when the union of leaf nodes (*prune-join*) is allowed, the rate-distortion performance becomes  $D(R) \sim c_0 2^{-c_1 R}$ . This assertion motivated the development of a modification in the base MMP algorithm, providing the union of image blocks through an extension of the one-dimensional *prune-join*. The simulations performed show that the resulting algorithm is

Waldir S. S. Júnior<sup>1†</sup>, Eddie B. L. Filho<sup>2†</sup>, Eduardo A. B. da Silva<sup>1</sup>, Murilo B. de Carvalho<sup>3</sup> e Gelson Vieira Mendonça<sup>1</sup>, <sup>1</sup>Fucapi - Fundação Centro de Análise, Pesquisa e Inovação Tecnológica, Av. Gov. Danilo Areosa, 381, Dist. Indust., Manaus - AM, 69075-351, BRASIL, <sup>2</sup>Genius Instituto de Tecnologia, Av. Açaí, 875, Bl. E, Dist. Indust., Manaus - AM, 69075-904, BRASIL, <sup>3</sup>PEE/COPPE/DEL/POLI, Universidade Federal do Rio de Janeiro, Cx. P. 68504, Rio de Janeiro - RJ, 21945-970, BRASIL, <sup>†</sup>TET/CTC, Universidade Federal Fluminense, R. Passos da Pátria, 156, Niterói - RJ, 24210-240, BRASIL, E-mails: waldirjr@fucapi.br, efilho@genius.org.br, murilo@telecom.uff.br, eduardo@lps.ufrj.br. Este trabalho foi realizado no âmbito do convênio UFAM-UFRJ/COPPE, com financiamento da SUFRAMA.

effective, presenting a superior rate-distortion performance when compared to the original MMP and reading reconstructed images with higher subjective quality.

**Keywords**—Recurrent Pattern Matching, Multiscale Decomposition, Multidimensional Signal Compression, *Side-Match*, *Prune-join*.

## I. INTRODUÇÃO

A humanidade, nas últimas décadas, tem gerado uma grande quantidade de informação no domínio digital, principalmente devido ao desenvolvimento de tecnologias para o seu armazenamento e transmissão. Entretanto, verifica-se que tais tecnologias não são capazes de suprir as necessidades humanas por completo, pois estas crescem mais rapidamente e demandam aplicações cada vez mais complexas. Sendo assim, torna-se necessária a manipulação desse imenso volume de informação, obtendo-se uma representação mais compacta para o mesmo. Isto é especialmente relevante no caso de imagens digitais, onde métodos de compressão são essenciais para tornar o seu uso viável. O estado da arte em métodos de compressão de imagens baseia-se na codificação por transformadas [2], [3], [4], na qual supõe-se que uma imagem possui, essencialmente, a maior parte de seu conteúdo de energia confinada em baixas frequências (imagens suaves). O algoritmo denominado MMP (*Multidimensional Multiscale Parser*), baseado na recorrência de padrões multiescalas e recentemente desenvolvido em [1], dedica-se justamente à classe de imagens que não satisfazem tal suposição (imagens não suaves), como, por exemplo, imagens com textos, gráficos ou mistas, enquanto ao mesmo tempo provê um bom desempenho para imagens suaves.

No MMP, a representação de um bloco de entrada  $B^j$  é feita através de um elemento do seu dicionário, que satisfaz um dado critério de aproximação (geralmente baseado na relação taxa-distorção). Se isto não ocorre, o bloco de entrada é particionado, originando-se dois outros blocos. Tal procedimento é executado recursivamente, até que o critério seja satisfeito. Devido a este tipo de partição, podemos dizer que o MMP é baseado em uma estrutura de árvore binária  $A(n_0)$ .

Recentemente, desenvolveu-se um novo método de codificação baseado em árvore binária, chamado de *prune-join* [8]. Esse método permite explorar, em uma estrutura baseada em árvore binária, a dependência entre os nós-folhas que possuem nós-pais diferentes, levando-se em consideração o custo Lagrangeano associado a cada um (dado por  $D + \lambda R$ , onde  $D$  é a distorção e  $R$  é a taxa) para decidir sobre sua

codificação conjunta (união). Nesse mesmo trabalho, mostra-se ainda que tal método consegue um comportamento da função taxa-distorção, para sinais polinomiais por partes, dado por  $D(R) \sim c_0 2^{-c_1 R}$ . Este resultado supera o comportamento obtido quando não se faz a união de nós-folhas, que é dado por  $D(R) \sim d_0 \sqrt{R} 2^{-d_1 \sqrt{R}}$ .

Como o método *prune-join* apresenta características de desempenho superiores para sinais polinomiais e o MMP possui uma estrutura semelhante, ou seja, ambos utilizam uma estrutura em árvore binária otimizada segundo um dado critério taxa-distorção, torna-se inevitável indagarmos sobre a possibilidade de se explorar, na estrutura do MMP, a dependência entre nós-folhas que possuem nós-pais diferentes. Neste artigo, mostramos uma modificação que responde a este questionamento, chamada de PJ-MMP (*Prune-Join MMP*).

O restante deste artigo está organizado conforme descrito a seguir. Na seção II, a versão original do algoritmo MMP é apresentada. Na seção III, expomos o método que contempla a estrutura de união de blocos com nós-pais diferentes, ou seja, o PJ-MMP. Resultados experimentais com imagens em níveis de cinza são apresentados na seção IV e a seção V apresenta as conclusões do trabalho.

## II. O ALGORITMO MMP

O algoritmo MMP [1] é um método universal de compressão de sinais, baseado na recorrência de padrões multiescalas, no qual um bloco de entrada  $B^j$ , com dimensões  $l \times c$ , é aproximado por elementos de um dicionário adaptativo  $D$ . A escolha da melhor aproximação é feita através da minimização do erro quadrático, dado por  $\sigma = \min\{\|B^j - \widetilde{B}_i^j\|^2\}$ , que é obtido através da comparação do bloco de entrada com todos os elementos ( $\widetilde{B}_i^j$ ) de  $D$ . A aproximação de  $B^j$  por  $\widetilde{B}_i^j$  é confirmada se  $\sigma$  é menor que um dado valor de distorção alvo  $d^*$  ( $\sigma < d^*$ ). Neste caso, o bloco de entrada é codificado e retorna-se um *flag* '1' e o índice  $i$  do elemento do dicionário que resultou na menor distorção. Caso  $\sigma$  ultrapasse o valor de distorção alvo  $d^*$  ( $\sigma > d^*$ ), o bloco de entrada é particionado na horizontal, se  $l \geq c$ , ou na vertical, se  $l < c$ . Então, retorna-se um *flag* '0' e a tentativa de aproximação continua recursivamente, utilizando-se os dois blocos  $B^{2j+1}$  e  $B^{2j+2}$  gerados pelo procedimento de partição, até que  $\sigma < d^*$  ou os blocos resultantes tenham dimensões  $l = c = 1$ . Por exemplo, podemos considerar um bloco bidimensional  $l \times c$  processado pelo MMP, cujas partições obtidas são mostradas na Figura 1. O arquivo codificado, ou seja, o *bitstream* gerado seria igual a 0, 0, 0, 1,  $ind(\widetilde{B}^7)$ , 1,  $ind(\widetilde{B}^8)$ , 1,  $ind(\widetilde{B}^4)$ , 0, 1  $ind(\widetilde{B}^5)$ , 1,  $ind(\widetilde{B}^6)$ , onde  $ind(\cdot)$  representa o índice do elemento do dicionário que minimizou o erro quadrático.

Podemos perceber, na Figura 1, um comportamento bastante peculiar do algoritmo, pois sempre são gerados dois novos blocos oriundos de uma partição. Assim, é possível se associar este comportamento a uma estrutura em árvore binária  $A(n_0)$ , conforme mostra a Figura 2, na qual o nó  $n_j$  representa o bloco  $B^j$  de dimensões  $(2^{-\lfloor \frac{p+1}{2} \rfloor} l \times 2^{-\lfloor \frac{p}{2} \rfloor} c)$ , onde  $p$  é a profundidade do nó  $n_j$  na árvore  $A(n_0)$ . Os nós que não sofrem partição são chamados de *nós-folhas* e representam

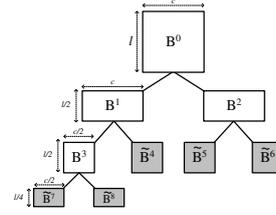


Fig. 1. Bloco bidimensional processado pelo MMP.

codificações efetivas do sinal de entrada, pois estão associados a elementos  $\widetilde{B}^j$  do dicionário. O sentido de formação da árvore é  $(n_0, n_1, n_3, n_7, n_8, n_4, n_2, n_5, n_6)$ .

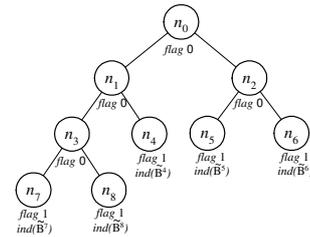


Fig. 2. Árvore binária associada às partições realizadas pelo MMP.

O dicionário do MMP tem  $N$  elementos de comprimento fixo. Apesar disso, as aproximações dos vários blocos resultantes do procedimento de partição são realizadas utilizando-se estes mesmos elementos, não importando as suas dimensões. Isto é possível devido à transformação de escala, que modifica as dimensões de um vetor antes da tentativa de casamento [1], [6]. A transformação de escala modifica o comprimento de um sinal através de sua dilatação ou contração [1], [6], permitindo a aproximação de elementos com comprimento fixo por elementos com comprimento variável. Em particular, para o caso unidimensional, se tivermos um vetor  $S$  de tamanho  $N_0$ , então a transformação será definida como  $S^N = T_{N_0}^N[S]$ , onde  $N$  representa o novo comprimento (escala) de  $S$ . Para imagens (caso bidimensional), definindo-se o bloco  $S$  de comprimento  $N_0 \times M_0$  com  $N_0$  sendo a quantidade de linhas e  $M_0$  a de colunas, então a transformação será  $S_{N,M} = T_{N_0, M_0}^{N, M}[S_{N_0, M_0}]$ , onde  $N \times M$  é a dimensão do novo bloco. Tal transformação é obtida processando-se primeiramente todas as linhas e depois todas as colunas, conforme definido na Equação 1, onde o bloco  $B^{n,m}$ , de dimensões  $n \times m$ , será transformado no bloco  $B^{l,k}$  e  $B_i^{n,m}$  representa a linha  $i$  do bloco  $B^{n,m}$ .

$$\begin{aligned} Y_i^{n,l} &= T_m^l [B_i^{n,m}], \quad i = 0, 1, \dots, n-1, \\ Z_j^{l,k} &= T_n^k \left[ \left( (Y^{n,l})^T \right)_j \right], \quad j = 0, 1, \dots, l-1, \\ B_h^{k,l} &= \left( (Z^{l,k})^T \right)_h, \quad h = 0, 1, \dots, k-1, \end{aligned} \quad (1)$$

Conforme mencionamos, o MMP possui um dicionário adaptativo  $D$ , cuja atualização é feita quando dois nós  $n_{2j+1}$  e  $n_{2j+2}$ , descendentes de um mesmo nó-pai  $n_j$ , já tiverem sido codificados (definidos como *nós disponíveis*). A atualização é realizada com a concatenação dos elementos (definida como  $\widetilde{B}^c$ ) associados aos nós disponíveis e sua transformação para inclusão no dicionário. Se considerarmos

um bloco com dimensões  $l \times c$ , então o esquema de partição define as seguintes dimensões:  $l \times c$ ,  $\frac{l}{2} \times c$ ,  $\frac{l}{2} \times \frac{c}{2}$ ,  $\frac{l}{4} \times \frac{c}{2}$ ,  $\frac{l}{4} \times \frac{c}{4}$ , ...,  $1 \times 1$ . Isto implica que existem, no máximo,  $1 + \log_2(l) + \log_2(c)$  diferentes dimensões ou escalas. Portanto, para se obter uma implementação eficiente, basta manter  $1 + \log_2(l) + \log_2(c)$  cópias do dicionário, uma em cada escala, para evitar a realização da transformação a cada nova tentativa de casamento.

A árvore  $A(n_0)$  pode ser otimizada num sentido taxa-distorção. Isto permite avaliar, ao mesmo tempo, a distorção e o número de *bits* necessário para a codificação. Esta otimização é realizada utilizando-se um critério de decisão global baseado no custo Lagrangeano [9], que define uma relação de compromisso entre a taxa e a distorção para uma sub-árvore pertencente à árvore  $A(n_0)$  e iniciada no nó  $n_x$ , dada pela expressão:

$$J[A(n_x)] = D[A(n_x)] + \lambda R[A(n_x)] \quad (2)$$

onde  $D[A(n_x)]$  é a distorção obtida utilizando-se a sub-árvore  $A(n_x)$  e  $R[A(n_x)]$  é a taxa. O procedimento básico de otimização, segundo [1], começa com uma árvore completa e segue dos nós-folhas para o nó  $n_0$ , podando o par de nós-filhos  $n_{2j+1}$  e  $n_{2j+2}$  sempre que o custo Lagrangeano da árvore que os contém for maior que o custo da árvore de segmentação sem os mesmos.

É importante ressaltar que sempre que um par de nós  $n_{2j+1}$  e  $n_{2j+2}$  for podado, as atualizações do dicionário resultantes da sua concatenação devem ser retiradas. Além disso, de modo a calcular corretamente os custos Lagrangeanos, é necessário manter-se um registro do número de vezes que cada vetor foi utilizado, assim como o número de ocorrências de cada *flag*.

### III. O ALGORITMO *Prune-Join* MMP (PJ-MMP)

Em [8], é proposto um método de compressão de sinais chamado de *Prune-Join*, cujo objetivo é aproximar funções unidimensionais por funções polinomiais por partes. O *Prune-Join* é construído de acordo com um cenário de decomposição em árvore binária, onde cada nó-filho pode sofrer, além da poda, uma codificação conjunta com um nó-vizinho (*joint-coding*), com o intuito de melhorar o desempenho taxa-distorção obtido. O método *Prune-Join* é composto por um passo de poda (*Prune*) da árvore binária e um passo de união (*Join*) de seus nós-folhas.

O passo *Prune* inicia com a segmentação do sinal de entrada, utilizando-se uma decomposição em árvore binária, até que a maior profundidade possível  $K$  seja atingida. Em seguida, cada segmento resultante é aproximado por um polinômio  $p(t)$ , utilizando-se como critério de escolha o menor erro quadrático médio. Gera-se, assim, uma curva RD (taxa-distorção) para cada segmento, dada pela aproximação deste a partir do polinômio quantizado  $\widehat{p}(t)$ <sup>1</sup>. Neste momento, inicia-se o processo de poda recursiva, para todos os nós, no qual decide-se pelo descarte dos nós-filhos se a soma de seus custos Lagrangeanos for maior ou igual ao do nó-pai. Isso significa que os segmentos referentes aos nós-filhos serão podados se:

$$\{(D_{f_1} + D_{f_2}) + \lambda(R_{f_1} + R_{f_2})\} \geq \{(D_p + \lambda R_p)\} \quad (3)$$

<sup>1</sup> $\widehat{p}(t)$  é obtido através da quantização escalar dos coeficientes de  $p(t)$

onde  $f_1$  e  $f_2$  são os nós-filhos e  $p$  é o nó-pai da árvore binária com profundidade  $K$  definida anteriormente. Ao final deste passo, obtém-se uma árvore binária otimizada com o melhor esquema de poda possível para os nós-folhas, chamada  $A(n_0)$ .

O passo *Join* tem como ponto de partida a árvore binária obtida anteriormente, ou seja,  $A(n_0)$ . Cada dois nós-vizinhos  $n_1$  e  $n_2$  (filhos de nós-pais diferentes) serão codificados em conjunto (unidos) se a soma de seus custos Lagrangeanos for maior ou igual ao custo Lagrangeano do nó unido  $n_J$ , de acordo com a Equação 4. A verificação da junção é feita através da varredura de todos os nós-folhas de  $A(n_0)$ , seguindo de baixo para cima e da esquerda para a direita.

$$\{(D_{n_1} + D_{n_2}) + \lambda(R_{n_1} + R_{n_2})\} \geq \{(D_{n_J} + \lambda R_{n_J})\} \quad (4)$$

Como exemplo, podemos descrever o processamento realizado pelo *Prune-Join* num sinal  $f(t)$  qualquer. A profundidade da decomposição, neste caso, é  $K = 4$  e está representada na Figura 3(a), juntamente com  $f(t)$ . Com o passo *Prune*, obtém-se uma árvore binária com a melhor configuração possível de nós-folhas, descrita na Figura 3(b), onde um nó é representado pelo par  $(i, j)$ , com  $i$  sendo a profundidade da árvore e  $j$  o número do nó. Observe que se  $j$  é nó-pai, então os nós-filhos serão  $2j$  e  $2j + 1$ . A verificação de junção possui a seguinte seqüência: (1,0), (2,3), (3,5), (4,8) e (4,9).

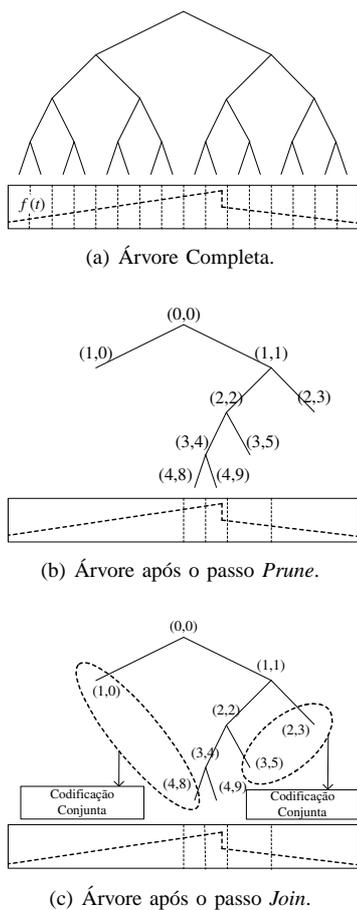
Com o passo *Join*, efetuaremos a codificação conjunta dos nós-folhas, segundo o critério de custo Lagrangeano definido na Equação 4 e mostrado na Figura 3(c). É importante observar que este método oferece a possibilidade de junção de nós-folhas que carregam a mesma informação. Outro fato importante consiste na asserção de que métodos estruturados sob a forma de uma árvore binária possuem uma curva taxa-distorção dada por  $D(R) \sim d_0 \sqrt{R} 2^{-d_1 \sqrt{R}}$ . No trabalho desenvolvido em [8], verificou-se que ao se considerar um cenário de codificação conjunta, a curva taxa-distorção passa a ser dada por  $D(R) \sim c_0 2^{-c_1 R}$ , o que implica, claramente, num desempenho superior.

Portanto, vimos que o método *Prune-Join* possui características de desempenho superiores para sinais unidimensionais. Sabemos, também, que o MMP possui uma estrutura baseada em árvore de segmentação binária, semelhante ao método descrito acima, e que os custos de cada nó desta árvore são calculados segundo o método Lagrangeano. Diante deste fato, somos levados a indagar o seguinte:

”Uma modificação no MMP para um cenário de codificação conjunta pode levar a uma melhoria de desempenho?”

Para responder a esta pergunta, desenvolvemos um método que contempla a codificação conjunta de blocos do MMP, chamado de *Prune-Join* MMP (PJ-MMP).

O MMP original realiza partições no sinal até encontrar um elemento do dicionário que o represente de forma satisfatória. Essas partições, conforme mencionamos, podem ser associadas a uma árvore binária  $A(n_0)$ , que pode ainda ser otimizada num sentido taxa-distorção. Porém, tal estrutura em árvore binária não explora a dependência de nós-filhos com nós-pais diferentes (definidos como *nós-vizinhos*), como é feito no *Prune-Join*. Sendo assim, propõe-se uma extensão


 Fig. 3. Processamento de um sinal pelo método *Prune-Join*.

bidimensional do *Prune-Join*, de tal forma que essas dependências sejam analisadas, além de permitir uma extensão da partição do MMP original, através da união dos elementos que serão codificados. A extensão do *Prune-Join* para o caso bidimensional não é simples, pois a codificação conjunta deve ser realizada somente em blocos com as mesmas dimensões horizontais ou verticais, ou seja, tal junção somente poderá ser realizada se o bloco resultante for retangular, além de satisfazer a condição de menor custo Lagrangeano.

O algoritmo de codificação PJ-MMP é dividido em dois passos principais: a *obtenção da árvore otimizada* e a *avaliação de junção*. No primeiro passo, obtém-se a árvore otimizada no sentido taxa-distorção  $A(n_0)$ , conforme mencionado no último parágrafo da seção II. A *avaliação de junção* é realizada com nós-vizinhos (filhos de nós-pais diferentes) cujos blocos associados estão à direita ou abaixo do bloco analisado, de acordo com o sentido de formação da árvore de segmentação, seguindo-se o critério de *custo de união* dado pelas Equações 5, 6 e 7. A união dos blocos é confirmada se  $J_{n_A, B} \leq J_{n_A} + J_{n_B}$ .

$$J_{n_A} = D_{n_A} + \lambda R I_{n_A} + \lambda R_{J0} \quad (5)$$

$$J_{n_B} = D_{n_B} + \lambda R I_{n_B} + \lambda R_{J0} \quad (6)$$

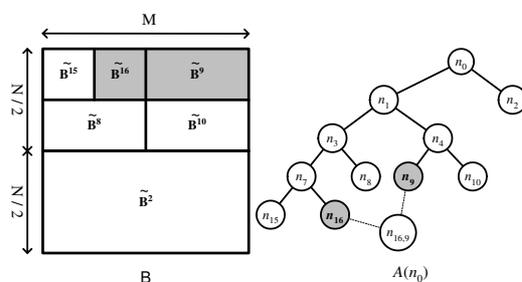
$$J_{n_A, B} = D_{n_A, B} + \lambda R I_{n_A, B} + \lambda R_{J1} + \lambda R_{Py} \quad (7)$$

Nas Equações acima,  $n_A$  e  $n_B$  são os candidatos a união,  $R I_{n_x}$  é a taxa para representar o índice do elemento escolhido,

$R_{Jk}$  é a taxa para o *flag* de junção, com  $k = 1$  para confirmação da união e  $k = 0$  caso contrário, e  $R_{Py}$  é a taxa para o *flag* de posição, com  $y = 0$  para união com o vizinho à direita e  $y = 1$  com o vizinho abaixo.

É importante comentar que existe a possibilidade de um bloco ser candidato a união tanto com o bloco à direita quanto com o abaixo. Neste caso, escolhe-se a junção que oferece a maior redução de custo.

Conforme a avaliação é feita, monta-se uma tabela que contém informações sobre a união dos blocos. Podemos visualizar uma avaliação de união na Figura 4, onde o bloco  $B^{16}$  é analisado. A codificação efetiva é realizada utilizando-se a árvore otimizada  $A(n_0)$ , *flags* de junção para informar que uma união ocorreu e *flags* de posição para informar se a junção foi com o bloco abaixo ou com o bloco à direita do atual. O dicionário  $D$  é expandido para contemplar os elementos originados das uniões e sua atualização não sofre alteração em relação ao MMP original.


 Fig. 4. Avaliação com o bloco à direita:  $n_{16}$  com  $n_9$ .

A reconstrução de cada bloco é realizada decodificando-se primeiramente a árvore  $A(n_0)$ , seguido da construção da tabela com informações de união (identicamente ao codificador) e, após, decodifica-se os índices dos blocos unidos ou não.

O desempenho do PJ-MMP pode ser melhorado, realizando-se ainda mais duas modificações. A primeira delas disponibiliza valores de custos reais para a avaliação da união. No algoritmo descrito acima, a união de um bloco é feita com um bloco que ainda não teve suas estatísticas atualizadas, pelo simples fato de que ele ainda não foi processado. Podemos ver um exemplo na Figura 5, onde verifica-se que o custo da união do nó  $n_3$  com o nó  $n_5$  é apenas estimado, pois o nó  $n_5$  ainda não foi processado (a ordem de processamento está indicada na Figura 5). Este problema é solucionado invertendo-se o sentido de avaliação de união, ou seja, avalia-se a união com o bloco acima e/ou a esquerda do atual. Assim, o método *aguarda* a árvore processar o candidato à união e a avaliação só é realizada neste momento.

A segunda modificação tem por objetivo aumentar o número de elementos no dicionário de blocos unidos, pois no PJ-MMP o dicionário não é atualizado quando a união de blocos ocorre. Os dicionários, nas dimensões correspondentes às uniões, são apenas versões transformadas dos demais dicionários, o que provoca uma sensível queda de desempenho. Desta forma, fomos levados a realizar cinco atualizações extras no dicionário, obtidas com deslocamentos de uma janela dentro de cada bloco gerado por concatenação. Tendo o bloco  $N \times M$  na Figura

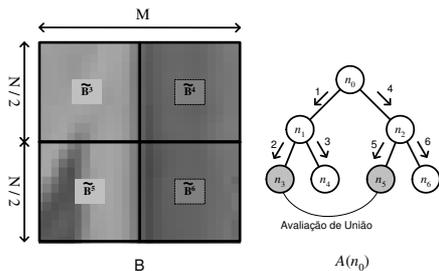
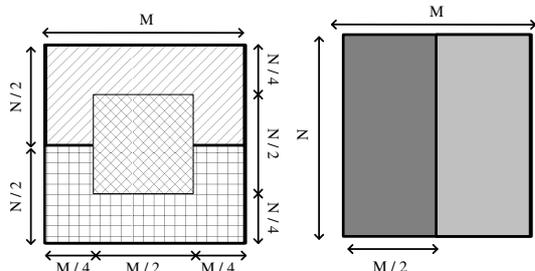


Fig. 5. O problema da estimativa de custo.

6, obtido da concatenação de dois elementos codificados, incluímos no dicionário também os elementos compreendidos por uma janela central, duas horizontais (Figura 6(a)) e duas janelas verticais (Figura 6(b)). Além disso, tomamos o cuidado de não incluir nos dicionários elementos iguais aos já existentes. Esta mudança levou a um aumento significativo de desempenho.



(a) Atualizações horizontais e (b) Atualizações verticais centrais.

Fig. 6. As cinco novas atualizações.

#### IV. RESULTADOS DE SIMULAÇÕES

O MMP com otimização taxa-distorção e o PJ-MMP foram implementados em *linguagem C*. Implementamos também uma estratégia *Prune-Join* para o codificador SM-MMP (*Side-Match MMP*) [6], [7], que gera dicionários de estado baseados em um critério de continuidade interblocos. O algoritmo resultante foi denominado de SI-MMP (*Smooth-Image MMP*), devido ao seu bom desempenho em imagens suaves. Os algoritmos mencionados foram utilizados para codificar imagens em níveis de cinza com resolução de 8 bits, divididas em blocos  $B^j$  com tamanho de  $16 \times 16$  e processadas da esquerda para a direita e de cima para baixo.

Os dicionários iniciais são construídos utilizando-se 64 elementos entre 0 e 255, com um passo  $\delta$  dado por  $\delta = 255/(64 - 1)$ . Para o MMP original trabalhando com blocos de  $16 \times 16$ , a quantidade de dicionários  $w$  é igual a 9. Para o PJ-MMP e o SI-MMP  $w$  vale 22, devido a união de blocos. Os elementos nas outras escalas são obtidos, em todos os casos, utilizando-se a transformação de escala bidimensional, definida na seção II, de blocos  $(1 \times 1)$  para as escalas correspondentes.

Os testes foram realizados com as imagens Lena, F-16 e Aerial, todas de dimensões  $512 \times 512$  e obtidas no site

<http://sipi.usc.edu/services/database/Database.html>, e com a imagem PP1209 obtida da digitalização da página 1209 do *IEEE Transactions on Image Processing*, volume 9, número 7, de julho de 2000. Esta última contém um misto de imagens em níveis de cinza (duas versões comprimidas da Lena), texto, fórmulas e gráficos. Os resultados para o SPIHT [4] e o JPEG [10] também são apresentados, possibilitando uma comparação imediata entre os métodos de compressão.

As curvas taxa-distorção obtidas para a imagem PP1209 podem ser vistas na Figura 7. Podemos verificar que tanto o PJ-MMP quanto o SI-MMP mantiveram os excelentes desempenhos para este tipo de imagem, com resultados aproximadamente 2dB acima do SPIHT e 4dB acima do JPEG. É importante comentar que o SI-MMP obteve um pequeno ganho de 0,15dB.

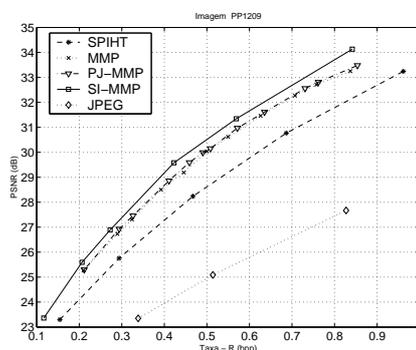


Fig. 7. Desempenho taxa-distorção para PP1209  $512 \times 512$ .

Para a imagem Lena, as curvas taxa-distorção são apresentadas na Figura 8, na qual é possível perceber um ganho de 0,6dB para PJ-MMP em relação ao MMP original. Quanto ao SI-MMP, tivemos uma melhora significativa em relação ao MMP original, ficando em torno de 1,5dB. Entretanto, o SPIHT ainda apresenta desempenhos melhores para esta imagem.

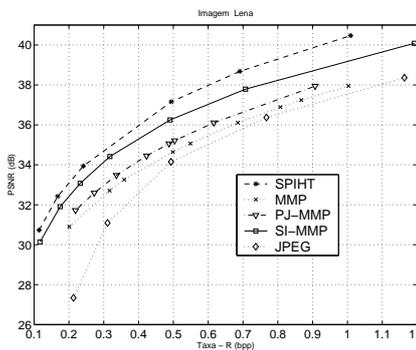


Fig. 8. Desempenho taxa-distorção para Lena  $512 \times 512$ .

O desempenho para as imagens F-16 e Aerial está evidenciado nas Figuras 9 e 10, as quais mostram que o desempenho do PJ-MMP foi superior ao do MMP original. O SPIHT e o SI-MMP apresentaram praticamente os mesmos resultados para a imagem F-16 até a taxa de 0,5 bpp. Para a imagem Aerial, o SI-MMP foi levemente superior.

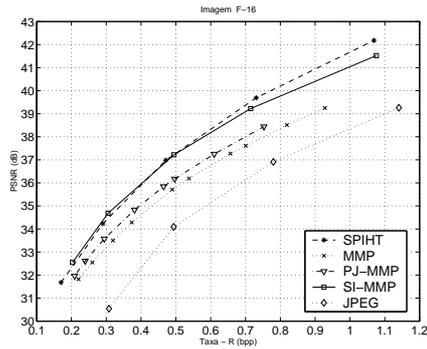


Fig. 9. Desempenho taxa-distorção para F-16 512 x 512.

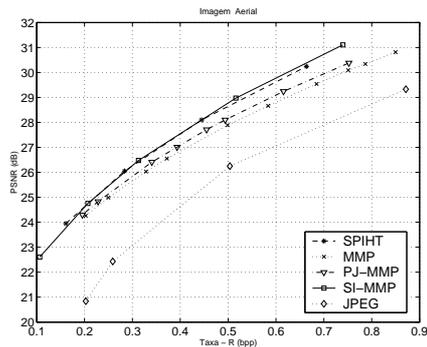


Fig. 10. Desempenho taxa-distorção para Aerial 512 x 512.



(a) MMP a 0,30 bpp. PSNR = 32,41dB.



(b) SI-MMP a 0,30 bpp. PSNR = 34,18dB.

Fig. 11. Imagem Lena - Detalhe.

A Figura 11 mostra a imagem Lena comprimida pelo MMP original e pelo SI-MMP, a 0,3 bpp. Podemos perceber uma grande melhora subjetiva (principalmente na região do nariz), principalmente devido à redução do efeito de blocagem. Os resultados objetivos estão indicados nas legendas correspondentes.

## V. CONSIDERAÇÕES FINAIS

Neste artigo, propomos uma extensão bidimensional do Prune-Join para ser empregada no algoritmo de compressão de sinais MMP, chamada de Prune-Join MMP (PJ-MMP). Com esta, torna-se possível realizar uma codificação conjunta dos blocos resultantes do procedimento de otimização da árvore de segmentação, além de permitir uma extensão da partição do MMP original. Tal extensão foi proposta com o objetivo de melhorar o desempenho do MMP na compressão de imagens suaves, sem prejudicar os bons resultados para imagens não suaves. Com o algoritmo proposto, consegue-se resultados superiores aos do MMP original, além de uma maior qualidade subjetiva da imagem reconstruída. Vale ressaltar, ainda, a relevância da pesquisa com o objetivo de aperfeiçoar a classe de métodos que utilizam recorrência de padrões multiescalas, devido ao seu comportamento universal e grande potencial para desenvolvimentos futuros.

## AGRADECIMENTOS

Este trabalho é resultado do programa de Mestrado interinstitucional UFAM-UFRJ/COPPE, financiado pela Superintendência da Zona Franca de Manaus-SUFRAMA. Os autores

gostariam de agradecer à coordenadoria do programa de mestrado e à SUFRAMA pelo apoio recebido para a sua realização.

## REFERÊNCIAS

- [1] M. B. de Carvalho, E. A. B. da Silva and W. A. Finamore, "Multi-dimensional Signal Compression using Multiscale Recurrent Patterns", *Elsevier's Signal Processing*, Vol. 82, No. 11, pp. 1559-1580, November 2002.
- [2] K. G. Vivek, "Theoretical Foundations of Transform Coding", *IEEE Signal Processing Magazine*, v.18, n. 5, pp. 9-21, 2001.
- [3] D. S. Taubman, M. W. Marcellin, *JPEG2000: Image Compression Fundamentals, Standards, and Practice*, Kluwer Academic Publishers, 2001.
- [4] A. Said and W.A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.6, pp.243-250, June 1996.
- [5] C. E. Shannon, *A Mathematical Theory of Communication*, Bell Syst. Tech. Journal, v. 27, 1948.
- [6] E. B. L. Filho, M. B. Carvalho and E. A. B. da Silva, "Multidimensional signal compression using multi-scale recurrent patterns with smooth side-match criterion," *IEEE International Conference on Image Processing*, Singapore, October, 2004.
- [7] T. Kim, "Side Match and Overlap Match Vector Quantizers for Images", *IEEE Transactions on Image Processing*, Vol. 1, No. 2, pp. 170-185, February 1992.
- [8] Shukla, Dragotti, Minh and Vetterli, "Rate-Distortion Optimized Tree Structured Compression Algorithms for Piecewise Smooth Images", *IEEE Transactions on Image Processing*, vol.14, No. 3, pp. 343-359, March 2005.
- [9] M. M. Denn, *Optimization by Variational Methods*, McGraw-Hill Book Company, 1969.
- [10] W. B. Pennebaker, J. L. Mitchell "JPEG Still Image Data Compression Standard", New York, Van Nostrand Reinhold, 1993.