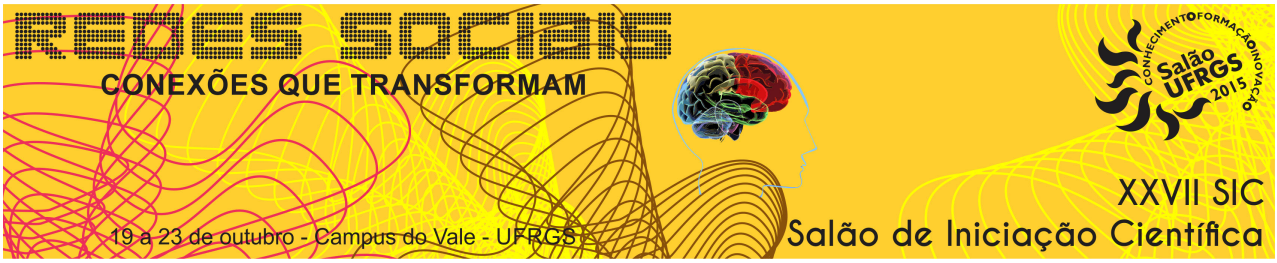

Evento	Salão UFRGS 2015: SIC - XXVII SALÃO DE INICIAÇÃO CIENTÍFICA DA UFRGS
Ano	2015
Local	Porto Alegre - RS
Título	Adaptação em Tempo de Execução do Tiling Usado na Codificação de Vídeo HEVC
Autor	GIOVANI MASSIERO MALOSSI
Orientador	ALTAMIRO AMADEU SUSIN

Adaptação em Tempo de Execução do *Tiling* Usado na Codificação de Vídeo HEVC

Giovani Massiero Malossi e Altamiro Amadeu Susin

Instituto de Informática – Universidade Federal do Rio Grande do Sul

A demanda por resoluções de vídeo digital cada vez maiores traz desafios técnicos em como lidar com a enorme quantidade de dados. O novo padrão de codificação de vídeo, HEVC (*High Efficiency Video Coding*), traz grandes melhoras à taxa de compressão quando comparado ao seu antecessor, mas leva a um mais elevado custo computacional. Ao mesmo tempo, processadores *multi-core* têm se tornado lugar-comum em uma enorme gama de aplicações.

O HEVC introduziu diversas ferramentas para auxiliar o aproveitamento de paralelismo na sua execução. Uma delas, *tiles*, são partições retangulares da imagem original, independentes entre si em termos de dados (diz-se que há quebra de contexto), que podem ser processados em paralelo, o que idealmente levaria a um tempo de codificação N vezes mais curto quando usados N *tiles*. No entanto, sem nenhum algoritmo que ajuste o *tiling* (quantidade e tamanho dos *tiles*) ao longo da codificação, o paralelismo chega a resultados próximos do ideal apenas em casos bastante específicos. Além disso, há a possibilidade de a quantidade de núcleos de processamento disponíveis para o codificador não permanecer constante durante o processo.

Já existem, na literatura, algoritmos que consideram o cenário de disponibilidade de núcleos fixa e igual ao número de *tiles*. Como o próximo quadro do vídeo só pode começar a ser codificado quando todos os *tiles* do quadro anterior terminaram, o problema é que, devido a características de textura e movimento intrínsecas de cada vídeo, alguns *tiles* demoram mais que os outros, havendo então um desbalanceamento do esforço computacional necessário para cada um. Esses algoritmos visam escolher tamanhos de *tile* que refaçam esse balanço (mais área para regiões simples, menos para complicadas), mas não modificam a quantidade.

Quando há variabilidade no número de núcleos, o problema é fazer o balanço na distribuição de esforço computacional entre eles, que podem ter que processar mais de um *tile*. Nessa situação, o tamanho e a quantidade, além da ordem de alocação, são todos fatores importantes. É interessante diminuir a quantidade de *tiles* sempre que possível, para haver menos quebras de contexto, que são prejudiciais à qualidade da compressão, pois o aproveitamento desses contextos e das redundâncias são a essência da codificação de vídeo.

Executamos diversas codificações usando o software de referência HM (HEVC *test Model*), com modificações para medir tempo por *tile*, usando sequências de vídeo e configurações diferentes. Observamos que, muitas vezes, mais *tiles* não levam a um aumento de velocidade significativo quando não há núcleos suficientes disponíveis, e com essa informação montamos uma tabela que dá, para cada número de núcleos disponíveis, a melhor configuração de *tiles*, levando em conta tanto a velocidade quanto a perda de qualidade em quebras de contexto. Para verificar nosso método, sintetizamos situações de variabilidade de núcleos de processamento, modificamos o HM novamente para ajustar, no início de cada frame, o *tiling* utilizado para ficar de acordo com a disponibilidade e tabela, e então executamos novas codificações.

Nosso método de ajuste fez com que, em comparação com o uso de um *tiling* fixo, provocássemos, em média, apenas metade da perda de qualidade causada pelo uso de paralelismo, sem afetar muito os ganhos de velocidade. Ou seja, parte do tempo antes perdido por haver núcleos parados, agora foi utilizado melhorando a qualidade da codificação.

No entanto, nosso método emprega bastante aproximação, e por ser baseado em uma tabela fixa e construída por resultados empíricos, não apresenta muita flexibilidade para casos extremos. O próximo passo do trabalho seria refinar o algoritmo para torna-lo genérico e se aproximar ainda melhor do caso ideal.