BALANCEAMENTO DE CARGA DE TRABALHO ENTRE TILES DO PADRÃO HEVC ATRAVÉS DE REDUÇÃO DE COMPLEXIDADE SELETIVA

IAGO STORCH, BRUNO ZATT, DANIEL PALOMINO, LUCIANO AGOSTINI
Universidade Federal de Pelotas – *Video Technology Research Group* (ViTech)
{icstorch, zatt, dpalomino, agostini}@inf.ufpel.edu.br

1. INTRODUÇÃO

Os avanços tecnológicos na área de captura e exibição de vídeo permitiram um aumento significativo na qualidade destes serviços, entretanto, esse aumento de qualidade levou a um aumento significativo na taxa de bits necessária para representar tais conteúdos. Para lidar com esse aumento na taxa de bits, novos padrões de compressão de vídeo foram criados, e entre eles, está o *High Efficiency Video Coding* (HEVC). O padrão HEVC herdou ferramentas do seu predecessor H.264 além de incorporar novas ferramentas, e através disso, conseguiu atingir o dobro da taxa de compressão do H.264 para uma mesma qualidade visual (OHM, 2012). Esse aumento na taxa de compressão, no entanto, veio ao custo de um aumento significativo de complexidade nos algoritmos de compressão, e para contornar esse aumento de complexidade, o HEVC foi projetado visando suporte a processamento paralelo.

Uma das novas ferramentas que possibilita o processamento paralelo é chamada de *Tile*, o qual consiste em arranjos retangulares compostos por um número inteiro de *Coding Tree Units* (CTUs) delimitados por *boundaries*, que são linhas horizontas e verticais cortando o quadro. Embora os *boundaries* cortem completamente cada quadro, o arranjo dos Tiles pode ser ajustado através do número e do posicionamento dos *boundaries*, sendo assim, é possível que um quadro seja dividido em *Tiles* de tamanho uniforme ou não-uniforme, no entanto, essas configurações devem ser feitas num momento anterior à compressão.

Como o processo de compressão de vídeo explora redundâncias espaciais e temporais, e a maioria dos algoritmos são iterativos e/ou recursivos, regiões homogêneas e estáticas demandam significativamente menor esforço computacional para serem comprimidas quando comparadas a regiões com textura e movimento, e isso pode fazer com que determinados *Tiles* demandem um tempo de compressão maior do que os demais, criando assim um cenário indesejado. Como o tempo de compressão de um quadro é dado pelo maior tempo de compressão dentre seus *Tiles*, um desbalanceamento na carga de trabalho entre os *Tiles* prejudica o *speedup* proporcionado pelo uso dos mesmos.

Visando contornar esse problema, um trabalho prévio (STORCH, 2016) propôs uma técnica de particionamento dinâmico que adapta o particionamento dos *Tiles* ao conteúdo do vídeo, tentando minimizar o desbalanceamento de carga de trabalho entre eles. Essa técnica apresenta resultados satisfatórios, no entanto, mudar frequentemente o tamanho dos *Tiles* pode diminuir o desempenho de sistemas com memória *on-chip* limitada. Visto que com essa técnica uma CTU pode mudar de *Tile* a qualquer momento, e só é possível manter um número limitado de referências na memória, pode ser necessário fazer acessos à memória *off-chip* frequentemente para buscar referências dentro do novo *Tile* da respectiva CTU, aumentando significativamente os acessos à memória *off-chip* e reduzindo o desempenho geral do sistema. Sendo assim, o objetivo deste trabalho é desenvolver uma técnica de balanceamento de carga de trabalho entre *Tiles* mantendo um particionamento fixo de *Tiles*, visando assim aumentar o *speedup* enquanto mantendo a taxa de acessos à memória.

2. METODOLOGIA

Para realizar o desenvolvimento da técnica proposta, o *software* de referência do padrão HEVC, o *HEVC Test Model 16.0* (HM-16.0), foi modificado para extrair o tempo de compressão por CTU, e como o conteúdo de dois quadros consecutivos costuma ser muito semelhante, o tempo de compressão por CTU co-localizadas em quadros consecutivos também tende a ser similar, sendo assim, é possível estimar o tempo de compressão de uma CTU do quadro atual analisando a CTU co-localizada no quadro anterior.

Sabendo quais *Tiles* são mais custosos, é necessário reduzir a carga de trabalho das suas CTUs para que a carga de trabalho por *Tile* se torne homogênea, e para isso, a profundidade máxima de suas *Coding Units* (CUs) foi limitada. Este parâmetro foi escolhido baseado num estudo (GRELLERT, 2016) que revelou que reduzir a profundidade máxima das CUs é o parâmetro que apresenta o melhor custo-benefício, além de permitir um ajuste fino através da limitação em diferentes níveis. Dentro deste trabalho, este parâmetro é nomeado *Profundidade Máxima da Partição* (PMP), sendo que ele pode variar de 0 (quando a CTU não é subdividida) a 3 (onde a CTU é subdividida em 3 níveis). A taxa de redução na carga de trabalho por redução de PMP é discutida em (GRELLERT, 2016) e apresentada no pseudocódigo da Figura 1.

Escolhido o parâmetro de atuação, o HM-16.0 foi modificado para que seja possível capturar e limitar a profundidade máxima das CTUs durante a compressão. Com as informações a respeito do tempo de compressão e profundidade máxima das CTUs, foi possível desenvolver o algoritmo para determinar quais CTUs devem ter suas PMP limitadas, e em qual intensidade. Os passos do algoritmo proposto estão apresentados na Figura 1.

A entrada do algoritmo é um vetor de estruturas contendo informações pertinentes às CTUs, como a profundidade máxima atingida durante a compressão, o tempo de compressão, e a posição dela dentro do quadro. As **linhas 1-3** representam uma etapa de pré-processamento, onde as CTUs são ordenadas em ordem decrescente de carga de trabalho e divididas entre seus respectivos *Tiles* (**linha 1**), a carga de trabalho de cada *Tile* é calculada com base na carga de suas respectivas CTUs (**linha 2**), e a carga de trabalho alvo é escolhida com base na menor carga de trabalho dentre os *Tiles* (**linha 3**).

As **linhas 4-19** representam os três estágios de atuação. Na **linha 4** um laço

```
Entrada: CTUs[]
 1: tiles[].CTUs[] ← ordenaDescendentePorTile(CTUs[])
 2: tiles[].workload ← soma(tiles[].CTUs[])
 3: workloadAlvo ← min(tiles[].workload)
 4: para todo tile em tiles[] faça
 5:     para todo ctu em tile enquanto tile.workload < workloadAlvo faça
 6:         se ctu.profAnt == 3 então
 7:             ctu.profMax ← 2
 8:             ctu.workload ← ctu.workload * 0,75
 9:             tile.workload ← soma(tile.CTUs[])
10:     para todo ctu em tile enquanto tile.workload < workloadAlvo faça
11:         se ctu.profAnt == 2 ou ctu.profMax == 2 então
12:             ctu.profMax ← 1
13:             ctu.workload ← ctu.workload * 0,6
14:             tile.workload ← soma(tile.CTUs[])
15:     para todo ctu em tile enquanto tile.workload < workloadAlvo faça
16:         se ctu.profAnt == 1 ou ctu.profMax == 1 então
17:             ctu.profMax ← 0
18:             ctu.workload ← ctu.workload * 0,55
19:             tile.workload ← soma(tile.CTUs[])
```

Figura 1 – Pseudocódigo da técnica proposta

avalia todos os *Tiles* buscando reduzi-los para a carga de trabalho alvo. As **linhas 5-9** representam o primeiro estágio de atuação, onde as CTUs que atingiram a profundidade 3 na compressão anterior serão reduzidas para alcançar PMP 2 na próxima compressão. Para isso, a **linha 5** avalia todas as CTUs do *Tile* ou até que a carga de trabalho alvo seja atingida. Caso a CTU se enquadre nos requisitos do estágio (ter atingido profundidade 3 na última compressão), sua PMP é limitada para 2 (**linha 7**), a carga de trabalho da mesma é ajustada (**linha 8**), e a carga de trabalho do *Tile* é atualizada com base nisso (**linha 9**). As **linhas 10-14** representam o segundo estágio, onde as CTUs que atingiram profundidade 2 na última compressão, ou foram limitadas no estágio anterior, serão limitadas a atingir PMP 1 na próxima compressão. As linhas **15-19** representam o último estágio, onde é feita a limitação para PMP 0 na próxima compressão.

Buscando evitar a propagação de erros e usar referências de tempo confiáveis, a técnica trabalha sobre conjuntos de 4 quadros: o primeiro quadro do conjunto é comprimido sem qualquer interferência, então o algoritmo é rodado e aplica as modificações aos próximos três quadros.

3. RESULTADOS E DISCUSSÃO

Após implementar a técnica proposta, o desempenho da mesma foi avaliado através da compressão de uma série de vídeos. Para isso, três parâmetros foram levados em consideração: (1) *speedup*, que mede quantas vezes a compressão paralela foi mais rápida do que a compressão sequencial, (2) média de tempo salvo (ATS), que mede o quão mais rápida a compressão paralela foi em relação à compressão sequencial, e (3) BD-Rate, que mede, para uma mesma qualidade, o aumento na taxa de bits quando utilizando uma técnica em relação a outra.

Para extrair os resultados, um conjunto de vídeos foi comprimido utilizando um particionamento de *Tiles* uniforme tradicional, e em seguida, o mesmo conjunto de vídeos foi comprimido utilizando o particionamento uniforme com a técnica proposta. Durante este processo, o tempo de compressão por *Tile* foi extraído, e isso possibilitou calcular o *speedup* e ATS para comparar com (STORCH, 2016), como apresentado na Tabela 1, onde as colunas “Ganho” representam o ganho de desempenho da técnica proposta quando comparada com (STORCH, 2016). É importante salientar que a referência para os resultados de *speedup* é uma compressão sem *Tiles*, enquanto que para os resultados de ATS é uma compressão com *Tiles* uniformes.

Avaliando os resultados apresentados na Tabela 1, é possível perceber que a técnica proposta apresentou um ganho de *speedup* na maioria dos casos. Em média, é possível observar um ganho de 1,9% e 2,3% para os particionamentos 2×2 e 3×3, respectivamente, e uma redução de *speedup* de 1,4% para o particionamento 4×4. Já quando avaliamos o ATS, é possível evidenciar um ganho de ATS significativo, visto que houve um ganho de 11,8%, 20,2%, e 18,2% para os particionamentos 2×2, 3×3, e 4×4, respectivamente. Essa diferença entre os resultados de *spedup* e ATS se justificam pela característica que cada métrica avalia: ao simplificar-se algumas CTUs é possível balancear ou não a distribuição de carga de trabalho entre os *Tiles*, o que melhora ou não o *speedup*, mas certamente reduz a carga de trabalho total, o que leva a uma melhora no ATS.

Após avaliar os resultados de paralelismo, a eficiência de compressão foi avaliada. Os resultados de BD-Rate, representando a eficiência de compressão em relação a uma compressão sem *Tiles*, são apresentados na Tabela 2. Vale ressaltar que como o BD-Rate representa um aumento de *bitrate*, valores positivos representam uma redução na eficiência de compressão. Ao avaliar os

Tabela 1 – Resultados de *speedup* e ATS

Resolução	Tiles	*Speedup*				ATS		
		Uniforme	Storch, 2016	Proposto	Ganho	Storch, 2016	Proposto	Ganho
832×480	2×2	3,4	3,5	3,7	5,7%	1,7%	19,5%	17,8%
	3×3	6,2	6,6	7,4	11,9%	6,8%	39,7%	32,9%
	4×4	11,9	12,5	12,1	-2,9%	5,2%	21,2%	16,0%
1280×720	2×2	3,6	3,7	3,7	0,2%	1,3%	9,1%	7,8%
	3×3	6,8	7,7	7,1	-7,8%	12,9%	21,8%	8,9%
	4×4	11,5	12,7	12,7	0,1%	10,1%	30,5%	20,5%
1920×1080	2×2	3,5	3,7	3,7	-0,2%	6,6%	16,4%	9,7%
	3×3	7,0	7,6	7,9	2,9%	8,4%	27,1%	18,7%
Média	2×2	3,5	3,6	3,7	1,9%	3,2%	15,0%	11,8%
	3×3	6,7	7,3	7,5	2,3%	9,4%	29,5%	20,2%
	4×4	11,7	12,6	12,4	-1,4%	7,6%	25,9%	18,2%

Tabela 2 – Resultados de eficiência de compressão em BD-Rate

Tiles	Uniforme	Storch, 2016	Proposto	Diferença
2×2	1,44%	1,41%	5,97%	4,56%
3×3	2,89%	2,97%	17,53%	14,56%
4×4	5,57%	5,50%	16,87%	11,36%

resultados de BD-Rate da Tabela 2, percebe-se que houve um aumento de BD-Rate para todos os particionamentos, ou seja, uma redução na eficiência de codificação, que fica entre 4,56% e 14,56% para os três particionamentos avaliados. Essa redução na eficiência de codificação é esperada, visto que a técnica proposta simplifica o processo de compressão sem avaliar o impacto que isso pode causar na qualidade do vídeo.

4. CONCLUSÕES

Este trabalho apresentou uma técnica de balanceamento de carga computacional entre *Tiles* visando aumentar o *speedup* e a redução de tempo média proveniente do uso dos mesmos, sem interferir no particionamento de *Tiles*. Resultados experimentais apontaram que a técnica proposta atingiu resultados superiores a trabalhos relacionados, mesmo que através de uma redução na eficiência da compressão. Com base nos resultados, é possível concluir que a técnica proposta pode ser utilizada para melhorar o desempenho em sistemas que exijam *Tiles* estáticos, além disso, trata-se de uma abordagem inovadora para a solução deste problema.

5. REFERÊNCIAS BIBLIOGRÁFICAS

OHM et al. Comparison of the Coding Efficiency of Video Coding Standards – Including High Efficiency Video Coding (HEVC). **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1669-1684, 2012.

STORCH et al. Algoritmo de Particionamento Dinâmico de Tiles Baseado em Histórico para Ganho de Paralelismo no Padrão de Compressão de Vídeo HEVC. **XXV Congresso de Iniciação Científica**. Pelotas, 2016.

GRELLERT et al. Complexity Control of HEVC Encoders Targeting Real-Time Constraints. **Journal of Real-Time Image Processing**, v. 13, n. 1, p. 5-24, 2016.