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

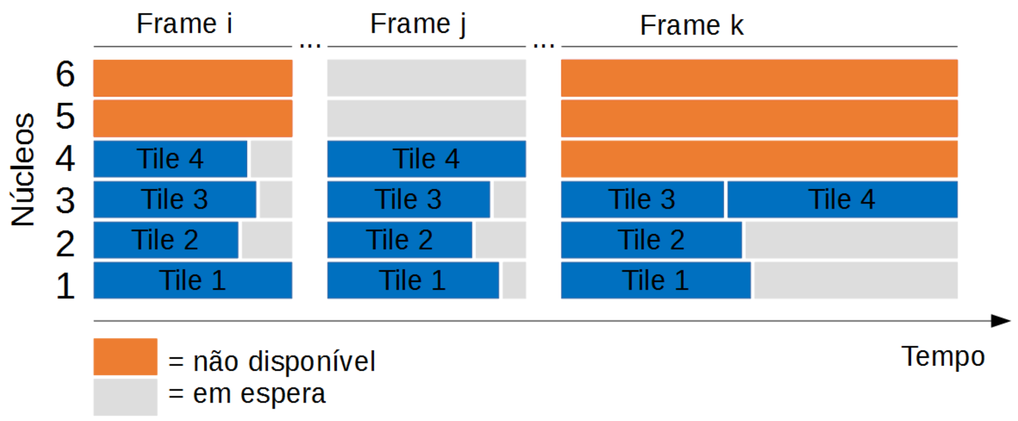
Giovani Massiero Malossi & Altamiro Amadeu Susin
Universidade Federal do Rio Grande do Sul - Instituto de Informática
gmmalossi@inf.ufrgs.br

Introdução

- Com resoluções de vídeo cada vez maiores, torna-se necessário melhor comprimir as informações neles contidas, para a transmissão e a armazenagem continuarem viáveis. O novo padrão de codificação, HEVC, traz grandes ganhos em compressão quando comparado ao seu antecessor, H264/AVC, ao custo de um maior esforço computacional. [2]
- Para mitigar esse aumento de esforço computacional, o padrão traz novas estruturas que auxiliam na paralelização da codificação, que é uma boa escolha uma vez que processadores multi-core estão em todas as plataformas atualmente, de embarcados a servidores. A estrutura que estudamos é o tile, que permite dividir cada frame em blocos menores e codificá-los ao mesmo tempo.
- No entanto, devido à natureza das dependências de dados da codificação, é muito difícil paralelizar sem perder alguma qualidade de compressão. Somado aos overheads e à não-homogenidade do custo computacional ao longo de um frame, que limitam os ganhos de velocidade, conclui-se que não é benéfico utilizar um particionamento com tiles muito agressivo. [1]

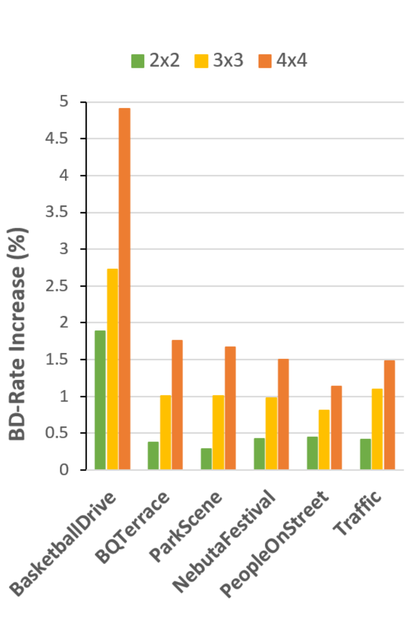
Desafio

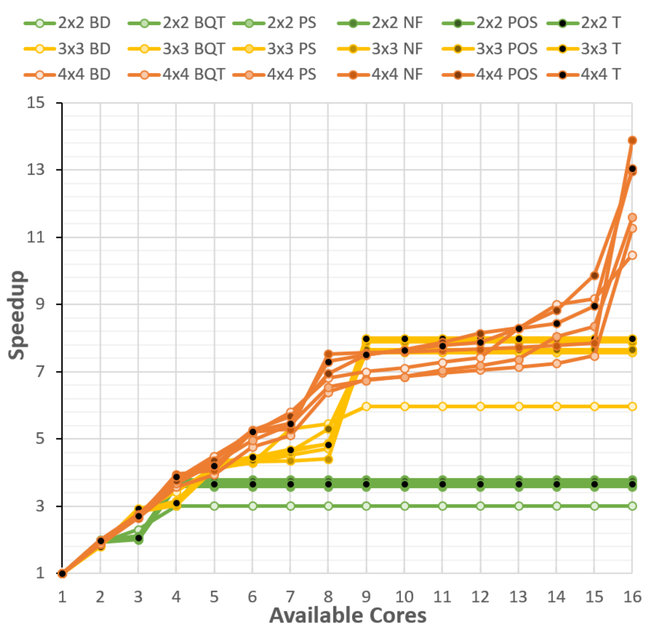
- A disponibilidade de núcleos de processamento para o codificador não é necessariamente constante: questões como prioridades das outras aplicações rodando no sistema e restrições energéticas podem causar variabilidade.
- Essa variabilidade não é tratada por padrão, e pode causar grande impacto negativo durante a codificação com tiles. A seguir três cenários ilustrativos:
  - Frame i) A situação ideal. O número de tiles é o número de núcleos e o tempo não utilizado é somente aquele devido à distribuição desigual de custo computacional, intrínseco de cada vídeo.
  - Frame j) Há mais recursos disponíveis do que o particionamento escolhido pode aproveitar; perde-se oportunidade de aceleração.
  - Frame k) A pior situação: o particionamento já ocorreu, e com ele as quebras das dependências que levam à perda de qualidade de compressão. No entanto, como não há núcleos o bastante, um dos tiles é processado depois dos outros, prejudicando fortemente o tempo de codificação, uma vez que o próximo frame só pode começar quando todos os tiles do atual terminaram.



Metodologia

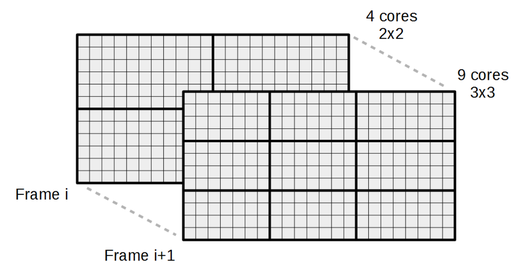
- Rodamos diversas codificações medindo o tempo que cada tile precisou para ser processado e a qualidade da compressão da codificação como um todo. Consideramos partições até 4x4 e um sistema de 16 núcleos, lembrando que particionamento muito agressivo não vale a pena.
- Medimos a qualidade de compressão com a métrica BD-rate, bastante utilizada na comunidade, que deve ser otimizada. Ou seja, quanto mais BD-rate pior.
- Medimos o ganho no tempo com a taxa entre o tempo de codificação com paralelização e sem, a qual também chamamos de Speedup. Para calcular o tempo com a disponibilidade de núcleos de 1 a 16, consideramos que o sistema escalona o próximo tile no núcleo que ficar disponível primeiro.





Solução Proposta

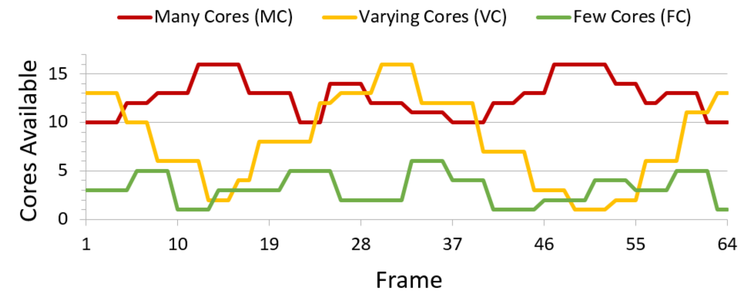
- Quando há variação na disponibilidade dos núcleos, é interessante variar o tiling de acordo. Assim, ficamos mais próximos de utilizar o máximo dos recursos computacionais do sistema com o mínimo de perdas por particionamento excessivo.
- A granularidade mais fina possível é a nível de frame e queremos decidir o particionamento com base no número de núcleos disponíveis. Um algorítmo muito complexo pode prejudicar o tempo total, pois terá que rodar de novo a cada frame; uma simples tabela, construída com análise estatística pode ser eficaz o bastante.
- Observando o gráfico anterior, vemos zonas em que os speedups de tilings diferentes ficam bastante próximos. Nesses casos, devemos escolher a menor partição, pois prejudica menos a qualidade de compressão.

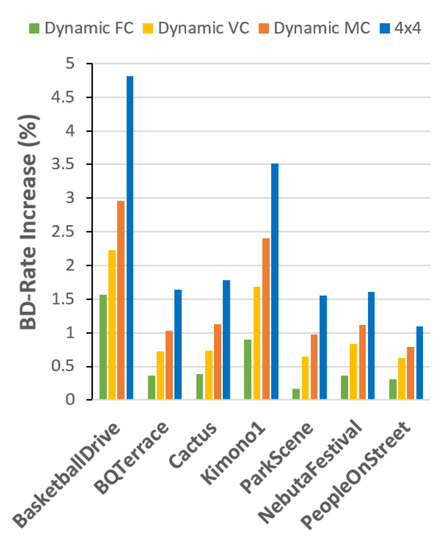


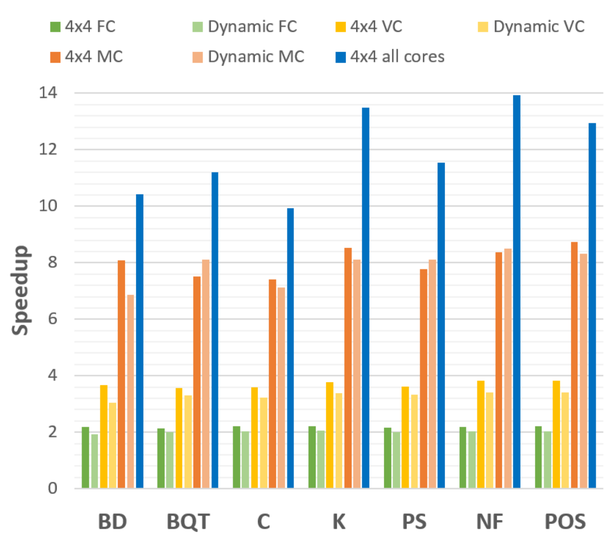
Núcleos Disponíveis	Particionamento em Tiles
1	1x1
2-8	2x2
9-15	3x3
16	4x4

Resultados

- Para testar nossa solução, modificamos o programa que implementa o padrão (HM - HEVC test Model) para utilizar nossa tabela para definir o tiling e o submetemos a algumas situações sintéticas de variabilidade de recursos.







Conclusão

- Como podemos observar, a simples adição de um tratamento à variabilidade de núcleos nos possibilitou derrubar as perdas com particionamento excessivo, ao mesmo tempo que mantivemos quase totalmente os ganhos de velocidade oferecidos pelo paralelismo.

Trabalhos Futuros

- Estamos estudando o uso de algoritmos de balanceamento de custo computacional em situações com variabilidade de recursos; também estamos aumentando a granularidade dos tilings inclusos na tabela, para melhorar a aproximação do ideal.

Referências

[1] C. Chi et al. Parallel scalability and efficiency of hevc parallelization approaches. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1827–1838, 2012.

[2] ITU-T and ISO/IEC. High efficiency video coding. *ITU-T Recommendation H.265 and ISO/IEC 23008-2*, 2013.