PARALELIZAÇÃO DE ALGORITMO DE RECLASSIFICAÇÃO DE RESULTADOS DE RECUPERAÇÃO DE IMAGENS BASEADA EM CONTEÚDO UTILIZANDO ESPAÇOS CONTEXTUAIS

Flávia Pisani
flavia.pisani@students.ic.unicamp.br

Daniel C. G. Pedronette
daniel@rc.unesp.br

Ricardo da S. Torres e **Edson Borin**
{rtorres,edson}@ic.unicamp.br

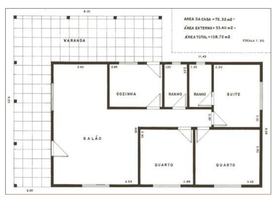
INSTITUTO DE COMPUTAÇÃO (IC)

Agradecimentos a AMD, FAPESP, CAPES e PIBIC/CNPq

Recuperação de Imagens - Computação Paralela - Reordenação de Imagens

Motivação

- Queda no custo de dispositivos de armazenamento
- Avanços tecnológicos: maiores coleções de imagens
- Sistemas de busca são importantes
- Desafio: descrever uma imagem em palavras-chave ou metadados textuais é subjetivo, ex.: *planta*



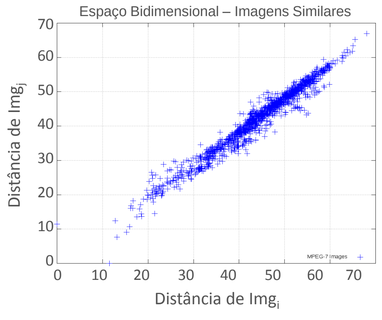
- Uma solução: **Recuperação de Imagem por Conteúdo (CBIR)** – dada uma imagem de consulta, um sistema de CBIR visa recuperar imagens similares em uma coleção através do uso de um ou mais descritores de conteúdo, que codificam propriedades visuais da imagem, como forma, cor e textura
- Há tentativas bem-sucedidas de aumentar a eficácia dos resultados através de métodos de **reclassificação**, mas ainda **falta eficiência**!

- CPUs multinúcleo e GPGPUs → Paralelização é uma área de pesquisa promissora

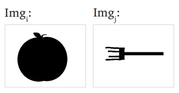
Algoritmo

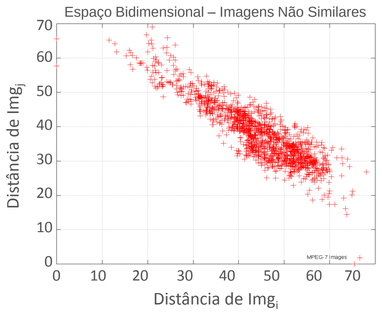
- **Que informações meus vizinhos mais próximos podem me dar sobre outras imagens da coleção?**
- Análise par a par da correlação entre imagens + informação contida no contexto da consulta
- *Espaço Contextual*: espaço bidimensional no qual todas as imagens da coleção são representadas de acordo com suas distâncias de uma imagem arbitrária e cada um dos seus K vizinhos mais próximos

Similar



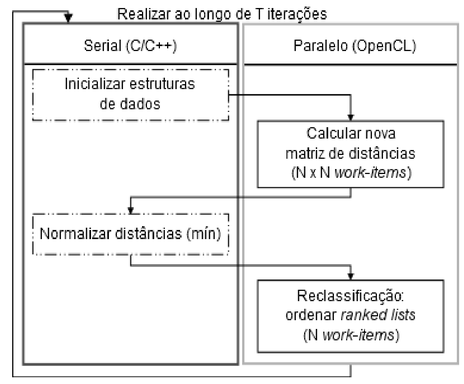
Não Similar





Paralelização

- Os dois passos do algoritmo foram divididos em dois *kernels* OpenCL para respeitar seus critérios de sincronização
- ***Cálculo de Distâncias***: atualiza a distância entre um par de imagens considerando o espaço contextual formado pelos K vizinhos mais próximos da imagem de consulta (executado paralelamente para cada par de imagens)
- ***Ordenação de Ranked Lists***: reordena o *ranked list* de cada imagem considerando as novas distâncias calculadas pelo passo anterior (executado paralelamente para cada *ranked list*)
- Código serial foi adicionado para fornecer os dados de entrada necessários para cada *kernel*

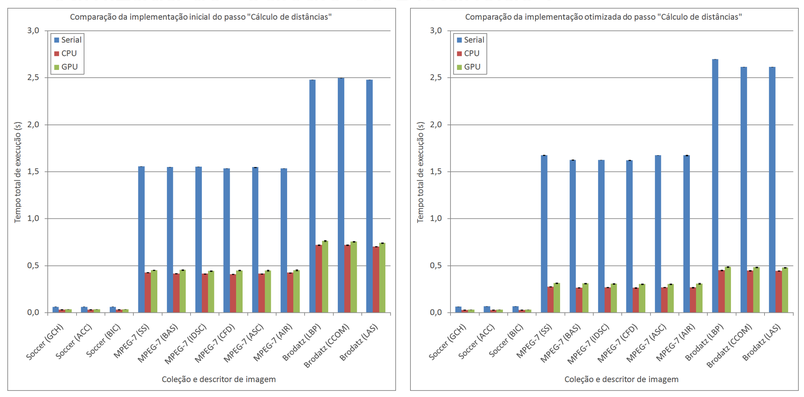


Metodologia

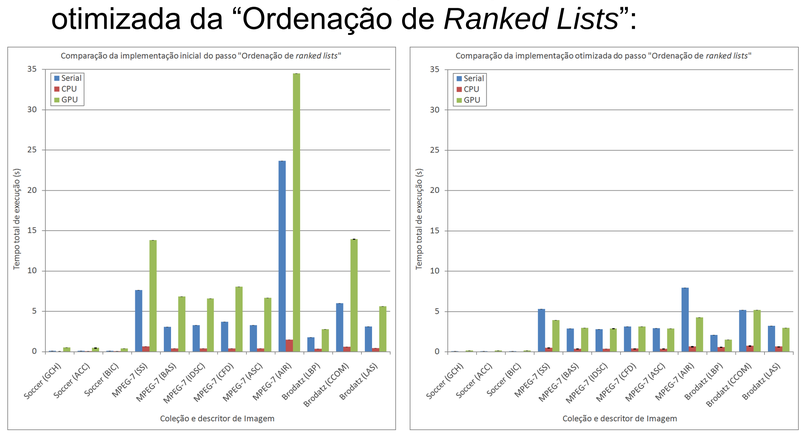
- APU: AMD A8-3850 (CPU com 4 núcleos + GPU AMD Radeon HD 6550D)
- Linux 3.3.4-5 Fedora 17
- AMD OpenCL SDK 2.8
- Coleções: **MPEG-7**, Soccer e Brodatz
- Descritores: **CFD**, SS, BAS, IDSC, ASC e AIR (forma); GCH, ACC e BIC (cor); LBP, CCOM e LAS (textura)
- **Implementação serial em C/C++ usada como base para as comparações de desempenho**
- **Acesso coalescido**: 51% de *cache hit* → i-ésima posição do *ranked list* de cada imagem é acessada paralelamente no "Cálculo de Distâncias" → transposição da matriz de *ranked lists* tal que os i-ésimos elementos fiquem na mesma linha → melhor aproveitamento dos valores da cache (84% de *hit*)
- **Acesso e uso da memória**: "Cálculo de Distâncias" acessa apenas K vizinhos mais próximos de cada imagem → diminuição da matriz de *ranked lists* de N x N para K x N; Uso de funções OpenCL mais eficientes para mapear as matrizes que serão apenas lidas
- **Simplificação da ordenação de *ranked lists***: ordenação de apenas K + 1 elementos utilizados pelo "Cálculo de Distâncias" ao invés de todos os N

Resultados

- Comparação entre os tempos totais (execução + transferência de memória) da implementação inicial e otimizada do "Cálculo de Distâncias":



- Comparação entre os tempos totais (execução + transferência de memória) da implementação inicial e otimizada da "Ordenação de *Ranked Lists*":



Conclusão

- **Eficácia da solução foi mantida**
- Coleção MPEG-7 e descritor CFD foram usados como parâmetro. *Speedups* obtidos considerando o tempo total de execução:

Cálculo de Distâncias	Ordenação de *Ranked Lists*
• Inicial: ~3,75x (CPU) e ~3,41x (GPU)	• Inicial: ~9,36x (CPU) e ~0,46x (GPU)
• Otimizada: ~**6,12x** (CPU) e ~**5,34x** (GPU)	• Otimizada: ~**7,75x** (CPU) e ~**1,00x** (GPU)
• Tempo total da versão serial inicial é ~0,95x o da otimizada	• Tempo total da versão serial inicial é ~1,18x o da otimizada

- Tempo de transferência de memória no "Cálculo de distâncias" diminuiu: ~33,8% → ~16,8% do tempo total de execução (CPU) e ~40,3% → ~20,9% (GPU)

Publicações

- PISANI, F. ; PEDRONETTE, D. C. G. ; TORRES, R. da S. ; BORIN, E. . *Contextual Spaces Algorithm Acceleration on APUs*. Em: IV Escola Regional de Alto Desempenho de São Paulo, 2013, São Carlos, SP