Algoritmo paralelo em *spark* para a transformada de distância euclidiana exata em imagens digitais de altíssima resolução

Lincoln Lima

São Paulo, Brasil

Abstract

Neste trabalho é implementada uma solução de transformada de distância euclidiana exata em imagens digitais utilizando o *framework spark* que pode em princípio ser calculada em imagens de altíssima definição, sendo o algoritmo apenas limitado pela quantidade de memória em disco disponível no cluster.

Keywords:
Euclidean Distance Transform, Image Processing, Spark, High Resolution Image

1. Transformada de distância

Transformada de distância é uma operação bastante utilizada em computação gráfica, geometria computacional e processamento de imagens podendo ser aplicada para esqueletização de modelos, segmentação de imagens, definição de rotas em robos, renderização 3D, etc. Uma das mais importantes transformadas de distância é a euclidiana exata definida em duas dimensões como segue:

$$EDT(x, y) = MIN(i, j : 0 \leq i < m \wedge 0 \leq j < n \wedge b[i, j] : (x - i)^2 + (y - j)^2)$$

Inúmeros papers apresentam eficientes transformadas de distância euclidiana em imagens como em [1, 2, 3] , no entanto, todos esses métodos são limitamos ou pela memória compartilhada da GPU ou pela memória RAM, aqui apresentamos um algoritmo, visando imagens de altíssima definição, da

ordem de centenas de *gigaBytes* de tamanho, que calcula a transformada euclidiana exata baseda em múltiplos discos sob um cluster gerenciado pelo *framework spark*.

Em palavras, o algoritmo começa em distribuindo os inúmeros *pixels* da imagem em diferentes discos, nós pelo *cluster*, em alto nível temos acesso a eles pela abstração RDD fornecida pelo *spark*, em um segundo momento atribuímos indexes únicos para cada ponto. Em seguida dois novos RDDs são gerados, um representando os pontos de objetos e o outro definindo os pontos de fundo. Então um produto cartesiano é feito entre os dois RDDs, ademais, um mapeamento neste RDD é efetuado de modo a calcular a distância euclidiana exata entre os dois pontos, posteriormente é realizado uma redução garantindo a atribuição da menor distância do ponto de fundo a um ponto de objeto e finalmente os conjuntos de *pixel* de objeto e fundo são unidos produzindo nosso mapeamento final. O código completo em jupyter notebook se encontra em github.com/llscm0202/BIGDATA2017/Projeto

Referencias

[1] F. de Assis Zampirolli, L. Filipe, A fast cuda-based implementation for the euclidean distance transform, in: High Performance Computing & Simulation (HPCS), 2017 International Conference on, IEEE, pp. 815–818.

[2] R. JBTM, H. WH, et al., A general algorithm for computing distance transforms in linear time (2002).

[3] P. Felzenszwalb, D. Huttenlocher, Distance transforms of sampled functions, Technical Report, Cornell University, 2004.