LECTURE NOTES FOR MTH 371, WINTER 2022
"LARGE SCALE DATA ALGORITHMS"

PANAYOT S. VASSILEVSKI

1. MTH 371 - Homework #4: due March 2, 2022

Problem 1.1 (Homework # 4: due March 2, 2022)**.** *Given a $n \times n$ symmetric sparse matrix A (e.g., one from the test matrices provided).*

- *Form the following graph matrices associated with the sparsity pattern of A and store them in CSR format:*
  - *Adjacency matrix $\mathbb{A} = (\alpha_{ij})$ where $\alpha_{ij} = 1$ for any non-zero entry a_{ij} of A and zero otherwise.*
  - *The edge_vertex connectivity matrix E where in row e of E we have only two nonzero entries (equal to one) at positions (e, i) and (e, j) where e runs over the edges of the graph, i.e., the pairs $e = (i, j)$ for which $a_{ij} \neq 0$.*
  - *Form the transpose of E, E^T, which is the vertex_edge relation matrix.*
  - *Form the edge_edge adjacency matrix $\mathbb{A}_E = EE^T$ as a product of two sparse matrices.*
  - *Form an edge-weight vector $\mathbf{w} = (w_e)$ where w_e is a random number between (0,1) for each edge e of the graph.*
- *Implement one step of the parallel Luby algorithm, to generate aggregates $\{\mathcal{A}\}$ which are either pairs of vertices or single vertices. The pairs of vertices are actually edges and each such edge has a locally maximal edge-weight. To select these edges, we use the Luby algorithm, namely:*
  - *We loop over each edge e of the graph and compare its weight w_e with the weights $w_{e'}$ of its neighboring edges e'. The neighbors e' we find from the edge_edge adjacency matrix $\mathbb{A}_E = EE^T$. If the weight w_e is larger than the weights $w_{e'}$ of all neighbors e' of e, we label e as an aggregate $\mathcal{A}$.*
  - *End of loop over the edges.*
- *Check if all vertices are covered by the selected edges (pairwise aggregates). For this we use the relation "vertex_edge", i.e., the matrix E^T. That is, if we find a vertex i that all its edges that it belongs to are not labeled as aggregates, we label that vertex as an aggregate, i.e., as a single vertex aggregate.*
- *Once the aggregates have been selected, i.e., have the relation "vertex_aggregate" created, we form the matrix P corresponding to that relation.*
- *Finally, form the coarse graph corresponding to the matrix $P^T \mathbb{A} P$.*
- *In summary, from a given graph represented by its adjacency matrix $\mathbb{A}$ and an edge-weight vector $\mathbf{w} = (w_e)$, we have created a coarse graph represented by its adjacency matrix $\mathbb{A}_c$ corresponding to the sparsity pattern of $P^T \mathbb{A} P$, and the "vertex_aggregate" relation matrix P relates these two graphs.*

- *Apply recursion, i.e., use the same algorithm as above, with input the coarse matrix* $\mathbb{A}_c$ *and a coarse edge weight vector* $\mathbf{w}_c$ *(e.g., random numbers in (0,1)). We stop the recursion if we have achieved required coarsening factor, i.e., the size of the coarse graph is* $\tau(= 8)$ *times smaller than the size of the original graph. The parameter* τ *is an input in the recursive algorithm.*
- *Make sure that the algorithm works for any edge-weight vector* $\mathbf{w} = (w_e)$*, i.e.,* $\mathbf{w}$ *has to be an input parameter in the Luby's algorithm.*
- *The recursive algorithm will produce on output a sequence of graphs represented by their adjacency matrices* $\{\mathbb{A}_k\}_{k=0}^{\ell}$*, where* $\mathbb{A}_0 = \mathbb{A}$ *is the initial (original) one, and a sequence of* P*-matrices,* $\{P_k\}_{k=0}^{\ell-1}$*, where* P_k *relates level* k *(fine) and level* $k+1$ *(coarse) graphs. If we want to relate level* $k+1$ *graph with the original one, we use the product* $\pi_k = P_0 P_1 \ldots P_k = \pi_{k-1} P_k$*,* $\pi_0 = P_0$*. The matrix* π_k *is the relation "*original graph vertices_$(k+1)$-level-aggregates*". Optionally, we could visualize all levels aggregates (one at the time).*

Fariborz Maseeh Department of Mathematics and Statistics, Portland State University, Portland, OR 97201
Email address: panayot@pdx.edu