Graph Partitioning with Structural Constraints

Saeed Hajebi†, Stephen Barrett†, Aidan Clarke§, and Siobhán Clarke†

†School of Computer Science and Statistics, Trinity College Dublin, Ireland.
§IBM Software Ireland Lab., Dublin, Ireland.

We formulate a graph partitioning problem with specific structural constraints, for which the current partitioning techniques are not sufficient, and propose a heuristic solution for it.

One is given a connected graph \( G = (\text{Nodes}, \text{Links}, \omega) \), consists of a set \( \text{Nodes} \) of \( N \) nodes together with a set \( \text{Links} \) of \( M \) links, connecting the nodes. Nodes are labeled as sources and consumers. Nodes have coordinates and links are fixed, undirected, and weighted \( (\omega : \text{Links} \rightarrow \mathbb{R}_{>0}) \). Each node is assigned a number at each step of time, called a supply (for a source), or demand (for a consumer). The aim is partitioning the graph into isolated components without adding any edge to the graph, in such a way that each component has a direct access to a source. This means if there is no source in a component, the paths from all nodes of the component to a source must not include any node in other components.

The objectives are: (a) minimum supply-demand imbalance, (b) minimum cut size, and (c) minimum component size imbalance.

The constraints are: (a) connectedness of components, (b) isolation of components, (c) direct access to a source for each component, and (d) components’ size, i.e., components’ size must be in a predefined boundary.

WDN-Partition

We propose WDN-Partition, which is a heuristic graph partitioning method that finds the major flow paths (by means of link weights) from the sources and considers the nodes in these paths as new potential sources; then, identifies components from these potential sources using the BFS algorithm. This will guarantee that each component has direct access to a source. Then the size of components are examined. If their size is in a predefined boundary of a proper component size, they will be considered as a component. If its size is smaller than the proper component size, they will be added to a set called Minor Islands, and we try to merge them with neighboring components if possible. If their size is larger than the proper component size, they will be added to a set called Major Islands. Then we partition the components in the Major Islands set into proper size components, using seed selection and graph growing. A near-optimal arrangement of components will be found using a multi-objective optimization algorithm.