A PARALLEL NUMERICAL ALGORITHM FOR UNDIRECTED S-T MIN-CUT

1. INTRODUCTION

As an algorithmic kernel, the undirected s-t min-cut has applications in improving graph partitions (Anderson and Lang, SODA2008), energy minimization in MRF (Boykov et al. PAMI2001, Kolmogorov and Zabih PAMI 2004), and MR imaging analysis (Bioucas-Dias and Valadao TIP2007, Trzasko et al. ISMRM2014). These applications have the features we need to solve the undirected s-t min-cut multiple times, and the graphs usually have floating point weights.

The goal of this research is to develop and implement a scalable parallel numerical algorithm for undirected s-t min-cut so that applications involving it could be solved more efficiently. Depending on applications, we would like this parallel numerical algorithm to be customizable and deployable on different platforms:

- Large scale (10^7-10^9 nodes) — distributed memory platforms.
- Medium scale (10^5-10^6 nodes) — floating point accelerators, e.g. GPUs.

2. A SEQUENCE OF LINEAR SYSTEMS FOR UNDIRECTED S-T MIN-CUT

Let \( G = (V, \varepsilon) \) be a weighted undirected graph with the edge weight function \( c((u, v)) \to 0 \) for \((u, v) \notin \varepsilon\). Without loss of generality, we assume \( G \) to be connected. Because \( G \) is undirected, we can choose an arbitrary orientation of its edges, and let the corresponding edge-node incidence matrix be \( B \in \{-1, 0, 1\}^{|V| \times |E|}\). Let \( C = \text{diag}(c((u, v))) \), \( \Phi = [e \mid e] \), \( u = [1 \ 0]^T \), then undirected s-t min-cut can be formulated as an IRLS minimization problem

\[
\begin{align*}
\min_x ||CB x||_1 \\
\text{s.t. } \Phi x = u \\
x \in \{0, 1\}^n
\end{align*}
\]

We note that similar formulations of undirected s-t min-cut as IRLS minimization have been proposed in (Bhusnurmath and Taylor PAMI2008, Chin et al. ITCS2013). We adopt the algorithm of Iteratively Reweighted Least Squares (IRLS) for solving (1). The IRLS algorithm solves the IRLS minimization problem (1) by solving a sequence of weighted least squares (WLS) problems. Given an IRLS iterate \( x^{(t-1)} \), we form the next WLS as follows:

- Step 1. Compute \( z^{(t)} = CB x^{(t-1)} \).
- Step 2. Compute the weight vector \( w^{(t)} \), of which each component is

\[
w^{(t)}_e = (z^{(2t)})^2 + c(e)^{-1/2},
\]

where \( c(e) > 0 \) is a parameter guard against the case of \( z_e \approx 0 \).
- Step 3. Let \( W^{(t)} = \text{diag}(w^{(t)}) \), the WLS problem is defined as

\[
\begin{align*}
\min_x \frac{1}{2} x^T \cdot CB \cdot CW^{(t)} \cdot CB^T x \\
\text{s.t. } \Phi x = u
\end{align*}
\]

Solving the above constrained WLS (2) leads to solving the saddle point system

\[
\begin{bmatrix}
L^{(t)} & 0 \\
\Phi & \Lambda
\end{bmatrix} \begin{bmatrix}
x \\
\lambda
\end{bmatrix} = \begin{bmatrix}
u \\
0
\end{bmatrix}
\]

where we have let \( L^{(t)} = CB \cdot CW^{(t)} \cdot CB^T \).

The connection between our IRLS approach and electrical flows (Christiano et al. STOC2011) is made explicit through the following fact.

**Fact 1.** **The WLS problem (2) solves an electrical flow problem, where the electrical flow is defined as \( x^{(t)} = CW^{(t)} CB^T x^{(t)} \).**

And its flow value is \( x^{(t)} L^{(t)} x^{(t)} \).

3. PARALLEL LINEAR SYSTEM SOLVER

We can show that the saddle point system (3) is nonsingular provided that \( \lambda \) is connected. Given the special structure of \( \Phi \), we use null space method to solve (3). Let \( L^{(t)} \) be the matrix acquired by deleting the s, t rows and columns of \( L^{(t)} \), and \( b^{(t)} \) be the vector from negating the s column and then deleting the s, t components. Applying the nullspace method (3) leads to the following linear system

\[
L^{(t)} x = b^{(t)}
\]

We call \( L^{(t)} \) the reduced Laplacian. Essentially, in each step we solve a Laplacian system. The nullspace method also allows us to prove the feasibility of \( x^{(t+1)} \), as summarized in the following fact.

**Fact 2.** **The solution \( x^{(t+1)} \) to the saddle point system (3) satisfies the constraint \( x \in \{0, 1\}^n \).**

The efficiency of solving the linear system (4) will determine the efficiency we solve the undirected s-t min-cut problem. For this purpose, we consider parallel linear system solvers. Depending on the characteristics of the graphs, we apply different parallel linear system solvers.

4. APPLICATIONS AND DATA

- Flow improve: asia-roadmap (\( |V| = 11950757, |E| = 127711603 \)) and europe-roadmap (\( |V| = 50912018, |E| = 54054660 \)).
- 2D MRI imaging analysis: T1 estimation (T1), steady state MR elastography (MRE), and Phase Unwrapping (PU). The grid size is 256 x 256, and we use the 8-point neighbor system.

5. RESULTS

We report some preliminary results using a MATLAB implementation of our IRLS based undirected s-t min-cut algorithm (denoted as MINCUT-IRLS in the following). In both flow improve and 2D MRI, we solve a sequence of undirected s-t min-cut, for which we adopt two heuristics:

- Inexact solution. We don’t solve each undirected s-t min-cut instance to completion. Instead, we fix the number of IRLS iterations to be 50.
- Warm start. We use the \( x \) vector output from solving one instance of undirected s-t min-cut to initialize the solve for the next instance.

We compare with the Boykov-Kolmogorov maxflow algorithm (Boykov and Kolmogorov PAMI2004), which is denoted as BK-MAXFLOW in the following. On flow improve, we use a MATLAB implementation of the BK-MAXFLOW, while on 2D MRI, we use its C++ implementation available online.

6. WORK PLAN

Our current and future work plan include the following:

- Efficient implementation on both distributed memory platforms and floating point accelerators.
- Analysis of the connection between the IRLS weight update and the multiplicative weight update used in (Christiano et al. STOC2011).
- Incorporate recent theoretical developments on fast Laplacian solvers (Koutis et al. FOCS2010, Kelner et al. STOC2013) into our parallel linear system solvers.