An Edge-Fencing Strategy for Optimizing SSSP Computations on Large-Scale Graphs

Huashan YU
Peking University
Outline

• Edge-fencing strategy: enabling SSSP algorithms to schedule edge relaxations in a path-centric manner
  • Graph computations are traditionally implemented in a vertex-centric or edge-centric manner

• Skipping edges by scheduling edge relaxations according to lengths of the constructed paths

• The length distribution of a SSSP tree’s shortest paths is correlated with the graph’s degree distribution

• Customizing the schedule for skewed graphs

• Experimental results
SSSP computations

• Mature relaxation: the extended path is a shortest path
  • The relaxations on the edges incident to $v_0$ after the edges incident to $rt$ have been relaxed

• Premature relaxation: the extended path is not a shortest path.
  • The relaxations on the edges incident to $v_1$ before its distance to $rt$ has been updated to be 3

• Inward relaxation: the relaxed edge is not in the SSSP tree
  • The relaxation on the edge between $rt$ and $v_1$

The objective is to reduce inward relaxations and premature relaxations
Scheduling edge relaxations according to lengths of the constructed paths

- The SSSP tree is divided into layers
- The paths with lengths greater than $h_{i+1}$ are not allowed to be extended before every vertex in layer $i$ has been settled
- In the right figure, the blue edge, the red edge, and the yellow edges are relaxed in different steps
  - The blue edge and the yellow edge are relaxed with mature relaxations
  - Premature relaxations may be executed on the red edges

Each $h_i$ is called a fence value
Skipping inward edge relaxations

• Using the push model to relax every edge incident to vertices whose distances are less than $h_0$

• If $i$ is less than $k$, using the push model to relax edges between layer$_i$ and vertices with the distances less than $h_i$

• If $i$ is no less than $k$, using the pull model to relax edges between layer$_i$ and vertices with the distances less than $h_i$
  • The yellow edge is skipped since the two connected vertices have been settled
  • The red edge is skipped since it is too long to be in the shortest path
Characteristics of large-scale graphs

• Skewness in degree distribution
  • Many real networks evolve with the generic mechanism of preferential attachment
  • The edges tends to increase more quickly than the vertices

• Randomness in nature
  • The weights are vertex-independent
  • Every vertex selects its neighbors independently

G=<V E w> is assumed to be undirected, connected and skewed enough, \( w(e) \) denote the edge \( e \)’s weight and is a random value that is uniformly distributed in (0 1)
The correlation between a vertex’s degree and its distance to the source vertex

• The probability that \( x \geq \text{dist}(rt, v_1) - \text{dist}(rt, p(rt)) \) is about \((1 - (1 - x)^{\deg(v_1)/2}) \times (1 - (1/2)^{\deg(v_1)})\)

• \( p(rt) \) is a vertex that both \( \sum\{\deg(u) : \text{dist}(rt, u) < \text{dist}(rt, p(rt))\} \) and \( \sum\{\deg(u) : \text{dist}(rt, u) > \text{dist}(rt, p(rt))\} \) are less than \( m \), which is the number of graph edges

• The probability that \( v_1 \) is a neighbor of \( \{u : \text{dist}(rt, u) \leq \text{dist}(rt, p(rt))\} \) is at least \( 1 - (1/2)^{\deg(v_1)}\)
  • \( \text{dist}(rt, v_1) - \text{dist}(rt, p(rt)) \) is less than \( w(e) \)

• The probability that \( w(e) \) is greater than \( x \): \( 1 - x \), assuming \( x \) is a positive less than 1

Most high-degree vertices have similar distances to the source vertex
A hierarchical graph model

- $rt_0$ denote the vertex that minimizes $\text{average}\{\text{dist}_f(rt_0, u) : u \in V_c\}$, where $V_c$ consists of the graph’s high-degree vertices.

- The distance that minimizes $e^{r_0}/\log(\text{sum}\{\text{deg}(u) : \text{dist}_f(rt_0, u) \leq r_0\})$ is selected as $r_0$.

- $r_i$ ($1 \leq i < s$) is derived from $r_{i-1}$ with $\text{average}\{\text{deg}(u) : \text{dist}_f(rt_0, u) = r_i\} \times \Phi = \text{average}\{\text{deg}(u) : \text{dist}_f(rt_0, u) > r_{i-1}\} \geq 2$.
  - $\Phi$ is a constant less than 1.

Most high-degree vertices are in the top layers.

\[
\begin{align*}
(1) & h_1 = \min\{\text{dist}_f(r, u) : u \in \text{layer}_0\} \\
(2) & h_i = h_1 + r_{i-1}
\end{align*}
\]
Approximating the hierarchical graph model

- Approach I: deriving from the incident edges from some high-degree vertices
  - 4 layers
  - Sorting the vertex’s incident edges according to their weights
  - $r_1$ is $x_i$ that minimizes $e^{x_i}/\log i$, $r_0 = r_1 \times (1 - \Phi)^2$ and $r_2$ is $\min\{1, r_1 \div (1 - \Phi)\}$, where $x_i$ denote weight of the $i_{th}$ edge

- Approach I: deriving from some SSSP tree in the graph
Evaluation

• Experimental environment: a Supermicro SYS-2049U-TR4 server running CentOS Linux release 7.8.2003, 64 cores (4 Intel(R) Xeon(R) Gold 5218 CPUs @ 2.30GHz) and 1536 GB of memory.

• Implementation: as the customized SSSP algorithm of Graph500 3.0
  • To compare with the Δ-stepping algorithm’s implementation in Graph500
  • Exploiting the graph generator in Graph500 to generate weighted R-MAT graphs
  • Two versions were implemented for evaluating our algorithm’s sensitivity to the hierarchical graph model’s accuracy
Evaluation: datasets

• 14 synthetic graphs generated with the Kronecker graph generator in Graph500 3.0
  • Edgefactor = 16 and scale = 26
  • \( k_{x\_y} \) denote the synthetic graph whose R-MAT model is configured with \( a=100x \) and \( b=c=100y \)

• 8 real graphs: selected from the publicly published networks, including social networks, Internets, Web networks and biological networks.
Evaluation: Parallel Performance

- a $3.83 \times - 55.27 \times$ improvement in GTEPS (Billion Edges Traversed Per Second) over the $\Delta$-stepping algorithm’s implementation in Graph500

- the performance is relatively insensitive to the hierarchical graph model’s accuracy
Evaluation: Efficiency

- Edge relaxation intensity (ERI): average number of relaxations on each graph edge

- the edge relaxations are reduced to $0.04\times - 0.44\times$ graph edges
Evaluation: parallelism

- the bulk synchronizations required in parallel settings are $1.55 \times - 4.06 \times$ maximum of edges in one shortest path created during the computation

- Normalized synchronization intensity (NSI): average phases for the SSSP tree’s every hop-level
Thank you