Near-maximum Weighted Independent Set

**Abstract:**

In the big graph, it's too complicated to figure out exactly what the maximum weighted independent set is, so greedy algorithms are often used to deal with this problem and obtain an approximate solution. However, the greedy algorithm takes too much time, and the results are not accurate enough. We solve these problems by using pruning strategy and defining new greedy feature. And we get a more accurate maximum weighted independent set in less time

**Introduction:**

Before finishing this paper, we referred to the research results of others, and benefited a lot. Because it's too complicated to figure out MWIS exactly, we're thinking about near-maximum weighted independent sets(near-MWIS). Greedy algorithms are a great idea, if we don't think about the costs. In the paper 《A note on greedy algorithms for the maximum weighted independent set problem》, the authors have provided us a good greedy algorithm: traverse, calculate

,

and updata this graph. The results of both methods have their own lower bounds.

On the other hand, in the paper《Efficient Weighted Independent Set Computation over Large Graphs》, our teacher Zheng Weiguo provided us two methods of pruning: the single-vertex reduction and the two-vertex reduction. And they provide the exact solution. If we combine greedy algorithms with pruning, we will get the near-MWIS and reduce the process time. As for how to combine greedy algorithms with pruning, we have a good idea. Before the greedy process, we compute WR, if , we use the single-vertex reduction for pruning, and if , we use the two-vertex reduction for pruning. Otherwise, we don’t prune. And in the greedy process, we compute WR、GR and to determine that which node will be put into the MWIS. And the results showed that NR works better.

**Previous work:**

Let be a weighted undirected graph without loops and multiple edges, where V is the set of vertices, E is the set of edges, and W is the vertex weighting function and for any nonempty set . the degree of vertex , the neighborhood of .

According to the greedy algorithm in the paper《A note on greedy algorithms for the maximum weighted independent set problem》. We compute WR or GR to determine that which node will put into the MWIS. On the other hand, according to two reduction rules from the paper《Efficient Weighted Independent Set Computation over Large Graphs》. We know that what is the single-vertex reduction and what is the two-vertex reduction.

The single-vertex reduction: the node can be extracted into the MWIS if the weight of a node is greater than the sum of the weights of its neighbors.

The two-vertex reduction: if it is the fact that the weights of two nodes are less than the sum of the weights of their neighbors, but the sum of the weights of two nodes is greater than the sum of the weights of their neighbors, we can know that two points must have the same neighbor, that is, the distance between them is 2-hop.

By the way, the nodes extracted by both types of reduction are accurate and we can extract many nodes in the same process. It saves a lot of time.

**Our work:**

How to combine greedy algorithms with pruning? We used the greedy algorithm to get the near-MWIS. In the process, we find that

is associated with the pruning process.

In the process of the single-vertex reduction, we know that

.

So we can get that WR>0.5. And if we know that WR>0.5, we can get the fact that the node must be put into the MWIS by the single-vertex reduction.

If WR<=0.5, in the process of the two-vertex reduction, we know that

}

So we can get the fact that WR of a or WR of b must greater than 0.33. When we know that WR >0.33, we can assume that this node is probably put into the MWIS by the two-vertex reduction.

So before the greedy process, we compute WR, if , we use the single-vertex reduction for pruning, and if , we use the two-vertex reduction for pruning. Otherwise, we don’t prune.

As for the greedy process, we can choose which node can be put into the MWIS by WR 、GR or NR(=2\*WR\*GR) . We repeat the process until all nodes are exhausted

**while** G contains vertexes **do**

compute；

**if** WR>0.5

apply single-vertex reduction on v；

update；

**elseif** WR>0.33

apply two-vertex reduction on v；

update；

**else**

Choose a vertex, say vi, in Gi;

;

;

Support:

For a big graph, accurate for MWIS is too complicated, so we requested a near-MWIS by using the greedy algorithm. Because of WR, in the greedy process, we can find it is associated with the pruning process. In addition, a traverse after pruning can extract a number of points, therefore, It can not only improve the weight of the near-MWIS, but also reduce the process time.

**Conclusion:**

We experimented with the following graphs:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Graph | |V| | |E| | Avg.Deg | Max.Deg |
| G1 | GD98\_c | 112 | 168 | 3 | 3 |
| G2 | astro-ph\_s | 1298 | 3229 | 4.99 | 58 |
| G3 | add20 | 2395 | 5378 | 4.49 | 83 |

And the experimental results are shown in figure: