Data Mining Homework - 5

K-way Graph Partitioning Using JaBeJa

Group – 7

Xing Zeng <xingzeng@kth.se> Sevket Melih Zenciroglu <smzen@kth.se>

**1. Task1**  
In task1, we implement the JabeJa algorithm in paper[1]. In method SampleAndSwap, we use hybrid heuristic for node selection and swap the node if the partner returned by method FindPartner is not null. Method FindPartner shows how the partner is selected.  
  
To avoid becoming stuck in a local optimum, in task1, we use simulated Annealing (SA) technique] which introduces a temperature (T) and decreases it over time linearly as shown saCoolDown() method.

Text

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

The result of task1 is as follows with default values provided,

Delta=0.003, T=2, round=1000

Chart

Description automatically generated

**2. Task2**

**2.1. Implement a Different Simulated Mechanism**

Task2, we tweak different JaBeJa to analyze how changing the simulated annealing parameters and the acceptance probability function affects the performance of Ja-Be-Ja. (findPartner function – else part)

* Change parameters  
  We set T = 1, and try different delta between 0.8 and 0.99. And we use acceptance probability functions in [2].

Below graphs show alpha=0.8 and 0.95

(T = T\*delta) (in the code, to avoid the confusion between the delta used in the first task and this implementation of annealing, we used it as   
T \*= T\_SA\_alpha instead of T = T\*delta)

|  |
| --- |
| Chart  Description automatically generated  T = 1, alpha = 0.8 |
| Chart  Description automatically generated  T = 2, alpha=0.95 |

According to the results, this method converges faster than the first simulated-annealing.

**2.2.1. Change Parameters**

Different values used for Round. When the number of rounds increased, the number of swaps increased, too and edge-cuts decreased.

(Rounds\_Converged shows at which round the edge-cut had its minimum value for this execution for the first time. While low number of rounds is used, it’s not converging, meaning that, it doesn’t have enough time to converge.)

**Chart, line chart

Description automatically generated**

Different values used for Delta. When the delta is small, the convergence happening late, for bigger values of delta, it converges faster.

(Rounds\_Converged shows at which round the algorithm converged for the given parameters.)

**Chart, line chart

Description automatically generated**

**2.2.2. Restart Simulated-Annealing**

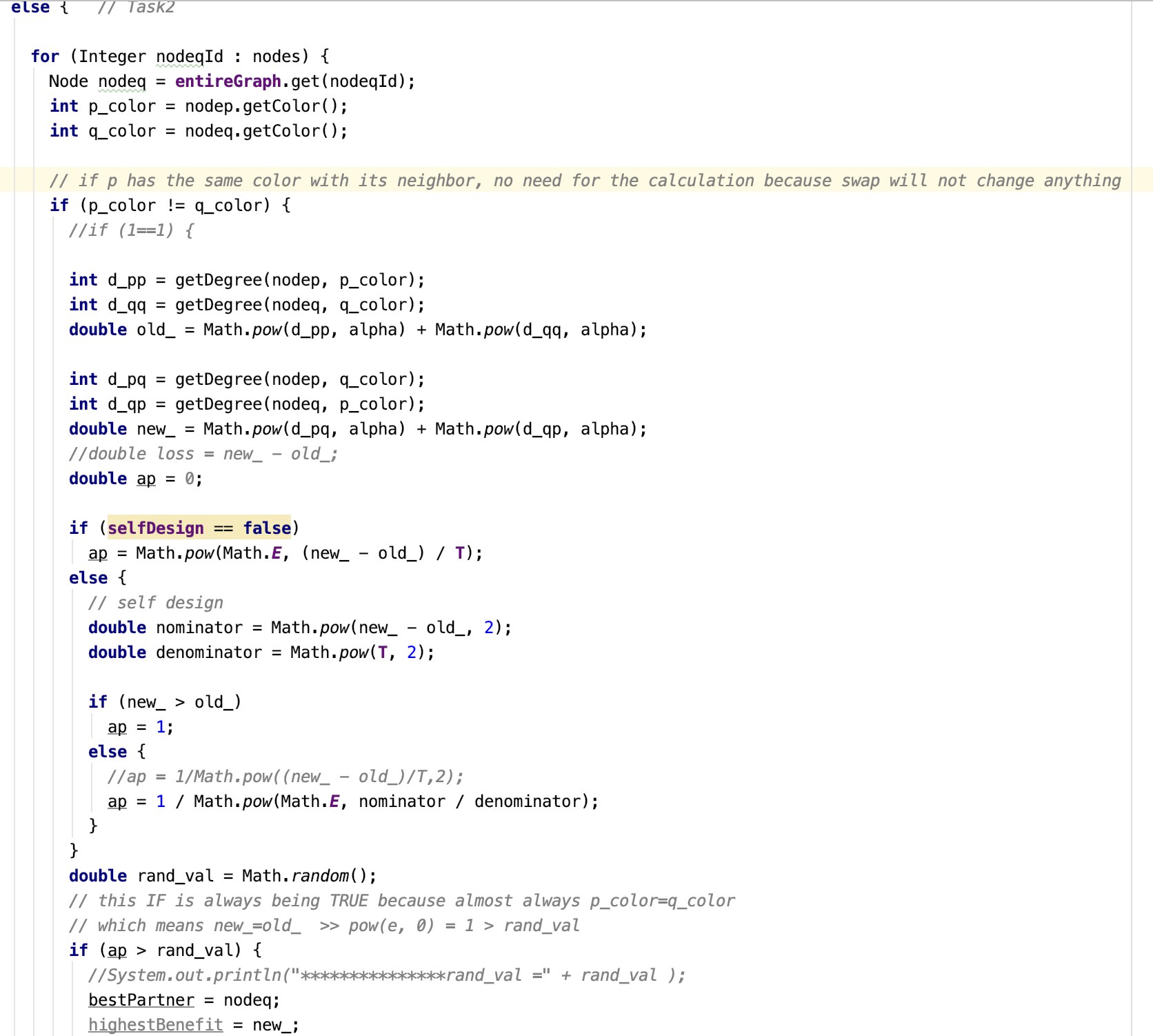
In the below graph “5 restarts” applied to the Twitter graph. Parameters used are:

T=2, Round=400, delta=0.01

As it can be seen from the graph, once the T was reset to 2 after 400 rounds, there’s a small peak and then it converges again. Edge-cut doesn’t change much (peak can be ignored, especially in small graphs, it is not that visible) but swaps keep adding following each T restart after a long convergence period.

Chart

Description automatically generated



**3. Bonus – Define Own Acceptance Probability Function**

* Change the acceptance probability

As shown in annotation in above screenshot (line 127), we define another acceptance probability as

The result of T=1, Delta = 0.8 and 0.95 are as follows,

(Delta = [T\_SA\_alpha](#T_SA_alpha) in the code)

Chart, line chart

Description automatically generated

T = 1, Delta = 0.8

Chart

Description automatically generated

T = 1, Delta = 0.95

**4. Analysis**

From the above result screenshots, we can notice that

1. The performance of simulated annealing approach in Task2 is better than Task1 since it converges more quickly, and the cutting edge is 2141 less than 2604 in task1.
2. From task2, once no more bad swaps are allowed, then Ja-Be-Ja converges to an edge cut rapidly and the edge cut does not change over time. And when T is set initially equals 1, the smaller the delta is, the less the swapping is and the quicker the converge rate is. While if the delta is larger, the cutting edge is less.
3. Number of swaps decreases while the delta value is increasing. All 4 graphs (3elt, add20, Twitter, Facebook) have similar characters in that manner, decrease rate looks similar.
4. Number of edge-cuts increases while the delta value is increasing. However, the bigger the graph, the smaller the variance it seems. Facebook and Twitter graphs don’t have dramatic changes when delta changes compared to other 2 smaller graph, especially 3elt.
5. Since the Facebook graph is too big, it was taking too long to finish the calculations when we used all the nodes in the graph. We thought of a way to reduce the time while maintaining a similar result. That’s why, instead of using all nodes, we tried to use some fraction of them. In our experiments, 20% of random data (randomly picked nodes for the algorithm from the entire graph) gave us a faster result with similar edge cuts. It was not asked in the assignment, but we wanted to mention it since it sounds like a good finding.

**5. Reference**

[1] F. Rahimian, A. H. Payberah, S. Girdzijauskas, M. Jelasity and S. Haridi, JA-BE-JA: A Distributed Algorithm for Balanced Graph PartitioningPreview the document, SASO2013, pp. 51-60.

[2] <http://katrinaeg.com/simulated-annealing.html>