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

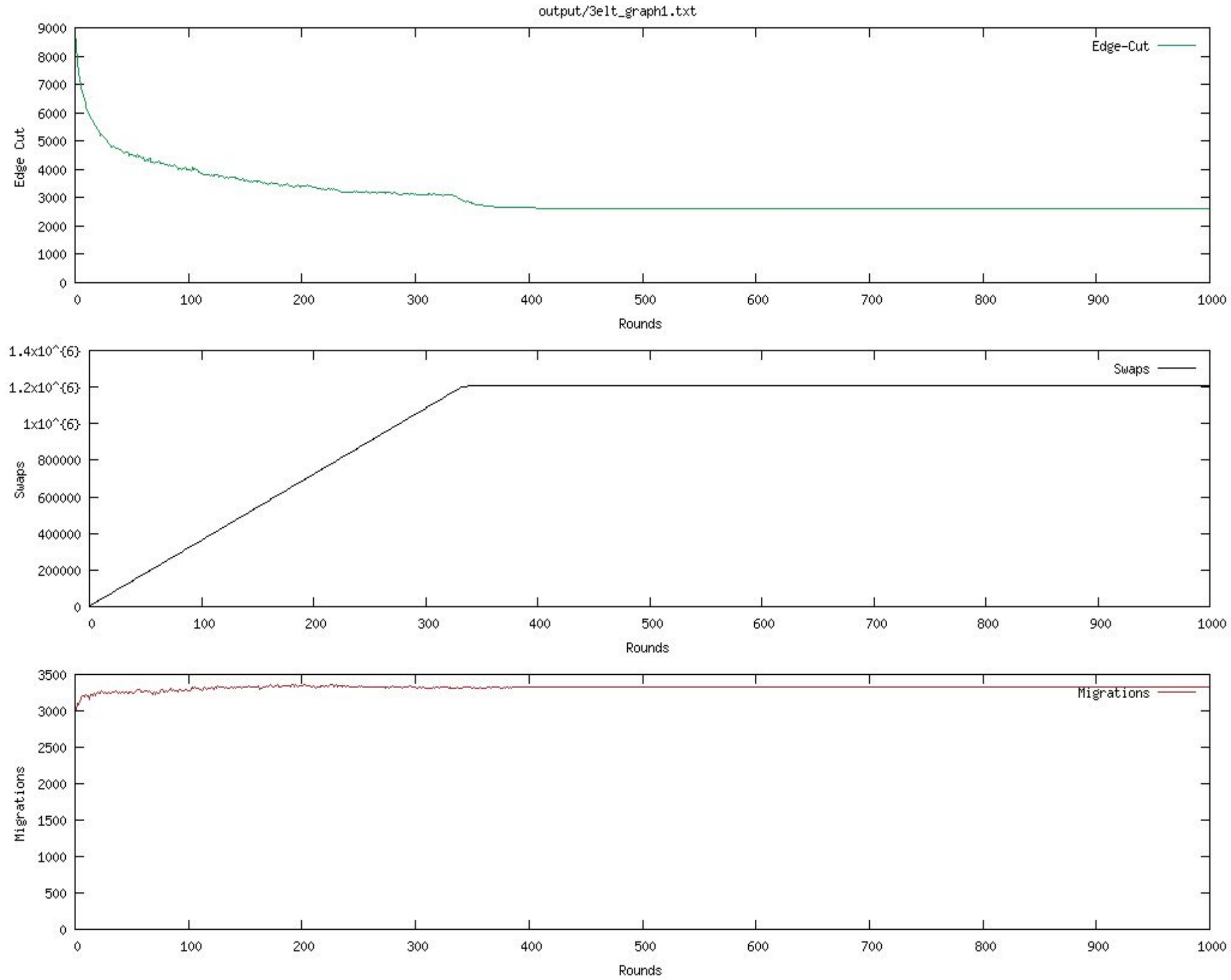
In task1, we implement the JabeJa algorithm in paper[1]. In method SampleAndSwap, we use hybrid heuristic for node selection and swap the node if the partner returned by method FindPartner is not null. Method FindPartner shows how the partner is selected.

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.

```java
public Node findPartner(int nodeId, Integer[] nodes){

  Node nodep = entireGraph.get(nodeId); //HashMap<Integer, Node>

  Node bestPartner = null;
  double highestBenefit = 0;
  double alpha = config.getAlpha();

  // TODO
  for(Integer nodeqId : nodes){
    Node nodeq = entireGraph.get(nodeqId);
    int d_pp = getDegree(nodep, nodep.getColor());
    int d_qq = getDegree(nodeq, nodeq.getColor());
    double old_ = Math.pow(d_pp, alpha) + Math.pow(d_qq, alpha);
    int d_pq = getDegree(nodep, nodeq.getColor());
    int d_qp = getDegree(nodeq, nodep.getColor());
    double new_ = Math.pow(d_pq, alpha) + Math.pow(d_qp, alpha);
    if(new_ * T  > old_ && new_ > highestBenefit){
      bestPartner = nodeq;
      highestBenefit = new_;
    }
  }
  return bestPartner;
}
```

```java
/**
 * Simulated analealing cooling function
 */
private void saCoolDown(){
  // TODO for second task
  if (T > 1)
    T -= config.getDelta();
  if (T < 1)
    T = 1;
}
```

The result of task1 is as follows,



2. Task2

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.

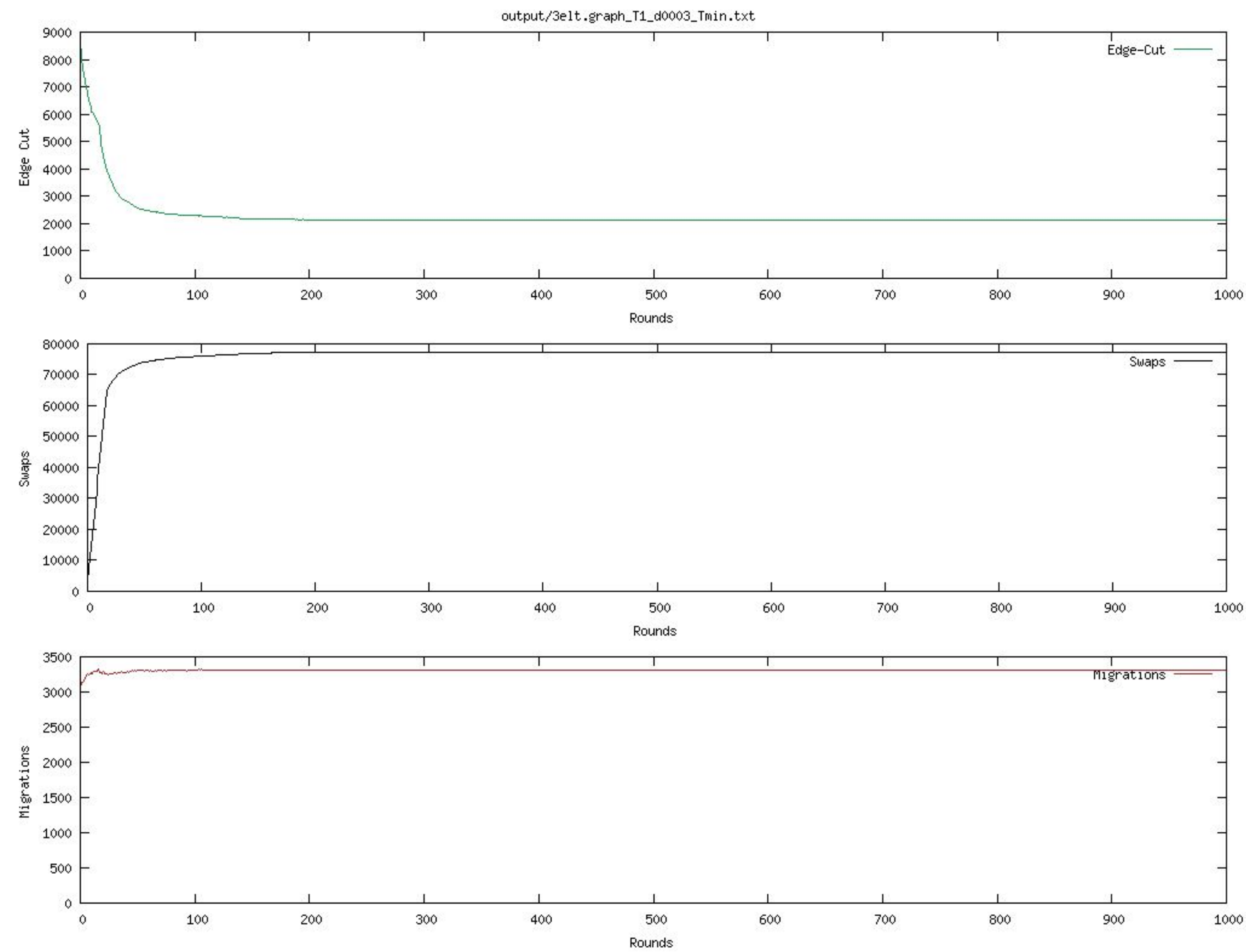
- Change parameters

We set T = 1, and try different delta 0.003 and 0.9. And we use acceptance probability functions $e^{\frac{new-old}{T}}$ in [2].

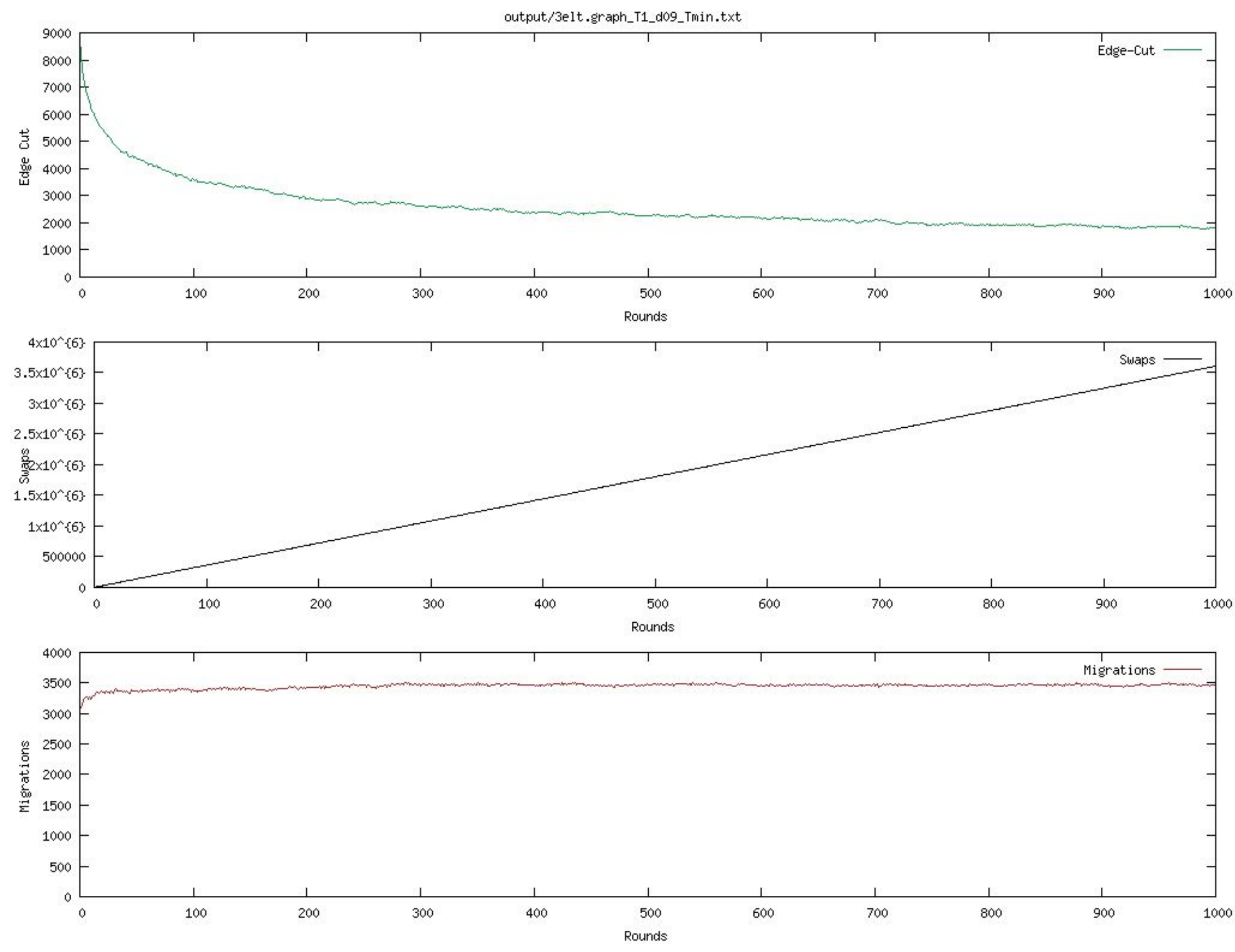
```java
/**
 * Simulated analealing cooling function
 */
private void saCoolDown(){
  // TODO for second task
  // if (T > 1)
  //    T -= config.getDelta();
  // if (T < 1)
  //    T = 1;
  // T *= 0.9;
  T *= config.getDelta();
}
```

```java
    // TODO
    for(Integer nodeqId : nodes){
      Node nodeq = entireGraph.get(nodeqId);
      int d_pp = getDegree(nodep, nodep.getColor());
      int d_qq = getDegree(nodeq, nodeq.getColor());
      double old_ = Math.pow(d_pp, alpha) + Math.pow(d_qq, alpha);
      int d_pq = getDegree(nodep, nodeq.getColor());
      int d_qp = getDegree(nodeq, nodep.getColor());
      double new_ = Math.pow(d_pq, alpha) + Math.pow(d_qp, alpha);
      double loss = new_ - old_;
      double ap = 0;
      // if new_solution is better than old one
      if(loss > highestBenefit){
        ap = 1;
      }
      else{
        //ap = Math.exp((newBenefit- highestBenefit)/T);
        //acceptance probability in paper
        ap = Math.pow(Math.E,(loss-highestBenefit)/T);

        // self design
        // ap = 1/Math.pow((loss-highestBenefit)/T,2);
      }
      if(ap > Math.random()){
        bestPartner = nodeq;
        highestBenefit = loss;
      }
    }
    return bestPartner;
  }
```

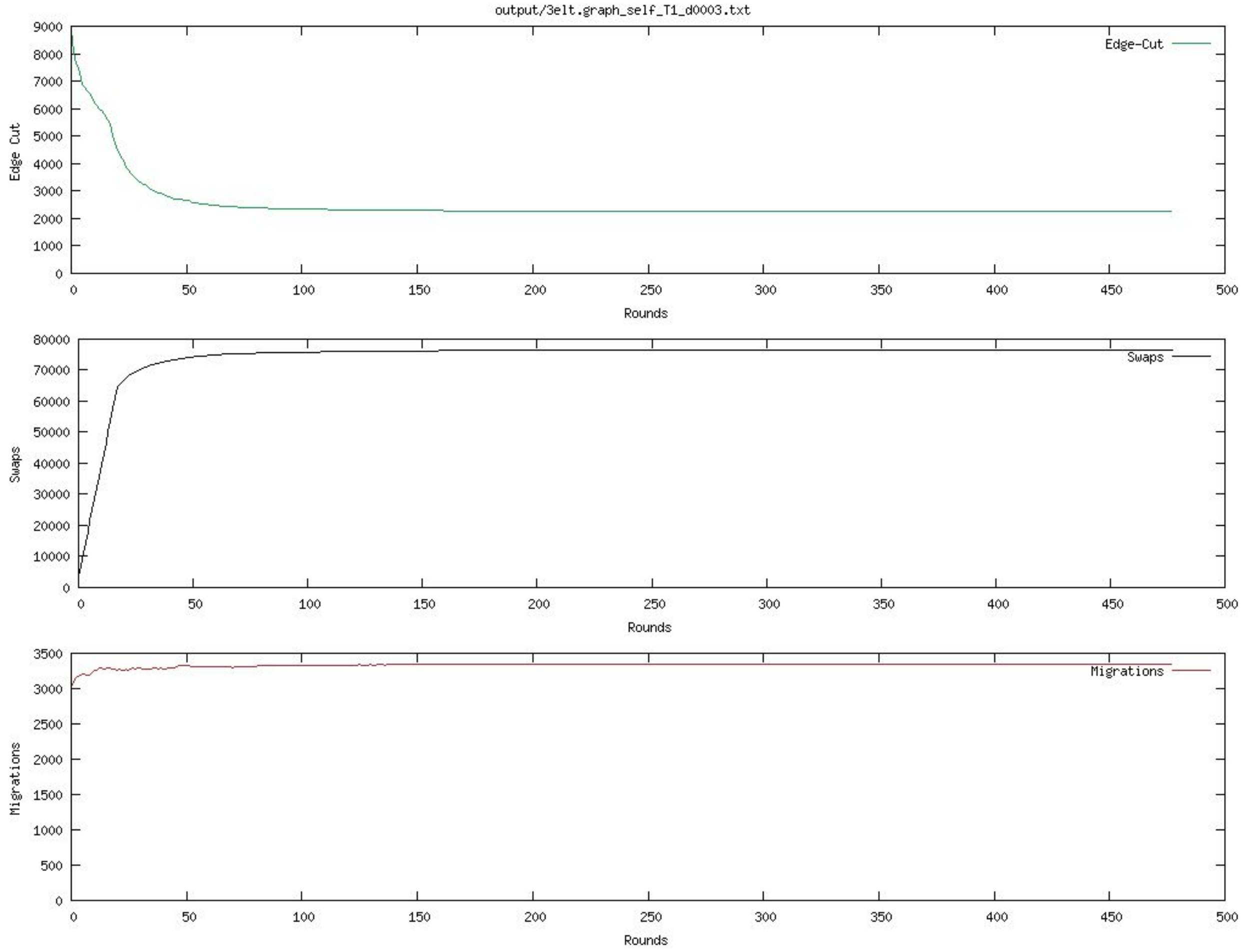
The result of T=1, Delta = 0.003 is as follows,



The result of T=1, Delta = 0.9



- Change the acceptance probability
  As shown in annotation in above screenshot (line 127), we define another acceptance probability as $\frac{1}{e^{\frac{(new-old)^2}{T^2}}}$
  The result of T=1, Delta = 0.003 is as follows,



3. 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.

4. 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