15-210 Assignment 06

Karan Sikka
ksikka@cmu.edu
Section C
October 31, 2012

1: Task 3.1

1. Convert edge sequence to adjacency table (vertex →vertex set). Call this graph G.
2. Let X be a set containing vertexes which are independent. X starts as the empty set.
3. Assign random numbers to the vertexes by using a hash function h.
4. Select all vertexes which have the highest $h(v)$ compared to their neighbors. Call them "localMaxVertexes".
5. The localMaxVertexes are guaranteed to form an independent set, since no more than one localMaxVertex exists in a set of adjacent vertexes. Add these independent vertexes to X.
6. Find the subgraph of the current graph which doesn't contain any localMaxVertexes, or any of their neighbors. Call this G'. Note that G' is independent from X.
7. Repeat from step 3, with G as G', until G' contains no vertexes.
8. When G' contains no vertexes, return X, which shall be a maximal independent set.

2: Task 3.4

$$p_v = \frac{1}{1 + \deg(v)}$$

3: Task 3.5

Since a vertex has probability $\frac{1}{1+\deg(v)}$ of being added to the MIS, it has probability $\frac{\deg(v)}{1+\deg(v)}$ of not being added.

Let X_v is be an indicator random variable, which is 1 if v is not being added, and 0 if added. Then $\mathbb{E}[X_v] = \frac{\deg(v)}{1+\deg(v)}$.

Let X be a random variable which is the number of vertices added to the MIS in the graph after one step of the algorithm. We see that

$$X = \sum_{v \in V} X_v$$

Therefore, by linearity of expectation,

$$\mathbb{E}[X] = \sum_{v \in V} \mathbb{E}[X_v] \leq |V| \frac{\Delta}{1 + \Delta}$$

Since delta is non-negative, we see that the delta term is some fraction in $[0, 1)$. Call the fractional delta term $\frac{1}{k}$ so that the expected number of vertexes added to the MIS after one step of the algo is $\frac{|V|}{k}$.

We know that in our current algorithm that all vertices added to the MIS are removed from the frontier/input of the recursive call. Therefore, in expectation, at most $\frac{|V|}{k}$ vertices are retained after 1 step of the algorithm.

$$W_{\text{MIS}}(|V|) \leq W(\frac{|V|}{k}) + c_1|V| + c_2$$

Since the input size decreases by a constant every iteration, we know that the work is root dominated, and will have $W \in O(|V|)$.

Further, more rigorous proof can be found using the tree method, which shows that at each level, the work at each level is $c_1 k^{-i}|V| + c_2$ at level i, and there are $\log_k(|V|)$ of levels in the tree. If we sum up the work in the tree, we get the summation,

$$W(|V|) \leq \sum_{i=0}^{log_k(|V|)} c_1 k^{-i}|V| + c_2$$

$$W(|V|) \leq c_2 \log_k(|V|) + c_1|V| \sum_{i=0}^{log_k(|V|)} k^{-i}$$

The summation evaluates to some constant, since $k \in [0, 1)$ and this is the sum of a geometric series. Therfore,

$$W(|V|) \leq c_2 \log_k(|V|) + c_1 c_3 |V|$$

$$W(|V|) \in O(|V|)$$

As for span, we know that the expected number of iterations of the algorithm is in $O(\log(|V|)$. The span of each step is at most $O(\log(|V|)$, because all operations are filter/map/reduce (with const. time functions), and all input sizes are $O(|V|)$. `isLocalMax` is constant time because it does work proportional to the degree of the vertex, which in this case is upper-bounded by a constant.

$$S(\log^2(|V|))$$

4: Task 4.1

We'll create a graph with the following properties. The exams will be vertices. Two exams will be connected by an edge if there exists a student who is taking both exams. Solving the graph coloring problem on this graph solves the exam scheduling problem, because connected graphs must have different colors, representing that if a student is taking 2 exams, they must occur at different times. The minimum number of colors necessary is the minimum number of time slots necessary.

Let n be the number of students and m be the number of exams. Say you are given a sequence of exam sequences. The index into the outer sequence represents the number of the student in the ordering, and the inner sequence is a list of exams which the student is taking.

We will build an adjacency matrix for the graph. For each student, for each exam the student is taking, "connect" the exam with all the other exams the student is taking by updating the adjacency matrix in constant work for each exam-exam pair. In the worst case, each student is taking every exam. In this case, the work of this algorithm would be

$$O(nm^2)$$

.

5: Task 4.5

Let n be the number of students, and m be the number of exams. The work of the function is

$$O(nm^2 \log(nm))$$

because it does what is described in the reduction for 4.1 but it has an additional log factor for the overhead of searching a BST for mapping string to ints and vice versa. This overshadows the work of MIS itself.

The span is around $O(\log^3(nm^2))$