COM3105 Advanced Algorithms

Homework 2

Joffray Hargreaves

Question 1

(5 points)

Given a simple graph $G = (V, E)$ (no loops or multiple edges), consider the following deterministic algorithm:

- start with an arbitrary partition.
- If moving a vertex from one part to the other increases the number of crossing edges, we will move it.
- We do this until there is no such vertex left.

Why does this algorithm terminate?
What can you say about the number of edges crossing when the algorithm terminates?

Answer

Termination of the Algorithm:

- Let C be the set of edges crossing the partition.
- Let state i be the state of the partition after i moves have been made.
- For each state i, we iteratively check a vertex v_j (where $0 \leq j < |V|$) to see if moving it increases $|C|$.
- If moving a vertex v_j increases $|C|$ we make this move and move to state $i + 1$ and restart the same process.
- Checking each state i takes $\mathcal{O}(|V|)$ steps, where V is finite.
- As V is finite, for a simple graph, E is also finite, so the maximum $|C|$ is $|E|$.
- Therefore, in the worst case:
  - at state 0,
    - $|C| = 0$ (no edges in partition)
    - Adding $v_{|V|-1}$, increases $|C|$ by 1.
  - at state 1,
    - $|C| = 1$
    - Adding $v_{|V|-1}$, increases $|C|$ by 1.
  - at state i,
    - $|C| =$ i
    - Adding $v_{|V|-1}$, increases $|C|$ by 1.

- at state $|E|$,
  - $|C| = |E|$
  - No movement of vertex can increase $|C|$, as $|C| = |E|$.
  - Algorithm stops as no more edges to add.
- So $|E|$ edges being added to C and taking $|V|$ checks at each state.
- Hence the runtime is $\mathcal{O}(|E| * |V|)$.
- Both E and V are finite sets, so the algorithm terminates

Number of Crossing Edges at Termination.

Let the final partition of G be (A, B). For each vertex $v \in V$:

$$d_A(v) = |\{\, u \in A : (u, v) \in E \,\}|, \quad d_B(v) = |\{\, u \in B : (u, v) \in E \,\}|.$$

At termination, moving any vertex to the other part does not increase the number of crossing edges. Hence:

$$d_B(v) \geq d_A(v) \quad \text{for all } v \in A, \qquad \text{and} \qquad d_A(v) \geq d_B(v) \quad \text{for all } v \in B.$$

This means that for every vertex $v \in V$, at least half of its incident edges cross the partition:

$$d_C(v) \geq \tfrac{1}{2} d(v),$$

where,
$d(v) = d_A(v) + d_B(v)$ (i.e. the total degree of v),
$d_C(v) = \textit{number of degrees from } v \textit{ crossing the partition.}$

Summing over all vertices gives:

$$\sum_{v \in V} d_C(v) \geq \frac{1}{2} \sum_{v \in V} d(v)$$

Since each crossing edge contributes to $d_C(v)$ for exactly two vertices,

$$\sum_{v \in V} d_C(v) = 2|C|$$

Combining these two expressions:

$$2|C| \geq \frac{1}{2} \sum_{v \in V} d(v) = \frac{1}{2}(2|E|) = |E|$$

Therefore,

$$|C| \geq \frac{|E|}{2}$$

Conclusion. When the algorithm terminates, the resulting partition has at least half of all edges crossing between the two parts. Although this partition may be only a local maximum, it guarantees that at least $\frac{|E|}{2}$ edges are cut.

Question 2 *(5 points)*

(MAX-SAT). Given a 3-CNF F where every clause has width exactly 3, let $OPT(F)$ be the maximum number of clauses in F that can be satisfied simultaneously, i.e., it is the maximum integer m such that there is an assignment to the variables that satisfies m clauses.
Give a randomised algorithm that produces an assignment that satisfies at least $\frac{7}{8}OPT(F)$ clauses on expectation.
You need to state the algorithm and give an analysis.

Answer

Algorithm:

1. For each variable x_i in F, assign it a value of 1 with probability $\frac{1}{2}$, and 0 with probability $\frac{1}{2}$.
2. Evaluate each clause in F. If at least one literal in the clause is satisfied by the current assignment, count the clause as satisfied.
3. Return the current assignment and the count of satisfied clauses.

Analysis:
Let C be the set of all clauses in F.
Let c be any clause in C.
The probability that any clause c is satisfied using the random algorithm:

- Each clause has exactly 3 literals.
- The probability that a specific literal is true is $\frac{1}{2}$.
- For the clause to be unsatisfied, all 3 literals must be false.
- The probability that the clause is unsatisfied is $\left(\frac{1}{2}\right)^3 = \frac{1}{8}$.
- Therefore, the probability that the clause is satisfied is $1 - \frac{1}{8} = \frac{7}{8}$.

Let X be the random variable representing the number of satisfied clauses in F under the random assignment.
Using the linearity of expectation:

$$\mathbb{E}[X] = \sum_{c \in C} \mathbb{E}[X_c]$$

Where X_c is a random variable indicating whether clause c is satisfied.
Expectation for a clause c in C:

$$\mathbb{E}[X_c] = \sum_{x \in \{0,1\}} x \cdot P(X_c = x)$$
$$\mathbb{E}[X_c] = 1 \cdot P(c \text{ is satisfied}) + 0 \cdot P(c \text{ is not satisfied})$$
$$\mathbb{E}[X_c] = P(c \text{ is satisfied}) = \frac{7}{8} \qquad \textit{(from above)}$$

Using this in the expectation of X:

$$\mathbb{E}[X] = \sum_{c \in C} \mathbb{E}[X_c]$$
$$\mathbb{E}[X] = \sum_{c \in C} \frac{7}{8}$$
$$\mathbb{E}[X] = |C| \cdot \frac{7}{8}$$

This means that on average, the random assignment satisfies $\frac{7}{8}$ of all clauses in F.
Let $OPT(F)$ be the maximum number of clauses that can be satisfied simultaneously in F:

$$|C| \geq OPT(F)$$

Therefore:

$$\mathbb{E}[X] = |C| \cdot \frac{7}{8} \geq OPT(F) \cdot \frac{7}{8}$$
$$\mathbb{E}[X] \geq \frac{7}{8} OPT(F)$$

Conclusion. The algorithm provided produces an assignment that satisfies at least $\frac{7}{8}OPT(F)$ clauses on expectation.