COMP6730 Homework 3

Chris Winsor

11/7/2021

# Problem 1a:

*Consider an instance of SAT with m clauses where every clause has exactly k literals. Prove there exists an assignment to the variables s.t. it satisfies at least clauses.*

Given a set of clauses let be the number of literals in the th clause.

If we choose the value of each literal randomly the probability of a clause being satisfied is .

The expected number of satisfied clauses is

If we have

Since the probability of satisfying clauses is strictly greater than zero, we conclude there exists at least one such assignment.

# Problem 1b:

*Give a derandomization of a simple randomized algorithm using conditional expectations and show it achieves the lower bound.*

Suppose we have established the first k clauses, with the remaining clauses established randomly. An algorithm that is locally greedy in choosing clause is expressed as:

# Problem 2a:

*Prove that for every integer n there exists a coloring of the edges of the complete graph by two colors s.t. the total monochromatic copies of is at most .*

Let be a complete graph.

There are edges and colorings.

If we color each edge randomly the probability of a particular coloring is .

Consider all 4-vertex cliques of . There are such cliques.

For each let be the event that clique is monochromatic.

Given that the first edge can be either color but the remaining edges must follow, we have

The Union Bound says:

and from that

Since the probability of choosing a 4-node clique that is monochromatic is strictly greater than zero, we conclude there exists at least one 4-node clique that is monochromatic.

# Problem 2b:

*Give a randomized algorithm ... that runs in time polynomial in n.*

To achieve polynomial time construction, we first need a polynomial time sampling algorithm. In this case we simply randomly choose color for each node in .

We also need to establish the number of samples. If p is the probability of generating a sample that satisfies our 4-vertex clique requirement, the number of samples required is a geometric random variable the expectation of which is . Thus we need in polynomial time. The number of 4-cliques is and probability that each is monochromatic is thus:

# Problem 2c:

*Show how to construct... using conditional expectations.*

(Reference Section 6.3 in Mitzenmacher/Upfal)

Imagine we have cliques If we randomly assign vertices to cliques then the average clique size is and the expected clique size is denoted:

If instead we choose to deterministically assign the first vertices, with the remaining assigned randomly then the expected clique size of such assignment is:

We wish to establish an algorithm such that our deterministic assignments result in expected clique size no larger than that of random assignment. In other words:

Consider a single assignment and place it randomly. We know:

(above is just summing across the conditioned variable)

It follows that

In other words - if we assign to the minimum sized clique then the resulting expected clique size will be no greater than it was before.

Above gives us a deterministic way to assign vertices such that expected clique size no larger than that of random assignment. Specifically, our algorithm is to assign the node to the clique having fewest vertices.

# Problem 3a

*Show that each is an independent set in*

We use induction: Given a permutation of the vertices, and subset of vertices that are an independent set we shall call "CurrentSet"

If is not a neighbor of any node in then

is an independent set.

The basis: , the set consisting of just the first node in the permutation, is an independent set.

# Problem 3b

*Suggest a natural randomized algorithm to product for which ... the expected cardinality of where is the degree of vertex .*

Our goal is an algorithm to produce (permutation on ) s.t. meets the criterion above.

We use Sample and Modify (Mitzenmacher pg. 142)

The first step is to select with probability elements from to form a permutation . This permutation does not meet our criterion but has the characteristic:

Above is considering the selection process as a geometric series w/ probability p and thus the expected number of elements .

The second step is to sequentially consider each element in and remove it if there is an element where . Another way to view this is by creating a new and only including elements from if . At each step the probability of including is the degree of the node over the number of nodes . Thus, for a particular node being considered the increase in the expected number nodes is:

and over all nodes in

When the algorithm terminates, we are left with having cardinality

We have thus shown a randomized algorithm to generate a permutation on having an expected cardinality that is dependent only on number of nodes and degree of each node.

# Problem 4a

*Give a winning strategy for chooser when*

On round 1 the chooser sets and . Tokens not in A or B are left behind.

For round 2 and beyond the chooser is left with tokens which they will evenly split. Thus, at round there are token (a winning strategy).

# Problem 4b

*Use probabilistic method to show winning strategy exists for remover when .*

If tokens are randomly chosen for removal, the probability of a token remaining at round r is , so the expected number of tokens at round r is

and where r = n:

Since the expected value is strictly less than one there must exist a scenario where the remover wins.

# Problem 4c:

*Explain how to use conditional expectations to derandomize the winning strategy for remover when .*

If is the number of tokens removed on round , the expected number of rounds is denoted

If we consider as given an initial set of rounds, with the remaining rounds random then we have

Given the constraint on the chooser that , where is to be left behind and the remover may remove A or B, then we conclude at least one of A or B must be less than . If the policy of the remover is to remove the larger of A or B then

In other words - if the remover removes the larger of A or B then the expected number of rounds does not increase.

# Problem 5a

*Prove that in any network of edges there is a way to assign the signs of the edges s.t. the number of "+" edges is no more than .*

Pick an arbitrary node and perform BFS. The depth of the resulting tree cannot exceed the diameter of the network, and . Thus, if we assign (+) to edges represented transitions in the BFS tree that go from even to odd levels, and (-) to edges represented as odd-to-even level transitions of the BFS tree then there will be no more than positive edges.

# Problem 5b

An algorithm to assign (+) and (-) to edges follows naturally from above. Starting with arbitrary node assign (+) to each edge. Performing BFS from that point we alternate between (-) and (+) to edges discovered at each level of the BFS. The algorithm achieves (+) edges no more than .