

Algorithms Seminar: The Twitter Problem

Notes by Sam McCauley

September 2, 2011

1 Introduction

Twitter can be queried by companies to receive a list of recent tweets about a topic. However, these companies can only make a limited number of queries per day, they receive a limited number of tweets per query, and they wish to maximize the amount of information they obtain. Very popular topics may always return too many tweets to be included in the single list, such as 'Obama', and less-discussed topics may return fewer tweets than the maximum, such as 'Skiena'. Queries can be of a single topic, or of multiple topics. For example, 'Obama Skiena' which would return topics relevant to Obama as well as topics relevant to Skiena. For this problem we assume all topics are independent, so there are no tweets related to both Obama and Skiena.

2 General Problem Statement

Assume that a company can only make q queries per day. These queries must each be a subset of the topics T_1, T_2, \dots, T_n the company is interested in learning about. Each query can give at most k results. Each result is related to exactly one of the topics in the query.

There are two distinct models for Twitter's behavior as well: either Twitter's results are random (and therefore a tweet can repeat in a subsequent query), or they are chosen from the tweets that have not been returned yet.

There must be some sort of utility function to determine the effectiveness of the queries. This is given by $U(x, i)$, where x is the number of tweets in that topic and i is the index of the topic queried. This can be simplified to $U(x)$ when all topics are equally valued to the company.

In its simplest form, $U(x) = c$ for some constant c and all x —in other words, we simply want to maximize the number of unique tweets regardless of topic. Another possibility is a decreasing $U(x)$, which encourages topic diversity since

multiple tweets from the same topic give decreasing return. A third possibility expands on this, and lets $U(1) = 1, U(x > 1) = 0$. This case maximizes the number of topics returned; multiple tweets on one topic are no more valuable than one tweet on the topic.

It is assumed that there is some sense of an expected value for the number of tweets that could be returned for a given topic. To continue the previous example, a topic like 'Obama' will have a much larger expected value than a topic like 'Skiena'.

Goal: What series of queries maximizes the expected total utility from all results?

Solution 1: Start with the lowest-index query and combine it with subsequent queries until the expected number of tweets returned is at least k .

This method ensures that k tweets will most likely be returned on each query, for a total of qk tweets, which is the maximum possible. Assuming an accurate expectation for the number of tweets returned, no repeats, and utility function $U(x) = c$ (only total number of tweets matters), this is optimal.

Solution 2: Make a large number of queries, cross off all queries that were returned, query again.

This method works well for utility function $U(1) = 1, U(x > 1) = 0$, where maximizing the number of topics returned is crucial.

Question: How does the value of n relate to how many topics you can get?

Question: More specifically, if $q \leq n \leq qk$, is it possible to get exactly 1 tweet from each topic?

3 Combinatorial Problem Statement

Assume there is an urn with N different colors of balls in it. There are n_1 balls of color 1, n_2 balls of color 2, etc, where $n_1 \geq n_2 \geq \dots \geq n_N$. We want to pick a subset of $\{1, 2, \dots, N\}$ to maximize the expected number of colors returned in a k -sample.

This is very similar to our original problem with $q = 1$ and the utility function $U(1) = 1, U(x > 1) = 0$.

Goal: To maximize the expected number of colors returned in a k -sample

Question: Can we prove that the optimal solution is always an interval $[i, j]$?

This may be solveable with an exchange argument.

Question: If we can prove the above we know the complexity is polynomial; if not can we prove hardness?