Unconstrained Submodular Maximization in MapReduce

Sikander Randhawa, Ben Chugg, Angad Kalra

The university of British Columbia
Vancouver, Canada

Background

Formally, a set function $f : 2^X \to \mathbb{R}$ is *submodular* iff

$$f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B),$$

for all $A \subset B$.

Intuitively, think of decreasing marginal gains: The more you have, the less you appreciate gaining something new.

These functions find applications in graph theory, game theory and machine learning.

Background

We are interested in maximizing such functions, i.e.,

$$\max_{S \subset X} f(S).$$

There are no constraints on the subsets over which we can maximize and the functions are non-monotone — *unconstrained, non-monotone, submodular maximization (USM)*.

In general, this problem is NP-Hard so we search for approximations.

Local Search

Local Search (LS) achieves a $(1/3 - \frac{\epsilon}{n^2})$-approximation in a centralized setting. The idea: Search for elements that increase the marginal by a factor of $(1 + \frac{\epsilon}{n})$, either by adding them to, or removing them from, the current solution.

Local Search

1. $v \leftarrow \max_{v \in X} f(\{v\})$, $S \leftarrow \{v\}$
2. If there exists an element $x \in X \setminus S$ such that $f(S \cup \{x\}) \geq (1 + \frac{\epsilon}{n}) f(S)$ then $S \leftarrow S \cup \{x\}$. Go back to step 2.
3. If there exists an element $s \in S$ such that $f(S \setminus \{s\}) \geq (1 + \frac{\epsilon}{n}) f(S)$ then $S \leftarrow S \setminus \{x\}$. Go back to step 2.
4. Return $\operatorname{argmax}\{f(S), f(X \setminus S)\}$.

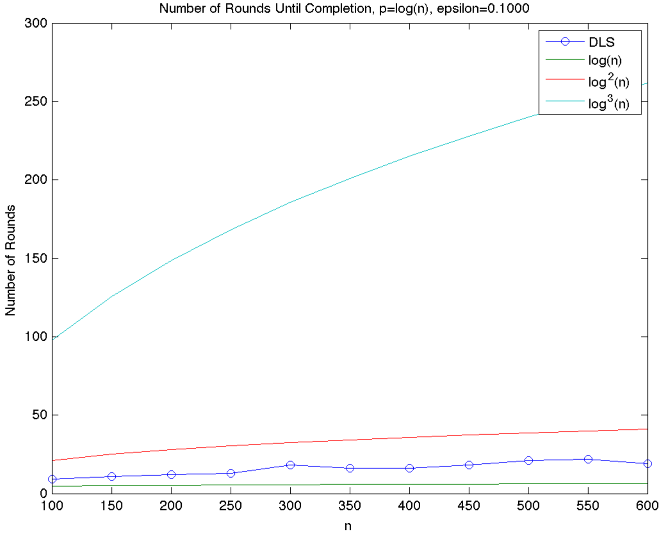
We want to parallelize Local Search.
I don't really know how to write this without having way too much detail.

Experiments were implemented in MATLAB.

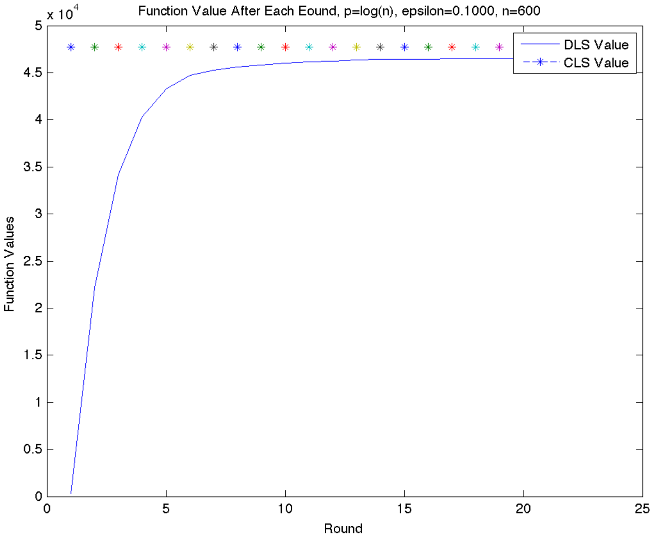
Parallelization was mimicked — the algorithm was not truly implemented in parallel.

It was also not implemented in MapReduce.

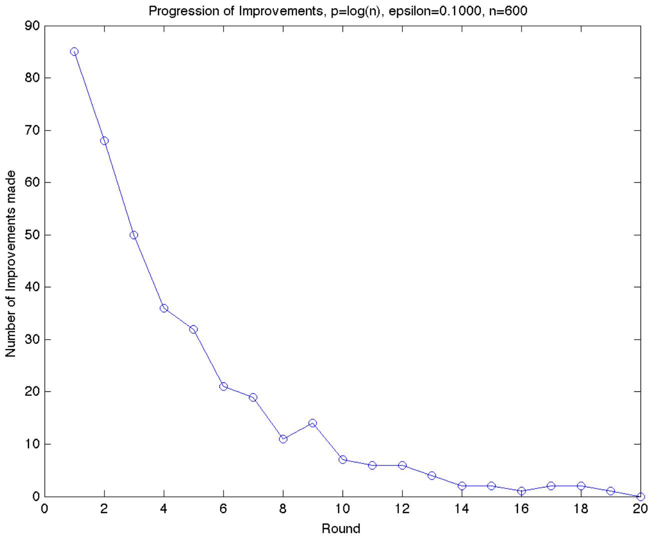
Number of Rounds



Function Value



Number of Improvements



What the f*** is going on?

Woooo!