List of topics

- About feasibility algorithm in its current form: bound and proof for single arm, bound and proof for multiple arms, patterns observed in multiple arm.
- About modified algorithm by building the sequence from the back to combat earlier drawbacks.
- Perfect algorithm for single arm, potential expansion into multiple arms.

Current feasibility algorithm

Single arm:

- Expain intuition
- Lower bound = 1-[(T-1)/2]/T
- Asymtotically towards approximation ratio of 50%.
- Is initial assumption correct? Cant think of any worse scenarios.

Current feasibility algorithm

Multiple arms:

- Expain intuition: Multiple identical arms with sharp spikes in reward
- Algorithm fails to see better alternatives. Alternative algorithm can be a solution.
- Lower bound = 1/k
- If initial assumptions for worst case scenario are correct, then this is a tight bound. Unable to think of any worse scenarios.
- The approximation ratio will never reach so low for normal bitonic functions.

Current feasibility algorithm

Single arm:

- For prior mentioned special functions which peak at single point.
- Let there be k functions and let them all peak at tau=T.
- Clear patterns emerge. Cant say how relevant they are.
- Approximation ratio = 100% till t=T+1.
- Periodic minimas in graph with a time period of T.
- First minima is seen at t=T+k. Value = 1/k.
- From t=T+1 to T=T+k, value decays predictably. For t=T+i, value=100/i.
- Number of consecutive minimas also follows a pattern.
- For large T, value seems to stabilize.

Modified Algorithm

- Employ similar feasibility concept, but start building sequences from the back.
- More high feasibility entries can be covered using this strategy.
- Solves problem of all maxima occurring at the same value of tau.(Clashing entries).
- Previous entries are not blindly allocated like before.
- Need to work out finer details, will do so over the next 2 days.
- Have not yet thought about proving its efficiency mathematically.

Perfect algorithm for one arm

- Advantage with only one arm: when we play the arm, it resets to initial state.
- Thus, use dynamic programming to solve. Will also need some notion of feasibility here.
- Define feasibility as reward(t)*[T/(t+1)]
- Find optimum for T=1. Then for T=2 and so on.
- For T=T, find element with highest feasibility. Hit it as many times as possible. Fill up the remaining slots using previously stored information.

Potential expansion

- For multiple arms, we can find the best way to fill a sequence of length T with each arm individually. If we find the N best ways with each arm, we will have KN sequences.
- Challenge: To find a compatible set of sequences that can be merged to maximize reward.
- Of the order N^K.
- If computational limit = L, take N = 10^(log(L)/K)

Other queries:

- Will the functions be defined until Tau=T-1?
- Do we want to find a measure for average performance along with worst performance?
- I have not been able to incorporate the bitonic function assumption in the algorithm or proof. Is there any way I can do so?
- How do we hope to apply such an algorithm?
- How likely are the worst case scenarios?
- Should I start working on trying to implement the problem variations? Will they be eventually incorporated anyway?