Using Variance to Infer Rate Limiting

By: David Priebel
The Problem

- ISPs limit the rate at which clients can upload
- They accomplish this by a packet queue
- Can cause high latency and packet loss
- UDP traffic fails with packet loss
- TCP traffic goes haywire
- Actually pushing to the limit is worse than being conservative
The Original Idea

- Determine ISP throttling
  - If being throttled reduce upload rate limit
  - If close to being throttled keep limit the same
  - If not in danger of being throttled increase limit

- Can be determined using ping
  - High latency and packet loss implies throttling

- Idea was to infer throttling by passive means
  - Create no new traffic
The Method

- Try to find a correlation between variance and throttling
- Accomplish this by using a genetic algorithm
- Need a data set
  - Collected using custom Azureus plugin
  - Set limit too high and then too low
  - Logged the rate, limit, and ping
- Later the data set was analyzed
The Algorithm

• Individuals
  - Stored upper and lower limits
    • Variance of upload rate
    • Latency value
    • Latency deviation
  - Each limit corresponded to a vote (-1, 0, 1)
    • Each vote is multiplied by a coefficient
The Algorithm

• Recombination
  - 3 limits are chosen from one parent
    • Other 3 limits are taken from other parent
    • Associated weights are copied as well
  - $2^{nd}$ child is formed as opposite of first

• Mutation
  - Performed at random with probability $p$
The Algorithm

• Fitness Determination
  – Individuals are fed the data and the correct decision
  – Making a correct decision adds 1 to its fitness
  – A wrong decision deducts 1 or 2

• Selection
  – Higher fitness individuals are more likely to be picked
Results

- High degree of correlation between standard deviation (and thus variance) and throttling
- This was surprising
  - Thought latency would have higher correlation
  - Latency had much lower coefficients
  - Deciding to increase was easier than deciding to decrease
    - Increase required more than one deciding factor
Problem

- Only works when upload is known to be saturated
  - Some other measurement is needed
Next Step

- Pick up where I left off
- Implement the results
- Design a new plugin to
  - Adjust upload rate limit based on variance
  - Self correct
    - Sparingly use ping for feedback
      - Adjust limits based on this data
- Implement another plugin to collect statistics
References

• For the original paper, code, and data set go here
  - http://users.csc.tnitech.edu/~djpriebel21/coursework_en: