Undergraduate Thesis Prospectus

String-Matching Algorithms Operating on LZ-compressed strings

(technical research project in Computer Science)

Success of Proposed Standards in and out of Computing

(STS research project)

by

Nathaniel Saxe

October 31, 2019

technical project collaborators:
Nathan Brunelle
Petar Duric

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

signed: _____ date: _____

approved: _____ date: _____
Peter Norton, Department of Engineering and Society

approved: _____ date: _____
Nathan Brunelle, Department of Computer Science

General Research Problem

How can we make the global computing network more efficient?

Demands for computing power and storage continue to rise, as data collection becomes more voluminous and more people join the Internet every year. Hardware performance has been pushed near its current physical limits, according to all analysts. Gordon Moore, the namesake of the optimistic Moore's Law, predicted in recent years that "I guess I see Moore's Law dying here in the next decade or so" (Moore, 2015).

Increasingly people are instead turning to algorithmic optimizations and distributed solutions like the Cloud which make better use of the global computing machine. At such scales, even an incremental improvement in efficiency is worthwhile. Implementing such an improvement requires both a good idea and the right kind of deployment strategy.

Adapting String Similarity Metrics to LZ-compressed strings

How can we modify existing string metric algorithms to instead operate on compressed strings?

This is Capstone research under Dr. Nathan Brunelle of the Computer Science department, in collaboration with Dr. Brunelle.

The goal is to write an algorithm which finds the length of the Longest Common Subsequence (LCS) between two strings which have been compressed using the common LZ algorithm. Solving the LCS problem has uses in bioinformatics and other string algorithms. There are established solutions for commonplace strings, but not compressed strings.

Problem Definition

In computer science, any sequence of text is called a *string*. Just as functions in math operate on numbers, *functions* in computer science can operate on any data, including strings. *Longest Common Subsequence (LCS)* is a function which takes in two strings and returns the longest string they both have as a subsequence. For example,

LCS(“hoopla”, “player”) = “pla”

The subsequence need not be contiguous in either string, for example

LCS(“abcdze”, “azbcde”) = “abcde”

In this research, we concern ourselves with a weaker version of LCS which outputs only the length in characters of the subsequence, not the subsequence itself.

Compression is the concept of using processing to encode the same data in less space. Compression can be *lossless*, meaning that data is exactly preserved, or *lossy*, meaning that data is only approximately preserved (typically allowing a higher degree of compression). *Lempel-Ziv compression (LZ)* is a family of lossless compression schemes ubiquitous in practice; LZ is used in the *zip* algorithm.

Roughly, LZ turns a string into a sequence of number-character pairs:

LZ(“bananasavanna”) = (0, 'b'), (0, 'a'), (0, 'n'), (2, 'n'), (2, 's'), (2, 'v'), (4, 'n'), (2, '')

with the goal of cutting recurring chunks of data by instead storing the places they first occurred.

Given two LZ-compressed strings, an easy way to find the LCS of them is to decompress them both and then call a typical LCS algorithm on them. This research aims

to find either a more sophisticated algorithm offering a way around this decompression step, or a proof that such a shortcut cannot exist.

Motivation

The LCS problem is one of a class known as string metrics, which are used to evaluate the "closeness" of two strings. These have immediate applications in bioinformatics, where genomes are ultimately large strings and related genomes score highly on string metrics. They are also the mechanism behind data comparison utilities such as `diff`, which is fundamental to any data processing or software testing. While LCS is the first target of the research, the aim is ultimately an algorithm for any useful string metric on compressed strings.

In the world of big data, text is often stored in compressed form and only decompressed when it is processed or read by humans. Decompressing and recompressing takes time, so skipping these steps could yield simpler and faster code.

Prior Work

LCS has many established solutions, and LZ compression is proven to give optimal compression ratios over the distribution of random strings (Goemans, 2015). Compression-aware algorithms are a comparatively niche subject. Dr. Brunelle's dissertation focuses on them, but applied to list and graph data rather than strings (Brunelle, 2017). In this research, we hope to apply the strategy of dynamic programming to solve problems on compressed strings without decompressing them.

Challenges

One difficulty posed by using LZ is that the same string can be compressed multiple ways depending on the width of the "sliding window" used in the compression.

While accounting for this fact will be involved, dynamic programming will still hopefully be possible because the LZ algorithm outputs pairs which "look similar" to the intermediate steps of existing dynamic programming algorithms for LCS on uncompressed strings. However, there is not any guarantee that this structure will transfer over cleanly, so we are prepared to shift focus. Pragmatically, we could choose any other of a large number of string metrics and attempt the same process. A more theoretical result would be to prove that a reliable, fast algorithm for LCS or some other metric is not possible on compressed strings. Both paths would yield insight into this realm of unexplored problems.

Success of Proposed Standards in and out of Computing

How do proponents of new standards create a discourse around adoption?

Researchers have many models for user adoption of a new technology; prominent examples are the Technology Acceptance Model (TAM) (Venkatesh & Davis, 2000) and economic models developed in *Logic of Collective Action* (Olson, 1965). Olson analyzes situations where individuals in a group have a common goal which would take work on the part of individuals to achieve. He concludes that when individuals can reap the reward without personally working toward the goal, there is a *free-rider problem* where rational agents do not choose to work. This research applies and adapts work like this to situations further qualified in two ways: 1) Adopters are specifically experts in the field, and 2) an existing standard imposes constraints on the adoption of the technology in question.

The following standards will be investigated:

- The IPv6 network protocol over IPv4
- The DVORAK keyboard layout over QWERTY
- The metric system over previous systems of measurement

I choose these particular topics to provide a contrast between levels of success in adoption. The metric system has been adopted in almost every country, with the notable exception of the United States. The DVORAK keyboard has devotees, but has failed to become standard in any sense. IPv6, though its adoption floundered in the decade since its release, is now gradually being introduced alongside the existing IPv4, if not replacing it.

The IPv6 Protocol

We commonly think of IP addresses as looking like this:

`216.3.128.12`

The protocol that uses this format is called IPv4, and it has been in use since the Internet began. There are 32 bits of information in an IPv4 address, so the number of addresses is 2^{32}, about 4 billion. In 2019, internet-connected devices number about 26 billion (Bustamante, 2019). The reason to replace IPv4 addresses is then as straightforward as any: there are more internet-connected devices than possible IPv4 addresses. IPv6 solves this problem for the forseeable eons by having an address that is four times as long. An IPv6 address looks like this:

`2001:0db8:85a3:0000:0000:8a2e:0370:7334`

Often, zeros are omitted to save space:

`2001:0db8:85a3:::8a2e:0370:7334`

Though it may seem that this simply delays the problem of address exhaustion until we have more devices, 2^{128} is in fact monstrously bigger than 2^{32} and the internet will have to double in size 96 times before we reach exhaustion for IPv6.

But since the standard's development by the IETF in 1998 and its official Launch Day promoted by the Internet Society in 2012, only 25-30% of Google requests are over IPv6 as of 2019 (Google, 2019). Replacing such a fundamental standard demands a graceful transition. The longer it takes, however, the longer network engineers will have to work with two incompatible protocols.

In the 1980s, the internet was developed as an experiment by researchers for the DoD, but was so successful that, according to the co-inventor of IP Vint Cerf, "We've been using the experimental internet design since … 1983 when we turned it on" (Cerf, 2014). The early engineers just wanted to get something working, with low anticipation that the project would be wildly successful. Many of these engineers fully support IPv6. Vint Cerf, now co-founder and Vice President of the Internet Society, attests that "It's been a long time since these standards were put in place, and it's time to get with the program" in regards to IPv6 connectivity (Cerf, 2018).

The Internet Society (ISOC) is a prominent nonprofit which defends the original vision of the internet as a global communications network. It promotes global development of Internet infrastructure, publish information about privacy risks, and puts standards like IPv6 in place through the IETF, now a subsidiary organization. ISOC has sponsored a World IPv6 Day in 2011 and a broader World IPv6 Launch in 2012.

The larger a network is, the more effort it takes to switch over to a different protocol. In response to ISOC, large telecom companies assigned teams to implement it,

but the task of switching to IPv6 has been more difficult for enterprise companies' internal networks. The effect of this is visible (fig. 1). The spikes in IPv6 usage occur weekly, with around 30% global adoption on weekends compared to around 25% on weekdays. During the week, many people use their employer's internal network, while on the weekend they go home and use WiFi from a large service provider.

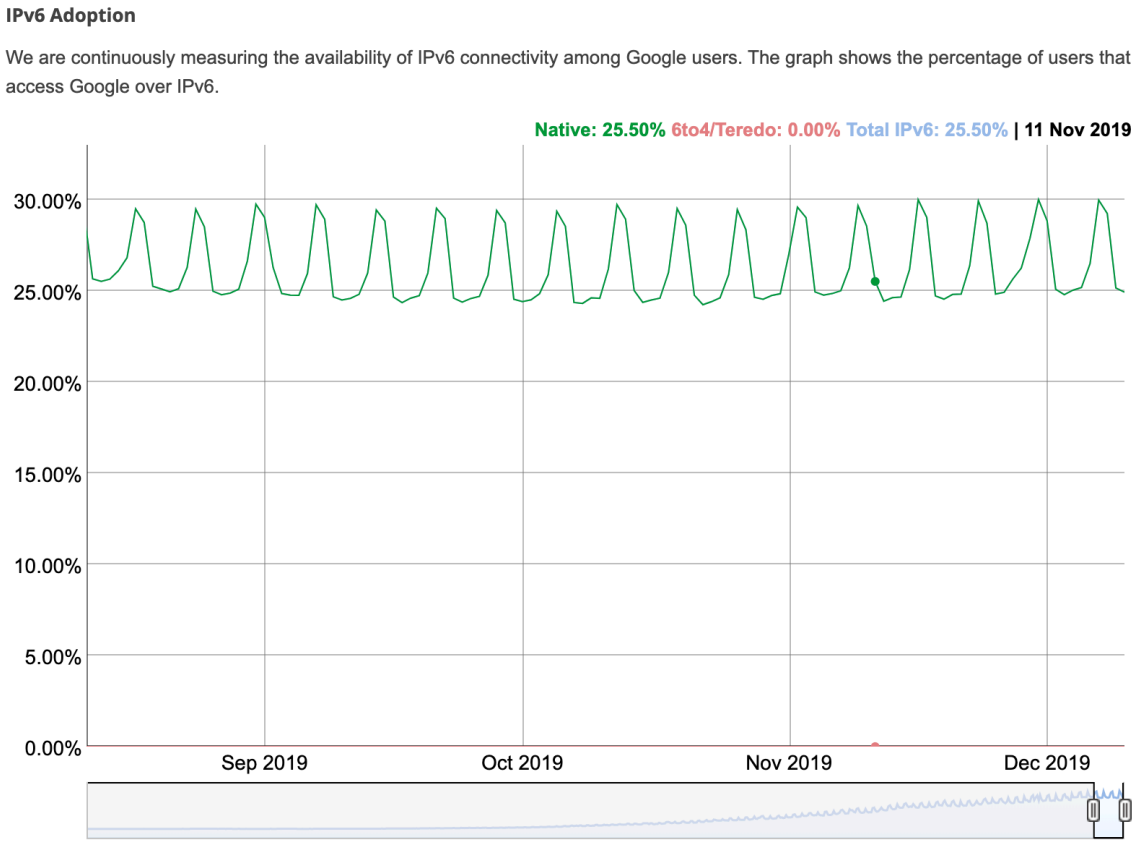

Figure 1. IPv6 Adoption, Aug-Dec 2019 (Google Analytics, 2019)

Major companies with legacy equipment that still want to expand must decide between converting to IPv6 and purchasing blocks of IPv4 addresses from their current owners. Amazon, for example, conspicuously purchased a large chunk from MIT in April 2017 (Schmidt, 2017).

IPv4 has benefited from decades of use and improvement, and as a result is considered more secure than IPv6, a fact the Internet Society does not dispute. As a

result, most VPNs choose not to support IPv6, and some encourage users to disable IPv6 on their devices (Craig, 2017).

References

Brunelle, N. (2017) *Super-Scalable Algorithms*
https://www.cs.virginia.edu/~njb2b/Brunelle_phdDissertation_UVACS_2017.pdf

Bustamante, J. (2019) *IoT Statistics*
https://ipropertymanagement.com/iot-statistics

Cerf, V. (2014) in an interview with Leo Laporte
https://www.youtube.com/watch?v=17GtmwyvmWE&feature=share&t=26m18s

Cerf, V. (2018) *Vint Cerf on IPv6 and the 6th Anniversary of World IPv6 Launch*
https://youtu.be/_YiAzfHQF-g?t=131

Craig, C. (2017) *NordVPN Implements IPv6 Leak Protection*
https://nordvpn.com/blog/nordvpn-implements-ipv6-leak-protection/

Goemans, M. (2015) *Lempel-Ziv Codes*
http://math.mit.edu/~goemans/18310S15/lempel-ziv-notes.pdf

Google (2019) *IPv6 Adoption Graph.*
https://www.google.com/intl/en/ipv6/statistics.html#tab=ipv6-adoption

Gunderson, S. et al. (2009) *Evaluating IPv6 Adoption in the Internet.*
https://storage.googleapis.com/pub-tools-public-publication-data/pdf/36240.pdf

Moore, G. (2015) in interview with Rachel Courtland for IEEE
https://spectrum.ieee.org/computing/hardware/gordon-moore-the-man-whose-name-means-progress

Schmidt, M. (2017) *Next-generation MITNet*
http://orgchart.mit.edu/node/6/letters_to_community/next-generation-mitnet

Venkatesh, V., Davis, F. (2000) *Management Science*, Vol. 46, pp 186-204, "A Theoretical Extension of the Technology Acceptance Model."
https://www.jstor.org/stable/pdf/2634758.pdf

York, D. (2019) *What Can You Do Today to Promote IPv6?*
https://www.worldipv6launch.org/celebrating-the-7th-launchiversary-what-can-you-do-today-to-promote-ipv6/