Recitation 13 — Physical Deployability

Motivation

- We covered datacenter networks in the previous lecture; this paper is about actually deploying new network designs in datacenters.
- In the next lecture (after spring break), we're going to start talking about failures. Failures are sometimes caused by deployability problems. We'll also talk about a system's "mean time to repair (MTTR)", which, as this paper states, is "an inherently physical problem".

Authors

- From Google, work with Google's datacenters; have expertise in this problem

Overview

- Physical deployability refers to "interactions between network equipment and the physical world". I.e., actually *physically building and maintaining* a new network (putting switches and cables in the right place, connecting things correctly, repairing things, etc.)
- Has traditionally not been a huge topic of concern for network researchers.
- Many designs that look good on paper turn out to not be physically deployable (or need changes to be deployed).
- Prioritizing physical deployability can mean prioritizing things such as fungibility over our "normal" performance metrics.

Thought questions

- Why do you think physical deployability isn't typically addressed in research literature? Is this a problem? If so, is there a solution?
- Is a proposed new design for a datacenter useful if it doesn't take physical deployability into account?
- The paper states that "Abstraction is unquestionably necessary, but the hidden constraints mean that designs that look appealing on paper can turn out to be infeasible." How should we, as system designers, respond to that? Are there things that we could do to make better abstractions? To get a better sense of the constraints?
- The paper uses metrics such as "time to deploy", "cost to deploy", "first-pass yield". How would we calculate those metrics? *Who* can calculate them? How should we think about these metrics in conjunction with "normal" performance metrics such as latency, throughput, etc. Are they ever at odds?