Flow Networks

Recap

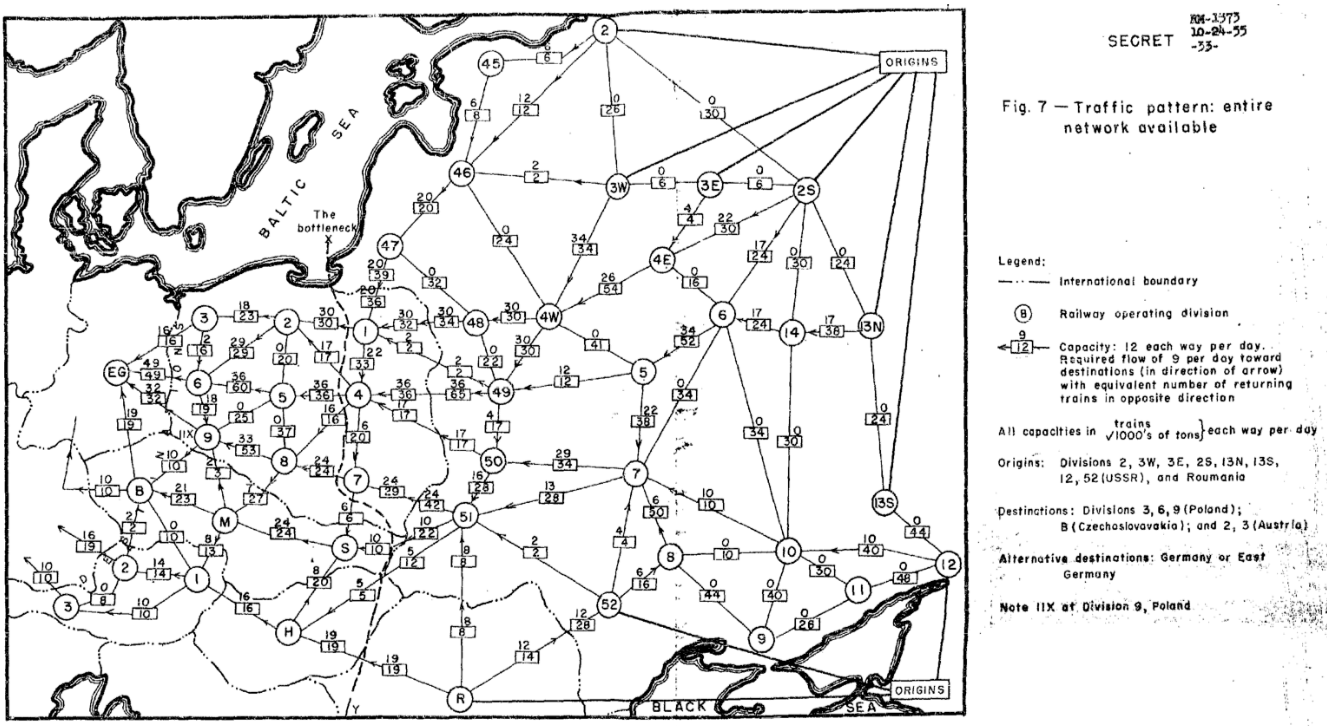
In Session 01 we have encountered graphs, especially the undirected and directed kind. Recap together:
what is a...

- graph
- node
- edge

What happens to graphs if you plot ...

- a relation (have you met before?)
- a property (do you wear glasses)?

historical context



In the mid-1950s, U. S. Air Force researcher **Theodore E. Harris** and retired U. S. Army general **Frank S. Ross** wrote a classified report studying the rail network of the Soviet Union, the Soviet occupied states and Soviet satellites in Europe.

The network was modelled as a graph with 44 nodes, representing geographic regions, and 105 edges, representing the rail network. Each **edge** was given a **weight**, representing the rate at which material could be shipped on it by train.

They determined both the maximum amount of material that could be moved from the Soviet Union, as well as the **cheapest way to disrupt the network by removing edges (or in less abstract terms, blowing up train tracks),** which they called “the **bottleneck**”.

Their report, which included the drawing of the network in Figure 10.1, was only declassified in 1999.

[short round for Af?ekt! method on this introduction.]

guiding quote for this session

"The main trouble with cyborgs, of course, is that they are the illegitimate offspring of militarism and patriarchal capitalism, not to mention state socialism. But illegitimate offspring are often, manifestly exceedingly unfaithful to their origins. Their fathers, after all, are inessential."
– ***Donna J. Haraway***, *A Cyborg Manifesto*

Keep this in mind while we engage in Graph algorithms. They and furthermore optimiziation algorithms are the children of the age of Cybernetics, and the cold war. Keep your STS eyes open for how that shows up!

Flow

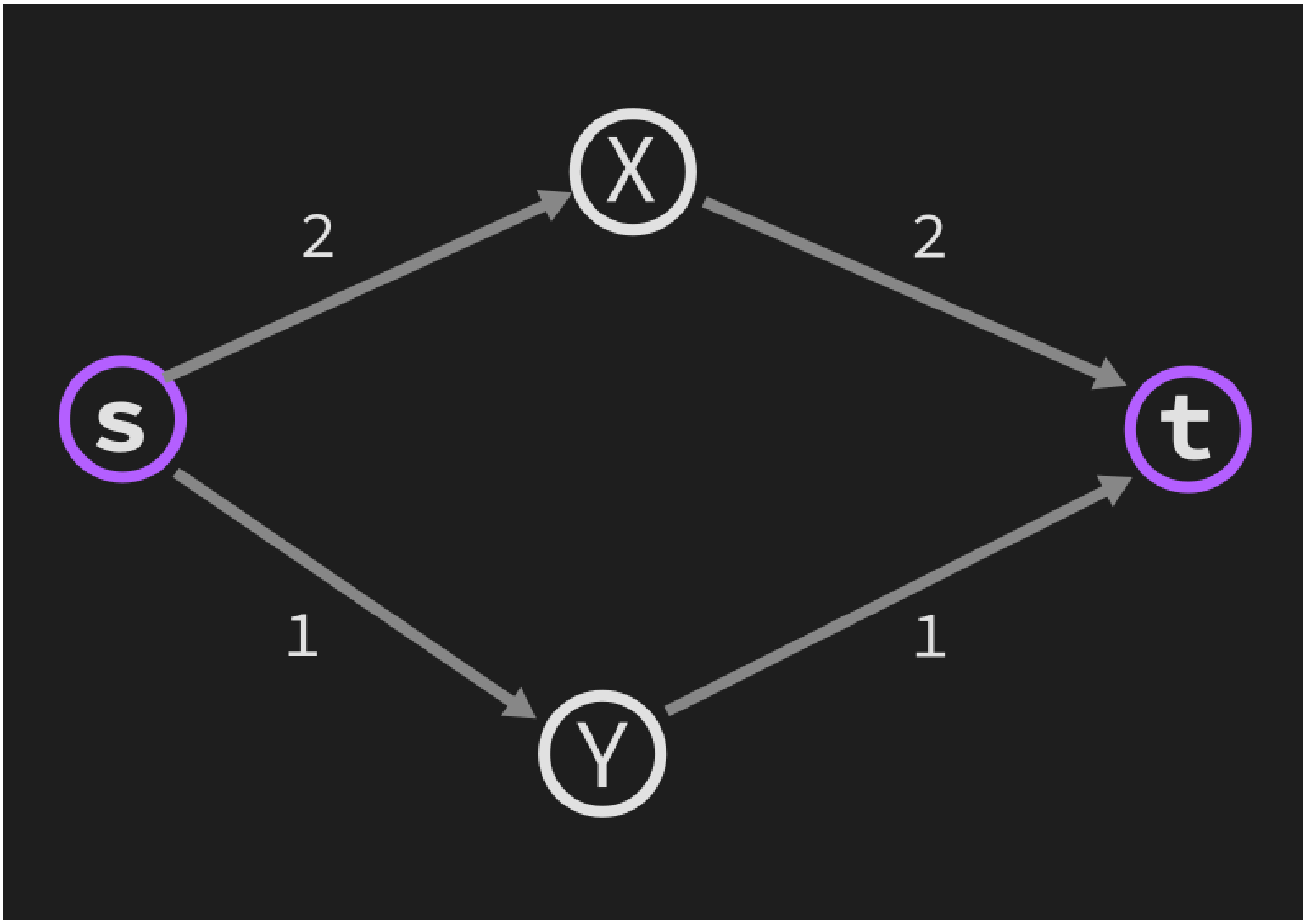
Flow is basically just units of "stuff" that we send over a network. As we can imagine a lot of things that we could model like this, train cargo, sewage water, money transactions, or tasks, we try to keep it abstract and just think of "flow"

structure of flow networks

In the graphs we encountered before, all nodes are alike, but Flow Networks have an exception:
We always need a node **s = source** of the flow

And a node **t = target** of the flow

example of a simple flow network:



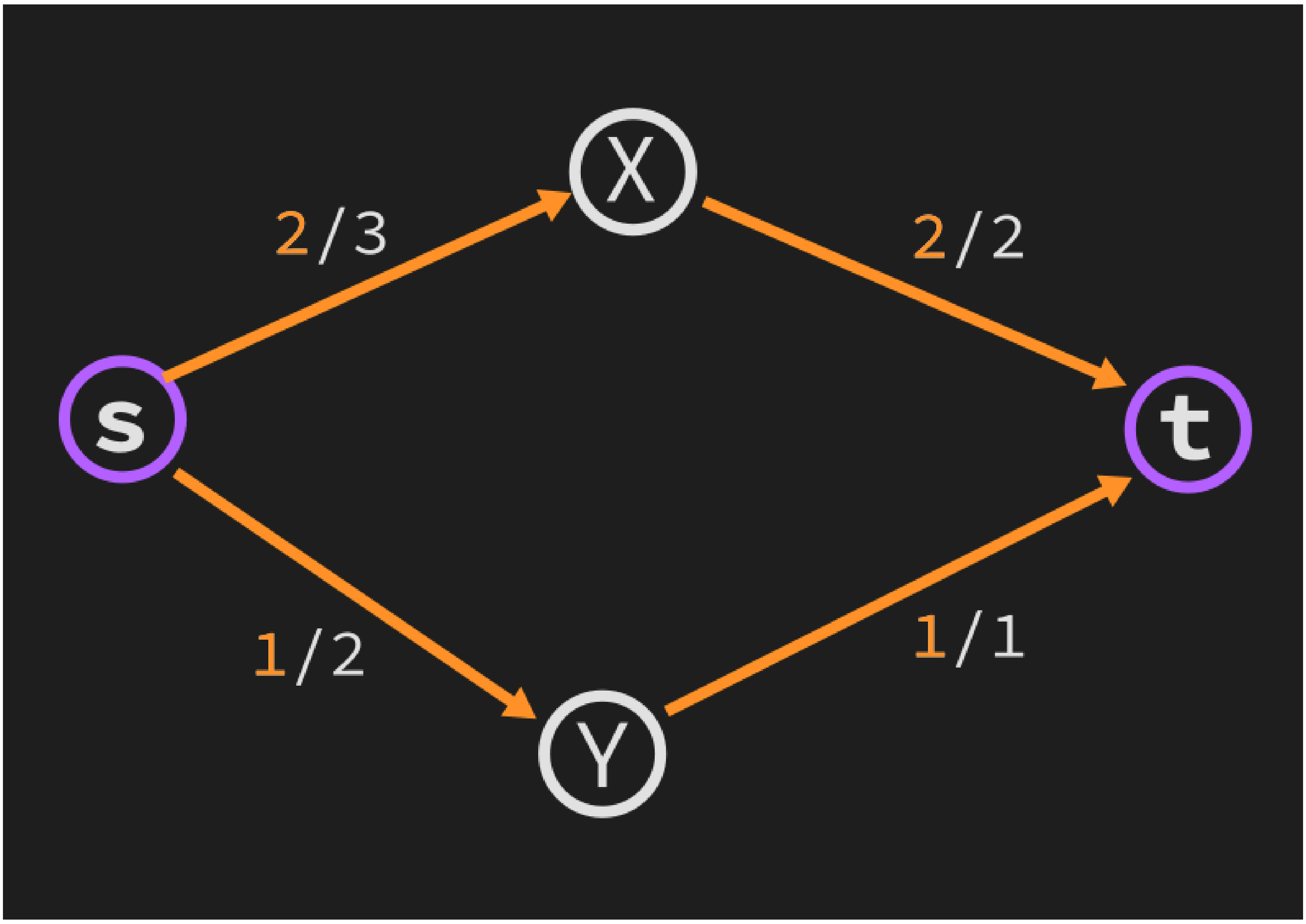
edge capacities

Again (remember session 01) we have edges, we denominate them with **e**.

Edges have **capacities**: how much flow could they potentially carry?
The capacity of an edge is denominated as **c(e)**

Edges have **flow**: how much flow is on this edge? We denominate the flow on a particular edge as **f(e)**
Flow is always less or equal to the capacity of the edge.
0 <= flow <= capacity

0 <= f(e) <= c(e)



conservation of flow

What amount of flow comes into a node, must be exactly equal as what comes out on the other side. Very important!

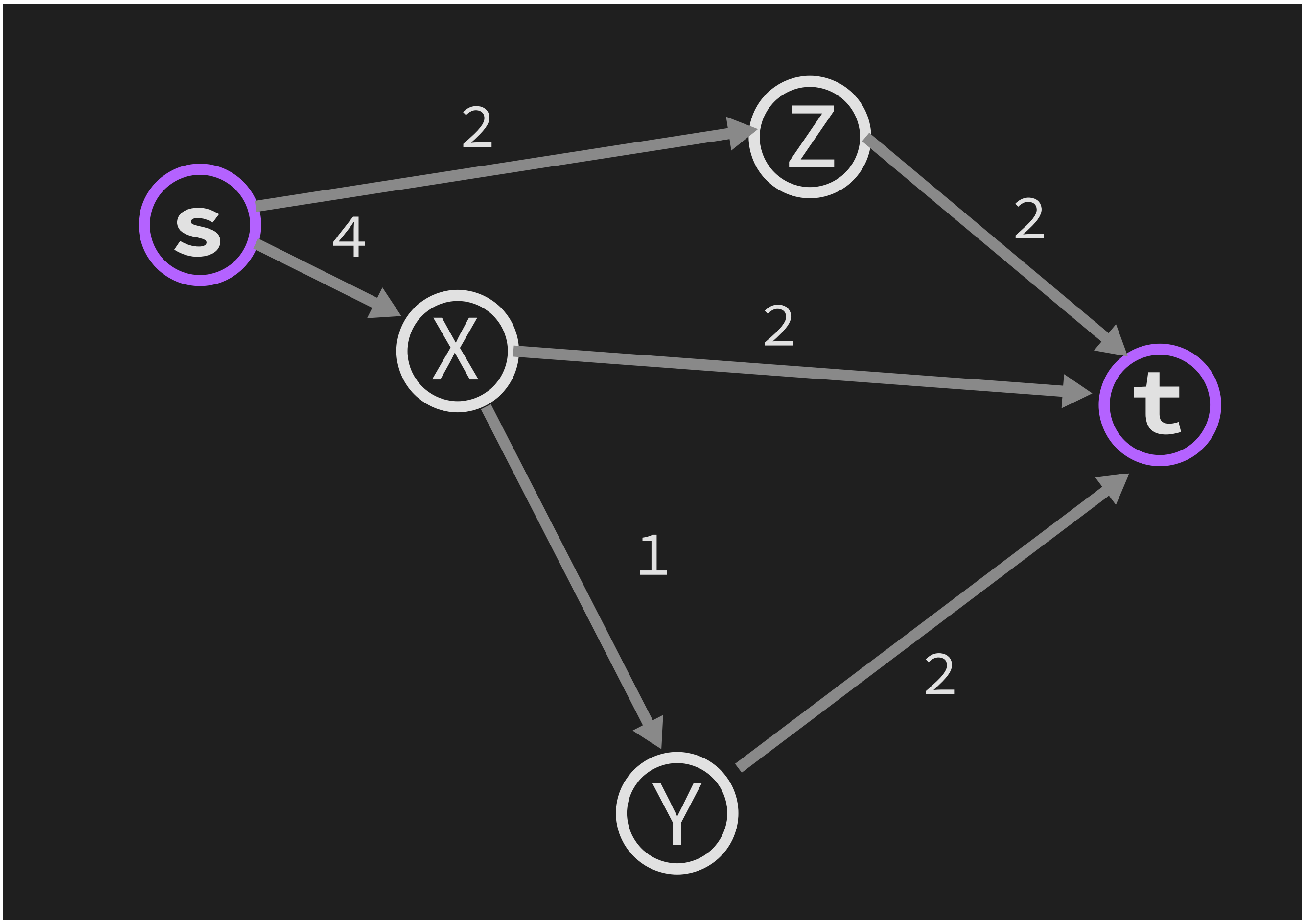
Now it's time for [excercise 03.1]

The Ford Fulkerson Algorithm

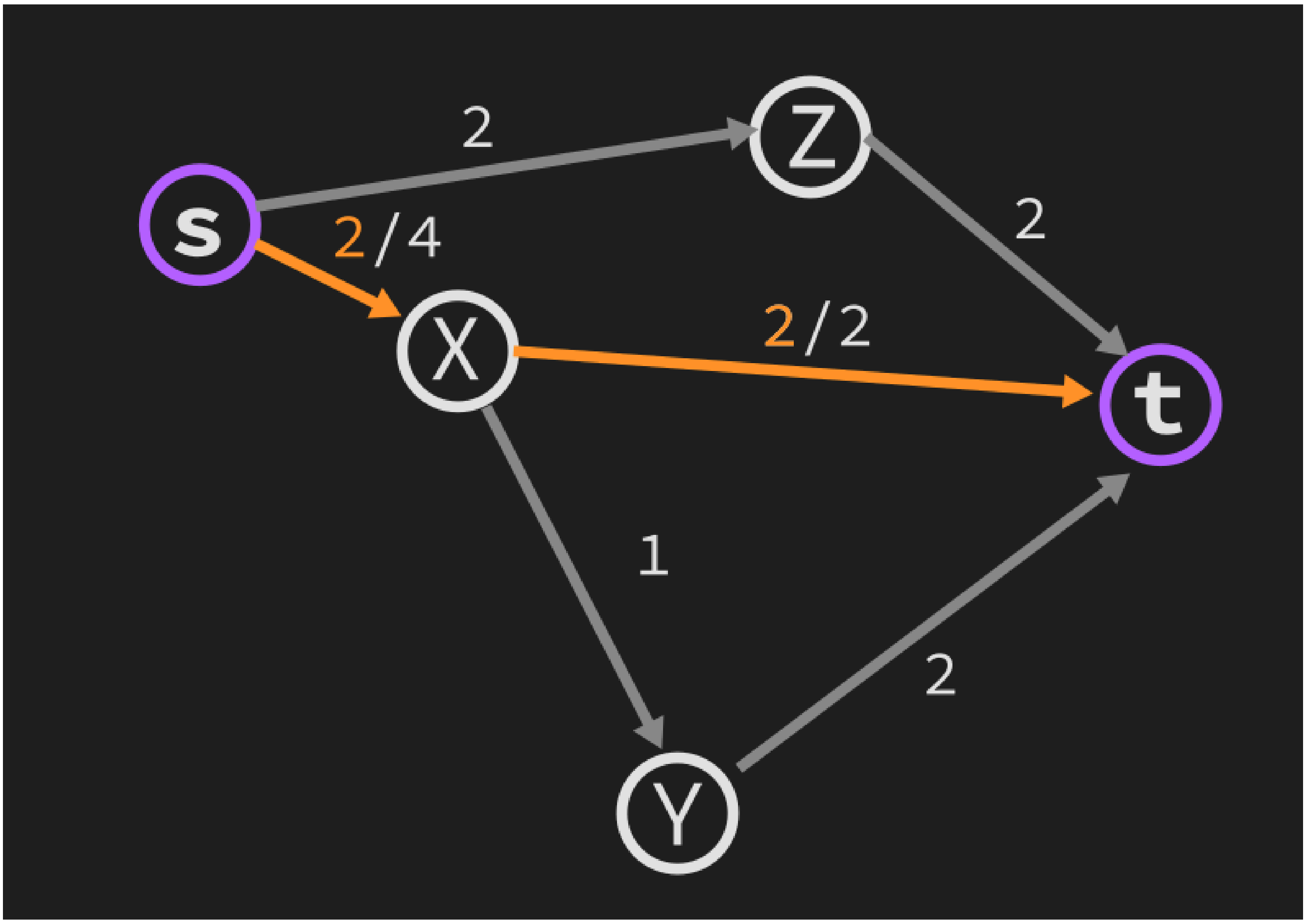
This Algorithm attempts to find the Maximum amount of flow in a Network. This is a slightly simplified version of Ford Fulkerson, for the sake of being approachable. The recipe of this algorithm is the following:

```
repeat until you can find no more paths from S to T:
    find a path from the source S to target T
    send the maximum amount of flow over this path.
    For each edge e on this path:
        Construct the so called "residual edges" e'.
        lower the capacity of the forward edge e to c(e)-f(e)
        insert a backwards edge e', that carries c(e') = f(e) capacity and flow
        if an edge's residual capacity c(e)-f(e)= 0 it is erased.
```

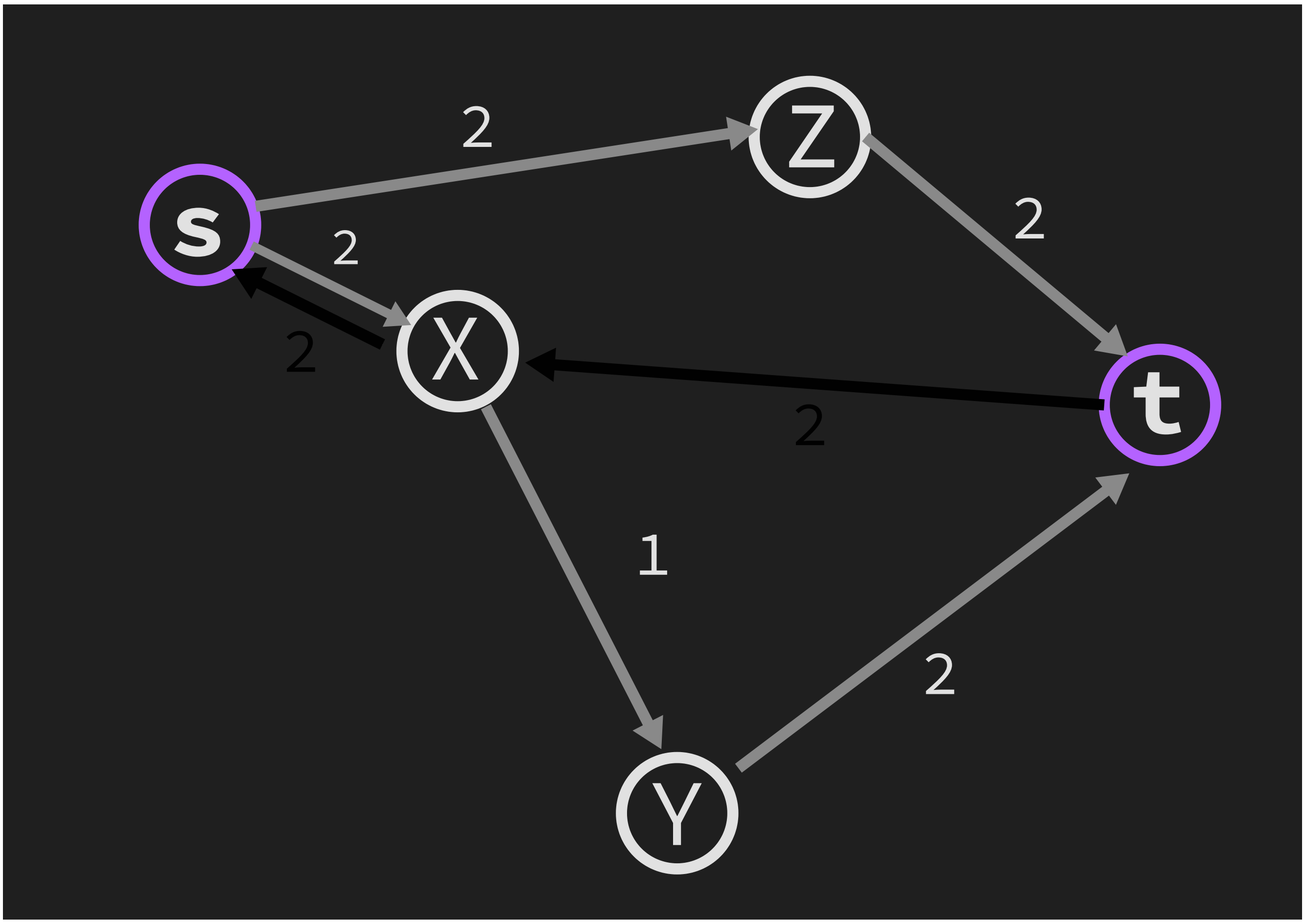
example in Pictures



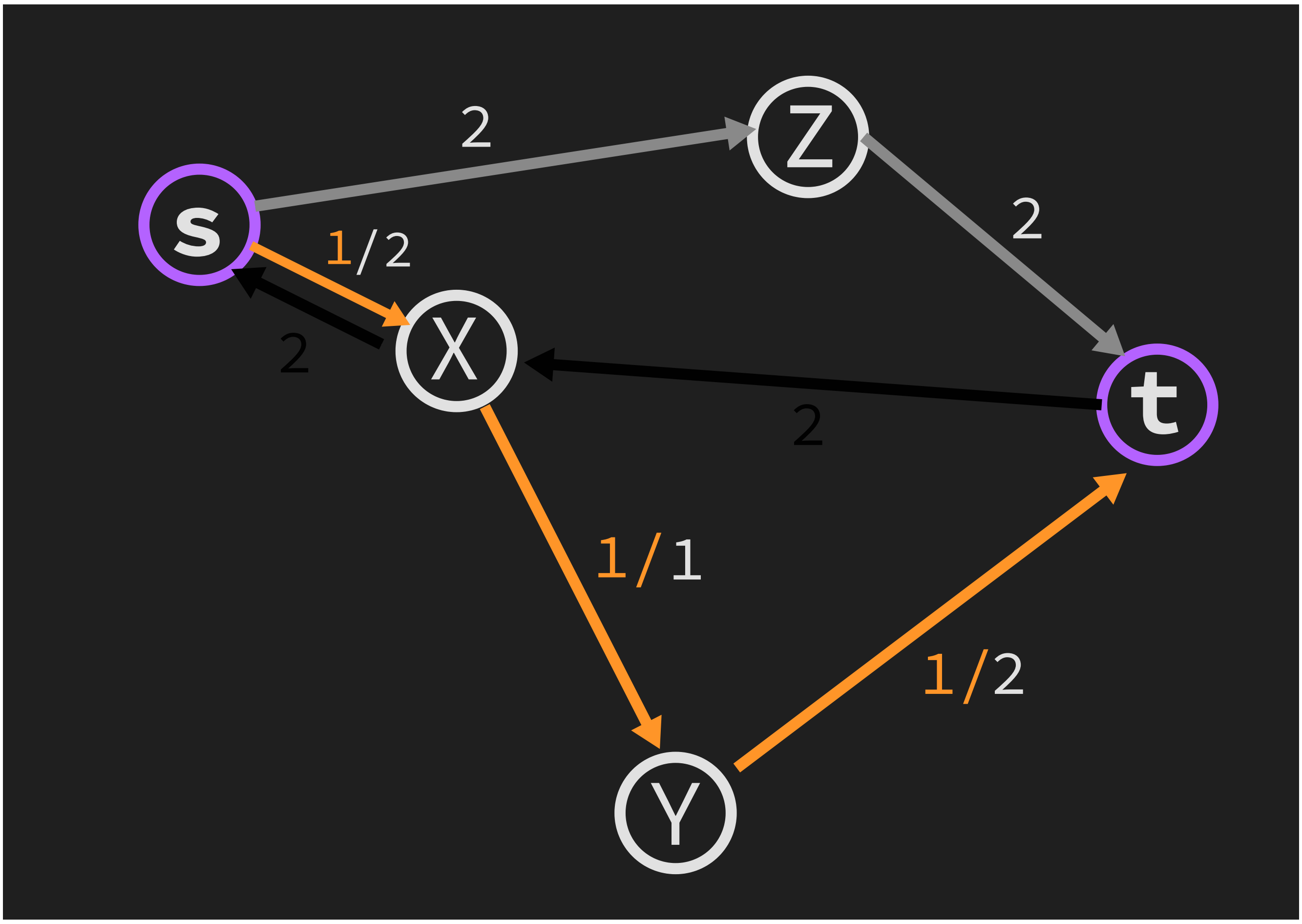
step 0: this is our network



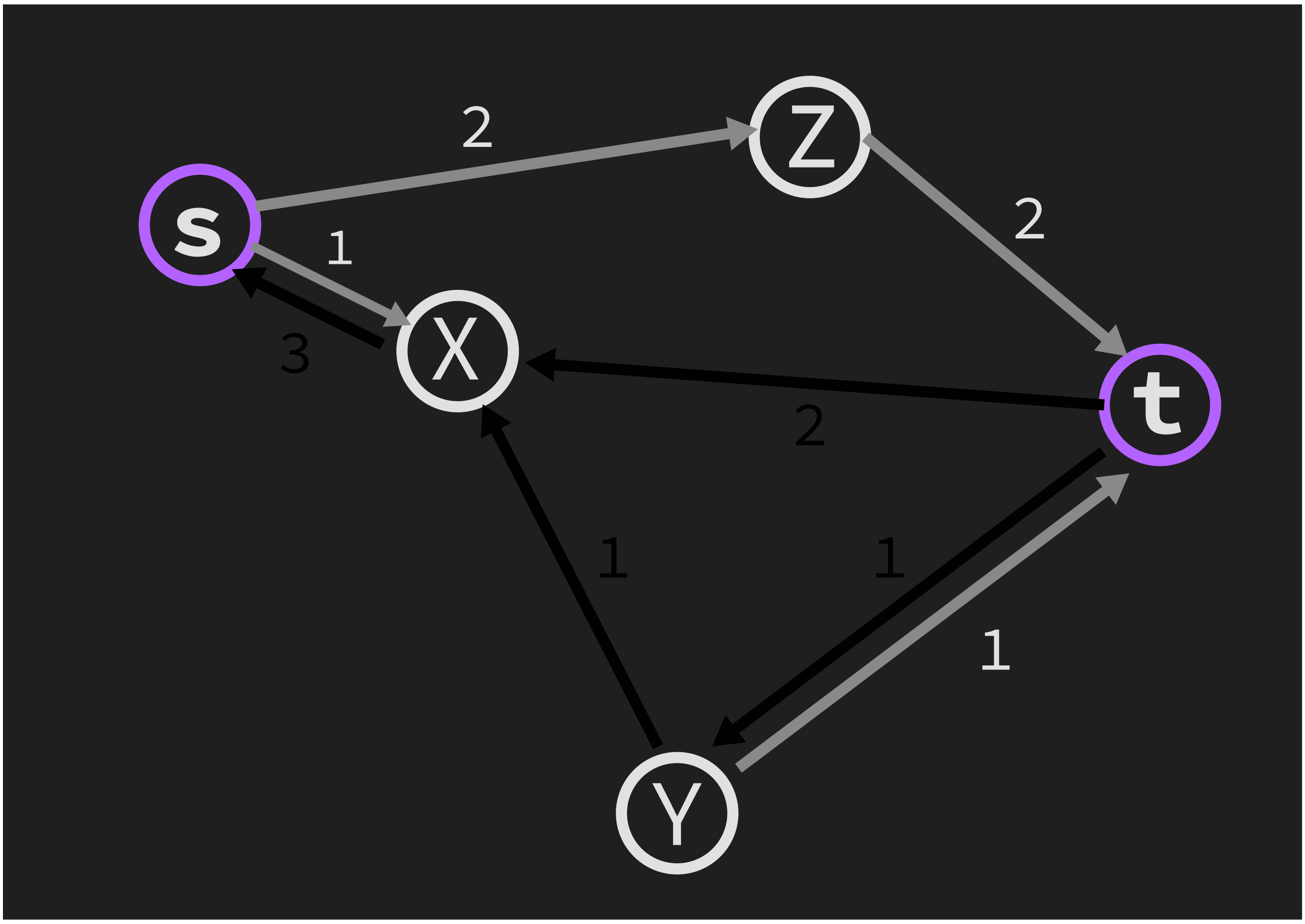
step 1: find a path from s to t = s, X, t
push flow over it.



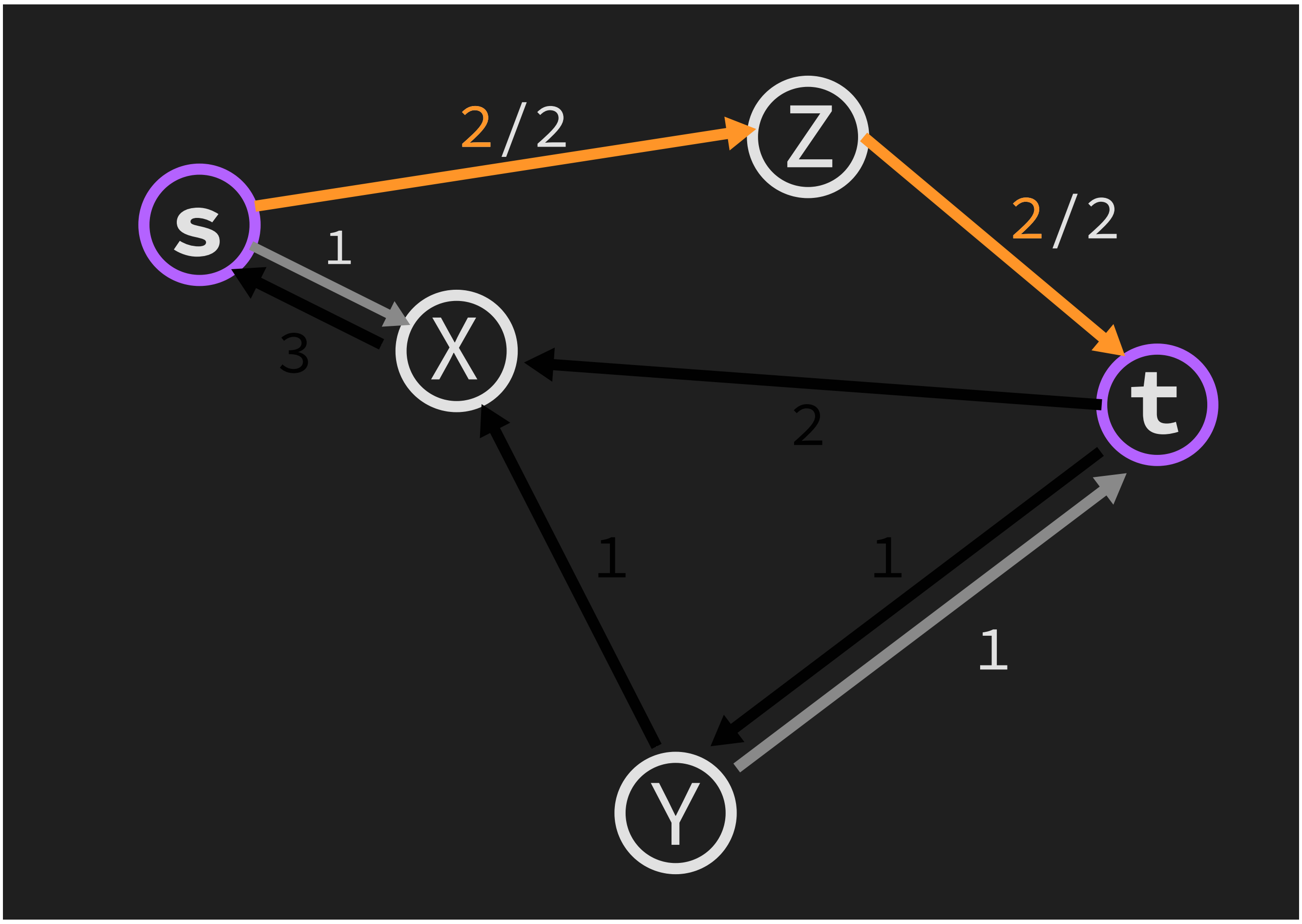
step 2: residual Network



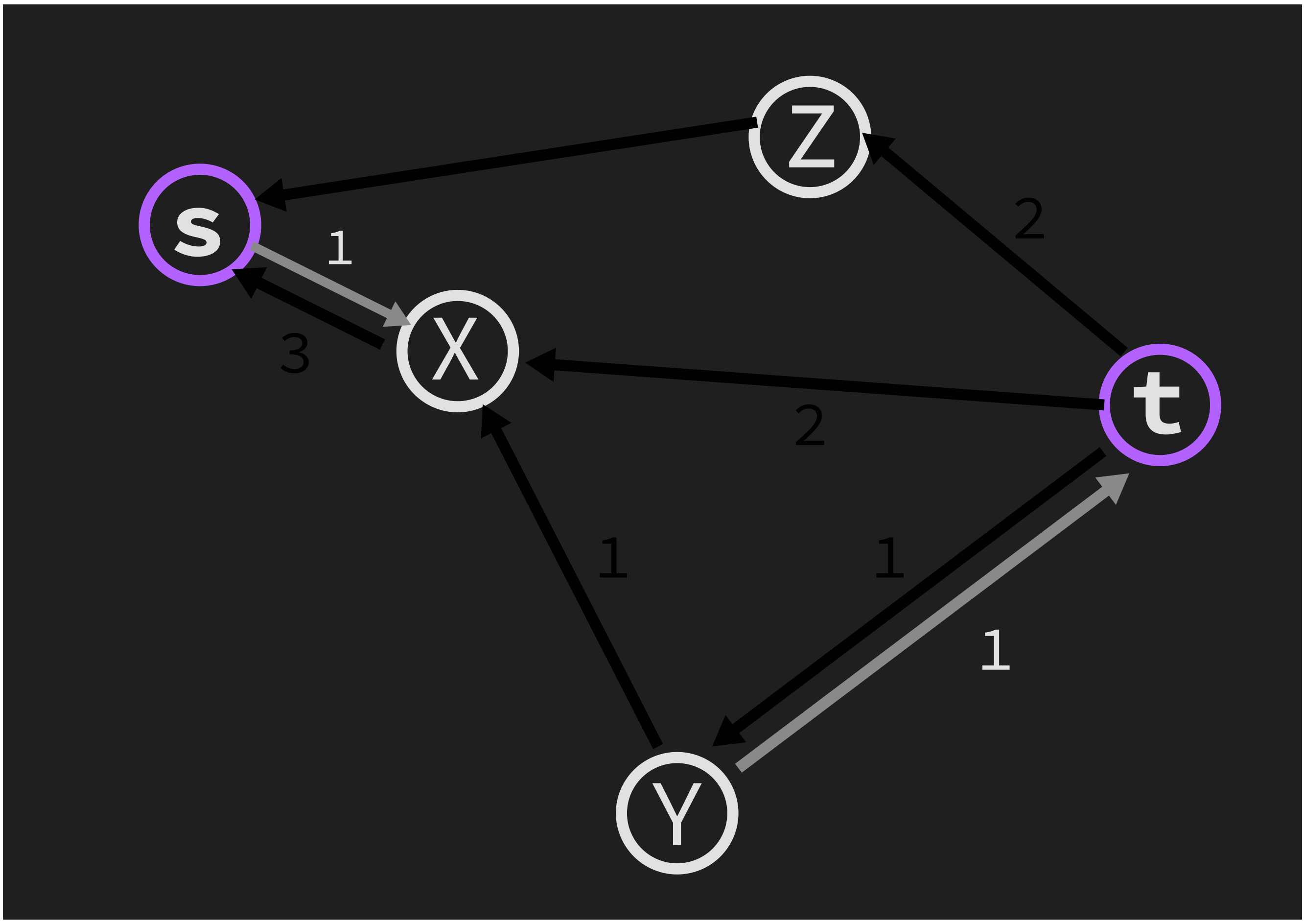
step 3: Find a path from s to t: s, X, Y, t
push flow over it.



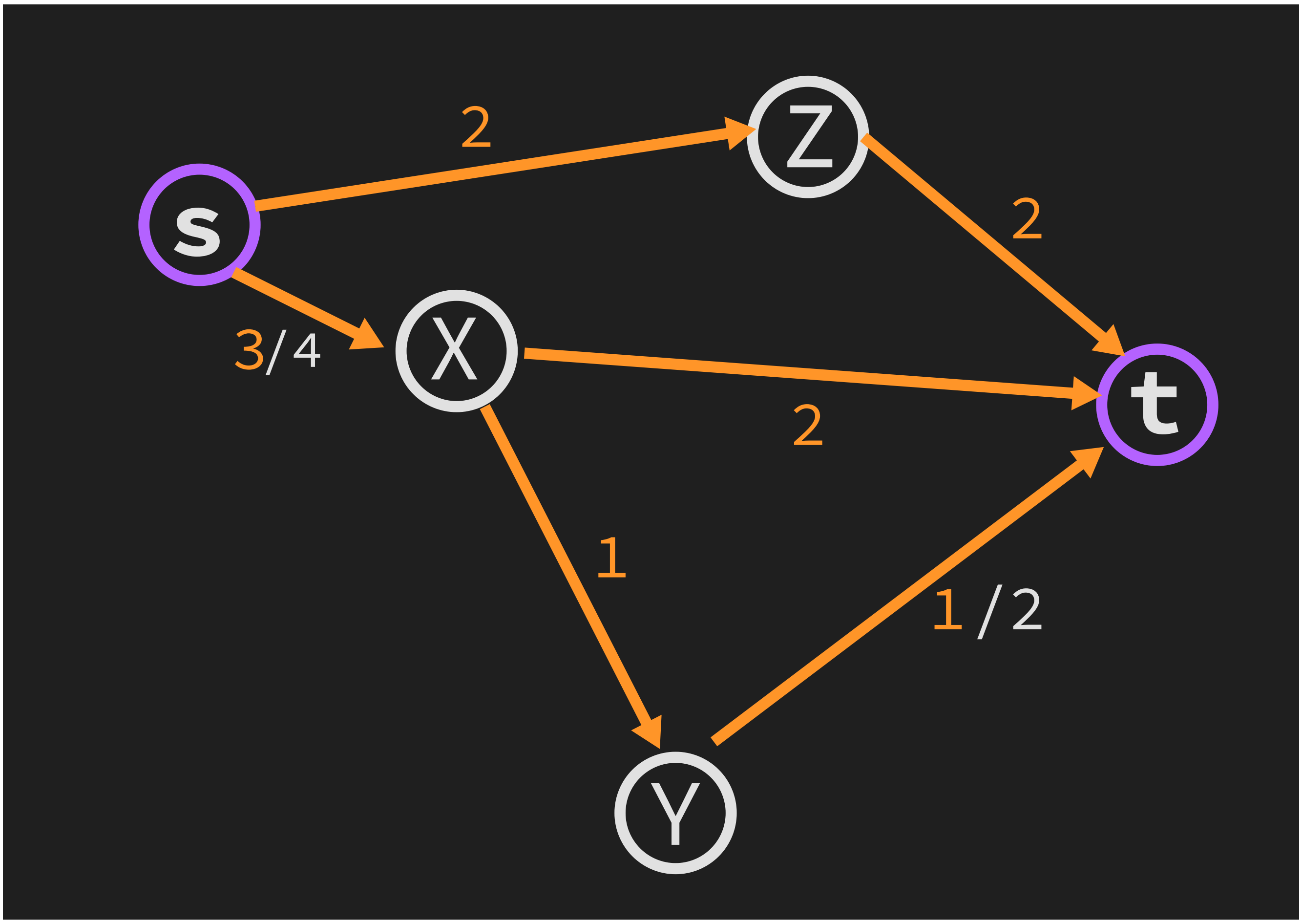
step 4: residual Network



step 5: find a path from s to t: s, Z, t
push flow over it.



step 6: residual network



step 7: there is no s,t path left!

That means we maxed out the flow network

Now it's time for [excercise 1d.3 Flow Networks]

BONUS

(s,t)-cuts

We can use an intriguing visual property to understand more about flow networks:
a (s,t)-cut is a set of edges in the network, that if we remove them, separate the source s and the target t. the **Maxflow-Mincut Theorem** tell us that the cut whose cut out edges have the minimum sum of capacities, that sum equals the maximum amount of flow in the Network.

Additional Material:

If you have become curious about flow networks and would like to dive in (much) more technically, for this presentation i used Jeff Erickson's "Algorithms" as source material:
https://jeffe.cs.illinois.edu/teaching/algorithms/book/Algorithms-JeffE.pdf

Haraway, Donna J.. Manifestly Haraway, University of Minnesota Press, 2016,
https://warwick.ac.uk/fac/arts/english/currentstudents/undergraduate/modules/fictionnownarrativemediaandtheoryinthe21stcentury/manifestly_haraway_----_a_cyborg_manifesto_science_technology_and_socialist-feminism_in_the_....pdf