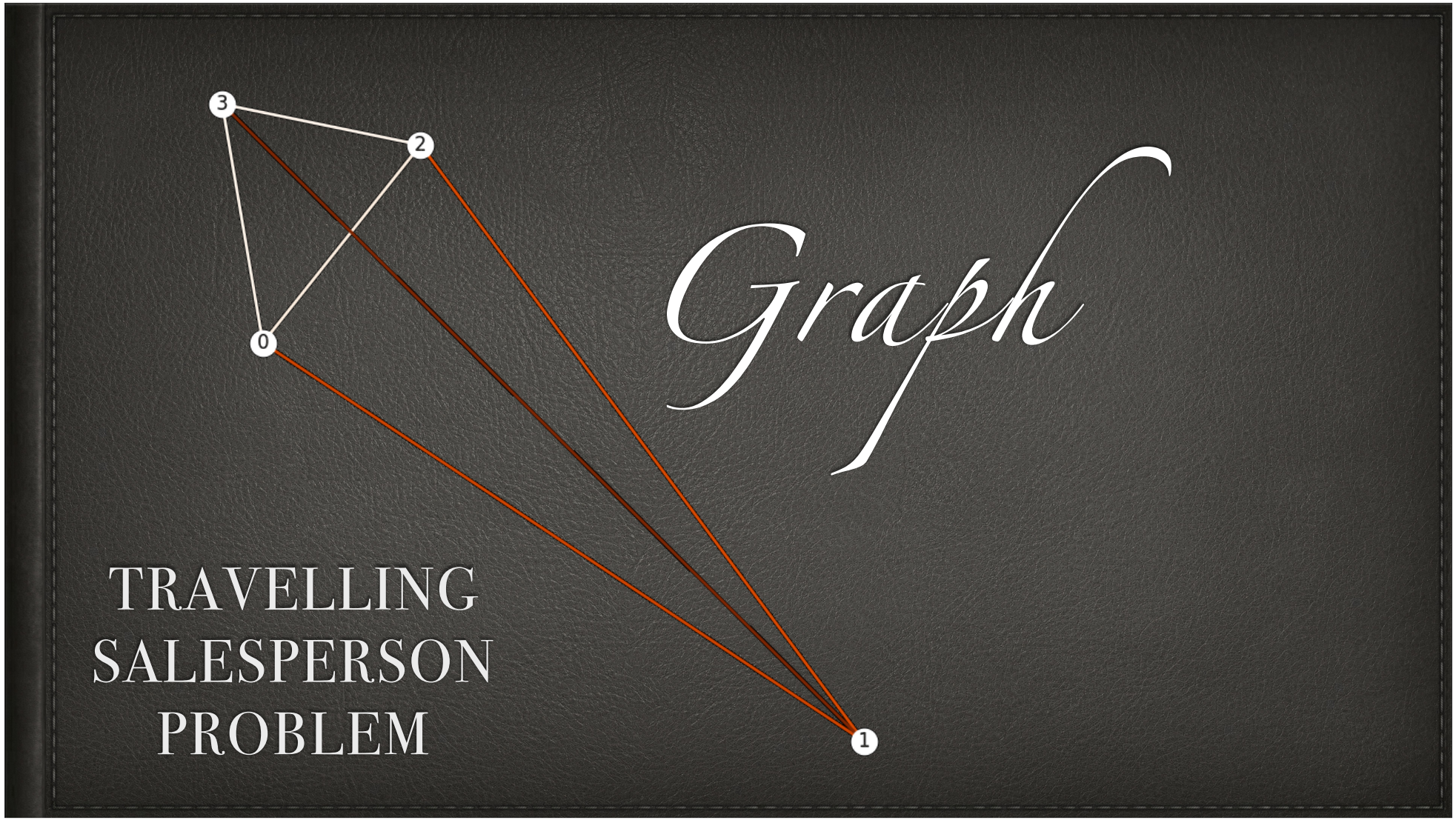
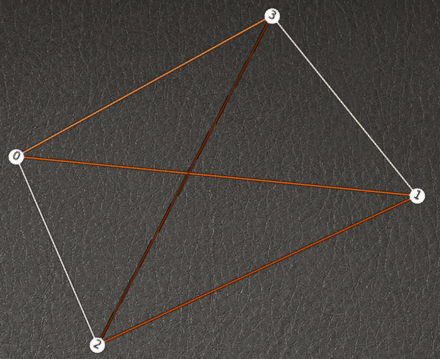
Discrete Math meets Deep Learning

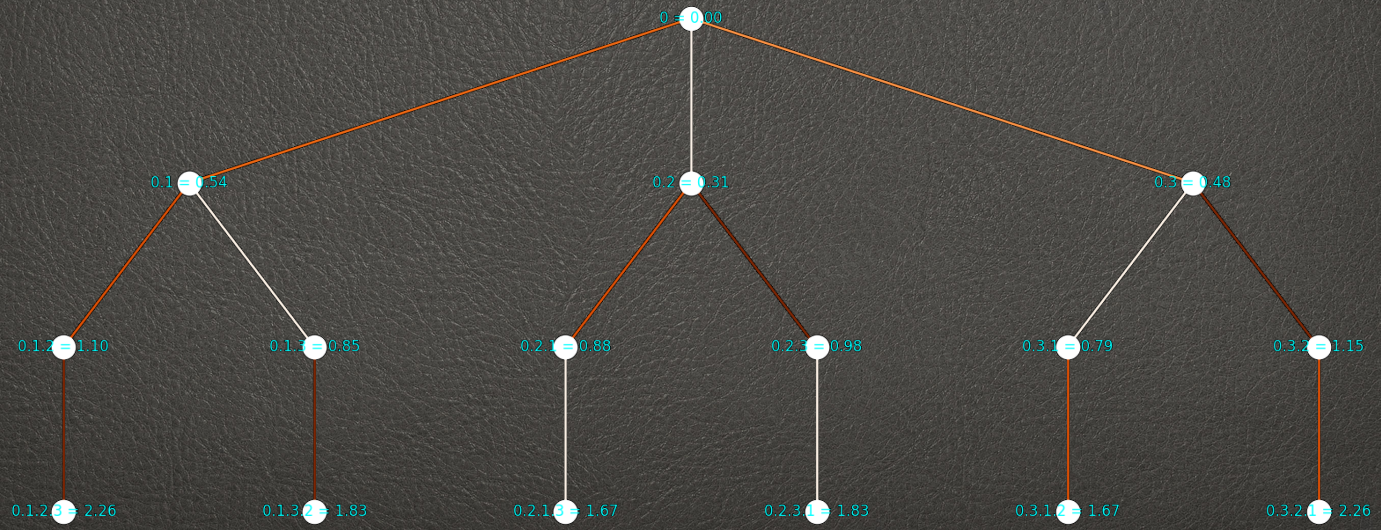
a hands-on case study with TSP

No previous knowledge required

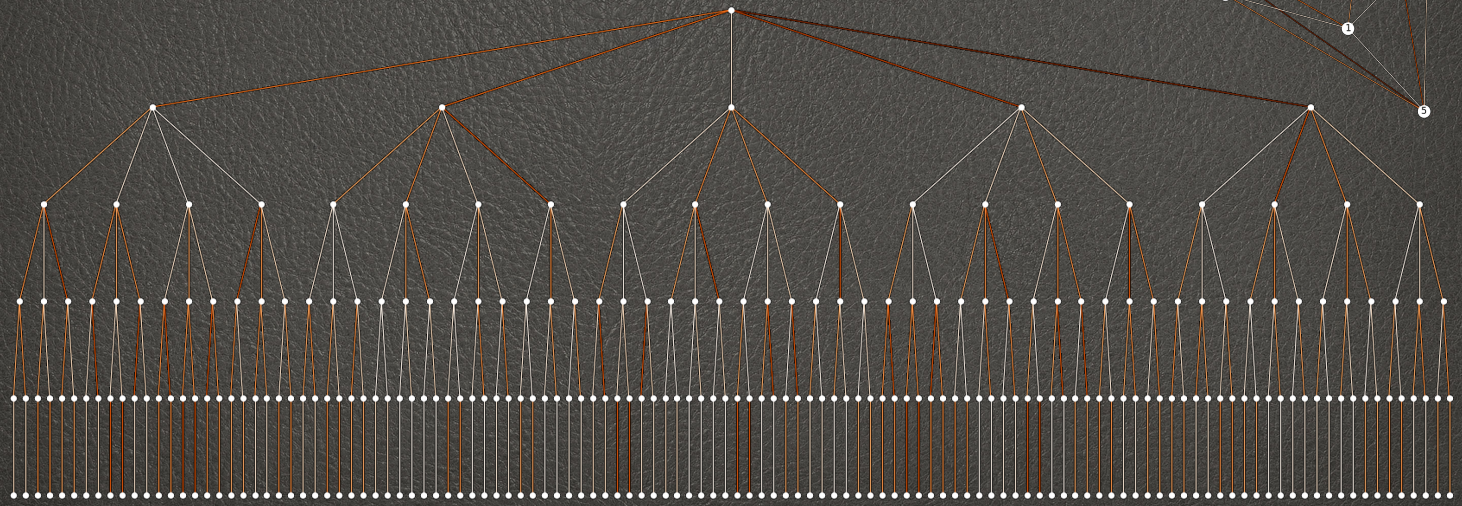

3
2
0
1
Graph
TRAVELLING
SALESPERSON
PROBLEM

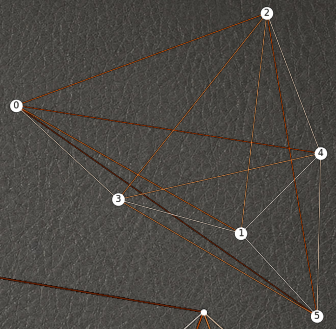
Exhaustive search





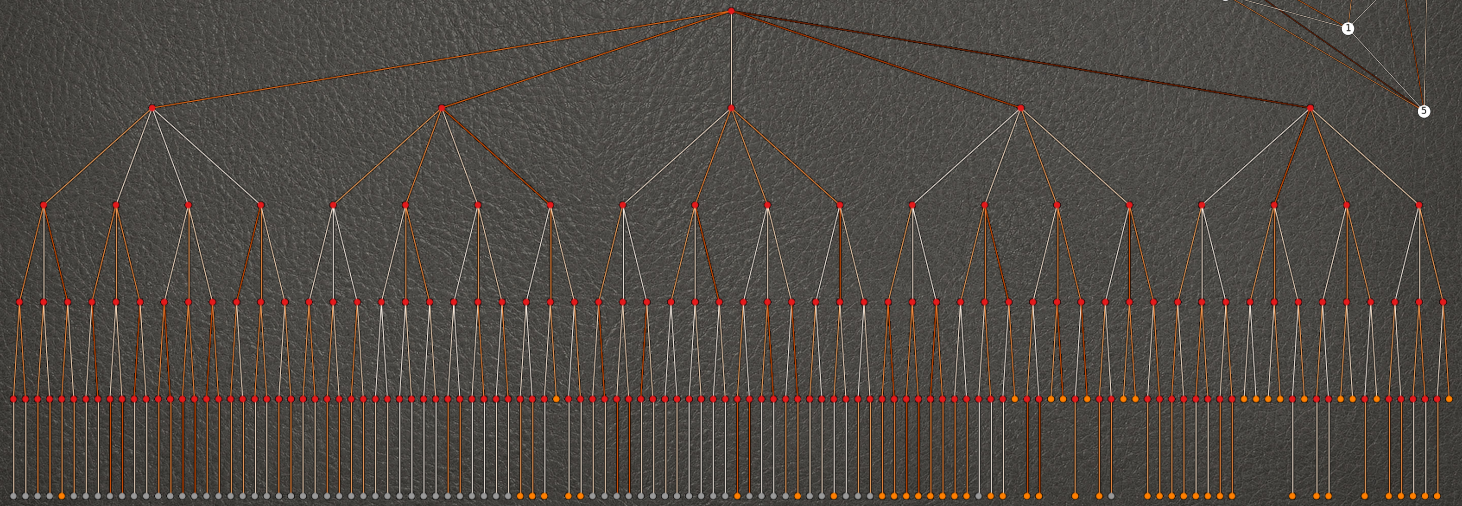
Combinatorial explosion

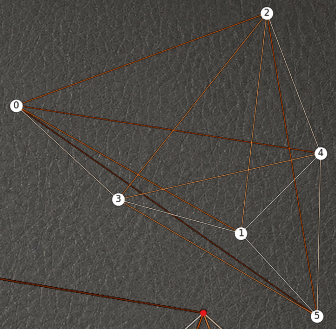




With just six 🐱

Pruning helps, a little





With just six 🐱

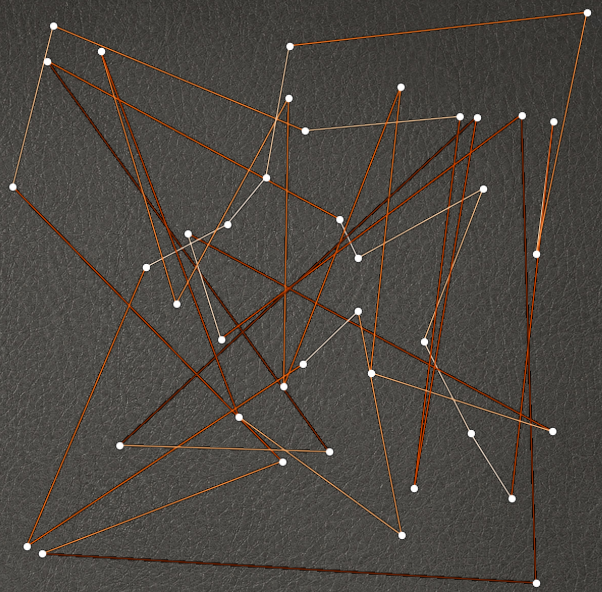
Global optimum

Example runtime in milliseconds

n	Exhaustive	Pruned
8	88	30
9	825	264
10	9021	1371
11	*lost patience*	3234
12	*lost patience*	23815
13	*lost patience*	*lost patience*

😺 Nothing can produce a lower cost

Those trees grow too big 😿

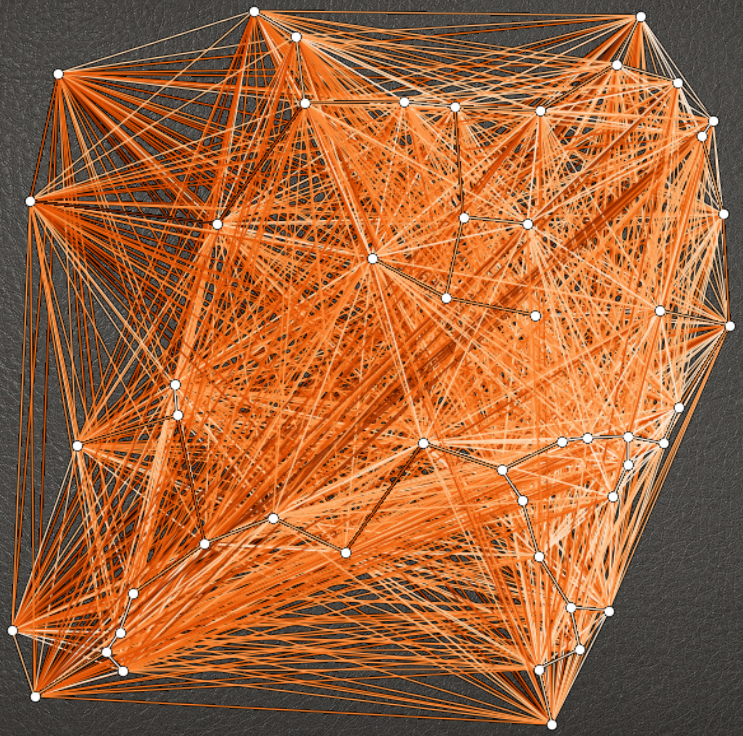


Replicas of self-avoiding random walks

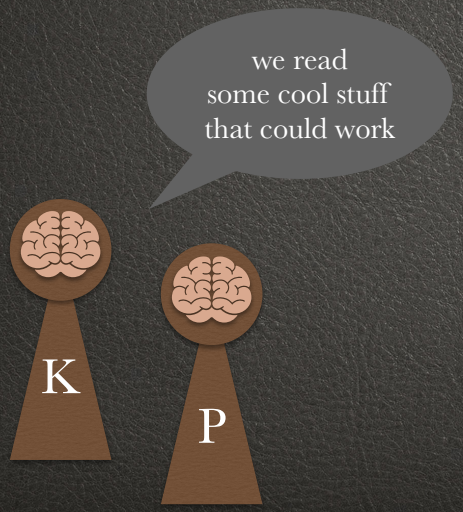
😸 Super fast even for large graphs

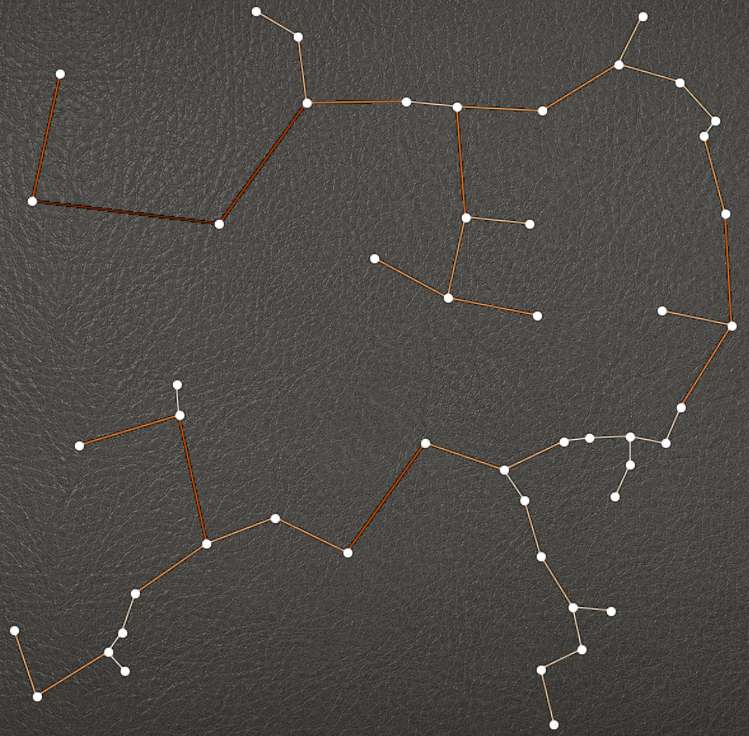
No guarantee of optimality 😿

WHAT ELSE CAN WE DO THAT IS FAST TO COMPUTE?



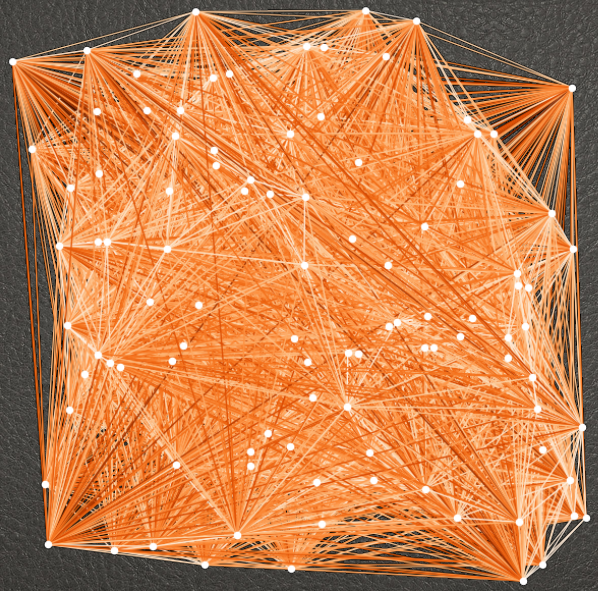
WHAT ELSE CAN WE DO THAT IS FAST TO COMPUTE?





That's not a cycle, fellas 😿

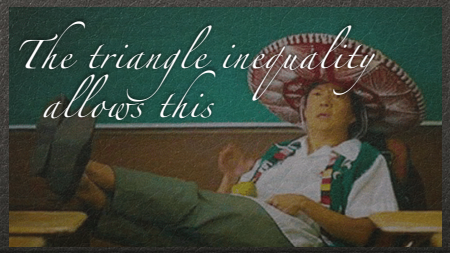
Minimum spanning tree



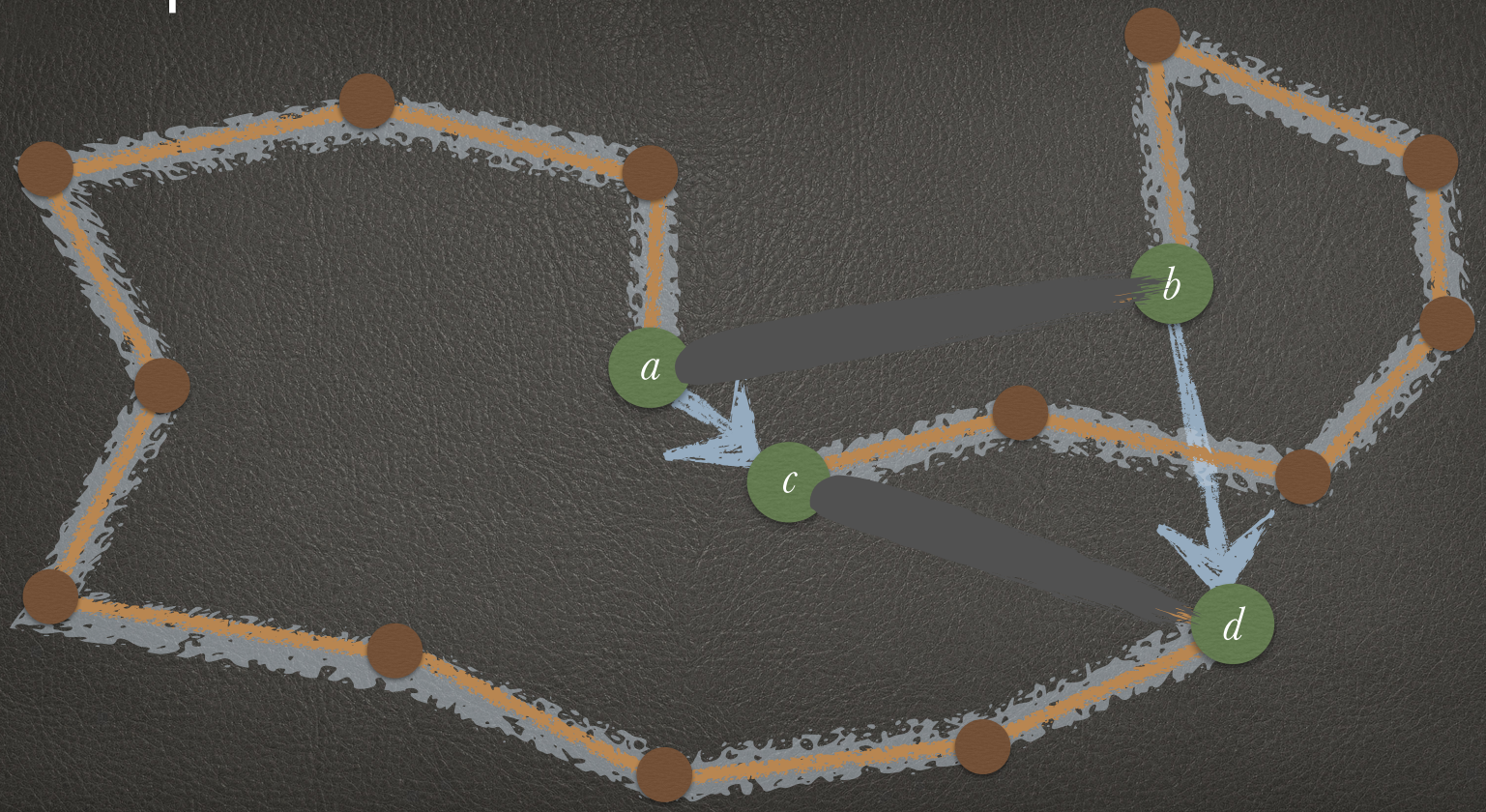
😺 This is quick to build and we can traverse it back and forth with a cost at most twice the optimum

That's a cycle, yes, but it is still infeasible 😿

Shortcuts



2-opt



😸 That could be cheaper than before

This was obviously never an MST, but just bear with me

Local search

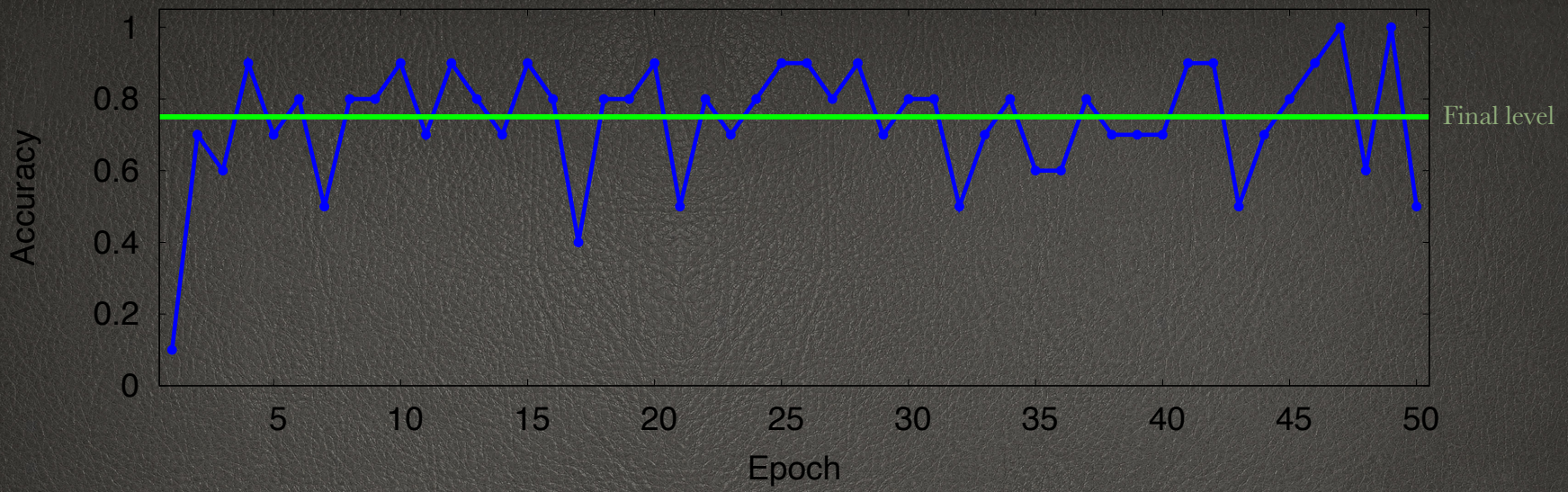
😸 It is feasible now and quick to compute

Those shortcuts are not all that short 😿

Deep learning

- Create a balanced *training set* **X** with *good* and bad *routes*
- Assign *labels* **y** *"costs less than what we are willing to spend"*
- Add an **input layer** with *n* inputs
- Add more *hidden layers* if you'd like
- Add an **output layer** with *one* output (sigmoidal)
- **Fit** that **X** to that **y**

Assume your graph will not change a lot over time



Epoch: execute an *optimizer*,
processing the data by *batches*,
updating *weights* after each batch

Testing

- Create a balanced *test set* **X***, again with *good* and bad *routes*
- Assign *labels* **y***
- **Evaluate** the model with **X*** and **y***

```
loss 0.479
```
(we want this to be low)

```
accuracy 0.733
```
(we want this to be high)

Prediction

```
for BRW  wanted True  got True  (raw: 0.8186)
for RW0  wanted False got False (raw: 0.3016)
for RW1  wanted False got True  (raw: 0.6631)
for RW2  wanted True  got True  (raw: 0.7742)
for RW3  wanted True  got True  (raw: 0.8186)
for RW4  wanted True  got True  (raw: 0.7611)
for MST  wanted False got True  (raw: 0.6497)
for SA   wanted True  got True  (raw: 0.6301)
```

that kinda works

https://tinyurl.com/tspdemocode

Just write

satuelisa

in your preferred search engine

if you need to get in touch