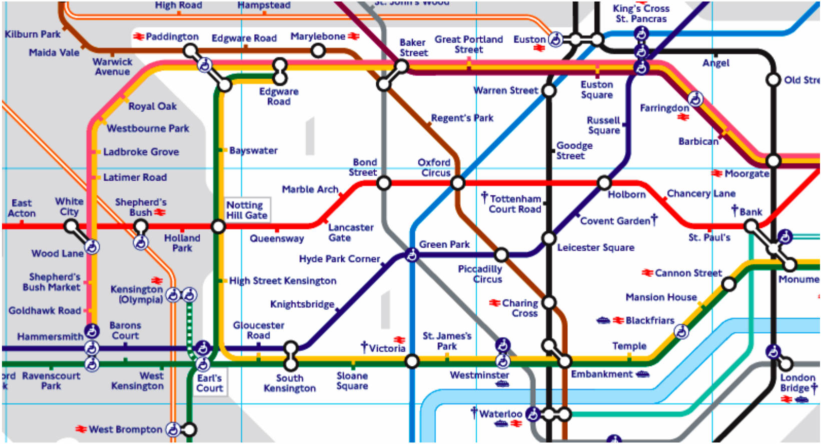
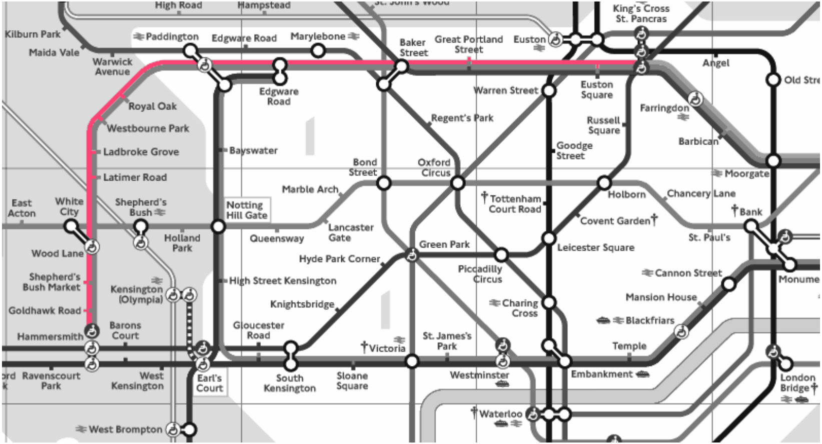
Online Sabotaged Shortest Path

Manfred K. Warmuth Wouter M. Koolen **Dmitry Adamskiy**

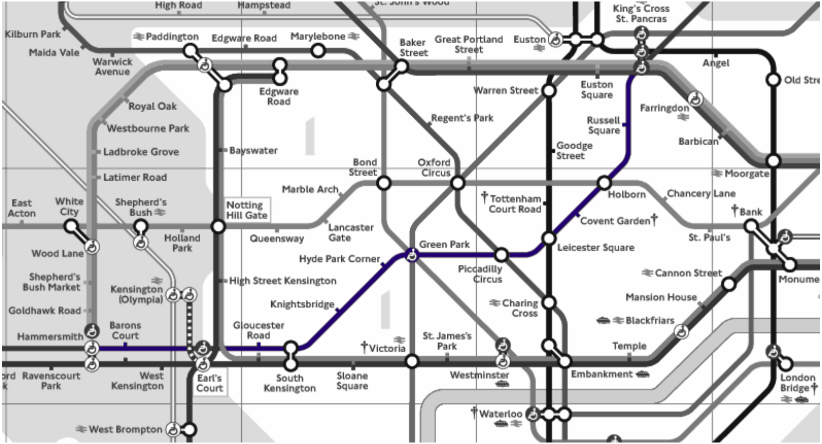
Online shortest path



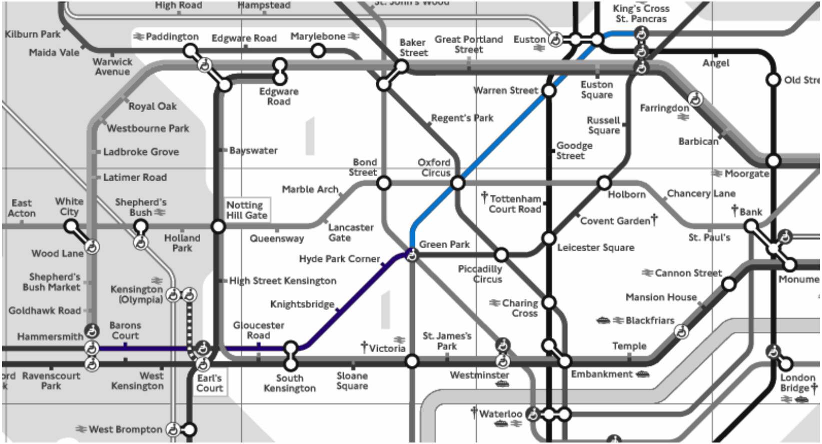
Online shortest path



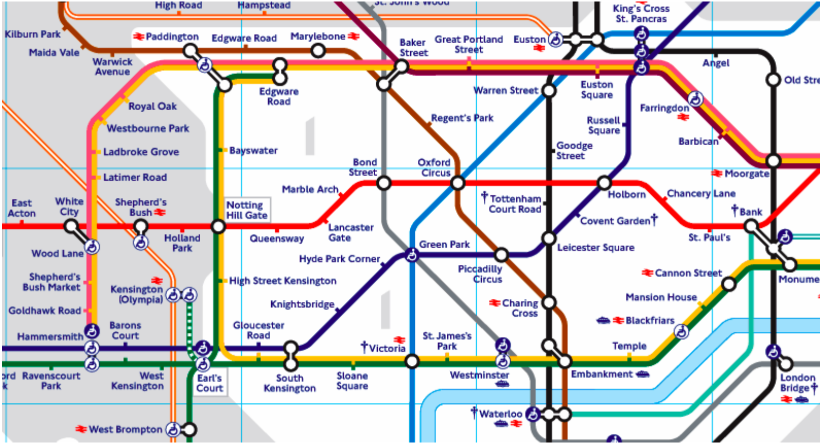
Online shortest path



Online shortest path



Online shortest path

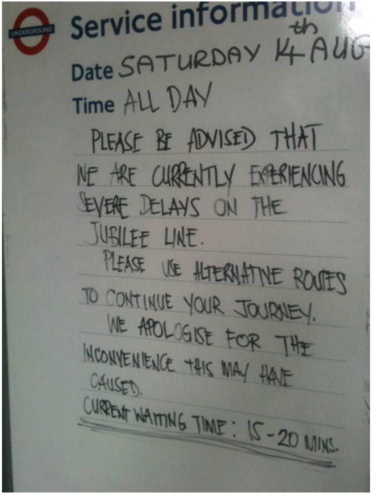


Goal: close to best path in hindsight

Solution: Component Hedge, Mirror Descent, FTPL

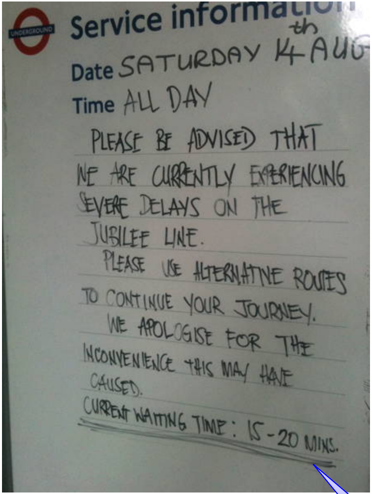
Delays, engineering works and strikes!

Adversarial losses. . .



Delays, engineering works and strikes!

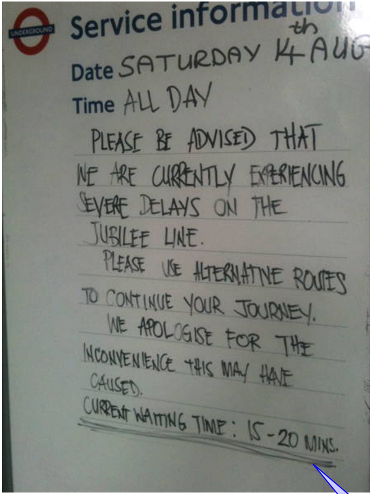
Adversarial losses. . .



"Good service on all other London Underground lines"

Delays, engineering works and strikes!

Adversarial losses. . .

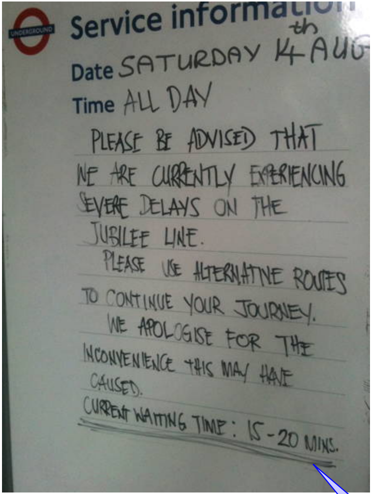


. . . and some paths are blocked

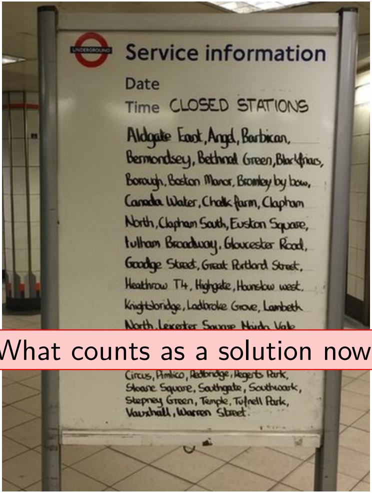
"Good service on all other London Underground lines"

Delays, engineering works and strikes!

Adversarial losses...



...and some paths are blocked



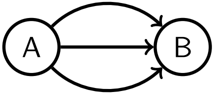
What counts as a solution now?

"Good service on all other London Underground lines"

Previous work: policy regret

Compete with policy for choosing alternatives to blocked paths...

- In fully adversarial setting it is computationally hard already for experts [Kanade and Steinke, 2014]:



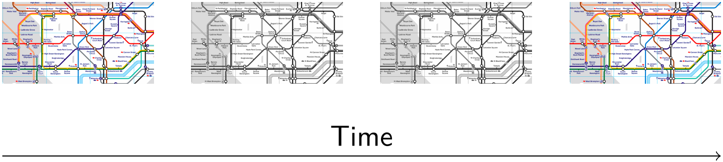
- If sabotages are stochastic and losses are decoupled from them, then efficient algorithms exist [Neu and Valko, 2014]

Proposed notion of regret

We seek a natural notion of regret that avoids the hardness.

Get back to basics [Freund et al., 1997] and compete with the path only on the rounds when it is awake.

$$\text{Regret}(\text{Path}) = \sum_{\substack{\text{rounds when path}\\ \text{is awake}}} \Big(\text{loss}(\text{Learner}) - \text{loss}(\text{Path})\Big)$$



The Open Problem

Is there an efficient algorithm for our regret?

- Less expressive than policies
- Historically the first notion of sleeping
- Efficient algorithm for expert setting
- Naive, grossly inefficient algorithm gets

$$\text{Regret}(\text{Path}) \leq \text{Diameter}\sqrt{T \log |\text{Paths}|}$$