Tïdtabelläggning med deep reinforcement learning

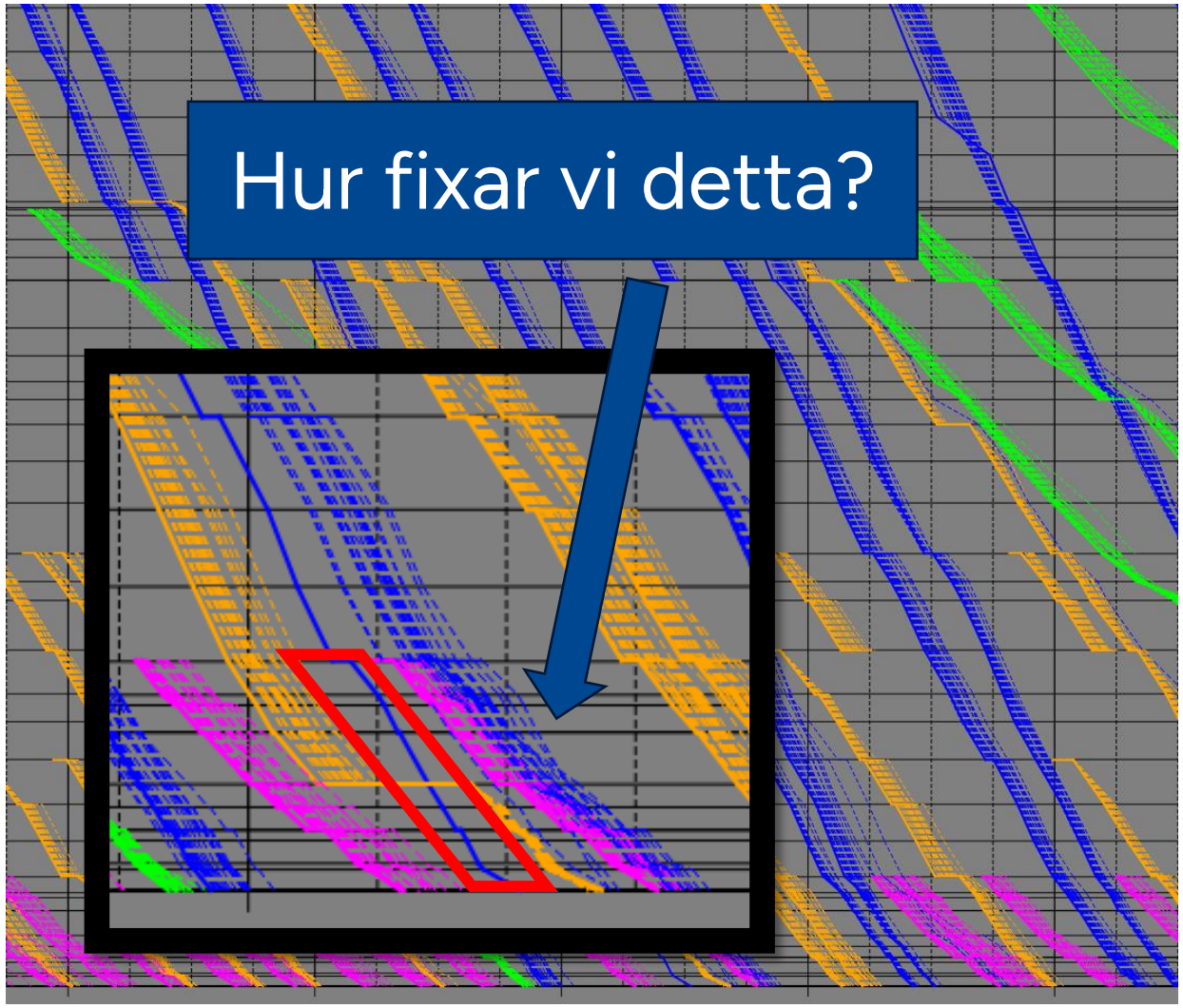
Johan Högdahl[1], Zhenliang Ma[1], Leizhen Wang[2]

[1] Avdelningen för Transportplanering, KTH

[2] Monash University, Department of Data Science and AI, Australia

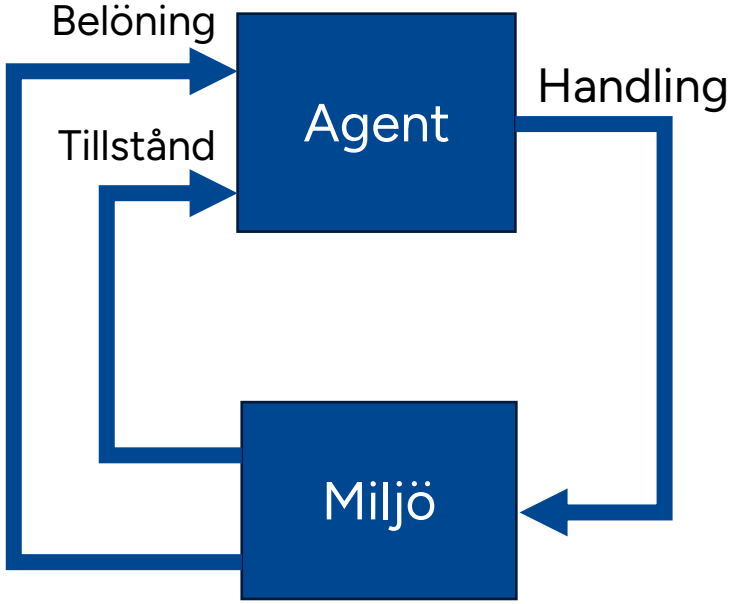
Bakgrund

- Robusta tidtabeller har ibland lokalt bristande robusthet
- **Lokalt bristande robusthet:** Ett tåg hamnar med hög sannolighet utanför sin planerade tidskanal.
- Kan vi använda en datadriven metod eller maskininlärning för att lösa lokala robusthetsbrister?
- Reinforcement learning?



Reinforcement learning (RL)

- Paradigm inom maskininlärning
- Metod för att lära en agent att lösa specifika problem
- En agent verkar inom en miljö och försöker maximera sin långsiktiga belöning
- Agenten vet miljöns nuvarande tillstånd och väljer en handling utifrån detta
- Agenten får en belöning och ett nytt tillstånd

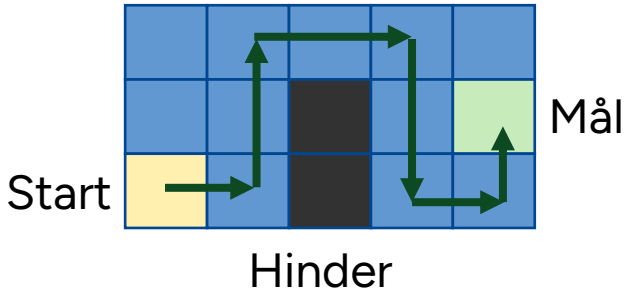


Reinforcement learning

- RL kan användas på problem som är Markov beslutsprocesser (MDP)
- Markov beslutsprocesser:
  - Belöning och nästa tillstånd beror endast på det nuvarande tillståndet och den valda handlingen
- Episod: från start till mål

Exempel: Labyrint

- Tillstånd: Position
- Handlingar: Upp, ner, höger, vänster
- Belöning: -1 för varje steg



RL och tidtabeller

Antaganden

- Givet uppehållsmönster + begr. GT-tillägg

Tillståndsvektor

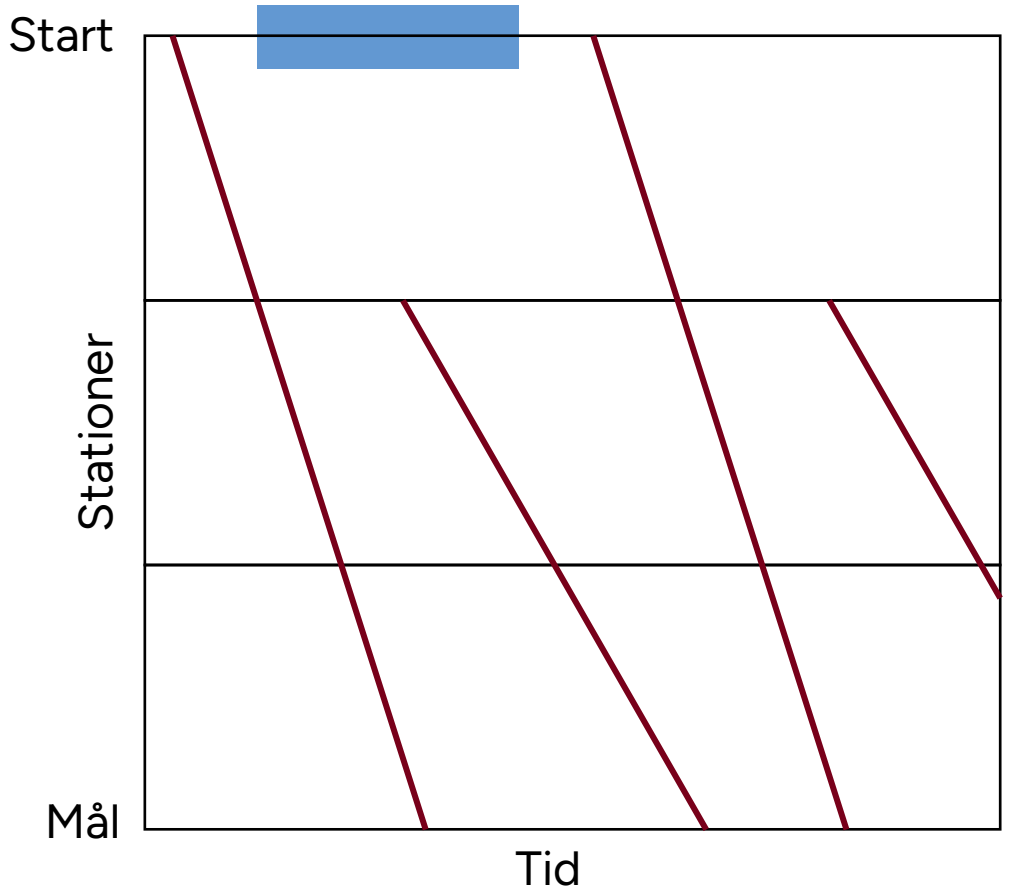
- Station, tidigaste avgångstid, sista avgångstid, planerade tider närliggande tåg

Handling

- Avgångstid inom intervall (kontinuerlig)

Belöning:

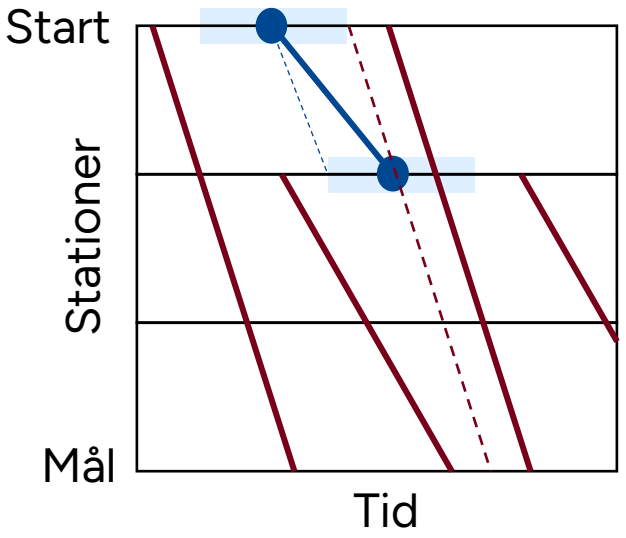
- Tillåten handling: liten negativ belöning
- Icke tillåten handling: större negativ belöning



Kontroll av tillåtenhet

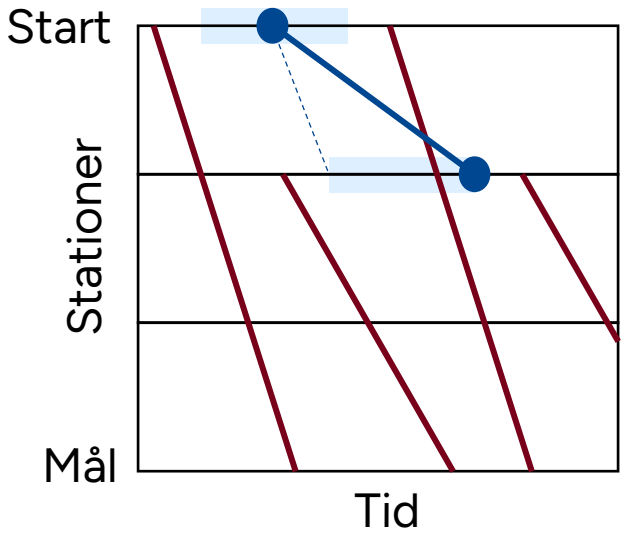
Headway

- Beräkna headway till närmaste tåg
- Mjukt villkor, mindre överträdelser kan tillåtas under träning

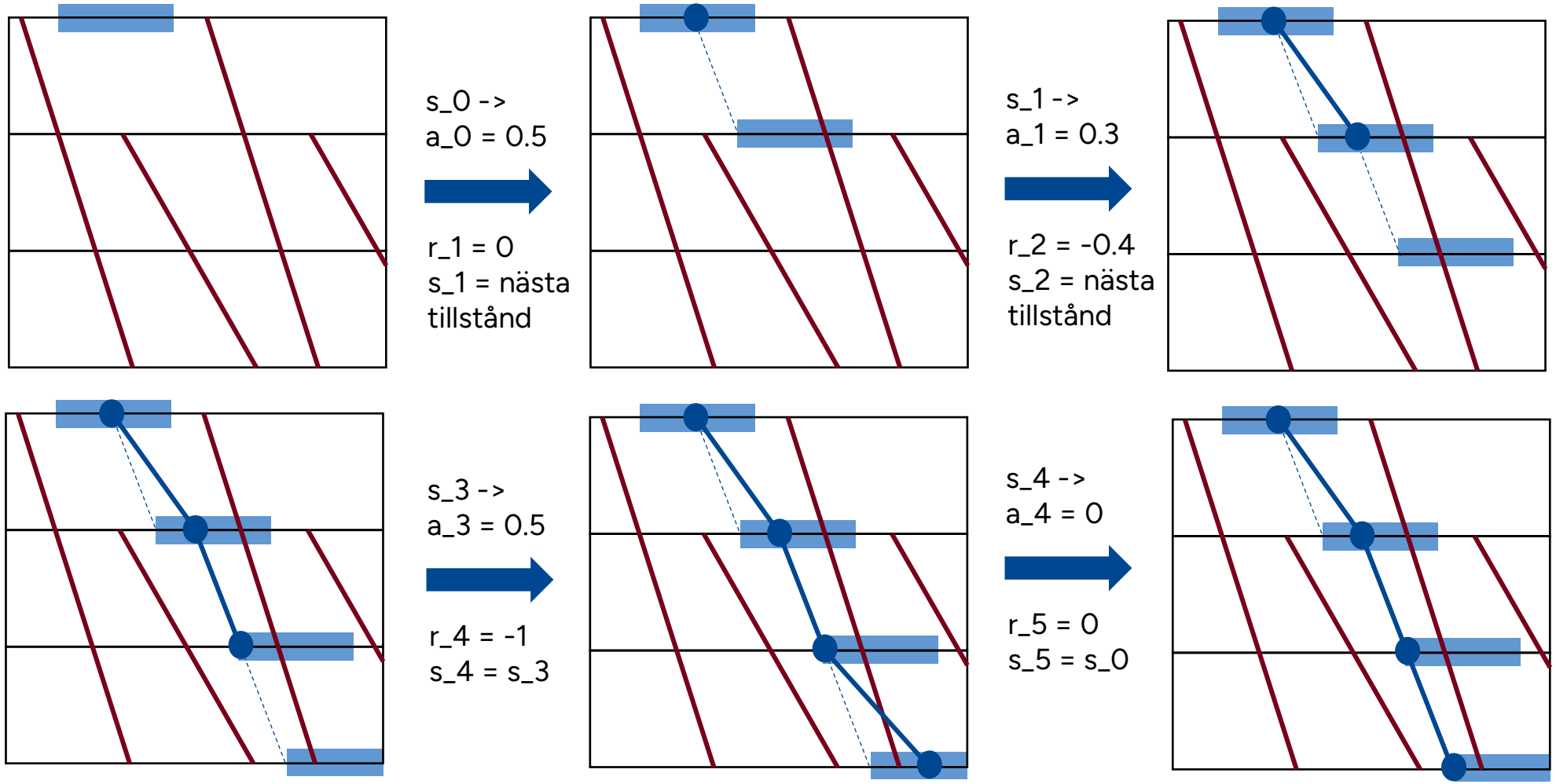


Tågens ordning

- Kontrollera att tågens ordning är tillåten.
- Hårt villkor, straffas hårt och agenten måste göra om handlingen.

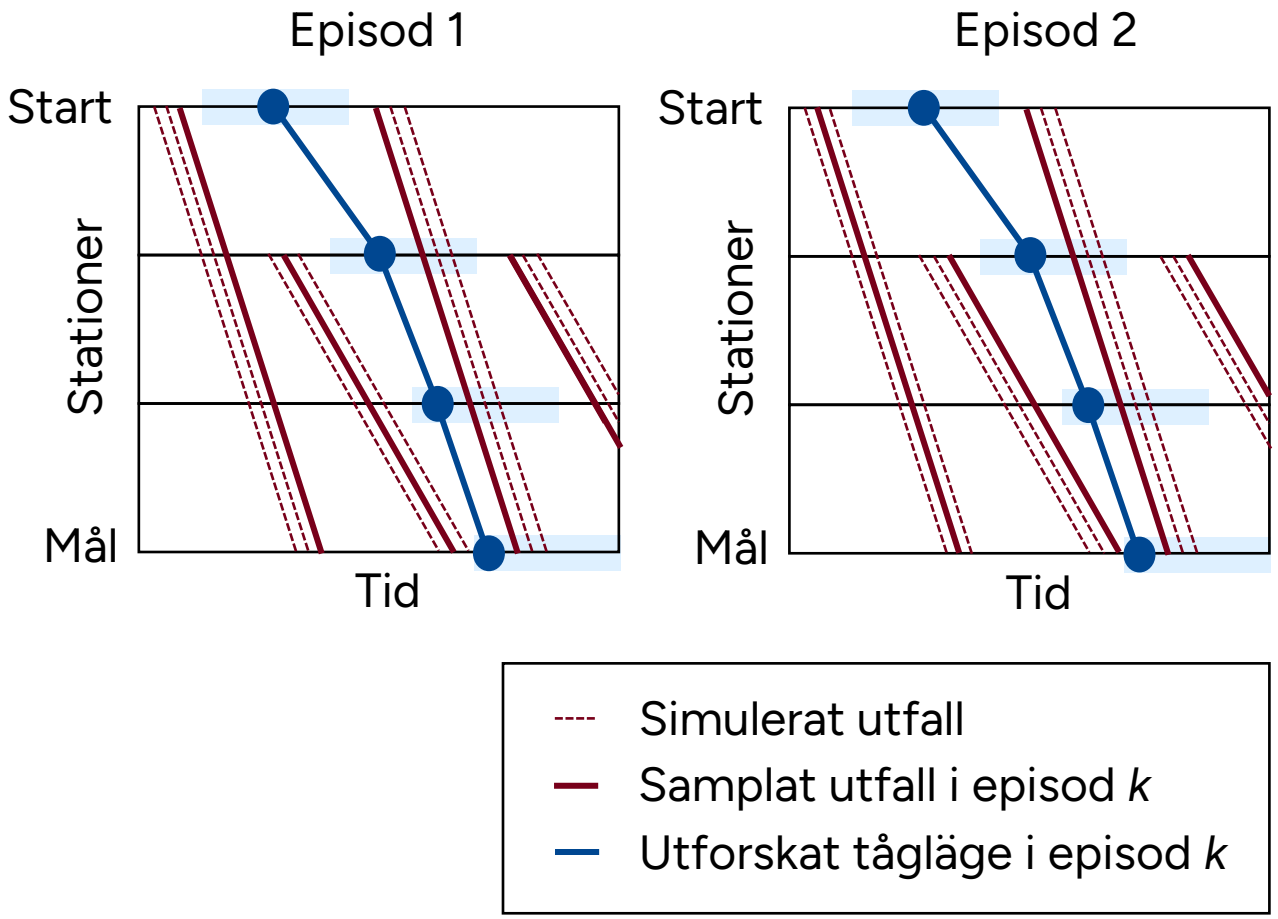


Exempel: 1 episod



Robusta tidtabeller genom RL

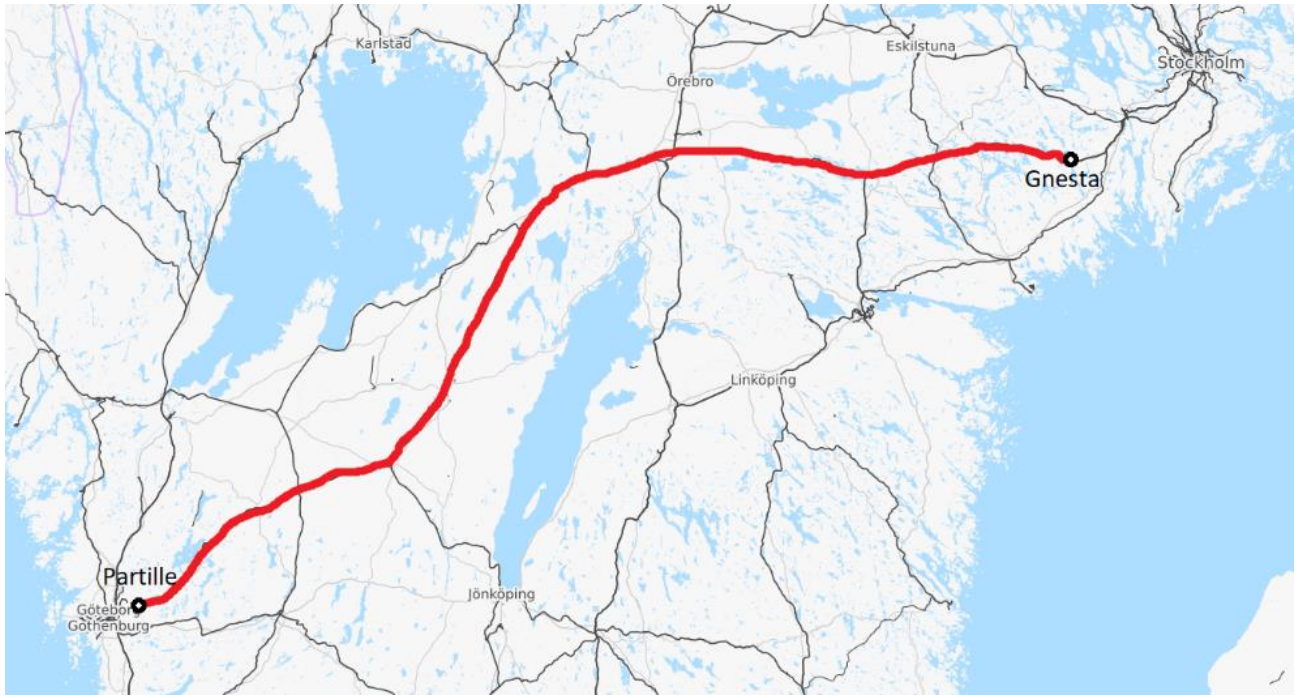
- Robust tågläge med stokastisk miljö
- Träningen sker i episoder
- 1 episod: Genererar ett tågläge från start till mål. Max 500 steg.
- I varje episod samplar vi ett utfall från Railsys-simulering



Preliminära resultat

- Västra stambanan
- Trafik i södergående riktning, Gn -> P
- Tidsfönster första station: 120 s
- Max GT-tillägg: 30 % eller 100 %
- Ingen sampling av simulerade utfall
- Minimera restid

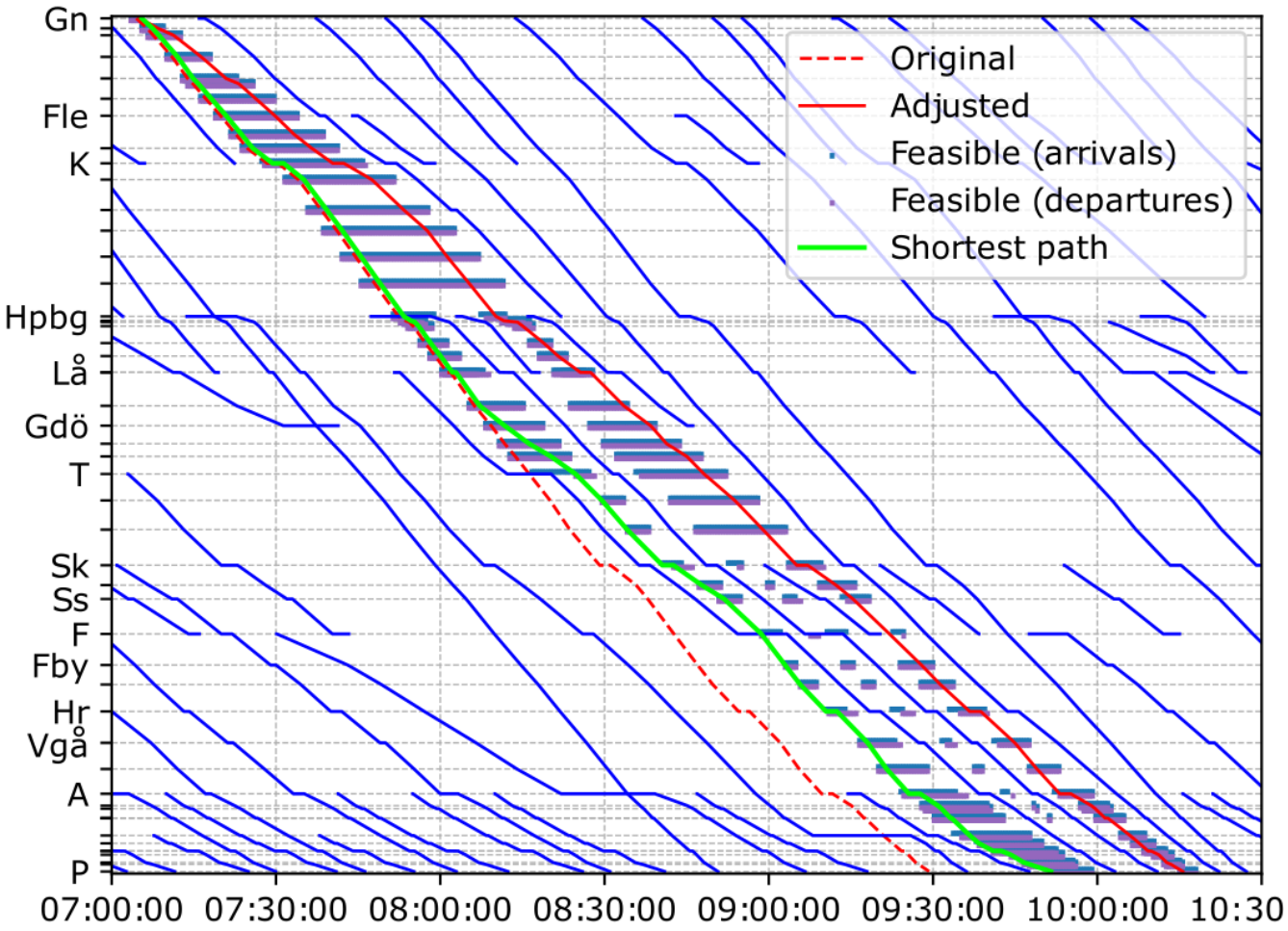
- RL-algoritm:
  - Soft-actor critic (SAC)
  - Prioritized replay buffer



(Source: openrailwaymap.org)

Preliminära resultat I

- Scenario: Agenten bestämmer avgångstid för varje station + max GT-tillägg: 100 %
- Rektanglarna visar ungefärlig kapacitet
- Sträckad röd linje är ursprungligt planerat tågläge
- Grön linje visar snabbaste tågläget inom den approximativt tillgängliga kapaciteten
- Heldragen röd linje visar bästa tågläget med RL
- Modellen presterar dåligt!

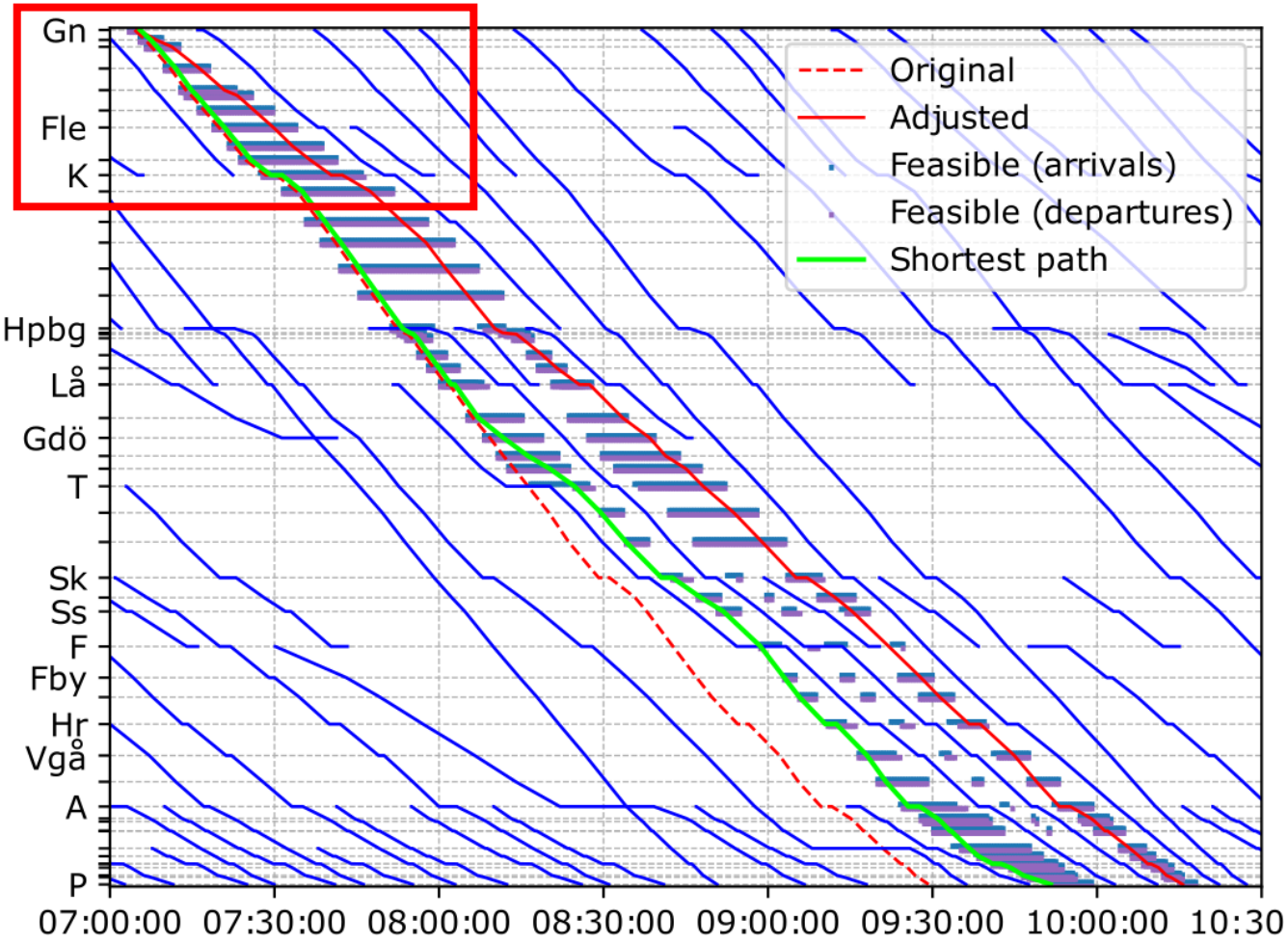


Preliminära resultat II

- Strategi: Hoppa över beslut vid ”onödiga” stationer, max 30 % GT-tillägg

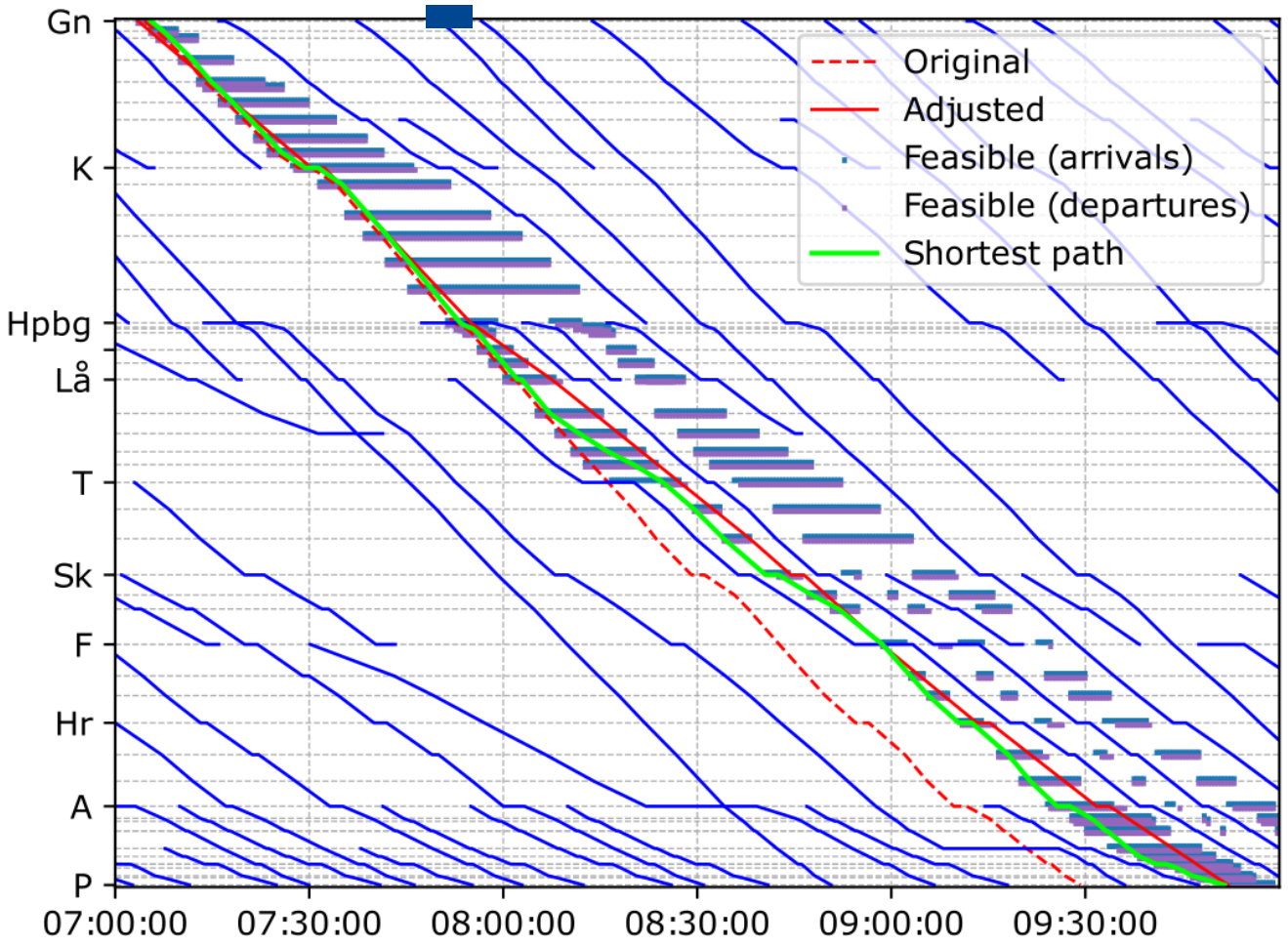
Preliminära resultat I

- Scenario: Agenten bestämmer avgångstid för varje station
- Rektanglarna visar ungefärlig kapacitet
- Sträckad röd linje är ursprungligt planerat tågläge
- Grön linje visar snabbaste tågläget inom den approximativt tillgängliga kapaciteten
- Heldragen röd linje visar bästa tågläget med RL
- Modellen presterar dåligt!



Preliminära resultat II

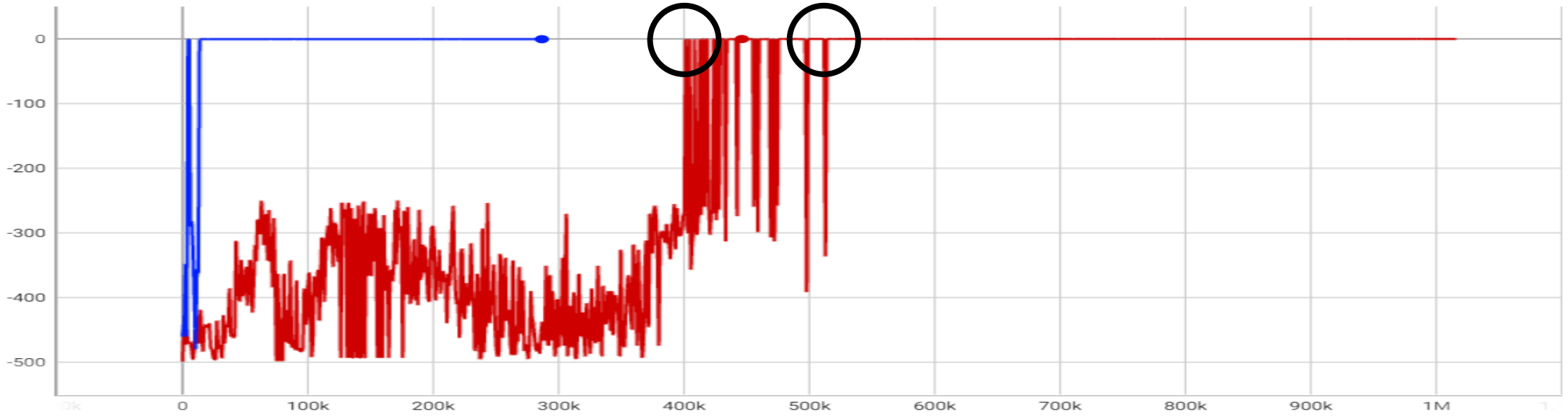
- Strategi: Hoppa över beslut vid ”onödiga” stationer , max 30 % GT-tillägg
- Figuren visar bästa tågläget som modellen genererade
- Bättre tågläge än i första försöket



Preliminära resultat III

- Strategi: Hoppa över beslut vid ”onödiga” stationer + slimmad tillståndsvektor (station, avgångstid från föregående station)
- Inlärningskurva över ca 1 000 000 steg

- Ca 2,5 h träning till modellen lärde sig hitta tåglägen
- 3 h träning till stabil modell

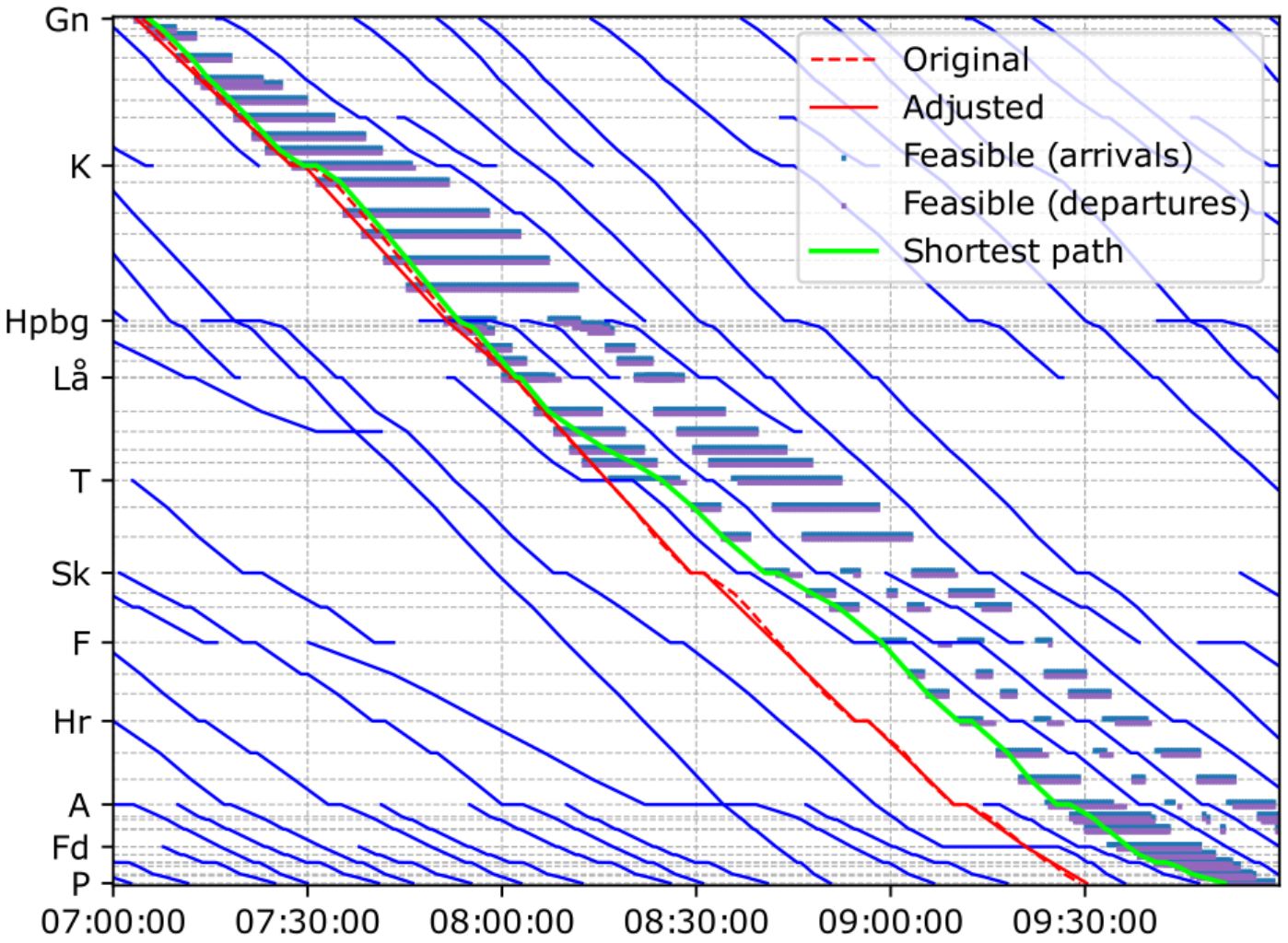


Preliminära resultat III

- Slimmade modeller: Finner bättre tågläge
- Föreslagna modellen: Lär sig snabbt

Preliminära resultat III

- Strategi: Hoppa över beslut vid ”onödiga” stationer + slimmad tillståndsvektor (station, avgångstid från föregående station)
- Figuren visar bästa tågläget som modellen genererade
- Mycket nära att minimera restiden (relativt gap ca 0.03 %)



Slutsatser

- Tidtabellsproblem kan formuleras som MDP
- 1D händelserum kan användas för att bestämma antingen uppehållstid eller gångtidstillägg
- RL applicerbart på realistiskt stora tidtabeller
- Slimmad tillståndsvektor: RL kan producera tåglägen med korta restider
- Mer information i tillståndsvektorn:
  - Agenten lär sig snabbare
  - Riskerar att hamna i lokalt maximum

Fortsatt arbete

- Lösningskvalitet vs mängd träning
  - Utvärdera alternativa formuleringar/algoritmer
  - Utveckla bättre utforskningsstrategier
- Utvärdera metoden för robusthet
- Fler frihetsgrader
  - Förbigångar/uppehåll
  - Stationsspår
- Generaliserbarhet