

Strojové učenie II

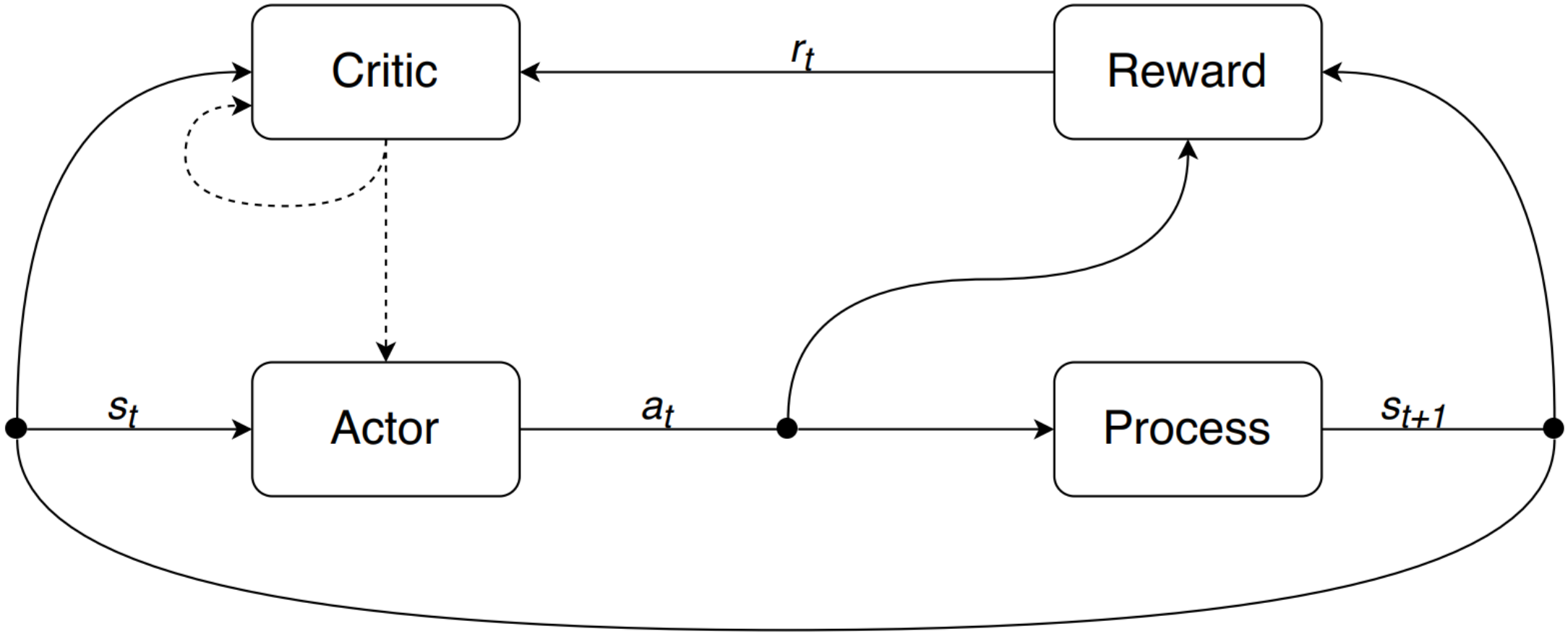
prednáška 8 – Nové prístupy v učení posilňovaním

Ing. Ján Magyar, PhD.
Katedra kybernetiky a umelej inteligencie
Technická univerzita v Košiciach
2021/2022 letný semester

Actor-critic metódy

- kombinácia dvoch modelov
  - actor – učená politika: $\pi(s, a; \theta)$
  - critic – učená hodnotová funkcia: $\hat{q}(s, a; \boldsymbol{w})$
- critic aproximuje hodnotovú funkciu vzhľadom na získané skúsenosti
- aproximácia sa použije na aktualizáciu politiky actora
- konvergencia je zabezpečená cez malý učiaci parameter – zmena hodnotovej aproximácie spôsobuje iba malú zmenu politiky

Trénovanie actor-critic architektúr



Aktualizačné pravidlá pre AC architektúry

- critic

$$\delta w = \beta(r_{t+1} + \gamma \hat{q}_w(s_{t+1}, a_{t+1}) - \hat{q}_w(s_t, a_t))\nabla_w \hat{q}_w(s_t, a_t)$$

- actor

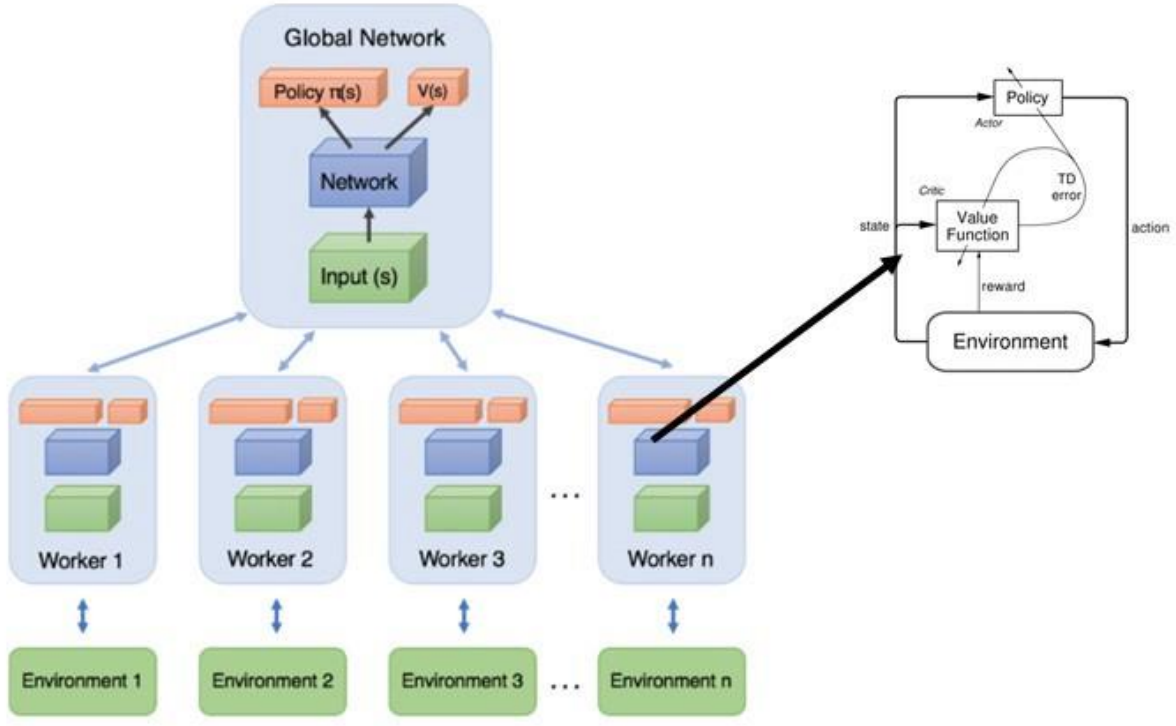
$$\delta\theta = \alpha\nabla_\theta(\log \pi_\theta(a_t|s_t))\hat{q}_w(s_t, a_t)$$

- advantage function

$$A(s_t, a_t) = Q(s_t, a_t) - v(s_t)$$
$$A(s_t, a_t) = r_{t+1} + \gamma v(s_{t+1}) - v(s_t)$$

Asynchronous Advantage Actor-Critic (A3C)

- kritik sa učí advantage funkciu
- paralelné trénovanie
- viacero agentov, každý pracuje s vlastnou kópiou prostredia
- aktualizuje sa globálna sieť
- efektívna explorácia stavového priestoru



- *Mnih et al.: Asynchronous Methods for Deep Reinforcement Learning*

Trénovanie A3C

// Assume global shared parameter vectors θ *and* θ_v *and global shared counter* $T = 0$
// Assume thread-specific parameter vectors θ' *and* θ'_v
Initialize thread step counter $t \leftarrow 1$
repeat
  Reset gradients: $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.
  Synchronize thread-specific parameters $\theta' = \theta$ and $\theta'_v = \theta_v$
  $t_{start} = t$
  Get state s_t
  repeat
    Perform a_t according to policy $\pi(a_t|s_t;\theta')$
    Receive reward r_t and new state s_{t+1}
    $t \leftarrow t + 1$
    $T \leftarrow T + 1$
  until terminal s_t **or** $t - t_{start} == t_{max}$
  $R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta'_v) & \text{for non-terminal } s_t \text{// Bootstrap from last state} \end{cases}$
  for $i \in \{t-1, \ldots, t_{start}\}$ **do**
    $R \leftarrow r_i + \gamma R$
    Accumulate gradients wrt θ': $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i;\theta')(R - V(s_i;\theta'_v))$
    Accumulate gradients wrt θ'_v: $d\theta_v \leftarrow d\theta_v + \partial \left(R - V(s_i;\theta'_v)\right)^2 / \partial \theta'_v$
  end for
  Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.
until $T > T_{max}$

Advantage Actor-Critic (A2C)

- jedna globálna sieť, viaceré inštancie agentov
- agentov trénujeme najprv nezávisle a potom sa vypočíta agregovaný výsledok
- rovnaká alebo lepšia konvergencia ako A3C
- efektívna práca na GPU
- väčšie trénovacie dávky

Rekurentné učenie posilňovaním

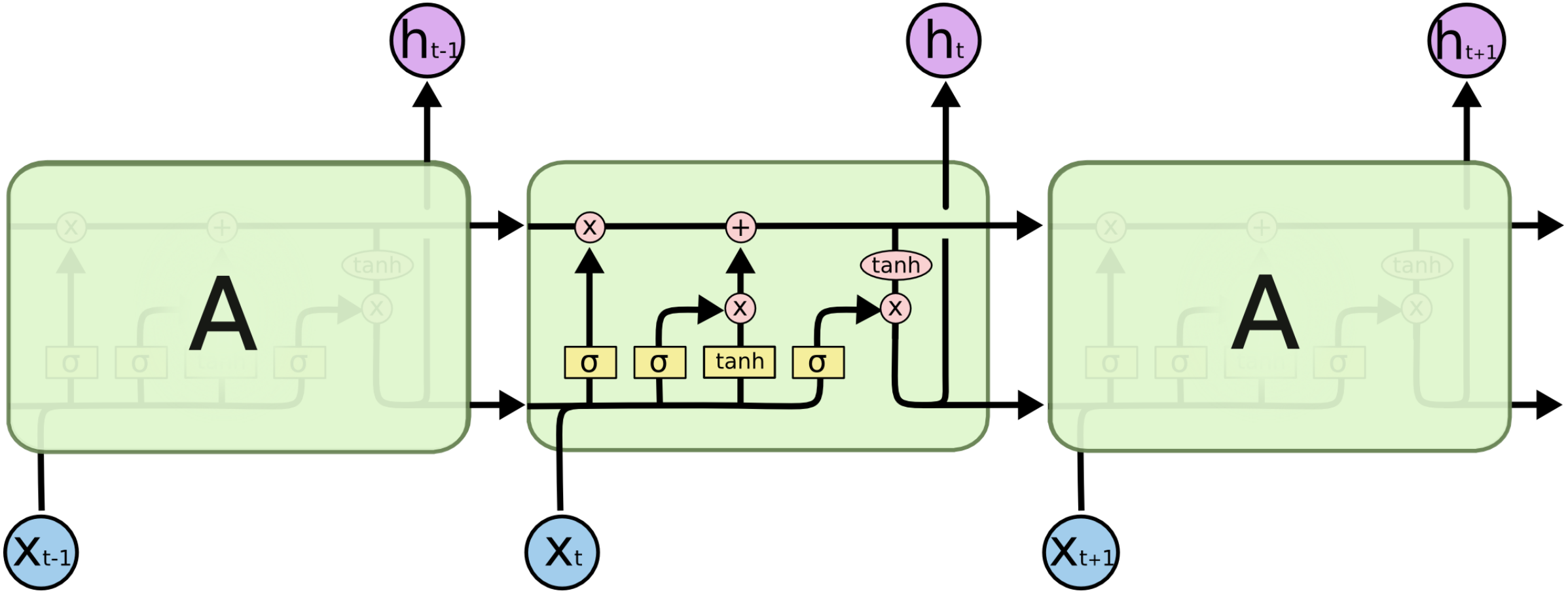
- v MDP nasledujúci stav závisí iba od aktuálneho stavu a akcie
- v skutočnosti aktuálny stav prostredia nám často nedá postačujúce informácie
- formálne čiastočne pozorovateľný MDP
- prvé riešenia: použitie *n* posledných stavov ako vstup
- podpora časových radov cez rekurentné učenie

Deep Recurrent Q-Network

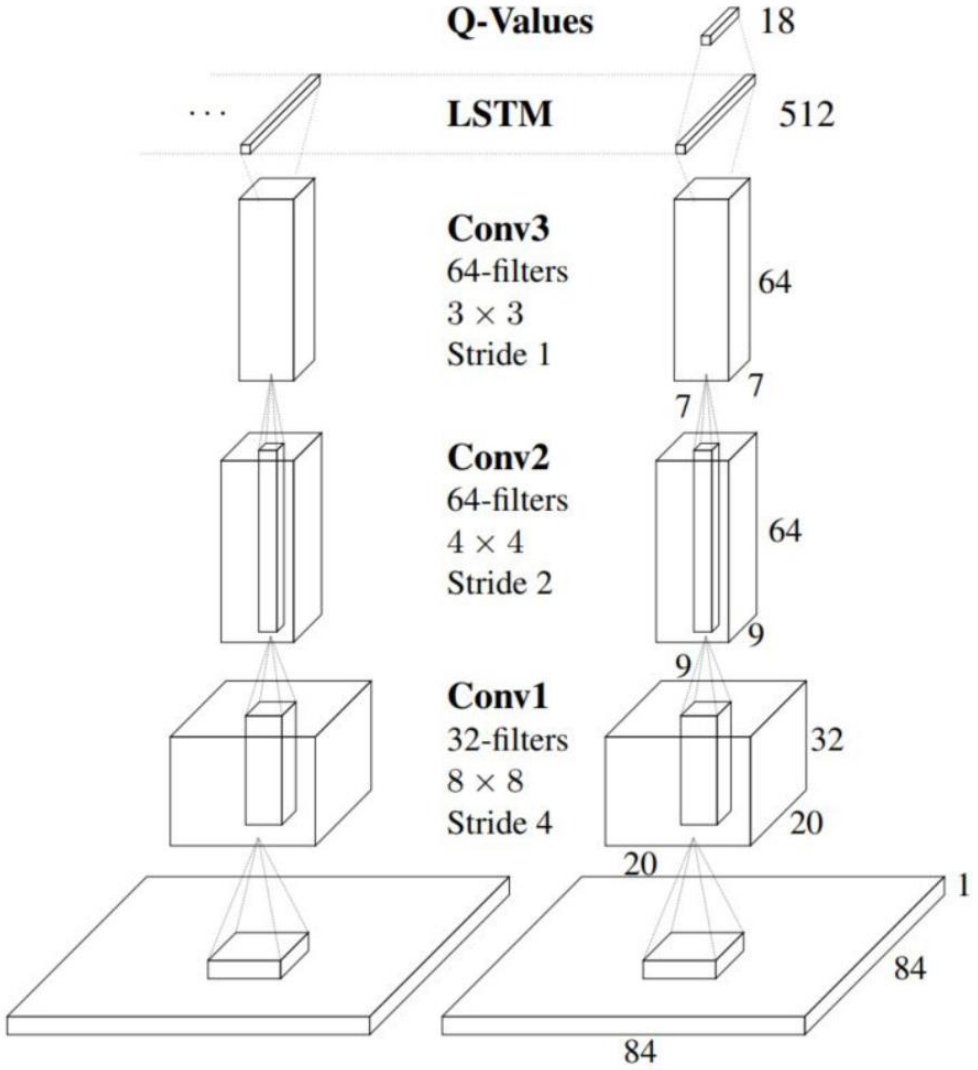
- používa rekurentnú vrstvu – LSTM – na zapamätanie predošlých stavov
- tri fázy
  - konvolúcia – predspracovanie stavov
  - rekurentná vrstva – pamätanie časovej zmeny
  - plne spojené vrstvy – predikcia Q-hodnôt
- zvyčajne lepší výkon ako DQN

- *Hausknecht and Stone: Deep Recurrent Q-learning for Partially Observable MDPs*

LSTM



Architektúra DRQN



Prehrávanie skúseností v DRQN

- náhodný výber nie je postačujúci
- boostrapped sequential updates
  - model si zapamätá celé epizódy
  - prehrávajú sa celé epizódy náhodne
  - skrytý stav LSTM sa používa počas celého prehrávania
  - lepšie pre trénovanie LSTM
- bootstrapped random updates
  - epizódy sú vyberané náhodne, ale prehrá sa iba časť z nich
  - prehrávanie sa začína z náhodne vybraného stavu
  - skrytý stav LSTM sa vynuluje pred každou epizódou
  - pokryje väčšiu časť stavového priestoru

Cielená explorácia

- explorácia bežne funguje náhodným štýlom
- neefektívne vyhľadávanie
- v ideálnom prípade by sme hľadali lepšie riešenie viac v stavoch, ktoré nepoznáme až tak dobre
- ε-decay – začať s vysokou hodnotou ε, následne znižovať hodnotu ako dostávame presnejšie a presnejšie odhady
- cielená explorácia
  - nastavovať hodnotu ε na základe skúseností s prostredím
  - counter-based
  - value-difference based

Counter-based explorácia

- agent si zapamätá informácie o samotnom trénovaní
- explorácia je smerovaná k menej navštíveným stavom
- agent udržiava počítadlo *c(s)* pre každý stav (s) – inicializované na 0
- akcia je vybraná kombináciou exploatácie a explorácie

$$eval_c(a) = \alpha \cdot f(a) + \frac{c(s)}{E[c|s,a]}$$

- vždy sa vyberie akcia s najvyššou hodnotou $eval_c(a)$

Value-difference based explorácia

- prístup založený na heuristike:
  - ak je vysoký rozdiel medzi očakávanou a reálnou hodnotou stavu, zvýš ε
  - ak rozdiel medzi očakávanou a reálnou hodnotou stavu je nízky, zmenši ε
- hodnota ε sa aktualizuje na základe rozdielu:

$$\varepsilon_{t+1}(s) = \delta \cdot \frac{1 - e^{\frac{-|\alpha \cdot TD-error|}{\sigma}}}{1 + e^{\frac{-|\alpha \cdot TD-error|}{\sigma}}} + (1 - \delta) \cdot \varepsilon_t(s)$$

σ – inverse sensitivity
δ – step size

Inverse reinforcement learning

- cieľom je nájsť funkciu odmeny k známej (optimálnej) politike
- možný problém je degenerácia
  - ako vybrať správnu funkciu v prípade, že algoritmus nájde ich niekoľko?
- podpora viacparametrových funkcií odmeny
- pomáha pri problémoch, kde nevieme jednoducho popísať cieľ
- nefunguje pri nekonečných stavových priestoroch

- *Ng and Russell: Algorithms for Inverse Reinforcement Learning*

Imitation learning

- cieľom je napodobňovať ľudské správanie pri riešení danej úlohy
- agent sa naučí namapovať pozorovania na akcie
- schopnosť naučiť komplikované správanie bez veľkého zásahu experta
- často sa spolieha na kvalitné senzory
- nie je potrebné navrhnúť presnú funkciu odmeny

Trénovanie v IL

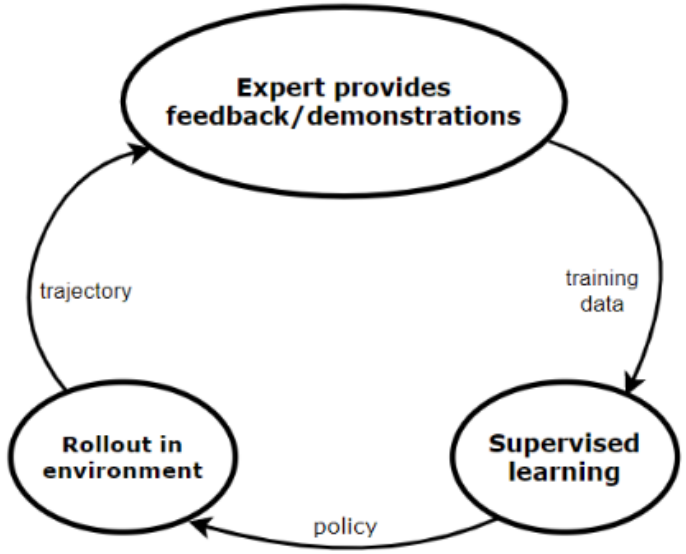
- prostredie stále vnímame ako MDP ale s neznámou funkciou odmeny
- máme k dispozícii demonštrácie od experta – trajektórie $\tau = (s_0, a_0, s_1, a_1, \dots)$
- správanie experta považujeme za optimálnu politiku π^*
- v niektorých prípadoch expert je priamo zahrnutý v trénovaní
- základné prístupy
  - klónovanie správania (behavioural cloning)
  - priame učenie politiky (direct policy learning)

Klónovanie správania

- využíva kontrolované učenie ale v kontexte učenia posilňovaním
- iba v prípade, ak skúsenosti sú nezávisle a rovnomerne rozdelené
- fázy učenia
  1. zbieranie skúseností od experta τ^*
  2. rozdelenie trajektórií na dvojice stav-akcia $(s_0^*, a_0^*), (s_1^*, a_1^*)$, ...
  3. trénovanie politiky π_θ vzhľadom na chybovú funkciu $L(a^*, \pi_\theta(s))$
- chyba sa stupňuje, môže dôjsť ku katastrofálnemu zlyhaniu

Priame učenie politiky

- vyžaduje interaktívneho demonštrátora počas trénovania
- agent sa natrénuje na zozbieraných trajektóriách a spustí sa simulácia, ktorá je hodnotená expertom
- nové skúsenosti sú použité na ďalšie učenie agenta až do konvergencie
- dôležité je použiť všetky predošlé tréningové skúsenosti
  - agregácia dát
  - agregácia politík



Priame učenie politiky

Initial predictor: π_0

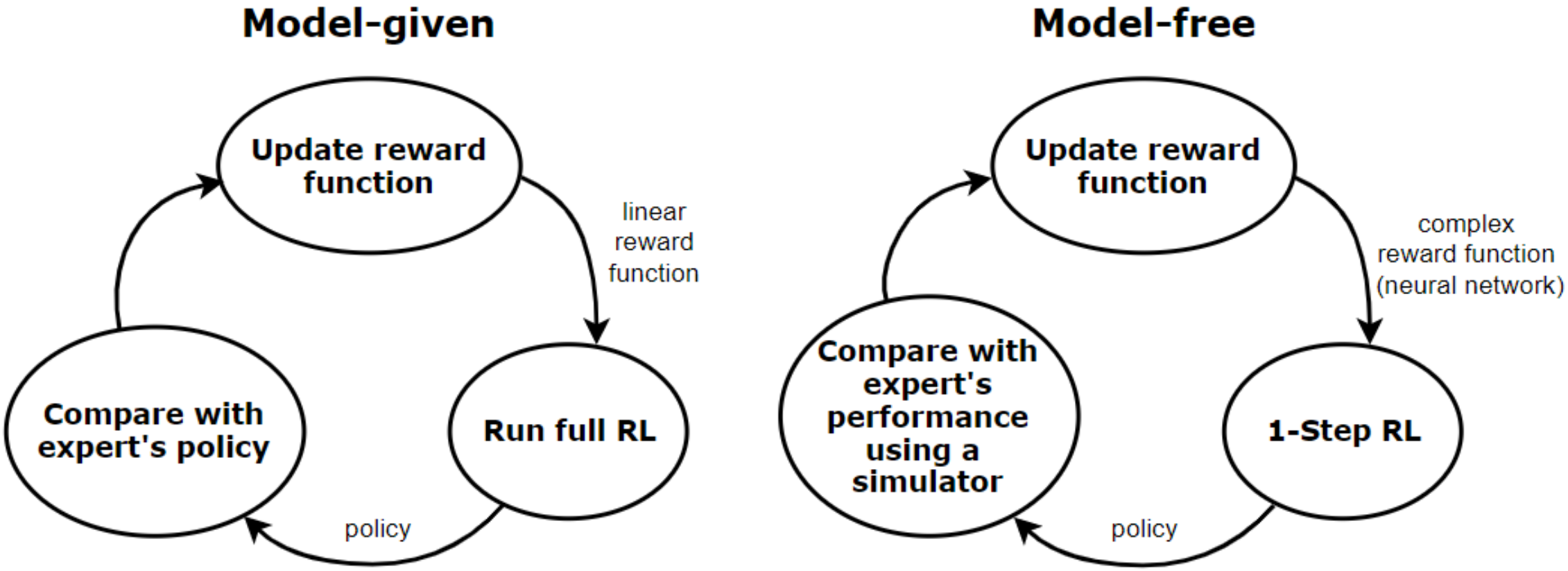
For $m = 1$:

- Collect trajectories τ by rolling out π_{m-1}
- Estimate state distribution P_m using $s \in \tau$
- Collect interactive feedback $\{\pi^*(s) \mid s \in \tau\}$
- Data Aggregation (e.g. Dagger)
  - Train π_m on $P_1 \cup \ldots \cup P_m$
- Policy Aggregation (e.g. SEARN & SMILe)
  - Train π'_m on P_m
  - $\pi_m = \beta\pi'_m + (1-\beta)\pi_{m-1}$

Imitation a inverse RL

1. začíname s množinou demonštrácií od experta a snažíme sa nájsť takú funkciu odmeny, ktorú správanie experta maximalizuje
2. natrénujeme agenta pomocou funkcie odmeny
3. porovnáme politiku agenta s politikou experta
4. aktualizujeme funkciu odmeny a vrátime sa k bodu 2 alebo ukončíme trénovanie

Úloha modelov v IRL

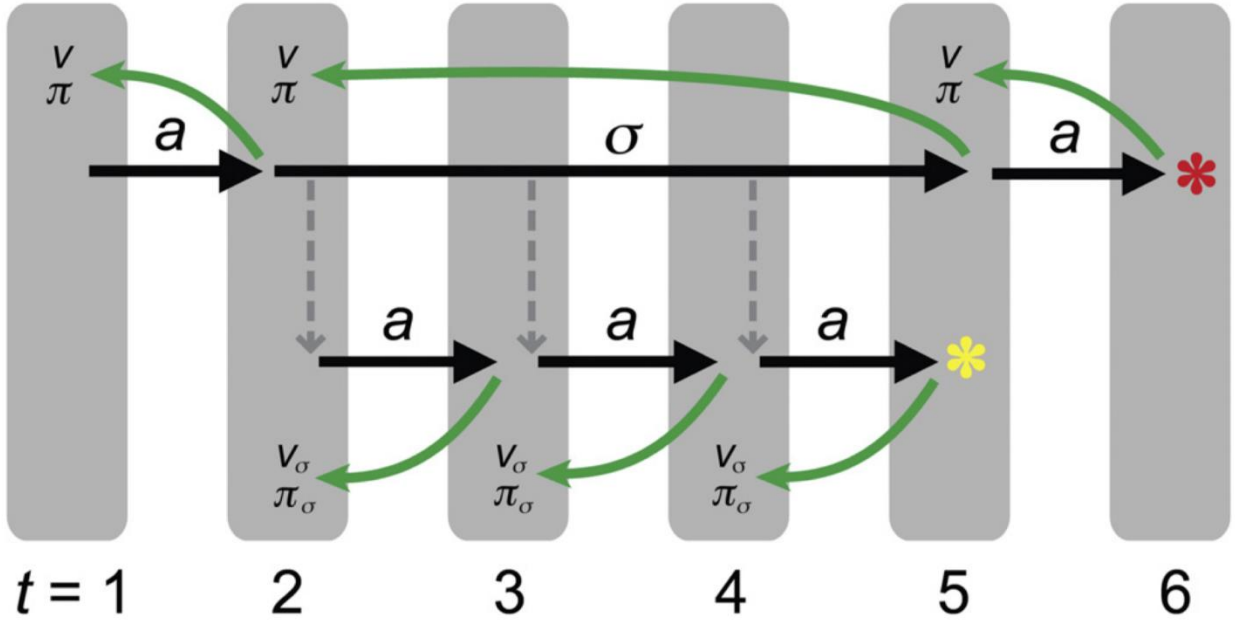


Hierarchical reinforcement learning

- zavedieme rôzne úrovne granularity do riešenia problému
- vychádza z pozorovaní ľudskej psychológie: riešenie problému rozdelíme na riešenie menších podproblémov, ktoré dokážeme vyriešiť tak, že sa spoliehame na „zdravý rozum“
- výhody HRL
  - sample efficiency
  - škálovateľnosť
  - generalizácia
  - abstrakcia
  - hierarchická reprezentácia stavového priestoru

Semi-MDPs

- okrem dostupných akcií definujeme aj makroakcie, ktoré sú postupnosti jednoduchých akcií
- prechod sa definuje ako $p(s', \tau | s, a)$



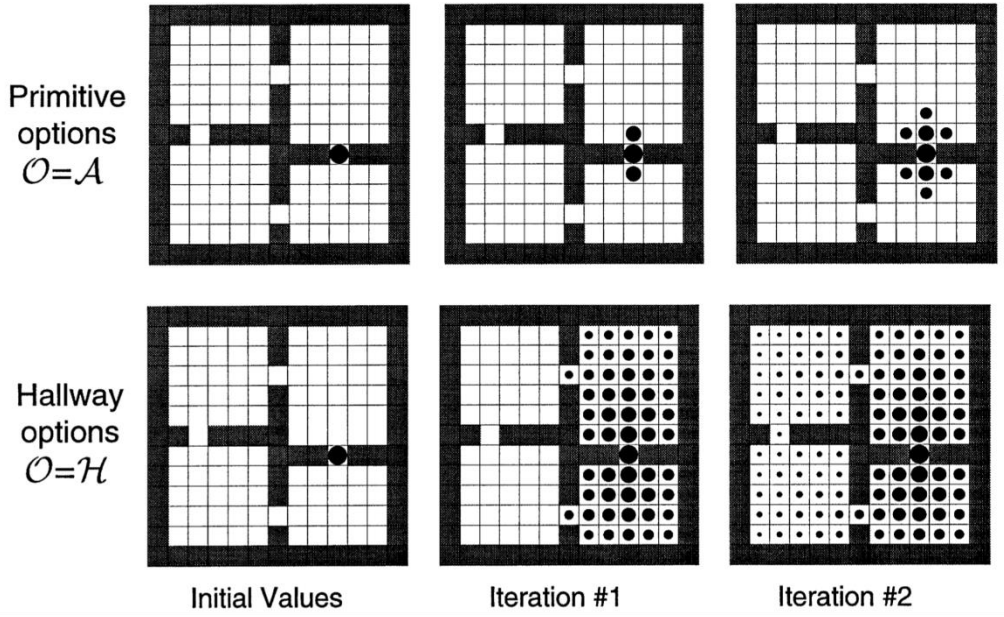
Feudal learning

- manager pridelí task (pod-úlohu) jednotlivým sub-managerom, ktorí sa ich naučia vyriešiť
- princípy
  - information hiding
  - reward hiding
- algoritmus Feudal Q-learning, ktorý funguje iba pre špecifické prípady

- *Dayan and Hinton: Feudal Reinforcement Learning*

Options Framework

- definuje Markovovské opcie $o =< I_o, \pi_o, \beta_o >$
- opcie sú akcie na vyššej úrovni abstrakcie
- algoritmy garantovane konvergujú k optimálnej politike ak priestor akcií obsahuje primitívne akcie a opcie
- naučená politika má dve úrovne:
  - sub-politika
  - politika nad opciami

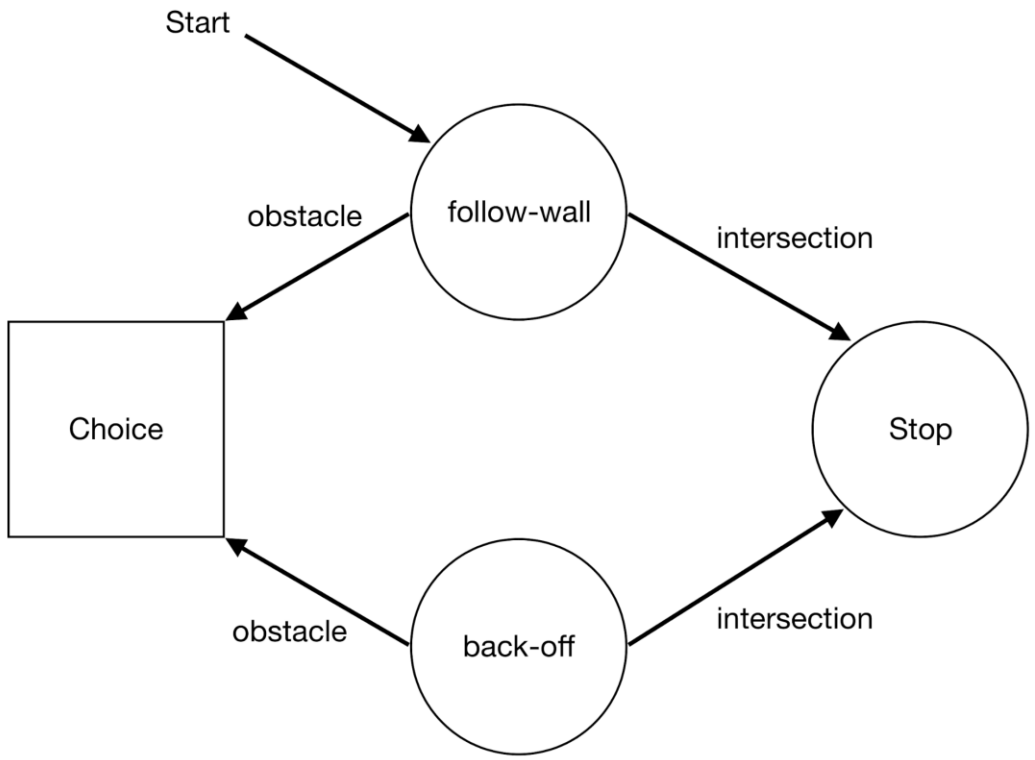


- *Sutton, Precup and Singh: Between MDPs and Semi-MDPS: A Framework for Temporal Abstraction in Reinforcement Learning*

Hierarchical Abstract Machines

- skladajú sa z nedeterministických konečnostavových automatov, kde prechody môžu zavolať automaty nižšej úrovne
- štyri stavy automatu:
  - action state
  - call state
  - choice state
  - stop state

- *Parr and Russell: Reinforcement Learning with Hierarchies of Machines*



MAXQ

- rozšírenie Q-hodnoty: $Q(p, s, a) = Q(s, a) + C(p, s, a)$
- založený na dekompozícii hodnotovej funkcie na kombináciu hodnotových funkcií menších členských MDP
- definícia menších podúloh, kde podúloha je formalizovaná cez
  - terminálny predikát
  - množinu akcií
  - pseudo-odmenu
- naučí sa rekurzívne optimálnu politiku

- *Dietterich: Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition*

Novšie hierarchické algoritmy

- Vezhnevets et al.: FeUdal Networks for Hierarchical RL
- Bacon et al.: The Option-Critic Architecture
- Nachum et al.: HIRO (Data Efficient Hierarchical RL)
- Pang et al.: On Reinforcement Learning for Full-length Game of StarCraft
- Gopalan et al.: Planning with Abstract Markov Decision Processes