PAGE 0: RECONFIGURABLE REINFORCEMENT LEARNING NETWORKS

In general, most online optimization problems can be expressed as fully observable Markov Decision Processes (MDPs) as $\langle S, A, T, R \rangle$ tuples (State S, Action A, Transitions T, Reward R).

In general, we can express behaviour in this domain as a policy $\pi : S \rightarrow R$ **[? looked like this, but would** $\pi : S \rightarrow A$ **make more sense?]**. Particular attention is given to the optimal strategy.

In prior work the issue of tractability and subsequent decomposition have been articulated. In this work the subject of learning and generalizing this decomposition work into a General framework is discussed.

- Ⓐ Theory
  - Introduction (4): reconfigurable RL introduction & overview
  - Mapping (11): a generalized set of mapping & deconstruction operations (parent, child)
  - Convergence (16): parent, child, reward optimization, complexity
  - Worst Case Performance (23): system behaviour with malformed problems
- Ⓑ NN paper
  - Neural Networks (24): RRLN are just feed forward Neural Networks
  - ↪ (N.)

Special topics:

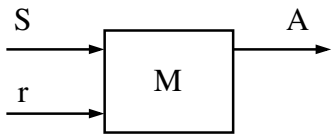
Temporal Difference (A1): how to discover & change time basis/scale

Transitional Learning (A2): how to re-use and generalize transitional models

Financial Systems (A3): how to use with financial systems

Origins (E1-E4): original examples and sketches

Transitional Encoding (E5-E6): Continuous bnns **[?]** & applications



assume a Markov decision process M which can be completely represented as a tuple $M = \langle S, A, T, R, \pi, \tilde{T}, \tilde{R} \rangle$

S – a set of states $s \in S$ which may be experienced by M

A – a set of actions $a \in A$ that may be executed

T – a true transitional probability, $T(s'|a, s)$ expressing the probability of executing an action a in state s before ending up in later state s'.

R – is a reward function which quantifies how desirable a transition $R(s'|a, s)$ is. $R : S \times S \to \mathbb{R}_{\geq 0}$

[I changed $\mathbb{R}^+$ to $\mathbb{R}_{\geq 0}$ because the former is ambiguous with respect to whether or not 0 is included (online research suggests there is no accepted convention) while the latter is unambiguous.]

π – is an action selection policy, ideally chosen to maximize expected reward, an optimal policy is denoted π^*. Typically

$$\pi^*(s) = \arg\max_a \sum_{s'} \underbrace{R(s'|a, s)T(s'|a, s) + \gamma V(s')}_{\text{expected reward}}$$

ENCODING

To encode the expected reward over all states, typically Q-values are kept: $Q(s, a) \sim \sum R(s'|a, s)T(s'|a, s) + \gamma V(s')$ and $Q_{t+1}(s, a) \leftarrow Q_t(s, a) + \alpha \left(R(s'|a, s) - Q_t(s, a) + \gamma \arg\max)_{a'} Q(s', a')\right)$.

In this paper we rely on a method of extracting dynamic Q-values from an encoded transition and reward function $(\tilde{T}, \tilde{R})$. The motivation for this encoding is that it allows mapping the transition function into multiple spaces, and allows the reward function to be altered. The significance of this finding is covered in **???** Price wash **???**.

PAGE 5

1 RECONFIGURATION

Reconfiguring **???** Process M allows some intractable MDPs to be rendered tractable. As an example, a three dimensional foraging experiment with three thousand positions on the x, y, and z axes respectively will consume over three billion memory locations and may be impossible to explore. If this system is broken into three sub problems, each targets a special axis, the only nine thousand memory locations need be consumed. This decreases memory requirements by an exponential factor.

This paper presents a method of decomposition that, when followed, introduces no degeneration of the found policy $\pi^*(s, a)$. The summary of these conditions is presented.

SUMMARY OF REQUIREMENTS

INTRODUCTION TO APPROACH

$$d_M^{M_i, M_k} = M \longrightarrow \left\{ M_i, M_k \middle| \begin{array}{l} S_i \times (S_k/s_i) = S, S_k \times (S_i/s_k) = S \\ A = A_i \cup A_k \\ \tilde{T} \sim d^{-1}(d(\tilde{T})), d(\tilde{T}) = \tilde{T}_i, \tilde{T}_k \\ \tilde{R} \sim d^{-1}(d(\tilde{R})), d(\tilde{R}) = \tilde{R}_i, \tilde{R}_k \end{array} \right\}$$

where d, d represent belief mapping functions that decompose and recompose mapping functions. This allows **???** to be mapped as new spaces and observes are encountered. The decomposition process breaks one MDP into a parent and child:

[This diagram hasn't been drawn yet because I'm prioritizing transcribing math and text over doing the diagrams.]

PAGE 7

The system can be broken into the following MDP definitions

$\underline{M_i}$ – Parent

S_i – a collection of states, $s_i \in S_i$

A_i – a collection of actions, $a_i \in A_i$

$$\left.\begin{matrix}\tilde{T} \\ \tilde{R}\end{matrix}\right\} \text{Covered Pages on BII p12-14}$$

$P(s_i'|s_i, a_i)$ is observed directly

$$R_t\left(\begin{array}{c|c} s_i' & s_i \\ a_k' \;,\; a_i & a_k \end{array}\right) = R_t\left(\begin{array}{ccc} s_i' & a_i & s_i \\ s_k' & a_k & s_k \end{array}\right)$$

S, T_i, S_k, S_k' are not directly observable

ii) $a_k = \pi_k(s_k)$

iii) $a_k' = \pi_k(s_k)$

iiii) (s_k, s_k') chosen indirectly by $\pi_k(\cdot)$ in a manner that

$$\boxed{A^*} \quad \longrightarrow \quad E[R_{t+1}(\cdot)] \geq E[R_t(\cdot)]$$

$\underline{M_k}$ – child

S_i' – all child states, $s_k \in S_k$

$a_k \in A_k$

$P(s_k'|s_k, a_k)$

$R_t(s_k', a_k, s_k) = R_t\left(\begin{smallmatrix} s_i \\ s_k' \end{smallmatrix}, \begin{smallmatrix} a_i \\ a_k \end{smallmatrix}, \begin{smallmatrix} s_i \\ s_k \end{smallmatrix}\right)$ s.t. s_i, s_i' are chosen by another process, and

$$\boxed{A^*} \quad \longrightarrow \quad E[R_{t+1}(\cdot)] \geq E[R_t(\cdot)]$$

PAGE 8

Definitions

$\underline{M}$ $S = (S_i/S_k) \times (S_k/S_i)$

$A = A_i \cup A_k$

$T = P(S \times A \times S)$

$R =$ real, positive, convergent stochastic as $t \to \infty$

$R(s', a, s) = R \begin{pmatrix} s'_i & a_i & s_i \\ , & , & , \\ s'_k & a_k & s_k \end{pmatrix}$

Parent MDP

$\underline{M_i}$ $S_i, s_i \in S_i$

$a_i \in A_i$

$P \left(\begin{array}{c|ccc} s'_i & a_i & , & s_i \\ a'_k & a_k & & s_k \end{array} \right)$

$R_t \begin{pmatrix} s'_i & & s_i \\ , & a_i & , \\ a'_k & & a_k \end{pmatrix} = R_t \begin{pmatrix} s'_i & a_i & s_i \\ , & , & , \\ s'_k & a_k & s_k \end{pmatrix}$ s.t. s_k, s'_k are not directly observable

$a_k = \pi_k(s_k)$

$a'_k = \pi_k(s'_k)$

$*$ — assume s_k, s'_k chosen s.t. as $t \to \infty$ $E[R_{t+1}()] \geq E[R_t(\cdot)]$

Child MDP

$\underline{M_k}$ $S_k, s_k \in S_k$

$a_k \in A_k$

$P(s'_k | s_k, a_k)$

$R_t(s'_k, a_k, s_k) = R_t \begin{pmatrix} s'_i & a_i & s_i \\ , & , & , \\ s'_k & a_k & s_k \end{pmatrix}$ s.t. s_i, s'_i are chosen by another process

$a_i = \pi_i(s_i)$

$*$ time monotonicity assumed.

PAGE 9

Basic mapping requirements

S_i, S_j, S_k: $\quad S_j \times (S_k/S_j) \supseteq S_i, \quad S_k \times (S_j/S_k) \supseteq S_i$

A_i, A_j, A_k: $\quad A_i \subseteq A_j \cup A_k$

T_i, $R_i \sim$ unknown/unknowable, stable decomposition

more$\longrightarrow$ assumed
$*$ important to select so that $\tilde{T}_i$ & $\tilde{T}_k$ seem independent

$\exists f_1 : \tilde{T}_i \to \tilde{T}_j, \tilde{T}_k$, invertible; $\tilde{T}_i = f_1\left(f^{-1}\left(\tilde{T}_i\right)\right)$
$\exists f_2 : \tilde{R}_i \to \tilde{R}_j, \tilde{R}_k$, invertible; $\tilde{R}_i = f_2\left(f_2^{-1}\left(\tilde{R}_i\right)\right)$

(Network approach)

Parent/child augmentation

j – parent $\quad k$ – child
$\tilde{R}_j \leftarrow E[\tilde{R}_k]$
$s_j \in S_j \leftarrow \{S_j, a_k = \pi_k(\cdot)\}$

Continuous
(Now)

old approach

Brech
Reword
(BI, p.72)

PAGE 11

Total Mapping

$$* \; A_R = \left\{ \begin{array}{l} \text{State 1, State 2, Action 1, Action 2} \\ \text{merge, time up, time down} \end{array} \right\}$$

Given $S \in \mathbb{R}^n$, define dimensions $\{i_s\}_{i_s=1}^n$

$A \in \mathbb{N}^m$, define dimensions $\{i_r\}_{i_r=1}^m$

Then, with an initial MDP $M = \langle S, A, T, R, \pi, M_R \rangle$, all possible "sub mdps" M_1, M_2, M_3, ... represent the family of MDPs which can be created from M, $\mathcal{P}(M) = \{M_x | S_x \subseteq S, A_x \subseteq A, R, \pi$ from MDPs **???**$\}$ and each member M_x is characterized by a language $J_{sx} \subseteq \{i_s\}_{i_s=1}^n$ or $J_{sy} \subseteq \{i_r\}_{i_r=1}^m$ where $J_{sx} \times J_{sy}$ defines a space S_R, for the reconfiguration MDP to explore, with actions from A_R.

$$J \left| \begin{array}{l} \text{Reward is defined as average expected reward over an epoch } e. \\ \text{in terms of transition} \end{array} \right.$$

$$* \; S_R = \mathcal{P}\left(\{i_s\}_{i_s=1}^n\right) \times \mathcal{P}\left(\{i_r\}_{i_r=1}^m\right) \qquad \longleftarrow \text{ exponentional increase in space (stupid!)}$$

Problems 1) exponential space consumption

2) how to handle chaining/nesting

3) how to structure action choice policy

PAGE 12

Mapping function rewards

Given $(S_{\text{map}}, A_{\text{map}}, T_{\text{map}}, R^i_{\text{map}}, \pi_{\text{map}})$, applied to $M = \langle S_x, A_x, \tilde{T}_x, R_x, \tilde{\pi}_x \rangle$, we may trivially define $M_y = \langle S_y, A_y, \tilde{T}_y, R_y, \pi_y \rangle$ in a method consistent with Bush, p. 74, with M_x being the parent process and M_y being the child.

a) for R_{map^i}, there are five versions $i \in \{1, \ldots, 5\}$
b) Given M_x, M_y, a merge is also possible, so recovering M
c) we can perform temporal Sink actions on an MDP (Book I, p. 88)
   $\hookrightarrow$ reduce resolution
   $\hookrightarrow$ re-increase resolution

Actions

$\therefore$ Seven "actions" can be performed on an MDP: (M_R)

???

$$\left\{R^i_{\text{map}}\right\}_{i=0}^{5} \cup \{\text{merge}\} \times \left\{\text{scale up}^i\right\}_{i=0}^{5} \cup \{\text{normal}\} \cup \{\varnothing\}$$

Reward

$$R(s', a, s) = \sum_{l \in e} R(l) \qquad \text{reward during a trajectory}$$

$e =$ epoch

Transition

-easy to explain in MS Word

$$T = \begin{cases} 1 - \text{allow } \textbf{???} \\ 0 - \text{otherwise} \end{cases}$$

PAGE 13

Reward function mapping (5 way)

knowing $R\left(\{s_x, s_y\}|a, \{s'_x, s'_y\}\right) = R(s|a, s)$

1) Average method:

$$R\left(s'_x|a, s_x\right) = \underbrace{\sum_{s_y}\sum_{s'_y} R\left(\{s'_x, s'_y\}|a, \{s_x, s_y\}\right)}_{|S_y|^2}$$

$m \in \{\max, \min\}$

$m' \in \{\max, \min\}$

$\max, \max$	2)	max method!
$\max, \min$	3)	$\min$
	4)	$R\left(s'_x|a, s_x\right) = m \quad m' \quad s_x \in S_x \quad s'_y \in S_y \quad R\left(\{s'_x, s'_y\}|a, \{s_x, s_y\}\right)$
$\min, \max$	5)	
$\min, \min$		
	6)	