A Deep Reinforcement Learning Framework for Optimizing Fuel Economy of Hybrid Electric Vehicles

Pu Zhao, Yanzhi Wang, Naehyuck Chang, Qi Zhu, Xue Lin

ASP-DAC 2018
Outline

- Motivation
- HEV System Architecture
- DRL Framework of HEV power control
- Experimental Results
Motivation

- Hybrid electric vehicles (HEVs) combine the energy efficiency of electric motor (EMs) and a long driving range of internal combustion engine (ICE)
- The relatively complicated powertrain structures of HEVs necessitate an effective power management policy to determine the power split between ICE and EM

BMW Concept 7 Series ActiveHybrid
Outline

- Motivation
- HEV System Architecture
- DRL Framework of HEV power control
- Experimental Results
HEV System Architecture

- HEV Components
  - Internal Combustion Engine (ICE)
  - Electric motor (EM)
  - Vehicle Dynamic
  - Powertrain Mechanics

- HEV Control
  - The HEV controller needs to control the operation of the ICE, EM and powertrain to meet the target propulsion
HEV System Architecture -- HEV Components

- Parallel hybrid powertrain
  - ICE and EM propel the vehicle in parallel

- Internal Combustion Engine (ICE)
  - ICE fuel efficiency:
    \[
    \eta_{ICE}(T_{ICE}, \omega_{ICE}) = \frac{T_{ICE} \cdot \omega_{ICE}}{(m_f \cdot D_f)}
    \]
    
    \[\omega_{ICE}^{\text{min}} \leq \omega_{ICE} \leq \omega_{ICE}^{\text{max}}\]
    
    \[0 \leq T_{ICE} \leq T_{ICE}^{\text{max}}(\omega_{ICE})\]
HEV System Architecture -- HEV Components

- **Electric Motor (EM)**
  - a motor to propel the vehicle solely or together with ICE
  - a generator to charge the battery pack

\[ 
\eta_{EM}(T_{EM}, \omega_{EM}) = \begin{cases} 
\frac{(T_{EM} \cdot \omega_{EM})}{P_{batt}} & T_{EM} \geq 0 \\
\frac{P_{batt}}{(T_{EM} \cdot \omega_{EM})} & T_{EM} < 0 
\end{cases} 
\]

\[ 
0 \leq \omega_{EM} \leq \omega_{EM}^{max}, \\
T_{EM}^{min}(\omega_{EM}) \leq T_{EM} \leq T_{EM}^{max}(\omega_{EM}). 
\]
HEV System Architecture -- HEV Components

- Vehicle Dynamics

\[ F_{TR} = m \cdot a + F_g + F_R + F_{AD}, \]
\[ F_g = m \cdot g \cdot \sin \theta, \]
\[ F_R = m \cdot g \cdot \cos \theta \cdot C_R, \]
\[ F_{AD} = 0.5 \cdot \rho \cdot C_D \cdot A_F \cdot v^2, \]

- The demanded power for propelling the vehicle:

\[ T_{wh} = F_{TR} \cdot r_{wh}, \]
\[ \omega_{wh} = \frac{v}{r_{wh}}. \]
\[ P_{dem} = F_{TR} \cdot v = T_{wh} \cdot \omega_{wh}. \]
Powertrain Mechanics

The speed and torque of the ICE and EM must satisfy the speed and torque relation:

\[
\omega_{wh} = \frac{\omega_{ICE}}{R(j)} = \frac{\omega_{EM}}{R(j) \cdot \rho_{reg}}, \\
T_{wh} = R(j) \cdot (T_{ICE} + \rho_{reg} \cdot T_{EM} \cdot (\eta_{reg})^{\alpha}) \cdot (\eta_{gb})^{\beta}
\]

\[
\alpha = \begin{cases} 
+1 & T_{EM} \geq 0, \\
-1 & T_{EM} < 0. 
\end{cases}
\]

\[
\beta = \begin{cases} 
+1 & T_{ICE} + \rho_{reg} \cdot T_{EM} \cdot (\eta_{reg})^{\alpha} \geq 0, \\
-1 & T_{ICE} + \rho_{reg} \cdot T_{EM} \cdot (\eta_{reg})^{\alpha} < 0. 
\end{cases}
\]
HEV System Architecture -- HEV Control

- The speed and acceleration are determined by the driver
- The HEV controller controls the operation of the ICE, EM and powertrain to meet the target propulsion
- Control Variables:
  - The battery output power
  - The gear ratio
  - The ICE torque
  - The EM torque
- Control variables follow the operating principles of HEV components
Outline

- Motivation
- HEV System Architecture
- DRL Framework of HEV power control
- Experimental Results
DRL Framework

- Basics of DRL
- DRL Formulation
  - State Space
  - Action Space
  - Reward Function
- DRL Procedure
  - Offline DNN Construction
  - Online Deep Q-Learning
DRL Framework -- DRL basics

- Interaction between agent and environment

- Agent selects actions
- Environment responds to actions and presents new situations to the agent
- Environment also gives rise to rewards
DRL Framework -- DRL basics

- value function
  - the expected accumulated reward with discount

\[
Q(s, a) = \mathbb{E} \left[ \sum_{k=0}^{\infty} \gamma^k r_k \middle| s_0, a_0 \right]
\]

- the reward
  - the negative of the fuel consumption in the time slot

\[
r_k = -\dot{m}_f \cdot \Delta T
\]

- The DRL agent targets at maximizing the Q value
DRL Framework

- Basics of DRL
- DRL Formulation
  - State Space
  - Action Space
  - Reward Function
- DRL Procedure
  - Offline DNN Construction
  - Online Deep Q-Learning
DRL Framework -- DRL Formulation

State Space

- A finite number of states, each represented by:
  - the propulsion power demand,
  - vehicle speed,
  - charge stored in the battery pack,
  - predicted propulsion power demand for the next time slot

\[
S = \left\{ s = [p_{dem}, v, q, pre]^T \mid p_{dem} \in P_{dem}, \\
v \in V, q \in Q, pre \in P_{pre} \right\}
\]
DRL Framework -- DRL Formulation

- **State Space**
  - $Q$ is constructed by discretizing the range of charge stored in the battery pack

  $$Q = \{q_1, q_2, \cdots, q_N\},$$
  $$q_{min} = q_1 < q_2 < \cdots < q_N = q_{max}$$

- incorporate future driving characteristics (i.e., $pre$) into consideration for more effective representation

  $$pre_i \leftarrow (1 - \alpha) \cdot pre_{i-1} + \alpha \cdot meas_{i-1}$$
DRL Framework -- DRL Formulation

- **Action Space**
  - A finite number of actions, each represented by:
    - the discharging current of the battery pack
    - the gear ratio

\[
A = \{a = [i, R(j)]^T \mid i \in I, R(j) \in R\}
\]

- **I** : a finite (discretized) number of discharging current values in \([-I_{max}; I_{max}]\)
  - \(i > 0\) : discharge the battery pack
  - \(i < 0\) : charge the battery pack

- **R** contains all allowable gear ratio values
  - Usually 4 or 5 values
DRL Framework -- DRL Formulation

- **Reward Function**

\[ r_k = -\dot{m}_f \cdot \Delta T \]

- \( r_k \) is the reward the agent receives after taking action \( a_k \) in state \( s_k \)
- the negative of the fuel consumption in the time slot

- **DRL agent targets at maximizing the expected return**

\[ \sum_{k=0}^{\infty} \gamma^k \cdot r_k, \]

- the discounted sum of rewards
DRL Framework

- Basics of DRL
- DRL Formulation
  - State Space
  - Action Space
  - Reward Function
- DRL Procedure
  - Offline DNN Construction
  - Online Deep Q-Learning
DRL Framework -- DRL Procedure

- The DRL procedure comprises:
  - an offline DNN construction phase
  - an online deep Q-learning phase

- offline DNN construction
  - derives the Q-value estimate for each state-action pair
  - employ a convolutional neural network as the DNN structure
  - The real world and testing driving cycles are utilized to obtain the Q value estimates for the training of DNN
DRL Framework -- DRL Procedure

- online deep Q-learning phase
  - At each decision epoch, the policy selects
    - the action with the maximum $Q(s_k, a)$ value estimate with probability $1 - \varepsilon$,  
    - or a random action with probability $\varepsilon$
  - At the next decision epoch, update Q-value

$$Q(s, a) \leftarrow Q(s, a) + \alpha \cdot e(s, a) \cdot \delta,$$

$$\delta \leftarrow r_{k+1} + \gamma \cdot \max_{a'} Q(s_{k+1}, a') - Q(s_k, a_k).$$

- $r_k(s_k, a_k)$ is the observed reward
- $\alpha$ is a coefficient controlling the learning rate
- $\gamma$ is the discount rate
Algorithm 1 The DRL Framework of HEV power control

Offline:
1: Simulate the control process using an arbitrary but gradually refined policy for enough long time;
2: Obtain the state transition profile and \( Q(s, a) \) value estimates during the process simulation;
3: Store the state transition profile and \( Q(s, a) \) value estimates in experience memory \( D \) with capacity \( N_D \);
4: Train a DNN with features \( (s, a) \) and outcomes \( Q(s, a) \);

Online:
5: for each execution sequence do
6:   for each decision epoch \( t_k \) do
7:     With probability \( 1 - \varepsilon \) select the action \( a_k = \arg \max_a Q(s_k, a) \), otherwise select an action randomly;
8:     Perform system control using the chosen action;
9:     Observe reward \( r_k(s_k, a_k) \) during time period \([t_k, t_{k+1}]\) and the new state \( s_{k+1} \) at the next epoch;
10:    Store transition set \((s_k, a_k, r_k, s_{k+1})\) in \( D \);
11:    Update \( Q(s_k, a_k) \) using \( \max_{a'} Q(s_{k+1}, a') \) and \( r_k(s_k, a_k) \) based on the Q-learning updating rule;
12:   end for
13: Update DNN weight set \( \theta \) based on the newly updated Q-value estimates in a mini-batch manner;
14: end for
Outline

- Motivation
- HEV System Architecture
- DRL Framework of HEV power control
- Experimental Results
Experimental Results

- based on both real-word and testing driving cycles
- compared with the rule-based policy

<table>
<thead>
<tr>
<th>Driving cycle</th>
<th>Rule-based</th>
<th>Proposed method</th>
<th>reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDDS</td>
<td>412.3g</td>
<td>303.5g</td>
<td>26.4%</td>
</tr>
<tr>
<td>NEDC</td>
<td>319.8g</td>
<td>203.5g</td>
<td>36.4%</td>
</tr>
<tr>
<td>NYCC</td>
<td>86.1g</td>
<td>37.6g</td>
<td>56.3%</td>
</tr>
<tr>
<td>HWFET</td>
<td>364.0g</td>
<td>201.9g</td>
<td>44.5%</td>
</tr>
<tr>
<td>Modem1</td>
<td>228.6g</td>
<td>162.6g</td>
<td>28.9%</td>
</tr>
<tr>
<td>Modem2</td>
<td>344.9g</td>
<td>225.6g</td>
<td>34.6%</td>
</tr>
<tr>
<td>total</td>
<td>1755.7g</td>
<td>1134.7g</td>
<td>35.4%</td>
</tr>
</tbody>
</table>
Experimental Results

- compared with the rule-based policy

The MPG values achieved by the proposed DRL framework and the rule-based policy
Experimental Results

- compared with the RL-based method
  - employs $TD(\lambda)$ learning algorithm

- Prediction can decreases fuel consumption
- DRL-based power control can achieve better fuel economy than the RL-based framework
Thank You!