Online Planning for F1 Race Strategy Identification

Diego Piccinotti,
Amarildo Likmeta, Nicolò Brunello, Marcello Restelli
diego.piccinotti@{mlcube.com, mail.polimi.it}

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

POLITECNICO MILANO 1863

Problem and Motivation

- **Markov Decision Processes** [4]: a sequential decision problem defined by the 6-tuple

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \mu \rangle.$$

Given a Markovian policy π, the policy-dependent value of each state is defined as:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,\middle|\, s_0 = s, a_t \sim \pi(s_t)\right]$$

The goal of an autonomous agent is to find the policy π maximizing the **value function** in each state:

$$\pi^*(s) = \arg\max_{\pi} V^{\pi}(s) \quad \forall s \in \mathcal{S}$$

- **Online Planning**: find the locally-optimal policy in a specific environment state by using a (possibly approximate) model of the environment to simulate trajectories

$$\pi^*(s) = \max_{\pi} V^{\pi}(s), \quad \forall s \in \mathcal{S}$$

Stochastic environment
Transition model $\mathcal{P}$ is stochastic

Continuous state-space
The set of possible successor states is infinite

Open Loop Planning

- **Goal**: find the optimal sequence of actions to perform starting from the current state, regardless of the intermediate states visited, averaging between them [2].
- The **value of the sequence** τ starting from the state s is defined as:

$$V_{OL}(s, \tau) = \mathbb{E}\left[\sum_{t=0}^{m} \gamma^t r_t \,\middle|\, s_0 = s, a_t \in \tau\right]$$

- The **optimal state-action function** is

$$Q^*_{OL}(s, a) = \max_{\tau_a} V_{OL}(s, \tau_a)$$

- Since $Q^*_{OL}(s, a) < Q^*(s, a)$, open-loop planning suffers a loss of performance, but limits the size of the search tree.

Contributions

- **Temporal Difference update** to reduce **variance** in returns from roll-out phase in Open-Loop MCTS algorithm.

$$Q(\mathcal{N}', a) \leftarrow Q(\mathcal{N}', a) + \alpha(\mathcal{N}'.r + \gamma\Delta - Q(\mathcal{N}', a))$$

- **Real-world application**: application of the Q-Learning Open Loop Planning algorithm to Formula 1 pit-stop strategy identification, using real-world F1 lap times datasets.

Formula 1 race strategy identification

Problem modeling

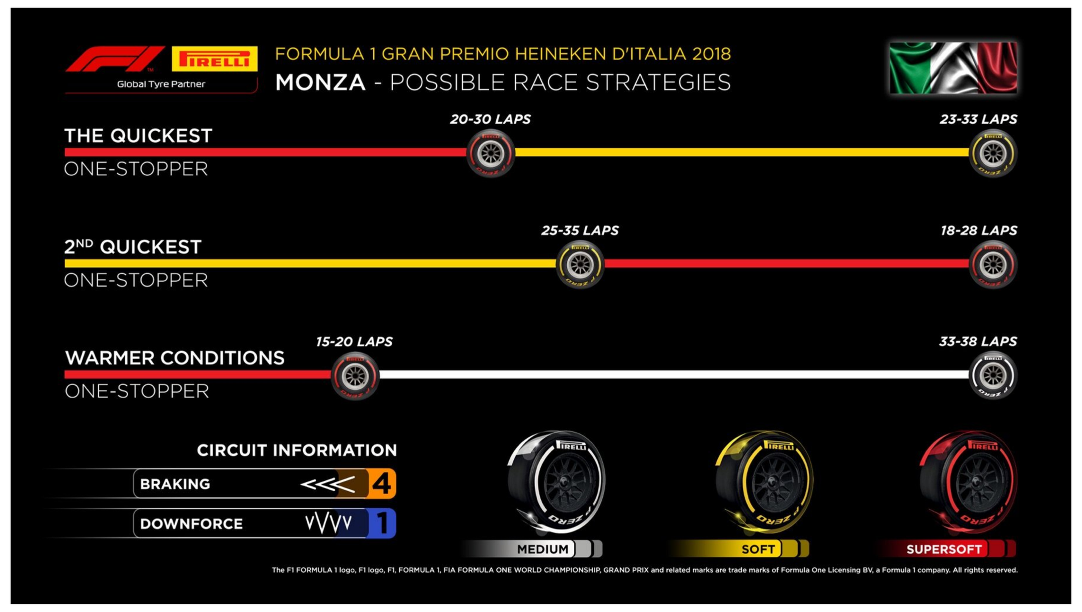
- **Problem**: decide, at each lap, whether to stop the car to change tires.
- **Constraints**: limited compound availability and at least two different compounds are to be used during the race.

We model a single-agent MDP

- **State**: features for each driver plus global race boolean flags.
- **Actions**: stay on track or pit-stop for one of the available compounds.
- **Reward**: negative normalized lap time.
- **Transition model**: Lap time simulator from [1], adapted to lap-by-lap planning
- **Discount factor**: set to 1, to consider full-episode outcome.
- Constrain actions to follow F1 rules on tire changes.

Main difficulties

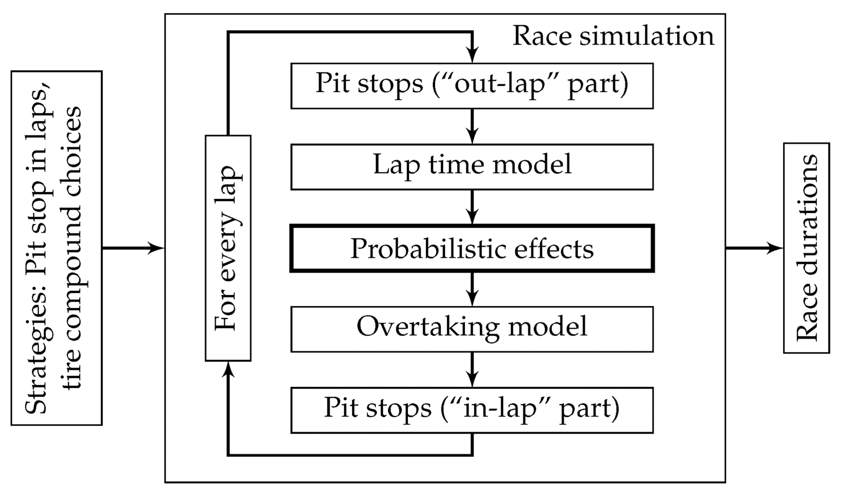
- **Continuous state-space**: cumulative race time is part of the state variables.
- **High stochasticity**: driver interaction, errors and random events.
- **Return difference between actions**: pit-stop actions cost around 30s more than staying on track.
- Good policies need to **balance pit-stop time cost** with performance given by fresh tires.



Sample Race Strategy from Pirelli [3]

Lap time simulator

For the transition model, we adapt a probabilistic lap-time simulator from literature [1].



Simulator diagram, taken from [1]

Original work features

- Lap time is computed as sum of contributes modeled independently.
- Each contribute is modeled with a probabilistic approach.

Extensions

- Specify actions for each driver lap by lap.
- Dynamically add Safety Car events during simulation.

Planner performance comparison - Sebastian Vettel, races from 2015-2018

Season	Track	ESPN	True	VSE	Sarsa UCT	Power UCT	OL UCT	QL-OL UCT	Ranking Gain
2015	Japan	4576.01±1.0	4577.52±1.0	4575.34±1.3	4575.36±1.2	4583.25±2.4	4577.99±1.1	**4570.35±1.0***	0.4
2016	Japan	4507.85±1.0	4507.54±0.7	4549.01±1.3	4508.90±0.8	4524.45±1.2	4519.03±1.3	**4505.35±0.9***	0.1
2017	Australia	4470.39±1.7	4466.22±1.9	4477.29±1.3	4466.56±2.9	4474.12±2.2	4479.90±2.2	**4459.71±2.4***	-1.3
2017	Spain	5202.42±1.4	5209.89±1.3	5207.94±1.1	5196.38±2.1	5200.83±2.0	5211.05±1.1	**5188.05±1.3***	0.1
2017	Austria	4525.88±1.2	4430.84±1.7*	4491.66±1.9	4476.43±2.8	**4444.38±2.4**	4484.14±1.8	4465.85±2.9	-2.4
2017	Belgium	4265.4±0.7	4256.44±1.0	4236.0±0.6*	4255.98±0.7	4259.52±0.7	4260.24±1.0	**4246.09±0.7**	2.9
2017	Russia	4419.98±1.3	4412.87±1.2*	4428.62±2.4	4425.10±2.1	4437.00±1.3	4430.93±1.7	**4421.54±1.3**	0.0
2018	China	5140.7±0.9	5134.01±1.0	5095.34±2.0*	5099.33±1.5	5128.63±0.9	5113.31±2.6	5098.93±1.5	4.2
2018	Italy	3909.37±1.9	3898.38±1.9*	3943.95±1.9	3907.22±1.4	3918.42±1.3	3911.24±1.3	**3903.67±1.5**	-0.3
2018	Brazil	4678.24±2.1*	4700.36±2.1	4711.61±1.7	4692.7±3.1	4699.25±1.7	4706.94±1.5	**4686.32±2.9**	2.0

Q-Learning Open Loop Planning

procedure OLSearch(s_0)
  Create root node $\mathcal{N}_{0,0}$ from state s_0
  while within computational budget **do**
    $\mathcal{N}_{d,i}, s \leftarrow$ TreePolicy($\mathcal{N}_{0,0}$)
    $\mathcal{V}(\mathcal{N}_{d,i}) \leftarrow$ Rollout($\mathcal{N}_{d,i}, s$)
    Backup($\mathcal{N}_{d,i}$)
  end while
  return BestChild($\mathcal{N}_{0,0}$)
end procedure
procedure TreePolicy($\mathcal{N}$)
  while $\mathcal{N}$ not terminal **do**
    if $\mathcal{N}$ not fully expanded **then**
      return Expand($\mathcal{N}$)
    else
      $\mathcal{N} \leftarrow$ BestChild($\mathcal{N}, C_p$)
    end if
  end while
  return $\mathcal{N}$
end procedure
procedure Rollout($\mathcal{N}$, s)
  $\Delta \leftarrow 0$
  while s is non-terminal **do**
    Choose $a \in A(s)$ according to rollout strategy
    Generate next state s' and reward r
    $\Delta \leftarrow \gamma\Delta + r$
    $s \leftarrow s'$
  end while
  return Δ
end procedure
procedure BestChild($\mathcal{N}, c$)
  $C(\mathcal{N})$ denotes children nodes of $\mathcal{N}$
  $C(\mathcal{N}, a)$ denotes the child of $\mathcal{N}$ corresponding to action a
  return $\arg\max_{a} Q(\mathcal{N}, a) + c\sqrt{\frac{2\ln \mathcal{N}.n}{C(\mathcal{N}, a).n}}$
end procedure
procedure Backup($\mathcal{N}, V$)
  $C'(\mathcal{N})$ denotes explored children nodes of $\mathcal{N}$
  $\mathcal{N}' \leftarrow$ parent of $\mathcal{N}$
  $\mathcal{N}.n \leftarrow \mathcal{N}.n + 1$
  while $\mathcal{N}'$ is not null **do**
    if $\mathcal{N}$ is leaf **then**
      $\Delta \leftarrow V$
    else
      $\Delta \leftarrow \max_{a' \in C'(\mathcal{N})} Q(\mathcal{N}, a')$
    end if
    $Q(\mathcal{N}', a) \leftarrow Q(\mathcal{N}', a) + \alpha(\mathcal{N}'.r + \gamma\Delta - Q(\mathcal{N}', a))$
    $\mathcal{N}'.n \leftarrow \mathcal{N}'.n + 1$
    $\mathcal{N} \leftarrow \mathcal{N}'$
    $\mathcal{N}' \leftarrow$ parent of $\mathcal{N}$
  end while
end procedure

References

[1] Alexander Heilmeier, Michael Graf, Johannes Betz, and Markus Lienkamp. Application of monte carlo methods to consider probabilistic effects in a race simulation for circuit motorsport. *Applied Sciences*, 10(12), 2020.

[2] Erwan Lecarpentier, Guillaume Infantes, Charles Lesire, and Emmanuel Rachelson. Open loop execution of tree-search algorithms. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2362–2368. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[3] Pirelli Motorsport. ItalianGP pit stop strategies. https://twitter.com/pirellisport/status/1036169705634054144, 2018. [Online;].

[4] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.