International Journal of Research in Engineering, Science and Management
Volume 6, Issue 7, July 2023
https://www.ijresm.com | ISSN (Online): 2581-5792

Reinforcement Learning in Real-World Scenarios: Challenges, Applications, and Future Directions

Harshita Srinath[1], Adithya Krishna V. Sharma[2*], M. R. Akhil[3]
[1]*Software Engineer, Accenture, Bengaluru, India*
[2]*Associate Software Engineer, Red Hat, Bengaluru, India*
[3]*Department of Computer Science and Engineering, PES University, Bengaluru, India*

***Abstract*: Reinforcement Learning (RL) has developed as a powerful machine learning paradigm that has achieved great success in a variety of applications such as gaming, robotics, and natural language processing. As academics become more interested in using RL in real-world contexts, they face new problems and complications. This work investigates the key issues of RL in real-world contexts, such as dealing with high-dimensional and continuous state-action spaces, as well as coping with partial observability via state estimation. It investigates the trade-offs between real-world experience and simulations, considering the expense and feasibility of getting real-world physical data for training. The importance of reward shaping in leading RL agents in real-world contexts is being researched. This paper discusses the limitations of RL in real-world applications, as well as potential future possibilities for developing the subject. Understanding these obstacles and opportunities will allow academics and practitioners to fully realize the potential of RL in tackling real-world problems and opening new paths for revolutionary applications.**

***Keywords*: Algorithms, autonomous, challenges, DQN, game-playing, MDP, recommendation systems, reinforcement learning, robotics, vehicles.**

1. Introduction

Reinforcement Learning (RL) has evolved as a fascinating paradigm within the larger subject of machine learning, offering a distinct approach to problem resolution. Reinforcement Learning, in contrast to traditional supervised and unsupervised learning approaches, employs agents who learn to make decisions by interacting with an environment and receiving feedback in the form of rewards or penalties based on their actions. Because of this dynamic learning process, RL excels at addressing complicated issues where traditional techniques fall short.

In recent years, Reinforcement Learning has received a lot of attention because of its extraordinary success in applications including gaming, robotic control, autonomous driving, natural language processing, and so on. These accomplishments have prompted interest in applying RL algorithms in real-world scenarios in order to address difficult challenges and improve numerous domains.

Implementing Reinforcement Learning in real-world contexts, on the other hand, poses its own set of distinct obstacles. They frequently feature high-dimensional, continuous states and behaviors, complicating the learning process. Furthermore, the assumption of perfect observability of the true state becomes untenable, resulting in partial observability and the requirement for state estimate via filters.

The cost and difficulty of gaining real-world physical experience for training agents also influence the viability of Reinforcement Learning in real-world scenarios. Real-world trials can be costly, time-consuming, and often impossible to replicate. As a result, researchers must identify reasonable approximations for state, policy, and value functions in order to accelerate learning while maintaining performance. We see how various RL algorithms accommodate continuous and high-dimensional state-action spaces, as well as state estimation techniques improving decision-making in partially observable situations.

We give insight into the current state of Reinforcement Learning in real-world circumstances, its limitations, and probable future routes for the field's advancement. Understanding the complexities of Reinforcement Learning's applicability allows academics and practitioners to harness its capacity to address critical difficulties and open new opportunities across a wide range of businesses and domains.

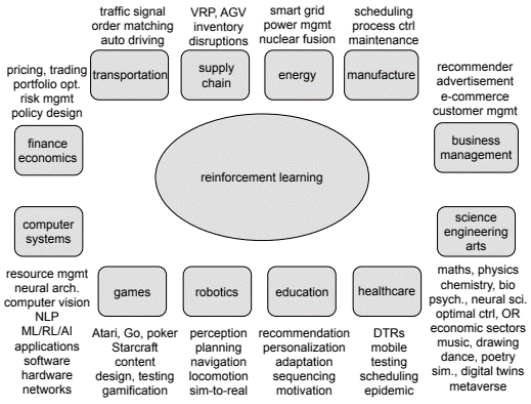

Fig. 1. Use Cases of Reinforcement Learning (Li Y. 2022)

*Corresponding author: aadithya794@gmail.com

2. Reinforcement Learning Algorithms and Techniques

Reinforcement Learning (RL) methods and techniques, different from supervised and unsupervised learning, are an important branch of machine learning. RL is powered by agents that function as domain experts, acting and learning from their interactions with the environment. In contrast to supervised learning, RL does not use labeled data and instead earns rewards or penalties based on the results of its actions. This approach enables the agent to analyze its activities in different states and fine-tune its decision-making over time.

The Markov Decision Process (MDP) is at the heart of Reinforcement Learning, in which an environment is represented as a set of states, and the agent navigates between these states by selecting actions [1].

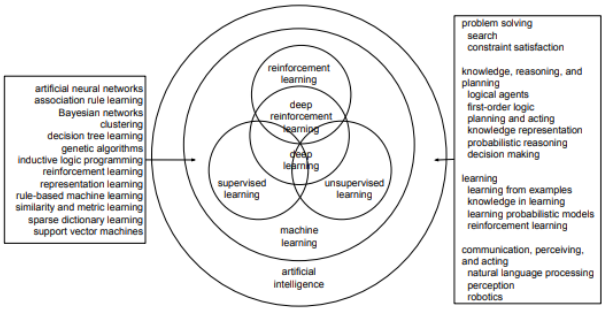

Fig. 2. Relationship of Reinforcement Learning (Li Y. 2022)

A. Q-Learning

A basic RL technique based on the bellman question in which the agent learns an action-value function (Q-function) to estimate the expected cumulative reward from performing a specific action in each state. Q-Learning is model free and off-policy, that is it can learn from data generated by any policy.

B. Deep Q-Networks (DQN)

It is an extension of the Q-Learning technique which uses deep neural networks to approximate the Q-function, allowing it to handle high-dimensional state spaces. This method also introduced experience replay and targets networks to improve stability and convergence.

C. Policy Gradient Methods

In this method, the algorithms directly optimize the policy which is to map from states to actions, to find the best optimal policy. [2] They use gradient ascent to update the policy parameters based on the expected cumulative reward.

D. Proximal Policy Optimization (PPO)

A common policy gradient method that uses a trust region approach to update policy in a more secure and efficient manner. PPO guarantees that policy updates do not depart too far from earlier policies, hence preventing policy collapse.

E. Actor Critic Methods

Actor-critic algorithms combine policy-based and value-based techniques, with an actor (policy) taking actions and a critic (value function) estimating the expected cumulative reward. The policy of the actor is revised based on the critic's judgment of acts.

F. Deep Deterministic Policy Gradients (DDPG)

An actor-critic algorithm specifically designed for continuous action spaces. DDPG uses deep neural networks to represent the policy and Q-function.

G. Advantage Actor Critic (A2C)

An efficient, parallelizable version of the actor-critic algorithm that updates the policy and value function simultaneously. A2C reduces the variance of policy gradient updates by using multiple parallel environments.

H. Trust Region Policy Optimization (TRPO)

An alternative to PPO, TRPO uses a trust region constraint to ensure that policy updates are within a safe region of the policy space. This helps maintain policy stability during training.

I. Temporal Difference Learning

A family of RL algorithms that update the value function based on the temporal difference between successive states. TD learning is widely used for value function estimation and forms the basis for many other RL algorithms.

J. Soft Actor Critic (SAC)

To improve exploration and stability, this off-policy approach combines entropy regularization and soft Q-learning. SAC encourages exploration through a stochastic policy and can handle continuous action spaces.

These algorithms are more flexible in dealing with continuous action spaces and partial observability and can handle a broader range of tasks [4].

Finally, reinforcement learning algorithms and approaches combine to offer a formidable arsenal for handling a wide range of issues. RL agents may adapt and optimize their decision-making through interactions with the environment and learning from experiences to produce extraordinary achievements across multiple domains. Understanding the benefits and limits of various RL algorithms enables researchers and practitioners to apply RL to real-world problems and advance this dynamic subject.

3. Reinforcement Learning in Robotics

Reinforcement learning (RL) allows a robot to develop optimal behavior on its own via trial-and-error interactions with its surroundings. Instead of explicitly explaining the solution to a problem, the designer of a control task in reinforcement learning offers feedback in the form of a scalar objective function that gauges the robot's one-step performance.

Consider an example [3], Trying to teach a robot to return a table tennis ball over the net. In this, the robot could see dynamic variables indicating ball position and velocity and the internal dynamics of joint position and velocity. This might record the system's states well, providing a comprehensive statistic for forecasting future observations. The robot may send torque to motors or desired accelerations to an inverse dynamics control system. The policy is a function that creates motor

commands depending on the approaching ball and current internal arm observations.

The goal of a reinforcement learning issue is to discover a policy that optimizes the long-term sum of rewards *R(s, a)*; a reinforcement learning method is one that is meant to find such a (near)-optimal policy [3]. In this case, the reward function might be dependent on the success of the hits as well as other variables such as energy usage.

When opposed to standard benchmark issues, robotics provides distinct obstacles for reinforcement learning. Complications emerge when dealing with high-dimensional, continuous states and actions, which are common in robotic systems. Furthermore, in robotics, full observability of the genuine state and noise-free data are frequently impractical assumptions. The learning system must deal with partial observability by applying filters to estimate the true state and incorporate uncertainty into its estimations.

One of the most challenging issues in robotics reinforcement learning is gaining real-world experience. Data collecting from physical systems is time-consuming, costly, and difficult to repeat, therefore each trial run is costly. Due to these limits, the emphasis shifts to tasks that are less prevalent in traditional reinforcement learning standards.

To achieve learning on a realistic timescale, approximations of state, policy, value function, or system dynamics are necessary. However, relying just on simulated learning is insufficient since slight modelling errors can result in extremely divergent behavior, particularly in highly dynamic environments. As a result, the algorithms must be robust to model uncertainty and under-modeling. Another key problem is developing suitable incentive functions. The function of rewards in steering the learning system toward success while regulating the expense of real-world experience is critical. Reward shaping is an important part of this process, requiring direct involvement to create incentives that properly influence the learning system. Creating appropriate reward functions in robots requires domain expertise and might be difficult in practice.

Robotics reinforcement learning introduces complexity linked to continuous states and actions, partial observability, and uncertainty, differing greatly from traditional benchmarks [3]. Real-world experience is valuable yet expensive, hence it is necessary to design strong algorithms that can deal with model uncertainty and mistakes. Additionally, designing appropriate reward functions calls for knowledge and careful thought.

4. Reinforcement Learning in Autonomous Vehicles

A well-known application of reinforcement learning is the usage of autonomous vehicles. In contrast to conventional rule-based or pre-programmed techniques, RL enables autonomous vehicles to learn from experience and optimize their behavior based on the feedback received from the environment which makes it more suitable than the supervised approach. However, training an autonomous vehicle using reinforcement learning in a real environment can cost expensive trial and error. Despite the visual differences between virtual and actual driving scenarios, both share a similar scene parsing structure. For instance, buildings, roads, and trees may all be featured in both virtual and actual driving scenes, but their textures may differ greatly. This suggests that we can create a simulation environment that closely resembles the real world in terms of both scene parsing structure and object appearance by translating virtual images to their realistic equivalents. Hence, it is preferable to train in a virtual setting before transferring to the real one.

In RL-based autonomous vehicles, the state representation takes various factors into consideration to analyze the current driving situation such as data from sensors, cameras, lighting, weather conditions, the speed and GPS coordinates of the vehicle, data about other vehicles, traffic signals, and road markings [12]. And the action space encompasses a range of possible actions that the RL agent can take based on the current state. It typically consists of a combination of continuous and discrete actions to enable smooth and precise driving maneuvers. Common actions in this space include acceleration and deceleration (braking) to control the vehicle's speed, steering to change direction and follow the desired path, lane changes for safe switching between lanes, and the use of turn signals to indicate intentions [13]. Additionally, maintaining a safe following distance from the vehicle ahead and executing emergency maneuvers to avoid collisions or unexpected obstacles are also critical parts of the action space for autonomous vehicles.

Through continuous learning, these vehicles can learn to handle diverse road conditions, traffic patterns, and obstacles which is crucial for ensuring the safety and efficiency of autonomous vehicles in real-world scenarios.

Additionally, there are ethical considerations that need to be addressed when using RL in autonomous vehicles. Ensuring the safety of passengers and pedestrians should be a priority, and the decision-making algorithms should be designed to prioritize human well-being in critical situations.

5. Reinforcement Learning in Recommendation Systems

We are living in a digital age where we are inundated with an overwhelming amount of data and options which makes it quite difficult for us to make decisions. A Recommendation system is an efficient data filtering algorithm that helps users to get a customized list of items/content based on their interests and preferences. Whether we are browsing through online stores, streaming music or movies or using social media, recommendation systems make our lives a little easier by helping us pick the right options by saving our valuable time and effort. The usage of recommendation systems also plays a crucial part in improving customer experience by providing the most suitable content to the users and hence are widely used.

Initially, the recommendation systems were considered classification or prediction problems. Content-based filtering and collaborative filtering systems are the two popular traditional recommendation systems. Although each has its strengths, they come with limitations such as scalability, sparsity, limited content exposure, cold start problem for new items and users and popularity bias. The hybrid system which

is a combination of both can also only eliminate a part of these problems.

The recommendation system can be treated as a Markov decision process (MDP) and can be solved using a reinforcement learning algorithm. RL addresses all the shortcomings of traditional systems by adopting a more dynamic approach designed to handle evolving user preferences, complex environments, changing market trends, etc. which ensure long-term user satisfaction.

The RL model is designed as a sequential decision-making problem, where the recommendation system (agent) interacts with the environment (users and items). Over time, the agent observes the user's behavior and acts (to recommend an item) based on the user's feedback and interaction. The main components of an RLRS are discussed below,

A. State Representation

State is the information available to the agent that includes relevant features or embeddings that capture user preferences and context which should obey the Markov property

B. Policy Optimization

The policy determines which action must be taken in each state. Various RL algorithms, such as Deep Q-Networks (DQN), Deep Deterministic Policy Gradients (DDPG), Proximal Policy Optimization (PPO), or Multi-Armed Bandits (MAB), can be used to train the recommendation system. Policy optimization algorithms used by DRL-based RSs could be generally divided into value-based, policy gradient, and actor-critic methods [14].

- *Value-function approaches:* The value-function approach in recommendation systems leverages reinforcement learning (RL) to estimate the expected cumulative reward associated with different item recommendations for users. It provides a quantitative measure of the potential rewards or benefits the recommendation system can achieve by suggesting specific items in various user contexts. It depends heavily on the samples to learn the optimal policy. Thus, these approaches are suitable for small discrete action spaces and are often applied in small-scale recommender systems, such as traditional interactive recommendation and conversational recommendation.
- *Policy search methods:* Policy search methods in recommendation systems are aimed at directly optimizing the recommendation policy without explicitly estimating value functions. Policy search approaches directly look for the optimum policy based on user interactions and feedback, in contrast to typical RL methods that first estimate value functions and then derive the policy. These techniques iteratively update the policy parameters using optimization algorithms like evolutionary algorithms, genetic algorithms, or stochastic gradient descent. Due to the policy gradient's enormous variances, they are only appropriate for continuous action spaces. Policy search approaches offer more effective adaptation, personalization, and exploration of item recommendations which increases overall user engagement with recommendation systems.
- *Actor-Critic algorithms:* Combines the benefits of both value-function approaches and policy search methods. The actor is responsible for learning the policy which uses policy gradients or other policy-based methods to optimize the policy directly based on the observed rewards and the critic is responsible for estimating the value function which provides feedback to the actor by assessing the quality of the policy and guiding its updates.

C. Reward Formulation

The RL agent receives a reward signal from the environment after each recommendation, indicating how well the recommended item aligns with the user's preferences. The reward formulation is critical for the training of the agent which is based on user feedback such as clicks, purchases, ratings, or any other relevant user interactions with the recommended item.

D. Environment Construction

Creating an appropriate environment is crucial for the success of the RL task. To better distinguish between different environment-building methods, they can be categorized into three groups: Simulated, Real-World, and Interactive Environments. Simulated environments are artificially created environments specifically designed for RL tasks, real-world environments are actual physical or virtual environments where the RL agent interacts with the real world and interactive environments involve interactions between the RL agent and human users.

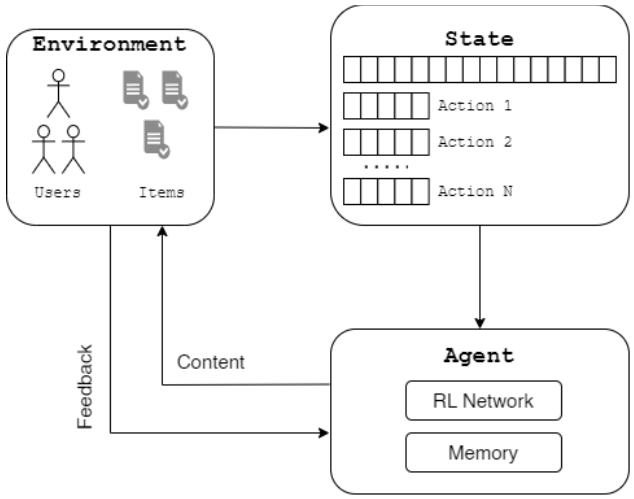

Fig. 3. Reinforcement Learning for recommendation system

6. Reinforcement Learning in Game Playing

Reinforcement learning (RL) has shown remarkable success in various applications, and one area that has gained significant attention is its application to playing video games. Typically, the reward system in a video game can be imagined as a Markov Decision Process (MDP) graph where the goal of the user is to take an action at any state such that the reward accumulated at the end of the traversal is maximized [10]. Similarly, a RL model would have its internal MDP graph akin to that of the

MDP of the video game in order to gain insights into the environment of the game.

The state space of Atari 2600 games [9] typically encompasses a vast range of 10^9 to 10^{11} states. In comparison, games like chess consist of approximately 10^{46} valid states, while more complex games like Go contain an astonishing number of 3^{361} valid states. Presently, despite the substantial advancements in computational speed, computers still encounter significant challenges when dealing with the exploration of the immense state space, aiming to traverse as many states as possible to develop a comprehensive and realistic model of the game environment.

Accurately attributing the impact of actions taken at different stages of a game on the final score presents a challenging problem known as "credit assignment" in technical terms [10]. Reinforcement learning (RL) has demonstrated remarkable success in disentangling actions worth in specific game-states. RL models tackle the credit assignment problem by associating a credit value with each state. This process involves two interwoven phases: learning and planning.

During the learning phase, the agent explores the model to gather information about the states. Subsequently, in the planning phase, the agent assigns credits to each state and determines the relative merits of different actions. Planning and learning represent iterative procedures. In each iteration, the agent first engages in learning, where it acquires knowledge about the states by exploring the environment. It then proceeds to planning, constructing transitions from one state to another by selecting those that maximize future rewards. In the subsequent learning iteration, when confronted with choosing an action for a specific state, the agent selects the transitions that lead to terminal states with the highest final scores.

However, given that numerous real-world problems feature exceedingly vast state spaces, RL agents cannot explore all states exhaustively. Instead, they operate with only the discovered portion of the environment and approximate the credit values for unvisited states based on their "knowledge" acquired from visited states. In order to approximate the values for the undiscovered state, Neural Networks can be used.

In the context of gameplay, researchers employ flexible neural networks (NNs) [9] capable of comprehending the diverse patterns present within the state space. Simultaneously, these NNs possess substantial depth, represented by multiple layers, enabling them to capture intricate distinctions among transitions within the state space.

7. Current Challenges

When applied to real-world circumstances, Reinforcement Learning (RL) confronts many problems. Handling high-dimensional and continuous state and action spaces is a significant challenge. Many real-world jobs necessitate fine-grained actions, which makes efficient exploration and learning complex policies difficult. The agent's capacity to make educated decisions based on incomplete and noisy observations is hampered by partial observability and uncertainty in the environment. Acquiring real-world experience is expensive and restricted, making sample-efficient RL algorithms critical. Furthermore, model uncertainty and robustness are critical considerations since real-world systems have complicated dynamics and learning models might introduce errors. Finally, for successful RL deployment in real-world applications, appropriate reward functions must be designed and function approximation trade-offs must be managed. Taking on these difficulties increases the use of RL in various and complicated situations. Some of the significant challenges or shortcomings related to real-world reinforcement learning are depicted below,

A. High-Dimensional and Continuous State-Action Spaces

Real-world activities frequently necessitate fine-grained actions, which results in high-dimensional and continuous state and action spaces. In such situations, efficient exploration and learning of complicated policies becomes difficult.

B. Partial Observability and Uncertainty

Due to sensor constraints or noisy observations, agents in many real-world contexts may not have complete information about their surroundings. This partial observability creates uncertainty, making correct decision-making difficult for agents.

C. Costly Real-World Experience

Gathering data through genuine interactions with the environment can be time-consuming and expensive in real-world applications. To develop effective rules, RL algorithms must be sample-efficient, making the most of the given experiences.

The data efficiency can be evaluated by looking at the amount of data required to reach the necessary performance threshold as depicted by the following equation,

$$J^{eff.} = \min |\mathcal{D}_i| \text{ s.t. } R(\texttt{Train}(D_i)) > R_{\min}$$

where R_{min} is the desired performance threshold and D_i is the data selected.

D. Model Uncertainty and Robustness

Real-world systems are uncertain, and trained models may not accurately reflect all features of the environment. To provide robust performance, RL algorithms should be able to tolerate model faults and adapt to new settings.

E. Reward Design and Function Approximation

Designing effective reward functions for agents to learn desired behaviors is challenging, requiring careful balance between complexity and performance.

Addressing these challenges is crucial for the successful deployment of RL in real-world applications, where safety, efficiency, and generalization are paramount. Researchers and practitioners continue to develop novel algorithms and techniques to tackle these obstacles and advance the use of RL in complex and dynamic environments.

8. Conclusion

In conclusion, this work investigated the various applications

of Reinforcement Learning (RL) in real-world scenarios, spanning from robots and autonomous cars to game playing and recommendation systems. We investigated the potential of RL to meet these difficulties, proving its adaptability to complex and changing contexts, user preferences, and market trends. Despite its bright future, we highlighted the existing obstacles that RL faces in modern-day real-world applications, such as scalability, sparsity, cold start issues, and popularity bias in recommendation systems, among others.

Recognizing these limitations allows researchers and practitioners to concentrate on developing novel solutions and advances in RL algorithms to improve their efficacy in real-world scenarios. As RL evolves, it has the potential to revolutionize a wide range of sectors and make substantial contributions to the development of intelligent systems and technology.

Moving forward, it is critical to stimulate interdisciplinary cooperation and research in RL approaches, as well as their integration into practical applications. We can unlock the full potential of RL by exploring new areas and exploiting emerging technology, moving us towards a future in which intelligent agents and systems augment human capabilities and revolutionize the way we interact with the environment.

References

[1] Singh R, Gupta A, Shroff NB. Learning in constrained Markov decision processes. IEEE Trans Control Netw. Syst., 2023;10(1):441–53.

[2] Bolland A, Louppe G, Ernst D. Policy gradient algorithms implicitly optimize by continuation, 2023.

[3] Kober J, Bagnell JA, Peters J. Reinforcement learning in robotics: A survey. Int J Rob Res., 2013;32(11):1238–74.

[4] AlMahamid F, Grolinger K. Reinforcement learning algorithms: An overview and classification, 2022.

[5] Lin Y, Liu Y, Lin F, Zou L, Wu P, Zeng W, et al. A survey on reinforcement learning for recommender systems. IEEE Trans Neural Netw Learn Syst., pp. 1–21, 2023.

[6] Afsar MM, Crump T, Far B. Reinforcement Learning based Recommender Systems: A Survey. ACM Comput Surv., 2023;55(7):1–38.

[7] Li Y., Reinforcement learning in practice: Opportunities and challenges, 2022.

[8] Dulac-Arnold G, Levine N, Mankowitz DJ, Li J, Paduraru C, Gowal S, et al. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. Mach Learn, 110(9):2419–68, 2021.

[9] Torrado RR, Bontrager P, Togelius J, Liu J, Perez-Liebana D. Deep reinforcement learning for general video game AI. In: 2018 IEEE Conference on Computational Intelligence and Games (CIG), 2018.

[10] Shao K, Tang Z, Zhu Y, Li N, Zhao D., A survey of deep reinforcement learning in video games, 2019.

[11] Li Y., Reinforcement learning in practice: Opportunities and challenges, 2022.

[12] Kiran BR, Sobh I, Talpaert V, Mannion P, Sallab AAA, Yogamani S, et al. Deep reinforcement learning for autonomous driving: A survey. IEEE Trans Intell. Transp. Syst., 2022;23(6):4909–26, 2022.

[13] Shalev-Shwartz S, Shammah S, Shashua A., Safe, multi-agent, reinforcement learning for autonomous driving, 2016.

[14] Afsar MM, Crump T, Far B. Reinforcement Learning based Recommender Systems: A Survey. ACM Comput. Surv., 55(7):1–38, 2023.