BAB 2
LANDASAN TEORI

2.1 Intelligent Tutoring System (ITS)

Intelligent Tutoring System (ITS) adalah aplikasi komputer yang dirancang untuk meniru peran pengajar manusia dalam memberikan materi pembelajaran secara individual [14].*Intelligent Tutoring System* (ITS) menggunakan pendekatan one-to-one, di mana sistem berinteraksi langsung dengan siswa, menilai kemampuan mereka, dan menyesuaikan materi serta metode pengajaran sesuai dengan kebutuhan individu. Hal ini bertujuan untuk mengatasi keterbatasan pembelajaran berbasis komputer sebelumnya yang kurang memperhatikan keberagaman siswa. Komponen-komponen utama dalam sebuah *Intelligent Tutoring System* (ITS) meliputi:

- Domain Model
  Berisi materi pembelajaran dan kumpulan soal yang akan diberikan kepada pelajar. Dalam penelitian ini, materi yang digunakan adalah mata pelajaran matematika dengan fokus pada topik-topik aljabar. Soal diberikan dalam tiga tingkatan kesulitan—mudah, sedang, dan sulit yang dipilih berdasarkan kebijakan *(policy)* hasil pelatihan model *Reinforcement Learning* (RL).

- Student Model
  Mewakili tingkat pemahaman pelajar yang diperoleh dari hasil interaksi pelajar dengan sistem. Dalam konteks *Reinforcement Learning* (RL), *Student Model* menyimpan informasi sebagai *state*, seperti riwayat soal yang telah dijawab, tingkat keberhasilan menjawab, dan estimasi kemampuan pelajar. Informasi ini digunakan sebagai dasar bagi agen *Reinforcement Learning* (RL) dalam menentukan tindakan selanjutnya, yaitu pemilihan soal dengan tingkat kesulitan yang sesuai.

- Tutoring Model
  Bertindak sebagai agent dalam algoritma *Reinforcement Learning* (RL) yang mengambil keputusan adaptif berdasarkan kondisi pelajar *(state)* dan reward yang diperoleh dari hasil interaksi. Model ini mempelajari kebijakan terbaik melalui iterasi, dengan tujuan untuk memaksimalkan peningkatan performa

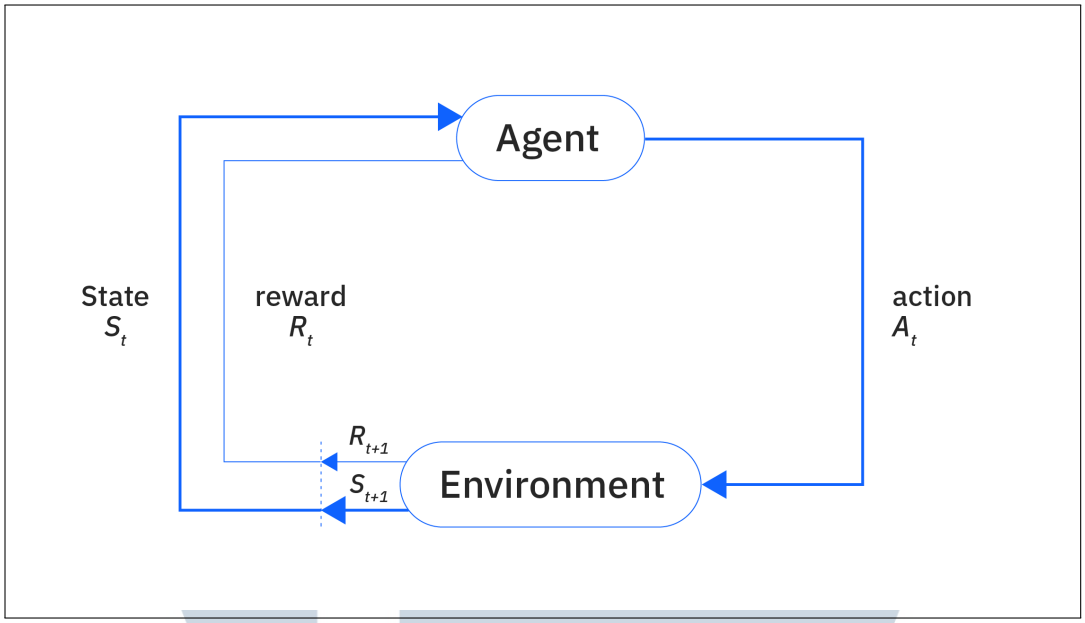
pelajar. Reward diberikan, misalnya, ketika pelajar berhasil menjawab soal dengan benar atau meningkat kemampuannya dari soal sebelumnya.

- User Interface
  Merupakan tampilan antarmuka yang menghubungkan pelajar dengan sistem. *User Interface* didesain agar ramah pengguna, mudah dipahami, dan mampu menyajikan soal serta umpan balik secara interaktif dan responsif. Selain itu, UI juga menyajikan progres pembelajaran pelajar selama proses interaksi berlangsung.

2.2 *Reinforcement Learning* (RL)

Sebuah model *Reinforcement Learning* (RL) adalah pendekatan dalam bidang *machine learning* yang memungkinkan agen (dalam hal ini sistem *Intelligent Tutoring System* (ITS)) untuk belajar melalui interaksi dengan lingkungan guna mencapai tujuan tertentu. Dalam konteks *Intelligent Tutoring System* (ITS), algoritma *Reinforcement Learning* (RL) digunakan untuk memodelkan proses pembelajaran adaptif berdasarkan performa pelajar, di mana sistem akan memberikan soal atau materi berdasarkan strategi optimal yang diperoleh dari pengalaman sebelumnya [15]. Model ini menekankan pada proses belajar dari umpan balik (*feedback*) berupa reward yang diperoleh setelah pelajar menjawab suatu soal.

Interaksi antara ketiga komponen tersebut memungkinkan sistem ITS untuk menyesuaikan materi pembelajaran secara adaptif berdasarkan performa pelajar. Hal ini didukung oleh Zhang Goh (2021) yang menggunakan metode RL untuk menyesuaikan kesulitan tugas secara otomatis agar performa siswa tetap stabil [16]. Proses ini bekerja secara iteratif, di mana sistem terus memperbarui kebijakan (*policy*) berdasarkan pengalaman sebelumnya. Misalnya, jika seorang pelajar berhasil menjawab soal kategori sedang dengan benar, sistem akan mencatat keberhasilan tersebut sebagai *reward* positif dan meningkatkan preferensi terhadap soal dengan tingkat kesulitan serupa atau sedikit lebih tinggi. Sebaliknya, jika jawaban salah diberikan, sistem akan menyesuaikan pilihannya agar tidak terlalu sulit bagi pelajar tersebut. Dengan cara ini, *Reinforcement Learning* mendukung pembelajaran yang bersifat personal, progresif, dan berbasis pengalaman nyata dari tiap pelajar.



Gambar 2.1. Gambar Arsitektur *Reinforcement Learning (RL)*

Sumber: Google

Gambar 2.1 menjelaskan skema dasar interaksi antara agen dan lingkungan dalam RL. Dalam hal ini, agen adalah sistem ITS yang mengambil keputusan soal apa yang diberikan, sementara lingkungan merepresentasikan kondisi siswa. Agen mengamati kondisi siswa (*state* S_t), memilih suatu tindakan (*action* A_t), lalu menerima *reward* R_t dari lingkungan, serta memperbarui pengetahuannya berdasarkan kondisi baru S_{t+1}. Proses ini berlangsung terus menerus hingga sistem memiliki strategi pemilihan soal yang optimal dan adaptif sesuai kebutuhan siswa secara individual.

Evaluasi performa model RL dalam ITS biasanya menggunakan metrik seperti akurasi prediksi keberhasilan siswa dalam menjawab soal dan tingkat retensi pengetahuan. Dalam beberapa eksperimen, model RL menunjukkan keunggulan dalam mempersonalisasi pembelajaran dibandingkan pendekatan statis atau heuristik, terutama dalam konteks peningkatan motivasi belajar dan hasil belajar siswa [17]. Sebagai contoh, apabila sistem memprediksi bahwa pemberian soal level menengah pada pelajar tertentu memiliki nilai Q tertinggi, maka sistem akan lebih sering memilih tindakan tersebut karena memiliki probabilitas keberhasilan yang tinggi dalam meningkatkan pembelajaran.

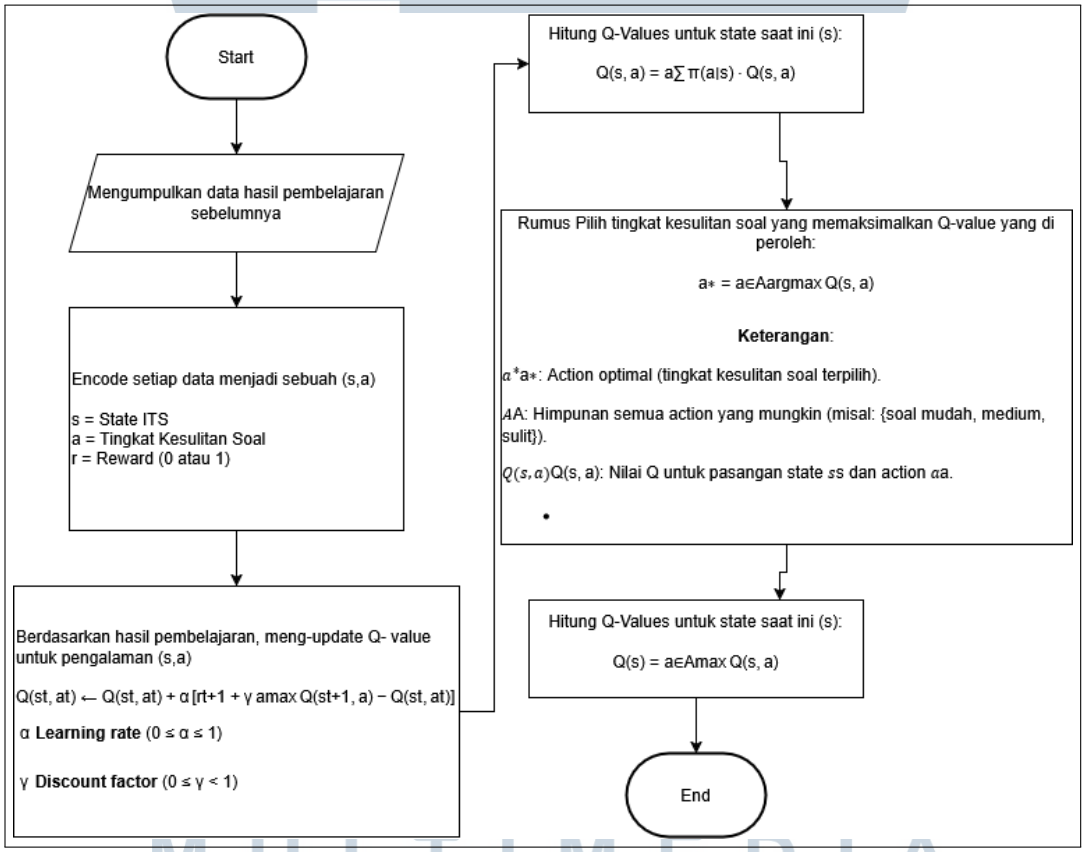
Dengan demikian, *Reinforcement Learning* merupakan pendekatan yang menjanjikan untuk diterapkan dalam sistem pembelajaran cerdas karena mampu memodelkan proses belajar yang interaktif, adaptif, dan berkelanjutan sesuai

dengan kebutuhan pelajar secara individual [18].

2.3 Q-Learning

Q-Learning adalah salah satu model dalam *Reinforcement Learning* yang digunakan untuk mempelajari kebijakan optimal dalam pengambilan keputusan berdasarkan pengalaman interaksi antara agen dan lingkungan [19]. Dalam konteks *Intelligent Tutoring System* (ITS), Q-Learning digunakan untuk menentukan tingkat kesulitan soal yang paling sesuai berdasarkan performa pelajar, tanpa memerlukan model eksplisit dari lingkungan pembelajaran [20].

Proses kerja algoritma Q-Learning dalam sistem ITS dapat dijelaskan dalam enam tahap sebagaimana ditunjukkan pada Gambar 2.2:



Gambar 2.2. *Flowchart* Proses Model *Q-Learning*

Proses kerja algoritma Q-Learning dalam sistem Intelligent Tutoring System (ITS) terdiri dari enam langkah utama yang saling berhubungan.

1. Mengumpulkan Data Hasil Pembelajaran Sebelumnya Sistem mengumpulkan data hasil pembelajaran dari log interaksi siswa. Data

ini mencakup informasi seperti `user_id`, `problem_id`, tingkat kesulitan soal (`difficulty`), jawaban siswa (`correct`), dan waktu pengerjaan (`timestamp`). Setiap baris dalam dataset mewakili satu interaksi siswa dengan sistem, yang nantinya digunakan untuk membentuk pengalaman belajar.

2. Encode Data Menjadi Pasangan *(s, a)* Setiap baris data di-encode menjadi pasangan *(state, action)* atau *(s, a)*. State (s) menunjukkan kondisi kemampuan siswa saat itu:

   - $s = 0$: Pemula
   - $s = 1$: Menengah
   - $s = 2$: Lanjutan

   Action (a) mewakili tingkat kesulitan soal:

   - $a = 0$: Soal mudah
   - $a = 1$: Soal sedang
   - $a = 2$: Soal sulit

   Contoh: Jika siswa berada pada level menengah dan diberi soal sedang, maka encode-nya adalah $(s,a) = (1,1)$.

3. Perhitungan Nilai Q (*Q-Value*) Nilai Q disimpan dalam sebuah tabel bernama Q-Table. Setiap kombinasi (s,a) memiliki nilai Q yang menunjukkan seberapa baik tindakan a saat berada pada kondisi s. Nilai ini diperbarui menggunakan rumus Q-Learning berikut:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot \left[r + \gamma \cdot \max Q(s',a') - Q(s,a) \right]$$

   Dengan keterangan:

   - $Q(s,a)$: nilai Q saat ini untuk state s dan action a
   - α : learning rate (contoh: 0,1)
   - r : reward (misalnya 1 jika jawaban benar, 0 atau -1 jika salah)
   - γ : discount factor (misalnya 0,9)
   - s' : state baru setelah tindakan dilakukan

- $\max Q(s', a')$: nilai Q tertinggi dari state berikutnya

4. Contoh Perhitungan

$$
\begin{aligned}
s &= 0 \quad \text{(pemula)} \\
a &= 0 \quad \text{(soal mudah)} \\
r &= 1 \quad \text{(jawaban benar)} \\
s' &= 1 \quad \text{(naik ke menengah)} \\
\alpha &= 0{,}1, \quad \gamma = 0{,}9 \\
Q(0,0) &= 0{,}3, \quad \max Q(1, a') = 0{,}5 \\
Q(0,0) &\leftarrow 0{,}3 + 0{,}1 \cdot (1 + 0{,}9 \cdot 0{,}5 - 0{,}3) = 0{,}415
\end{aligned}
$$

   Nilai Q meningkat karena reward positif, artinya sistem percaya bahwa memberi soal mudah kepada pemula adalah strategi yang baik.

5. Pemilihan Soal Berdasarkan Q-Value Tertinggi Sistem akan memilih soal dengan nilai Q-Value tertinggi untuk state siswa saat ini. Misalnya, jika siswa berada pada state $s = 1$, maka sistem membandingkan semua nilai $Q(1, a)$ dan memilih action a dengan nilai tertinggi. Ini disebut "soal tertentu" karena sistem tidak memilih secara acak, melainkan berdasarkan perhitungan Q-value.

6. Pembaruan Q-Value Berdasarkan Pengalaman Baru Setelah siswa menjawab soal, sistem mencatat reward, memperbarui state siswa (s'), dan memperbarui Q-Value dari pasangan (s, a) sesuai pengalaman tersebut dengan rumus sebelumnya. Proses ini berlangsung terus-menerus selama proses belajar berlangsung.

7. Perulangan Proses Hingga Konvergen Langkah-langkah di atas terus diulang sampai sistem mencapai kondisi konvergen, yaitu ketika nilai Q-Value pada Q-Table sudah stabil (tidak banyak berubah lagi), atau proses pelatihan telah selesai. Ketika konvergen, sistem dianggap telah "belajar" dan mampu merekomendasikan soal yang sesuai kemampuan siswa.

8. Istilah Tambahan

   - s_t : state pada waktu ke-t, kondisi siswa saat mengerjakan soal ke-t
   - a_t : action pada waktu ke-t, soal yang dipilih sistem saat itu

- α dan γ adalah parameter yang ditentukan oleh perancang sistem. Nilai umum seperti $\alpha = 0{,}1$ dan $\gamma = 0{,}9$ dipilih karena terbukti stabil dan efektif, terutama dalam konteks edukasi
- Agent dalam Reinforcement Learning adalah entitas yang memilih action berdasarkan Q-Table, menerima reward, dan memperbarui nilai Q-value

9. Akurasi dan Q-Value Akurasi siswa, misalnya 80% dari 10 soal (8 benar), tidak langsung menentukan Q-Value. Namun, digunakan untuk menentukan reward. Jika akurasi $\geq 80\%$, maka reward diberikan sebagai $r = 1$, yang selanjutnya memengaruhi pembaruan nilai Q-Value.

2.4 Adaptive Learning

Adaptive learning merupakan pendekatan pembelajaran yang menyesuaikan materi atau tingkat kesulitan soal berdasarkan kemampuan dan performa masing-masing pelajar. Salah satu pendekatan yang efektif dalam mendukung pembelajaran adaptif ini adalah dengan menggunakan *Reinforcement Learning* (RL) [21]. Dalam konteks penelitian ini, *Reinforcement Learning* digunakan untuk membangun logika adaptasi soal berdasarkan interaksi langsung antara pelajar dan sistem.

Penerapan *Reinforcement Learning* dalam sistem *Intelligent Tutoring System* (ITS) memungkinkan sistem untuk memilih tingkat kesulitan soal secara dinamis melalui mekanisme *trial-and-error*. Model RL belajar dari pengalaman sebelumnya dengan mengevaluasi tindakan (soal yang diberikan), hasil (jawaban benar atau salah), dan reward (umpan balik positif/negatif) untuk memutuskan soal berikutnya yang paling sesuai dengan kondisi pelajar.

Implementasi adaptive learning pada sistem ITS dalam penelitian ini difokuskan pada algoritma *Q-Learning*, yang secara eksplisit membentuk *Q-table* berdasarkan kombinasi state dan action. State direpresentasikan oleh progres pelajar pada topik tertentu, sementara action merepresentasikan pemilihan tingkat kesulitan soal (easy, medium, hard). Berdasarkan reward yang diterima, sistem akan memperbarui nilai Q menggunakan fungsi pembaruan sebagai berikut:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right] \tag{2.1}$$

di mana α adalah *learning rate*, γ adalah *discount factor*, dan r_t merupakan reward yang diberikan sesuai hasil jawaban.

Sistem ITS yang dikembangkan dalam proyek ini menggunakan komponen backend berbasis Flask untuk mengimplementasikan logika adaptasi ini, dengan perhitungan Q-Value yang terletak dalam file `q_learning_agent.py`. Proses integrasi ini terhubung langsung dengan alur pengerjaan soal pada frontend, di mana setelah pelajar menjawab soal, backend menghitung reward dan menentukan tingkat kesulitan soal berikutnya berdasarkan nilai Q-Value yang diperbarui.

Dengan pendekatan ini, sistem ITS dapat memberikan pengalaman belajar yang lebih personal dan relevan. Pelajar yang menunjukkan performa tinggi akan memperoleh soal dengan tingkat kesulitan yang lebih menantang, sedangkan pelajar yang kesulitan akan tetap berada di tingkat yang sesuai hingga performanya membaik. Hal ini selaras dengan tujuan utama adaptive learning, yaitu menciptakan jalur pembelajaran yang fleksibel dan efisien bagi setiap individu berdasarkan data interaktif yang dikumpulkan secara langsung dari proses pembelajaran.

2.5 Pengujian Aplikasi

Pengujian aplikasi *Intelligent Tutoring System* (ITS) dilakukan untuk memastikan bahwa seluruh komponen sistem berjalan sesuai dengan fungsinya, khususnya pada proses personalisasi soal berbasis algoritma *Reinforcement Learning* (RL). Pengujian yang dilakukan meliputi validasi fungsional, evaluasi efisiensi sistem, dan pengamatan terhadap perilaku adaptasi tingkat kesulitan soal berdasarkan interaksi pelajar [22].

2.5.1 Pengujian Fungsional dengan *Black Box Testing*

Metode *Black Box Testing* digunakan untuk mengevaluasi apakah fitur-fitur utama dalam aplikasi, seperti autentikasi pengguna, pemilihan materi, penampilan soal, dan pengolahan jawaban, berjalan sesuai dengan spesifikasi tanpa melihat implementasi internal kode program. Pada pengujian ini, pelajar berinteraksi langsung dengan sistem melalui antarmuka web, mengerjakan soal, dan menerima respons adaptif dari sistem [23]. Hasilnya menunjukkan bahwa sistem berhasil memberikan soal sesuai topik dan tingkat kesulitan yang ditentukan oleh kebijakan *Q-Learning*, sesuai dengan konfigurasi dan integrasi dalam modul backend Python (`data_processor.py`, `rl_agent.py`).

2.5.2 Pengujian Efisiensi dengan *Performance Benchmarking*

Selain pengujian fungsionalitas, evaluasi performa sistem sangat penting, khususnya ketika sistem menggunakan algoritma *Reinforcement Learning* yang bersifat iteratif dan memerlukan banyak proses komputasi [24]. Oleh karena itu, dilakukan *Performance Benchmarking* untuk mengukur efisiensi sistem, termasuk waktu respons saat menentukan tindakan (soal) berdasarkan nilai *Q-table*. Proses ini memastikan bahwa pemrosesan keputusan *RL* tidak menghambat pengalaman belajar pelajar, terutama ketika sistem digunakan.

2.5.3 Evaluasi Adaptivitas Sistem

Pengujian adaptivitas dilakukan dengan mensimulasikan pengguna dengan tingkat performa berbeda-beda. Dari log interaksi pengguna yang tercatat dalam tabel `UserProgress` dan `AttemptLog`, terlihat bahwa sistem mampu menyesuaikan tingkat kesulitan soal secara progresif. Misalnya, pelajar dengan rasio jawaban benar tinggi cenderung mendapatkan soal dengan level *medium* hingga *hard* lebih awal, sedangkan pelajar dengan performa rendah tetap diarahkan pada soal level *easy*. Evaluasi ini mengindikasikan bahwa kebijakan pemilihan aksi berdasarkan fungsi `recommend_next_questions()` berhasil merepresentasikan prinsip eksploitasi dan eksplorasi dalam algoritma Q-Learning.