

Q-Learning 기반 지능형 협상 에이전트

상세 이론 및 최종 설계 명세서

전민규 선임연구원

(주) 에이아이오투오

수정 날짜: March 6, 2026

버전: v4

Contents

Contents	2
1 소개	3
2 Q-Learning 이론적 배경	3
2.1 강화학습의 기본 요소	3
2.2 가치 함수와 벨만 기대 방정식 유도	3
2.3 최적 가치 함수와 벨만 최적 방정식	4
2.4 Q-Learning 업데이트 규칙	5
3 State (상태) 설계	5
3.1 State 변수 정의	5
3.2 Q-Table 크기	5
4 보상함수 (Reward Function) 설계	5
4.1 설계 철학	5
4.1.1 가격 보상 (R_{price})	6
4.1.2 종료 보상 (R_{end})	6
4.1.3 동적 가중치 (W) 재설계 (5-변수 기반)	7
4.1.4 라운드 페널티 (R_{penalty})	7
5 정책(Policy) 및 UCB 이론 배경	8
5.1 UCB (Upper Confidence Bound) 정책	8
5.2 탐험 상수 (c) 설계 가이드	9
5.3 Q-Learning의 Off-Policy 특성	9
6 운영 전략	9
6.1 운영 순환 구조	9
6.2 주요 하이퍼파라미터 설계 가이드	10
7 결론	10
References	11

1 소개

본 문서는 강화학습(Reinforcement Learning)의 Q-Learning 알고리즘을 기반으로 하는 지능형 협상 에이전트의 상세 이론 및 최종 설계 명세서입니다.

본 문서의 목적은 다음과 같습니다.

- **이론적 기반 확립:** Q-Learning의 핵심인 벨만 방정식(Bellman Equation)의 유도 과정을 포함하여, 알고리즘의 근본적인 수학적 원리를 상세히 기술합니다.
- **탐험-활용 전략 심층 분석:** 에이전트의 행동 선택 정책으로 채택된 UCB(Upper Confidence Bound)의 이론적 배경을 설명하고, 핵심 하이퍼파라미터의 설계 가이드라인을 제공합니다.
- **안정적인 시스템 설계:** 이전 설계에서 발견된 논리적 모순과 수학적 불안정성을 전면 수정한 최종 시스템 아키텍처를 제시합니다. 여기에는 상태(State), 보상함수(Reward Function), 정책(Policy), 운영(Operation) 전략이 포함됩니다.
- **구현 가이드 제공:** 실제 시스템 구현에 필요한 모든 기술적 명세와 하이퍼파라미터 권장 값을 제공하여, 재현 가능하고 안정적인 시스템 구축을 지원합니다.

본 문서를 통해 독자는 협상 에이전트의 작동 원리를 깊이 이해하고, 제시된 설계를 바탕으로 실제 환경에서 효과적으로 작동하는 지능형 시스템을 구축할 수 있을 것입니다.

2 Q-Learning 이론적 배경

강화학습은 에이전트가 누적 보상을 최대화하는 방향으로 행동 정책을 학습하는 과정입니다. Q-Learning은 그 중에서도 **가치 기반(Value-Based)** 학습의 대표적인 알고리즘입니다. 이 섹션에서는 Q-Learning의 근간이 되는 벨만 방정식의 유도 과정을 상세히 기술합니다.

2.1 강화학습의 기본 요소

- **에이전트 (Agent):** 학습의 주체. 환경을 관찰하고 행동을 결정합니다.
- **환경 (Environment):** 에이전트의 외부 세계. 에이전트의 행동에 따라 상태를 변경하고 보상을 제공합니다.
- **상태 (State, s):** 특정 시점의 환경에 대한 관찰 값입니다.
- **행동 (Action, a):** 에이전트가 특정 상태에서 취할 수 있는 행위입니다.
- **보상 (Reward, r):** 에이전트가 특정 행동을 취한 결과로 환경으로부터 받는 즉각적인 신호입니다.
- **정책 (Policy, π):** 상태가 주어졌을 때 행동을 선택하는 규칙 또는 확률분포입니다. $\pi(a|s)$ 는 상태 s 에서 행동 a 를 선택할 확률입니다.

2.2 가치 함수와 벨만 기대 방정식 유도

에이전트의 목표는 즉각적인 보상이 아닌, 미래까지 고려한 누적 보상의 기댓값, 즉 **반환값(Return)**을 최대화하는 것입니다.

- **반환값 (G_t):** 시점 t 이후의 할인된 보상의 총합입니다.

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \quad (1)$$

여기서 γ 는 할인율(Discount Factor, $0 \leq \gamma \leq 1$)로, 미래 보상의 현재 가치를 결정합니다.

- **상태-가치 함수 ($V_\pi(s)$):** 정책 π 를 따를 때, 상태 s 에서부터 시작하여 받을 것으로 기대되는 반환값입니다.

$$V_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] \quad (2)$$

- **행동-가치 함수 ($Q_\pi(s, a)$):** 정책 π 를 따를 때, 상태 s 에서 행동 a 를 취한 후 받을 것으로 기대되는 반환값입니다.

$$Q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \quad (3)$$

가치 함수는 다음과 같은 재귀적 관계를 가집니다. 이를 **벨만 기대 방정식(Bellman Expectation Equation)**이라 합니다.

- $V_\pi(s)$ 의 유도:

$$\begin{aligned} V_\pi(s) &= \mathbb{E}_\pi[G_t | S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r | s, a) [r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s']] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r | s, a) [r + \gamma V_\pi(s')] \end{aligned}$$

- $Q_\pi(s, a)$ 의 유도:

$$\begin{aligned} Q_\pi(s, a) &= \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \\ &= \sum_{s', r} p(s', r | s, a) [r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s']] \\ &= \sum_{s', r} p(s', r | s, a) [r + \gamma V_\pi(s')] \end{aligned}$$

2.3 최적 가치 함수와 벨만 최적 방정식

강화학습의 목표는 수많은 정책 π 중에서 가치 함수를 최대화하는 **최적 정책(π_*)**을 찾는 것입니다.

- **최적 상태-가치 함수 ($V_*(s)$):** $V_*(s) = \max_\pi V_\pi(s)$
- **최적 행동-가치 함수 ($Q_*(s, a)$):** $Q_*(s, a) = \max_\pi Q_\pi(s, a)$

최적 가치 함수는 **벨만 최적 방정식(Bellman Optimality Equation)**을 만족합니다.

- $V_*(s)$: 상태 s 의 최적 가치는, 가능한 모든 행동 중 가장 높은 Q-값을 선택하는 것과 같습니다.

$$V_*(s) = \max_a Q_*(s, a) \quad (4)$$

- $Q_*(s, a)$: 상태 s 에서 행동 a 의 최적 가치는, 즉시 받는 보상과 다음 상태 s' 에서 얻을 수 있는 **최대** 미래 가치의 합에 대한 기댓값입니다.

$$Q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma V_*(S_{t+1}) | S_t = s, A_t = a] \quad (5)$$

이를 $V_*(s') = \max_{a'} Q_*(s', a')$ 관계로 치환하면 Q-Learning의 핵심 방정식을 얻습니다.

$$Q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma \max_{a'} Q_*(S_{t+1}, a') | S_t = s, A_t = a] \quad (6)$$

2.4 Q-Learning 업데이트 규칙

Q-Learning은 벨만 최적 방정식을 **시간차 학습(Temporal-Difference, TD)**을 통해 반복적으로 풀어냅니다. 환경의 모델($p(s', r|s, a)$)을 모르기 때문에, 실제 경험 (s, a, r, s') 을 통해 Q값을 점진적으로 업데이트합니다.

- **TD 타겟 (새로운 추정치):** $r + \gamma \max_{a'} Q(s', a')$
- **TD 에러 (오차):** $(r + \gamma \max_{a'} Q(s', a')) - Q(s, a)$

최종 업데이트 규칙은 현재 Q-값을 TD 타겟 방향으로 학습률(α)만큼 이동시키는 형태가 됩니다.

$$Q(s, a) \leftarrow Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \quad (7)$$

이것이 바로 Q-Table의 각 셀 값을 업데이트하는 핵심 공식입니다.

3 State (상태) 설계

상태는 에이전트가 의사결정을 내리는 데 필요한 모든 정보를 압축하여 표현한 것입니다. 사용자의 최종 요구사항을 정확히 반영하여, 5개의 핵심 변수의 조합으로 State를 재구성합니다.

3.1 State 변수 정의

State는 다음 5가지 핵심 변수의 조합으로 구성됩니다.

3.2 Q-Table 크기

- **총 상태 수:** 3 (매출액) $\times$ 3 (유통구조) $\times$ 3 (파트너사) $\times$ 3 (수용률) $\times$ 2 (입력금액) = **162개**
- **총 행동 수:** 9개 (가정)
- **최종 Q-Table 크기:** $162 \times 9 = 1458$ 개 셀

State 공간이 162개로 정의됨에 따라, Q-Table을 충분히 채우기 위해 더 많은 데이터와 학습 시간이 필요합니다. 각 State를 방문하는 빈도를 모니터링하여 데이터 부족 문제를 관리해야 합니다.

4 보상함수 (Reward Function) 설계

4.1 설계 철학

- 1) **명확한 범위:** 보상 값의 범위를 사전에 정의하여 학습 안정성을 확보하고, 불필요한 실시간 정규화 과정을 제거합니다.
- 2) **균형된 인센티브:** 새로운 5-변수 State 구조를 활용하여 '가격 성과'와 '협상 종료'라는 상충하는 목표 간의 균형을 더욱 정교하게 동적으로 조절합니다.
- 3) **현실성 반영:** 협상 실패 페널티, 앵커링 목표 초과 달성 보너스, 협상 길이 비용 등 현실적인 요소를 반영합니다.

변수	구분 (Levels)	설명
1. 매출액 가격구간	3개	협상 대상의 전체 금액 규모를 나타내는 정적 변수입니다. Low: 1,000만원 이하 Mid: 1,000만원 초과 3,000만원 이하 High: 3,000만원 초과
2. 유통 구조	3개	상품의 유통 단계를 나타내는 정적 변수입니다. 제조: 제조사 직접 공급 총판: 총판 경유 유통: 유통사 경유
3. 파트너사 종류	3개	상품을 공급하는 파트너사의 경쟁 구조를 나타내는 정적 변수입니다. Single: 단독 파트너사 Multiple: 다수 파트너사 None: 파트너사 없음
4. 가격 수용률 구간	3개	현재까지의 가격 할인율을 3개 구간으로 분류하는 동적 변수입니다. Low: 3% 이하 Mid: 3% 초과 ~ 6% 이하 High: 6% 초과
5. 입력 금액 구간	2개	현재 제안가(P)가 초기 목표가(T) 대비 어느 위치에 있는지를 나타내는 동적 변수입니다. ($A < T$) PZ1 (Success): $A \leq P \leq T$ PZ2 (Fail): $P > T$

Table 1: 최종 State 변수 정의 (5개 변수)

최종 보상 R 은 다음 네 가지 구성 요소의 조합으로 계산됩니다.

$$R = W \times R_{\text{price}} + (1 - W) \times R_{\text{end}} - R_{\text{penalty}} \quad (8)$$

4.1.1 가격 보상 (R_{price})

현재 제안가 P 의 위치에 따라 3단계로 차등 계산됩니다.

- **앵커링 목표 초과 달성 ($P < A$):** 기본 보상 1.0에 추가 보너스(β)를 지급합니다.

$$R_{\text{price}} = 1.0 + \beta \cdot \frac{A - P}{A} \quad (9)$$

- **목표 범위 내 ($A \leq P \leq T$):** $P = T$ 일 때 0, $P = A$ 일 때 1이 되는 선형적인 보상을 지급합니다.

$$R_{\text{price}} = \frac{T - P}{T - A} \quad (10)$$

- **목표 미달성 ($P > T$):** 가격 보상은 0입니다. (페널티는 별도 고려 가능)

$$R_{\text{price}} = 0 \quad (11)$$

4.1.2 종료 보상 (R_{end})

협상의 최종 결과를 반영합니다.

- 성공 (Success): 1.0
- 진행 중 (Ongoing): 0.0
- 실패 (Failure): -0.5 (권장: -0.5 -1.0)

4.1.3 동적 가중치 (W) 재설계 (5-변수 기반)

동적 가중치 W 는 '가격을 얼마나 더 중시할 것인가'를 결정하는 핵심 파라미터입니다. 새로운 5-변수 State 구조를 반영하여 다음과 같이 재설계합니다.

- 계산 로직: 각 State 변수를 정규화된 값(S)으로 변환하고, 이를 메타 가중치(w_i)와 함께 가중 합산하여 W_{raw} 를 계산합니다.

$$W_{\text{raw}} = w_1 S_{\text{amount}} + w_2 S_{\text{manu}} + w_3 S_{\text{partner}} + w_4 S_{\text{accept}} + w_5 S_{\text{pricezone}} \quad (12)$$

- State 변수별 가중치 (S):

변수	정규화 값 (S)	설계 의도
1. 매출액 가격구간	Low: 0.2, Mid: 0.5, High: 1.0	금액이 클수록 가격의 중요도가 높다고 판단합니다.
2. 유통 구조	제조: 0.2, 총판: 1.0, 유통: 0.5	대기업 제품은 가격 협상 여지가 적고, 중소기업 제품은 크다고 판단합니다.
3. 파트너사 종류	Single: 0.2, Multiple: 1.0, None: 0.5	경쟁사가 많은 '다수' 구조에서 가격 협상의 여지가 가장 크다고 판단합니다.
4. 가격 수용률 구간	Low: 1.0, Mid: 0.5, High: 0.2	아직 할인이 충분히 이루어지지 않은 '낮음' 상태에서 가격 압박의 필요성이 가장 높다고 판단합니다.
5. 입력 금액 구간	PZ1 (Success): 0.2, PZ2 (Fail): 1.0	이미 목표가 내에 진입한 '성공' 상태에서는 타결이 중요하므로 가중치를 낮춥니다.

Table 2: 동적 가중치 계산을 위한 State 변수별 가중치

- 메타 가중치 (w_i): 각 State 변수의 중요도를 조절하는 하이퍼파라미터입니다. (기본값 권장: $\sum w_i = 1$)
 - w_1 (매출액 가격구간): 0.1
 - w_2 (유통 구조): 0.2
 - w_3 (파트너사 종류): 0.3
 - w_4 (가격 수용률 구간): 0.3
 - w_5 (입력 금액 구간): 0.1
- 최종 범위 제한 (Clamping): 이전 설계와 동일하게, 극단적인 경우를 방지하기 위해 최종 W 값의 범위를 제한합니다.

$$W = \max(0.2, \min(0.8, W_{\text{raw}})) \quad (13)$$

4.1.4 라운드 페널티 (R_{penalty})

협상이 길어지는 것에 대한 비용을 부과하여 효율적인 타결을 유도합니다.

$$R_{\text{penalty}} = \lambda \times \text{RoundNumber} \quad (14)$$

(λ : 비용 계수, 권장: 0.01 ~ 0.03)

5 정책(Policy) 및 UCB 이론 배경

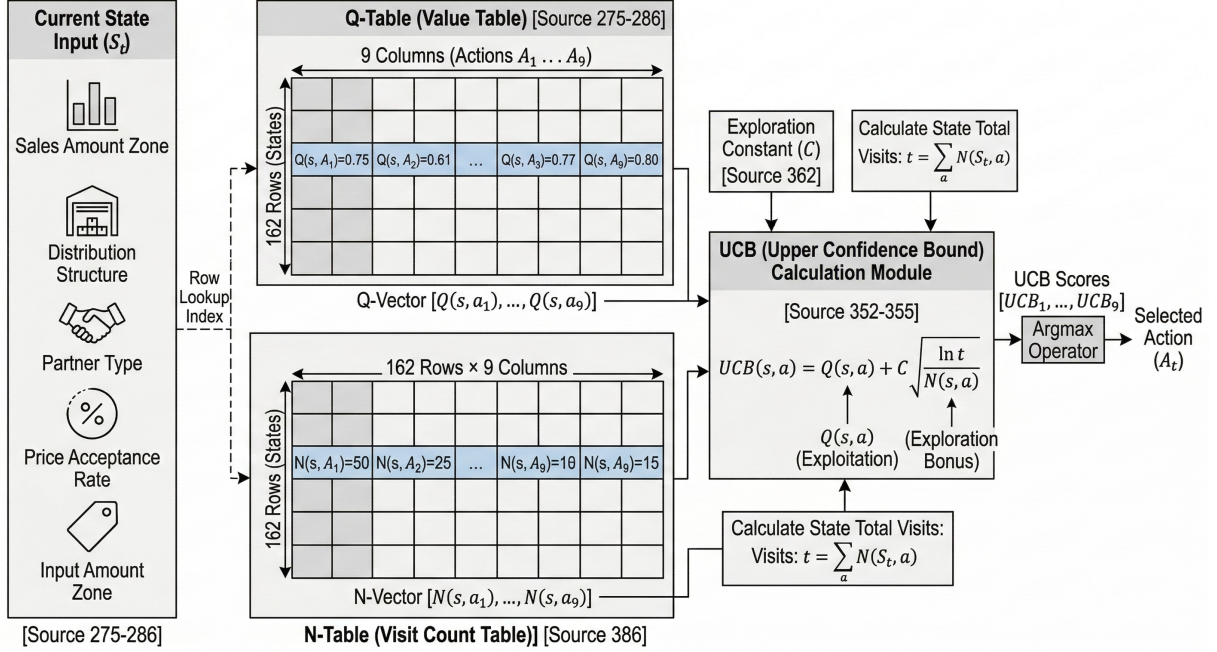


Figure 1: Q-Learning 기반 에이전트의 상태 입력, Q/N-테이블 및 UCB(Upper Confidence Bound) 기반 행동 선택 구조

정책은 에이전트가 특정 상태에서 어떤 행동을 취할지 결정하는 규칙입니다. 강화학습의 중요한 과제는 탐험(Exploration)과 활용(Exploitation) 사이의 균형을 맞추는 것입니다.

- **활용:** 현재까지 학습된 최적의 행동을 선택하여 단기 보상을 극대화합니다.
- **탐험:** 새로운 행동을 시도하여 더 나은 장기적 보상을 찾을 가능성을 모색합니다.

5.1 UCB (Upper Confidence Bound) 정책

본 시스템은 탐험과 활용의 균형을 효과적으로 맞추는 UCB (Upper Confidence Bound) 정책을 채택합니다. UCB는 "불확실성에 대한 낙관주의(Optimism in the Face of Uncertainty)" 원칙에 기반합니다 [3].

UCB는 각 행동의 가치를 추정할 때, 현재까지의 평균 보상(Q-값)에 불확실성 보너스를 더하여 평가합니다. 이 보너스는 해당 행동이 적게 선택될수록 커집니다.

$$a_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}} \left[\underbrace{Q_t(a)}_{\text{활용}} + \underbrace{c \cdot \sqrt{\frac{\ln t}{N_t(a)}}}_{\text{탐험 (불확실성 보너스)}} \right] \quad (15)$$

- $Q_t(a)$: 시점 t 까지 행동 a 의 평균 보상 (Q-값)
- t : 총 시도 횟수 (모든 행동에 대해)
- $N_t(a)$: 시점 t 까지 행동 a 가 선택된 횟수

- **c: 탐험 상수.** 탐험의 강도를 조절하는 하이퍼파라미터입니다.

이 정책은 Q-값이 높으면서도(활용), 동시에 적게 시도되어 불확실성이 높은(탐험) 행동에 보너스를 부여하여 선택될 기회를 줍니다.

5.2 탐험 상수 (c) 설계 가이드

탐험 상수 c 는 UCB 알고리즘의 성능에 결정적인 영향을 미칩니다.

- **이론적 배경 (Hoeffding's Inequality):** UCB의 이론적 근거는 회프딩 부등식에서 나옵니다. 이는 샘플 평균이 실제 평균에서 벗어날 확률의 상한을 제공합니다. UCB의 보너스 항은 이 상한을 반영하여, 실제 가치가 현재 추정치보다 높을 가능성을 고려합니다.
- **c의 역할:**
 - **c가 크면:** 탐험을 더 강조합니다. 에이전트는 불확실성이 높은(적게 선택된) 행동을 더 자주 시도합니다. 수렴은 느려지지만, 최적해를 놓칠 위험은 줄어듭니다.
 - **c가 작으면:** 활용을 더 강조합니다. 에이전트는 현재 가장 좋다고 알려진 행동을 더 자주 선택합니다. 수렴은 빠르지만, 차선택에 갇힐(Local Optima) 위험이 커집니다.
- **권장 값 및 튜닝 전략:**
 - **이론적 권장 값:** 많은 연구에서 $c = \sqrt{2}$ 를 이론적으로 좋은 출발점으로 제안합니다. 이는 리그렛(Regret)의 상한을 최소화하는 값 중 하나입니다.
 - **실용적 튜닝:** 실제 문제에서는 c 값을 데이터에 맞게 튜닝해야 합니다. 일반적인 튜닝 범위는 $[0.1, 2.0]$ 입니다. 그리드 서치(Grid Search)나 베이지안 최적화(Bayesian Optimization)를 사용하여 오프라인 환경에서 최적의 c 값을 찾을 수 있습니다.
 - **동적 조절:** 학습 초기에는 c 를 높게 설정하여 충분한 탐험을 유도하고, 학습이 진행됨에 따라 c 를 점차 줄여나가는 동적 스케줄링(Dynamic Scheduling) 전략도 효과적일 수 있습니다.

5.3 Q-Learning의 Off-Policy 특성

Q-Learning은 Off-Policy 알고리즘으로, 행동 선택 정책(Behavior Policy)과 학습 대상 정책(Target Policy)을 분리합니다.

- **행동 정책:** UCB 정책 (탐험 포함)
- **타겟 정책:** Greedy 정책 ($\max Q$)

Q-러닝 업데이트 규칙의 $\max_{a'} Q(s', a')$ 항은, 에이전트가 UCB에 따라 실제로 어떤 탐험적 행동을 했는지와 무관하게, "만약 다음 상태에서 최적으로 행동했다면 얻었을 가치"를 의미합니다. 이 덕분에 에이전트는 탐험의 결과에 휘둘리지 않고 일관되게 최적의 Q-값을 학습하여 수렴할 수 있습니다.

6 운영 전략

6.1 운영 순환 구조

1. **[Deploy]** 현재 학습된 Q 테이블과 N 테이블(방문 횟수)을 서비스에 배포합니다. 에이전트는 UCB 정책에 따라 행동을 선택합니다.
2. **[Collect]** 일정 기간(예: M 회 상호작용) 동안의 (s, a, R, s') 데이터를 `new_data_buffer`에 수집합니다.
3. **[Aggregate]** 수집된 `new_data_buffer`를 마스터 데이터셋 D 에 추가하고, N 테이블을 업데이트합니다.

4. [Train] 수정된 보상함수를 사용하여 마스터 데이터셋 D 전체에 대한 보상(R)을 재계산하고, 이를 기반으로 Q-Table을 오프라인에서 재학습합니다.
5. 1번으로 돌아가 새로 학습된 Q 테이블을 배포합니다.

6.2 주요 하이퍼파라미터 설계 가이드

파라미터	기호	권장 값/범위	설계 가이드
학습률	α	0.1 ~ 0.3	값이 크면 최신 정보에 민감하게 반응하나 불안정할 수 있고, 작으면 학습이 느리지만 안정적입니다. 초기에는 높게, 점차 줄여나가는 스케줄링이 효과적입니다.
할인율	γ	0.9 ~ 0.99	미래 보상의 중요도를 결정합니다. 1에 가까울수록 장기적인 보상을 더 중요하게 고려합니다. 협상과 같이 최종 결과가 중요한 경우 높은 값을 사용합니다.
탐험 상수	c	≈ 1.414	UCB 탐험 강도를 조절합니다. 이론적 값에서 시작하여 오프라인 테스트를 통해 문제에 맞는 최적 값을 찾아야 합니다.
데이터 수집 주기	M	3,000 ~ 10,000	재학습 주기를 결정합니다. 너무 잦으면 불안정하고, 너무 길면 최신 데이터 반영이 늦어집니다. 시스템 트래픽과 안정성 사이의 트레이드오프를 고려해야 합니다.
에포크	K	5 ~ 20	재학습 시 전체 데이터셋을 몇 번 반복할지 결정합니다. 데이터셋이 클수록 적은 에포크로도 충분히 수렴할 수 있습니다.

Table 3: 하이퍼파라미터 설계 가이드

7 결론

본 최종 설계 문서는 Q-Learning 기반 협상 에이전트의 견고하고 안정적인 구현을 위한 상세한 이론적 배경과 기술적 청사진을 제시합니다. 핵심적인 개선 사항은 다음과 같습니다.

- **이론적 깊이 강화:** 벨만 방정식의 상세 유도 과정과 UCB 정책의 이론적 배경 및 하이퍼파라미터 설계 가이드를 포함하여, 알고리즘의 근본 원리에 대한 깊은 이해를 제공합니다.
- **보상함수의 안정성 확보:** 보상 범위를 명확히 정의하고, 협상 실패 및 길이 비용 등 현실적 요소를 반영하여 수학적 안정성과 논리적 타당성을 모두 확보했습니다.
- **Off-Policy 명확화:** Q-Learning의 Off-Policy 특성을 명시적으로 설명하여, 탐험(UCB)과 학습(Greedy)이 어떻게 조화롭게 작동하는지 밝혔습니다.

이 설계를 기반으로 구현된 에이전트는 예측 가능하고 안정적으로 학습을 수행하며, '가격 성과'와 '효율적 타결'이라는 두 가지 목표 사이에서 최적의 균형을 찾아내는 지능형 협상 파트너로 기능할 것입니다.

References

- [1] Watkins, C. J. C. H., Dayan, P. (1992). Q-learning. *Machine learning*, 8(3-4), 279-292. <https://link.springer.com/article/10.1007/BF00992698>
- [2] Sutton, R. S., Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press. <http://incompleteideas.net/book/the-book-2nd.html>
- [3] Auer, P., Cesa-Bianchi, N., Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2), 235-256. <https://link.springer.com/article/10.1023/A:1013689704352>