1 Logic and AI 2: First-Order Logic (FOL)

1.1 Einführung und Motivation

Die Aussagenlogik (Propositional Logic) hat erhebliche Einschränkungen. Sie behandelt Fakten atomar und besitzt kein Verständnis für Objekte und deren Beziehungen untereinander.

- In der Aussagenlogik haben Terme keine formale Bedeutung, nur eine intuitive (z. B. ist "RoommateCarryingUmbrella" technisch gesehen nur eine Variable P).
- **Prädikatenlogik erster Stufe** (First-Order Logic, FOL) führt Objekte, Variablen und Quantoren ein, um Wissen kompakter und strukturierter darzustellen.

1.2 Syntax und Elemente der FOL

Die FOL baut auf folgenden Grundbausteinen auf:

Elemente der FOL

- **Objekte:** Konstanten, die spezifische Entitäten bezeichnen (z. B. $John, Umbrella_0, Earth$).
- **Relationen (Prädikate):** Beziehungen zwischen Objekten.
  - *Unäre Relationen*: Eigenschaften eines Objekts (z. B. $IsUmbrella(x)$).
  - *n-äre Relationen*: Beziehungen zwischen n Objekten (z. B. $Carrying(John, Umbrella_0)$).
- **Funktionen:** Abbildungen, die sich auf Objekte beziehen, ohne sie explizit zu benennen (z. B. $Roommate(Person_0)$ oder $LeftLegOf(John)$). Wichtig: Eine Funktion liefert ein Objekt zurück, keinen Wahrheitswert.
- **Gleichheit:** $Term_1 = Term_2$ (z. B. $Roommate(Person_0) = Person_1$).

1.2.1 Quantoren

Um Aussagen über Mengen von Objekten zu treffen, werden Variablen (x, y, z) und Quantoren verwendet:

1. **Allquantor** ($\forall$): "Für alle..."
   - Beispiel: Alle Löwen sind Katzen.
   - $\forall x : Lion(x) \implies Cat(x)$
   - *Hinweis:* Der Allquantor wird meistens mit der Implikation ($\implies$) verwendet.
2. **Existenzquantor** ($\exists$): "Es existiert (mindestens) ein..."
   - Beispiel: Es gibt eine Katze, die kein Löwe ist.
   - $\exists x : Cat(x) \land \neg Lion(x)$
   - *Hinweis:* Der Existenzquantor wird meistens mit der Konjunktion ($\land$) verwendet.

Dualität der Quantoren: Die Quantoren lassen sich ineinander umformen:

$$\forall x : P(x) \equiv \neg \exists x : \neg P(x)$$

$$\exists x : P(x) \equiv \neg \forall x : \neg P(x)$$

1.3 Modellierung: Von natürlicher Sprache zu FOL

Das Übersetzen von Sätzen erfordert präzise Muster. Hier einige wichtige Beispiele und Strukturen (auch basierend auf Übungsaufgaben):

- **Einfache Eigenschaft:** "John hat einen Regenschirm" $\exists y : (Has(John, y) \wedge IsUmbrella(y))$
- **Verschachtelung:** "Jede Person, die einen Schirm hat, ist nicht nass" $\forall x : (Person(x) \implies ((\exists y : Has(x, y) \wedge Umbrella(y)) \implies \neg Wet(x)))$
- **Mindestens zwei (Distinctness):** "John hat mindestens zwei Schirme" $\exists x, y : (Has(John, x) \wedge Umbrella(x) \wedge Has(John, y) \wedge Umbrella(y) \wedge \neg(x = y))$
- **Höchstens zwei:** "John hat höchstens zwei Schirme" $\forall x, y, z : ((Has(John, x) \ldots \wedge Has(John, z) \ldots) \implies (x = y \vee x = z \vee y = z))$
- **Genau ein (Uniqueness):** "Es liegt genau eine Münze in der Kiste" $\exists x(Coin(x) \wedge InBox(x) \wedge \forall y((Coin(y) \wedge InBox(y)) \implies x = y))$

1.4 Inferenz in FOL

1.4.1 Substitution (SUBST)

Um logische Schlüsse zu ziehen, müssen Variablen oft durch konkrete Terme ersetzt werden.

- Syntax: $SUBST(\{x/John\}, IsHealthy(x)) \to IsHealthy(John)$
- **Universal Instantiation:** Aus $\forall x : \alpha$ kann $SUBST(\{x/g\}, \alpha)$ für jeden Grundterm g abgeleitet werden.
- **Existential Instantiation:** Aus $\exists x : \alpha$ kann $SUBST(\{x/k\}, \alpha)$ abgeleitet werden, wobei k eine **neue** Konstante ist (Skolem-Konstante), die noch nicht in der Wissensbasis vorkommt.

1.4.2 Skolemierung

Die Eliminierung von Existenzquantoren ist ein zentraler Schritt für Resolutionsbeweise. Dabei ist die Position des Quantors entscheidend:

Skolemierung Regeln

1. **Existenzquantor steht allein (oder ganz außen):** Ersetze die Variable durch eine neue Konstante (Skolem-Konstante).
$$\exists x : P(x) \to P(A)$$
2. **Existenzquantor steht im Wirkungsbereich eines Allquantors:** Die existierende Variable hängt von der allquantifizierten Variable ab. Ersetze y durch eine **Skolem-Funktion** $f(x)$.
$$\forall x \exists y : IsParentOf(x, y) \to \forall x : IsParentOf(x, f(x))$$
Hierbei bildet $f(x)$ das passende y für jedes x ab.

1.4.3 Unifikation

Ein Algorithmus, der eine Substitution θ findet, sodass zwei Ausdrücke identisch werden. Dies ist die Grundlage für den verallgemeinerten Modus Ponens und die Resolution. Beispiel:

- Term 1: $Knows(John, x)$
- Term 2: $Knows(y, Jane)$
- Unifikator $\theta = \{x/Jane, y/John\}$ resultiert in $Knows(John, Jane)$.

1.5 Resolution in FOL

Wie in der Aussagenlogik ist die Resolution eine Widerlegungsmethode (Beweis durch Widerspruch). Um sie anzuwenden, müssen Sätze in die **Konjunktive Normalform (CNF)** gebracht werden.

Umwandlung in First-Order CNF

1. **Implikationen eliminieren:** $A \implies B$ wird zu $\neg A \lor B$.
2. **Negationen nach innen ziehen:** (De Morgan Regeln), sodass $\neg$ nur direkt vor Atomen steht.
3. **Variablen standardisieren:** Umbenennen, sodass jeder Quantor eindeutige Variablennamen verwendet.
4. **Skolemierung:** Eliminierung aller Existenzquantoren (durch Konstanten oder Funktionen).
5. **Allquantoren verwerfen:** Da nun alle Variablen allquantifiziert sind, können die $\forall$ weggelassen werden (implizite Annahme).
6. **Verteilung (Distributivgesetz):** Umwandlung in Konjunktion von Disjunktionen (UND von ODERs).
7. **Klauselbildung:** Darstellung als Menge von Klauseln.

Beispielablauf: Satz: "Jeder, der nicht getötet wird und isst, ist Nahrung"

$$\forall x \forall y : (Eats(x, y) \land \neg Killed(x)) \implies Food(y)$$

$\rightarrow$ Implikation weg: $\neg(Eats(x, y) \land \neg Killed(x)) \lor Food(y)$ $\rightarrow$ De Morgan: $\neg Eats(x, y) \lor Killed(x) \lor Food(y)$ Dies ist bereits eine Klausel.

1.6 Prolog (Programming in Logic)

Prolog ist eine Programmiersprache, die auf FOL (speziell Horn-Klauseln) basiert und Unifikation sowie Backtracking-Suche verwendet.

- **Fakten:** 'eats(sam, dal).' (Entspricht $Eats(Sam, Dal)$).
- **Regeln:** 'head :- body.' (Bedeutet: Head ist wahr, *wenn* Body wahr ist. Logisch: $Body \implies Head$).
- **Syntax:**
  - *Großbuchstaben* sind Variablen (z. B. 'Person', 'X').
  - *Kleinbuchstaben* sind Atome/Konstanten (z. B. `sam`, `curry`).
  - _ ist eine anonyme Variable.
  - Komma **,** bedeutet UND (Konjunktion).
- **Inferenz:** Prolog nutzt Tiefensuche (Depth-First Search) mit Backtracking. Wenn eine Unifikation fehlschlägt, geht das System zurück und probiert die nächste Alternative.

1.7 Grenzen der Prädikatenlogik

- **Entscheidbarkeit:** FOL ist **semi-entscheidbar**.
  - Wenn ein Satz aus der Wissensbasis folgt, existiert ein Algorithmus, der dies beweist (er terminiert).
  - Wenn ein Satz *nicht* folgt, terminiert der Algorithmus möglicherweise nie.
- **Ausdrucksstärke:** Man kann nicht über Relationen selbst quantifizieren (z. B. "Für alle Eigenschaften p..."). Dies würde Logik höherer Stufe (Higher-Order Logic) erfordern.
- **Gödels Unvollständigkeitssatz:** In jedem formalen System, das mächtig genug ist, um Arithmetik (Induktion) abzubilden, gibt es wahre Aussagen, die innerhalb des Systems nicht beweisbar sind. Vollständigkeit in komplexen Systemen ist unmöglich.

1.8 Neuro-Symbolic AI (Ausblick)

Moderne Ansätze versuchen, die logische Schlussfolgerung (Symbole, Prolog) mit neuronalen Netzen (Wahrnehmung, Lernen) zu kombinieren.

- Ziel: Ein System, das sowohl aus Daten lernt (Gradient Descent) als auch logische Regeln befolgt.

- Beispiel: Ein Agent, der Pixeldaten wahrnimmt (Neural), aber Entscheidungen auf Basis logischer Regeln (z. B. “Wenn Gegner nah, dann springen”) trifft oder diese Regeln lernt.