NL2STL: CCG 解析・意味合成を用いた確率付き信号時相論理仕様候補の生成

芹澤 和伸　伏見 洸星　橋本 和宗
大阪大学 大学院工学研究科
serizawa.kazunobu.ogp@ecs.osaka-u.ac.jp　hashimoto@eei.eng.osaka-u.ac.jp

概要

自然言語（NL）から信号時相論理（STL）仕様を生成する NL-to-STL では，意味曖昧性により複数の妥当な解釈が成立し得る場合に，複数の妥当な仕様候補を生成する必要がある．これは，解釈の違いが仕様の違いに直結するためである．本研究では，CCG に基づく *n*-best 構文木と ccg2lambda による意味合成を用いて，複数の解釈候補に対応する STL 仕様候補を生成し，各 STL 仕様候補にスコア由来の確率を付与する枠組みを提案する．

1　はじめに

信号時相論理（Signal Temporal Logic: STL）を含む時相論理（Temporal Logic: TL）は，「いつ」「どの状態が」「どの順序で」成立すべきか，といった時間的・空間的な制約を厳密な形式で記述することができる．ロボティクスや自律システムを扱う工学分野では，TL/STL 仕様に基づく経路計画や制御器設計の手法が数多く検討されている．一方で，TL/STL のシンタックスに基づいた仕様記述には一定の専門知識が求められるため，最近では自然言語（Natural Language: NL）仕様から TL/STL 仕様へ自動変換する枠組み（NL-to-TL/STL）が注目されている．

例えば，先行研究 [1] では，大規模言語モデル（Large Language Models: LLMs）を用いて大規模な{NL,TL}のペアデータを構築し，テキスト変換モデルを教師あり学習する枠組みを提示している．このような LLM を用いた学習型アプローチは，十分なデータがある状況では高精度化が期待でき，実用に向けた重要な方向性である．

しかしながら，自然言語に基づく仕様の生成では，自然言語の意味曖昧性に由来する解釈の不確定性が残る．より具体的には，自然言語文は，同一の表現であっても複数の妥当な解釈を許すことがあり，STL のように作用域（scope）や入れ子構造が意味に直結する形式仕様へ変換する際に，その不確定性が顕在化しやすい．例えば，

Within 10 seconds, reach B or reach C while avoiding A.

という文では，*while avoiding A* が *reach C* のみに係る（局所スコープ）解釈と，*reach B or reach C* 全体に係る（大域スコープ）解釈の双方が成立し得る．このような場合において，どちらか一方のみを「正解」とみなすことは必ずしも適切ではなく，むしろ複数の妥当な仕様候補を提示し，人間（仕様設計者）が選択・検証できる形にすることは，システムの信頼性や安全性を担保するうえで非常に重要である．

そこで本研究では，曖昧な NL 入力に対して複数の解釈候補に対応する STL 仕様候補集合を生成し，さらに各候補に確率（重み）を付与して提示する枠組みを構築する．具体的には，組合せ範疇文法（Combinatory Categorial Grammar: CCG）に基づく構文解析から得られる *n*-best 構文木（parse tree）を用いて複数の統語構造をもとに，`ccg2lambda` [2] により意味合成（semantic composition）を行うことにより，NL の解釈候補を明示的に列挙し，曖昧性が生じる箇所（例：修飾句の係り先）を候補分岐として保持したまま複数の STL 候補を得ることを目指す．

本研究の主な貢献は以下のとおりである．

- STL 仕様生成に適した semantic template（語彙・構文範疇への意味付与）を設計し，CCG に基づく構文解析に沿って意味合成できる形に整理する．
- *n*-best 解析により同一の NL から複数の妥当な解釈構造を導出し，曖昧性を STL 候補集合として扱う枠組みを示す．
- 候補に対してスコアに基づく重み（確率）を付与し，さらに同一 STL（あるいは同一の構造パターン）に対応する導出を集約することで，曖昧性を確率として提示できることを示す．

2 準備

2.1 STL の基礎

本節では，STL のシンタックスを簡潔に整理し，本稿で用いる原子命題（と経路計画問題における仕様の書き方をまとめる．なお，STL の標準的な定義・記法の詳細は [3, 4] などを参照されたい．

STL のシンタックス STL φ は基本命題，論理演算子，時相演算子を用いて再帰的に構成され，そのシンタックスは次のように与えられる：

$$\varphi ::= \top \mid \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \, \mathbf{U}_I \, \varphi_2. \quad (1)$$

ここで，$\top$ は恒真，論理演算子 $\neg, \wedge$ および $\vee$ は，それぞれ論理否定，論理積，論理和を表す．$\mu : \mathbb{R}^n \to \mathbb{B}$ はブール値を持つ述語であり，例えば実数値関数 $h : \mathbb{R}^n \to \mathbb{R}$ を用いて $\mu(x_t) = \text{True} \Leftrightarrow h(x_t) > 0$ のように定義できる．

時相演算子 $\mathbf{U}_I$ は，時間区間 $I = [a, b] = \{t \in \mathbb{Z}_{\geq 0} : a \leq t \leq b\}$ $(a, b \in \mathbb{Z}_{\geq 0})$ に対する *until* 演算子を表す．他の時相演算子である *eventually* 演算子 $\mathbf{F}_I$ および *always* 演算子 $\mathbf{G}_I$ は，*until* 演算子を用いて

$$\mathbf{F}_I\varphi := \top \, \mathbf{U}_I \, \varphi, \qquad \mathbf{G}_I\varphi := \neg\mathbf{F}_I\neg\varphi \quad (2)$$

と定義される．自然言語では，$\varphi_1 \mathbf{U}_I \varphi_2$ は「区間 I のどこかで φ_2 が真，それまで（直前まで）φ_1 が真」，$\mathbf{F}_I\varphi$ は「区間 I のどこかの時刻で φ が真」，$\mathbf{G}_I\varphi$ は「区間 I のすべての時刻で φ が真」を表す．

経路計画問題の STL 仕様 本稿の NL-to-STL では，経路計画問題のための STL 仕様を想定し，「対象が領域にいる（in-region）」を表す原子命題を明示的に導入する．領域ラベル集合を $\mathcal{R} = \{A, B, C, \ldots\}$ とし，各 $R \in \mathcal{R}$ に対して原子命題 ϕ_R を

$$\phi_R(x_t) = \text{True} \iff x_t \in \mathcal{X}_R \quad (3)$$

で定義する（$\mathcal{X}_R \subset \mathbb{R}^n$ は領域 R の空間集合）．以降，ϕ_A, ϕ_B, ϕ_C はそれぞれ「時刻 t に領域 A, B, C にいる」ことを表す．上記の原子命題 ϕ_R を用いると，到達・回避などの基本タスクは次のように表せる：

$$\text{時間内到達（reach } R \text{ within } I\text{）} : \mathbf{F}_I\phi_R, \quad (4)$$

$$\text{時間内回避（avoid } A \text{ over } I\text{）} : \mathbf{G}_I\neg\phi_A. \quad (5)$$

2.2 曖昧性と NL-to-STL

自然言語には，修飾句の係り先，接続（and/or）構造，前置詞句の付加位置などに起因する曖昧性がある．NL-to-STL では，これらの曖昧性が STL の作用域（scope）や入れ子構造の分岐として直接現れる．先ほど示した *Within 10 seconds, reach B or reach C while avoiding A.* という仕様は *while avoiding A* の作用域により，少なくとも次の 2 候補に分岐する：

$$\varphi_{\text{local}} := \left(\mathbf{F}_{[0,10]}\phi_B\right) \vee \left(\mathbf{F}_{[0,10]}\phi_C \wedge \mathbf{G}_{[0,10]}\neg\phi_A\right), \quad (6)$$

$$\varphi_{\text{global}} := \mathbf{F}_{[0,10]}\left(\phi_B \vee \phi_C\right) \wedge \mathbf{G}_{[0,10]}\neg\phi_A. \quad (7)$$

φ_{local} は回避が C 側の到達にのみ付随する局所スコープ，φ_{global} は回避が全体に付随する大域スコープを表す．両者はいずれも自然言語として妥当な解釈であるが，STL としては異なる制約を与えるものである．したがって本研究では，このような分岐を統語構造（構文木）の分岐として明示し，候補として列挙することを目指す．

2.3 CCG と意味合成

組合せ範疇文法（Combinatory Categorial Grammar: CCG）は，語に割り当てられた範疇（category）と少数の合成規則に基づき，文全体の統語構造を導出する枠組みである．確率的 CCG parser を用いると，最尤構文木に加えて，上位 K 個の構文木（n-best）と各構文木のスコアを得られる．本研究では，CCG の n-best 解析（depccg[5]）と ccg2lambda を組み合わせ，曖昧性が生じる箇所を候補分岐として保持したまま，STL へ変換する．

3 提案手法

3.1 問題設定

本研究では，自然言語による仕様 ℓ を入力として受け取り，複数の STL 仕様候補を生成し，各候補に確率（重み）を付与して提示する．入力文 ℓ に対する出力を確率付き STL 仕様候補集合

$$S(\ell) = \{(\varphi_j, p_j)\}_{j=1}^{m} \quad (8)$$

と定義する．ここで φ_j は STL，p_j はその確率（重み）であり $\sum_{j=1}^{m} p_j = 1$ を満たす．また候補の重複を避けるため，φ_j は相異なるもの（$j \neq k \Rightarrow \varphi_j \neq \varphi_k$）とし，自然言語の曖昧性により $m \geq 2$ となる場合に，その候補を $S(\ell)$ として可視化・提示することを目指す．

3.2 提案手法の概要

提案手法の流れを図 1 に示す．入力文 ℓ に対して CCG parser（例：depccg）の n-best 解析により構文木

集合と導出確率を得て，各構文木に対して semantic template に基づく意味合成を行い，得られた意味表現を STL へ変換したうえで同一性判定のために正準化する．最後に，同一の STL・構造パターンについて集約して，$S(\ell)$ を構成する．

3.3 原子命題とタスク表現

本稿では STL の原子命題を ϕ_R で統一し（式 (3)），到達・回避などのタスクを式 (4)–(5) のように表す．これにより，NL 側の語彙（reach, avoid, within, or, while など）が最終的にどの時相演算子（$\mathbf{F}, \mathbf{G}, \mathbf{U}$）とどの 原子命題（$\phi_A, \phi_B, \phi_C$）へ変換されるかが明確になる．

3.4 CCG による n-best 構文解析

入力文 ℓ に対して，CCG parser により上位 K 個の構文木

$$\tilde{\mathcal{T}}_K(\ell) = \{(T_i, \tilde{q}_i)\}_{i=1}^{K} \tag{9}$$

を得る．ここで T_i は構文木，$\tilde{q}_i$ は parser が出力する導出確率（導出スコアに基づいて変換された確率）である．n-best 解析では，$\sum_{i=1}^{K} \tilde{q}_i \le 1$ となり得る．以降では，n-best の範囲で確率が総和 1 となるように

$$q_i := \frac{\tilde{q}_i}{\sum_{k=1}^{K} \tilde{q}_k} \qquad (i = 1, \dots, K) \tag{10}$$

で再正規化した q_i を用いる．

3.5 semantic template と意味合成

区間を引数に取る中間表現　自然言語に現れる時間表現（例：within 10 seconds）を一貫して扱うため，本研究では文（あるいは VP）レベルの意味を，時間区間 I を受け取り STL を返す関数 $f : \mathcal{I} \to \text{STL}$ として表す中間表現を導入する．ここで $\mathcal{I}$ は時間区間の集合であり，例えば $\mathcal{I} = \{[a, b] \mid 0 \le a \le b\}$ のような有界区間全体を想定する．例えば「reach R」は I 上での到達制約 $\mathbf{F}_I \phi_R$ を返す関数として表される．この型付けにより，within 節は「区間 $I = [0, T]$ を導入して $f(I)$ を評価する演算」として自然に合成できる．

構文木に沿った意味合成　各構文木 T_i に対して，ccg2lambda に基づき意味合成を行う．語彙項目（または構文範疇）に STL 指向の semantic template（λ 式）を割り当て，構文木に沿って関数適用（および簡約）を実行することで，文全体の意味表現 M_i を得る：

$$M_i = \text{Comp}(T_i; \Theta), \tag{11}$$

ここで Θ は semantic template の集合である．修飾句の係り先の違い（例：while 節の付加位置）は，どの部分式に semantic template が適用されるかの違いとして現れ，結果として STL のスコープの違いに対応する．

語彙項目（例）	semantic template（λ 式）
`reach`	$\lambda R.\lambda I.\ \mathbf{F}_I\, \phi_R$
`avoid`	$\lambda R.\lambda I.\ \mathbf{G}_I\, \neg\phi_R$
`or`	$\lambda f.\lambda g.\lambda I.\ f(I) \vee g(I)$
`and`	$\lambda f.\lambda g.\lambda I.\ f(I) \wedge g(I)$
`while`	$\lambda f.\lambda g.\lambda I.\ f(I) \wedge g(I)$
`within`	$\lambda T.\lambda f.\ f([0, T])$

表 1　STL 指向の semantic template（λ 式）の代表例．

semantic template の例　紙面の都合上，主要語彙の代表例のみを表 1 に示す．以下では型注釈を簡略化して記す（R は領域ラベル，T は時間長）．

例えば **Within 10 seconds, reach B or reach C while avoiding A.** に対して，while 節が **reach C** に付加される構文では

$$\text{or}\big(\text{reach}(B),\ \text{while}(\text{reach}(C), \text{avoid}(A))\big)$$

が構成され，while 節が選言全体に付加される構文では

$$\text{while}\big(\text{or}(\text{reach}(B), \text{reach}(C)), \text{avoid}(A)\big)$$

が構成される．両者に `within 10` が適用されることで，第 2 章で示した例（局所／大域スコープ）に対応する STL 候補へ変換される．

3.6 STL 候補の同定と確率集約

STL への変換と正準化　構文木ごとに得た意味表現 M_i を STL へ変換し，正準化することで $\hat{\varphi}_i$ を得る：

$$\hat{\varphi}_i = \text{Cano}(M_i). \tag{12}$$

ここで Cano(·) は，例えば以下を含む：

- 中間表現が $f : \mathcal{I} \to \text{STL}$ の形であれば，既定の区間 I_{def} を適用して閉じた STL を得る（within 節が存在する場合はその指定区間が優先される）．
- $\wedge/\vee$ の結合順の統一，冗長な括弧の除去，重複部分式の簡約．

これにより，異なる構文木から得られた式が同一 STL に対応する場合に，機械的に同定できるようにする．

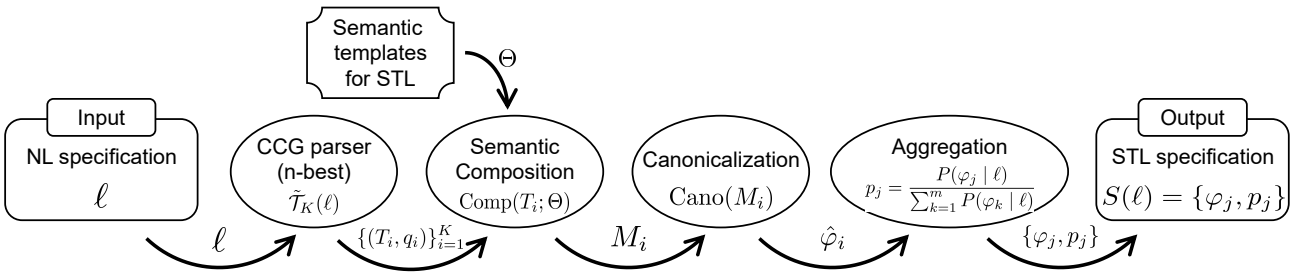

図 1　提案手法の概要.

同一 STL への集約　異なる構文木が同一の STL へ変換される場合があるため，導出確率を STL 側で集約する．候補 STL φ の確率を

$$P(\varphi \mid \ell) = \sum_{i:\hat{\varphi}_i=\varphi} q_i \tag{13}$$

で定義する．得られた m 個の異なる候補 $\{\varphi_j\}_{j=1}^{m}$（$m \le K$）に対し，（n-best 近似や失敗導出の除外等に備えて）再正規化した確率

$$p_j = \frac{P(\varphi_j \mid \ell)}{\sum_{k=1}^{m} P(\varphi_k \mid \ell)} \tag{14}$$

を用いて $S(\ell)$ を構成する．

4　実験

4.1　設定

本章では，図 1 で示した提案手法が，曖昧な自然言語仕様 ℓ から複数の STL 候補を生成し，式 (8) の形式で確率付き候補集合 $S(\ell)$ を得られることを，代表例により確認する．

CCG 構文解析には確率的 CCG パーサの一つである depccg を用い，入力文 ℓ に対して上位 $K = 50$ 個の構文木集合 $\tilde{\mathcal{T}}_K(\ell) = \{(T_i, \tilde{q}_i)\}_{i=1}^{K}$ を取得した．スコア $\tilde{q}_i$ は式 (10) により正規化し，意味合成（式 (11)），正準化（式 (12)），集約（式 (13)–(14)）を行った．

そして，自然言語仕様 ℓ を次のように与えた：

ℓ =*Within 10 seconds, reach B or reach C while avoiding A.*

前述のとおり，この文は *while avoiding A* の作用域により局所／大域スコープの 2 通りの妥当な解釈を許すものである．

4.2　生成された STL 候補および確率

本稿の記法に従い，到達は $\mathbf{F}_{[0,10]}\phi_R$，回避は $\mathbf{G}_{[0,10]}\neg\phi_A$ と表す．このとき，局所／大域スコープに対応する代表的な STL 候補は，第 2 章で導入した式 (6), (7) で与えられる．

STL 候補	確率
φ_{local}	0.5935
φ_{global}	0.4065

表 2　例文 ℓ に対する STL 候補と確率.

提案手法では，各構文木 T_i から得られる $\hat{\varphi}_i$ を正準化して同一性判定を行い，式 (13) により同一 STL に確率を集約する．本例では，$K = 50$ 個の導出は最終的に 2 種類の STL に集約され，候補数は $m = 2$ となった．ここで，正準化後の STL 候補が φ_{local}（局所）または φ_{global}（大域）のいずれに一致するかにより，導出インデックス集合を

$$I_{\text{local}} := \{i \mid \hat{\varphi}_i = \varphi_{\text{local}}\}, \qquad I_{\text{global}} := \{i \mid \hat{\varphi}_i = \varphi_{\text{global}}\}$$

と定義する．このとき，式 (13) に基づく確率は

$$P(\varphi_{\text{local}} \mid \ell) = \sum_{i \in I_{\text{local}}} q_i = 0.5935, \tag{15}$$

$$P(\varphi_{\text{global}} \mid \ell) = \sum_{i \in I_{\text{global}}} q_i = 0.4065, \tag{16}$$

となった．したがって，最終出力は式 (8) の形式で

$$S(\ell) = \{(\varphi_{\text{local}}, 0.5935), (\varphi_{\text{global}}, 0.4065)\} \tag{17}$$

と与えられる．結果は表 2 に示すとおりである．

4.3　まとめ

本例では，式 15–16 より，曖昧な自然言語仕様から 2 つの STL 仕様候補とその確率が得られた．このように，提案手法は STL 仕様を一意に定めるのではなく，妥当な解釈候補を複数保持し，複数の STL 仕様とその確率を提示できることを確認した．

5　おわりに

本稿では，曖昧な NL 入力に対して複数の解釈候補に対応する STL 仕様候補集合を生成し，さらに各候補に確率（重み）を付与して提示する枠組みを構築した．

謝辞

本研究は，JST CREST JPMJCR201, JST ACT-X JPMJAX23CK, and JSPS KAKENHI Grant 25K07794 and Grant 22KK0155 の支援を受けたものである．

参考文献

[1] Yongchao Chen, Rujul Gandhi, Yang Zhang, and Chuchu Fan. Nl2tl: Transforming natural languages to temporal logics using large language models, 2024.

[2] Pascual Martínez-Gómez, Koji Mineshima, Yusuke Miyao, and Daisuke Bekki. ccg2lambda: A compositional semantics system. In Sameer Pradhan and Marianna Apidianaki, editors, **Proceedings of ACL-2016 System Demonstrations**, pp. 85–90, Berlin, Germany, August 2016. Association for Computational Linguistics.

[3] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In **International symposium on formal techniques in real-time and fault-tolerant systems**, pp. 152–166. Springer, 2004.

[4] Alexandre Donzé and Oded Maler. Robust satisfaction of temporal logic over real-valued signals. In **International Conference on Formal Modeling and Analysis of Timed Systems**, pp. 92–106. Springer, 2010.

[5] Masashi Yoshikawa, Hiroshi Noji, and Yuji Matsumoto. A* ccg parsing with a supertag and dependency factored model. In **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 277–287. Association for Computational Linguistics, 2017.