CSC 390: Natural Language Understanding
Smith College Department of Computer Science
tnyamdorj@smith.edu, lgonzalezherrera@smith.edu

Tseegi Nyamdorj, Lesly Herrera
Northampton, MA, 01063
May 4th, 2025

Expanding the Conceptual Lexicon of McDypar

Abstract

This paper presents an extension of McDypar, a Lisp-based natural language parser grounded in Roger Schank's Conceptual Dependency (CD) Theory, which aims to represent sentence meaning using conceptual primitives rather than surface-level linguistic semantics. Originally limited to parsing a single sentence, McDypar lacked broader lexical coverage and structural flexibility. We expand its lexicon with new verbs, nouns, filler words, prepositions, and sentence templates inspired by Schank and Dyer's work, enabling it to handle a broader range of sentences, including recursive structures and ambiguity, through new demons such as expact and recursive. Additionally, we connect McDypar to MARGIE's BABEL system to generate English paraphrases of CD structures, demonstrating the system's interpretability. Our work contributes to the evolving field of symbolic semantic parsing by addressing ambiguity, embedded meaning, and system modularity, reaffirming the relevance of rule-based approaches in modern commonsense reasoning and AI.

Introduction

Inspired by a project idea proposed by Professor Jamie Macbeth, we set out to contribute to the existing body of research on Natural Language Understanding (NLU), with a focus on conceptual parsing using McDypar, a Lisp-based parser designed to map natural language input into CD representations. McDypar is built upon Roger Schank's CD Theory, which seeks to represent the meaning of sentences through conceptual primitives rather than traditional word-level semantics. Unlike conventional parsers that rely on syntactic or grammatical rules, McDypar employs a "conceptual lexicon" that directly maps individual words to CD conceptual structures (Dyer, 1982). These structures are then assembled into coherent sentence-level interpretations using a novel mechanism called "demons," autonomous agents that monitor the parsing process and actively work to integrate new information into the evolving conceptual representation.

CD structures offer a symbolic formalism that abstracts away from surface linguistic forms and instead focuses on modeling the underlying actions, actors, and objects involved in an event. At the start of our project, McDypar had limited functionality, capable of successfully parsing only a single sentence: "John picked up the ball and dropped it in the box." This example demonstrated the system's baseline ability to resolve references (e.g., correctly interpreting "it" as "the ball") and to infer continuity in agency (recognizing that "John" is the subject of both actions). However, McDypar lacked the flexibility to handle a wider variety of sentence structures and vocabulary outside of the provided sentence.

Background

McDypar (micro-version of DYPAR) was originally developed in Spring 1979 as an extension and improvement of McELI (Riesbeck and Schank, 1981), an earlier expectation-based parser that implemented linguistic expectations via a mechanism known as "requests" (Riesbeck, 1978). In contrast, McDypar introduced a more dynamic and process-driven approach by modeling expectations using autonomous agents known as demons. These demons actively monitor the parsing environment and "fire"

when their specific conditions are met or gap is filled, allowing them to integrate conceptual elements into a unified representation of sentence meaning.

McDypar offers several advantages over McELI and other similar systems, which are comprehensively outlined in Dyer's book, *In-Depth Understanding*. Some of the system's key improvements include: nameable and parameter-driven demons that are capable of being factored out during ambiguity resolution; a flexible working memory (WM) implemented as a double-linked queue, allowing bidirectional search and dynamic insertion/deletion; support for dynamically generated gaps that are used to bind CD structures upon demon activation; and an explicit/automatic disambiguation frameworks that uses demon behavior and active context, respectively, to resolve meaning during parsing.

Before describing our specific contributions, it is essential to outline the operational architecture of McDypar. The parser processes input from left to right, and as it encounters each word, it saves the associated "meaning" (typically a CD primitive or structure) into WM and spawns one or more demons to represent conceptual expectations. These demons monitor the input stream, waiting for contextual or structural cues that signal their firing conditions. When a demon fires, it executes a binding or structural operation, such as attaching an actor to an action or resolving a referential dependency.

Each CD structure includes a HEAD representing the main CD act (e.g., MTRANS, MBUILD, INGEST), along with associated SLOTs that define semantic roles such as actor, object, recipient, or location. GAPs or "*" are placeholders used when information is incomplete or expected but not yet observed, and are eventually filled by active demons when the corresponding cues are encountered[1]. This architecture formed the basis of our analysis and experimentation throughout the project, as we sought to expand McDypar's capacity to parse more diverse and semantically complex sentence structures.

Expanding Lexicon

To build upon McDypar's foundational capability of parsing a single sentence, our work focused on expanding its lexicon, refining its conceptual parsing scope, and developing a working conversion from McDypar to BABEL. This involved adding new verbs, noun structures, filler words, subjects, prepositions, and sentence templates that align with CD theory. We targeted common sentence forms involving action, transfer, and communication verbs, each of which required adjustments to the lexicon and grammar rules in McDypar's Lisp-based parsing engine. The sentences that we chose mostly derived from the example sentences used in Dyer and Schank's scholarly studies.

For instance, the sentence "Mary gave John a book" introduces a three-participant structure (ACTOR, RECIPIENT, OBJECT) that needed appropriate CD mapping, particularly for the GIVE action. Likewise, "Fred told Mary that John eats lobster" tested the system's handling of embedded clauses and the separation of SPEAK and INGEST actions, with semantic roles assigned across both parts of the sentence.

Some of the new sentences we implemented and tested include:

- Mary gave John a book.
- Fred told Mary that John eats lobster

[1] See Image [1] of Appendix for visual representation of the framework structure.

Tseegi Nyamdorj, Lesly Herrera
Northampton, MA, 01063

- Fred told Mary a secret.
- John went into a restaurant.
- John gave Mary an aspirin.
- Sarah pushed the chair.
- John pushed the table to the wall from the window.
- Mary said John ate an apple
- John went home
- John bought a book
- The gun shot Mary

These additions required semantic distinctions and enhancements in reference resolution and verb disambiguation. For example, “John pushed the table to the wall from the window” tested the spatial parsing capability, requiring representation of source and goal locations in the MOVE action.

Some parsing examples demonstrate successful representation, such as:

Sentence: George left the restaurant

Parse Results: (MOVE ACTOR (HUMAN NAME (GEORGE) GENDER (MALE)) OBJECT (PHYSICAL-OBJECT CLASS (BUILDING) NAME (RESTAURANT)))

In contrast, this other example reveals a current limitation in the system, incorrectly linking the embedded clause and resolving “apple” as the INGEST object for the second verb.

Sentence: Mary said John ate an apple

Parse Results: (MTRANS ACTOR (HUMAN NAME (MARY) GENDER (FEMALE)) OBJECT NIL) (INGEST ACTOR (HUMAN NAME (JOHN) GENDER (MALE)) OBJECT NIL) (PHYSICAL-OBJECT CLASS (FOOD FRUIT NIL) NAME (APPLE))

Conversion of McDypar to BABEL

Another important component of our project was exploring the conversion of McDypar’s CD representations into MARGIE’s BABEL system. The MARGIE project (1969–1974), developed by Riesbeck, Goldman, Reiger, and Schank at the Stanford Artificial Intelligence Laboratory (SAIL), was an early implementation of CD theory. MARGIE demonstrated the potential of CD structures by accepting natural language sentences as input and generating paraphrases and inferences as output. One of MARGIE’s key modules was BABEL, a paraphrasing system designed to generate surface realizations of CD structures in the form of English paraphrases.

As an extension of our work with McDypar, we decided to investigate the compatibility between McDypar’s conceptual outputs and BABEL’s input expectations. Our goal was to test whether some of the newly added McDypar lexicons could be translated and passed through BABEL to produce meaningful English paraphrases, which further validates the expressive power and usability of the system. Early in the project, we debated whether to work solely within the MARGIE framework, but we ultimately chose McDypar due to its more modular design, improved runtime structure, and extensibility. While MARGIE is historically important, its development in MLISP and its less dynamic architecture posed limitations for

further growth. In contrast, McDypar offers a clearer pathway for expansion through its use of demons and lexicon-based extensibility.

The core BABEL conversion pipeline and framework were developed in prior years by Professor Macbeth and his lab. Our contribution focused on file organization, integration, and testing. The pipeline operates by loading the necessary files, parsing the sentence via McDypar, and using a nested series of function calls, including get-cd, the conversion function, and the express function, to produce paraphrases in natural English[2]. Using this system, we successfully generated paraphrases for two of our newly added lexicons: “Mary gave John a book” and “John went into a restaurant.” The first sentence produced four paraphrases, varying in verb usage and voice (e.g., active vs. passive). The second sentence returned two paraphrases, differing primarily in verb choice ("went" vs. "came") but both referring to a "building" rather than a "restaurant." This result appears to stem from how "restaurant" is encoded in CD form as a subclass or instance of a generic building object[3]. Other test sentences resulted in an error message, specifically, “CAR: 0 is not a list.” We suspect this occurred either because the relevant CD structures were not defined in BABEL’s lexicon or were formatted incompatibly. Due to time constraints, we were not able to fully debug these issues or explore the missing entries in depth, but we hope future iterations of the project can continue this line of investigation.

Expanding Demons

While adding new lexicons, we began to recognize structural limitations in the original McDypar code, which led us to focus on enhancing its ability to represent recursive CD units within a sentence. Originally, McDypar could only handle simple and compound sentences with independent clauses connected by conjunctions, such as and. However, in cases where a sentence includes a dependent clause, for example, “John told Mary that Fred eats lobster,” McDypar would generate two separate CD structures: one for “MTRANS” and one for “INGEST.”

To address this issue, we developed a new demon called expact, modeled after the existing exp demon. The exp demon searches for a concept with one of the given classes in the specified direction until it hits a boundary, such as a clause break or the end of a sentence. Once found, it binds that concept to a gap. Similarly, expact is designed to search specifically for full CD acts (e.g., MTRANS, MBUILD, INGEST). It uses the function is-act, which checks whether a given structure is a CD act, and expact ensures that the identified act is not the demon’s own MYCONCEPT, thus avoiding self-matching. With this demon, McDypar can now correctly process sentences like “John told Mary that Fred eats lobster” by nesting the INGEST act inside the MTRANS act, accurately modeling that the information being transferred is the act of Fred ingesting lobster.

We further tested the distinction between recursive and non-recursive sentence structures using the example “Fred told Mary a secret,” which, while still an instance of MTRANS, is a simple sentence rather than a complex one. To allow McDypar to distinguish between these cases, we developed two disambiguation demons: recursive and non-recursive. The recursive demon uses limited look-ahead ability to check if the word “that” appears in the immediate next or next-next word, signaling a complex sentence with embedded clauses. In contrast, the non-recursive demon searches for a person, mental

[2] See Image [2] of Appendix for full pipeline.
[3] See Image [3] for conversion results.

object, or physical object following the MYCONCEPT, indicating a simple sentence structure. Once one of these demons is triggered and the test is satisfied, it fires the +ACT, assigning the correct meaning of the word "told" based on the sentence's structure.

Challenges of Representing Purpose/Goal

Inspired by the success of the previous section, we sought to further our project by enhancing McDypar's ability to parse sentences that include an adverbial clause of purpose or use an infinitive of purpose to express the goal or reason behind an action. Our focus centered on the word "to," which can function both as a marker of direction and as a marker of purpose. To distinguish between these uses, we developed two disambiguation demons: to-purpose and to-direction. The to-purpose demon searches for an ACT following the MYCONCEPT, indicating that the phrase expresses a purpose or reason. In this case, the word "to" is mapped to an MBUILD CD structure to signify the intention or planning of a future act. Conversely, the to-direction demon searches for a setting, person, or physical object after the MYCONCEPT, identifying when "to" functions as a simple preposition indicating direction or destination. For example, consider the sentence: "John bought a book to give it to Mary." The first instance of "to" introduces an infinitive of purpose (buying the book in order to give it), while the second "to" functions as a prepositional phrase (giving the book to Mary). Our aim was to enable McDypar to distinguish and represent both of these meanings appropriately.

Unfortunately, due to time constraints, we were unable to fully implement this disambiguation. In our latest version, both demons terminate prematurely after spawning because MYCONCEPT is currently mapped by default to the preposition "to," thereby preventing the demons from assigning alternative interpretations. We did not have sufficient time to resolve this issue. Additionally, we encountered conceptual challenges when trying to represent goal-directed behavior using MBUILD in CD structures. Specifically, expressing intentionality, planning, or reasoning through MBUILD can result in paraphrases that feel unnatural or awkward. For example, paraphrasing the earlier sentence as "John's thinking of giving the book caused him to buy the book," while technically correct in a CD framework, lacks the natural fluency of the original sentence. This example, based on Macbeth (2020), highlights the limitations of using causal links like MBUILD to represent intentionality. It suggests that while the conceptual structure captures the notion of premeditated action, the causal linking may be the wrong type or may lack additional contextual framing necessary for natural paraphrasing. We think that this area represents a rich opportunity for future work. Addressing these issues may involve introducing new CD primitives or demons better suited to representing intentional actions and the nuanced semantics of infinitives of purpose.

Related Work

Our project builds on a rich history of research in natural language understanding (NLU), particularly within the tradition of symbolic approaches inspired by Roger Schank's Conceptual Dependency Theory (Schank, 1975). CD theory aims to represent the meaning of sentences in a language-independent manner, using a finite set of conceptual primitives (such as ATRANS for transfer or PTRANS for motion). This symbolic representation allows for deep reasoning over sentence semantics and has influenced many early AI systems.

Among the first major implementations of this theory were the Script Applier Mechanism (SAM) and Fast Reading Understanding and Memory Program (FRUMP), both of which focused on narrative understanding. SAM, for example, could interpret news stories and generate paraphrases based on predefined scripts (Cullingford, 1978), while FRUMP was designed to scan wire service stories and extract gist-level meaning (DeJong, 1982). These systems showcased the potential of conceptual parsing for structured story comprehension but were heavily reliant on pre-scripted domains.

In contrast, more recent NLU systems such as TRIPS (Allen et al., 2007) and Sparser (McDonald & Pustejovsky, 1985) have pursued more generalized parsing and reasoning capabilities, often incorporating machine learning and statistical NLP techniques. These systems can integrate systactic, semantic, and pragmatic analysis, and they operate on larger, more open-domain corpora. However, their complexity can obscure the explicit, interpretable semantic structures that symbolic systems like McDypar aim to preserve.

McDypar's rule-based design offers a fine-grained level of control, which remains advantageous for research in explainable AI and systems requiring deterministic outputs. Unlike probabilistic systems, symbolic parsers allow for more detailed inspection and debugging of semantic decisions, making them particularly suitable for tasks like commonsense reasoning and knowledge base construction.

Conclusion

In conclusion, our project aimed to expand the functionality and interpretive capacity of McDypar, a Lisp-based parser grounded in CD theory, by increasing its lexicon, improving its handling of complex syntactic structures, and enabling cross-system interoperability with MARGIE's BABEL paraphrasing system. Through careful analysis and integration of new verbs, nouns, prepositions, and sentence templates, many drawn from canonical examples in Dyer and Schank's foundational work, we extended McDypar's ability to process a broader range of sentence structures. We also addressed core limitations of the original system, particularly its difficulty parsing recursive and intentional structures, by introducing new demons such as expact, non-recursive, recursive, to-destination, and to-purpose. These demons enable McDypar to better capture the semantics of embedded clauses and goal-directed behavior. Our exploration of MARGIE and the conversion pipeline provided further validation of McDypar's utility and laid the groundwork for future integration of symbolic reasoning and surface-level paraphrasing. While we encountered challenges with ambiguity, edge cases, and representation of intention, our contributions underscore the continued value of symbolic approaches in commonsense reasoning and language understanding. This work not only revives interest in early AI methods but also suggests pathways for hybrid systems that combine rule-based semantics with modern statistical models in future research.

Acknowledgements

We would like to thank Professor Jamie Macbeth for his invaluable guidance throughout this project, including his insightful suggestions, prior code developed in collaboration with former students, and for providing key scholarly materials related to McDypar and Conceptual Dependency structures.

Reference

- Allen, J. F., Ferguson, G., & Stent, A. (2007). *An architecture for more realistic conversational systems*. In *Intelligent User Interfaces*.
- Cullingford, R. E. (1978). *Script Application: Computer Understanding of Newspaper Stories*. Ph.D. Dissertation, Yale University.Schank, R. C. (2014). *Conceptual information processing* (Vol. 3). Elsevier.
- DeJong, G. F. (1982). *An Overview of the FRUMP System*. In *Strategies for Natural Language Processing* (pp. 149–176). Lawrence Erlbaum.
- Dyer, M. G. (1982). *In-depth understanding: A computer model of integrated processing for narrative comprehension*. Yale University.
- Macbeth, Jamie C. "Enhancing learning with primitive-decomposed cognitive representations." In *Proceedings of The First Annual International Workshop on Self-Supervised Learning (IWSSL-2020), Cambridge, MA*. 2020.
- McDonald, D., & Pustejovsky, J. (1985). *Sparser: A better chart parser*. In *Proceedings of the 23rd Annual Meeting on ACL*.
- Schank, R. C. (1975). *Conceptual Information Processing*. North-Holland.
- Schank, R. C., & Riesbeck, C. K. (2013). *Inside computer understanding: Five programs plus miniatures*. Psychology Press.

Appendix

Image [1] - page 387 of Dyer, 1982

```
concept = (HEAD SLOT GAP   <== demons
                SLOT GAP   <== demons
                   . . .
                SLOT GAP)
```

Image [2] - pipeline of McDypar to BABEL conversion

```
(load 'prph)
(load 'surf)
(load 'mcdypar)
(load 'mcdypar_babel_converter)
```

```
(parse '(mary dropped the ball))
(express (mcdypar-convert-cd-to-babel-style (get-cd)))
```

Tseegi Nyamdorj, Lesly Herrera
Northampton, MA, 01063

Image [3] - results of McDypar to BABEL conversion for “Mary gave John a book,” and “John went into a restaurant.”

```
(MARY RETURNED BOOK TO JOHN)
(MARY GAVE BOOK BACK TO JOHN)
(JOHN GOT BOOK FROM MARY)
(JOHN RECEIVED BOOK FROM MARY)
```

```
(JOHN WENT TO BUILDING)
(JOHN CAME TO BUILDING)
```