Intertwining CP and NLP:
The Generation of Unreasonably Constrained Sentences

Alexandre Bonlarron[1,2] and **Jean-Charles Régin**[2]

[1]Université Côte d'Azur, Inria, France
[2]Université Côte d'Azur, CNRS, I3S, France
alexandre.bonlarron@gmail.com, jcregin@gmail.com

Abstract

Constrained text generation remains a challenging task, particularly when dealing with hard constraints. Traditional Natural Language Processing (NLP) approaches prioritize generating meaningful and coherent output. Also, the current state-of-the-art methods often lack the expressiveness and constraint satisfaction capabilities to handle such tasks effectively. This paper presents the *Constraints First* Framework to remedy this issue. This framework considers a constrained text generation problem as a discrete combinatorial optimization problem. It is solved by a constraint programming method that combines linguistic properties (e.g., n-grams or language level) with other more classical constraints (e.g., the number of characters, syllables, or words). Eventually, a curation phase allows for selecting the best-generated sentences according to perplexity using a large language model.

The effectiveness of this approach is demonstrated by tackling a new more tediously constrained text generation problem: the iconic RADNER sentences problem. This problem aims to generate sentences respecting a set of quite strict rules defined by their use in vision and clinical research. Thanks to our CP-based approach, many new strongly constrained sentences have been successfully generated in an automatic manner. This highlights the potential of our approach to handle unreasonably constrained text generation scenarios.

1 Introduction

Constrained text generation is a great challenge in natural language processing (NLP), demanding the creation of textual output that adheres to specific requirements. This task encompasses a broad spectrum of text generation scenarios, ranging from crafting lyrics adhering to rhythmic, lexical, and metrical constraints to producing text summaries prioritizing length, semantic entailment, or other desired properties. This article focuses on the first category of constrained output, where strict adherence to specific requirements is paramount.

In NLP, several methods have been developed to tackle contained text generation. The major approach relies on a Large Language Model (LLM) guided search [Lu *et al.*, 2022; Liu *et al.*, 2021; Post and Vilar, 2018; Hokamp and Liu, 2017]. For instance, BeamSearch excels when there are lexical constraints (i.e., enforce or avoid a set of keywords).

In addition, the latest series of conversational agents like ChatGPT [OpenAI, 2023], PaLM [Google, 2023], Alpaca, Vicuna [Touvron *et al.*, 2023] show that LLM can achieve decent constraint satisfaction rates (99%) on lexical constrained generation task (see sent-04 task in [Yao *et al.*, 2024] for more information). Nevertheless, at the same time, even fine-tuned prompted LLMs fail to generate a sentence that contains for example exactly 82 characters (see sent-01 task).

Intuitively, enforcing that a sentence must contain exactly 82 characters can be reduced to a knapsack problem where each word has a weight equal to its number of characters and where the total number of characters of the sentence is the capacity of the backpack (i.e., 82). It is known to be an NP-complete problem.

In the NLP taxonomy, this requirement type is called hard constraint [Garbacea and Mei, 2022], characterized by binary functions associated with the constraints that must be true. Interestingly, the concept of hard constraints in NLP aligns with the classical definition of a constraint in discrete combinatorial optimization (CO). Various hard constraints in NLP find their counterparts in CO, such as the length constraint in NLP corresponding to a knapsack constraint in CO. In addition, the knapsack constraint has been well-studied in CO as it is ubiquitous in real-world problems. Moreover, several efficient algorithms exist to solve this constraint using dynamic programming [Trick, 2003]. Thus, using a dynamic programming approach to compute a set of sentences of 82 characters sounds promising.

Furthermore, over the past decade, research in assisted song composition, spearheaded by Pachet's ERC Flow-Machines[1], has highlighted the connection between output specifications in songs or texts and constraints in the context of CO. Various output requirements were successfully modeled and solved in CO, including style modeling [Pachet and Roy, 2011], virtuoso melodies [Pachet *et al.*, 2011], lyrics [Barbieri *et al.*, 2012], meter [Roy and Pachet, 2013], plagiarism [Papadopoulos *et al.*, 2014], and palindrome [Papadopoulos *et al.*, 2015]. Notably, this research has paved the

---

[1]https://www.flow-machines.com

way for formulating constrained generation tasks in text and music as constraint satisfaction problems (CSPs).

In the context of text generation under constraints, viewing this problem as a CSP implies defining: (1) The set of decision variables that represent words of a sentence. (2) The domain of each variable is the set of allowed vocabulary. (3) Finally, the constraints are the set of rules that the text (the assigned variables) must satisfy.

A well-known technique to solve a CO problem when formalized as a CSP is Constraint Programming (CP). CP is a paradigm that computes solutions that satisfy constraints. CP is mainly based on filtering algorithms (also known as propagators), which remove values from variable domains that do not belong to a solution of a constraint. The association of such algorithms with global constraints is one of the main strengths of CP because they exploit the specific structure of each constraint. Thus, CP is an excellent candidate for tackling constrained generation tasks.

The paper introduces *Constraints First* a CP-based framework to generate sentences under constraints. This framework combines the semantic aspects of sentences in a given language with the constraints imposed on the words that make them up. On the one hand, it uses n-grams and possibly their associated likelihoods. On the other hand, it uses variables whose domains are words and constraints between variables, to allow, for example, a limit on the number of characters used. The framework proposes to transform the initial problem into a combinatorial optimization problem based on the multi-valued decision diagram (MDD) data structure, which is particularly well-suited to representing a corpus. This problem is then solved using constraint programming.

Additionally, it introduces the problem of RADNER sentence generation, which is a clinical test for people with potential vision problems. This problem is used to demonstrate the interest of our framework in solving a particularly difficult problem for which no generic solution is known to date.

The structure of this paper is outlined as follows:

- Section 2: Background information on Constraint Programming (CP) and NLP is introduced.

- Section 3: A constraint programming model is presented for generating constrained sentences.

- Section 4: The RADNER sentence problem is formalized as a CSP.

- Section 5: The effectiveness of the proposed approach is showcased by detailing the generated output.

- Section 6: Open challenges, limitations, and future perspectives associated with the approach are discussed.

2 Preliminaries

2.1 Multi-valued Decision Diagram

Multi-valued decision diagrams (MDDs) are directed acyclic graphs (DAG) data structures used in discrete optimization. They consist of nodes and arcs, with two particular nodes: the root node and the terminal node. Here, only Deterministic Reduced Ordered MDDs [Amilhastre *et al.*, 2014] are considered.

MDDs are mainly used as archives for *Table Constraint* [Cheng and Yap, 2010; Lecoutre, 2011] and are generic to compute and store various other constraints [Malalel *et al.*, 2023; **?**; Wang and Yap, 2022; Gentzel *et al.*, 2022; Verhaeghe *et al.*, 2019; Verhaeghe *et al.*, 2018]. MDDs as a tool for solving optimization problems can be found in [Jung and Régin, 2022; Gillard and Schaus, 2022; Rudich *et al.*, 2022; van Hoeve, 2022; Bergman *et al.*, 2016].

MDDs are organized into layers, where each layer contains nodes and outgoing arcs. Each layer represents a variable, and therefore, any MDD model represents an n-ary function $f(x_1, x_2, ..., x_n) \mapsto \{true, false\}$. Equivalent nodes (i.e., nodes with the same outgoing arcs with the same labels) are merged, compressing the data structure. Various operations (intersection, union, difference...) can be applied between MDDs without the need to decompress them.

MDD of a Constraint
Given a constraint C defined over a set of variables $X = \{x_1, x_2, ..., x_n\}$, an MDD can be compiled to store the valid assignments (tuples) associated with the constraint C. Therefore, MDD(C) is the MDD associated with the constraint C.

Cost-MDD
A cost-MDD is an MDD whose arcs have additional information: the cost c of the arc. Let M be a cost-MDD and p be a path of M. The cost of p is denoted by $\gamma(p)$ and is equal to the sum of the costs of the arcs it contains.

2.2 Constraint Satisfaction Problem

A Constraint Satisfaction Problem (CSP) is a triplet: $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$, where:

- $\mathcal{X} = \{X_1, X_2, ..., X_n\}$ is the set of variables of the problem.

- $\mathcal{D} = \{D_{X_1}, D_{X_2}, ..., D_{X_n}\}$ is the set of domains, where each domain D_{X_i} corresponds to the set of possible values for the variable X_i.

- $\mathcal{C} = \{C_1, C_2, ..., C_m\}$ is the set of constraints of the problem. A constraint represents a property of the solutions to the problem.

A solution is an assignment of all the variables to a value present in their respective domains, such that all the constraints are satisfied.

2.3 Language Model

N-gram Model
An n-gram [Shannon, 1951; Jurafsky and Martin, 2009] is a contiguous sequence of n words extracted from a text corpus. Consider the sentence, "The princess wears a red dress". This sentence contains several 3-grams, such as "The princess wears," "princess wears a," "wears a red," and "a red dress" We can define chaining rules between n-grams based on shared word n-tuples. Two n-grams are considered chained if they share a common sequence of $n-1$ consecutive words. More precisely, the last $n-1$ words of the first n-gram must be identical to the first $n-1$ words of the second n-gram.

Figure 1: Example of MDD representing the set of $x_1 + x_2 + x_3 \in [5, 9]$. For each variable x_i, the domain $D(.)$ is $D(x_1) = \{1, 3, 7\}$, $D(x_2) = \{0, 2, 4\}$, $D(x_3) = \{2, 3, 4\}$. For example, $(7, 0, 2)$ belongs to the set of solutions defined by the MDD.

LLM

Recent advancements in NLP have witnessed the emergence of powerful language models (LLMs) (e.g., PaLM [Google, 2023], GPT-4 [OpenAI, 2023], LLaMa [Touvron *et al.*, 2023]) leveraging the Transformer architecture [Vaswani *et al.*, 2017]. These models are trained on massive datasets, enabling them to generate human-quality text through various decoding strategies, such as greedy decoding or more complex strategies. As language models, their core function relies on estimating the probability of a given word sequence.

Perplexity

Text perplexity (PPL), a measurement of uncertainty rooted in Shannon's information theory, can be readily computed in large language models (LLMs), given their inherent ability to calculate text probabilities. It represents the geometric mean of the inverse conditional likelihoods associated with a given sequence of words [Jurafsky and Martin, 2009]. Formally, let S_n denote a sequence of n words, written as $S_n = w_1 w_2 \cdots w_n$. The PPL of S_n is calculated as:

$$PPL(S_n) = \sqrt[n]{\frac{1}{P(w_1 w_2 \cdots w_n)}}$$

where $P(\cdot)$ denotes the LLM's assigned probability. Intuitively, PPL reflects the model's perceived likelihood of generating a particular text, with lower values signifying higher confidence [Garbacea and Mei, 2022]. This metric is valuable for LLM evaluation, ensuring that the model correctly identifies well-formed samples through low PPL scores.

3 The Constraints First Framework

This section presents the major steps of the *Constraints First* framework. An overview is depicted in Fig. 3.

3.1 Input Data: N-gram Corpora

First, a corpus (i.e., a set of sentences) is needed. This corpus can be made from various sources (e.g., books, newspapers, subtitles, Wikipedia,...). From the sentences set, all n-grams are extracted, and some are filtered out if necessary. For instance, if some words are forbidden, all n-grams where the words appear are deleted.

Figure 2: Example of MDDtrie storing 3-grams (successions of 3 words): "The black dog"; "A red pot"... Any path from the *root* to *tt* is a valid n-gram. To find the successors of the n-gram "The white dog" (in red), more precisely the following potential words, we start from *root* to walk along the arcs that contains the labels of the two last arc, i.e., "white" and "dog" (in blue). In that case, one outgoing arc from the node can be reached with "white cat". Thus, the successor of "The white dog" is "likes" (in green).

Using n-grams is essential for integrating language into generation. A sentence in a language is very different from a random sequence of words. Although it is difficult to manage the meaning of a sentence (i.e., generate only sentences that make sense), this does not mean that the meaning of a sentence should be completely abandoned during generation.

3.2 Ngram reTRIEval

Next, the set of n-grams is stored within an MDD (see Fig. 2). This MDD is used as a Trie; thus, it is named MDDTrie. It manages the operations to store and retrieve n-grams.

3.3 Constraint Programming Model

The Constraint Programming Model is based on an MDD that was built from MDDTrie. It contains all the feasible sentences satisfying the constraints.

MDD Compilation

MDD can be used to encode and solve CSP [Hoda *et al.*, 2010; Perez and Régin, 2015]. Figure 1 gives an example of such encoding. In particular, an MDD-based model has been shown to be nearly adequate for this task [Bonlarron *et al.*, 2023]. The approach developed claimed to be general, but its description is too close the resolution of the MNREAD sentences problem.

The CSP model is compiled into MDD (i.e., each constraint in the CP model (see Tab. 1) is associated with a particular MDD). The number of layers of the MDD is the number of variables. The potential label that an arc can have is a function of the domains of the associated variables of the layer. The number of characters or syllables of a word is a cost. Two cost functions, respectively, $\#Char(.)$ and $\#Syll(.)$, are available for the label of an arc. Thus, all MDDs in this approach are cost-MDDs.

A Constraint Programming Model of Standardized Sentences	
$\mathcal{V}$	$X_1, X_2, ..., X_N$
$\mathcal{D}$	$\forall i, D(X_i) = \{words \in AllowedLexicon\}$
Language modelling with k-grams	
$C_{LM^{start}}$	$IsNgramOfStart(X_1, X_2, X_3, ..., X_k) = \top$
$C_{LM^{succ}}$	$i > k, X_i = NgramSuccession(X_{i-k}, ..., X_{i-3}, X_{i-2}, X_{i-1})$
$C_{LM^{end}}$	$IsNgramOfEnd(X_{14-k}, X_{14-k-1}, ..., X_{14}) = \top$
C_{Length}	Format (length) Constraints
C_{L^1}	$\lfloor LineSize \rfloor \le \sum_{i=1}^{L_1} \#Char(X_i) \le \lceil LineSize \rceil$
C_{L^2}	$\lfloor LineSize \rfloor \le \sum_{i=1+L_1}^{L_2} \#Char(X_i) \le \lceil LineSize \rceil$
C_{L^3}	$\lfloor LineSize \rfloor \le \sum_{i=1+L_2}^{L_3} \#Char(X_i) \le \lceil LineSize \rceil$
C_{L^4}	$\lfloor SentenceSize \rfloor \le \sum_{i=1}^{N} \#Char(X_i) \le \lceil SentenceSize \rceil$
C_{Meter}	Meter (syllable) Constraints
C_{M^1}	$\sum_{i=1}^{N} \#Syll(X_i) = SyllableCount$

Table 1: Minimalistic CSP model to generate generic standardized sentences. The decision Variables are words, and the Domains are the authorized words. Then, the n-gram constraints ensure that any assignment of the variables is consistent with the n-gram chaining rules. Finally, several knapsack constraints are defined to control the total size of the sentence, its display property (3 lines, for instance), or its meter property. $\#Syll(X_i)$ is the number of syllables of the assigned variable X_i and $\#Char(X_i)$ is the number of characters of X_i.

The MDD MDD_{final} containing all the solutions that satisfy the conjunction of the various constraints can be obtained in two ways. Either there is an interest in keeping intermediate results, and MDD_{final} is computed by intersecting the MDDs associated with each explicitly defined constraint. Alternatively, the intersection of the MDDs associated with each constraint is performed on the fly. This means that when performing an intersection between MDD_1 and MDD_2 s.t., $MDD_1 \bigcap MDD_2$, MDD_2 is given in intention, so its solutions are never computed. This second approach saves a lot of memory and is faster. Formally, we have

$$MDD_{final} = \bigcap_{i=1}^{m} MDD(C_i)$$

therefore,

$$MDD_{final} = MDD(\wedge_{i=1}^{m} C_i)$$

So far, we have assumed that it is possible to build MDD_{final} explicitly by successive intersections, whatever constraints are added. This turns out to be the case for our application domain. This approach is quite realistic for the types of constraints we have been led to integrate. However, should it become impossible to calculate MDD intersections for new constraints, this does not render the proposed framework useless. Indeed, there is no need to build the final MDD using intersections alone. We could very well build an MDD integrating a subset of the constraints, then use a CP solver integrating this MDD (using the MDD4R propagator) with the constraints not yet integrated to calculate the solution set of the problem.

3.4 Sentence Curation Helped by an LLM

Once the MDD_{final} is computed, all solutions (sentences) can be extracted. The solution set is then evaluated by LLM.
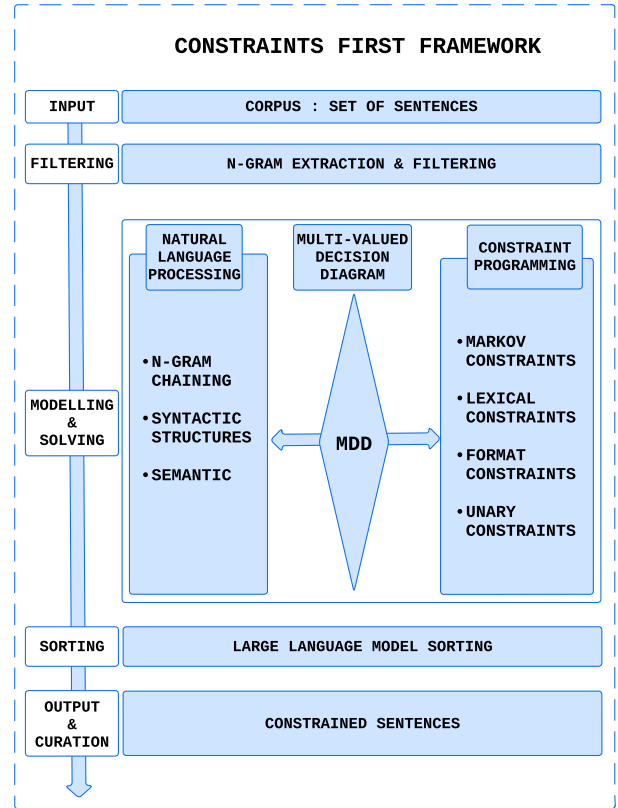
Figure 3: This figures summarizes the major step of the *Constraints First* framework. Once the filtered n-grams are gathered. The MDD data structure acts as a bridge between the constraint satisfaction relying on CP techniques and the n-gram chaining that takes account of the structure of the language.

Thus, the solutions are sorted by the perplexity score computed by an LLM. This simplifies the sentence selection process dramatically.

4 Case Study: Standardized Text

4.1 Why Standardized Text Is Critical?

Reading performance assessment has become a prominent method for evaluating vision [Radner, 2017]. Standardized text, particularly sentences, has established itself as a cornerstone for reading performance assessments in vision research [Mansfield *et al.*, 2019]. Due to the intricate nature of reading involving various individual-specific parameters, standardized materials effectively minimize person-related biases in measurements [Legge, 2016]. Several reading charts utilizing standardized sentences exist, such as the MNREAD test [Mansfield *et al.*, 1993] or the RADNER test [Radner and Diendorfer, 2014]. While sharing a common purpose, each chart possesses its strengths and limitations. Vision research necessitates standardized sentences and tools that empower researchers to readily produce standardized material to assess reading performances and test hypotheses related to visual impairment.

Our father always hired this
bad bricklayer, who did not
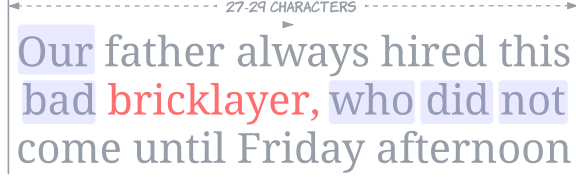come until Friday afternoon

Figure 4: Example of an English RADNER sentence incorporating relative clauses with three lines and 14 words. A subset of the RADNER rules are highlighted. Each line contains between 27 and 29 characters; also, in purple, several words should be of exactly one syllable. In red, the second word of the second line must contain 10 characters and three syllables.

This motivation originating from the vision research field converges with a compelling challenge in artificial intelligence regarding the state-of-the-art in constrained text generation. To the best of our knowledge, all existing RADNER sentences were handcrafted. Therefore, this paper introduces the first approach, which is able to generate them automatically. Solving this real-world problem leads to thinking about the potential intertwining of CP and NLP.

The work of Bonlarron et al. [Bonlarron *et al.*, 2023] consists of applying this framework to solve the MNREAD sentence problem. The following section introduces a new application to demonstrate the genericity of the framework.

4.2 The RADNER Sentences Problem

The basic rules that a RADNER sentence must obey are the following :

RADNER Core Rules

- (1) The sentences incorporated relative clauses and had three lines and 14 words.

- (2) Each line had 27–29 characters, including spaces (82–84 characters per sentence).

- (3) The first and second lines had 5 words.

- (4) The third line had 4 words.

Language Dependent Rules, Here German

- (5) The first word of the first and of the second line was a word of 1 syllable and 3 letters.

- (6) The second word of the second line was a word of 3 syllables and 10 letters.

- (7) This word was followed by a relative clause that (8) began with 3 short words, each of 1 syllable.

- (9) In the first line there was one noun of 2 syllables and another word (not a noun) of 2 syllables.

- (10) The third line began with a word of 2 syllables, followed by (11) a noun of 2 syllables.

- (12) The next word was the verb of the sentence, composed of 3 syllables.

A Constraint Programming Model of Radner-like Sentences	
$\mathcal{V}$	$X_1, X_2, ..., X_{15}$
$\mathcal{D}$	$\forall i, D(X_i) = \{words \in Vocabulary\}$
Language Modelling Constraints	
$C_{LM^{start}}$	$IsNgramOfStart(X_1, X_2, X_3, ..., X_k) = \top$
$C_{LM^{succ}}$	$i > k, X_i = NgramSuccession(X_{i-k}, ..., X_{i-3}, X_{i-2}, X_{i-1})$
$C_{LM^{end}}$	$IsNgramOfEnd(X_{14-k}, X_{14-k-1}, ..., X_{15}) = \top$
C_{Length}	**Format (length) Constraints**
C_{L^1}	$27 \le \sum_{i=1}^{5} \#Char(X_i) \le 29$
C_{L^2}	$27 \le \sum_{i=6}^{11} \#Char(X_i) \le 29$
C_{L^3}	$27 \le \sum_{i=11}^{15} \#Char(X_i) \le 29$
C_{L^4}	$82 \le \sum_{i=1}^{15} \#Char(X_i) \le 84$
C_{Meter}	**Syllable Constraints**
C_{M^1}	$\sum_{i=1}^{15} \#Syll(X_i) = 23$
C_{Unary}	**Unary Syllable and Character Constraints**
	First Line Constraints
C_{U^1}	$\#char(X_1) \le 4 \bigwedge \#Syll(X_1) = 1$
C_{U^2}	$\#char(X_2) \le 6 \bigwedge \#Syll(X_2) = 2$
C_{U^3}	$\#char(X_3) \le 7 \bigwedge \#Syll(X_3) = 2$
	Second Line Constraints
C_{U^6}	$\#char(X_6) \le 4 \bigwedge \#Syll(X_6) = 1$
C_{U^7}	$\#char(X_7) = 10 \bigwedge \#Syll(X_7) = 4$
C_{U^8}	$X_8 = ","$
C_{U^9}	$\#Syll(X_9) = 1$
$C_{U^{10}}$	$\#Syll(X_{10}) = 1$
$C_{U^{11}}$	$\#Syll(X_{11}) = 1$
	Third Line Constraints
$C_{U^{12}}$	$\#Syll(X_{12}) = 2$
$C_{U^{13}}$	$\#Syll(X_{13}) = 2$
$C_{U^{14}}$	$2 \le \#Syll(X_{14}) \le 3$
$C_{U^{15}}$	$2 \le \#Syll(X_{15}) \le 3$

Table 2: CSP model to tackle the RADNER-like sentences problem. It is a particular case of standardized sentences. Therefore, the various knapsack constraints are fixed. For instance, the total number of characters range is [82,84]. Additionally, several unary constraints are added to further model the RADNER problem. For instance, C_{U^7}: $\#char(X_7) = 10 \bigwedge \#Syll(X_7) = 4$, ensure that variables X_7 is assigned with a word of 10 letters and 4 syllables. Remark: The comma is viewed as a word, so the unary constraints associated with X_8 (i.e., C_{U^8}) enforce its assignment with a comma.

- (13) The last word was a part of the verb and had 2 syllables.

Then, the RADNER test, designed initially in German, was adapted for other languages; while sharing the same core rules, several side constraints differ from one language to another. For instance, the first word of the first line must contain three characters and one syllable in the original version. Whereas in the Spanish version of the sentences (Radner-Vissum version [Alió *et al.*, 2008]), the first word satisfies the same rule or contains precisely two characters. Surprisingly, In the Portuguese version (Radner-Coimbra version [Rosa *et al.*, 2016]), the first word can also contain only one character.

The CP model described in Tab. 2 defines the RADNER core problem independent from a particular target language. For a particular language, some constraints need to be hardened: For instance

$$C_{U^1} : \#char(X_1) \le 4 \bigwedge \#Syll(X_1) = 1,$$

becomes in German,

$$C_{U^1} : \#char(X_1) = 3 \bigwedge \#Syll(X_1) = 1.$$

In the Spanish case it becomes,

$$C_{U^1} : 2 \leq \#char(X_1) \leq 3 \bigwedge \#Syll(X_1) = 1.$$

And finally in Portuguese, it is defined as,

$$C_{U^1} : 1 \leq \#char(X_1) \leq 3 \bigwedge \#Syll(X_1) = 1.$$

From a CP point of view, C_{U^1} (as described in the model) is a relaxation of the C_{U^1} associated with a given language. Therefore, after the compilation of the RADNER CP model in MDD_{final}, *Constraint First* allows to perform a refinement of the solutions set, thanks to the MDD propagator or intersection as mentioned in Sec. 3. Thus, either the model or the solving or both are parametric.

5 Results

5.1 Experimental Conditions

The approach described in Sec. 3 was implemented in Java 17. The code is available upon request.

Application: The *Constraints First* model is evaluated on a standardized RADNER-like sentence generation task.

Generation: The generation experiments were performed on a machine using an Intel(R) Xeon(R) W-2175 CPU @ 2.50GHz with 256 GB of RAM and running under Ubuntu 18.04.

Corpus Definition: The n-grams are extracted from Wikipedia English articles (multistream1-7)[2]. Then, the generation is performed with 4-grams. A substantially more significant corpus is needed to perform generation in 5-grams. In addition, if an n-gram does not belong to the NLTK English words list, the n-gram is filtered out. This is indeed a coarse lexicon criterion. But it avoids the introduction of bad n-grams (containing numbers, symbols, links). This criterion can be refined to adapt to more specific needs (e.g., language level).

Target Language: English RADNER-like sentences are generated. RADNER sentences in another language can be generated if another language is used as in corpus input.

Evaluation: The sentence curation of the solution set is performed thanks to the PPL assessments of an LLM (GPT2).

5.2 Results Analysis

Table 3 summarizes the benchmark performance on the Wikipedia articles dataset in 4-grams. The input corpus contains roughly 70 million of 4-grams. Only 6961 solutions are generated. For instance, with a corpus of only 3 million 5-grams, a similar approach generates seven thousand MNREAD sentences (see [Bonlarron *et al.*, 2023]). Thus, as expected, due to the various unary constraints and the meter constraints, the RADNER sentences problem is harder.

---

[2]https://dumps.wikimedia.org/enwiki/20240201/

n	nodes	arcs	mem (GB)	time (m)	sols
4	1682	3268	50	23	6961

Table 3: Number of arcs, nodes, solutions, gigabytes (GB), and minutes (m) for computing the MDD of RADNER-like sentences in 4-grams.
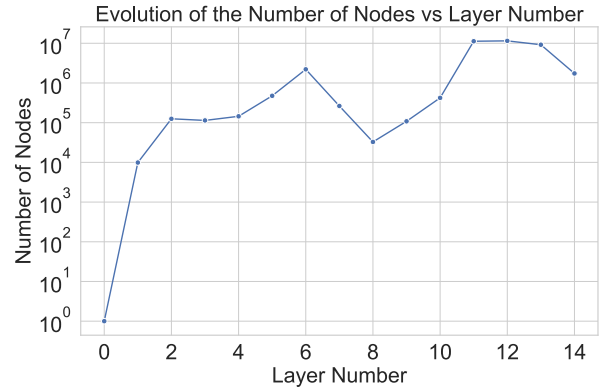

Figure 5: Number of nodes for each layer of the MDD during the solving. The x-axis is the layer number, and the y-axis is the number of nodes in the layer (also called the width of the layer). Y-axis is in logarithmic scale. N.B.: There is at least one ingoing arc for each node in the MDD (except the root node).

Figure 5 further explains duration and memory consumption. The MDD width quickly reaches several dozen thousand nodes during the process. The impact of unary constraints that enforce a comma for the variables X_8 is dramatic (from 263469 nodes at layer 7 to 32738 nodes at layer 8).

In Table 4, some of the best and worst-ranked sentences can be found. There is excellent discrimination between good and poor sentences thanks to PPL sorting for extreme values. Above 50 of PPL valuation sentences become poor semantically and then the PPL ranking is not helping so much. Therefore, going further above is not advisable. Finally, a bigger corpus is needed to ensure more variability in the generated sentences.

6 Discussion

6.1 Creative Settings

Constraints First enables the integration of constraints related to the layout and combination of tokens, pre-tokens (words), or syllables. This makes it possible to directly model certain problems encountered in artistic context. For instance, when generating lyrics, constraints on sounds and rhymes could be defined and included (e.g., rhythmic templates [Roy and Pachet, 2013]). Furthermore, the curation phase can be simplified (resulting in fewer outputs) by integrating LLM scoring earlier in the solving pipeline [Bonlarron and Régin, 2024]. This reduction in the number of outputs encourages the use of interactive mode, providing quick feedback to enhance creativity.

RADNER-like generated sentences	
sentences	PPL
But after only three months of operations, they had been making very little progress	28
With only about three years of operations, the school is divided into seven sections	31
With only about three years of operations, the school is divided into seven boroughs	32
With only about three years of operations, they had been making very little progress	32
With only about three years of occupation, the school is divided into seven sections	32
With only about three years of operations, the state was divided into seven boroughs	32
With only about three years of retirement, he was ranked second among active players	33
With only about three years of occupation, the group was divided into seven sections	33
With only about three years of opposition, the house was divided into seven sections	34
With only about three years of occupation, the group has expanded into other markets	34
His career over three years for conspiracy, and with the only other unlapped runners	749
His career over three years for conspiracy, to which the local area network standard	750
His career over three years for conspiracy, and with the station wagon body variants	769
The sagas describe three of the television, the left the country after Warsaw Chopin	794
The sagas describe three of the television, in the blood plasma within narrow limits	856
The sagas describe three of the television, and that the station wagon body variants	862
His career over three years for conspiracy, and that the only other unlapped runners	862
The sagas describe three of the television, the film was divided into Middle Harbour	862
His career over three years for conspiracy, and that the station wagon body variants	1217

Table 4: Radner-like sentences generated in 4-grams sorted according to PPL

6.2 The Challenge of Intertwining NLP and CP for Constrained Text Generation

Faced with a constrained text generation problem, machine learning was not used due to the lack of available data, and no existing approach that proved appropriate from a combinatorial point of view was adopted. Instead, the problem of constrained text generation has been approached as a discrete combinatorial optimization problem. However, this paradigm choice raised a new challenge: how can certain constraints concerning the text itself, such as the meaning of a sentence, be expressed?

Why is this so difficult? Let us take a step back; in NLP, the philosophy is to "act as a human." Meanwhile, in CP, it is "take a decision as a human". Satisfying rules requires *decide as* while making sense rather requires *act as*. Although *act as* and *decide as* seem close, they are very far apart conceptually and, in fact, when we observe the techniques used to achieve them. This ambivalence of point of view to generate constrained text is revealed in a loose combination of symbolic and statistical reasoning. This naturally pushes us towards a symbolic approach to resolution (*decide as* for the combinatorial aspect) but also forces us to look at more statistical considerations (*act as* to express what we do not know how to model symbolically).

6.3 LLM Have the Attention but Not the Intention

At its core, meaning is rooted in *intention*. When we communicate, we have a specific purpose in mind, a message we want to convey. This intention drives the selection of words and the structure of our utterances. LLMs are attention-based. They process vast amounts of text data but do not have a personal "I" behind these computations.

If there is no *I*, then there is no *intention*. It is not easy to conceive the idea of meaning being a function of the *intention* of a machine. And yet, by taking inspiration from bits of text written by humans (n-grams), do not we recover bits of *intentions*? So we built sentences from bits of intention taken from corpora, and finally selected the best ones based on a democratic vote of general intention (distribution induced by training corpus) calculated by an LLM. The general intention

does not necessarily make sense (because there is no *I* behind a statistical distribution calculated from millions of texts either). However, at least it has the merit of being intelligible. The result is an illusion of meaning in the sentences selected.

6.4 Almost Bona-Fide Sentences for LLM Evaluation?

The proposed method enables the generation of "unreasonably" constrained sentences, rendering them extremely unlikely to occur naturally. This suggests that the model consistently produces original sentences, effectively creating a new type of benchmark for evaluating LLMs.

A crucial question arises: if these sentences are so unlikely, does it imply that LLMs have never encountered them during their training phase? If so, could this approach serve as a new benchmark for assessing LLM performance? This question can be further explored by examining the perplexity (PPL) scoring of such a dataset. What should we make of a perfect sentence that receives a poor PPL score? Is it simply an expected outlier, like any other sample in the testing phase where the LLM assigns a poor PPL score to a perfectly valid sentence? Or could it be a subtle indication that while LLMs excel in well-structured scenarios driven by greedy decoding, they are susceptible to being misled by almost well-formed sentences? (*With..., the school is divided into seven boroughs*)

This observation raises a possibility: LLMs may excel at processing typical language patterns but struggle to effectively handle sentences that lie outside the boundaries of their training data. This suggests that their ability to generalize to unseen data may be limited, particularly when faced with highly constrained structures.

Investigating the PPL scores of sentences generated using this method could provide valuable insights into the strengths and limitations of LLMs in dealing with unexpected linguistic contexts. This approach could be further developed to create more sophisticated benchmarks that challenge LLMs to handle diverse and nuanced aspects of human language.

7 Conclusion

The paper presents *Constraints First* framework as a hybrid modeling approach that loosely combines CP and NLP, to exploit the best of both worlds. Our comprehensive methodology guarantees the satisfaction of an arbitrary list of prerequisites while we use LLM expertise to select the best solutions.

Constraints First effectively solves a real-world problem, The RADNER sentences problem, while raising a fundamental question: How can CP be integrated with NLP? By combining symbolic and statistical approaches, we use CP to formulate and solve the well-defined part of the problem and LLM to deal with those we do not know how to solve and formulate clearly. In the particular case of sentence generation, as it is unclear how to define symbolically a meaning constraint, we rely on statistical computation to ensure the intelligibility of a sentence.

Acknowledgments

in the development of this article.

References

[Alió *et al.*, 2008] Jorge Alió, Wolfgang Radner, Ana Plaza-Puche, Dolores Ortiz, Mª Neipp, María Quiles, and Jesús Rodríguez-Marín. Design of short spanish sentences for measuring reading performance: Radner-vissum test. *J. Cataract Refractive Surg.*, 34:638–42, 05 2008.

[Amilhastre *et al.*, 2014] Jérôme Amilhastre, Hélène Fargier, Alexandre Niveau, and Cédric Pralet. Compiling csps: A complexity map of (non-deterministic) multivalued decision diagrams. *Int. J. Artif. Intell. T.*, 23(04):1460015, 2014.

[Barbieri *et al.*, 2012] Gabriele Barbieri, Francois Pachet, Pierre Roy, and Mirko Degli Esposti. Markov constraints for generating lyrics with style. *Frontiers in Artificial Intelligence and Applications*, 242, 08 2012.

[Bergman *et al.*, 2016] David Bergman, Andre A. Cire, Willem-Jan Hoeve, and John Hooker. *Decision Diagrams for Optimization*. Springer Publishing Company, Incorporated, 1st edition, 2016.

[Bonlarron and Régin, 2024] Alexandre Bonlarron and Jean-Charles Régin. Markov constraint as large language surrogate. In *Proc. of the 33th Int. Joint Conf. on Artif. Intell. , IJCAI-24*, 2024. in press.

[Bonlarron *et al.*, 2023] Alexandre Bonlarron, Aurélie Calabrèse, Pierre Kornprobst, and Jean-Charles Régin. Constraints first: A new mdd-based model to generate sentences under constraints. In Edith Elkind, editor, *Proc. of the 32th Int. Joint Conf. on Artif. Intell. , IJCAI-23*, pages 1893–1901, 2023.

[Cheng and Yap, 2010] Kenil C.K. Cheng and Roland H.C. Yap. An MDD-based generalized arc consistency algorithm for positive and negative table constraints and some global constraints. *Constraints*, 15:265–304, 2010.

[Garbacea and Mei, 2022] Cristina Garbacea and Qiaozhu Mei. Why is constrained neural language generation particularly challenging? *arXiv preprint arXiv:2206.05395*, 2022.

[Gentzel *et al.*, 2022] Rebecca Gentzel, Laurent Michel, and Willem-Jan van Hoeve. Heuristics for MDD Propagation in HADDOCK. In Christine Solnon, editor, *28th Int. Conf. on Principles and Practice of Constraint Programming (CP 2022)*, volume 235 of *Leibniz Int. Proc. in Informatics (LIPIcs)*, pages 24:1–24:17, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Gillard and Schaus, 2022] Xavier Gillard and Pierre Schaus. Large neighborhood search with decision diagrams. In *Int. Joint Conf. on Artif. Intell., IJCAI-22*, 2022.

[Google, 2023] Google. Palm 2 technical report, 2023.

[Hoda *et al.*, 2010] Samid Hoda, Willem-Jan Van Hoeve, and J. N. Hooker. A systematic approach to mdd-based constraint programming. In *Proc. of the 16th Int. Conf. on Principles and Practice of Constraint Programming*, CP'10, page 266–280, Berlin, Heidelberg, 2010. Springer-Verlag.

[Hokamp and Liu, 2017] Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid beam search. In *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1535–1546, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[Jung and Régin, 2022] Victor Jung and Jean-Charles Régin. Efficient operations between MDDs and constraints. In *Proc. of the 19th Int. Conf. on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, 2022.

[Jurafsky and Martin, 2009] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, Upper Saddle River, N.J., 2009.

[Lecoutre, 2011] Christophe Lecoutre. STR2: Optimized simple tabular reduction for table constraints. *Constraints*, 16(4):341–371, oct 2011.

[Legge, 2016] Gordon E Legge. Reading digital with low vision. *Visible language*, 50(2):102, 2016.

[Liu *et al.*, 2021] Yixian Liu, Liwen Zhang, Wenjuan Han, Yue Zhang, and Kewei Tu. Constrained text generation with global guidance - case study on commongen. *CoRR*, abs/2103.07170, 2021.

[Lu *et al.*, 2022] Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. NeuroLogic A*esque decoding: Constrained text generation with lookahead heuristics. In *Proc. of the 2022 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 780–799, Seattle, United States, July 2022. Association for Computational Linguistics.

[Malalel *et al.*, 2023] Steve Malalel, Arnaud Malapert, Marie Pelleau, and Jean-Charles Régin. MDD archive for boosting the pareto constraint. In Roland H. C. Yap, editor, *29th Int. Conf. on Principles and Practice of Constraint Programming, CP 2023, August 27-31, 2023, Toronto, Canada*, volume 280 of *LIPIcs*, pages 24:1–24:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[Mansfield *et al.*, 1993] J. Stephen Mansfield, Sonia J. Ahn, Gordon E. Legge, and Andrew Luebker. A new reading-acuity chart for normal and low vision. *Ophthalmic and Visual Optics/Noninvasive Assessment of the Visual System Technical Digest*, 3:232–235, 1993.

[Mansfield *et al.*, 2019] J. Stephen Mansfield, Nilsu Atilgan, Angela Lewis, and Gordon Legge. Extending the MN-

READ sentence corpus: Computer-generated sentences for measuring visual performance in reading. *Vision research*, 158:11–18, 2019.

[OpenAI, 2023] OpenAI. Gpt-4 technical report, 2023.

[Pachet and Roy, 2011] François Pachet and Pierre Roy. Markov constraints: Steerable generation of markov sequences. *Constraints*, 16(2):148–172, apr 2011.

[Pachet *et al.*, 2011] Francois Pachet, Pierre Roy, and Gabriele Barbieri. Finite-length markov processes with constraints. In *Proc. of the 22th Int. Joint Conf. on Artif. Intell. , IJCAI'11*, pages 635–642. AAAI Press, 2011.

[Papadopoulos *et al.*, 2014] Alexandre Papadopoulos, Pierre Roy, and François Pachet. Avoiding plagiarism in markov sequence generation. In *Proc. of the 28th AAAI Conf. on Artif. Intell.*, AAAI'14, page 2731–2737. AAAI Press, 2014.

[Papadopoulos *et al.*, 2015] Alexandre Papadopoulos, Pierre Roy, Jean-Charles Régin, and François Pachet. Generating all possible palindromes from ngram corpora. In *Proc. of the 24th Int. Conf. on Artif. Intell.*, IJCAI'15, page 2489–2495. AAAI Press, 2015.

[Perez and Régin, 2015] Guillaume Perez and Jean-Charles Régin. Efficient operations on mdds for building constraint programming models. In *Proc. of the 24th Int. Conf. on Artif. Intell.*, pages 374–380, 2015.

[Post and Vilar, 2018] Matt Post and David Vilar. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proc. of the 2018 Conf. of the North American Chapter of the Association for Computational Linguistics*, pages 1314–1324, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[Radner and Diendorfer, 2014] W Radner and G Diendorfer. English sentence optotypes for measuring reading acuity and speed - the english version of the radner reading charts. *Graefe's archive for clinical and experimental ophthalmology = Albrecht von Graefes Archiv fur klinische und experimentelle Ophthalmologie*, 252, 05 2014.

[Radner, 2017] W. Radner. Reading charts in ophthalmology. *Graefe's Archive for Clinical and Experimental Ophthalmology*, 255(8):1465–1482, August 2017.

[Rosa *et al.*, 2016] Andreia Rosa, Cláudia Farinha, Wolfgang Radner, Gabriela Diendorfer, Maria Loureiro, and Joaquim Murta. Development of the portuguese version of a standardized reading test: The radner-coimbra charts. *Arquivos Brasileiros de Oftalmologia*, 79:238–242, 07 2016.

[Roy and Pachet, 2013] Pierre Roy and Francois Pachet. Enforcing meter in finite-length markov sequences. *Proc. of the AAAI Conf. on Artif. Intell.*, 27(1):854–861, Jun. 2013.

[Rudich *et al.*, 2022] Isaac Rudich, Quentin Cappart, and Louis-Martin Rousseau. Peel-and-bound: Generating stronger relaxed bounds with multivalued decision diagrams. In Christine Solnon, editor, *28th Int. Conf. on Principles and Practice of Constraint Programming, CP 2022, July 31 to August 8, 2022, Haifa, Israel*, volume 235 of *LIPIcs*, pages 35:1–35:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[Shannon, 1951] C. E. Shannon. Prediction and entropy of printed english. *The Bell System Technical Journal*, 30(1):50–64, 1951.

[Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[Trick, 2003] Michael Trick. A dynamic programming approach for consistency and propagation for knapsack constraints. *Annals of Operations Research*, 118:73–84, 2003.

[van Hoeve, 2022] Willem-Jan van Hoeve. Graph coloring with decision diagrams. *Mathematical Programming*, 192(1):631–674, 2022.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[Verhaeghe *et al.*, 2018] Hélène Verhaeghe, Christophe Lecoutre, and Pierre Schaus. Compact-mdd: Efficiently filtering (s) MDD constraints with reversible sparse bit-sets. In *27th Int. Joint Conf. on Artif. Intell. IJCAI-18*, 2018.

[Verhaeghe *et al.*, 2019] Hélène Verhaeghe, Christophe Lecoutre, and Pierre Schaus. Extending compact-diagram to basic smart multi-valued variable diagrams. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 16th Int. Conf., CPAIOR 2019, Thessaloniki, Greece, June 4–7, 2019, Proceedings*, pages 581–598. Springer, 2019.

[Wang and Yap, 2022] Ruiwei Wang and Roland H.C. Yap. Encoding multi-valued decision diagram constraints as binary constraint trees. *Proc. of the AAAI Conf. on Artif. Intell.*, 36(4):3850–3858, Jun. 2022.

[Yao *et al.*, 2024] Shunyu Yao, Howard Chen, Austin W. Hanjie, Runzhe Yang, and Karthik R Narasimhan. COLLIE: Systematic construction of constrained text generation tasks. In *12th Int. Conf. Learning Representations*, 2024.