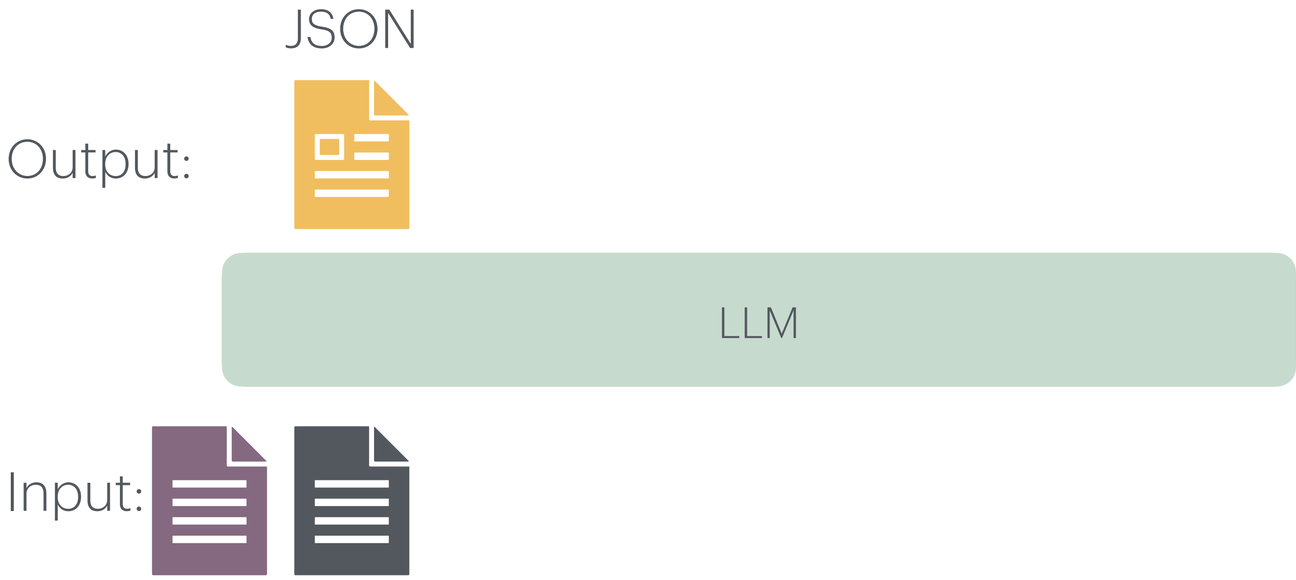
Constrained Decoding

Philipp Krähenbühl, UT Austin

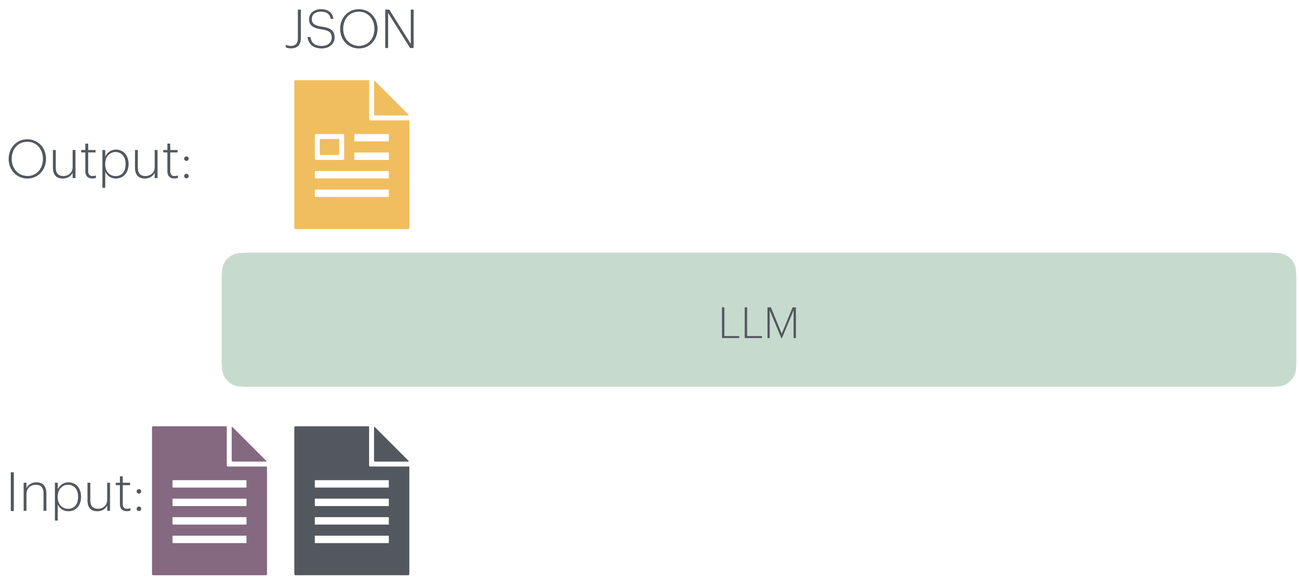
Structured output

- What if we only want to **parse** output of LLM?
  - Option 1.1
    - Write a robust parser (in python)
    - Let LLM know that you failed to parse
    - Hope for the best
  - **Option 1.2: Constrain output**



Constrained Decoding

- Change Sampling of LLM
  - $x_{i+1} \sim P(x_1 \ldots x_i)$
  - Only sample valid next tokens x_{i+1}
- How to define valid?



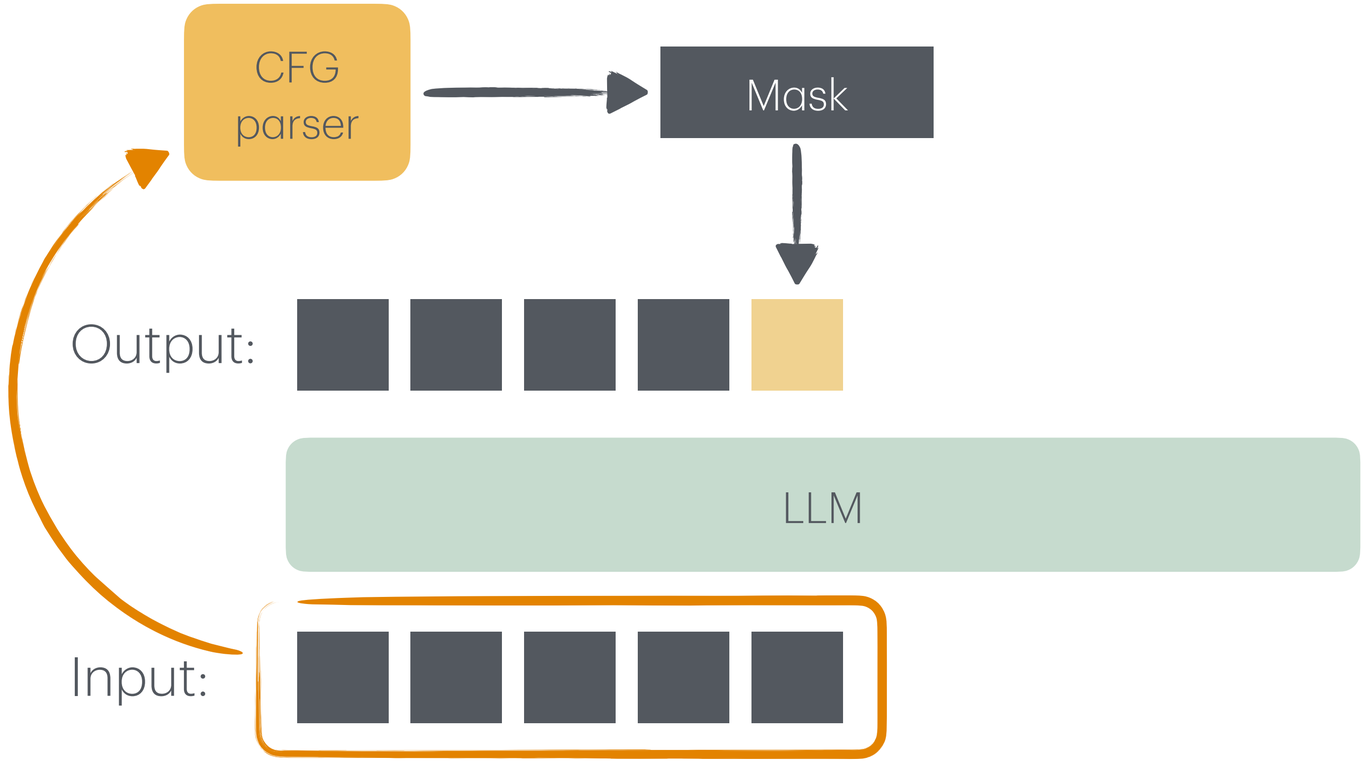
Constrained Decoding

- Change Sampling of LLM
  - $x_{i+1} \sim P(x_1 \ldots x_i)$
  - Only sample valid next tokens x_{i+1}
- Use a Context Free Grammar
  - Regex++

```
root   ::= object
value  ::= object | array | string | number | ("true" | "false"
| "null") ws

object ::=
  "{" ws (
            string ":" ws value
    ("," ws string ":" ws value)*
  )? "}" ws

array  ::=
  "[" ws (
            value
    ("," ws value)*
  )? "]" ws

string ::=
  "\"" (
    [^"\\\x7F\x00-\x1F] |
    "\\" (["\\bfnrt] | "u" [0-9a-fA-F]{4}) # escapes
  )* "\"" ws

number ::= ("-"? ([0-9] | [1-9] [0-9]{0,15})) ("." [0-9]+)?
([eE] [-+]? [0-9] [1-9]{0,15})? ws

# Optional space: by convention, applied in this grammar after
literal chars when allowed
ws ::= | " " | "\n" [ \t]{0,20}
```

https://github.com/ggerganov/llama.cpp

Constrained Decoding



- CFG parser
  - Computes valid completions $v \in V_{i+1}$
- Only sample valid next tokens $P \rightarrow \hat{P}$
  - $\hat{P}(v) = 0 \quad \forall v \notin V_{i+1}$
  - $x_{i+1} \sim \hat{P}(x_1 \ldots x_i)$

Algorithm 1 Constrained Decoding

Input: Checker C, LLM f, Tokenized Prompt x
Output: Completion o adhering to C

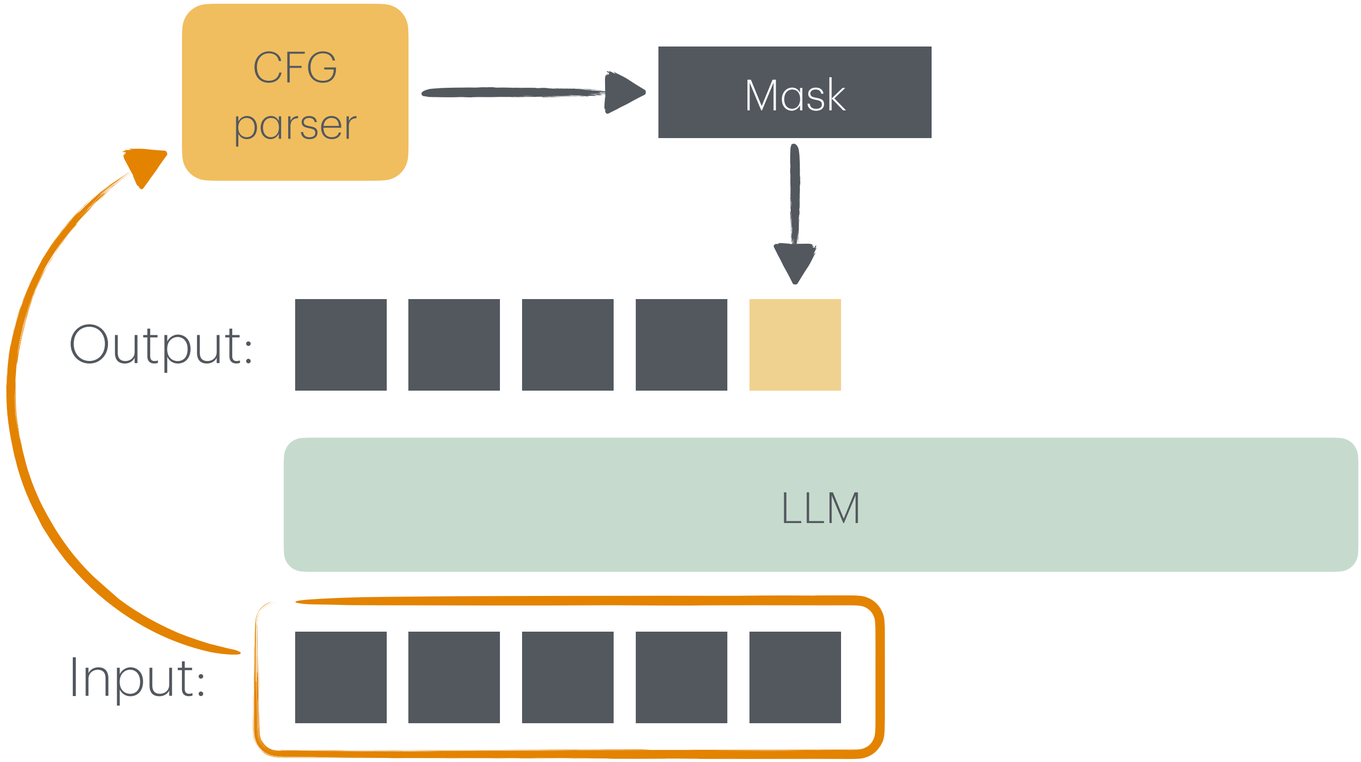
```
o ← []
C.init()
loop
    C.update(o)                    // advance state of C
    m ← C.mask()                   // compute mask
    v ← f(x + o)                   // compute logits
    v' ← m ⊙ v'
    t ← decode(α')                 // e.g., argmax or sample
    if t = EOS then break
    o.append(t)
end loop
return o                           // optionally detokenize
```

Synchromesh: Reliable code generation from pre-trained language models, Poesia etal 2022

Constrained Decoding

Issue

- Constraints: On text / output
- Decoding: on tokens
  - Many tokens streams = one text
  - Model only likes one



Guiding LLMs The Right Way: Fast, Non-Invasive Constrained Generation, Beurer-Kellner etal 2024

Constrained Decoding

Solution

- Lift CFG to token level

Algorithm 2 Construct Terminal Tree

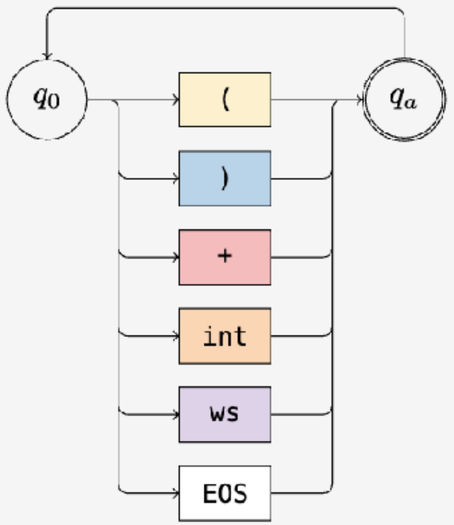
Input: CFG G, Alphabet Σ, Vocabulary $\mathcal{V}$
Output: Scanner S

$\mathcal{T} = \{\}$
for all $q \in S.\texttt{states}()$ **do**
  $\alpha \leftarrow q.\texttt{subterminal}()$ // get current (sub)terminal
  for all $l \in \mathcal{V}$ **do**
    $\{\alpha_1^j, \ldots, \alpha_{m_j}^j\}_j \leftarrow q.\texttt{traverse}(l)$
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{(\alpha_1^j, \ldots, \alpha_{m_j}^j), l\}_j$
  end for
  $T_q \leftarrow \texttt{PrefixTree}(\mathcal{T})$
end for

(a) Example Grammar

```
E = E | E + E | (E) | int
int = ([1-9][0-9]*)|(0+)
```

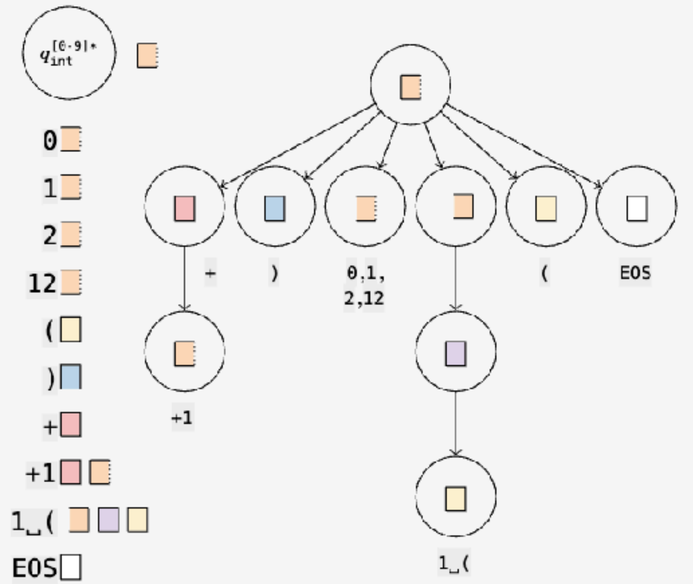
(b) Character Scanner



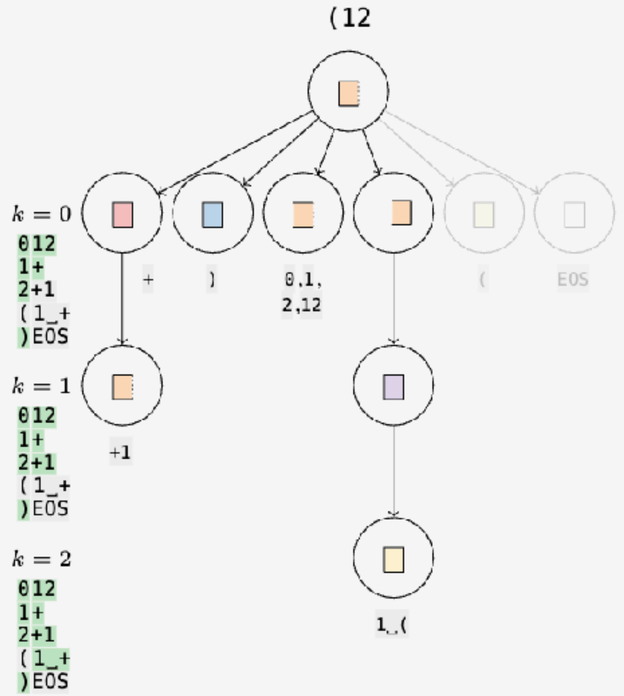
(c) Vocabulary

$$\mathcal{V} = \{0, 1, 2, 12,), (, +, +1\,1\_(, \texttt{EOS}\}$$

(d) Vocabulary-aligned Subterminal Tree (offline, per node)



(e) Parser (online)



Guiding LLMs The Right Way: Fast, Non-Invasive Constrained Generation, Beurer-Kellner etal 2024

Constrained Decoding

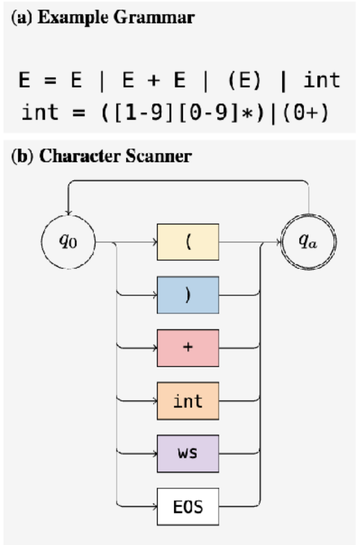
Solution

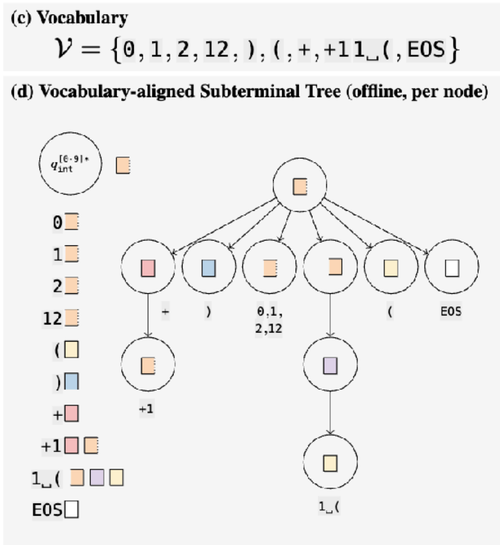
- Very complex to implement

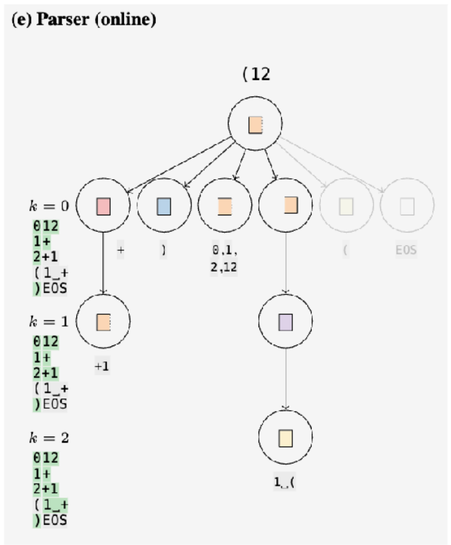
Algorithm 2 Construct Terminal Tree

Input: CFG G, Alphabet Σ, Vocabulary $\mathcal{V}$
Output: Scanner S

$\mathcal{T} = \{\}$
for all $q \in S.\texttt{states}()$ **do**
  $\alpha \leftarrow q.\texttt{subterminal}()$ // get current (sub)terminal
  for all $l \in \mathcal{V}$ **do**
    $\{\alpha_1^j, \ldots, \alpha_{m_j}^j\}_j \leftarrow q.\texttt{traverse}(l)$
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{(\alpha_1^j, \ldots, \alpha_{m_j}^j), l\}_j$
  end for
  $T_q \leftarrow \texttt{PrefixTree}(\mathcal{T})$
end for





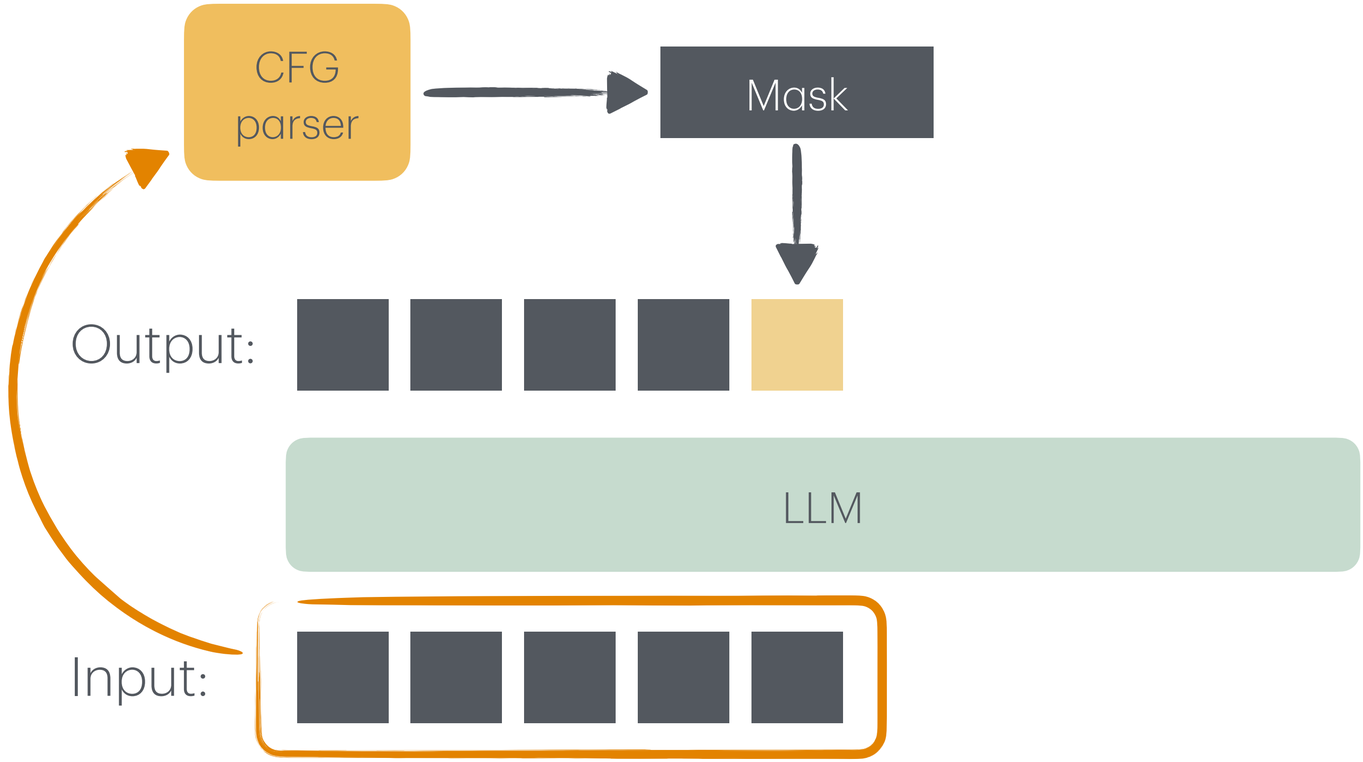


Guiding LLMs The Right Way: Fast, Non-Invasive Constrained Generation, Beurer-Kellner etal 2024

Constrained Decoding

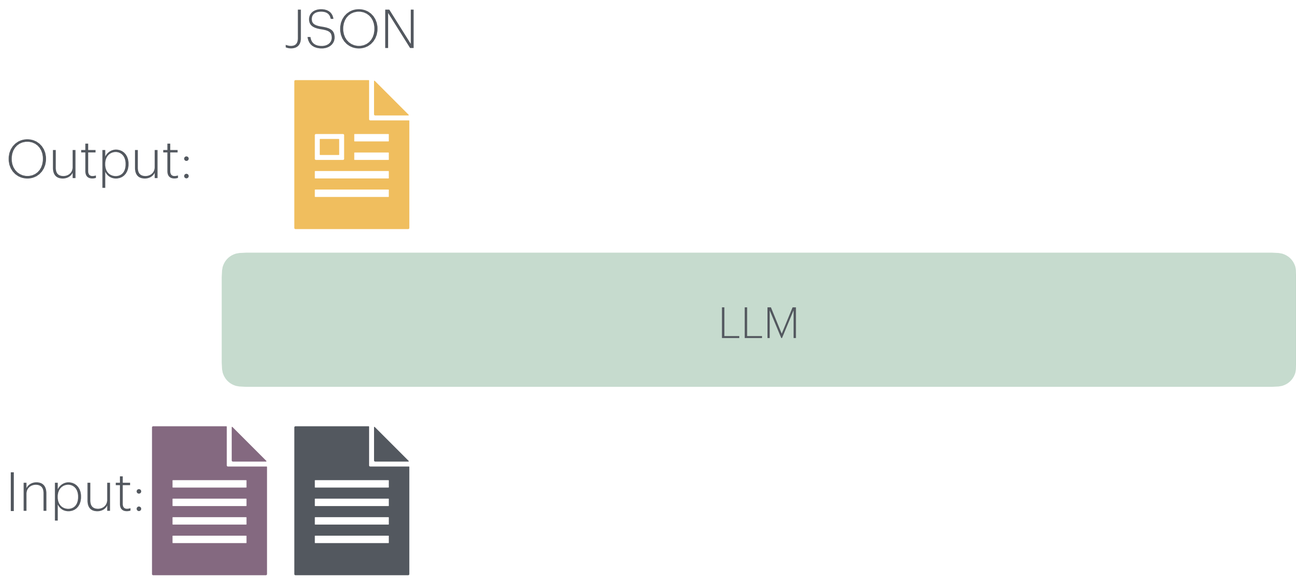
Issue

- Outputs are Biased
- Example CFG
  - S -> aSb | Sc | ε
  - $$P(x_{i+1} \mid x_1 \ldots x_i) = \begin{cases} 0.9 & \text{if } x_{i+1} = a \\ 0 & \text{if } x_{i+1} = b \\ 0.1 & \text{if } x_{i+1} = c \end{cases}$$
  - Sampling will never stop: aaaaaaaa



Structured output

- What if we only want to **parse** output of LLM?
  - Option 1.1
    - Write a robust parser (in python)
    - Let LLM know that you failed to parse
    - Hope for the best
  - Option 1.2: Constrain output



References

- [1] Synchromesh: Reliable code generation from pre-trained language models, Poesia etal 2022

- [2] Guiding LLMs The Right Way: Fast, Non-Invasive Constrained Generation, Beurer-Kellner etal 2024