O padrão recursivo de PARBAL

Nós podemos codificar o padrão recursivo da linguagem PARBAL da seguinte maneira[a]

$$x \quad = \quad (\; x \;)$$

Quer dizer, a ideia é que dos dois lados o x significa *"uma palavra de* PARBAL".

E daí, o que a equação está dizendo é

- *um palavra de* PARBAL *pode ter uma outra palavra de* PARBAL *em seu interior*

Mas, nós também podemos interpretar a equação de maneira ligeiramente diferente.

Quer dizer, nós podemos interpretar a equação como uma *regra de substituição*

- toda vez que a gente encontra o x, nós podemos substituí-lo pelo bloco (x)

$$x \quad \rightarrow \quad (\; x \;)$$

E agora, a regra de substituição se torna um mecanismo para construir palavras de PARBAL.

Quer dizer, começando com a variável x e aplicando a regra uma vez, nós obtemos a palavra

$$(\; x \;)$$

Em seguida, aplicando a regra outra vez, nós obtemos a palavra

$$(\; (\; x \;) \;)$$

e assim por diante.

Mas, a regra $x \rightarrow (\; x \;)$ não é ainda um mecanismo completo gerador de palavras de PARBAL.

Nós precisamos de uma regra para eliminar o x da expressão

$$x \quad \rightarrow \quad \mathtt{e}$$

E nós precisamos de uma regra que nos permitar produzir padrões repetitivos

$$x \quad \rightarrow \quad x \, x$$

Uma *gramática* é basicamente uma coleção de regras de substituição que gera as palavras de uma linguagem

$$\begin{aligned} x \quad &\rightarrow \quad (\; x \;) \\ x \quad &\rightarrow \quad x \, x \\ x \quad &\rightarrow \quad \mathtt{e} \end{aligned}$$

Em geral, as gramáticas podem ter diversas variáveis diferentes, e os livros utilizam a letra S para indicar onde tudo começa.

Abaixo nós temos a gramática da linguagem PARBAL.

1. $S \quad \rightarrow \quad (\; S \;)$
2. $S \quad \rightarrow \quad S \, S$
3. $S \quad \rightarrow \quad \mathtt{e}$

E abaixo nós temos a *derivação* (produção, geração, construção, etc.) de uma palavra de PARBAL utilizando as regras da gramática

$$\mathtt{S} \overset{(2)}{\Longrightarrow} \mathtt{S\,S} \overset{(1)}{\Longrightarrow} \mathtt{(S)\,S} \overset{(3)}{\Longrightarrow} \mathtt{(\,)\,S} \overset{(1)}{\Longrightarrow} \mathtt{(\,)\,(S)} \overset{(1)}{\Longrightarrow} \mathtt{(\,)\,((S))} \overset{(3)}{\Longrightarrow} \mathtt{(\,)\,((\,))}$$

[a] Note como essa equação é semelhante ao código de um procedimento recursivo

```
proc F()
 {   ....
     F();
     ....
 }
```